Using Semantic Mapping for Semantic Based Publish/Subscribe System

A thesis submitted to the

University of Dublin, Trinity College

for the degree of

Doctor in Philosophy

Song Guo

Knowledge and Data Engineering Group,

School of Computer Science and Statistics,

Trinity College,

Dublin

2009

Declaration

I, the undersigned, declare that this work has not been previously submitted as an exercise for a degree at this or any other University, and that, unless otherwise stated, it is entirely my own work.

_____

Song Guo

June 2009

Permission to lend or copy

I, the undersigned, agree that the Trinity College Library may lend or copy this thesis upon request.

_____

Song Guo

June 2009

ACKNOWLEGDEMENTS

I would like to take this opportunity to express my gratitude to those who helped me with various aspects of conducting research and the writing of this thesis. First and formost, I would like to thank my superviors Declan O'Sullivan and David Lewis for their education, support, patience and gentle persistence keep me motivated throughout my Ph.D. study and the many revisions of this thesis. They taught me a lot on how to conduct research, write quality papers, and present academic talks. Especially thanks to John Keeney, for his posdoctorial support and contributions to this work. Many research works I completed were attributed to his insights and innovative ideas.

I would like to especially thank my love Yanhua Lu for her absolute belief, constant encourage, unwavering support and the many sacrifices.

Thank you to my parents, I own thanks for their wonderful love and encouragement. I would also like to thank my brother, since he insisted that I should do so.

Finally thanks to all at KDEG group, for the simulating work atmosphere. It is a great pleasure to record my appreciation of the joy I shared with them and the help I received from them. A warm thanks to Domnic Jones, whom I shared office with for three years and Wei Tai, whom I have shared lunch with for the last two years, for all jokes, laughters, and lively discussions.

ABSTRACT

Routing of information within heterogeneous, distributed network domains (e.g. communication networks, ubiquitous computing environments) is a key challenge that must be tackled for such environments to be successful. Increasingly, it is accepted that such routing systems need to cope with mobile and volatile sources and destinations of heterogeneous information, and Semantic-based Publish/Subscribe (SBPS) systems have been proposed as a means to support this. However such systems typically assume a common semantic model underpinning the routing.

Devices, network resources and applications within network domain environments are likely to be heterogeneous, as they will be produced by different vendors who will each have different priorities and design philosophies. It is unlikely that a single standardized semantic model for communication will gain widespread adoption and the need for exchange of heterogeneous information will persist. The SBPS systems must allow applications/SBPS routers to use different semantic models to communicate with each other without the necessity of custom building gateways. Finally the system must support flexible, adaptive semantic interoperability to cater for the dynamics of such environments.

The KBNMap system proposed in this thesis uses an adaptive, multi-strategy mapping service approach (comprising a set of mapping strategies and adaptive strategy selection mechanism) for the purpose of supporting multiple semantic models within a SBPS, in support of routing heterogeneous information over a network in an appropriate and adaptive manner. The mapping strategies are used for efficiently loading semantic mappings for use with distributed and heterogeneous knowledge-based applications. The probabilistic-based strategy selection mechanism enables the system to dynamically select the appropriate mapping strategy, in order to adapt to the dynamics of the environment where aspects of the semantic models, applications and network environment are constantly changing.

This research advances the state of the art by introducing and evaluating a multi-strategy mapping approach and adaptive mapping selection mechanism within a SBPS system, including identification of the key factors that will influence strategy selection.

TABLE OF CONTENTS

TABLE OF FIGURES

TABLE OF TABLES

ABBREVIATIONS

API	Application Programmers Interface
CBN	Content-Based Networking
CIM	Common Information Model
CQS	Continuous Querying Syndication System
DAML	DARPA Agent Mark-up Language
DL	Description logic
DMTF	Distributed Management Task Force
HTML	Hypertext Mark-up Language
IETF	Internet Engineering Task Force
KBN	Knowledge-based Networking
KBNFix	A special version of KBNMap that is encoded with a fixed strategy
KBNImpl	The Implementation of Knowledge-based Networking
KBNMap	Knowledge-based Networking with Mapping
KBNRan	A special version of KBNMap that randomly selects Mapping Strategy
MIB	Management Information Base
NGN	Next Generation Network
NGS	Next Generation Service
OAEI	Ontology Alignment Evaluation Initiative
OPS	Ontology-based Publish/Subscribe System
OWL	Web Ontology Language
OWL-S	Web Services Ontology Languages
P2P	Peer to Peer
RDF	Resource Description Framework
RDFS	Resource Description Framework Schema
SBPS	Semantic-based Publish/Subscribe System
SOA	Service-oriented Architecture
SPS-SP	Semantic Publish/Subscribe with Super-Peers
S-ToPSS	Semantic Toronto Publish/Subscribe System
TMN	Telecommunication Management Network
UML	Unified Modelling Language
W3C	World Wide Web Consortium
XML	Extensible Mark-up Language

1 INTRODUCTION

In this thesis, the key challenges of distributing semantic information drawn from different semantic models in Semantic-based Publish/Subscribe (SBPS) networks will be identified. A novel adaptive, multi-strategy seman

tic mapping approach will be presented in order to allow SBPS systems to efficiently make use of semantic mappings for addressing these key challenges.

In this chapter, the motivation of this thesis is given in Section 1.1. Then Section 1.2 identifies the research question of the thesis on the basis of the motivation. Section 1.3 outlines the contributions of this thesis to the area of Semantic-based Publish/Subscribe systems. Finally the research approaches adopted and overview of the thesis chapters is outlined in Section 1.4.

1.1 Motivation

Increasingly, coping with extreme heterogeneity and rapid evolution of applications and information, in combination with the need for high throughput, low-latency of messages between large, volatile populations of applications, is a key challenge identified in (Goksel et. al., 1999; Wong et. al., 2005; Chakraborty et. al., 2005; Patrick et. al., 2006; Deng et. al., 2007;) that needs to be addressed in several important networked domains. These include communication network environments (Strassner et. al., 2007); and ubiquitous computing environments (Power 2008).

In the area of communication networks, network resources will always be heterogeneous and thus have different functionalities and programming models (Strassner et. al., 2007). For example, information models used within network devices, network elements and services can be different (e.g. Distributed Management Task Force's (DMTF) - Common Information Model (CIM); TeleManagement Forum (TMF) - Shared Information/Data Model (SID); Internet Engineering Task Force (IETF) - Structure of Management Information (SMI) and so on). When management information is passed between such heterogeneous devices, network elements, services and managed networks, it is desirable to support the publication of information from these information models in their own format rather than forcing them to use a common canonical information model.

In the domain of ubiquitous computing, devices, and the information they produce and share is likely to be described according to heterogeneous information models, as they will be produced by different vendors who will each have different priorities and design philosophies. For

example, when it comes to contextual information, many applications choose to design their own context model, rather than reusing existing ones (Power 2008). These models evolve over time, as applications evolve and new devices and application emerge. Thus, it is unlikely that a single standardised model for context will gain widespread adoption and the need for exchange of heterogeneous information will persist.

The Publish/Subscribe (Pub/Sub) paradigm (Meier et. al., 2005) holds promise for providing loosely coupled communication between various applications and distributing information in such large scaled, distributed networked environments. Applications can subscribe to information that are of interest to them without considering the origin of the information, while applications simply publish information without addressing it to any destination in particular. However, existing Pub/Sub systems require that decoupled publishers and subscribers either share a common type set in order to express their mutual interests such as subject-based Pub/Sub; or match messages to consuming client interests by specifying a filter on the messages's attribute values such as Content-based Pub/Sub. However, there is increasingly a need for Pub/Sub systems to support a richer expressiveness that can cope with frequently changing range of message content and consumer subscriptions. The standardisation of ontology languages by the Semantic Web initiative at the World Wide Web Consortium (W3C) has spurred an increasing number of researchers to use ontology-based semantics as a means to support richer descriptions of information based on a shared encoding of semantic model. (Wang et. al., 2004; Masuoka et. al., 2004; Belecheanu et. al., 2004).

Several researchers have attempted to combine ontology-based semantics with Pub/Sub systems (Halaschek-Wiener et. al., 2007; Skovronski et. al., 2006; Chirita et. al., 2004; Wang et. al., 2004; Burcea et. al., 2003), such that the semantics of the message play an important part in the matching of publications to subscriptions. Such systems are termed Semantic-based Publish/Subscribe (SBPS) systems. The majority of developed SBPSs to date have been focused on applications that assume an homogeneous domain, and so mainly enable the publisher and subscriber applications share a single agreed common semantic model. Thus the common semantic model allows communication and knowledge sharing among distributed applications by providing a semantically rich description and a common understanding of a domain of interests.

However, given the rapid evolution and dynamism of networking, and the inherent heterogeneity of networking environments (e.g. telecommunications management, ubiquitous computing, etc.), it is desirable to allow applications which were designed independently and using different semantic models to communicate semantically enhanced information without the necessity of custom building gateways or adopting a scalable common information model. From

this perspective, it is desirable to have a solution for SBPSs to allow interoperability between semantically related, but heterogeneously instantiated semantic models.

Rather than force multiple parties in different domains to agree on shared semantic model, a more tractable approach is to support interoperability between different semantic models. To solve this kind of heterogeneity, semantic mappings are typically used to express semantic correspondence between diverse semantic models. Thus, in order to support the distribution of knowledge based on different models in a Pub/Sub manner, SBPS system can leverage such use of semantic mappings. The incorporation of semantic mappings within the SBPS routers would mean that applications or brokers that subscribe to information according to one semantic model can expect to receive information published according to a different semantic model, if there exists a mapping between the semantic models. However, the introduction of such a feature inevitably would have an impact on the performance of the system. This is clear from the research described in (Lewis et. al., 2006b), which shows the impact of introducing different semantic models into a SBPS router's knowledge base at loadtime, or indeed runtime. In addition, the dynamic nature and diversity of the networked domains mentioned, require that the handling of multiple semantic models and mappings in the SBPS routers will need to be designed in a flexible fashion, so that the performance of SBPS systems can be tuned depending on the deployment scenario.

1.2 Research Question

The research question answered in this thesis is *how to enable the management of a Semantic-based Publish/Subscribe system to distribute heterogeneous information, in an application and ontology appropriate manner.* Here "heterogeneous" refers to the heterogeneity between the multiple semantic models that applications or routers use to express their information. Rather than focus on any one particular application domain or set of ontologies, it was decided to design a general solution applicable to several domains and types of ontologies. Thus, the phrase "... application and ontology appropriate manner" indicates that the solution developed must be based on a characterisation of both ontology, environment and application features. To aid readability, the thesis however provides some examples applications throughout generally in the pervasive computing/networking domain.

To address the question above, the KBNImpl (Keeney et. al., 2007; Keeney et. al., 2008a; Keeney et. al., 2008b) was chosen as an example SBPS system that could be extended with a novel adaptive, multi-strategy semantic mapping approach designed by the author to cope with semantic heterogeneity. This choice of KBNImpl arose out of the availability of the source code and access to the development team based in TCD. The resulting new system developed is

called **K**nowledge **B**ased **N**etworking with **M**apping (KBNMap). As the research area of semantic mapping is itself quite large, it was decided to focus only on approaches that would use semantic mappings rather than discover or create semantic mappings. In addition only one-to-one mappings, expressed in OWL (OWL 2004a; OWL 2004b) were used. These mappings provide a set of semantic correspondences between entities in different semantic models using the following operators: *semantic equivalence, subclass, superclass, subproperty, superproperty*.

To address the presented research question, the following objectives were derived:

Objective 1:Identify the key features, semantic heterogeneity and criteria to influence how to handle heterogeneity in SBPS systems

The first objective was to conduct a survey on existing approaches that use semantic-web technologies for enhancing Pub/Sub systems, to establish the state of the art in this area. This survey identified the key features identified by others when building Semantic-based Pub/Sub systems. In addition, the author proposes a set of criteria that are needed in the classification for related to support for semantic interoperability. A survey was undertaken to identify the key factors that influence how to address heterogeneous information in SBPSs.

Objective 2:Present a comprehensive framework to handle heterogeneous information in SBPS systems

The second objective was to present a comprehensive framework to deal with heterogeneous information in SBPS systems in an appropriate manner. Taking account of the diversity of the identified influential factors in objective 1, a range of semantic mapping strategies was required, along with a dynamic mechanism to adaptively select one of the mapping strategies. The multiple mapping strategies aim at addressing semantic heterogeneity for SBPS systems in different situations while the mapping strategy selection method enables the dynamic selection of the appropriate mapping strategy to adapt to the changes of ontology, application and environment based characteristics.

Objective 3:Implement, evaluate and verify this presented approach.

The final objective was to implement the comprehensive framework; evaluate the effect of the designed semantic interoperability framework on the performance of SBPS routers; and verify whether the adaptive strategy selection mechanism appropriately manages the semantic interoperability mechanisms in an SBPS system. The presented approach, called KBNMap, was implemented as an extension to an existing SBPS system (KBNImpl). It uses an ontology platform for executing mapping strategies, and a probabilistic modelling mechanism (Bayesian Network) to drive the selection of an appropriate mapping strategy. In addition, a

comprehensive evaluation and verification of the implemented approach was achieved through a series of experiments using KBNMap.

1.3 Thesis Contributions

1. ***Design of a multi-strategy mapping approach for SBPS systems to deal with heterogeneous semantic information***

The first major contribution of this thesis is the design of a multi-strategy mapping framework for any SBPS systems to deal with semantic heterogeneity. The semantic mapping approach uses semantic mappings to resolve heterogeneous information and allow multiple diverse semantic models to coexist in SBPS networks, which is a current gap in the state of the art. This is important, as relying on a single common semantic model shared among applications in a heterogeneous, distributed deployment environment is unrealistic. The proposed multi-strategy mapping approach also enables SBPS systems to cater for different deployments and situations through the provision of multiple mapping strategies. This is also important, as providing a fixed mapping service to handle all situations and to satisfy various requirements is impracticable. The enhanced SBPS network could therefore supports both heterogeneity and dynamism in the environment.

2. ***A comprehensive literature review of existing SBPS systems and a taxonomy of key features for evaluating SBPS systems***

Given the relative immaturity of the area, there is no work conducting a survey of existing Semantic-based Publish/Subscribe systems; and there has been little effort to date in the identification of key features of SBPSs or their classification. The literature review surveys the state of the art SBPS systems, and evaluates, compares them according to the taxonomy identified. The developed taxonomy is structured as a hierarchy of properties that serves as a basis for identifying key features required by a SBPS system, in order to support semantic interoperability in an adaptive manner. The taxonomy can be further divided into two sub taxonomies: the basic taxonomy will lead to classification and comparison of existing systems on the basis of the identified key features. This sub taxonomy also paves a way to evaluate the *expressiveness* and *scalability* of a new designed SBPS. Whilst the advanced taxonomy identified a number of criteria that a SBPS would follow in the support of semantic interoperability and highlights existing systems' ability to adaptively support flexible semantic interoperability based on the criteria identified**.**

3. Identification of key factors that influence provision of a semantic mapping service in a SBPS system

Another contribution is the findings arising from both the literature review and the experimentation that a characterisation of the ontologies, applications, and environment for a given SBPS deployment is very useful in determining which semantic interoperability strategy is appropriate to ensure efficient performance of the SBPS system. To support the dynamics of networks, a SBPS system might have multiple semantic interoperability strategies to address heterogeneity. A critical challenge arises with the appropriate management and runtime configuration of multiple mapping strategy mechanisms, adapting to the semantic complexity of the ontologies, dynamism of applications and constant changes of environmental conditions within the network. For example, the environmental state of a network may be changed (e.g., nodes dynamically join and leave); messages that are semantically ambiguous at one time may be unambiguous if viewed later after the range of possible messages becomes more refined.

4. Provide an adaptive mapping strategies selection approach, based on the identified influential factors above.

Another major contribution is the provision of an adaptive mapping strategy selection approach to support SBPS systems to dynamically and adaptively respond to changes of key factors identified above. This is important, as the dynamic and constantly changing nature of the large-scale networked domains will inevitably lead to unpredictable and unobservable changes of the environmental conditions, which in turn will influence the efficiency of the mapping service. Through a probabilistic-based mechanism, KBNMap has the capability to adapt to the dynamics of environmental conditions to ensure the efficiency of its mapping service. As conditions change, the developed adaptive mechanism can realise automatic selection of an appropriate semantic interoperability service. Through experimentation it has been demonstrated that the adoption of a static or inappropriate strategy for executing mappings in the SBPS can lead to significant performance degradation. This mechanism also allows easier configuration of SBPS semantic interoperability aspects without requiring significant human intervention.

5. Provide a reference for implementation and a benchmark for evaluation.

Finally, a further minor contribution is the implementation of KBNMap infrastructure and the construction of the evaluation experiments. As it is the first effort to implement an adaptive, multi-strategy mapping approach for SBPS systems, it paves a way for others to implement additional semantic interoperability frameworks/approaches for SBPSs and perform comparative evaluations of their systems.

1.4 Technical Overview

This section presents the technical approach taken in the investigation of the thesis and provides an overview of how the thesis report is structured.

1.4.1 Research Approach

In this thesis, the research approach adopted consists of designing, implementing, and evaluating an extended SBPS system that would use semantic mappings for supporting heterogeneous information delivery, while using a Bayesian Network (BN) to support the dynamic configuration of multiple semantic interoperability strategies to adapt to the dynamics of ontology, application and environmental characteristics.

A comprehensive literature review was carried out to identify the key features of Semantic-based Pub/Sub systems and to form the basis of a proposed taxonomy for classifying SBPSs with respect to key features. This characterises the key features for defining *expressiveness* and *scalability* of a SBPS system. For example, the languages used by a SBPS for expressing subscriptions could demonstrate aspect of its *expressiveness* level and the routing scheme used by a SBPS for routing subscriptions and publications could demonstrate its *scalability*. The key criteria to use semantic mappings for flexibly resolving heterogeneous information in an adaptive manner was then explored, which also contributes to an advanced taxonomy for classifying SBPSs with respect to semantic interoperability support. A comparative analysis of the state of the art SBPS systems on the basis of the proposed taxonomies was conducted. The results of the analysis led to the identification of key challenges and requirements that need to be addressed by SBPS systems in the context of efficiently distributing heterogeneous information.

Based on the state of the art analysis and key challenges of SBPSs gained through the analysis, a novel architecture that allows semantic mappings to be used in SBPSs was designed. The KBNMap consists of three flexible mapping strategies and an adaptive strategy selection mechanism, prototyped in the KBNImpl system. The semantic mappings used in KBNMap are expressed as OWL statements; the mapping strategies are written in Java, with the use of the Jena library for ontology processing. The strategy selection process is modelled by using Bayesian Network (BN) model, which allows a KBNMap router to adaptively select mapping strategy according to the changes of identified key factors. On the basis of the criteria identified in the state of the art survey work, a number of ontology, application and environment based key factors having significant influence on strategy selection were identified for implementation in the BN model. For example, the ontology size and expressivity, the number of mappings are

selected as the key factors, while the concept relations and class hierarchy of the ontology are not.

A large number of experiments were designed to evaluate various aspects of the KBNMap system. The ontologies used for the experiments were selected from current popular ontology databases such as Swoogle (Swoogle 2008), OAEI (OAEI 2007). The detailed mappings of the selected ontologies were generated by the author. The results from the experiments were analysed, supporting the claim that the use of semantic mappings can bring important flexibility and performance benefits for SBPS system in the distribution of heterogeneous information. Additionally, a training and validation process was also conducted to validate the BN model developed for driving strategy selection.

1.4.2 Overview of Thesis

Chapter 2 – Background: provides a basic grounding of ontology engineering, especially the importance of semantic mappings to address heterogeneity between different ontologies. An initial literature review of the publish/subscribe communication paradigm and Knowledge-Based Networking is also given.

Chapter 3 – State of the Art: first constructs a basic taxonomy for classifying SBPS systems with respect to key features, with an advanced taxonomy for classifying SBPS systems with respect to semantic interoperability support. A selected set of SBPS systems are described and reviewed according to the taxonomies identified. The chapter then presents the key challenges that need to be addressed for SBPS systems in order to distribute heterogeneous information in an appropriate manner.

Chapter 4 – Adaptive Semantic Mapping Service: first describes the approaches taken to address the two key challenges identified in Chapter 3. Afterwards, the chapter then presents the high-level architecture of using the adaptive semantic mapping service in SBPS systems. The design of the extension of the KBNImpl with semantic interoperability strategies that use semantic mappings presented. A detailed description of how to extend KBNImpl with designed semantic mapping strategies is given. An initial evaluation of proposed semantic mapping strategies is presented in Appendix C.

Chapter 5 – Mapping Strategy Selection: describes the identification of key factors that influence strategy selection, along with the design of the strategy selection mechanism according to the key factors identified.

Chapter 6 – Implementation describes the implementation of the KBNImpl extension (KBNMap) and gives a simple, concrete example of how to use KBNMap system.

Chapter 7 – Evaluation presents how KBNMap was evaluated. The performance of KBNMap was evaluated respectively in twenty one cases that are derived from different combinations of key factors. Each of the tests that were undertaken is described, results are presented and findings discussed. In addition, the correct operation and adaptability of KBNMap to cater for the key factors were verified by using the identified cases above.

Chapter 8 – Conclusion outlines the contribution of this research work and concludes with a short discussion of further work.

2 BACKGROUND

In the previous chapter, the motivation and research question of this thesis are presented. This chapter presents the background knowledge to support the thesis.

2.1 Introduction

This chapter investigates the area of semantic interoperability especially focusing on ontology mapping techniques. This chapter also provides the background information related to the publish/subscribe paradigm and briefly introduces the concept of Knowledge-Based Networking for the purpose of helping readers become familiar with the essential technology, concept, terminologies, and definitions used throughout this thesis.

Section 2.2 first introduces the basic concept of ontology. Section 2.3 provides the definition of semantic interoperability and an overview of the ontology mapping technique. In Section 2.4 a "XG policy distribution service" scenario is also presented in order to illustrate the importance of using ontology mapping to address heterogeneity between different ontologies. Section 2.5 provides the background information for the Publish/Subscribe paradigm. Section 2.6 briefly introduces the concept of Content-based Networking. Section 2.7 briefly introduces XML-based Pub/Sub systems then. Finally an overview of Knowledge-based Networking is given in Section 2.8.

2.2 Introduction to Ontology

As described in Chapter 1, one of the most important characteristics of existing SBPS systems is that they all make use of semantic models to express the information or data derived from different applications. A semantic model is a kind of knowledge model that consists of a network of concepts and the relationships between those concepts (Gruber 1993). Therefore the use of semantic models could ease interoperability between different networking domains and applications not at the level of data exchange but also the level of knowledge sharing. This therefore enables different applications and/or management software components to clearly understand the definitions, information structures, and goals of information provided by other applications. In addition, it also enables the common understanding of what the applications and/or the software components really mean when they provide these definitions, information structures and so on. In other words, semantic models could offer a formal representation of knowledge that is required by applications.

To define semantic models used by SBPS systems, a formal definition of the application level knowledge that describes what data means and where it fits in the SBPS systems is required. Ontologies can model the semantics of entities managed or used by the applications and the relationships existing between them. The following subsection gives an overview of ontology knowledge representation, particularly in the way to define and represent ontology.

2.2.1 Ontologies

It is possible to find in literature several definitions of ontologies. One of the most cited is the one proposed by the author in (Gruber 1993): *an ontology is a formal explicit specification of a shared conceptualisation*. The author in (Fensel 2003) analyses this definition in four aspects: A *conceptualisation* refers to an abstract model of some phenomenon in the world that identifies relevant concepts of the phenomenon. *Explicit* means that types of concepts used and the constraints on their used are explicitly defined. *Formal* means that the ontology should be machine understandable, while *share* refers to the fact that consensual knowledge captured by ontology is not restricted to one individual but accepted among a group (Fensel 2003). In addition, the *sharing* aspects also indicate the reusability of that knowledge which enables semantic interoperability between intelligent agents and applications.

The definition proposed above is general, however ontologies can be defined in specific contexts. For example, taking into account the paradigm of agents, the authors in (Russell et. al., 1995) define *ontology is a formal description of the concepts and relations which can exist in a community of agents.* In addition, the importance of terms of an ontology can be perceived by the next definition that is proposed in (Swartout et. al., 1996): *an ontology is a hierarchically structured set of terms to describe a domain that can be used as a skeletal foundation for a knowledge base*. More definitions of ontologies are described as follows:

- An ontology is a common, shared and formal description of important concepts in an specific domain (Fensel 2000);
- An Ontology is a set of knowledge terms, including the vocabulary, the semantic interconnections, and some simple rules of inference and logic for some participants (Hendler 2001);
- An ontology is a formal explicit representation of concepts in a domain, properties of each concept describes characteristics and attributes of the concepts known as slots and constrains on the slots (Noy et. al., 2001);
- An ontology is a theory which uses a specific vocabulary to describe entities, classes, properties and related functions with certain point of view (Fonseca et. al., 2002);
- An ontology necessarily includes a specification of term uses, (“terminology”) and

agreements to determine the meaning of these terms, along with the relationships between them (Starlab 2003).

From these definitions, some essential aspects of ontologies can be identified:

- Ontologies are used to describe a specific domain.
- The terms and relations are clearly defined in that domain.
- There is a mechanism to organise the terms, for example, a hierarchical structure is used as well as IS-A or HAS-A relationships.
- There is an agreement between users of an ontology in such a way the meaning of the terms is used consistently..

In summary, the ontology can be considered as a formal data model that represents a set of concepts within a domain, individuals and the relations between these concepts. It is used to reason about the concepts in the domain and properties of that domain, which are described formally so that logic reasoners can make implicit information explicit Ontologies are also deterministic in nature.

The basic components of ontology are shown as follows:

- **Classes**: families of objects. They may contain individuals, other classes or the combination of both.
- **Individuals**: the basic components of an ontology. Individuals may refer to concrete objects as well as abstract individuals such as numbers and words.
- **Attributes**: each attribute has at least a name and a value, and is used to describe information of objects such as properties, features, characteristics, or parameters.
- **Relations**: an important usage of attributes is to describe the relations between objects. Normally, relations are the attributes whose attribute values are other objects in the ontology.

Ontologies can be used to support a great variety of tasks in diverse research areas such as knowledge representation, natural language processing, information retrieval, databases, knowledge management, and so on. Given the context of this thesis's focus on networked domains, the application of ontologies in networked domains is mainly presented. The applications in a distributed networked domain can benefit with the incorporation of ontologies. As a concrete example, the authors (Strassner et. al., 2007) employ ontologies to capture concepts in autonomic network management domain in order to provide a shared common understanding of this domain, enabling interoperability and reuse but also machine-readability and reasoning about information through inferencing. Another representative example is the

semantic integration of information sources for large-scale distributed heterogeneous networks (e.g., ubiquitous computing network). Some applications use a domain ontology to integrate information resources and others allow each resources to uses its own ontology. Each application can also have his own ontology according to his interests, languages, or role in a determine domain.

2.2.2 Ontology Language

An ontology language is a formal language used to encode the ontology. They encode the knowledge of a specific domain into the ontology as well as include the reasoning rules that are used to process that knowledge. Over the years, a number of ontology languages have been developed, focusing on different aspects of ontology modelling. For example, traditional ontology language (e.g., Ontolongua, F-logic) are only focusing on modelling ontology in a formal way, while web-based ontology languages (e.g., RDF, OWL) are more concerned with expressing and annotating metadata of information and data published by the web. As most Semantic Based Pub/Sub systems are using web-based ontology languages to express their information, the web-based ontology specification languages are only briefly presented here:

2.2.2.1 RDF & RDF Schema

RDF (RDF 2004) is a graph-based data model that provides foundations for representing and processing metadata. An RDF graph is a set of statements, which describe information of web resources in the form of *subject-predicate-object* expressions, or triples. The *subject* of an RDF statement is a resource, either anonymous, or named by a URI. The *predicate* shows the traits and aspects of resources and expresses a relation between *subject* and *object*. The *object* is a resource or a literal data value.

RDFS (RDF Schema (RDFS 2004)) is an extensible knowledge representation language that builds on the top of RDF and provides mechanisms for describing specific domains. Some of the key primitive ontology constructs provided by RDFS shown are:

- **rdfs**:**Resource** is the class of all resources.
- **rdfs**:**Class** is the class of all classes.
- **rdf**:**Property** is the class of all properties.
- **rdf**:**type** relates resource to its class.
- **rdfs:subClassOf** allows declaration of hierarchy of classes.
- **rdfs:domain** of a **rdf:Property** declares the class of the subject in a triple using this property as predicate.

- **rdfs:range** of a **rdf:Property** declares the class or datatype of the object in a triple using this property as predicate.

2.2.2.2 DAML+OIL

DAML+OIL (Horrocks et. al., 2002) is a product of efforts in merging two languages – DAML (DARPA Agent Modeling Languages) and OIL (Ontology Inference Layer) aiming at providing a general-purpose markup language for the semantic web (Fensel et. al, 2000). DAML+OIL builds on RDF(S) for providing richer modelling primitives than RDF(S) and assigning a specific meaning to certain RDF triples. The model-theoretic semantics (DAML Model 2001) specifies exactly which triples are assigned a specific meaning, and what this meaning is. It provides an additional way to constrain the allowed values of properties, and what properties a class may have. For example, there are two kinds of restrictions provided for properties. The first kind is *ObjectRestriction* properties that relate objects to other objects. The second kind is *DatatypeRestriction* properties that relate objects to datatype values. Both kinds of restrictions are created using the same syntax, with the only difference being whether a class element or a datatype reference is used.

2.2.2.3 OWL

RDF and RDFS provide a limited representation of ontological knowledge. Their primary concern is the organisation of concepts in typed hierarchies. However, some sophisticated ontology constructs are missing. For example, using RDF and RDFS it is impossible to express that some classes are disjoint; or to create class definitions that are union, intersections, or complements of other classes. It is also not possible to express cardinality expressions. These are realised by the Web Ontology Language: OWL (OWL 2004a, OWL 2004b), which is a family of knowledge representation languages that emphasises support for richer logical inference. OWL is an extension of RDF and RDFS with a larger and more semantic vocabulary. Three variants of OWL trade off computational complexity and the expressiveness of ontology constructs:

- **OWL-Lite:** is the simple variant and basically consists of class, property, subclass relation, and restrictions. OWL-Lite does not use the entire OWL vocabulary and some OWL terms are used under restriction.
- **OWL-DL:** is grounded on Description Logics, and focuses on common formal semantics and inference decidability. Description Logics offer additional ontology constructs such as (conjunction, disjunction, and negation) besides class and relation, and have two more inference mechanism: subsumption and consistency. Horrocks and Sattler (OWL 2004a) argued that basic inference in most variations of Description

Logic is decidable with complexity between polynomial and exponential time. The strong Set Theory background makes Description Logic suitable for capturing knowledge about a domain in which instances can be grouped into classes and relationships among classes are binary. OWL-DL uses all OWL ontology constructs with restrictions.

- **OWL-Full:** is the most expressive version of OWL but it does not guarantee decidability. The biggest difference between OWL-DL and OWL-Full is that class space and instance space are disjoint in OWL-DL but not in OWL-Full. That is, a class can be interpreted simultaneously as a set of individuals and as an individual belonging to another class in OWL-Full. The entire OWL vocabulary can be used without any restrictions in OWL-Full.

2.3 Semantic Interoperability between Ontologies

Interoperability is the ability of two or more computer systems to exchange information and have the meaning of that information automatically interpreted by the receiving system accurately enough to produce useful results, as defined by the end users of both systems (Heflin et. al., 2000). In order to achieve interoperability, the applications should have a common understanding of the meaning of the exchanged information. Ontologies are generally considered useful for providing interoperability (Gruber 1993), since they provide a shared semantically rich description and common understanding of a domain of interests that can be exchanged between people and application systems. For example, a core part of semantic web ontology is to enable interoperability between applications (Berners-Lee 2001), since they allow applications to agree on the terms that they use when communicating. It facilitates communication by providing precise notions that can be used to compose messages (queries, statements) about the domain. For the receiving party, the ontology helps to understand messages by providing the correct interpretation context. The advantage of using ontology in networking domain has been demonstrated through the conversion of various network management information models (e.g., CIM and SMI) into OWL representations (Vergara et. al., 2002). Such features are also desirable in large scale web commerce environments (Omelayenko 2002; Fensel et. at., 2003) or open networked systems (Castano et. al., 2005).

2.3.1 Ontology Heterogeneity

As described above, interoperability can be achieved by using an ontology. However, given the rapid evolution and dynamism of networking, there is increasingly a desire to allow people, organisations, devices and applications to have freedom of choice in terms of how they structure

their semantics and the representation they use. In other words, applications may wish to interact with each other on the basis of their own independent ontologies. It seems clear that ontologies face the same or even harder problems with respect to heterogeneity as any other piece of information (Valente et. al., 1999). For example, given the decentralised nature of the development of the networked domains described in Chapter 1, the number of ontologies will become huge as the number of applications grows. Many of these ontologies will describe similar domains, but using different terminologies, structures, relationships and others will have overlapping domains. Therefore, it is necessary to find a solution to reconcile these different ontologies in order to have a shared aggregate composed model that is understood by the heterogeneous applications. The multiplicity and diversity of independent ontologies is termed as *ontology heterogeneity* and can be characterised by different kinds of mismatches between ontologies (Hakimpour et. al., 2001):

2.3.1.1 Ontology Mismatches

The authors in (Chalupsky 2000; Klein 2001) categorise the different types of mismatches between ontologies into *language level* and *ontology level*. The former refers to syntax and expressivity while the latter refers to the semantic layer. As the main concern in SBPS is more focusing on mismatches between ontologies at semantic level rather than language level, the general ontology level is only described here: the mismatches happen when two or more ontologies that describe partly overlapping domains are combined. The mismatches are further divided into *conceptualisation mismatch* and *explication mismatch*. The *conceptualisation mismatch* is a difference in the way a domain is interpreted, which leads to different concepts or relations between those concepts; whereas the explication mismatch is a difference in the way the conceptualisation is specified. The *conceptualisation mismatches* are further categorised into:

- *concept scope* that describe the scenario that two classes represents the same concept, however, the instances belongs to each class are different, although they may intersect.
- *model coverage* refers to a mismatch in the part of the domain that is covered by the ontology, or the level of detail to which that domain is modelled. *Explication mismatches* are further characterised as terminological, modelling style and encoding.

2.3.2 Overview of Ontology Mapping

In this thesis, having chosen ontologies as the mechanism for expressing semantic models, it was natural to investigate the capability of ontologies for expressing mappings between models in order to achieve semantic interoperability. The ontology mappings provide a set of semantic correspondences between the entities in different ontologies. These mappings allow terms in one ontology correctly understood in the context of another ontology. As an example of using

mappings in SBPS systems, the administrators of SBPS systems can use these mappings to integrate applications' different semantic models into the single shared common semantic model thereby building an aggregate model between different semantic models. These mappings can be specified by an administrator who has no access to the application or SBPS system source code, only to their semantic models. These mappings are therefore the functional unit of semantic interoperability- they provide the bridge between heterogeneous applications with heterogeneous semantic models. As application domains are updated, and provide updated ontologies to express their semantic models, these mappings can also be updated accordingly without requiring source code modification to either the SBPS systems or to other applications.

A brief overview of ontology mapping technique is given in this section. However, in this thesis the use of ontology mappings by Semantic-based Pub/Sub systems is the main focus, rather than how to discover, characterise, create mappings between different ontologies. While the type of mappings the SBPSs would use is presented in Chapter 3, this section introduces some common mapping representations.

2.3.2.1 Ontology Mapping

Ontology Mapping is the common technique being adopted to achieve semantic interoperability between heterogeneous ontologies (Noy et. al., 2000; Ouksel et. al., 1999; Orgun et. al., 2006). A general definition of *Ontology Mapping* in (Noy et. al., 2000) is used in this thesis: *to establish correspondences among the source ontologies, and to determine the set of overlapping concepts, (concepts that are similar in meaning but have different names or structure), and concepts that are unique to each of the sources*.

The author in (Su 2004) summarises that "mapping" activity can be used for a range of purposes, including merging, mapping, integrating, aligning, and translation of ontologies:

- **Merging, integrating:** provide a single coherent newly ontology that is created from two or more existing ontologies with overlapping concepts.
- **Aligning:** bring two or more ontologies into mutual agreement, making them consistent and coherent with one and another.
- **Mapping:** make explicit the correspondences (i.e. through creation of a separate ontology) referring to but without changing the original ontologies.
- **Translating:** changing the representation formalism of source ontology to that of target ontology while preserving semantics. Normally finding mappings between two ontologies is considered as the valuable pre-processing step for translation (Dou et. al., 2003).

- **Transforming:** changing the semantics of an ontology slightly (possibly also changing the representation) to make it suitable for purpose other than the original one.
- **Combining:** using two or more different ontologies for a task in which their mutual relations are relevant.

2.3.2.2 Ontology Matching Techniques

As stated in (Castano et. al., 2005), ontology matching techniques are essential to enable knowledge discovery and sharing in open networked systems, in order to determine ontology mappings between semantically related concepts, relations, individuals of different ontologies. The relationship between ontology matches and mappings is that: *ontology matches* (Euzenat et. al., 2007) only provide a set of correspondences between semantically related entities of different ontologies while *ontology mappings* are established after an analysis of the similarity of the entities in the ontology matches. For example, given two different things an "animal having four legs" and a "dog with 4 legs" respectively, it can be said that the concept "four leg" is simply matched to the concept "4 legs" as the two concepts have the same name. However, if taking into account the whole context of concepts, the "four leg" of animal is not mapped to "4 legs" of chair, as "dog" is a kind of animal, while "chair" is a kind of furniture.

In order to generate mappings, various matching techniques can be applied to the ontologies. Several types of matching techniques have been classified (Giunchiglia et. al., 2003; Giunchiglia et. al., 2004). String, Language, Linguistic, and Constraint-based matching techniques work at the *element level* of an ontology and are generally classed as "lexical" matchers, where match considers only the atomic granularity elements of the schema such as attributes in xml schema or columns in relational schema.Reuse, Graph, and Taxonomy –based matchers work at the *structure level* and are generally classed as "structural" matchers, where match refers to matching combinations of elements that appear together in a structure. Finally, Model-based matchers use semantic interpretation (i.e. Description Logics) to find matches and are thus classed as "semantic" matchers, where match refers to matching combinations of elements semantically however, as yet, the fully automatic generation of mappings from different ontology information is generally considered impractical (Noy 2004). This is because there is a degree of uncertainty in any automatic approach to matching two ontologies, with this uncertainty caused by the different syntactic representation of the ontologies, the combination of the similarity measures produced by different matchers, and the heuristic approaches inherent in some matchers (Cross 2003). For now, semi-automatic techniques for creating mappings from ontology matching information will continue to dominate (Uschold et. al., 2004).

2.3.2.3 Mapping Representation

Prior to representing mappings, the mapping correspondences between different ontology elements with similar meanings must be discovered and identified. Mappings are discovered in two ways: 1) detecting mismatches between ontologies by applying a set of matching rules; 2) evaluating interesting similarity measures that compare a set of possible correspondence and help to choose valid correspondence from them. A detailed description about mapping discovery can be found in (Doan et. al., 2003; Ehrig et. al., 2004; Kalfoglou et. al., 2002; Noy et. al., 2003).

Mapping representation reflects the sophisticated structure of ontology mappings and the relations among different concepts, therefore this representation needs to have well-defined semantic to enable reasoning and comparison between mappings (Madhavan et. al., 2002). According to different requirements and specifications, the mappings identified or derived from mapping discovery process can be represented in various ways. However, the way of representing mapping should be on the basis of three criteria (Noy 2004; Su 2004):

- **Clear semantics**: the meaning of mappings should be formally defined, since the semantics provide a basis for reasoning about mappings (e.g., deterring if two mappings are equivalent or if a certain mapping formula is entailed by a mapping).
- **Accommodate incompleteness**: the loss of information when two ontologies cover different domains could lead to incompleteness. However, even if the mapping is incomplete, it still may suffice for a particular task.
- **Heterogeneity allowance**: mapping between ontologies may involve multiple representation languages. Therefore, a mapping language needs to be able to represent mappings between models in different languages. An alternative way would be to first translate all the ontologies into a common representation languages, and then specify the mappings as formulas in this language.

The representation approaches are generally classified into two groups: 1 to 1 instance mapping representation, and complex mapping representation (Noy 2004).

- **1 to 1 instance mapping representation:** the corresponding concepts inside two ontologies are directly linked with each other, and no semantic relations are defined for the link between two mapped concepts (Dou et. al., 2003). According to this approach concentrating on the instance level of mapping between ontologies, it can be used for fast ontology translations, especially for transformation between two similar ontologies. The main disadvantage of this approach is that it is not efficient for ontologies with different information granularities and not feasible for complex

ontology mappings, especially for one to many concepts mappings.

- **Complex mapping representation:** it is using ontology to represent mappings. In this approach, mappings between elements are usually expressed as a pair of related entities in some mapping format that is normally output as a separate ontology. This approach provides more accurate and effective representation solutions for complex ontology mappings and this attributes to the expressive power of ontology itself. For example, mapping ontology provided by (Crubezy et. al., 2003) defines the structure of specific mappings and the transformation function to transfer instances from one to another. This mapping ontology can then be used by systems to perform the transformation between two different ontologies. Additionally, comparing to 1 to 1 instance mapping approach, such a mapping ontology usually provides more complex mapping relations (i.e., more specific, inverse, disjoint etc.) rather than simple equivalence relation encoded between elements. For example, class a in source ontology is more specific than class b in target ontology. Then the mapping can be used by applications to translate data from the source ontology to target ontology. The mapping ontology could also provide many-to-one aggregation relations between concepts in the source and target ontologies, as well as one-to-many concept-decomposition relations. It also allows specification of recursive mappings, complex mappings between that collect information from several related concepts, and other mechanisms.

Several software systems have been developed to analyse two ontologies and identify potential correspondences between them. This includes systems such as GLUE (Doan et. al., 2002), ONION (ONtology compositION system) (Mitra et. al., 2002) , OKMS (Ontology-based knowledge management system) (Maedche et. al., 2003) and OISIN (O'Sullivan et. al., 2004). Other systems are surveyed by authors in (Choi et. al., 2006). These tools can be used to greatly speed up the process of integration, and also to identify potential mismatches where a more complex mapping may be required to bridge the heterogeneity between the two domains.

Many of these systems produce mappings in a proprietary format. For example, the OntoMerge system (Dou 2004) uses bridging axioms written in first order logic language to express the translation rules between the concepts in the ontologies. Another example is the MAFRA system (Maedche et. al., 2002), which uses Semantic Bridges based on an ontology called the Semantic Bridging Ontology (SBO).

There is still no common way to specify the mappings using one particular language. So this thesis takes approach of expressing mappings used in SBPS systems using the semantic correspondences which are provided by OWL.

OWL's standardisation and direct support for semantic relations (i.e., *equivalence*, *subclassof etc*) made it the ideal choice for use in our designed KBNMap. If KBNMap were to support more complex mapping types, another mapping language might be chosen. However, the choice of mapping languages must be considered carefully in order to maximise interoperability and mapping reusability. The semantic relations provided by OWL were sufficient to demonstrate the operation of model mapping for the purposes of this work, since KBNMap is the extension of KBNImpl which makes use of OWL ontologies to express its semantic models, subscriptions and event models. Mapping specification tools such as those described above can be used to partially automate the process of mapping between two ontologies. Typically they present a side-by-side hierarchical structure of concepts, and use a variety of techniques such as lexical analysis of concept names and structural analysis of the ontology to identify candidate matches between concepts. These candidates are presented to the user for verification, after which the user can choose to add additional matches to complete the mapping.

2.4 "XG Policy Distribution" Scenario

As an example scenario to illustrate the importance of using ontology mappings to address ontological heterogeneity problem derived from different ontologies, the case of exchanging policies between different service providers for the management of dynamic spectrum access (DSA) to the radio frequency spectrum in a large decentralised cognitive radio environment was chosen. DSA (Akyidiz et. al., 2006) is an emerging paradigm to exploit existing wireless spectrum opportunistically in order to enable the next generation devices to dynamically use and release spectrum wherever and whenever they are available rather than fixed spectrum assignment policy. This is a rich scenario as the policy approach being advocated in DARPA XG (xg-policy 2004) is ontology-based, while a SBPS system could be applied in this scenario for distributing contextual information between service providers that require this contextual information to evaluate policy rules. But it is unlikely that all service providers will exchange contextual information according to one single contextual model. For example, DARPA XG developed a geographical region ontology associated with the main policy ontology for providing location contextual information. Service Providers may be arbitrarily located in different regional areas, so they need to subscribe their interests for policies relating to specific geographical locations. Thus in order to support policy exchange between the service providers, a policy delivery service that can support heterogeneous ontologies could be a good candidate solution The XG policy language is developed by DARPA XG group for dynamic spectrum access management (xg-policy 2004).

In order to provide examples of the kind of mappings that would be possible in this scenario, two sample geographical region ontologies are used. Figure 2-1 describes a hierarchical structure of mapping relations between classes and properties from the two OWL ontologies. xgpl-regn1 and xgpl-regn2 are region description ontologies made by author for XG policies delivery between communication service providers (CSP). To create OWL mappings between these two ontologies, firstly, there is a need to reference them from the independent mapping ontology so that the rest of the ontology description will be able to refer to the existing elements that are previously defined in an involved ontology. Second, mappings between elements of the involved ontologies were established. The core OWL mapping constructs are: *owl:equivalentClass, owl:equivalentProperty, owl:sameAs, owl:subClass* and *owl:subProperty* because they explicitly relate classes to classes, properties to properties or individuals to individuals (OWL 2006). Therefore, these mapping constructs were only presented in this example. Figure 2-1 shows a number of possible mappings. One class of an ontology may be considered as a *subclass* of another class of another ontology (Village is *subclass* of RuralUnit). Two relations (*subsumption* and *equivalence*) between properties from the involved ontologies can be determined by comparing their members (ishouseof is *subProperty* of isbuildingof in Figure 2-1).

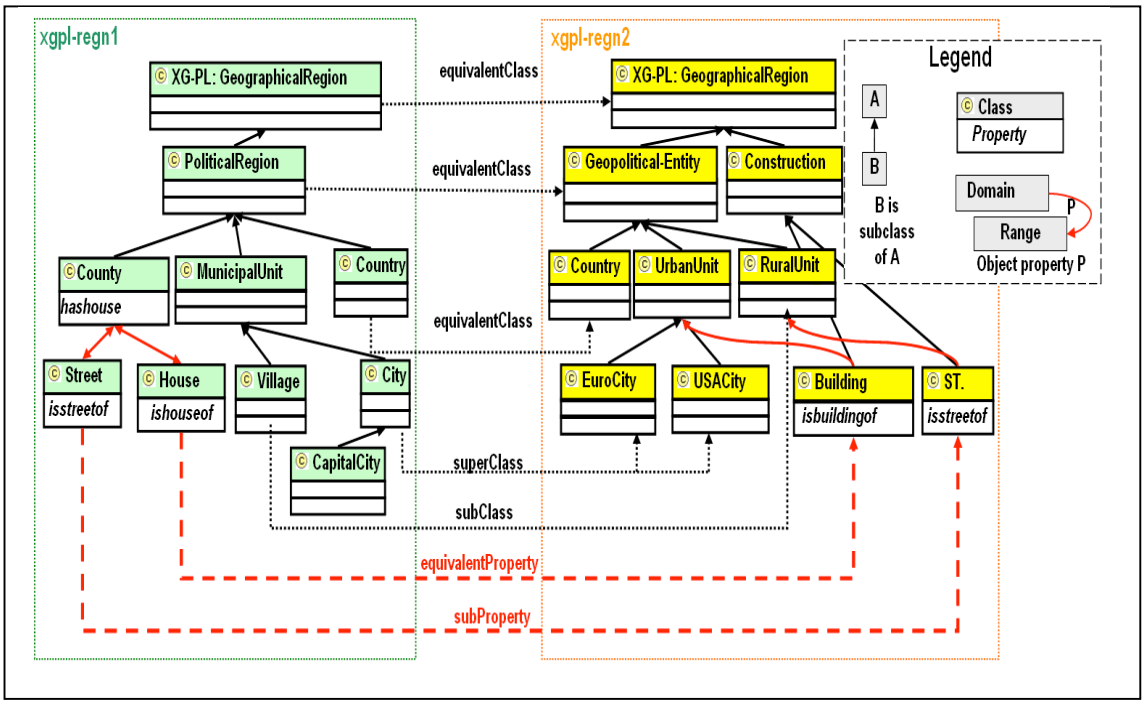


Figure 2-1: Global hierarchy of region mapping ontology

Now, let us assume that a SBPS system is employed to deliver policies between policy providers, and xgpl-regn1 is the main routing ontology distributed among the SBPS routers. If there is a service provider interested in polices within city range, it subscribes to a query expressed by concept city to its closest router. If this router receives a notification that has the

same query name but the concept is EuroCity (EuroCity is not referred in xgpl-regn1 but in xgpl-regn2). This SBPS router needs to explore the mappings to find mapping relations containing town and city to resolve this unknown concept. In this case, the region mapping ontology is explored, where the concept city is identified as *superclass* of the concept EuroCity.

2.5 Publish/Subscribe Middleware

Publish-subscribe (Pub/Sub) (Eugster et. al., 2003; Meier et. al., 2005; Mühl et. al., 2006) is a loosely decoupled communication paradigm for large-scale distributed environments, as it is well suited for integrating software components and applications into complex systems by means of exchanging events, without requiring priori knowledge about the consumers of events it can evolve and scale easily. A Pub/Sub system consists of the following constituents (see Figure 2-2): events and publications as means of communications, publishers and subscribers are used to describe interacting clients, subscriptions expressing a subscriber' interest in certain notification, and the event notification service responsible for delivering publications between subscribers and publishers.

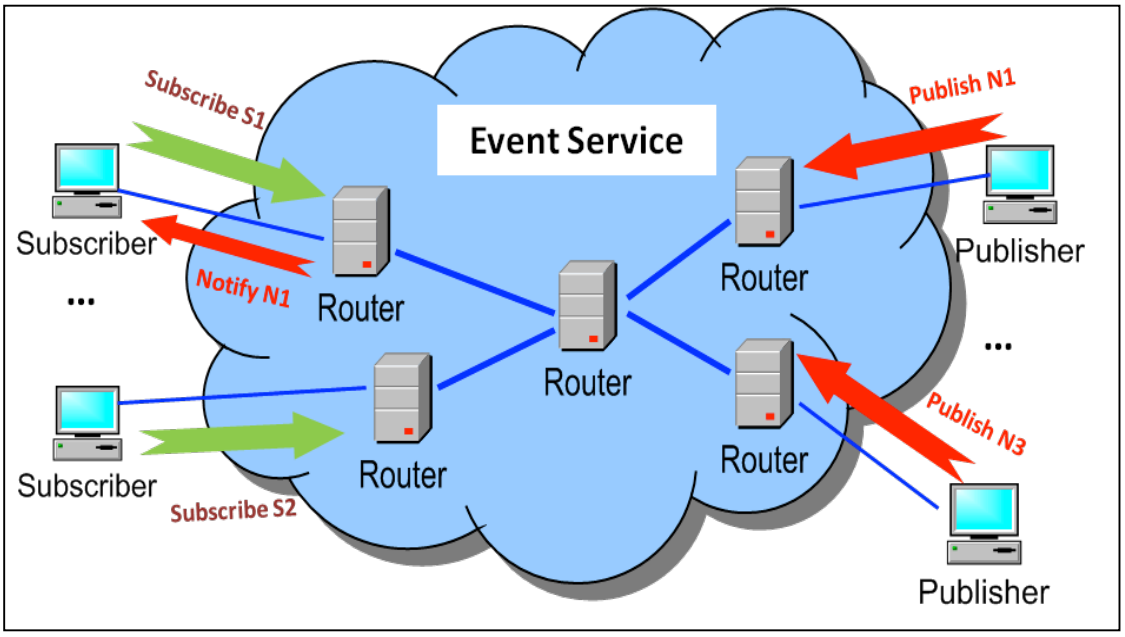


Figure 2-2: Publish/Subscribe middleware

In an *event notification service* system publisher and subscriber clients asynchronously exchange publications, and collaborating distributed routers provides a notification service between clients. **Subscribers** can subscribe to information that are of interest to them without considering the origin of the information, while **publishers** simply publish information without addressing it to any destination in particular. The **router***s* are responsible for matching

publications to subscriptions that express interests to the publications and delivering the selected publications to those subscribers who forwarded those subscriptions.

2.5.1 Traditional Pub/Sub Systems

The differences between Pub/Sub systems typically reflect differences in functionalities with respect to expressiveness of subscription scheme that they can support, as well as the efficiency of underlying dissemination and maintenance schemes. Therefore, existing Pub/Sub systems are characterised according to their subscription scheme reflecting notification selection. There are three common Pub/Sub systems:

- **Channel-based:** it is the first generation of publish/subscribe system. In channel-based Pub/Sub systems (Harrison et. al., 1997; OMG, 2000; Sun, 1998), the notification is published with respect to a specific channel that is specified by the producer. Each notification that is published to a channel is delivered to the subscriber who has subscribed to this channel

- **Subject-based:** in Subject-based Pub/Sub systems (Pfluegl et. al., 1993; TIBCO 1996), publishers publish publications with respect to a certain subject that is usually specified as a dot separated string (i.e., market.Quotes.NASDAQ). Subjects are arranged in a subject tree that must be predefined and shared between all applications, and clients can either subscribe to a subject (i.e., market.Quotes.NASDAQ.fooinc) or to a subject and all of its subordinate subjects (i.e., market.Quotes.NASDAQ.*).

- **Type–based:** in Type-based Pub/Sub systems (Bates et. al., 1998; Eugster et. al., 2001), they use path expressions and subtype inclusion tests to choose publications. With multiple inheritances, the subject tree is extended to type lattices that allow for rooted paths to the same node.

2.5.2 Advantages of Pub/Sub Systems

The main characteristic of a Pub/Sub system is that publishers and subscribers are fully decoupled in time, space and flow (Eugster et. al., 2003). This means: 1) the clients do not need to know each other, it allows subscribers simply specify the notification they are interested in; 2) the communication between clients is asynchronous, thereby removing disadvantages and inflexibility of synchronous communication; 3) publishers and subscribers do not need to be available at the same time. The main advantages of Pub/Sub systems are shown as follows:

- **Scalability:** Pub/Sub provides the opportunity for better scalability than traditional client-server, through parallel operation, message caching, tree-based, semantic clustering based or network-based routing and so on. For example, Siena (Carzaniga et.

al., 2001) makes uses of the combination of efficient filtering-based routing schemes and scalable topology algorithm to offer a wide-area event notification service.

- **Ease of use:** As a Pub/Sub system is a middleware paradigm, it can be easily applied to various domains. For example, Pub/Sub system is well-suited for information dissemination applications like news delivery, stock quoting, air traffic control (Liebig et. al., 1999). The use of Pub/Sub techniques has also been adopted in the area of mobile agents (Skarmeas et. al., 1999), work flow systems (Cugola et. al., 2001), ubiquitous computing (Langheinrich et. al., 2000), peer-to-peer systems (Heimbigner 2001), process control systems (Kaiser et., al., 1999). Furthermore, the use of Pub/Sub was proposed for loose coupling of components (Bates et. al, 1998) of several independent applications.
- **Ease of implementation:** A major advantage is in the simplicity and flexibility of a decentralised implementation is that it enables the system to support a large number of clients and a large amount of data transfers. For example, the subject-based Pub/Sub can take advantage of group mechanisms such as IP multicast for implementing, the predefined subjects can be considered as multicast groups.
- **Small footprint on clients/routers:** the publications are only forwarded to the subscribers that express interests. The notification that does not match any subscription is not sent to any client, saving network resources.

2.5.3 Disadvantages of Traditional Pub/Sub Systems

Traditional Pub/Sub systems require that decoupled publishers and subscribers share a common type-set, or a shared knowledge of channel identifiers, in order to express their mutual interests. This places severe restrictions on the heterogeneity and dynamism of the information elements that can be exchanged. Therefore, traditional Pub/Sub systems have a number of drawbacks:

- **Limited expressiveness:** the notification filtering capability of these systems is limited, because publications can only be classified with respect to a number of channels or specific subjects/types. For example, in channel-based systems, if no channel exists that perfectly matches the interests of a subscriber, the subscriber has to subscribe to multiple channels or has to carry out additional filtering on its own.
- **Not allowing flexible changes:** in channel based system**,** the changes in the assignment of publications to channels would lead both producer and consumer to change. For example, the producer may be forced to publish publications to different channels. In subject-based system, changes to the subject tree would require major application fix: subscribers may have to subscribe to other subjects and may have to

carry out distinct subscriber-side filtering.

- **Not fully decoupled:** the publishers determine the channel/subject/types under which a notification is published. For example, the publishers in channel-based Pub/Sub system must have a clear agreement on what types of messages go in which channels, this cause the clients subscribing or publishing information not based on meaning or semantics of that information. Whereas, if it *was* based on semantics, then the meanings and semantics of available information are implicit, and the clients do not need to be pre-agreed on the same typed information.

These issues with traditional Pub/Sub systems spurred the development of Content-based Networking. The messages in Content-based Networking are delivered to clients based on a subscriptions which are applied to all the information presented in the messages, not just header information. Content-based Networking is discussed in the next section.

2.6 Content-based Networking

The author in (Carzaniga et. al., 2001) presents an introduction to Content-based Networking (CBN). In CBN, subscribing clients present a subscription, which are based on the data contents of events that they are interested in. Publications, whose payloads contain matching information are therefore implicitly addressed to a range of subscribers by the content they contain. CBNs operate as overlay networks, providing them with the flexibility to run on top of any underlying network protocol such as the Internet Protocol (IP).

CBN subscriptions are specified as a set of filtering constraints constructed using a filter operator and some values used to compare against the contents of any incoming notification. The range of these operators determines the type of Pub/Sub network in which the message is being sent. The author in (Mühl et. al., 2006) describes this content-based matching as a set of "Filters which are evaluated against the whole contents of notifications". An example subscription SUB containing two filters might look like:

> Laptop_brand = "IBM" AND Price > £600
>
> Where "=" and ">" are the operators.
> The two filter constraints are combined as a conjunction, using the Boolean AND operator.

Within this work, notifications can be thought of as publications. A publication is a set of attribute-values. An example publication PUB might be as follows:

> Laptop_brand: IBM
> Price: £750
> Colour: Black

Publications are only forwarded to a user when the contents of their subscriptions' filtering constraints matches a subset of the messages contents. For example, the publication: PUB is forwarded to the clients which have subscription SUB. This allows for a more flexible message format and, in comparison to traditional Pub/Sub systems, such as topic-based networks, allows an even looser coupling between publishers and subscribers.

However, open standards for CBNs have been slow to emerge due to the difficulty in reaching a general compromise between the expressiveness of event attribute types and subscription filters exasperated by the need to both match these efficiently at CBN nodes and to efficiently maintain routing tables. The need for efficient network utilisation requires that publications are routed towards nodes and subscribers that are interested in a particular message, using a routing table composed of subscription filters, rather than flooding the network in the search for all interested possible parties. This is usually combined with a mechanism to exploit multicast efficiencies made possible by aggregating/covering subscriptions with ones that match the same or wider range of messages. For example, in the Siena CBN (Carzaniga et. al., 2001) subscription covering is achieved by restricting attribute types and subscription filters to simple number, string and boolean types using a set of transitive operators to filter across them (i.e. greater/less than, super/sub string etc.). The underlying routing structure on which messages pass allows for a message inserted on one side of the network to propagate across the network based on positive matches to the filters (subscriptions) until every client, and only those interested in the message, have been delivered the message. In the following subsections, several implementations of CBN systems are described.

2.6.1 Siena

In the Siena CBN system (Carzaniga et. al., 2001) a notification is seen a set of typed attributes. Each attribute is comprised of a name, a type and a value. The existing version of Siena supports the following types: String, Long, Integer, Double and Boolean. A Siena notification is a set of typed attributes. Each attribute is a triple consisting of a name, type and a value, where the type is limited to one of "string", "time", "float" and "integer". A filter is constructed from a set of constraints which are each applied to the content of notifications. A constraint is a triple, consisting of the attribute name, a constraint operator, and a value. Where multiple constraints exist in a single filter they are evaluated as a conjunction. A filter "covers" a notification or event if that event satisfies each constraint applied to it by the content filter. An event or notification **n** is delivered to an interested party **X** if **X** has submitted a subscription filter that covers the notification. Also, a filter **f** "covers" another filter **f′** where together the set of constraints in **f** are more general than all of the individual constraints in **f′**, and so all of the

notifications that would be delivered or forwarded for **f′** would also be delivered or forwarded for **f**, i.e. **f** is more general than **f′**.

Carzaniga (Carzaniga et. al., 2001) defined three basic types of Siena topology: hierarchical client/server, acyclic peer-to-peer, and general peer-to-peer. All topologies provide the same functionality, however they differ in non-functional features, like time complexity, scalability and fault tolerance. Within this thesis as the implementation of proposed KBNMap architecture is an extension of KBNImpl (see Section 3.4.7), which is itself a major extension of the hierarchical version of Siena to provide semantic datatypes and subscription filter operators. It is worthwhile therefore to briefly describe the principle of how Siena routers work in a hierarchical topology.

Hierarchical-based Siena

In the hierarchical-based Siena, notification routers/brokers/servers are arranged in a hierarchy of nodes as seen in Figure 2-3, where each node maintains a partially ordered subscription tree structure that keeps track of subscriptions and so informs the notification forwarding strategy for that node. In this subscription tree structure more general covering subscriptions are at the top and more specific covered subscriptions are arranged as subtrees.

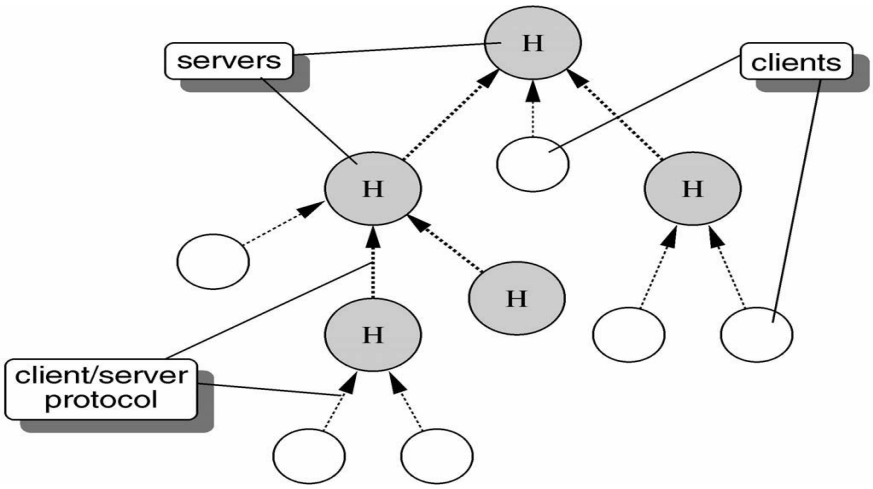


Figure 2-3: Hierarchical client/server topology (Carzaniga et. al., 2001)

Each node in the hierarchical topology may have any number of incoming connections, other than clients, but only one outgoing connection to its parent node. Conceptually, the nodes have a client/server relationship. Thus, a hierarchical node need only propagate information it receives to its parent node in the form of (root) covering subscriptions and publications. The main routing principle behind Siena is to push notifications as close as possible to parties that may be interested in that information. Known as downstream replication, this can be achieved both by subscription forwarding and advertisement forwarding. Subscription forwarding is the method used for routing in the Siena hierarchical implementation.

Routing Subscriptions

The tree of subscriptions is used to assist in pruning the number of subscriptions forwarded. Essentially, root subscriptions are the only ones sent. As such, subscriptions covered by previously forwarded subscriptions are pruned and network traffic is kept to a minimum. In order to ensure consistent notification across the network, Siena employs publication forwarding to master nodes, and leaves further notification beyond that of root subscriptions to the nodes on which the more specific subscriptions reside.

When the Siena node acting as the server to a notification producer **X** receives a subscription filter **f** from a client or sub-server **X**, the subscription tree is searched starting at each root subscription. If a subscription is found that covers the filter **f** and contains **X** in its subscriber set the search terminates. Otherwise, if the filter **f** already exists in the subscription tree, **X** is simply placed in the subscriber set of that particular filter. Finally, should neither of these apply a new subscription is inserted under the most specific covering filter, possibly a leaf node, with **X** added to its subscriber set. If no covering filter exists, the subscription is inserted as a new root subscription. All root subscriptions are forwarded to master nodes, possibly propagation right to the top of the Siena node hierarchy, with sub nodes acting exactly like subscribers.

Upon reception of notifications at a Siena router node (either from below from a notification producer or sub-server, or from above from a super-node) the set of clients or other sub-nodes with subscription filters covering the notification are sent that notification. If the master server was not the source of the notification than a copy of this notification is also sent to the master server, with each publications propagating all the way to the root master server. In fact, the relationship between a Siena node and its master is very similar to that of a subscriber client and the Siena node itself. The net effect of this is that no matter where a publication, or subscription, takes place on the network the correct subscriber subset is notified.

2.6.2 Elvin

Elvin (Segal et. al., 2000) is a content-based routing system similar to the Siena system introduced in the previous section. The use of content-based routing and a subscription mechanism with quenching, together with federation mechanisms enables it to use in large-scale applications. Aside from the presence of a mature, efficient implementation, the Elvin subscription language can be considered as a more expressive form of that outlined for Siena. Introducing logical operators and tri-state logic gives the Elvin subscription language an advantage over the Siena subscription language which still relies on conjunctions for pattern matching. The closed-source nature of the Elvin implementation is also prohibitive to us in

terms of enhancing content based routing to include more complex datatypes, such as ontological classes and properties.

2.6.3 Hermes

Compared with Siena that has a simple, predefined overlay topology, Hermes (Peitzuch et. al., 2002) is a CBN that uses a peer-to-peer technique to build a more scalable overlay topology of event router network for event dissemination. The peer-to-peer technique utilised in Hermes is a pastry-like routing substrate called PAN (Rowstron et. al., 2001). From a content-based routing algorithm perspective, it has two variants of routing: i) type-based routing that allows subscribers receive all events of a certain type, and ii) attribute-based routing that allows subscribers to further filter on the event type's attributes.

2.6.4 Gryphon

Gryphon is a content-based Pub/Sub system developed at IBM Watson research center (Gryphon 2007). The system is constructed as a redundant overlay network consisting of routers distributed across multiple geographic locations. Clients can use the Java Message Service API (JMS 1.1) to access the service. Gryphon has been deployed over the Internet to support real-time applications such as score distribution for Wimbledon 2001 and statistics reporting at the Sydney Olympics (Gryphon 2007).

Some important research contributions of Gryphon include an event matching algorithm with sub-linear complexity (Zhao et. al., 1999); a scalable protocol to achieve exactly-once delivery of information to a large number of subscribers in either publisher order or uniform total order; a subscription propagation algorithm which supports in-order gapless delivery and a congestion control mechanism to protect a Pub/Sub system against network failure and link congestion.

2.6.5 Advantages of CBN

Content-based Networks facilitate much looser coupling than traditional Pub/Sub systems through subscriptions that express interest only in some attributes of the event message's contents or payload, rather than just header information. This feature provides a more powerful and flexible notification selection than traditional Pub/Sub systems. Content-based Networks have formed around the necessity to match a varying subscriber base to that of publication creators. This achieves a further "de-coupling" of the parties involved in the communication process and allows for message routing to be conducted based on who is interested in a particular message, through a routing table compiled from with subscription filters. The underlying routing structure over which messages pass allow for a message inserted on one side

of the network to propagate across the network based on matching filters (subscriptions) until each and every client interested in the message has been delivered the message.

2.6.6 Disadvantages of CBN

The limitation of current CBN approach is that it only supports a very limited range of datatypes and operators for use in matching consumer subscriptions to message attributes, typically: Strings, Integers, Booleans, and associated equality, greater than, less than, and regular expression matches on strings. This falls well short of supporting the heterogeneity and flexibility that elements in a large scale network require to gather essential information. In other words, for a CBN to work on a large scale it needs to support a richer expressiveness that can cope with the widely heterogeneous and frequently changing range of message content and consumer subscriptions. In addition, most CBN implementations only allow a single comparison value to be specified in each filter, and since multiple constraints in a single subscription are usually combined as a conjunction, this greatly restricts the expressiveness of any single subscription filter. Rather than extending the subscription mechanism to support a disjunction of filtering constraints, which would greatly affect the ability to aggregate filters for efficient routing tables. Furthermore, scalable implementation of such system is difficult to realise. The authors in (Von 2002; Carzaniga 2001) state that the expressiveness of notification selection has a large impact on the scalability of any content-based notification services.

To address the disadvantages of CBNs, the Knowledge-based Networking (KBN), which is a Semantic-based Content-based Networking, is introduced. The messages of KBN containing semantic mark-up and queries, make KBN is potentially far more flexible, open and reusable to new applications. Knowledge-based Networking is discussed in the next section.

2.7 XML-based Pub/Sub System

Prior to describing Knowledge-based Networking, it is worthwhile to introduce XML-based Pub/Sub systems first, as they provide more expressive XML-based language (i.e., XPath) and more complex matching than CBN such as providing “many-to-one” matching rather than “one-to-one” matching in CBN.

With the emergence of XML as the standard for data representation and exchange, a lot of attention has been focused on pub/sub systems for XML-based dissemination. XML-based systems (Diao et. al., 2003; Gupta et. al., 2003; Chan et. al., 2006) publish events as XML documents, where XPath (Clark et. al., 1999) queries allow expressive and flexible subscriptions, where arbitrarily complex queries are applied to a DOM tree derived from the published XML document. The message is then forwarded only when a match is found. XML-

based systems do provide an increased level of expressiveness in comparison to CBN. However XML-based systems only add the structure information in subscriptions language rather than providing semantic mark-up and query, the message architecture is exclusively based on tree patterns without describing semantics and metadata contained in the data, and thus is less expressive and flexible than Semantic-based Pub/Sub systems.

2.8 Knowledge-based Networking

Keeney's work on the Knowledge Based Networking (Lewis et. al., 2006; Keeney et. al., 2007; Keeney et. al., 2008a; Keeney et. al., 2008b) has lead to a model for the filtered dissemination of semantically enriched knowledge over a large loosely coupled CBN of distributed heterogeneous agents. Knowledge-based networking (KBN) is defined as a semantic-based CBN in which the semantics of the message play an important part in the matching of publications to subscriptions. A KBN based on subscriptions and publications containing semantic mark-up is more than potentially far more flexible, open and reusable to new applications. This approach promises loose semantic coupling between applications, which is vital as new waves of applications increasingly rely on using the information and services offered by existing heterogeneous distributed applications. It should be noted that KBN as a semantic-based CBN is a subclass of SBPS systems.

An implementation of Knowledge-based Networking (KBNImpl) is developed by authors in (Lewis et. al., 2006; Keeney et. al., 2007; Keeney et. al., 2008a; Keeney et. al., 2008b). It offers an extension of Content-based Networking (Siena) which involves the routing of an event across a network, based not just on the values of an event's contents but also on some semantics of the data and associated metadata contained in the event itself. Producers of knowledge express the semantics of their available information based on an ontological representation of that information. Consumers express subscriptions based upon that information as simple semantic queries, thus enabling the efficient distributed routing of distributed heterogeneous knowledge to, and only to, nodes that have expressed a specific interest in that knowledge.

2.9 Conclusion

This chapter has aimed at outlining background foundations for the research. It has introduced the basic knowledge and background of ontology, especially how to use of ontology mappings to achieve semantic interoperability between heterogeneous ontologies. It has also presented the concepts and requirements of Publish/Subscribe middleware, which is an effective platform for heterogeneous information integration and distribution. To address the limitations of Pub/Sub

systems, the Content-based Networking systems (CBN) enhanced the expressiveness and scalability of traditional Publish/Subscribe system is introduced, which filter messages based on matching subscriber's subscription to message attribute values. Finally, in order to improve expressiveness and flexibility of CBN, the Knowledge-based Networking (KBN) is generally described, which enables CBNs to route an event across a network, based not just on the values of an event's contents but also on some semantics of the data and associated meta data contained in the event. The prototype KBN implementation is described in more detail in Section 3.4.7.

The next chapter gives detailed information of Semantic-based Pub/Sub systems, particularly, identified the taxonomies of SBPS systems with respect to their key features and semantic interoperability support. A comparison of existing SBPS systems is also presented in order to identify the key challenges faced by current SBPS systems.

.

3 STATE OF THE ART: SBPS SYSTEMS WITH SEMANTIC INTEROPERABILITY

3.1 Introduction

The use of ontological information in publish/subscribe networks is still far from wide-scale deployment. Exploratory research and a detailed review of the State of the Art shows a gradual shift towards an understanding of both the importance and power of semantics when combined with Publish/Subscribe networks. To our knowledge, there is limited number of Pub/Sub systems using ontology-based semantics. For this reason, it is worthwhile to propose a classification for such systems. As mentioned in Chapter 1, the Publish/Subscribe system (SBPS) as an abstraction was introduced to introduce a Pub/Sub system in which semantics of the message play an important part in the matching of publications to subscriptions. SBPS as an abstraction is different to Knowledge-based Networking (KBN), since it generally refers to any Pub/Sub systems making use of ontology-based semantics rather than KBN referring to the semantic enhanced Content-based Pub/Sub Networking. In other words, KBN is a subclass of SBPS similar to CBN that is a subclass of Pub/Sub system.

The aim of this chapter is to provide a state of the art survey of Semantic-based Publish/Subscribe (SBPS) systems and their adaptive support for dynamic Semantic Interoperability. To our knowledge, there is no taxonomy identifying the key features of SBPSs and classifying them based on the identified key features. Section 3.2 presents a basic taxonomy of SBPSs according to key features. Section 3.3 describes an advanced taxonomy of SBPSs that has been developed to take into account semantic interoperability features. Section 3.4 describes selected state of the art SBPSs and reviews them in the light of the key features and their ability to support semantic interoperability. Section 3.5 provides a summary of the selected state of the art systems and how they address the two taxonomies identified. Finally, Section 3.6 summarises the key challenges that are explored through the analysis of current SBPSs in the sections above which need to be addressed by a SBPS in order to deliver heterogeneous information in an efficient manner.

3.2 Basic Taxonomy of SBPSs with respect to Key Features

There are a number of taxonomy papers available that classify existing Publish/Subscribe systems (Eugster et. al., 2003; Meier et. al., 2005; Wun et. al., 2007; Mahambre et. al., 2007) that have focused on classifying and comparing Publish/Subscribe systems according to their fundamental characteristics such as subscription scheme, matching algorithm, system architecture and so on. To author's knowledge, there is no effort so far in the direction of identifying key features of Semantic-based Publish/Subscribe systems and classifying them based on their identified features.

This section presents a structural taxonomy that is structured as a hierarchy of properties that serves as a basis for identifying key features required by a SBPS system, in order to support semantic interoperability in an adaptive manner. The *basic* taxonomy will lead to classification and comparison of existing systems on the basis of the identified key features. Whilst a second *advanced* taxonomy highlights existing systems' ability to adaptively support flexible semantic interoperability based on a number of criteria and brings out key challenges that need to be addressed by current SBPSs in the context of distributing heterogeneous information in decentralised, large scale environments.

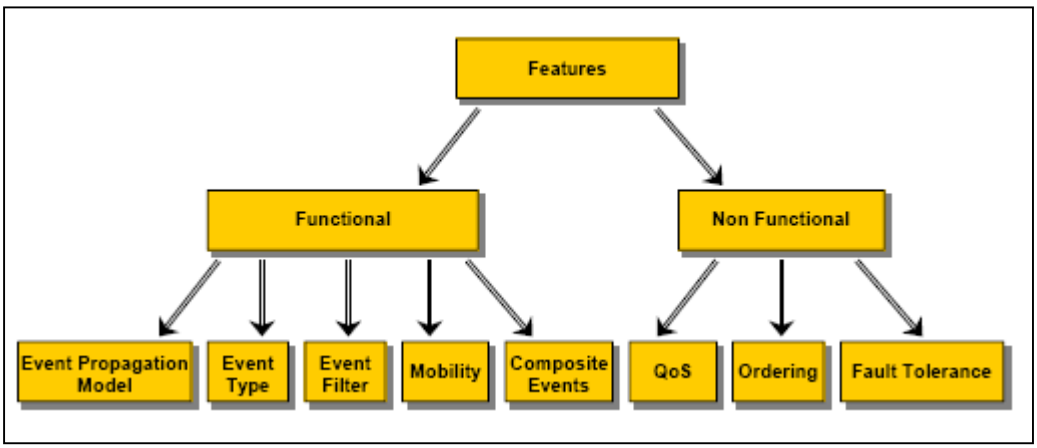
3.2.1 Structural Taxonomy

It is shown in other papers (Petrovic et. al., 2003; Burcea et. al., 2003; Wang et. al., 2004; Halaschek-Wiener et. al., 2007) that the Semantic-based publish/subscribe system is an effective platform for integrating and distributing ontological heterogeneous information. However, *expressiveness* and *scalability* that are difficult to trade off in traditional Pub/Sub systems (Carzaniga et. al., 1999; Carzaniga et. al., 2000) are still two major conflicting features for realising such a system. In the context of SBPS, *expressiveness* refers to the richness of the semantic data model that is offered to publishers for publishing semantic publications and the richness of subscription language that is offered to subscribers for subscribing to publications. *Scalability* refers not only to the numbers of routers, publishers and subscribers, and the constructed network topology, but also the schemes used to route subscriptions and publications, and the algorithms employed for filtering irrelevant publications. Clearly the level of *expressiveness* influences the schemes and algorithms used to route and deliver publications, and the extent to which those algorithms can be optimised. As the richness of the data model and subscription language increases, so does the complexity of the algorithms. For example, some systems such as CQS in (Halaschek-Wiener et. al., 2007) offer a rich event matching

mechanism but with a centralised architecture, while others such as SPS-SP in (Chirita et. al., 2004) adopt a more scalable distributed architecture, but they give low level accuracy in matching and filtering events. The authors in (Wang et. al., 2004) stated that the more expressive a SBPS is, the more difficult it is to have an efficient matching algorithm and vice versa, because highly expressive subscriptions require complex and expensive matching and routing algorithms, and thus limit the scalability.

Figure 3-1: Fundamental characteristics of Pub/Sub systems (Meier et. al., 2005)

Most existing SBPSs aim to enhance the expressiveness of the event model and improve subscription languages to be more expressive so that subscribers can more effectively express their subscriptions, whilst keeping the efficiency of matching algorithm to enable the scalability of the system. In this research, *expressiveness* and *scalability* properties are considered as **key features**. As SBPS systems are still a subclass of Publish/Subscribe systems, the **key features** of SBPS systems were characterised on the basis of the **fundamental features** of traditional Pub/Sub systems. In the existing taxonomy works for Pub/Sub systems (Eugster et. al., 2003; Meier et. al., 2005), the **fundamental features** are generally classified as the *functional* or *non-functional* characteristics for a Pub/Sub system. As can be seen in Figure 3-1, the authors in (Meier et. al., 2005) identified a taxonomy that addresses the *functional* and *non-functional* features provided by a Pub/Sub system.



T

systems rather than reflect the functionalities or QoS level of SBPS systems. Extending Figure 3-1, the **key features** of Semantic-based Pub/Sub systems can be termed as **semantic features** of Pub/Sub systems, and considered as an additional subclass of features in Pub/Sub systems. In addition, in the proposed *advanced* taxonomy (Section 3.3), existing SBPSs are also categorised according to their capability to adaptively support semantic interoperability.

The **semantic** feature indicates how a SBPS system semantically enhances its event model and subscription language so as to improve its expressiveness and scalability, rather than reflects a concrete functionality. Thus it can be considered as a subclass of the *non-functiona*l feature in

Figure 3-1. As can be seen from Figure 3-2, the **semantic feature** is a new additional sub-branch of *non-functional* features for Pub/Sub systems, which is further expanded with *key features* and *semantic interoperability support* sub branches. The *key features* branch can be used to easily evaluate the semantic enhanced level of existing SBPS systems. It can be observed that the subclasses of *key features* in Figure 3-2 are similar to that of *Functional features* in Figure 3-1 (i.e., Event model feature of Figure 3-2 to Event type of Figure 3-1). But as stated before, these subclasses (fundamental characteristics) are used to reflect the *expressiveness* and *scalability* that in turn reflect the semantic enhanced level of SBPS systems, rather than their functionalities. For convenience, these fundamental characteristics are used as subclasses again in Figure 3-2. Whilst the *semantic interoperability support* (linked with dash line in the figure) branch is the advanced characteristics of SBPS systems, which can be used to evaluate a SBPS's capability for dealing with heterogeneous information (more detail of this branch is described in Section 3.3).

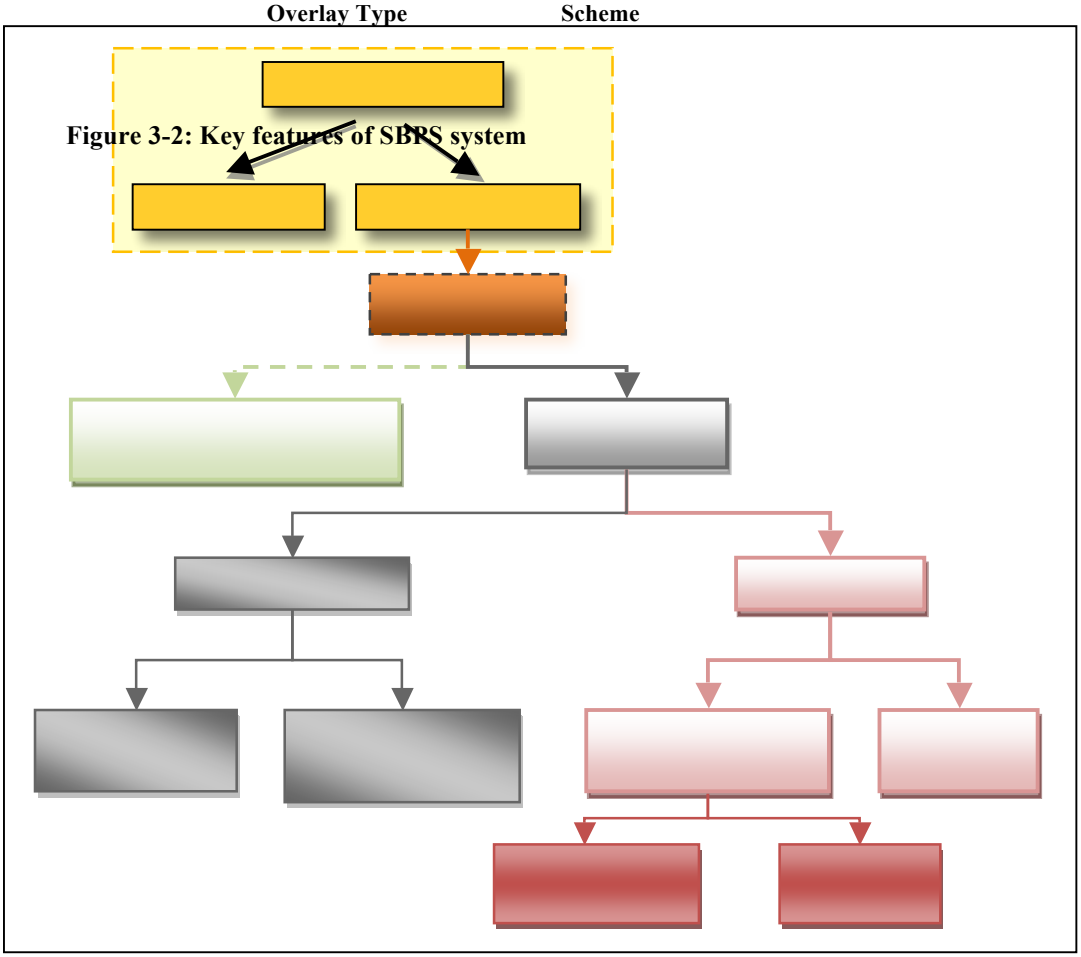


Figure 3-2: Key features of SBPS system

3.2.2 Key Features

In this section each of the key features identified are described. The key features are characterised at the first level as *expressiveness* and *scalability*: the event model and subscription language playing an important role on characterising *expressiveness*; whereas the construction of network topology, information routing scheme, efficient event filtering method and subscription aggregation have more significant impact on system *scalability*.

3.2.2.1 Key Features: Expressiveness

The authors in (Cazaniga et. al., 1999) stated that expressiveness is the key requirement in publish/subscribe systems, which demands a rich subscription language so that applications has a flexible and fine-grained selection mechanism to describe precisely those events or combinations of events in which they are interested.

Expressiveness: Event Model Feature

Any happening of interest that can be observed from within a computer is considered as an event (Mühl et. al., 2006). Events are reified by a populated notification data structure which is delivered to an interested party (Mühl et. al., 2006). Generally, the notification is a set of typed attributes that reflect the events; each attribute is comprised of an attribute name, type, and value. The data model or the encoding schema of notifications is referred to as an event notification model or simply event model. Within this thesis, the notification can be thought of as publication. The event model defines what information can be communicated among publishers and subscribers by means of events, or at least how the information has been encoded. The event can be characterised by the kind of data it stores. Most existing CBNs support the following generic data types: *String, Long, Integer, Double and Boolean*.

SBPSs utilise semantic web technologies to enhance the events model semantically, so that the events can be distributed across a network not just based on the values of any attributes contained but also on some semantics of the data and associated meta-data contained in the attributes. With ontologies, the semantics inside events can be described by concepts, types, and instances of types, which enable the addition of relations between concepts, separations, similarities, and most importantly allow for classification of concepts in a taxonomic manner. Furthermore, it is worthwhile to note that it is desirable to allow a SBPS to support both generic and semantic datatypes, so that the SBPS systems like traditional Pub/Sub systems can also be flexibly used in the application domains where their information model is not expressed by semantics. Two semantic-web ontology languages have been mainly used in SBPS systems to express event models:

- **OWL language:** as described in Section 2.2.2, OWL specifies a way to represent things using the classes, properties, and individuals, it brings possibility for typical Pub/Sub systems to support new ontological types: classes, properties and individuals. Therefore the producers of knowledge could express the semantics of their available information based on an ontological representation of that information. The expressiveness of an OWL-based SBPS system with respect to semantic event model can be categorised by the degree to which it fully supports the OWL constructs and relationships.
- **RDF language:** as described in Section 2.2.2, RDF and RDFS are another ontological way to describe facts using the subject, property, object triples. Each triple is called a statement, in which subject and property are URI, and object can be URI or literals. So the event model can be represented as a directed labelled graph, in which nodes represent subjects and objects of statements, and edges represents properties of statements. The expressiveness of a RDF-based SBPS system with respect to semantic event model can be evaluated by the degree to which it fully supports the RDF constructs and relationships.

In addition, XML is also used by some SBPS systems such as OBPS in (Skovronski et. al., 2006) for expressing event model, in which the XML tags represents the name of classes and properties of a semantic event model.

Expressiveness: Subscription Language Feature

Tightly related to the event model is the subscription language, which defines the format of the filtering expression submitted with subscriptions. The types of semantic data contained in the subscriptions and types of semantic operators that the subscriptions can support, significantly determine the expressiveness of a SBPS system with respect to subscriptions. There are two characteristics of subscriptions that are important for the expressiveness of SBPS systems:

- **Subscription expression:** this refers to how user subscriptions are formed. Similar to CBN, the basic format of subscriptions in OWL-based SBPS systems such as KBNImpl and S-ToPSS is [attribute_name, operator, value]. Whilst the basic format of subscriptions in RDF-based SBPS systems and SPS-P2P is either RDF graph pattern [Subject, Object, meta-statement] (i.e., OPS) or a Datalog-inspired RDF query (i.e., SPS-P2P), or a SPARQL query (OPBS). Additionally, if a SBPS system allows both generic content-based and semantic-based subscriptions to coexist, it would be more flexible and expressive than the SBPS system only supporting semantic-based subscriptions.

- **Type of operator used:** The operators used in subscription determine the sophistication of subscriptions. The author in (Keeney et. al., 2008a) states that a subscription would be expressive and flexible if it could support both non-semantic operators and semantic operators. Similar to CBN, the generic typed operators include common equality and ordering relations (i.e. =, >, <, etc) for all of its types; the string operators; and any operator (i.e. probability in S-ToPSS) that matches any value. The semantic typed operators include subsumptive subscription operators, i.e. sub-class/property (MORESPEC) i.e. *more specific*, super-class/property (LESSSPEC) i.e. *less specific*, and semantic equivalence (EQUIV). The KBNImpl implementation in (Keeney et. al., 2008a) also supports a generic ontological property operator to support filters in the form [attribute_name , property, object].

3.2.2.2 Key Features: Scalability

This taxonomic heading in Figure 3-1 is addressed by separating the scalability of *event distribution* with the scalability of *event matching*. In the *event distribution layer*, the scalability concern is how a system organises routers, publishers and subscribers together for cooperatively distributing the subscriptions and events: the routers must be arranged into an interconnected topology and make use of one or more routing communication protocols. The combination of physical topology and communication protocol refers to network overlay type that a system adopts for deploying networks. In addition, the scalability in this layer also refers to how to efficiently route publications/subscriptions from publishers/subscribers to subscribers/publishers, which is termed as *Routing Scheme*. The scalability concern in event matching layer is how to efficiently match subscriptions with publications. The following subsections first describe the scalability of SBPSs with respect to event distribution layer and then present how the event matching algorithms used by SBPS affect scalability.

Scalability: Event Distribution Layer: Network Overlay Type

In a SBPS communication environment, the knowledge publishers, subscribers, and knowledge routers, all work together to construct the overlay network topology, which determines the formation of the knowledge dissemination tree in the SBPS system when routing knowledge, and reflects the level of scalability of a system. It is observed that most existing SBPSs use topologies adopted by traditional Pub/Sub systems to connect their routers. For example, Figure 3-3 illustrates that KBNImpl (Keeney et. al., 2008a) basically adopts the hierarchical topology from Siena (Carzaniga et. al., 2001) for configuring router network. Another example OPS (Wang et. al., 2004), adopts both hierarchical and peer-to-peer topologies for configuring router network. In this part of the taxonomy, as shown in Figure 3-3, a set of topologies has been

identified on which most of the existing SBPSs are based. All the SBPSs implement one or a combination of the topologies discussed below.

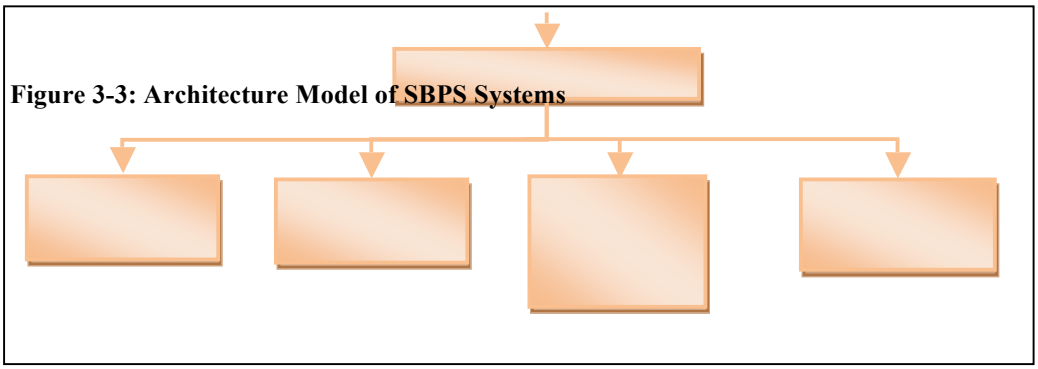


Figure 3-3: Architecture Model of SBPS Systems

- **Centralised server:** some SBPS systems such as OBPS (Skovronski et. al., 2006) and CQS (Halaschek-Wiener et. al., 2007) only deploy one router to offer a centralised knowledge delivery service, in which the central broker is responsible for storing all subscriptions and matching all events. However, as the system scales with a large number of subscriptions and a vast volume of event messages, the matching performance of this central server will be inevitably degraded.
- **Hierarchical topology:** the hierarchical topology (See Figure 3-4) is distinguished by a hierarchical relation between routers. The routers are connected in a hierarchy of parent-children relationships. Each parent node can have a number of children that may be publishers, subscribers, or intermediate routers; whereas each child node could only have one parent. In this topology, the communication between router-router and client-router follows the same protocol. The purpose of hierarchical organisation is scalability. A parent will receive published events and subscriptions from all of its clients but will forward to its children nodes only events destined for those children nodes. In other word, the parent node acts as a gatekeeper aiming at shielding general traffic from its children nodes.

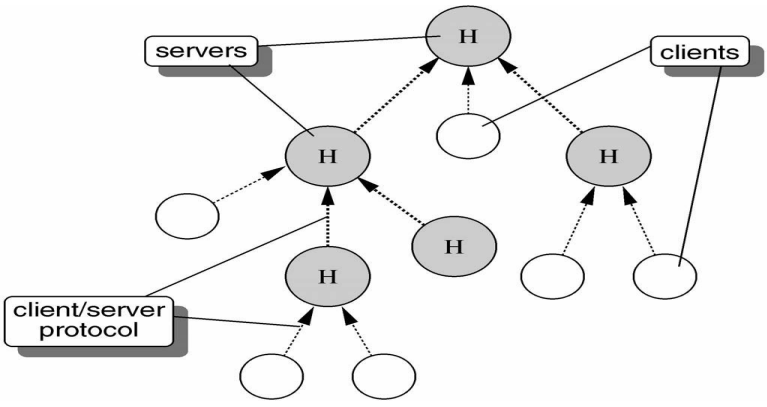


Figure 3-4: Hierarchical client/server topology (Carzaniga et. al., 2001)

- **Semantic clustering topology:** the idea of this topology introduced by (Keeney et. al., 2008b) is to cluster routers, publishers, and subscribers based on their semantic footprint and interests (see Figure 3-5). So the clients can subscribe or publish semantically similar information within the scope of cluster rather than to the entire network. The main advantage of this is to reduce the processing time involved in making routing decisions and reduce hops to get information from source to destination.

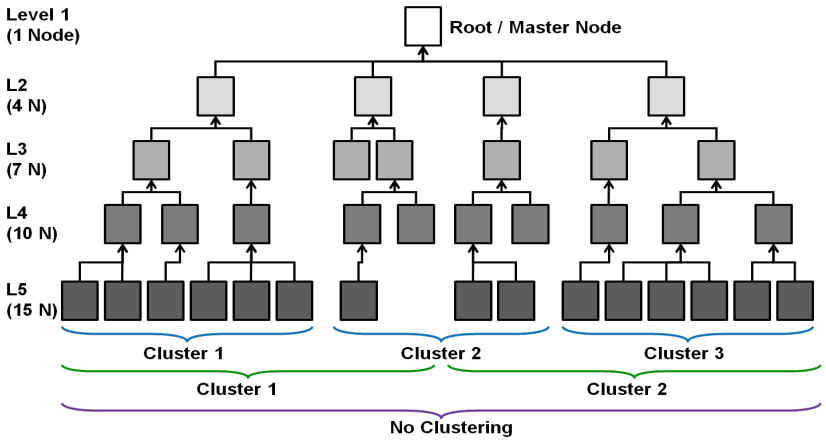


Figure 3-5: Semantic clustering topology (Keeney et. al., 2008b)

- **Peer-to-peer topology:** in this topology, each node can act as a publisher, subscriber, root of a multicast tree, internal node of a multicast tree, or any reasonable combination thereof. The router nodes are arranged in a P2P manner and also some of the router node functionality is embedded as a local part of each client node. As an example, the SPS-P2P system (Chirita et. al., 2004) is comprised of peers with the role of publisher and subscriber. In addition, it also adopts super-peers that are organised within the HyperCup (Schlosser et. al., 2002) semantic topology (See Figure 3-6) that takes care of processing messages by way of P2P semantic queries.

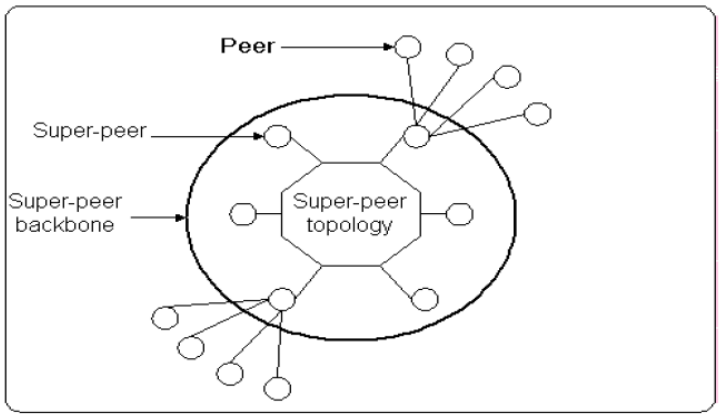


Figure 3-6: peer-to-peer topology with super peers (Chirita et. al., 2004)

Scalability: Event Distribution Layer: Routing Scheme

Once the topology of routers is set up, the routers must establish appropriate routing paths to ensure that publications are correctly forwarded to all subscribers that expressed interest for them. The publications must be matched with subscriptions somewhere in the network in order to filter out the irrelevant publications and only deliver the appropriate publications to subscribers. This principle can be achieved by routing schemes. Usually the routing scheme represents two distinct aspects: subscription forwarding and publication forwarding. Subscription forwarding is used to propagate clients' interests in the system, while publication forwarding algorithm decides how to disseminate the events to the interested clients.

- **Subscription forwarding**: the main idea behind this scheme is that the routing path for publications is set by subscriptions, which are propagated through the network so as to form a tree that connects the subscribers to all routers in the network. When a publisher publishes a publication that meets the subscription, the publication is forwarded to the interested subscriber following the reverse path put in place by the subscriber.
- **Publication forwarding:** the main idea of this scheme is that the subscriptions are maintained locally at their access point and to disseminate events through the whole network; when a notification meets and matches a subscription, the interested subscriber can get the notification immediately and locally.

Scalability: Event Matching

The event matching algorithm controls the way in which events are matched to the interested subscribers. A matching method determines the subscriptions, and thus the recipients that are matched by a given notification. Considering that most SBPSs are an enhancement of CBNs, the matching methods implemented by CBN are applicable to SBPSs as well. Most existing CBNs use a matching tree algorithm. This algorithm pre-processes a set of subscriptions into a matching tree, with each node a partial condition on the attributes of a predicate. Each sub level of the tree is the refinement of the test performed at the next higher level. Subscriptions are resident at the leaves of the tree. Upon arrival of an event, the subscriptions matching the event are found by navigating the decision tree starting at the root. This algorithm's linear time-complexity with respect to the amount of subscriptions, and linear space complexity is efficient for most Pub/Sub systems. However, other matching algorithms are considered in the discussion of some individual SBPS systems. For example, CQS (Halaschek-Wiener et. al., 2007) uses a composition matching algorithm that is composed of three individual sub-matching algorithms to match events. A detailed description of CQS can be found in Section 3.4.4. For a large-scale distributed system, the workload for matching events in routers might become heavy with the

large number of clients. Therefore the efficiency of a matching method has significant impact on the performance and scalability of a SBPS system. So the design of event matching method in SBPS must keep balance of expressiveness and scalability.

In summary, the key features that reflect the semantic enhances level of a SBPS system are listed as follows:

- **Expressiveness**: refers to how a SBPS semantically enhance its event model and subscription language.

  1. *Event Model Feature:* refers to which semantic languages are used to express event model.

  2. *Subscription Language Feature:* refers to how to use ontology-based semantics to improve the subscription to be more expressive.

- **Scalability**: refers to how a SBPS system scales its system.

  1. *Network Overlay Type:* refers to how a SBPS system organises its routers and clients to construct the network topology.

  2. *Routing Scheme*: how a SBPS to route its information via the network.

  3. *Event Matching:* how a SBPS router efficiently match events with subscriptions.

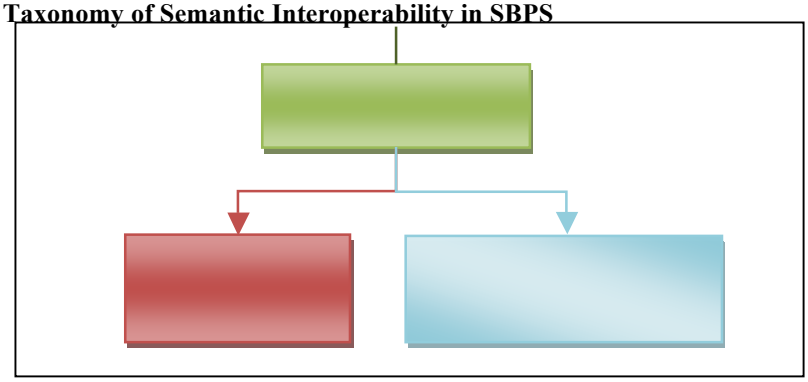
In next subsections, the *advanced taxonomy* with respect to semantic interoperability support in SBPS systems is presented.

3.3 Advanced Taxonomy of SBPSs with respect to Semantic Interoperability

The majority of investigations on current SBPS systems have assumed that the publisher and subscriber applications share a single common semantic model. As described in Section 2.3.2, the semantic model is the key factor for achieving a level of semantic interoperability. However, considering the ontology heterogeneity case study described in Section 2.4, if a SBPS system is to be used to offer this policy delivery service between policy decisions points, it needs to allow heterogeneity between policy decisions points using different information models. Therefore, it is the task of the SBPS to ensure that the heterogeneous knowledge can be resolved and distributed to its destination correctly and efficiently. This section presents the taxonomy of SBPSs with respect to the adaptive and flexible semantic interoperability feature necessary to support SBPSs in delivering heterogeneous information. Figure 3-7 represents an extended taxonomy of SBPS (See Figure 3-2), in order to help to illustrate “adaptive semantic interoperability” explicitly, this functionality is further divided into two directions which are

"semantic interoperability" support and "adaptive" configuration of semantic interoperability. The former refers to a SBPS's basic ability to deal with the heterogeneity between different semantic models, while the latter refers a SBPS's capability to configure the semantic interoperability dynamically in order to adapt to changes of environmental conditions.

Figure 3-7: Basic Taxonomy of Semantic Interoperability in SBPS



3.3.1 Semantic Interoperability Support

As described in Section 2.3.2 and Section 2.4, it can be considered that the ontology mapping technique is a popular and effective approach to achieve semantic interoperability. The advantage of ontology mapping is that it does not intend to unify semantic models and their data, but to transform terms in semantic models according to semantic relations (mapping relations) defined at a conceptual level. Ontology mapping could provide a common layer for SBPSs by which multiple and diverse semantic models could be accessed and information exchanged. The multiple semantic models are therefore kept separated, independent and distinct, maintaining their complete semantics and content.

In this taxonomy, the SBPS systems are categorised according to their way of expressing ontology mappings, the purpose for using ontology mappings and when/where mappings are used in the SBPS. The structure of the taxonomy is shown in Figure 3-8, and described in the next subsections.

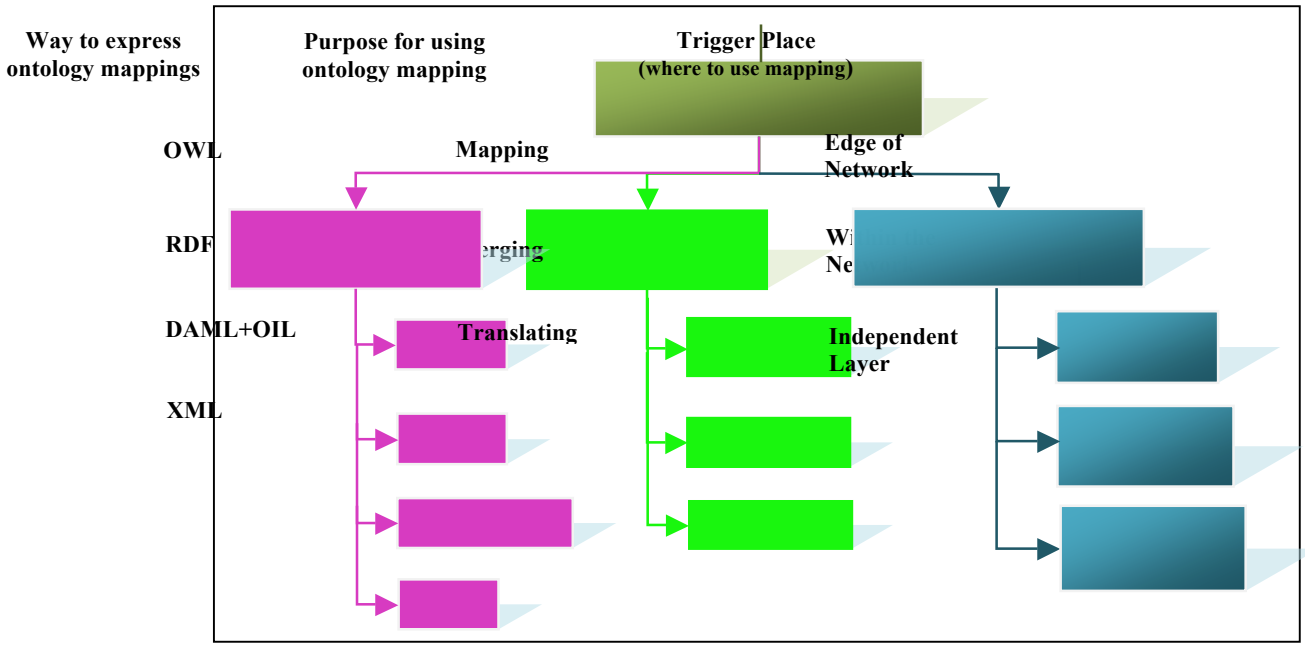


Figure 3-8: Semantic Interoperability Support in SBPS system

3.3.1.1 Way to Express Ontology Mappings

Having chosen semantic models as the means for expressing information models, it was natural to investigate the capability of languages for expressing mappings between models in order to achieve interoperability. As mentioned in Section 2.3.2, no clear standard has yet emerged to provide a format for expressing semantic mappings. The format of mappings can be expressed by using ontology languages, such as RDF, DAML+OIL, and OWL which are standardised by W3C and widely used by semantic web and applications; and web languages such as XML that is popular for sharing structured data in internet. A SBPS system could take advantage of using ontologies to express mappings since they facilitate semantic interoperability in significant ways: first they provide the structure and technical framework to reuse existing ontologies; second they provide formal mechanisms to express logical equivalences, subsumptions between concepts, relations, and individuals in different ontologies; third, they can enable computers to do automated reasoning. On the other hand, a SBPS system could also use XML to specify the format of their mappings, due to not only its amenability to application processing and human interpretation but also its popularity and the wide availability of technology for processing. However, the choice of mapping language must be carefully considered by each type of SBPS system in order to maximise interoperability and mapping reusability.

3.3.1.2 Purpose for Using Ontology Mapping

A number of candidate mapping ontologies can be provided by administrators at network load time and utilised at a SBPS system runtime to achieve semantic interoperability. This criteria focuses on the purpose of a SBPS system to utilise ontology mapping rather than their design, which is derived from the purpose of using ontology mappings in Section 2.3.2:

- **Mapping** (Noy et. al., 2000)**:** a SBPS system could use appropriate ontology mappings to relate any similar concepts or relations from different sources to each other according to the requirements of applications.
- **Merging** (Su 2004)**:** refers to enabling a SBPS system to merge two or more existing ontologies with overlapping parts. The overlapping parts are obtained from ontology mappings that usually are collected into an independent mapping ontology. This method is well suited in the situation where there is a common scalable semantic model shared among SBPS routers, the routers could merge new sub-models or new terms into the shared semantic model.
- **Translating** (Dou et. al., 2003)**:** refers to the process of translating vocabularies or structures from one ontology to another. A SBPS system can make use of ontology mappings to actually translate the concepts, properties, individuals from one ontology into the other for applications.

3.3.1.3 Trigger Place (where to execute ontology mappings)

This refers to where the system encounters heterogeneous information so that it triggers execution of ontology mappings to address the heterogeneity. It is clear that the trigger place highly depends on where the notification meets the subscription in the network. The ontology mapping execution therefore could be triggered at:

- **Edge of Network:** this scenario happens in the network: a knowledge subscriber receives a publication that actually matches his interest; however this incoming notification originated from a publisher who uses a different ontology from the subscriber. So the conflicting notification and subscription mismatches need to be resolved at the access point by using ontology mappings.
- **Within the Network:** the publication can also match subscriptions at intermediate routers within the network, if the network employs a subscription forwarding scheme. If a router encounters the heterogeneous ontological information that is not originated from its own ontology, it can resolve the heterogeneity through the use of ontology mappings.

- **Independent Layer:** it means that the heterogeneous information could possibly be solved independently of the SBPS middleware. The independent layer could be built between the SBPS and the Pub/Sub applications.

3.3.1.4 Flexibility

The authors in (Castano et. al., 2005) provide a set of different ontology matching methods to deal with different ontology matching scenarios in open networked systems. Similar to that, rather than relying on a single semantic interoperability mechanism, a SBPS system should also provide a set of mechanisms suited for dealing with many different scenarios where the situations of encountered heterogeneous information and the requirements of applications to address heterogeneity might different. For example, several evaluation works in (Pan 2005; Pellet 2003; Lewis et. al., 2006) and our evaluation work described in Section 5.2.1 have shown that various semantic models could lead to differences in reasoning performance of a specific reasoner, a vital component in the SBPS systems for managing and reasoning the semantic models. The existence of various semantic models in turn directly determines the multiplicity and diversity of mapping files. Therefore, it is necessary to have a set of mechanisms for loading mappings to deal with many different mapping situations where the number and type of ontology, and mappings features that can be exploited during the mapping process is not known in advance.

3.3.2 Adaptive Configuration of Semantic Interoperability

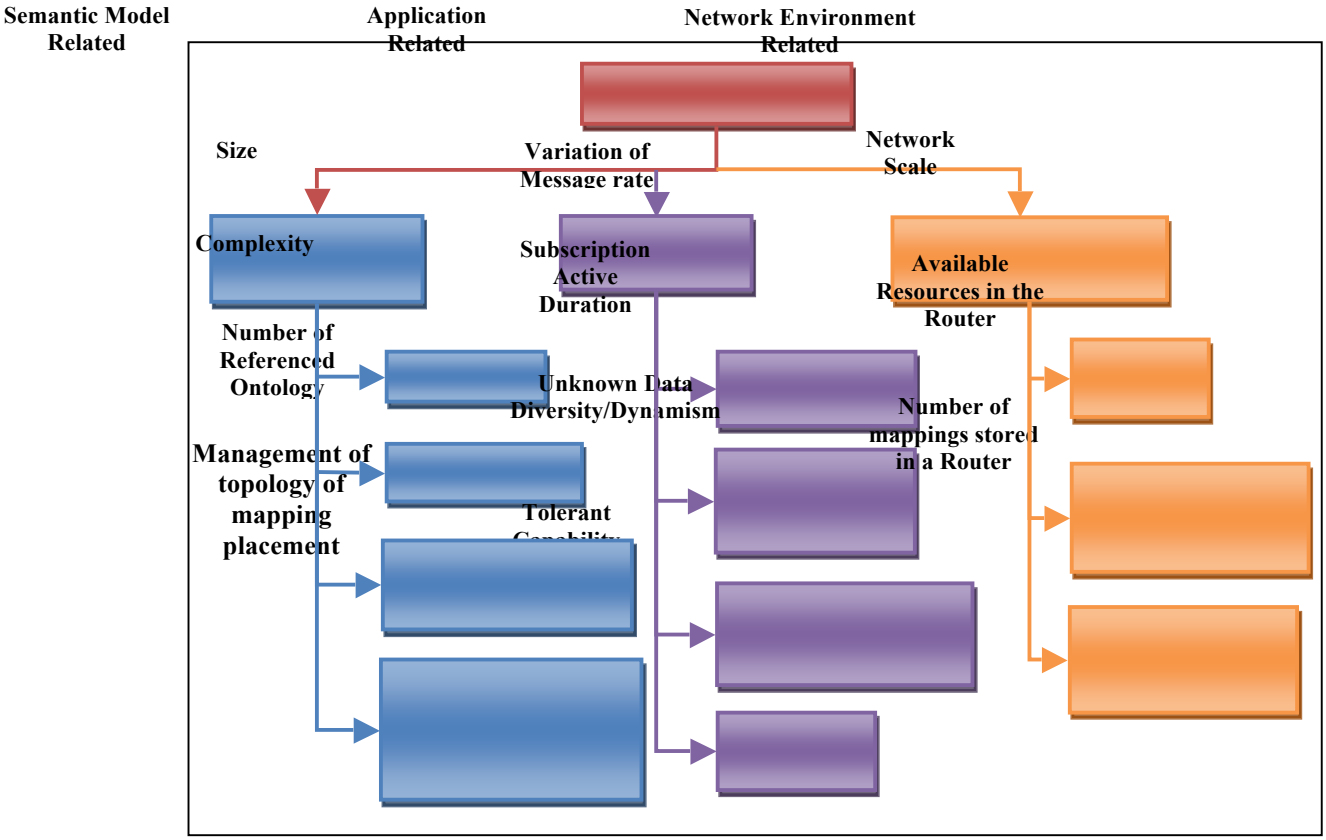
As discussed in the previous Section 3.3.1.4, it is desirable to enable a SBPS to have multiple mechanisms for loading and using mappings suited to different scenarios in order to provide a flexible semantic interoperability support for networked systems and applications. However, a critical challenge arises with the appropriate management and dynamic configuration of mapping services, adapting to the semantic complexity of the ontologies, dynamisms of network and changes of environmental conditions. For example, the environmental state of a network may be changed (nodes dynamically join and leave); messages that are semantically ambiguous at one time may be unambiguous if viewed later after the range of possible messages becomes more refined. Furthermore, in a small and centralised environment, it may be possible to examine the range of semantic models and applications running over the network to statically determine which mechanism is the most appropriate. However, in a large and decentralised environment, new semantic models and mapping files could be introduced at runtime and the state of applications using the network changes in unanticipated ways. Thus it is necessary to enable the SBPS to dynamically configure its semantic interoperability mechanisms in accordance with the changes. If some states change, the system should be able to execute a series of actions in order to ensure the quality of semantic interoperability and to meet the

requirements of applications. Some examples of changing states include the variation of semantic model, application and the dynamism of network environment. In this section, the adaptation requirements in SBPS were modelled using the concept of *Influential factors*.

As shown in Figure 3-9, the possible influential factors were categorised into three main sets:

- according to the importance of semantic model in SBPS systems, the first set covers semantic model related influential factors which includes the inherent structure of semantic models, mapping ontology and imported ontology referenced in the mapping ontology (size and complexity); the numbers of imported ontologies in a mapping ontology.
- as the ultimate goal of SBPS system is to offer a high quality and efficient knowledge service in order to satisfy applications' demands, the configuration of mapping mechanisms should be in accordance with the changes of application characteristics that are termed as application related influential factors, which contains the client mobility; variation of message rate; changes of unknown data occurrence rate; the message active/inactive duration and tolerance capability of application towards message mismatches.
- The authors in (Agoulmine et. al., 2006) mentioned that a self-managing system should have the capability to dynamically configure itself in order to adapt to changes of environment states. Therefore, a well-designed mapping service also needs to enable a SBPS to be adaptable to the changes of environment states that are termed as network environment state relevant influential factors. These influential factors include the network scale of a SBPS, the available resources allocated to a router and the numbers of mapping ontologies stored in a router. The following sub-sections describe each type influential factors in detail.

Figure 3-9: Trigger factors for supporting adaptive Semantic Interoperability



3.3.2.1 Semantic Model Related Influential Factors

Ontology-based semantic systems or applications require some kind of reasoner to manage and reason their ontologies in order to represent implicit information of ontologies explicitly. For example, DL reasoners such as Pellet (Pellet 2004) provide sound and complete reasoning services for OWL-based applications to manage their OWL ontologies properly. However, the constructs of ontologies have direct impact on the reasoning performance of a reasoner. Research in (Lefort et. al., 2006; Motik et. al., 2006; Termpich et. al., 2003) have shown that the size and complexity (DL expressivity) of ontologies are reasonable indicators with respect to reasoning performance. In addition, the author in (Halaschek-Wiener et. al., 2007) indicates that any small changes over a semantic model would lead to re-reasoning over the changed semantic model. As ontology-based SBPS systems also need reasoners to reason the required mapping files and ontologies, the semantic model characteristics that may cause a SBPS to adjust its mapping strategy are: the size and complexity of the applications' semantic models, routers' semantic models, mapping ontologies. Furthermore, our evaluation work described in Section 5.2.1 has shown that even though the size of a mapping file is usually small, the ontologies referenced in the mapping file still introduces expensive reasoning overhead when a reasoner reasons over the mapping file. The detail of ontology related characteristics is shown as following:

- **Semantic model/ mapping file size and shape:** the number of statements in an ontology determines how much the resource is consumed by a reasoner embedded in the SBPS router for reasoning this ontology (Lefort et. al., 2006; Motik et. al., 2006). In addition, the complexity of ontology structure has a negative effect on the reasoning process (Termpich et. al., 2003).
- **Number of Referenced Ontologies:** Once a reasoner in SBPS router processes a mapping ontology, the ontologies referenced in the mapping file are also to be processed by the reasoner. Therefore, the numbers of referenced ontologies impacts the overhead consumed by reasoning a mapping file.
- **Management of topology of mapping placement:** mapping ontologies may be replicated among SBPS nodes, or not. Considering that diverse mapping files could be assigned in different routers, managing these multiple and diverse mapping files in an efficient way may improve the performance of SBPS. For example, the master/root router in the hierarchical topology could distribute its own mapping files to the sub routers to help them to solve heterogeneity problem.

3.3.2.2 Application Related Influential Factors

If a SBPS system aims to ensure the quality of a mapping service it offers and adequately meet the requirements of applications, it should have the capability to adaptively configure its mapping service in accordance with the dynamics of applications. The dynamic features could include any variations of the applications such as client mobility that is represented by a "disconnected" followed by a "reconnected" operation, and message rate variation referring to the changes in the frequency of events published and subscriptions subscribed. However, as the main concern of SBPS with respect to semantic interoperability is how to successfully distribute heterogeneous information to the applications in an efficient manner, the variations relevant to the messages generated by the applications is only considered. Since the main processing overhead of the router is the maintenance of its subscription tables, and the matching of publications to subscriptions, it is necessary that the semantic interoperability service is cognisant of the processing load in the router.

- **Variation of message rate:** the constantly changing rate of publications and subscriptions will result in varying volumes of potential heterogeneous data. If the increasing number of heterogeneous data is beyond the amount that a mapping mechanism could handle, a part of heterogeneous data may still remain unknown. In other words, it could directly lower the grade of semantic interoperability that a SBPS supports. On the other hand, the decrease of publication and subscription rates would lead to the reduction in the occurrence of heterogeneous information. Therefore, it is

desirable to enable the SBPS to accordingly reconfigure its semantic interoperability in order to efficiently utilise resources.

- **Subscription active/inactive duration:** When taking into account with the subscription/publication rate, subscription active duration refers to the time between a client subscribing the subscription and unsubscribing that subscription. As it will take time for the router to operate a mapping service at runtime, the router must ensure resolution of heterogeneous subscriptions within their active duration. For instance, if the active duration of subscription is becoming shorter, the system should accordingly use a faster mapping mechanism.

- **Tolerance capability:** refers to the tolerance capability of the applications to respond gracefully to a mismatch of a publication to a subscription, which can be further explained as the tolerance of application to handle false positive or false negative matches. False positive denotes that a router/application receives a message that it is not interested in; and false negative denotes that a router/application misses a message that matches its subscriptions. In other words, the system may deliver out-of-interest messages and may fail to deliver some messages of interest. This is due to a router not being aware of the interests of its neighbours and only forwarding messages based on its own interests. Alternatively, appropriate mappings may not have been loaded, so the router does not possess a complete knowledge base to inform publication matching. It should be noted that false positive are usually benign as the effective filtering algorithms of SBPSs can easily filter out irrelevant messages in later hops, whereas false negative can adversely impact application consistency. The possible solution to address these mismatches in SBPSs is that the missed matching messages could be either discarded with warning, or forwarded to the router's neighbours, or sent to a master server that would be responsible for these missed matches. Therefore, the mapping service provision should be dynamically adjusted in accordance with the applications' tolerance to mismatches.

- **Unknown data diversity/dynamism:** the concept "unknown data" in SBPS systems refers to any messages whose ontological terms are not expressed by the semantic model used at a SBPS broker/router. The implication of this factor is captured in several other factors above. In a decentralised large-scale networking environment, as the rate of unknown data occurrence could vary at any moment, an increase in the number of unknown data would increase the publication mismatches rate. In addition, the diversity and dynamics of clients would lead to diverse messages that possibly increase unknown data occurrence rate. It is necessary that the SBPS system would has the capability to adapt to such unknown data changes.

3.3.2.3 Network Environment Related Influential Factors

The environmental conditions in computing networks can be considered as any information that can be used to characterise the situation of a network. The typical examples of environment conditions are the available bandwidth for the networking communication, the temperature condition around the network, and the location of computing nodes and so on. As for event-based middleware, environmental conditions can be scoped as any information which influences the event communication between applications in the environment. The authors in (Mahambre et. al, 2007) stated that an event-based system should be adaptable to the changes of several environmental conditions (i.e., available bandwidth, connections between routers) in order to guarantee the quality of service it offers to the applications. There is no doubt that SBPS systems, as a class of event-based middleware also need to be adaptable to the changes of environmental conditions in order to guarantee its mapping service. However, as this thesis is mainly focusing on how to guarantee semantic interoperability in the event distribution service rather than ensuring quality of service for communication in middleware in general (Mahambre et. al, 2007), the environmental conditions that need to be adapted by SBPSs for achieving efficient semantic interoperability service is thus:

- **Network Scale:** a SBPS network can be deployed in different scales, ranging from a local deployment size, to enterprise size, to internet size. An enterprise network can be flexibly enlarged to a global scale network along with the participation of new SBPS members or the integration of several enterprise-scale SBPS networks. The provision of semantic interoperability for a small deployment network will not be sufficient enough for a large scale network, as the rate of unknown data occurrence would increase with more applications using different ontologies joining the network. Therefore the provision of mapping mechanisms will need to adaptively change in order to meet the new heterogeneous information situation.

- **Available resources in the router**: the incorporation of semantic interoperability mechanisms within the SBPS routers will potentially increase the workload of an individual SBPS router. Therefore, efficiently utilising resources allocated to the routers becomes important. It is desirable to enable the router to adaptively configure its semantic interoperability services along with the variation of resources allocated to the router.

- **Number of mapping files stored in a router:** as mentioned in the semantic model section above (Section 3.3.2.1), the numbers of referenced ontologies in a mapping file have an adverse impact on the reasoning performance of a router, the number of mapping files stored in a router also influence on the configuration of mapping mechanisms in a router.

3.4 Existing Semantic-based Pub/Sub Systems

As there is a limited number of existing Semantic-based Publish/Subscribe systems developed, in this section these systems are briefly described and an analysis of how they can be characterised according to the basic taxonomy identified in Section 3.2 is presented. Additionally, the challenges these systems face to support semantic interoperability are also discussed, as there has been no systems to date that have addressed all requirements of the advanced taxonomy identified in Section 3.3. For convenience a comparison table is provided at the end of this discussion. The name of the systems and reasons for their selection follows:

- The S-ToPSS (Petrovic et. al., 2003; Burcea et. al., 2003) system is a CBN system whose matching algorithm has been extended to support a semantic information model. S-ToPSS also provides a solution of addressing the information heterogeneity problem. This partly addresses the requirements identified in the advanced taxonomy.

- The OPS system (Wang et. al., 2004) is a RDF-based SBPS system that contributes towards the achievement of key requirements (expressiveness and scalability) in the basic taxonomy by using RDF and DAML+OIL techniques to semantically describe events and subscriptions.

- The OBPS system (Skovronski et. al., 2006) is another ontologically based Pub/Sub system. It makes use of the XML language to make event models semantic and SPARQL query as its subscription language in order to address the expressiveness requirement in the basic taxonomy.

- The CQS system (Halaschek-Wiener et. al., 2007) targets the formalisation of a semantically expressive syndication architecture. It aims to offer a high volume, selective content dissemination service on the web. It is a good example of using description logic techniques to express events and subscriptions.

- The Elvin(O) (Keeney et. al., 2006) is a proof-of-concept system implemented by the KBNImpl researchers to investigate the feasibility of knowledge based networking. It is a good example of addressing both key requirements in basic taxonomy and ontological heterogeneity problem in advanced taxonomy.

- The SPS-SP (Chirita et. al., 2004) is a RDF-based SBPS peer-to-peer system for managing arbitrary digital resources. To address expressiveness requirement, it uses a first-order query language to express subscription and uses RDF triples to represents events. It is a good example of addressing scalability, as it uses HyperCup (Schlosser et. al., 2002) semantic topology to build super-peers topology, which contributes to efficient and non-redundant information broadcasting.

- Even though Cashua (Power 2008) itself is not a typical SBPS system, it provides a way of using semantic mappings to resolve heterogeneity between semantic models for applications in CBNs. Therefore it is included for comparison. Cashua is an OWL-based context management system that integrates semantics and Content-Based Networking for context distribution. It uses OWL-based models of contexts for managing heterogeneity and routing context information over a CBN. The CBN is used to route queries and results, but also routes mappings between context models, allowing heterogeneous applications to communicate where an appropriate mapping between models has been provided on the network.
- The KBNImpl system (Lewis et. al., 2006; Keeney et. al., 2008a; Keeney et. al., 2008b) is a good example of Semantic-based Publish/Subscribe system; it addresses the key requirements identified in basic taxonomy. Even though it does not support semantic interoperability, its way of expressing ontological information model (using OWL) and its way of combining semantic technology with CBN (merged into the core of Siena) makes it a good candidate to support the communication and interaction of applications with different ontologies through the integration of an additional semantic mapping mechanism. It should be noted that this system is the one chosen in this thesis for extension. The reasons for choosing KBNImpl is presented in Section 4.3.

In the following subsections, each of the selected systems are described in more detail and analysed with respect to the key features and semantic interoperability presented in section 3.2 and Section 3.3.

3.4.1 Semantic-Toronto Pub/Sub System (S-ToPSS)

S-ToPSS (Petrovic et. al., 2003; Burcea et. al., 2003) offers a CBN network that is targeted at making an existing centralised syntactic matching algorithm semantically-aware whilst keeping the efficiency of current event matching techniques. S-ToPSS proposes a two-level distributed semantic Pub/Sub system. The top-level routers have an upper ontology that contains high level descriptions of ontologies gained from the lower level routers. The lower level router can maintain their own ontology for communication, as heterogeneous ontological information for each application is distributed between multiple routers. These low level routers advertise more general descriptions of the ontologies they hold to higher level routers. Milenko (Milenko et. al., 2003) describes this additional semantic matching mechanism with an employer-employee scenario. The employer is looking for a candidate from a “certain university”, “with a PhD degree”, “with at least 4 years work experience.” This is matched using the semantically enhanced publish/subscribe system, S-ToPSS, in which an employee who has a PhD, from a particular school, with at least four years work experience is matched with the prospective

employer. The match is only made because the semantic system is aware that "school" and "university" have the same semantic meaning, particularly in the North American Educational System. S-ToPSS manipulates subscriptions as they arrive to add semantic synonyms and super-class concepts to the subscription (with support for more general mapping functions). In addition, when events enter the router new synthetic events are created depending on the semantics contained in the event. In this way the semantics stage of subscription merging and publication matching is performed first, outside of the standard matching engine.

Analysis with Respect to Key Features

Referring to expressiveness, as described above, S-ToPSS proposed three extra matching algorithms to provide a semantic event model. It uses the *synonym translate* approach for matching events and subscriptions that are syntactically different but have the same meanings, and uses *hierarchy relate* approach for providing semantic relations between attributes names and values. However, unlike KBNImpl, it neither employs an ontology language such as OWL to express its event model nor uses any concrete ontological operators for matching semantic events with subscriptions.

With regard to the scalability issue, S-ToPSS is only deployed as a single router to provide a centralised information dissemination service, and so there is no need for a routing scheme for routing subscriptions and events. In addition, it adds three extra matching algorithms to semantically enhance the CBN event matching algorithm, but the basic principle of event matching is the same as KBNImpl. However, no evaluation has been published showing how the system minimises the impact of introducing extra algorithms on the performance of event matching in CBN.

Analysis with Respect to Semantic Interoperability

The S-ToPSS makes use of multiple ontologies to allow routers to exchange heterogeneous information. However, it requires that any low-level ontologies used by low-level routers are related to an upper-level ontology owned by the high-level routers. This two-level ontology approach is preferable to a single global ontology as it allows applications to change their ontologies as they evolve. However, any changes to the global ontology would require changes to applications. In addition, S-ToPSS employs the *mapping* algorithm for correlating one or more attribute-value pairs to one or more semantically related attribute-value pairs, however, there is no concrete ontology language used to represent its mapping format. Furthermore, S-ToPSS does not provide a flexible mapping service composed of multiple mapping mechanisms for dealing with different situations. Therefore, it is not able to dynamically configure its semantic interoperability level in order to adapt to the changes of influential factors.

3.4.2 Ontology Pub/Sub System (OPS)

An ontological Pub/Sub system called Ontology-based Pub/Sub (OPS) system is presented in (Wang et. al., 2004) and shares the motivations of KBNImpl in Section 3.4.1 to improve the expressiveness of events and subscriptions within the system. Wang's work is achieved using RDF and DAML+OIL to semantically describe events and subscriptions, both of which are represented as RDF graphs and graph patterns respectively. The application concepts within events are integrated together to form a concept model that is represented as ontology, so OPS can match events with subscriptions both semantically and syntactically. As both subscriptions and events are represented as graphs, if every node and arc in a subscription can be mapped to a corresponding node and arc in the event graph, the subscription is said to match the event.

The OPS system uses an index structure to facilitate its matching algorithm to match events in an efficient way. The main idea of the OPS matching algorithm is that: each RDF graph pattern is decomposed into a set of statement patterns, which are the basic unit of matching. The decomposition process is generally performed in order to avoid unnecessary statement patterns created. After decomposition, an index structure of statements patterns is then built up based on the concept model.

Analysis with Respect to Key Features

With respect to improving expressiveness, the OPS is shown to match events and subscriptions both semantically and syntactically through the use of RDF and DAML+OIL. In order to improve scalability, central to the OPS matching algorithm is a very efficient and scalable index structure based on the complete set of possible statement patterns (decomposed by RDF graph patterns). These are used as the basic unit of matching using AND-OR trees as matching trees, and subsequently avoids the backtracking of the RDF graphs. However, it does not include/provide the ability to perform generic content-based subscriptions. It is important to note that it is desirable to enable a SBPS allowing semantic and syntactic operators to coexist for increasing both expressiveness and flexibility.

Analysis with Respect to Semantic Interoperability Support

OPS does not support flexible and adaptive semantic interoperability, as it makes use of a single common RDF ontology shared among routers and applications for information sharing and discovering. Therefore, publishers and subscribers have to agree to only use this common RDF ontology, which limits their freedom of designing their own ontologies for exchanging information. As OPS is a RDF-based SBPS system, it brings the possibility of incorporating RDF-based ontology mappings with itself. However it is difficult to estimate the feasibility of adding the functionality of semantic interoperability support to OPS system and evaluate the

performance of OPS with respect to delivering heterogeneous information, as the source code has not been made publicly available.

3.4.3 Ontology-based Pub/Sub System (OBPS)

The Ontology-based Pub/Sub System (OBPS) (Skovronski et. al., 2006) is another ontology based publish/subscribe system. This system expresses event models by using the XML language: publications are composed of an XML message where the tags are the names of the ontological classes or property within the ontology and the root node of the notification must contain the name of the ontology to which the notification is destined. To assist in the matching of events to the interested subscriptions, each publisher has its own publisher agent which is responsible for processing published events within the router. Each agent inside the router maintains their own ontological model and stays in scope as long as the publisher continues publishing messages. The interested subscribers register a topic which defines the ontology they are interested in, guaranteeing that all publications for that topic are routed to all interested subscribers. The system uses the SPARQL query language (Kremen et. al., 2008), as its subscription language, allowing the subscribers to easily parse the returned notification message based on their knowledge of which XML tags will be within the notification.

Analysis with Respect to Key Features

Again like OPS, this system does not include capability to perform generic content-based subscriptions. All processing of messages and query analysis is done on the client-side In addition the substantial overhead introduced by SPARQL and the lack of mechanisms to aggregate subscriptions and construct network overlays, means that the overhead and scalability of this system is uncertain.

Analysis with Respect to Semantic Interoperability

The system allows all publishers to register their new publications into a single common ontology, however, it does not consider the substantial overhead that would be introduced by re-reasoning over a newly constructed ontology. Our evaluation work in Chapter 7 has shown that even merging a single individual mapping into the common ontology would result in a considerable amount of overhead. In addition, the system requires all subscribers have to share the common knowledge with the ontology that is composed of all publications. Again, this severely limits the subscribers to have their knowledge bases mirror the domain ontology appropriate for expressing their interests. Furthermore, it does not provide any additional semantic mapping mechanisms to address heterogeneous ontological information challenges.

3.4.4 Continuous Querying Syndication System (CQS)

CQS (Halaschek-Wiener et. al., 2007) targets the formalisation of a syndication architecture that utilises web ontologies and logic-based reasoning for a large volume of selective content dissemination on the web. This aims towards an expressive syndication system, where each of its subscriptions is comprised of a conjunctive ABox (ontology instance) query; while the publication is defined to be composed of a set of ABox assertions. Intuitively, CQS offers only one syndication router which maintains a local knowledge base into which newly published information is integrated. To match newly published information with subscription requests in an efficient and practical manner, the events are matched to subscribers by employing a composition matching algorithm as follows: *information matches* refers to the individuals that are stored in the local knowledge base bound to the variables of a continuous query (subscription), hence the result returned to the interested subscriber is actually the query answer in the local knowledge base rather than the matched publications. Whilst the *publication matches* refers to the collection of publications satisfying subscriptions; the router delivers the minimal sets of publications to the interested subscribers.

Analysis with Respect to Key Features

This system does not include a capability to perform generic content-based subscriptions. Furthermore, it uses ABox queries as a subscription language for querying events that are stored as DL ABox assertions in the DL knowledge base. As a result the expressiveness of subscription language is not as rich as in KBNImpl. As for *scalability*, as it only uses one router to provide centralised knowledge syndication service, the system will be inevitably overloaded as the number of publishers and subscribers publishing information and expressing interests increase. In addition, there are no mechanisms for distributing routers to construct wide-scale syndication networks.

Analysis with Respect to Semantic Interoperability

CQS proposes a generic update function for merging newly published information into the local DL knowledge base, which allows the router to merge a set of new publications into its knowledge base. However, it does not allow for loading other knowledge bases and ontology mappings into the syndication router. The one single fixed knowledge base limits the communications between applications with different application semantic models.

This system does provide a good algorithm to minimise the extra overhead introduced by re-reasoning over the updated knowledge base, when new publications are merged in. An incremental consistency checking algorithm (Halaschek-Wiener et. al., 2006) for incrementally updating tableau completion graphs under syntactic ABox updates, adds the new components

introduced by the update, to a completion graph derived from consistency check prior to the update. Subsequently standard tableau completion rules are re-fired to ensure that the model is complete. This alleviates the workload burden for both matching and reasoning. However, it does not take into account the system reaction to the changes of influential factors in accordance with the requirements of applications and business goals.

3.4.5 Elvin with Ontology-Elvin(O)

Elvin with Ontology, Elvin(O), is a proof-of-concept system implemented by the researchers of KBNImpl to investigate the feasibility of knowledge based networking (Keeney et. al., 2006b). The system uses closed-source Elvin (see Section 2.6.2) CBN as an underlying subscription matching mechanism. In this system ontological synonyms were added to publications at the edge of the network at a generic mapping gateway, allowing subscribers to register subscriptions according to their own ontology, and have the matching publications delivered to them, where the publication may have contained information defined in a different (mapped) ontology. In (Keeney et. al., 2006b), the authors describes one specific application scenario for delivering heterogeneous ontological network management models of managed telecom devices: the mapping file containing mapping relations between "CIM" and "SMI" network management models is applied if needed at the edge of the network. Alongside the semantic subscriptions, this system maintained the very expressive Elvin subscription language thereby supporting semantic and content based subscriptions.

Analysis with Respect to Key Features

Elvin(O) does provides a mechanism to semantically enhance publications and subscriptions. Additionally, with respect to non-ontological issues, the Elvin subscription language is more expressive than that provided by KBNImpl (actually Siena), in particular supporting logical combinatorial operators that support combinations and disjunctions of constraints. However the closed-source nature of the Elvin implementation was prohibitive in terms of enhancing the content-based routing algorithms to include more complex datatypes, such as OWL classes, instances or properties inside the router rather than at the edge of the network. And there is no ontological operators like KBNImpl used, it only support semantically equivalent operator. When a subscription goes in, the ontology model is queried and ontological synonyms are used to synthetically create additional semantically similar syntactic subscriptions in parallel.

To address scalability, Elvin provides a clustering overlay for local area server replication and also offers a federation mechanism for static wide area hierarchical subscription space partitioning. However, there is limited information available about how it forwards subscriptions and publications.

Analysis with Respect to Semantic Interoperability

Elvin(O) provides a mechanism of using OWL mappings to resolve heterogeneity problem derived by different ontological models. It explores the provided mapping file to translate the terms expressed by one ontology into the related terms expressed by another ontology at the edge of the network in a generic mapping gateway. However, as Elvin(O) is only a proof-of-concept system used for evaluating the feasibility of Knowledge-based Networking in telecom network management scenario, its applicability to the other areas needs to be further investigated. Additionally, Elvin(O) provides a fixed semantic mapping service, which indicates that this system is not capable of dynamically adapting to the changes of environmental conditions.

3.4.6 Semantic Pub/Sub with Super-Peers (SPS-SP)

The Semantic Pub/Sub with Super-Peers (SPS-SP) (Chirita et. al., 2004) system is an RDF-based system for managing arbitrary digital resources. Subscription queries are expressed by using a typed first-order RDF query language called "L". A subscription is a conjunction of RDF triples, where a RDF triple is comprised of subject (s), predicate (p), and object (o). The predicate can be the operator of ">" for integer type or "⊇" meaning "contains" for string type. The publication is a pair (T, I), where T is a set of ground atomic formulas of L of forms of the form of RDF Triple (s, p, o) with the same constant s, and I is a client identifier. A publication PUB matches a subscription SUB if the ans(SUB, T) ≠ Ø; where the ans(SUB, T) denote the answer set of SUB when it is evaluated over T. Publications and subscriptions are matched at super-peers and appropriate subscribers are notified. The advertisement of messages is also used to help super-peers to route information efficiently. The SPS-SP system is comprised of peers with the role of publisher or subscriber and a super-peer that offers a more powerful capability than a typical peer. Super-peers are organised within the HyperCup (Schlosser et. al., 2002) semantic topology which takes care of processing publications, advertisement, and subscriptions by way of P2P semantic queries. The HyperCup algorithm is capable of organising super-peers into a binary hypercube that is a type of Cayley graph. The advantage of HyperCup is that it limits the path between any two super-peers to $\log_2 N$, which enables efficient and non-redundant information broadcasting.

Analysis with Respect to Key Features

For improving expressiveness, SPS-SP concentrates on the routing of digital resources expressed in RDF. It uses the RDF language to semantically express its subscriptions and publications. However, again, like several of the systems discussed, this system does not include capability to perform generic content-based subscriptions. As for scalability, it utilises the

HyperCup algorithm to organise super-peers, to broadcast information in an efficient and scalable manner. Additionally, The SPS-SP system uses the selective advertisement forwarding algorithm to selectively broadcast advertisement messages to the super-peers, so that the peers that are willing to submit subscriptions have the sense of which peers contains information that they are interested in.

Analysis with Respect to Semantic Interoperability

SPS-SP provides a communication abstraction, as applications send queries to peers, who then forward those queries on their behalf. However, it does not provide a mechanism for conversion and interpretation between formats used by different applications. Ontologies are used to express the models of application data, but reliance on a shared ontology for routing purposes requires that applications commit as much as possible to that ontology in order to achieve most efficient routing.

3.4.7 Knowledge-based Networking Implementation (KBNImpl)

As briefly introduced in Section 2.8, KBNImpl (Lynch et. al., 2006; Lewis et. al., 2006; Keeney et. al., 2007; Keeney et. al., 2008a; Keeney et. al., 2008b) is an implementation of Knowledge-based Networking based on the Siena CBN. KBNImpl enables the efficient routing of distributed knowledge to, and only to, nodes that have expressed a specific interest in that knowledge.

KBNImpl introduces two extensions to the existing type set and the set of filter operators already provided by the hierarchical Java version of Siena (Carzaniga et. al., 2008) thereby increasing the expressiveness and flexibility of Siena's publications and subscriptions. These extensions add the use of ontological types and operators and support for bags of values. The first extension named ***semantic extension*** provides ontological concepts as an additional message attribute types, onto which subsumption relationship, equivalence, type queries and arbitrary ontological subscription filters can be applied. The second extension named ***bag extension*** provides for a bag type to be used in subscription filters, possibly composed with any of the Siena operators or the ontological operators previously mentioned. As this thesis concentrates on SBPS systems which address the support of distributing semantic heterogeneous knowledge, the ***semantic extension*** is mainly introduced in the following subsections, while the detail of ***bag extension*** can be seen in (Keeney et. al., 2008a).

3.4.7.1 The Semantic Extension

In KBNImpl, each KBNImpl router holds a copy of a shared knowledge base (OWL ontology), within which each ontological class, property and individual used is described and reasoned upon by a specific reasoner embedded in each router. Producers of knowledge express the semantics of their available information based on an ontological representation of that information. Consumers express subscriptions upon that information as simple semantic queries.

Ontological Data Types

In addition to generic data types supported by Siena, KBNImpl extends three new ontological types in order to semantically enhance Siena, which are classes, individuals, and properties. They can be used by any KBNImpl subscription or notification, along-side the standard Siena types and operators. This allows messages to be matched to subscriptions based on extensible type information, which can effectively represent meta-data for the message without having to maintain an ever-growing set of universal attribute names. Instead, a simple set of shared attribute names can be used for a concept type, which uses values from a taxonomy that is maintained, distributed and reasoned over at run-time using existing standardised ontology techniques.

Semantic Operators

The primary contribution of KBNImpl is to semantically enhance the Siena subscription language by adding a set of ontological operators. In (Lynch et. al., 2006; Keeney et. al., 2007) the first class of ontological operators in KBNImpl is given, and they are outlined as follows:

- **LESSSPEC operator:** stands for *less specific* and super-class/property. This relation describes how an ontological entity is more general than another ontological entity. For example, as seen in the Wine ontology, an excerpt of which is shown in Figure 3-10 after it has be reasoned over, the ontological type “wine” is less specific than (subsumes) the type “white wine”.
- **MORESPEC operator:** stands for more specific and sub-class/property. This relation describes how an ontological entity is more specific than another ontological entity. As seen in Figure 3-10, it can be said that “white wine” is more specific than “wine” since “wine” is a superclass of “white wine”.
- **EQUIV operator:** stands for semantic equivalence, referring to the relationship between two ontological types that refer to the same type of entity yet may be different ontological classes. As seen in Figure 3-10, the ontological class “DryWine” has been found to be equivalent to “TableWine”.

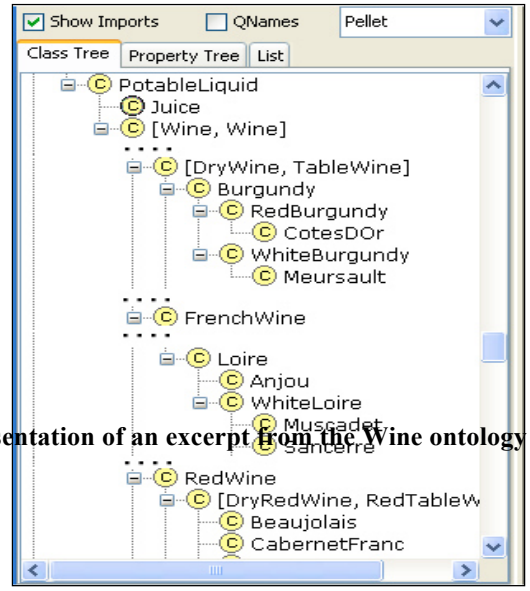


Figure 3-10: A graphical representation of an excerpt from the Wine ontology

Thus given these operators, if an event consumer was interested in receiving events about some ontological entity E, classes equivalent to E, or entities more specific than E, this can be easily achieved by creating a filtering constraint such that the entity described in a field x of the message is subsumed by E, i.e., (x MORESPEC E). This means a subscriber can subscribe to all KBNImpl messages that contain an attribute whose value is a concept more/less specific than the named concept in the subscription.

In (Keeney et. al., 2008a), the second class of ontological operators were developed, and they are briefly introduced as follows:

- **ISA operator**: is used to match an ontological individual/instance against its ontological types/classes. If an individual I is defined as being of an instance of a certain type C then the ISA operator will match the individual I to class C, all classes equivalent to C, and all superclasses of C. This kind of a subscription filter was not previously possible since the EQUIV, MORESPEC, and LESSSPEC operators could only compare classes with classes, properties with properties, and individuals with individuals.

- **IS_NOT_A operator**: is again used to compare an ontological individual/instance against its ontological types. If an individual I is defined as being of an instance of a certain type C then the IS_NOT_A operator will match the individual I to all classes except class C, all classes equivalent to C, and all superclasses of C.

- **ONTPROP operator**: is used to match ontological individuals against each other using any ontological object property. Ontological object properties define named relationships between individuals of two classes.

- **NOT_EQUIV operator**: Since Siena does not support a generic NOT (!) operator, this operator was added for completeness. This operator is the opposite of the ontological EQUIV operator discussed above. It is used to compare classes with classes, properties with properties, and individuals with individuals.

However, these operators are not available in the KBNMap prototype implementation, since it is based on an earlier version.

The Covering Relationships between Semantic Operators

In addition to providing semantic operators, KBNImpl also enhanced and modified the Siena subscription tree structure and subscription forwarding architecture. In order to maintain the partial ordering between subscriptions within the subscription tree structure, in order to achieve subscription covering or aggregation, a set of conservative covering relationships between the new semantic subscription operators was defined.

Consider two filtering constraints ***A*** and ***B***, such that ***A*** is given as (**x** *op* **a**), and ***B*** is given by (**x** *op* **b**), where *op* is one of ontological operators such as EQUIV (equivalent to). The variable **x** is the variable for the field in each notification to be compared to the constant ontology class names **a** or **b**, given in the filter specification. Table 3-1 describes when filter constraint **A** covers filter constraint **B,** i.e., when the set of possible notifications matching filter constraint **A** is a superset of the set of notifications matching filter constraint **B**.

Table 3-1: Covering relations between semantic operators

A	B	A covers B when
x EQUIV a	x EQUIV b	b EQUIV a
x MORESPEC a	x EQUIV b	b MORESPEC a
x LESSSPEC a	x EQUIV b	b LESSSPEC a
x EQUIV a	x MORESPEC b	Never
x MORESPEC a	x MORESPEC b	b MORESPEC a
x LESSSPEC a	x MORESPEC b	Never
x EQUIV a	x LESSSPEC b	Never
x MORESPEC a	x LESSSPEC b	Never
x LESSSPEC a	x LESSSPEC b	b LESSSPEC a
x EQUIV a	x NOT_EQUIV b	Never
x MORESPEC a	x NOT_EQUIV b	Never
x LESSSPEC a	x NOT_EQUIV b	Never
x NOT_EQUIV a	x EQUIV b	Never
x NOT_EQUIV a	x MORESPEC b	Never
x NOT_EQUIV a	x LESSSPEC b	Never
x NOT_EQUIV a	x NOT_EQUIV b	b EQUIV a
x ISA a	x ISA b	b MORESPEC a
x IS_NOT_A a	x ISA b	Never
x ISA a	x IS_NOT_A b	Never
x IS_NOT_A a	x IS_NOT_A b	a MORESPEC b
x $ONTPROP_{prop1}$ a	x $ONTPROP_{prop2}$ b	a EQUIV b AND prop1 EQUIV prop2

3.4.7.2 Analysis

Analysis with Respect to Key Features

In respect of *expressiveness*, to our knowledge, KBNImpl currently is the only existing effort aiming to support both non-ontological and ontological event models and subscriptions, which allows KBNImpl to flexibly and expressively subscribe and publish information in both a syntactic and semantic way. As shown in Chapter 2, there are various type- and topic-/subject-based distributed event-based systems whose popularity has both increased and subsequently decreased over time. Several systems use hierarchical addressing in the organisation of topics using a containment relationship, in a manner that is loosely similar to the taxonomical class hierarchy defined in an ontology. The mechanisms addressed by these systems allows the use of the notion of sub- and super-class subscriptions with regard to the topic hierarchies, which in this respect allow a simple comparison to KBNImpl subsumption operators. However the subscriptions using topic-based hierarchical addressing only operate in an "up/down" subscription creation algorithm. With the use of equivalence and disjoint relationships between classes, the KBNImpl allows a multidimensional type tree to be created in which sub-classes not only point to classes further down the class hierarchy, but can point to other classes in other parts of the ontology, a unique addition to the traditional taxonomical topic-based subscription.

Furthermore, rather than extending the subscription mechanism to support a disjunction of filtering constraints, which would greatly affect the ability to aggregate filters for efficient routing tables, the bag values and operators in KBNImpl support a disjunction of values within a single filtering constraint, with multiple constraints being combined as a conjunction as before. This allows messages to be matched to subscriptions based on extensible type information, which can effectively represent meta-data for the message without having to maintain an ever-growing set of universal attribute names.

As for *scalability*, KBNImpl adopts the hierarchical topology from Siena for configuring its router network topology. In addition, KBNImpl also allows the incorporation of Semantic-based clustering, aiming to provide a network environment in which routing nodes, publishers and subscribers are clustered based on their semantic footprint and interests (Keeney et. al., 2008b). The benefits of this are threefold: First, this reduces the processing time involved in making routing decisions based on the messages content. Its takes fewer hops to get from source to destination, as these are already closely linked based on the likelihood of there being a match between the two. Second, this allows for natural grouping of likeminded publishers and subscribers as seen in traditional web-forums/newsgroups and society in general. Third, it allows certain areas of the network to have specialised sub/super ontologies which do not need to contain the semantics of the whole network, meaning that knowledge base sizes can be

reduced and knowledge base updates can be localised. Clusters can therefore be seen as organic structures in which users and routers join and leave as their own personal interests drift, grow, reform and are refined.

Analysis with Respect to Semantic Interoperability

As the main characteristic of KBNImpl is that the router holds a copy of a shared OWL ontology, it currently requires all the publishers, subscribers and routers of KBNImpl network to use this shared OWL ontology for publishing and subscribing information. This means it is impossible for KBNImpl to allow the publishers and subscribers which own different designed ontologies about the same domain as the shared OWL ontology to exchange information in the network. As the KBNImpl itself does not provide a semantic mapping service, it does not support ontological heterogeneous knowledge delivery.

However, as KBNImpl is an OWL-based SBPS system, it brings the possibility of incorporating OWL-based ontology mappings to integrate individual heterogeneous ontological entities or different ontologies with the shared routing ontology. More detail on the incorporation of ontology mappings in KBNImpl is available in Chapter 4.

3.4.8 Cashua

The Cashua (Power 2008) is an OWL-based context management system that integrates semantics and Content-Based Networking for context distribution. It uses OWL-based models of contexts for managing heterogeneity and routing context information over a CBN. As it is based on Elvin (Section 2.6.2) and so all heterogeneous information is resolved at edge of CBN. The CBN is used to route queries and results, but also routes mappings between context models, allowing heterogeneous applications to communicate where an appropriate mapping between models has been provided on the network. The fundamental component of Cashua system is the Context Service Node (CSN), which is responsible for resolving context queries passed to it by a context-aware application. Both producer and consumer have to register their context models with their CSNs, and the integrator (a type of network administrator for managing heterogeneous context models) has to register mappings with the context producer's CSN. Once a query from consumer's CSN arrives at the producer's CSN, the CSN extracts the query and list of concepts from the subscription, and uses its available ontology and mappings information to translate the concepts in the query into those understood by the context producer. When the result is returned it is translated back into the context model of the consumer, which will be received by the consumer. The result is then sent across the CBN as a notification to the consumer.

Context Service Node

The CSN node functionality is composed of two main parts. The first part consists of the components support for heterogeneity, including its ontology-based context modelling and mapping mechanism. The second part is made up of the components support for dynamism, explaining how Cashua uses the Elvin CBN to route context information and manage the changes of mobile clients in the environment. As this thesis focus on the use of mappings to address heterogeneity, the first functionality part of CSN is only presented here.

The main components of CSN for supporting heterogeneity are presented as follows:

Ontology Repository: stores ontologies supplied to the context service for its local client applications.

Ontology Mapping Repository: stores mappings provided by applications or discovered during the course of querying.

Ontology Registration Interface: allows application to register their context models with the CSN.

Mapping Registration Interface: allows integrator to register mappings with the CSN.

Query Engine: supports heterogeneity by translating queries and results between models. It discovers models and mappings from the repositories and uses this information to perform translation of queries sent to it.

Analysis with Respect to Key Features

The Cashua system uses OWL ontologies for modelling context models of context applications and makes use of a CBN for context dissemination. The use of Context Service Nodes as gateway bridges between applications and CBN can manage the variable availability of clients and also can upgrade them freely without requiring changes to clients, allowing for evolution of the system. However, as a context management system, Cashua's work does not focus on semantic enhancement and modification of content-based networking. So there are no ontological data types available inside the CBN and it does not semantically enhance CBN subscription languages. In addition, the semantics of context models can be only realised at the edge of network rather than inside network. In other words, there is no semantic model available in routers. As for scalability, its Elvin basis enables it to be applied to large-scale context distribution environments.

Analysis with Respect to Semantic Interoperability

Cashua allows for the diverse nature of ubiquitous environments in catering for heterogeneity between applications with different context models, so it provides mappings between application models to support this heterogeneity. The design of CSN provides good instruction for the design of a semantic interoperability service in SBPS system, since CSN nodes make use of ontology mappings when applications have chosen different context models, allowing each application to communicate the context system using its own context model. In the Cashua system, the formats of mappings are expressed in OWL and so provides formal and expressive semantics. In addition, Cashua also uses CBN to distribute ontology mappings to the applications that require those mappings, allowing for the evolution of the system without requiring the services of an integrator.

One of the issues with Cashua's approach is the mappings are only used at CSN nodes at edge of CBN, rather than using mappings in CBN routers. This is the drawback for SBPS networks, as the SBPS routers storing a shared semantic model need to allow for integrating new sub semantic models and resolve heterogeneous information locally. In addition, Cashua only considers using mappings in context publisher's CSN side to translate the concepts (that are not expressed by the producer's context model) in the query it received into those understood by the context publisher, without considering heterogeneous publications. This is also necessary in the SBPS environments, as the SBPS systems need to find solutions to resolve both heterogeneous subscriptions and publications. In addition, Cashua only considers concept hierarchies, composed of superclass/subclass mapping relations, and equivalence relations between concepts without allowing for further more complex mappings such as property, individual relations, and individuals with classes.

3.5 Comparative Analysis

In this section, a comparative analysis of the selected SBPSs is presented. Section 3.5.1 presents a classification of SBPSs, on the basis of their key features, while another classification of SBPSs according to their support for semantic interoperability is presented in Section 3.5.2. Table 3-2 classifies existing efforts based on the functional characteristics that mainly focus on the expressiveness and scalability for architecting such a system. Table 3-3 provides insight into the current level of support provided by existing SBPSs to semantic interoperability.

3.5.1 Comparison of SBPSs with respect to Key Features

In Section 3.2.2 a number of key features reflecting *expressiveness* and *scalability* that a SBPS should have were introduced, and in Section 3.4 the selected state of the art systems were

analysed with respect to these features. This section provides a summary of the analysis through the introduction of a comparison table based on the key features.

For convenience, the following summarises the features detailed in Section 3.2.2 and introduced the headings used for comparing those SBPSs described in Section 3.4:

1. The SBPS system should use semantic technologies to enhance its event model semantically, so that the events can be distributed across a network based on semantics of the data and associated meta-data contained in the events. Which semantic languages used to express event model and how to form the event model could reflect the expressiveness and flexibility level of a SBPS system. This feature is examined under the "Event Model" heading of the comparison table.
2. Related to Event model, how the system uses semantic technologies to enhance its subscription language is also vital in improving the expressiveness and flexibility of a SBPS system. This includes what type of ontologies we used to express subscription and what type of operators used for filtering events. This feature is examined under the "Subscription Language" heading of the comparison table.
3. The scalability of a SBPS system is strongly related to the network topology used. A well designed network topology could allow applications/routers to communicate with each other in a reasonable and effective manner. This feature is examined under the "Overlay Type" heading of the comparison table.
4. The scalability of a SBPS system is also reflected by the routing schemes used for forwarding subscriptions and publications. A scalable routing scheme can enable a system to route subscriptions and publications across a large scale network environment to the destination in an efficient manner. This feature is examined under the "Routing Scheme" heading of the comparison table.
5. The scalability of a SBPS system is also reflected by the event matching algorithm employed for matching subscriptions with events. As the event matching algorithm employed has significant impact on the performance and scalability of a SBPS system, a well designed matching algorithm could enable a system to match events in an efficient way. This feature is examined under the "Event Matching" heading of the comparison table.

The comparison table is presented in Table 3-2.

Table 3-2: Comparison of Selected SOA systems wrt key features

System	Event Model	Subscription Language		Overlay Type	Routing Scheme		Event Matching
	Data Expression: *The format of knowledge within the network.*	**Subscription Expression:** *How user subscriptions are formed.*	**Subscription Operators:** *The operators used in the user subscriptions.*	*The overlay network design and deployment*	**Sub Forwarding:** *The method in which a router forwards subscriptions to other nodes in the network*	**Event Forwarding:** *How the pubs are forwarded to interested subs*	**Pub to Sub Matching:** *How incoming pubs are matched to stored subscriptions.*
S-ToPSS	***Basic Format:*** [attribute_name, value] ***Non-ontological_value***: Integer, Long, Double, Boolean, String ***Ontological_value:*** uses three extra matching algorithms to make a non-ontological event semantically.	***Basic Format:*** [attribute, operator, value] ***Note:*** Similar to ToPSS	***Crisp***: >, <, =, ≤, ≥, etc. ***Approximate***: (i.e., ~= modelling approximate equality) ***Probability***: (i.e., modelling a random event)	***Centralised:*** The system is deployed as an information dissemination service ***Two level layers:*** top routers owns a general ont, low routers have their own onts	***None:*** it is a centralised, therefore no routing	***None:*** it is a centralised, therefore no publication forwarding	***Matching tree:*** Event iterates through sub tree using breadth first search. ***For semantic matching:*** Existing matching assisted with***:*** -Synonyms matching -Concept matching -Hierarchy matching
OPS	***Basic Format:*** [attribute_name, value] ***Non-ontological_value***: Integer, Long, Double, Boolean, String ***Ontological:*** RDF graph	***Basic Format:*** RDF graph pattern (subject, object, meta-statement, [filter_func(object)])	***Note***: When the object is a variable and its type is literal, it includes *filter_func(object) to refine value* ***filter:*** >, <, =, ≤, ≥, etc	***Hierarchical structure*** ***Peer-to-peer*****:** (limited information available)	***Aggregation method:*** Aggregation by graph merging. ***Forwarding method****:* the most general subscriptions are sent to parent servers.	***Reverse path forwarding:*** Follows the reverse path of subs.	***AND-OR matching tree:*** Node and arc in sub graph must be mapped to node and arc in the event graph.
OBPS	***Basic Format:*** [attribute_name, value] ***Non-ontological_value:*** Integer, Long, Double, Boolean, String ***Ontological_value:*** XML-based	***Basic Format:*** SPARQL query.	***No operators:*** subscription is represented by SPARQL query.	***Centralised***: a single router	***None:*** it is a centralised, therefore no routing	***None:*** it is a centralised, therefore no publication forwarding	***Ontological topic based matching using a SPARQL engine.***
CQS	***Basic Format:*** (***α, t, p***) where ***α*** is a set of DL ABox assertions, ***t*** is time unit and ***p*** is the client identifier	***Basic Format:*** (***Q, t***) where Q is a Continuous conjunctive ABox query with respect to a DL KB and t is time unit	***No operators:*** Subscription is represented by ABox query.	***Centralised:*** a single syndication router	***None:*** it is a centralised, therefore no routing	***None:*** it is a centralised, therefore no publication forwarding	***Composition matching algorithm:*** -Information matches. -Publication matches. -Composite matches.
Elvin(O)	***Basic Format:*** [attribute_name, value] ***Non-ontological_value:*** Integer, Long, Double, Boolean, String.	***Basic Format:*** [Attribute, operator, value]	***Non-ontological***: >, <, =, etc. ***Logic***:| |, &&, !, etc.	***Clustering:*** local area server replication for fault tolerance ***Federation:*** static wide area hierarchical subscription space partitioning	***Forwarding method:*** Subscription kept locally, possibly broadcast within clusters (limited information available)	***Hybrid:*** Between federations publications are routed according to subscription space partitioning, possibly broadcast within clusters (limited information)	***Non-ontological:*** unknown, internal to Elvin ***String based label Ontological matching****:* OWL labels and synonyms are merged into/from string labels at the network edge
SPS-P2P	***Basic Format:*** (***T***, ***I***) where ***T*** is a set of ***t*(*s, p, o*)** and ***I*** is a client identifier ***Ontological_value***: RDF Event Model	***Basic Format:*(*s, p, o*)** where ***s*** is a variable, ***p*** is constant, ***o*** is distinct variable ***Note:*** Datalog-inspired RDF Query	***Binary:*** >, <, =, ≤, ≥, etc. ***Subsumes:*** ⊇	***Peer-to-Peer*** ***Note:*** Super-peer arranged in HyperCup topology.	***Aggregation method:*** Aggregation by subscription covering. ***Forwarding method:*** the most general subscriptions are broadcasted to all neighbours	***Reverse path forwarding:*** Follows the reverse path of subs (similar to Siena).	***Matching tree:*** The event iterates through the subscription tree by using breadth first search algorithm (similar to Siena).
KBNImpl	***Basic Format:*** [attribute_name, value] ***Non-ontological_value:*** Integer, Long, Double, Boolean, String ***Ontological_value:*** OWL Classes, Properties, and Instances.	***Basic Format:*** [attribute, operator, value] ***Note***: Similar to Siena	>, <, =, ≤, ≥, etc. ***Semantic:*** equivalent, subsumes, subsumed by, instance_of, object property ***Bag:*** sub/ superbag	***Hierarchical structure*** ***Semantic-based cluster***	***Sub Aggregation method:*** aggregated by subscription covering. ***Forwarding method****:* The most general subscriptions are sent to hierarchical master servers.	***Reverse path forwarding:*** follows the reverse path of subs.	***Matching tree:*** Event iterates through sub tree using breadth first search.
Cashua	***Basic Format:***[attribute_name, value] ***Non-ontological_value:*** Integer, Long, Double, Boolean, String. ***Ontological_value:*** OWL Classes,	***Basic Format:*** [Attribute, operator, value]	>, <, =, etc. ***Logic***: | |, &&, !, etc. ***Semantic:*** equiv, subsumes, subsumed by classes	***Any generic CBN-based topologies*** ***Note:*** Cashua could make use of any CBN	***Any generic CBN-based sub forwarding*** ***Note:*** Cashua could make use of any CBN	***Any generic CBN-based event forwarding*** ***Note:*** Cashua could make use of any CBN	***Any generic CBN-based event-matching*** ***Note:*** Cashua could make use of any CBN

From Table 3-2 it can be stated that KBNImpl, OPS, CQS, SPS-P2P, and Cashua enhance their event model semantically, as they are using standardised OWL or RDF ontologies to express event model. Even though not utilising semantic web technologies, the event representation of OBPS is also reasonable, as it uses XML tags to represents classes, properties, and individuals of event model. Finally, S-ToPSS is the only system using its own event representation to express events semantically, rather than using any popular web languages. Therefore, the level of semantics of its event model is hard to determine. KBNImpl is the only existing research in supporting both non-ontological and ontological subscription languages by using OWL, which means it could flexibly expressively subscribe to information in a syntactical and semantic manner, while the other SBPS systems do not include/provide the ability to perform generic content-based subscriptions. Additionally, KBNImpl is the only platform to use ontological operators for filtering ontological values of events. S-ToPSS uses two matching algorithms to enhance subscriptions semantically, however, it can be argued that it is not as flexible and expressive as KBNImpl, as there is no standardised ontology language being used. OPS and SPS-P2P use RDF to represent subscription, but it does not support generic content-based subscriptions. OBPS uses a SPARQL as its subscription language. Again this system does not include capability to perform generic content-based subscriptions. CQS uses only ABox query for individuals in the subscriptions, which determines that it lacks capability to handle with classes and properties. Elvin(O) simply makes effort of converting semantic event matching into synonym-based syntactic matching at the generic gateway, due to the closed-source nature of Elvin. Finally Cashua only support OWL classes within subscriptions, compared to KBNImpl which supports classes, property and individual types.

Different systems utilise various types of network topologies to deploy the routers, publishers and subscribers. KBNImpl basically adopts the Siena hierarchical topology to deploy the routers. Moreover, it also describes a semantic-based clustering mechanism for deployment. The major benefit of this method is that it takes fewer hops to deliver messages from source to destination. A centralised overlay design is used in S-ToPSS, OBPS, and CQS systems by providing one router for serving various applications, but it will be inefficient to deploy such a system in a large-scale environment, as the performance of the system will be severely degraded when serving a large number of applications. As for OPS, it adopts both hierarchical and P2P mechanisms for organising its nodes. The nodes in Elvin(O) can be organised in clusters (inherited from Elvin), where local area servers can be replicated for fault tolerance. Furthermore, it also offers a federation method for static wide area hierarchical subscription space partition. Finally, SPS-P2P makes use of P2P technology to achieve scalability. Particularly, the concept of super-peers is introduced for delivering information efficiently.

With respect to routing schemes, KBNImpl, OPS, SPS-P2P make use of similar subscription aggregation algorithms as Siena and reverse path method to optimise subscription and event forwarding. The main advantage of this is that it alleviates the workload burden of routers by decreasing the numbers of messages distributed in the network. As S-ToPSS, OBPS, CQS are centralised systems, there is no routing scheme available. As for Elvin(O), the subscriptions are kept and broadcast locally within the cluster organising the subscriptions with similar meanings, while the publications are routed between federations according to subscription partitioning. Considering the combination of topology mechanism and routing scheme employed, the main drawback of Elvin(O) is that due to its static nature, it would be too difficult to restructure the clusters once the semantic meaning of clusters are changed.

Finally, KBNImpl, S-ToPSS, SPS-P2P systems make use of a matching tree algorithm approach that is considered mature in CBN to match and filter events in an efficient way. OPS also offers a efficient "AND-OR" matching tree algorithm for matching events. OBPS uses a SPARQL engine to provide ontological topic based matching, but the substantial overhead introduced by SPARQL, and the lack of mechanisms to aggregate subscriptions and construct network overlays, means that the overhead and scalability of this system is uncertain. In addition, CQS provides a composition method for event matching in order to enhance accuracy and efficiency of current matching algorithm. Finally, due to the closed-source code nature of Elvin, the internal matching algorithm is unknown. Instead, the researchers of Elvin (O) used String-based ontological label matching to match events at the edge of network, but it will inevitably decrease the efficiency of the matching.

In summary, KBNImpl exhibits a good capability with respect to both expressiveness and scalability. The OPS and SPS-P2P also address the key requirements well, but the only disadvantage is that they do not provide generic CBN based subscription language, resulting in a less flexible system than KBNImpl.

3.5.2 Comparison of SBPSs with respect to Semantic Interoperability Support

In Section 3.3, the advanced taxonomy with respect to adaptive semantic interoperability support in SBPS systems was introduced. Particularly, two major requirements with a number of criteria in the context of adaptively and efficiently distributing heterogeneous information were identified. The first requirement is that SBPS systems would make use of ontology mappings to achieve semantic interoperability between heterogeneous semantic models while the second requirement is that the semantic interoperability service provided by the SBPS systems would adapt to changes of ontology, application and environment conditions. In addition, in Section 3.4 the selected state of the art systems were also analysed with respect to

these requirements and criteria. This section provides a summary of the analysis with a comparison table (Table 3-3) based on the requirements.

For convenience, the following summarises the requirements detailed in Section 3.3 and introduced the headings used for comparing those SBPSs described in Section 3.4:

1. The SBPS system should make use of semantic mapping techniques to overcome heterogeneity between applications using multiple semantic models. This requirement is examined under the "Semantic Mapping Service" heading of the comparison table. Allied to this requirement a number of criteria are identified:

    a) The mappings used to relate heterogeneous semantic models need to be expressed by a specific language. This requirement is examined under the "the way to express mapping" subheading of the comparison table.

    b) As various ways of using mappings existing, it is worthwhile to know which approach is taken by each SBPS to use mappings. This criteria is examined under the "The way to use mapping" subheading of the comparison table.

    c) Additionally where to use mappings could also characterise the mapping service of a SBPS system. This criteria is examined under the "Trigger Place" subheading of the comparison table.

    d) A SBPS system should be flexible in supporting semantic interoperability to face different situations and requirements of applications and business goals. This criteria is examined under the "Flexibility" subheading of the comparison table.

2. The SBPS system should provide a mechanism for dynamically configuring its semantic interoperability support, in order to adapt to the changes of influential factors identified in Section 3.3.2. This requirement is examined under the "Adaptive Configuration of the mapping service" heading of the comparison table. The influential factors are categorised into three major types:

    a) Semantic model related: it refers to possible changes in size, DL expressivity of a semantic model and variations in the number of mapping files

    b) Application related: it refers to possible changes in message rate, clients mobility and end users' tolerance level.

    c) Network environment related: it refers to possible changes in the resources allocated to the router and the network scale.

The comparison table is presented in Table 3-3.

Table 3-3: Comparison of Selected SOA systems wrt Semantic Interoperability

	Semantic Mapping Service				Adaptive Configuration of the mapping service
System	**The way to express mapping:** *OWL, RDF or other typed ontology/language*	**The way to use mapping:** *mapping, merging or translating, or other ways*	**Trigger Place:** *where the mappings are used*	**Flexibility:** *have a single or multiple mapping mechanism*	*how the system dynamically adapts to the changes of three influential factors(**semantic model, application, network environment**)*
S-ToPSS	No concrete ontology language is used. ***note:*** Uses mapping function to correlate different attribute value pairs	Mapping: map different attribute value pairs.	**within the network:** inside router When the router execute event matching	No mapping mechanisms are provided.	Does not have the capability to adapt to the changes of influential factors
OPS	Does not have mappings ***note***: all routers and applications use a RDF ontology	Limited to **merge** new information into the RDF model but not use mappings	When the router gets new information	No mapping mechanisms are provided.	Does not have the capability to adapt to the changes of influential factors
OBPS	Does not have mappings ***note***: all routers and clients have the same ontology	No mappings are provided.	No mappings are provided.	No mapping mechanisms are provided.	Does not have the capability to adapt to the changes of influential factors
CQS	Does not have mappings ***note***: A single local Knowledge base (KB) in a centralised router	Supports **merging** new publications to into the KB.	**within the network:** inside router When the router meet new pub	No mapping mechanisms are provided.	Does not have the capability to adapt to the changes of influential factors
Elvin(O)	OWL language Comment: a OWL mapping file for CIM and SMI is provided	Uses OWL mappings to **translate** CMI/SMI into SMI/CMI format	**Edge of network:** triggers at generic gateway when the subscriptions are sent into the publisher's side	Provide one mapping mechanism	Does not have the capability to adapt to the changes of influential factors
SPS-P2P	No mappings are provided.***note:*** all routers and applications have the same and common ontology	No mappings are provided.	No mappings are provided.	No mapping mechanisms are provided.	Does not have the capability to adapt to the changes of adaptation triggers
KBNImpl	No mappings are provided.***note:*** all routers and applications have the same and common ontology	No mappings are provided.	No mappings are provided.	No mapping mechanisms are provided.	Does not have the capability to adapt to the changes of key triggers
Cashua	OWL language Comment: mappings related to different models are expressed in OWL	Uses OWL mappings to translate different context models	**Edge of network:** triggers at CSNs when clients receiving different ontological model.	Automatically distributing mappings	Does not have the capability to adapt to the changes of adaptation triggers

As can be seen from Table 3-3, Elvin(O) and Cashua are the only attempts to make use of semantic mapping techniques to resolve heterogeneity problem derived from applications using different semantic models. The mappings between the semantic models are expressed in OWL. Additionally, they adopt translation methods to translate the terms expressed by one model into the related terms expressed by the other semantic models through exploring the appropriate mapping file at the edge of network in a generic mapping gateway. However, they do not offer other semantic mapping mechanisms in order to meet different situations. In addition, due to the closed-source nature of Elvin, the mapping service is limited to serve at

the edge of the system rather than inside the core of system. S-ToPSS uses its mapping function algorithm to correlate different attribute-value pairs that are semantically equivalent. However, the mapping relations between different terms are created at network runtime rather than design time, which means the accuracy of mappings between different terms remains uncertain, as it is impossible to evaluate the quality of generated mappings at runtime. Furthermore, the performance of the system will be inevitably influenced, as the system requires extra-processing to generate mappings. CQS and OPS could provide limited semantic interoperability service through allowing new publications to be merged into the common knowledge base/RDF ontology, however, the reliance on a unique semantic model still limits communication between the applications with other semantic models. Additionally, CQS employs a consistency checking algorithm to reduce the overhead of reasoning task for updating Knowledge Base, so a slight change in semantic model will inevitably lead the entire ontology to be re-reasoned, while OPS does not provide any algorithms or mechanisms to reduce the expensive overhead consumed for reasoning. As for KBNImpl, OBPS, SPS-SP, none of them make efforts to support heterogeneous information distribution among applications with multiple semantic models. The classification in Table 3-3 shows that none of existing SBPSs supports the capability at runtime to adapt to changes of the semantic model, application, or environment. Even though Cashua supports automatically distributing mappings across the network to where they are needed, it does not consider how factors such as constantly changing environmental conditions would change the requirements of a router or an application for the mappings.

In summary, none of the systems reviewed, which cover the state of the art in Semantic-based Pub/Sub systems, address both key requirements or meet all criteria envisaged for a system that could distribute heterogeneous ontological information in an ontology, application, environment appropriate manner.

3.6 Key Challenges of SBPS Systems

Based on the analysis of the state of the art SBPS systems, it is clear that support for multiple semantic models and adaptation to influential factors are two key challenges for existing SBPS systems to efficiently distribute information in decentralised heterogeneous environments.

Challenge 1: Support for Multiple Semantic Models

Form Section 3.5.2, it can be concluded that providing semantic interoperability support in SBPSs is still at an early stage, as the majority of infrastructures do not fully support flexible

and adaptive semantic interoperability. In other words, they do not allow multiple diverse semantic models to co-exist among applications and routers. This limitation raises the doubt about the feasibility of SBPS systems when they are applied to a large-scale environment. It is the observation of the author that it is impractical to envision decoupled heterogeneous applications exchanging knowledge while requiring a centralised and agreed common semantic model in a highly dynamic and decentralised environment. Thus the first challenge is to provide the SBPSs with a way of flexibly integrating or merging multiple diverse semantic models used by different applications.

Challenge 2: Adaptation to Changes

The second challenge is how to enable SBPS systems to dynamically configure their semantic interoperability level to adapt to changes of semantic model, application, and environmental related influential factors. Within this challenge lies issues with how to identify the key factors that have most significant impact on a particular SBPS from the numbers of influential factors that are explored in this chapter and how to enable specific systems to be adaptable to the changes of these key factors identified to improve their semantic mapping services.

3.7 Conclusion

The basic taxonomy identified provides a framework to evaluate and compare Semantic-based Pub/Sub systems, particularly with respect to semantic enhancement of their event model and subscriptions. The second advanced taxonomy can be used to evaluate and compare SBPS systems with respect to adaptive semantic interoperability for distributing heterogeneous information. The comparative analysis of existing SBPS systems demonstrated that the support for multiple semantic models and the provision of adaptive semantic interoperability service are key challenges for SBPS systems. In Chapter 4, the design of an adaptive semantic mapping serviceKBNMap is presented to enable SBPS systems to address the key challenges identified.

4 AN ADAPTIVE SEMANTIC MAPPING SERVICE

In Chapter 3, the existing SBPS systems are compared with respect to key features and semantic interoperability, and the key challenges of existing SBPS systems summarised. This chapter describes the design of the KBNMap system to address the key challenges, especially the components of KBNMap supporting heterogeneity.

4.1 Introduction

This chapter describes an adaptive semantic mapping service for SBPS that has been designed. This design addresses the key challenges identified in Section 3.6 for supporting multiple models and supporting adaptation in the face of dynamic changes in the influential factors identified in Section 3.3.2..

Section 4.2 presents the design of a novel mapping service to address the key challenges. Then the overall extended SBPS with an adaptive semantic mapping service is set out in Section 4.3. This section also describes how to extend an existing SBPS (KBNImpl) infrastructure to incorporate the novel semantic mapping service and the concrete mapping strategies comprising the service. Section 4.4 gives an example of a heterogeneous subscription and notification resolution to illustrate how the semantic mapping enhanced KBNImpl router uses the proposed adaptive mapping service to address heterogeneity. Finally, an insightful discussion of the designed mapping service's strengths, weakness, applicability to the other SBPS systems, and difficulties involved is presented in Section 4.5.

4.2 Design to Address Key Challenges

This section presents a high level discussion of the design created to address two key challenges identified in Section 3.6. They are "Support for multiple semantic models" and "Adaptation to changes".

4.2.1 Addressing "Support for Multiple Semantic Models"

To address the first key challenge of "support for multiple semantic models", the SBPS infrastructure must allow multiple and diverse ontologies to coexist in the network. Instead of relying on a common fixed semantic model, a solution for SBPS should support the exchange

of heterogeneous information as needed, thus allowing semantic interoperability between different information models that are semantically related but heterogeneously instantiated. - It is proposed to make use of ontology mappings within the SBPS routers . This will allow applications that subscribe to information according to one ontology to receive information published according to a different ontology, where mappings between the ontologies exist. This approach is novel in making use of ontology mappings inside the network. The advantage of this approach is that it enables an SBPS system to merge other semantic models or new ontological terms (i.e., concepts, properties, instances) into its semantic model used for routing through the use of ontology mappings. A semantic mapping service can then be used at runtime to allow the router to route messages containing heterogeneous information to recipients who expressed specific interest in that information. Additionally, it allows applications to connect to the SBPS network directly without the need for custom built gateways.

However, although this feature lowers the barrier for participation by applications in any particular SBPS, it will potentially increase the workload of an individual SBPS router. The assumption in this research is that the drawback of the extra processing is far outweighed by the benefits that are brought in enabling semantic interoperability between applications. However, it is still important to embed this type of mapping service into the SBPS routers in an efficient and optimised manner, so that the extra processing can be managed to a level that is acceptable. To achieve this, the novel mapping service is designed to such that a number of different mapping strategies to perform this searching and merging of mappings, which is executed in an efficient manner. The evaluation results in Chapter 7 demonstrates that the overhead introduced by the extra processing in the KBNImpl router due to the mapping service is sufficiently small to be considered acceptable. In addition, the semantic mapping service can be shown to be flexible enough to handle different situations and requirements of applications.

4.2.2 Addressing "Adaptation to Changes"

To address the second key challenge discussed in Section 3.6, according to the influential factors identified in Section 3.3.2, first a set of key influential factors with respect to semantic interoperability are identified as a baseline for dynamically configuring the semantic mapping service in SBPS networks. As no two network deployments are exactly the same – even in the same domain, different requirements for semantic interoperability can exist. Thus the configuration of a semantic mapping service specified in an individual SBPS router must take into account both the typical characteristics of the entire SBPS network and the characteristics at the individual router. To address the challenge, it is proposed that the configuration and

selection of mapping strategies in each individual KBNMap router be designed based on the combination of characteristics of the individual router's environmental states (i.e., its routing ontology characteristics, the number of available mapping files stored in its router, the subscription/publication rate it receive, the amount of resource allocated) and the entire KBNImpl network's characteristics (i.e., the network scale and user-defined tolerance capability). The research reported upon in Section 3.3.2 explored many possible factors that might influence configuration of a mapping service for generic SBPS systems. However, it is necessary to identify the key factors that have significant impact on the selection of mapping strategies.

The second key challenge also states that SBPS systems should have the ability to self-configure the semantic mapping service and dynamically adapt to changing states of key influential factors identified. To address this aspect, the mapping enhanced KBNMap routers need an adaptive mechanism for selecting mapping strategies, so that the behaviours of routers can respond to the dynamics of key factors identified.

The process of identifying key factors and the components of KBNMap supporting adaptive selection of mapping strategies are discussed in detail in the next chapter.

4.3 Semantic Mapping Enhanced KBNImpl Architecture (KBNMap)

Large scale decentralised networking environments, such as those described in Section 1.1, will naturally contain applications produced by different developers for different domains, which will nevertheless wish to communicate to share information with each other, possibly over the same SBPS deployment. This requires the semantic interoperability not just between the semantic models used by the applications but also requires interoperability between applications' and routers' semantic models.

4.3.1 High Level Architecture of Extended SBPS

In order to cater for the heterogeneous semantic models of different applications/routers, the design of the adaptive semantic mapping service for SBPS requires a number of elements. Figure 4-1 presents a high level architecture of the semantic mapping enhanced SBPS router integrated with the adaptive semantic mapping service. It should be noted that the shaded parts in the Figure are the SBPS's inherent functionality components. The first of the newly added components is a **semantic model registration interface** that the network administrator could use to register semantic models with the SBPS router. The semantic models include both the semantic models used for routing and the semantic mappings for heterogeneity. A

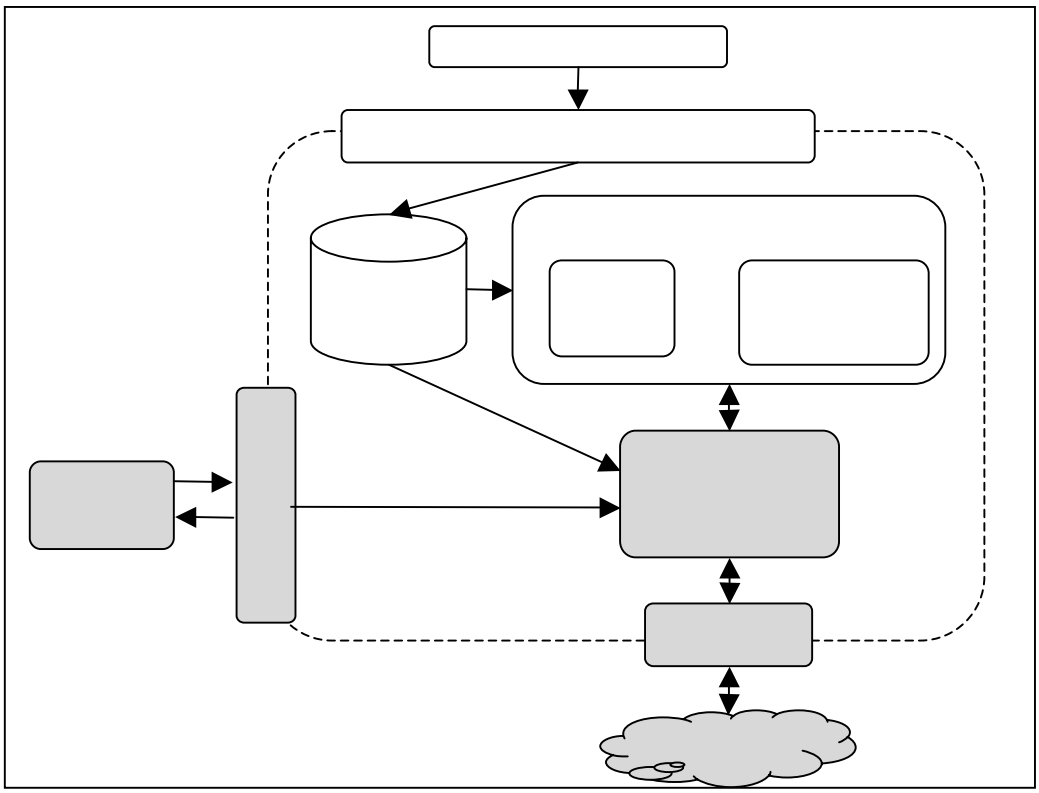
Figure 4-1: A high level architecture of the SBPS router extended with the Adaptive Semantic Mapping Service

semantic model repository is used to store registered semantic models and semantic mappings.

The heart of the extended function block in the SBPS router is the **adaptive semantic mapping service** block, which is responsible for resolving heterogeneous information the router receives, merging different semantic models through the use of semantic mappings (**semantic mapping strategies** in Figure 4-1), and configuring the semantic mapping service to cater for the dynamics of influential factors identified in Section 3.3.2 (**mapping strategies selection mechanism** in Figure 4-1).

Generally, the **application interface** is used for the applications to communicate with the SBPS, especially for registration and subscription/publication resolution. The SBPS routers deal with two types of client: **local** and **remote**. **Local** clients are the applications that directly register with the SBPS router in order to subscribe to information and also to make their own information available. **Remote** clients are applications are similar to local clients except they communicate over a network interface.

The core part of the SBPS system is the **routing/matching engine**, which is responsible for both resolving subscriptions and publications on behalf of local and remote applications, and routing information to the other routers/remote clients. The **routing/matching engine** uses a **network communication** module to manage its communication with other routers.



The following subsections describe how the architecture of a particular SBPS system has been extended with the proposed adaptive semantic mapping service illustrated in Figure 4-1 in order to support the heterogeneity of semantic models between applications/routers and the delivery of heterogeneous information within the network.

In order to evaluate the feasibility of the proposed semantic mapping approach, a concrete SBPS system was selected to integrate the adaptive semantic mapping service into. The selection of an appropriate system was based on the taxonomies identified in Chapter 3. The selected SBPS system should be expressive and scalable enough from key features perspective. Although the state of the art systems are not capable of using semantic mappings, the selected system should be potentially compatible with semantic mappings from semantic interoperability perspective.

For this thesis an implementation of KBNImpl, in particular the version described in (Keeney et. al., 2006c; Keeney et. al., 2007), was extended with features to support semantic interoperability. The primary rationale for this choice was selection criteria above; availability of the source code; and access to the key developers of the platform (within TCD). However, the choice was also made as the KBNImpl has several advantages over other systems that are identified in Table 3-2 of Section 3.5.1:

1. First, the KBNImpl system is an OWL-based SBPS system which uses OWL ontologies to represent event models and subscriptions. The formal and decidable semantics of OWL ontologies allow more expressivity than other languages such as XML and RDF. Additionally OWL's level of formality also allows for the checking of ontologies in order to determine their correctness, so that KBNImpl routers can validate the correctness of ontologies provided by the applications.

2. Second, the ontological operators used by KBNImpl provide a more expressive set of features for describing characteristics of the domain of interests in subscriptions. Furthermore, the KBNImpl also has the capability of supporting generic-content based data types, which makes the KBNImpl more flexible and expressive compared with other SBPS systems.

3. Third, according to the OWL-based nature of KBNImpl, the proposed adaptive semantic mapping service can make use of the OWL-based semantic mappings to semantically enhance the KBNImpl systemKBNImpl. As discussed in Section 2.3.2.3, the use of OWL mappings is an ideal choice in the designed semantic mapping service, due to its open-source nature, standardisation and direct support for semantic relations.

4.3.2 Overall Architecture of KBNMap

The overall design of the extended KBNMap architecture is shown in Figure 4-2. The modules outside the dashed line are external to the KBNMap router, but are drawn to show their interaction (maybe local or remote) with the KBNMap router. The modules that intersect with the dashed line are those that interface with the external modules. It can be observed that each publisher and subscriber has its own semantic model for expressing information respectively. The administrator is responsible for injecting both new semantic models and semantic mappings into the router as required.

The design of the KBNMap architecture for semantic interoperability support required a number of abstract elements and these are shown in Figure 4-2, and the interaction taking place in Figure 4-2 is described in Section 4.3.3 and Section 4.3.4. The *semantic model store* and *semantic mapping store* provide the functionality of **semantic model repository** of Figure 4-1, while the *model and mapping registration interface* provide the **semantic model registration interface** functionality of Figure 4-1. Finally as discussed in Section 4.3.1, the *adaptive mapping service block* is the core part of KBNMap to provide functionalities for resolving heterogeneous information in an efficient and adaptive manner. The detail of these components is described below:

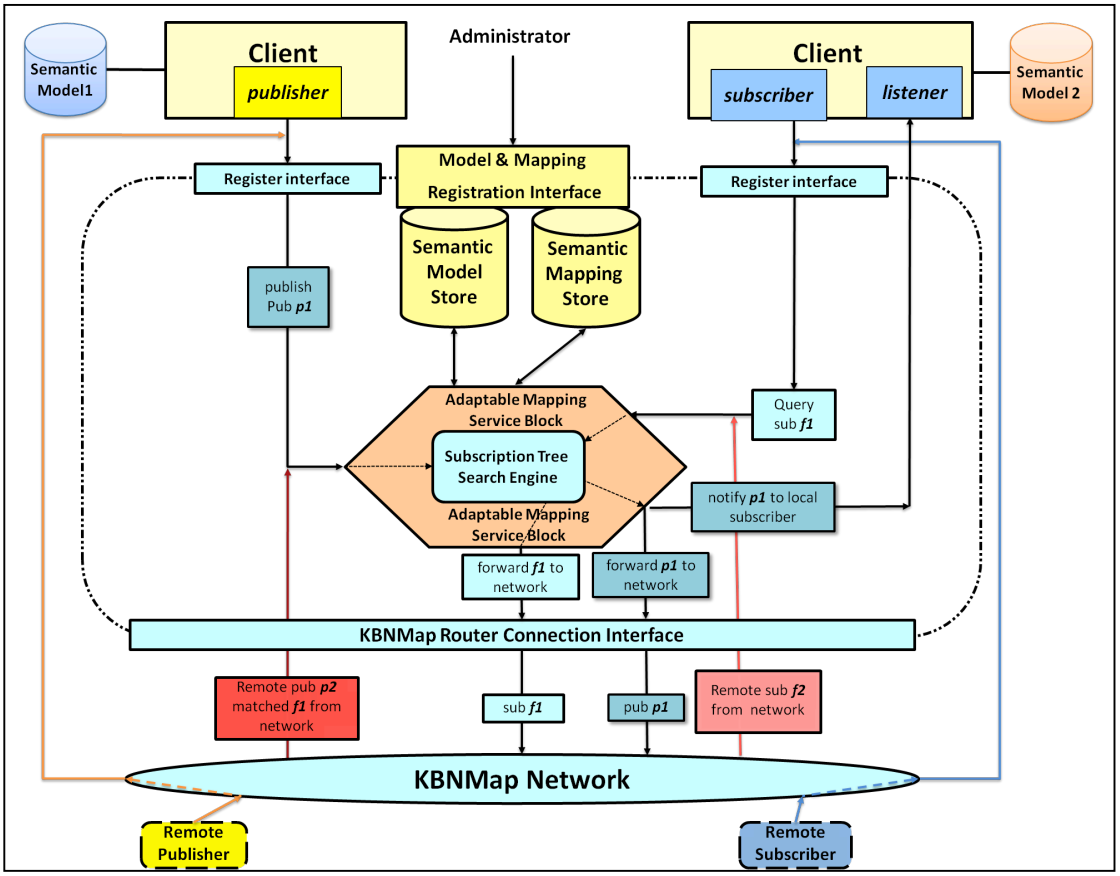


Figure 4-2: KBNMap Architecture

Semantic Model Store

This repository stores semantic models supplied to the KBNMap router. These models are injected by the administrator and loaded by the router at either network setup time or runtime. The terms used in these semantic models make up the domain of knowledge (the common model) that the KBNMap router can resolve the incoming subscriptions and publications that are expressed by the common semantic model.

Semantic Mapping Store

This repository stores mappings provided by applications or network administrators. The mapping files can be stored in routers at network setup time or network runtime, as new mapping files become available. The router will explore the mapping store to find out the appropriate mapping files at network runtime as required.

Model & Mapping Registration Interface

This interface allows a network administrator to register applications, routing semantic models and mappings with KBNMap routers. On one hand, during registration the administrator provides a routing semantic model, which defines the domain of knowledge that the router uses for distributing information. On the other hand, a network administrator could provide these mappings between different application and routing models to KBNMap routers anytime. These mappings are stored in the mapping store where they can be retrieved by the KBNMap router through triggering the semantic mapping service.

Adaptive Mapping Service Block

The adaptive mapping service block is composed of mapping strategies (discussed in Section 4.3.4) and allows KBNMap routers to support heterogeneity by efficiently merging mappings that relate known and unknown terms into the routing semantic model. It retrieves models and mappings from the stores and uses this information to support semantic interoperability. This functionality is presented in more detail in Section 4.3.3 and Section 4.3.4. In addition, the mapping service block also provides the adaptable functionality for KBNMap routers so that they have the ability to dynamically configure their mapping strategies to adapt to changes of influential factors. This functionality is discussed in Chapter 5.

Administrator

The administrator must have knowledge of the network environment (i.e., number of nodes of the network, the amount of resources allocated to the router) and the available semantic models (i.e., the size of semantic model, the complexity of semantic model, the number of mapping files). The administrator must take the application and deploy it into the

environment, so that it can communicate with the network. Additionally, once an administrator finds new mapping files, they can be registered into the semantic mapping store.

The next subsection discusses the operation of the extended KBNImpl to resolve heterogeneous information.

4.3.3 Components Supporting Heterogeneity

The KBNMap router is an extension of the KBNImpl router to support semantic mappings and to support heterogeneous ontologies in the network, based on the proposed architecture shown in Figure 4-2. Each KBNMap router is implemented with two ontology repositories: the *routing ontology store* provides the ontology for the KBNMap operation, whereas the mapping files and the reference ontologies in the *mapping ontology store* are used for helping the KBNMap router achieve semantic interoperability. In the original KBNImpl router, every router had a copy of the same routing ontology; however in this extension each router can have a different local routing ontology, and a different set of mapping ontologies to support interoperability between application ontologies. All ontologies are provided by the administrators of the network. The *ontology registration interface* allows administrators to register both application ontologies and mappings with KBNMap routers. As the *subscription tree search engine* is mainly used to deal with publication matching and subscription routing, it is the first place to encounter unknown information. Thus the engine was extended with an "*unknown information check*" operation and "*try mapping strategies*" operation. Before the *subscription tree search engine* processes the received subscription/publication, the former operation is used first to check if the message received is unknown data by examining the ontological terms in the routing ontology. While the latter operation is used to trigger the mapping strategies if some contents of the message is confirmed as *unknown*. Detailed information about how these two operations could be implemented can be found in the implementation chapter (Chapter 6). It should be noted that the concrete semantic mapping strategies (Section 4.3.4) associated with the adaptive mapping strategy selection mechanisms (Chapter 5) called by the "try mapping strategies" operation should be implemented independently rather than integrated with the *subscription tree search engine*. The main reason is that it is helpful for extendibility, so that potential new mapping strategies can be added , and the strategy selection mechanism can be easily reconfigured according to different tasks. In addition, this approach avoids major changes of the original KBNImpl codebase.

Subscriptions can arrive at the KBNMap router either directly from a client or from another node in the KBNMap network. The subscription query, using terms from the subscriber's local ontology, is passed to the *subscription tree searching engine*, which searches the

subscription tree and inserts the subscription in the appropriate position. However, the subscription may use ontological terms that are not contained in the router's local ontology, and so its covering relationship with other subscriptions cannot be immediately resolved.

Similarly, when a publication arrives at a KBNMap router, again either directly from a client or from another KBNMap node, the *subscription tree searching engine* walks the subscription tree to find appropriate matching subscriptions to find the set of subscribers (clients and other KBNMap nodes) that should be notified with the publication. Again, the publication may use ontological terms that are not contained in the router's local ontology, and so the set of matching subscriptions cannot be immediately resolved.

If the *subscription tree searching engine* gets a subscription or notification with ontological terms which are not expressed in terms from the routing ontology, the *adaptive semantic mapping service block*' components supporting heterogeneity is called to explore the *mapping store* where the mappings have been stored previously. The next section discusses a number of different mapping strategies that the KBNMap routers can employ to handle unknown ontological concepts, properties or individuals.

Difficulties involved in implementation

The main difficulty in design of an implementation for integrating the mapping service is the choice of which place is appropriate for integrating the adaptive mapping service to maximally diminish the extra overhead introduced, in order to ensure the performance of subscription tree search engine with respect to processing messages. For example, if the adaptive mapping service is used to resolve the unknown message after it iterates through the subscription tree, the resolved unknown data has to re-search the subscription tree again, resulting in more extra time being used.

4.3.4 Semantic Mapping Strategies

As stated in Section 3.3.1.4, relying on a single strategy semantic mapping mechanism for a SBPS system to address heterogeneity is not optimal, as the SBPS system would deal with many different scenarios where the conditions of encountered heterogeneous information and the requirements of applications/systems to address heterogeneity will differ significantly. In addition, it is also necessary to enable the KBNMap routers with multiple mapping strategies to adapt to the semantic model, application and network environment related influential factors identified in Section 3.3.2. Even though there is a number of possible mapping strategies identified which could be used to deal with heterogeneous information in SBPS systems, the most appropriate mapping strategies should be selected for the KBNMap router to efficiently explore mappings to achieve semantic interoperability. The mapping strategies

are selected based on the influential factors' impact on the performance of the KBNMap deployment and KBNMap's characteristics.

From an ontological point of view, multiple ontologies determine the diversity of mappings which are generated from these ontologies, resulting in a number of different mapping files stored in the KBNMap routers. In addition, the KBNImpl authors in (Lynch et. al., 2006; Keeney et. al., 2007) have demonstrated that loading of new ontologies into a reasoner embedded in a KBNImpl router is computationally expensive, as the reasoner has to merge the new ontologies with the routing ontology to construct a new knowledge base, and re-reason it. This means loading and merging the mappings into the routing ontology of KBNMap will inevitably consume time and memory resources. Like normal ontologies, it is natural that the mapping files are different in size and complexity, and are particularly effected by the characteristics of referenced ontologies imported in the mapping files. The research in (Lefort et. al., 2006; Motik et. al., 2006; Termpich et. al., 2003) has shown that the size and complexity (DL expressivity) of ontologies are reasonable indicators with respect to reasoning performance (Section 3.3.2.1), which indicates that it will take a different amount of time and resources to load different mapping files into the routing ontology. Thus the selected mapping strategies for the KBNMap routers must have different approaches to load and reason mappings and associated referenced ontologies.

From an application point of view, the KBNMap network could be configured from small-scale deployment to large-scale deployment, resulting in differences in the rate of heterogeneous information occurrence. The KBNMap deployment might want to resolve the heterogeneous information as soon as possible in a large heterogeneous information situation where a number of applications/routers might have their own application/router ontologies. If the mapping files of different ontologies are available, this requires a mapping strategy that enables the KBNMap routers to load all these mapping files and merge different ontologies into their routing ontologies, so that all the heterogeneous information from different ontologies could be resolved. On the other hand, the situation might exist where a large number of mapping files are stored in the KBNMap routers but the unknown information encountered might be originated from only one unloaded ontology. For example, a new router or new application with its own ontology joins in the network. This requires a mapping strategy which is capable of discovering and merging only the appropriate mapping file into the KBNMap router that encounters unknown ontological terms. Furthermore, a mapping strategy that empowers the routers to discover and merge the appropriate individual mappings is also needed for the KBNMap, as it is not always necessary to load the entire mapping file where there is only small amounts of unknown ontological terms encountered at the router.

With respect to environmental influences, the KBNMap implementation could run on a desktop that is allocated with sufficient memory resources, and it could also run on a mobile device such as mobile sensor that is allocated with small memory resources. Given that loading mappings into the routing ontology is a resource consuming process, it is conceivable that mapping strategies that differ in resource consumption are needed for the KBNMap deployment.

In Figure 4-3, the workflow for mapping strategies selection at runtime for each individual KBNMap router is shown. Each strategy takes a different approach to selecting which mapping ontologies to load (and ontologies referenced in the mappings) in order to deal with unknown concepts (or properties, or individuals). Each strategy is also designed with the characteristics of consuming different memory resources. Each of the strategies is described in the following subsections. It should be noted that other candidate mapping strategies KBNMapare also discussed in the subsections.

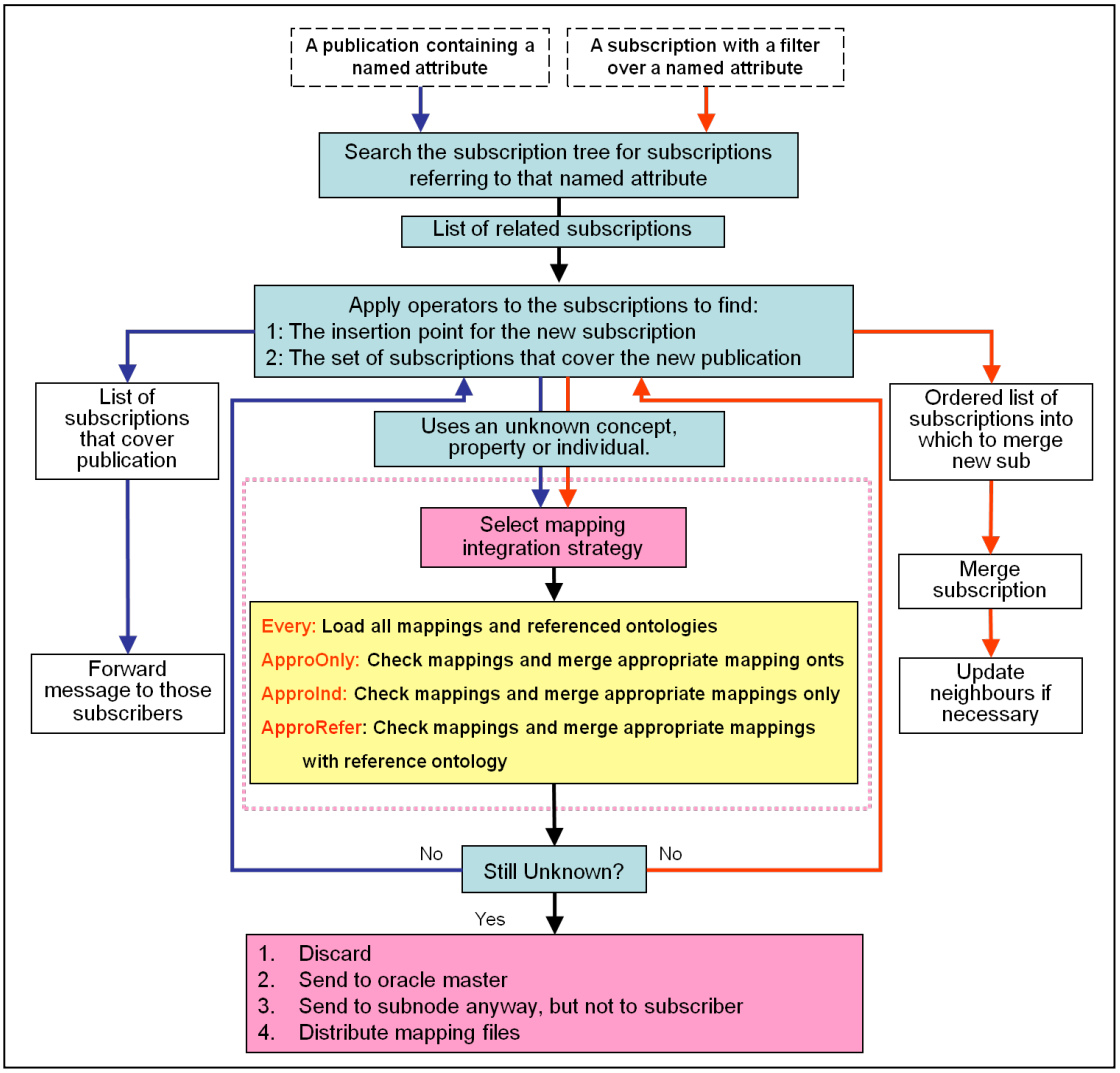


Figure 4-3: Workflow of Mapping Strategies

4.3.4.1 The "Every Mapping File" Strategy (Every)

When a KBNMap router encounters an unknown concept, it loads all known mapping files and their imported ontologies in the ontology mapping store. This strategy maximises the exploration of mappings to tackle heterogeneity problem. Intuitively this strategy is suitable for the situation where the rate of unknown data occurrence is large. This strategy would reduce the probability that further unknown concepts will be encountered at a later stage, at the expense of a potentially very high once-off cost. However, this strategy may be wasteful if there are many mappings, with many referenced ontologies but occurrences of unknown concepts are rare or all unknown terms are derived from the same ontology. As loading unnecessary mapping files into the routing ontology is a highly expensive process, especially for the KBNMap deployment that is configured with limited memory resources. Section 4.3.5 describes the experiment undertaken to explore the intuition.

4.3.4.2 The "Appropriate Mapping File" Strategy (ApproOnly)

This strategy searches the mapping files stored in the KBNMap router in order to select only mapping files which contain at least one unknown term used by the conflicting subscription or notification. The router then loads the entirety of the selected mapping files without necessarily loading, reasoning and merging all of the available mapping files. Furthermore, when compared with the **Every** strategy, the KBNMap does not load the ontologies imported by the mapping ontologies. This is because many of these imported ontologies may not be relevant and to load all imports could introduce considerable overheads. This strategy should be beneficial where KBNMap router stores a large number of specific and fine grained mappings. For example, based on the ontologies introduced in the "XG-Policy Distribution" scenario (Section 2.4), where the *xgpl-regn1* ontology is already loaded as the routing ontology, but the *xgpl-regn2* and the mapping file *regn1-regn2* that providing mapping correspondences between this two ontologies is not loaded. Assuming there are also some other mapping files available in the KBNMap router, if the unknown concept *xgpl-regn2:EuroCity* from the *xgpl-regn2* ontology is encountered, the KBNMap router searches the available mapping files. A mapping is found linking the unknown *xgpl-regn2:EuroCity* concept to the previously known *xgpl-regn1:City* concept, then the mapping file *regn1-regn2* containing this mapping is loaded. As with the **Every** strategy, Section 4.3.4 outlines experiments undertaken exploring when selecting this strategy is appropriate.

4.3.4.3 The "Appropriate Individual Mapping" Strategy (ApproInd)

This strategy also searches the set of available mappings to try to determine which mappings to load into the KBNMap router's routing ontology. Here only the individual mapping is

incorporated into the routing ontology, rather than the whole mapping file it was contained in. Ontologies referred to or imported by the mapping ontology are again not loaded. This strategy only succeeds if a direct mapping exists between the unknown concept and a known concept. For example, based on the ontologies introduced in the "XG-policy Distribution" scenario (Section 2.4), where the *xgpl-regn1* ontology is already loaded, but the *xgpl-regn2* ontology is not loaded. If the unknown concept *xgpl-regn2:EuroCity* from the *xgpl-regn2* ontology is encountered, the mappings are searched. A mapping is found linking the unknown *xgpl-regn2:EuroCity* concept to the previously known *xgpl-regn1:City* concept. In this scenario, only the specific individual mapping is loaded, but neither the *xgpl-regn2* ontology, nor referenced ontologies, are loaded. Again Section 4.3.4 explores in more detail the usefulness of the strategy.

4.3.4.4 Other Candidate Mapping Strategies

As demonstrated in Figure 4-3, if none of the strategies above resolve the unknown concept, property or individual, the KBNMap is left with a number of possible alternatives for dealing with the unknown ontological data. These include:

The "Appropriate individual Mapping with Reference ontology" Strategy (ApproRefer)

This strategy is very similar to the **ApproInd** strategy above. However, depending on the combination of mapping relation found, and the operator to be applied to the unknown concept (for subscription ordering, or subscription matching), unloaded ontologies referred to in the mapping may also be loaded. This strategy could be used where there may not exist a direct mapping relationship between the unknown concept and any known concept, yet the unknown concept may be present in the referenced ontology. Table 4-1 describes when the router needs to load the second referenced ontology, where **r** is the unknown concept, and a mapping exists between **a**, a known concept in the loaded ontology, and **y** a concept in an ontology referenced by the mapping. We have considered the equivalence, subclass and superclass relationships and the three new KBN operators, EQU, MORESPEC and LESSSPEC.

Line 4, 5, 7 and 9 in Table 4-1 shows when the router needs to load second ontology to tackle the unknown concept **r**. For example, the KBN router gets a subscription containing a named attribute filter using the LESSSPEC operator over a known concept "xgpl-regn1:city" (**a**), and a notification arrives containing an attribute with the same name with its value an unknown concept "xgpl-regn2:town" (**r**). After checking mapping files, the router finds the "xgpl-regn1:city" has subclass "xgpl-regn2:EuroCity" (**y**). At this stage, it is unknown if "xgpl-regn2:town" is related to "xgpl-regn2:EuroCity" (e.g. **r** = **w**, or **r** = **x**, or **r** = **z**). Thus the

xgpl-regn2 ontology (**ont2**) referenced in the mapping must be loaded. Afterwards, operation "xgpl-regn2:town" LESSSPEC "xgpl-regn1:city" can complete and the KBN router's operation can proceed as normal.

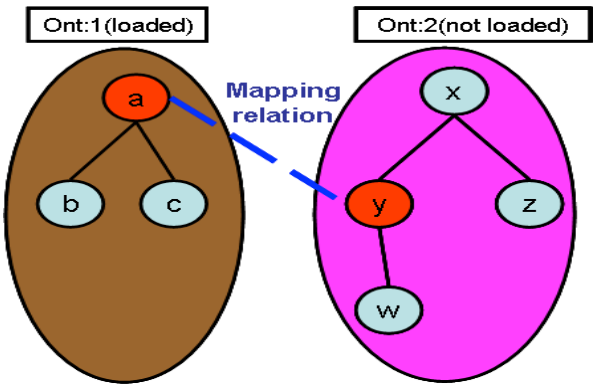


Table 4-1: Conditions of Loading Second Ontology

Matching operator	Mapping relation	if (r ==y)	if (r !=y)	
r EQU a	Y map_equiv a	Don't import ont 2	Don't import ont 2	1
	Y map_subclass a	Don't import ont 2	Don't import ont 2	2
	y map_superclass a	Don't import ont 2	Don't import ont 2	3
r MORESPEC a	Y map_equiv a	Don't import ont 2	Do import ont 2	4
	Y map_subclass a	Don't import ont 2	Do import ont 2	5
	y map_superclass a	Don't import ont 2	Don't import ont 2	6
r LESSSPEC a	Y map_equiv a	Don't import ont 2	Do import ont 2	7
	y map_subclass a	Don't import ont 2	Don't import ont 2	8
	y map_superclass a	Don't import ont 2	Do import ont 2	9

Despite the **ApproRefer** strategy takes advantage of the KBNImpl's ontological operators to load the appropriate reference ontology into the routing ontology, it was clear during the implementation stage that the KBNMap router had to run the **ApproRefer** strategy frequently when it encounters unknown data, leading to an inefficient performance. For example, as Section 3.4.7 described, once a router receives a subscription, it will walk through the subscription tree to compare the received subscription with the subscriptions stored in the subscription tree. If the received subscription is an unknown data, it indicates that the KBNMap has to trigger the **ApproRefer** strategy every time when the unknown subscription is compared with each stored subscription. In addition, as stated in Appendix C, the initial evaluation of the **ApproRefer** mapping strategy demonstrated that it required a considerable

amount of extra processing for executing the strategy. Frequently executing the strategy would significantly influence the process of subscription/publication matching and subscription routing, especially in a high unknown data rate situation. As stated in Section 3.2.2, the efficiency of message matching and routing is important for the scalability of a SBPS system, so this strategy is not adopted in the current version of the KBNImpl router.

Discard the message with a warning

KBNMapIf the KBNMap router realises the mapping strategy executed cannot resolve the encountered unknown ontological term, then this indicates that the appropriate mappings for that unknown data may not be available. The KBNMap router can then drop the message containing this unknown data with a warning to inform the administrator that the unknown data could not be resolved. If the warning information appears frequently, the administrator then has to look for the proper mappings for the heterogeneous information and register them to the KBNMap routers that need the mappings.

The current KBNMap implementation adopts this option to deal with remaining unknown messages, as the KBNMap router does not require extra processing or resources to resolve this remaining unknown message due to the administrator being responsible for discovering the required mappings instead. This is important, as one of the thesis objectives is to try to diminish the amount of overhead introduced by using the designed adaptive semantic service in KBNMap.

Forward the message to a “oracle” node

If the KBNMap router adopts this possible option, it then forwards remaining unknown messages to a predetermined “oracle” node that would be responsible for resolving such unrecognised terms. The KBNMap router can take advantage of the inherent subscription language to express the query to the “oracle” node. The “oracle” master could be a root KBNMap node within the hierarchical topology or a KBNMap node, perhaps running on a more powerful computer with more memory resources. The “oracle” master may then have the necessary resources and reduced load to have a global view of the entire KBNMap network and has the knowledge of all available mappings registered with each node of the network. Once the “oracle” master gets the “unknown message resolution” query from the router who is willing to resolve this, it will retrieve the registered mappings and find the appropriate mappings that could resolve the unknown data. And then the “Oracle” master can either send the mapping file or send the URL of the mapping file to the router who needs it. It is also envisioned that this option would require more relaxed time constraints to reduce load on the “oracle” node.

It should be noted that this strategy is highly dependent on further development of KBNImpl (Lewis et. al., 2006b; Keeney et. al., 2008a; Keeney et. al, 2008b) where the identification of a candidate "oracle" master is a part of ongoing work. This is out scope of this thesis so it is not implemented in the current KBNMap version.

Forward the message to its sub nodes

This option allows the KBNMap to forward the unknown message that it cannot resolve to its child router nodes (but not directly to subscribers, thereby avoiding false-positive subscription matches), in the hope that they can handle it. If the term is unknown at that node then it can execute its own mapping strategies to try resolve this unknown message. If the unknown message is resolved, similar to the "oracle" node above, the child nodes can send the appropriate mapping file directly or the URL of the mapping file to the KBNMap router who needs it. As the research work in this thesis is focusing on the router's inherent capability of efficiently using mappings to address heterogeneity, this aspect of KBNMap routers' operation is not further discussed in this thesis.

Distribute available mappings

This option is inspired by the Cashua (Power 2007) context system, which allows for the automatic dissemination of ontology mappings among context service nodes without requiring a human to manually register them at the context service nodes. In KBNMap deployment, through the use of subscription query, a KBNMap router without a mapping can express interest in it. Once other routers are aware of this interest, they publish the address where the required mapping information can be found. Again, this aspect of KBNMap routers' operation is not further discussed, as this thesis is focusing on the router's inherent capability of efficiently using mappings to address heterogeneity.

4.3.5 Initial Evaluation/Discussion of Mapping Strategies

In order to determine which of the implemented strategies suit different application scenarios, it was necessary to undertake an experiment to initially compare performance of each of the strategies and the impact of different numbers of mapping ontologies and imported ontologies on the strategies' performances. This evaluation work has been published in (Guo et. al., 2007). A detailed description of this experiment is available in Appendix C.

From the evaluation results it was concluded that each strategy has its own advantages and disadvantages. The **Every** strategy has "once-off" cost benefit of loading all of the mapping and referenced ontologies, which is beneficial where a KBNMap network is likely to encounter unknown concepts in its future operation. If the mappings between heterogeneous

ontologies are available, this strategy is particularly well suited for the KBNMap deployment where there are a large number of applications and routers with their own ontologies and the communication between the applications/routers is frequently. On the other hand, its main disadvantage is the operating cost of loading all ontologies in the ontology store, which has been verified to be extremely expensive by this initial evaluation. Thus the **Every** strategy is not appropriate for the KBNMap with small memory resources. In addition, it is conceivable that executing the **Every** strategy is wasteful if the unknown data is derived from the same source.

The **ApproOnly** strategy could be the appropriate strategy for scenarios where the rate of unknown data occurrences remains a medium level, particularly for the case where a new application or router with its own ontology frequently express or query information to the network. The evaluation results demonstrate that executing the **ApproOnly** strategy requires a considerable amount of memory resources, indicating the KBNMap routers allocated with sufficient resources are able to use this strategy. On the other hand, the considerable resources required for executing the **ApproOnly** strategy indicate that this strategy is not appropriate for the network deployment where the resources available is scarce.

While it is beneficial to use the **ApproInd** strategy in the resource-poor circumstances, as the evaluation results demonstrate it has the lowest overhead for processing heterogeneous information. However it becomes inefficient if unknown data occurs frequently in network, mainly due to the KBNMap routers having to resolve the unknown data individually.

Furthermore, the **ApproRefer** strategy as a supplemental strategy for the **ApproInd** strategy can be used to further explore the reference ontologies to resolve the unknown data. This strategy is especially suitable for the situation where the unknown data occurrences are rare but the requirement of addressing heterogeneity is high. On the other hand, since this strategy highly depends on the ontological operators applied in the subscriptions that the KBNMap router receives, it is not appropriate to use this strategy in the KBNMap routers that process the subscriptions and publication-subscription matching frequently. Executing the strategy would inevitably lead to unacceptable long delays for processing the messages, which has been evidenced from the efforts made to implement this strategy into the KBNMap router.

For convenience, Table 4-2 summarises mapping strategies with respect to their capability for semantic interoperability and environments they are suitable for.

Table 4-2: Characteristics of Mapping Strategies

Strategy Name	Characteristics	Semantic Interoperability Level	Suitable Scenario
Every	Load all mapping files and reference ontologies	Once-off benefit for all potential unknown data derived from multiple ontologies. & high resource consuming	It is suitable for the scenario where a large number of applications and routers have their own ontologies
ApproOnly	Load appropriate mapping file without referenced ontology	Once-off benefit for the unknown data origianted from one source &medium resource consuming	It is suitable for the case where few applications and routers with their own ontologis occur.
ApproInd	Load appropriate individual mapping only	Resolving the unknown data on an individual basis & low resouce consuming	It is suitable for the case Heterogenous information rarely occur.
ApproRefer	Load appropriate individual mapping and also the referenced ontologies	Deeply Resolving the unknown data on an individual basis & medium resouce consuming	It is suitable for the situation where the unknown data is rare, however highly demanding on addressing the heterogenity

As the mapping strategies differ in loading mappings and consuming resources, it is better that all mapping strategies are available to the KBNMap router rather than relying on one single mapping strategy, so that the KBNMap router is capable of handling different situations. However, this brings another challenge for the KBNMap router: how to select the appropriate mapping strategy to handle the heterogeneous information in a certain circumstance? The selection of the appropriate mapping strategy must be achieved by the KBNMap router itself without administrator's intervention, so as to ensure that unknown data is resolved rapidly and also reduce the workload on the administrator. In addition, given the dynamic nature of networks, the KBNMap router needs to dynamically self-select the appropriate mapping strategy in accordance with the changes of the network. As stated in Section 3.3.2, these changes can be categorised into the ontology, application and environment related influential factors. In the next chapter, the KBNMap's components to support for adaptively selecting the appropriate mapping strategies are described and discussed in detail.

4.4 Heterogeneity Resolution Example

This a simple example to illustrate how a KBNMap router deals with the heterogeneous subscription/publication it receives. The heterogeneous information resolution is shown in the UML activity diagram of Figure 4-4: assuming it is a small KBNMap network which is composed of one subscriber, one router and one publisher. In addition the region ontologies and mapping files described in Section 2.4 were used. At network setup time, the network administrator registers the *routing* ontology: *xgpl_Region1* that will be used for routing information with the ontology store. In addition, the administrator also registers the *reference*

ontology: *xgpl_region2* and the mapping file *Region1-Region2* that provides mapping correspondences between the *routing* and *reference* ontologies with the mapping store. At network initialisation time, only the routing ontology *xgpl-Region1* is loaded into the KBNMap router.

Publication Matching Resolution with unknown data

The diagram shows a KBNMap subscriber_1 on the left, who wishes to subscribe information about "region area that is more specific than city". It then forwards a subscription *sub1*: [region @> city][i] to the KBNMap router. It is important to note that this subscriber and the KBNMap router have the same ontology: *xgpl-Region1* for communication.

Once the KBNMap router gets the subscription *sub1*, it will iterate through its subscription tree to see if there are some stored subscriptions that are more general than *sub1* or not, and then insert *sub1* into the appropriate position of subscription tree. As ontological term "city" is expressed by routing ontology *xgpl_Region1*, the semantic mapping service will not be triggered by the KBNMap router. If the subscription *sub1* is not covered or aggregated by another subscription, and so becomes a "root" subscription, then the KBNMap router should inform its parent router (if present).

On the other hand, there is a KBNMap publisher shown on the right side of Figure 4-4. Unlike subscriber_1, this publisher has its own ontology (*xgpl-Region2*) for publishing region relevant information. Once this publisher has a piece of information about "the region is Europe city", it will publish a publication *pub*: [region: europe_city][ii] to the KBNMap router. It is important to note that the term "europe_city" is expressed by publisher's own ontology: *xgpl_Region2* rather than routing ontology: *xgpl_Region1*.

Once the KBNMap router receives this publication *pub*, it is matched against the subscriptions stored in the subscription tree. As *pub* is not expressed by the routing ontology, the router will recognise it as an *unknown* data, which leads to the *mapping service* being called by the router. The router selects a mapping strategy according to a set of specific rules (i.e., selecting mapping strategies on the basis of the scale of network: using the **ApproInd** strategy for small-scale deployment and the **Every** strategy for large-scale deployment) to deal with this unknown *pub*. For simplicity, assuming the **ApproInd** strategy being used in

---

[i] As the basic format of KBNImpl subscription filter is [attribute_name, operator, attribute_value], "region" is the attribute name, @> is a shorthand notation for the "more specific than" ontological operator, "city" is an ontological concept.

[ii] The basic publication format of KBNImpl is a pair of attribute-value: [attribute_name: attribute_value], "region" is attribute name, "europe_city" is attribute value.

this example, the particular individual mapping relation_1 "europe_city **subclassof** city" will be loaded and merged into the *routing* ontology after the KBNMap router discovers and explores the mapping file *Region1-Region2* through using the **ApproInd** mapping strategy. After this mapping process the unknown term "europe_city" becomes known from router's perspective[iii]. Therefore *pub* is matched against a list of subscriptions whose attribute name is "region". Obviously, this publication *pub* will be matched to subscription *sub*, as the KBNMap router now knows the "europe_city" is more specific than "city". Then the KBNMap router sends *pub* to the subscriber1. Eventually this initially unknown publication *pub* will be received by the KBNMap subscriber.

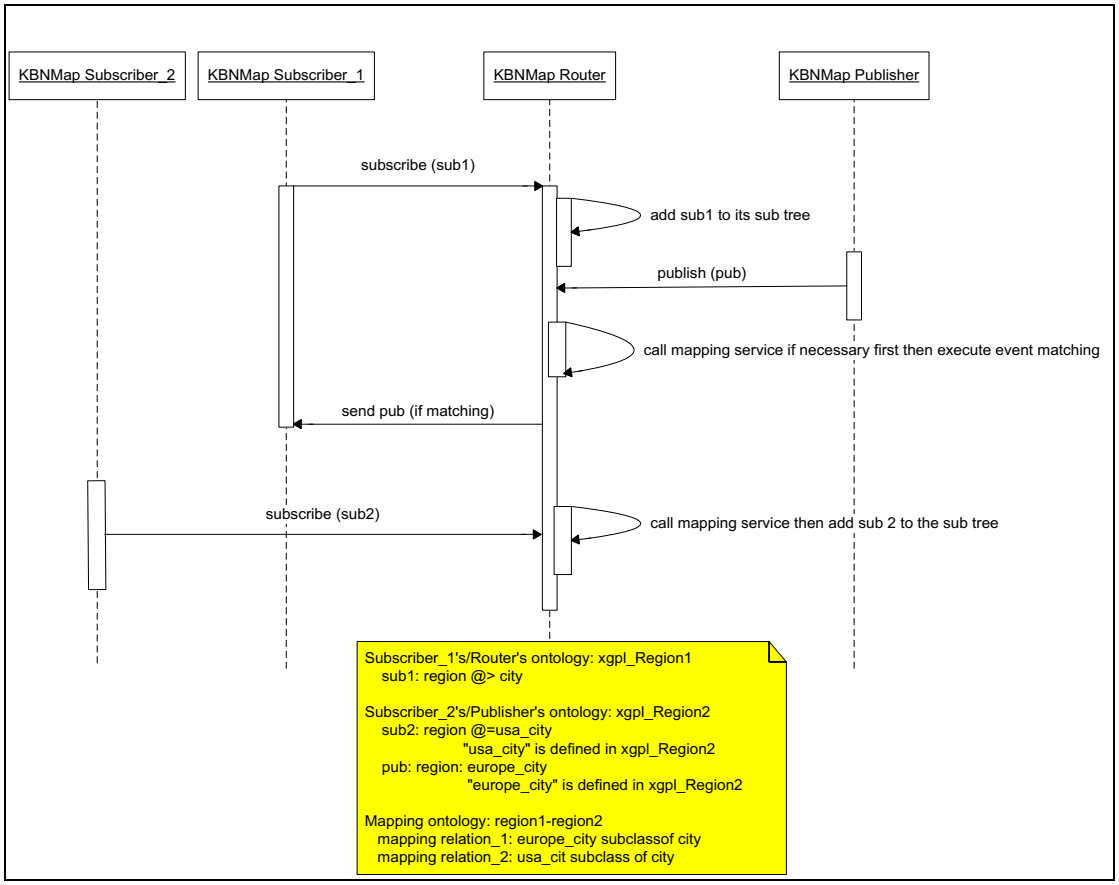


Figure 4-4: Subscription/Publication Resolution Process

Subscription Resolution with unknown data

If there is another subscriber2 joining the network and it also uses the reference ontology *xgpl-Region2* as its application ontology to express information. If this subscriber2 is interested in the region information "region area that is equivalent to usa_city", it forwards

[iii] It should be noted that if the unknown publication/subscription still remains unknown after executing the mapping strategy, the current implemented KBNMap router just simply drops it with a warning.

the subscription *sub2* [region @= usa_city] (@= is the "ontological equivalence" operator while the "usa_city" is expressed by the *xgpl_region2* ontology) to the KBNMap router. It should be noted that in the publication resolution case, only the mapping relation1 was merged and loaded into the routing ontology rather than the entire mapping file. Thus once the KBNMap router gets the subscription *sub2*, it will indicate it is as an unknown subscription. The semantic mapping service is then called again to resolve this unknown subscription. Assume this time the KBNMap router executes the **ApproOnly** mapping strategy which helps the router load and merge the entire mapping file *region1-region2* into the routing ontology rather than only the individual mapping relation_2 "usa_city is more specific than city". After the mapping process the subscription *sub2* is known by the KBNMap router, it is then inserted into the subscription tree. According to the covering relation in KBNImpl (see Table 3-1), *sub2* is covered and aggregated by the *sub1* in the subscription tree; so there is no need for the KBNMap router to inform its parent router (if present).

4.5 Design Discussion

Strengths

The proposed adaptive semantic mapping service has been applied in the KBNImpl router, so that the KBNImpl router can flexibly use ontology mappings to allow multiple ontologies coexist and support heterogeneity. According to the high-level architecture illustrated in Figure 4-1, it can be envisaged that the designed semantic mapping service architecture is also applicable to other SBPS systems, that are similar to KBNImpl incorporating event matching/routing algorithms, such as OPS or S-ToPSS described in Section 3.4. The only requirement on the SBPS system is that it can identify when a semantically enhanced publication or subscription contains an ontological term that cannot be resolved, and so trigger the semantic mapping service. This is a common operation in all SBPS systems and so this approach is applicable to other SBPS systems. This makes it possible to integrate the semantic mapping strategies used in KBNMap with other SBPS routers, as they mainly facilitate the matching/routing engines to resolve heterogeneity. In addition, the number of mapping strategies designed in this chapter provides multiple choices for the SBPS systems to use. The SBPS systems can either use one particular mapping strategy for simplicity, or use several mapping strategies alternatively in order to address heterogeneity properly. This also addresses the criteria of providing flexible semantic interoperability in Section 3.3.1.4. Furthermore, the mapping strategies also facilitate the evolution of the SBPS system, as a shared common semantic model could be updated as applications evolve and require new sets

of knowledge (sub models), Simply by providing mappings, the SBPS system can integrate the sub models with the shared semantic model. It is also envisioned that other SBPS systems could support additional mapping strategies to those presented here, depending on their implementation characteristics.

One strength of having the mapping service inside the router is that it allows the SBPS routers to "wait and see", before taking action to address heterogeneity. In other words, it helps the routers to examine the unknown data encountered and its own contxt and so as select the appropriate mapping strategy to use to address the heterogeneity. Otherwise, use of an incorrect mapping strategy will unnecessarily expend resources and delay message processing. In experiment 2 in Chapter 7 the significant impact of using a mapping strategy in a wrong way on the routers performance of processing publication matching and delivery has been demonstrated. Another concrete scenario to load appropriate mapping a priori, before the network begins to encounter unknown data would be wasteful of resources if nobody uses unloaded ontologies.

From an implementation perspective, the mapping strategies applied in the KBNMap mainly use the Jena ontology model (Jena 2005) to resolve heterogeneous information. Jena provides a programming environment for working with RDF, RDFS and OWL files. Thus the mapping strategies can be easily implemented with the RDF/OWL based SBPS systems such as OPS, SPS-P2P described in Section 3.4. In addition, as mentioned in Section 4.3.3, it is easy to add new mapping strategy with the KBNMap router without changing the core part of original KBNImpl codes, as the mapping strategies are implemented independently.

Challenges/Difficulties

Similar to KBNImpl, KBNMap uses OWL itself to express ontology mappings. However, the ontology mappings can be expressed in many different knowledge sharing formats as discussed in Chapter 2. Thus if some ontologies mappings are expressed by other languages, they will be not interpretable to the KBNMap routers and the network administrator. In order to use these mappings, the administrator or router will need to undertake extra effort to transform these ontologies into an appropriate machine interpretable format. However, if one of these formats begins to emerge as the one most commonly used, this transformation work may disappear over time.

In addition, as the mapping service is designed to address heterogeneity inside the router, the extra time and resources that are required for the routers to use the mapping service, will inevitably influence the efficiency of SBPS routers operation for matching and routing information. Due to a number of mapping strategies being designed, as discussed in Section 4.3.5, the main difficulties involved in implementation of this mapping service is to decide the

impact of mapping strategies on the router's operation and how that impact could be diminished. This is important, as the findings of these evaluation are helpful to support the SBPS systems to adaptively select the appropriate mapping strategy to cater for the changes of influential factors identified in Chapter 3. Detailed information of how the designed mapping service can be implemented to handle the impact of loading mappings using different strategies and how to achieve adaptability is presented in Chapter 5.

From an implementation perspective, the adaptive mapping service has to face the normal issues involved with the integration of different systems. Particularly, as stated in Chapter 3, the Semantic-based Pub/Sub system (SBPS) is an emerging technology; there is no common specification or standard for designing and developing. This results in existing SBPS systems being designed and developed using different methodologies and technologies. For example, Chapter 3 described that the investigated SBPS systems use different ontology languages to express their knowledge model and subscriptions such as the OPS system uses RDF language while the OBPS uses SPARQL language.

4.6 Conclusion

In this chapter, an adaptive semantic mapping service scheme has been proposed to solve the heterogeneity problem in Semantic-based Pub/Sub systems. A high level architecture of SBPS routers integrated with the adaptive semantic mapping service is given. The KBNImpl deployment was selected to integrate with the proposed mapping service for evaluating the feasibility of the extended architecture. The semantic mapping enhanced KBNImpl system is called KBNMap. In addition several mapping strategies are identified for KBNMap system to determine how semantic mapping information can be incorporated into the router's routing ontology. Each of the strategies were discussed and compared, with scenarios identified to suggest when different strategies should be selected. Different mapping strategies should be configured in different KBNMap routers depending on the particular characteristics of the routers' ontologies, the characteristics and number of available mappings, the type of application operating over the KBNMap as well as the network environment state.

In the next chapter, the components of the *adaptive semantic mapping service* in KBNMap to address the second key challenge "Adaptation to Changes" are addressed, this includes the identification of key influential factors, and the realisation of the adaptation functionality for KBNMap, by using a probabilistic modelling approach.

5 MAPPING STRATEGY SELECTION MECHANISM

In the previous chapter, the designed architecture of KBNMap is presented to address the key challenges outlined in Section 3.6; particularly the novel mapping strategies supporting heterogeneity is outlined. The chapter presents the components of KBNMap to adaptively select mapping strategies in order to address the second key challenge for "adaptation to changes".

5.1 Introduction

The diversity of the three presented mapping strategies for resolving heterogeneous information (Section 4.3.4) brought a requirement for designing a decision making component to support strategy selection, to enable KBNMap routers to dynamically select the appropriate mapping strategy at runtime. The strategy selection mechanism must be dribven by changes of the ontology, application and environmental characteristics, which will be important in influencing strategy selection.

Therefore, in this chapter, deriving from the influential factors identified in Section 3.3.2, a number of key influential factors (named as trigger factors) that have significant impact on mapping strategy selection are identified in Section 5.2 The ontology related factors that have most significant impact on mapping strategies, are identified through an experimental method. Section 5.3 discusses the investigation that leads to the choice of the probabilistic approach for selecting the appropriate mapping strategy. Section 5.4 describes the probabilistic approach being employed for modelling and predicting strategy selection in detail. Section 5.5 briefly describes how the entire process of selecting mapping strategies and using the selected mapping strategy operates inside the KBNMap router.

5.2 Identification of Trigger factors

Due to the dynamic nature of the network environment, different KBNMap routers could store different routing ontologies along with different numbers of mapping ontologies. This can cause significantly different repercussions on the reasoning, matching and routing performance of a KBNMap router executing a specific strategy to deal with unknown data. For instance, the "*Every mapping file*" strategy is well-suited for routers which store a small number of mapping ontologies, whereas strategies that do not import some of the ontologies

referenced by mappings are well suited for the routers with large number of ontologies. It was observed in a small scale scenario that it may be possible to examine the application running over the KBNMap to statically determine which strategy is most appropriate for a given deployment. However, in a large scale deployment, or where the ontologies stored in KBNMap and applications using the KBNMap may change, then it is necessary to dynamically manage and adapt the strategy that is most appropriate. Hence, different mapping strategies may be configured in different KBNMap routers depending on a number of factors, some of which may be changing dynamically. Based on the influential factors identified in Section 3.3.2, the possible trigger factors were categorised into three main sets: characteristics of the *ontologies,* which focus on how the changes of routing ontology and mapping ontologies influence on the reasoning performance of the KBNMap; *application* characteristics, which concentrate on the impact of the dynamics of applications on the strategy selection; and *environment* characteristics, which focus on how the changes of networking environment determines the mapping strategy selection.

Given different possible mapping strategies, the following subsections focus on identifying which of the *ontology*, *application* and *environmental* characteristics will be important in influencing strategy selection and what that influence might be with a view to building a decision making component.

5.2.1 Ontology-based Trigger factors

Ontology characteristics that may impact strategy selection at an individual KBNMap router are: the size and complexity of the applications' ontologies, routers' ontologies, mappings and referenced ontologies; the number of imported ontologies referenced in a mapping file; the ontological mapping operators used in the mapping files; and the concept's position in the ontology's class hierarchy tree. These candidate factors are selected on the basis of the semantic model relevant influential factors identified in Section 3.3.2.

5.2.1.1 Ontology Characteristics

For convenience, the concept of ontology is briefly introduced first in order to help to understand the characteristics of ontology. The detailed description of ontology can be found in Chapter 2.

An ontology consists of classes, properties and instances. Classes describe the characteristics or concepts of individual things within the ontology, while properties describe relationships between or about things. The class hierarchy tree in an ontology is based on the taxonomic relationships between concepts, based on equivalence or sub-/super-class semantic relationships. Due to the formal nature of how many ontologies are specified, it is possible to

perform some reasoning over the classes and their properties to correctly derive this class hierarchy (classification or TBox reasoning). The root node is semantically the most generic class; whereas the leaves are the most specific classes. A sub-class is said to be *subsumed by* its super-classes, while a class *subsumes* its subclasses.

In state of the art surveys (Lefort et. al., 2006; Motik et. al., 2006; Termpich et. al., 2003) most researchers have taken the number of classes, properties and individuals along with the languages that are used to describe an ontology, as ontology characteristics which impact ontology reasoning. The results of such research have shown that these simple ontology characteristics are reasonable indicators with respect to reasoning performance. For instance, the number of classes and properties of an ontology influences the time for a reasoner to compute TBox classification (arranging the classes and properties into their reasoned hierarchy), while the number of individuals determines the amount of ABox realisation (finding the types of an individual, in particular its most specific type). However, this research was only interested in exploring a wider set of ontology characteristics given that the KBNMap router's reasoning performance will be influenced by having to cope with several ontologies as opposed to just one ontology at a time. In order to do this, an experiment to explore what these influential ontology characteristics might be was designed. As an aside, this work has been published in (Guo et. al., 2008)

5.2.1.2 Hypothesises for Experiment

The first hypothesis for the experiment was that size and expressivity of an individual ontology is a good indicator of the reasoning overhead required for that ontology.

The second hypothesis was that the reasoning overhead for a merged ontology could be predicted or bounded from the reasoning overheads of its constituent individual ontologies. This would help us select appropriate strategy to efficiently tackle heterogeneous data by considering the reasoning overhead involved in loading new mappings and ontologies, as the process of loading and reasoning has proven to be a serious overhead (Lewis et. al., 2006b).

The third hypothesis was that different ontological operators used in the mappings would not lead to significantly different reasoning overheads in a merged ontology. The mapping operators considered in this work are rather restrictive, i.e., that an individual in one ontology is the same as an individual in another ontology, or that a class (or property) in one ontology be *equivalent to* or be a *sub-/super-class* (or property) in another ontology. For instance, given a mapping where the *equivalence* operator is applied to link two different classes in different ontologies the reasoning requirement for that merged ontology is similar to the situation where a *sub-class* mapping operator is used. If the mapping operator does impact performance then it may be included as a factor in our strategy selection.

The fourth hypothesis was that the positions of the classes within the class hierarchy trees of the constituent ontologies will have little effect on the reasoning overhead of the merged ontology. Figure 5-1 shows a concrete example: concept **a** (root node in ont1) and **y** (middle node in ont2) are the mapped classes in a mapping ontology **m1** that only contains this one mapping. Assuming there is another mapping ontology **m2** that only specifies that classes **a** (root node in ont1) and **x** (root node in ont2) are mapped, would **m2** have more or less reasoning overhead than **m1,** if the same ontological mapping operator being used?

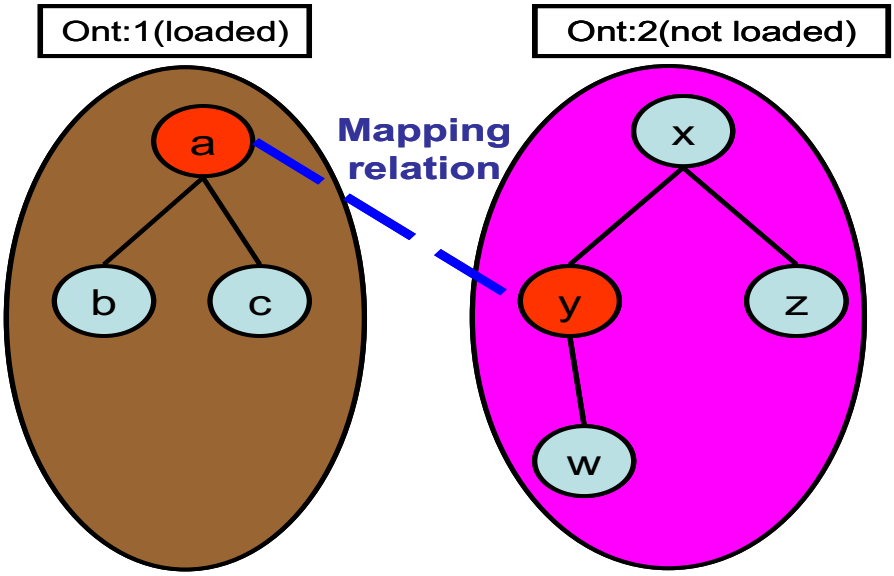


Figure 5-1: Illustration of classes positions in class hierarchical tree

5.2.1.3 Experimental Method

Three types of ontologies were used in the experiment: a **routing ontology** that is already loaded by a router and used to match and route publications and subscriptions; a number of **reference ontologies** that contain definitions for classes or properties which are not described in the routing ontology; and a number of **mapping ontologies** that each contain semantic mappings between classes or properties in the routing ontology and reference ontologies. When a mapping ontology and a referenced ontology are incorporated into the routing ontology, the term **merged ontology** was used to refer to the resulting ontology. Ontologies from the semantic web research community were used as the routing and referenced ontologies in the experiment, along with manually-created the mapping ontologies. These ontologies are discussed below and the URLs for these ontologies can be found in Appendix D. In the experiments, the reasoning overhead of the ontologies individually was measured and afterwards the reasoning overhead of the merged ontology was measured.

5.2.1.4 Test Ontologies

Routing and Referenced Ontologies

14 commonly available semantic web ontologies were used for the experiment. The selection of these ontologies was motivated by the fact that the ontologies are created by different people with diverse technical backgrounds. In this sense, the ontologies can be considered as representative of the natural range and diversity of ontologies that will be expected in a ubiquitous and distributed computing environment.

The ontologies were chosen in order to reflect a range of expressiveness and ontology sizes, from ontologies with small number of statements to ontologies with large number of statements. (Gardoso 2007) indicates that normally the smallest ontologies has less than 100 concepts, medium ontologies has between 100 and 1000 concepts, and large ontologies has more than 1000 concepts. However, in the experiment, these ontologies are categorised into four sets according to the number of statements that are the basic elements in ontology terminology, as the number of statements can more precisely reflect the size of ontology than the number of concepts.

In addition, depending on the expressiveness of an ontology some of the following letters can be used to denote the presence of description logic features in the ontology, thereby capturing its reasoning complexity: AL - Attribute Logic: Conjunction, Universal Value Restriction, Limited Existential Quantification; C - Complement (together with AL allows Disjunction, Full Existential Quantification); R - Role Transitivity; H - Role Hierarchy; I - Role Inverse; O - Nominal; N - unqualified number restrictions; Q – qualified number restrictions; F - only functional number restrictions; (D) – Datatypes.

Table 5-1 summarises the ontologies according to the ontology characteristics of interest:

Table 5-1: Individual ontology characteristics

Name	Number of Statements	DL Expressivity	Imports	Annotation
Teams	262	ALCIF	0	Small ontologies where number of statements is less than 1,000
Foodswapper	350	ALC(D)	0	
University	453	ALCR+OIF(D)	0	
Beer	576	ALHIF(D)	0	
Foaf	808	ALCHIF(D)	0	
Mad_cow	1,012	ALCHOIN(D)	0	Medium ontologies where number of staments is between 1,000 and 5,000
Mindswappers	2,303	ALCHIF(D)	3	
Pizza	3,201	ALCF(D)	1	
Transportation	4,847	ALH(D)	0	
CongoService	5,199	ALCR+HOIF(D)	12	Large ontologies where num of statements is between 5,000 and 10,000
Economy	5,489	ALH(D)	0	
Wine	5,710	ALCR+HOIF(D)	1	
MGED	14,501	AL(D)	0	Very large ontologies where num of statements is greater than 10,000
Galen	64,673	ALCR+HF	0	

Manually Created Merged Ontologies

15 sample merged ontologies were manually created by the author to compare the reasoning overhead of each merged ontology with the reasoning overheads of its constituent individual ontologies. An individual merged ontology has only two mapped classes, where the two constituent individual ontologies are randomly chosen from the ontologies described above. The characteristics of the merged ontologies that were created are shown in Table 5-2:

Table 5-2: Characteristics of ontologies merged using manually created mappings

Name	Number of Statements	DL Expressivity	Constituent Ontologies
FoodUniversity	695	ALCHOIN(D)	Foodswap & University
FoafFood	1,077	ALCHIF(D)	Foaf & Foodswap
FoafUniversity	1,157	ALCR+HOIF(D)	Foaf & University
BeerMadcow	1,485	ALCHOIN(D)	Beer & Mad_cow
FoafMadcow	1,716	ALCHOIN(D)	Foaf & Mad_cow
BeerSwapper	2,774	ALCHIF(D)	Beer & Mindswappers
FoodPizza	3,460	ALCF(D)	Foodswap & Pizza
FoafPizza	3,901	ALCHIF(D)	Foaf & Pizza
BeerTransport	5,320	ALHIF(D)	Beer & Transportation
EconomyFood	5,374	ALCH(D)	Economy & Foodswapper
EconomyBeer	5,964	ALHIF(D)	Economy & Beer
EconomyTransport	10,239	ALH(D)	Economy & Transportation
EconomyCongo	10,599	ALCR+ HOIF(D)	Economy & CongoService
EconomyWine	12,336	ALCR+ HOIF(D)	Economy & Wine
EconomyMGED	19,882	ALH(D)	Economy & MGED

Created Mapping Ontologies

In order to investigate the effect of mapping operators and mapping positions upon performance, 70 mapping ontologies (categorised into 7 types) were created by altering some of the merged ontologies above to include different mapping operators and applying different mapping positions. The *equivalence, sub-class, super-class* mapping operators were all applied to map between classes in the two source ontologies.

In order to evaluate the impact of class position on reasoning performance, different classes from the top (T), middle (M) and bottom (B) of the class hierarchy trees were mapped using the *sub-class* ontological mapping operator. The class position measured are shown as following:

- $CA_T \xleftarrow{Sub} CB_T$ **:** CA and CB refer to classes in two different ontologies while the sub-script T means that the class is at the top (root) of the ontology's class hierarchy. $\xleftarrow{Sub}$ is the sub-class mapping operator used to link mapped classes.
- $CA_T \xleftarrow{Sub} CB_B$: Here the sub-script B means that the class is at the bottom (leaf) of the ontology's class hierarchy.

- $CA_B \xleftrightarrow{Sub} CB_B$: Both mapped classes are leaf nodes
- $CA_M \xleftrightarrow{Sub} CB_M$: Here the sub-script M means that the class is at the middle of the ontology's class hierarchy. Both mapped classes are the intermediate nodes

Table 5-3 outlines 70 created mapping ontologies, which are categorised into seven types according to the type of mapping operators applied and the impact of class position. It is important to note that an individual mapping ontology has only two mapped classes and two constituent individual ontologies. Each column name stands for each type of mapping ontology. It can be observed that each merged ontology derives seven different mapping ontologies, each of which has either different mapping operator or different mapped class position. For example, for candidate merged ontology "FoafUniveristy", 7 different mapping ontologies were derived, one for each type of mapping. For example, for the "Mapping 1" type, a mapping ontology was created, in which the two classes are mapped with "equivalence" operator. For "Mapping 4" a mapping ontology was created where the "sub-class" mapping was applied to link a top class (root) of the Foaf ontology's class hierarchy with a bottom class of the University ontology's class hierarchy.

Therefore, different categorised set of mapping ontologies was used to investigate for different purposes. For example, the mapping ontologies of "Mapping 1" set were used to investigate the effect of equivalence mapping operator upon the reasoning performance.

Table 5-3: 70 mapping ontologies, categorised in 7 types

Name	Mapping 1	Mapping 2	Mapping 3	Mapping 4	Mapping 5	Mapping6	Mapping7
FoafUniversity	Equivalence	Sub-class	Super-class	SUB(TT)	SUB(TB)	SUB(BB)	SUB(MM)
BeerMadcow	Equivalence	Sub-class	Super-class	SUB(TT)	SUB(TB)	SUB(BB)	SUB(MM)
FoafMadcow	Equivalence	Sub-class	Super-class	SUB(TT)	SUB(TB)	SUB(BB)	SUB(MM)
BeerSwapper	Equivalence	Sub-class	Super-class	SUB(TT)	SUB(TB)	SUB(BB)	SUB(MM)
FoodPizza	Equivalence	Sub-class	Super-class	SUB(TT)	SUB(TB)	SUB(BB)	SUB(MM)
EconomyFood	Equivalence	Sub-class	Super-class	SUB(TT)	SUB(TB)	SUB(BB)	SUB(MM)
EconomyTransport	Equivalence	Sub-class	Super-class	SUB(TT)	SUB(TB)	SUB(BB)	SUB(MM)
EconomyCongo	Equivalence	Sub-class	Super-class	SUB(TT)	SUB(TB)	SUB(BB)	SUB(MM)
EconomyWine	Equivalence	Sub-class	Super-class	SUB(TT)	SUB(TB)	SUB(BB)	SUB(MM)
EconomyMGED	Equivalence	Sub-class	Super-class	SUB(TT)	SUB(TB)	SUB(BB)	SUB(MM)

5.2.1.5 Metrics Used

Before introducing the metrics used for this experiment, it is worthwhile to briefly describe the "DL Reasoner Experiment", as presented in Appendix B, since the metrics used in this experiment were obtained from it. Note that this experimental work has been published in (Lewis et. al., 2006).

Overview of "DL Reasoner Experiment"

In order to determine the extent to which ontology reasoner implementations differed with respect to initial loading and subsequent querying times; an experiment to compare OWL Description Logic (DL) reasoner implementations was undertaken. There are three typical DL reasoners that are measured in the experiment: Racer (Haarslev et. al., 2001), the Jena framework (Jena 2005), and Pellet (Parsia et. al., 2004). The exploratory results of that experiment led to the adoption of the Pellet reasoner framework(accessed through the Jena) as the default reasoning configuration for KBNImpl routers. However, in situations where very good runtime performance is required, and this performance requirement outweighs the need for complete and correct reasoning, the use of cut-down or application specific reasoners may prove beneficial.

In addition, that experiment also confirmed the results and observations obtained from the exploratory experimentation of KBNImpl in (Lynch et. al., 2006; Keeney et. al., 2006), which are shown as follows:

1. First, the loading of new ontologies into a reasoner embedded in a KBNImpl router is computationally expensive due to load-time inference, so the frequency of changes to the ontological base of a given KBNImpl router must be minimised since changes will need to be distributed to each of the nodes in the network.
2. Second, ontological reasoning is memory intensive and memory usage is proportional to the number of concepts and relationships loaded into the reasoner so memory usage can be controlled by limiting this number in any given KBNImpl node.
3. Third, once loaded and reasoned over, the querying of such an ontological base is relatively efficient, with performance relative to size of the ontological base.
4. Finally, based on the observations above, the (re-)loading and (re-)reasoning of ontologies is expensive and that such operations will greatly affect the scalability of the KBNImpl network.

Metrics used for Measuring Reasoning Overhead

Based on the research in the "DL Reasoner Experiment" in Appendix B the following observations are of particular importance: **loadtime** reasoning, as compared to **runtime** querying is relatively expensive; the performance of different reasoners, and the reasoning load, will also change in a non-linear fashion depending on the size and expressiveness of the ontologies used and the level of ontology language used (i.e. OWL-Lite vs. OWL-DL) (Pellet 2003; Motik et. al., 2006; Pan 2005; Guo et. al., 2004; Guo et. al., 2005). These observations

are particularly important if ontologies are added or removed dynamically, as would be typical in many scenarios where applications and clients will join and depart from the network. Table 5-4 summarises the reasoning metrics used.

Table 5-4: provides a summary of the reasoning metrics

Measure metrics	Description
Loadtime without parsing time	Is given as the time taken for different reasoners to load, and check the ontology, combined with the time taken to perform TBox classification, perform ABox realisation and an initial query of all concepts
Loading time	The time takes to load ontologies into reasoner
Consistency checking time	Consistency checking ensures that an ontology does not contain any contradictory facts. In DL terminology, this is the operation to check the consistency of an ABox with respect to a TBox
Classification time	Classification can be defined as the computation of the subsumption hierarchy for classes and properties
Realisation time	Realisation finds the most specific classes that an individual belongs to, in other words computes the direct types for each of the individuals. It should be done after classification since direct types are defined with respect to a class hierarchy.
Concept querying time	First time to list all classes, all individuals and all properties of an ontology. This is required since some reasoners only complete their reasoning process when aspects of the ontology are queried, resulting in a high once-off query answering time for the first such query.
Runtime	The time taken to perform subsequent queries over the reasoned ontologies

5.2.1.6 Experimental setup

A version of the KBNImpl router, in particular the version described in (Keeney et. al., 2007), was extended to implement all three of the mapping strategies discussed earlier in Section 4.3.4. The Pellet reasoner (Parsia et. al., 2004) version 1.3.beta was embedded into the KBNImpl router. Jena (Carroll et. al., 2004) was used throughout to access the ontologies and to measure the reasoning performance of Pellet. In order to minimise the adverse effect of inconsistent network connection speeds on reasoning performance all tested ontologies and their imported ontologies were cached locally on the machine running the tests. All tests were undertaken on a Dell Inspiron 9300 laptop with 1.73 GHz Intel processor, 2GB of RAM, running Windows XP Service Pack 2. For Java-based tools, Sun's JDK 1.6.0 was used. All tests were run at least 20 times to provide statistically appropriate averages.

5.2.1.7 Evaluation

To address the first hypothesis the selected source ontologies were reasoned to see how reasoning performance was dependent on both the number of statements and the expressivity of the different ontologies, as shown in subsection 1 below.

To test the second hypothesis, the reasoning overhead of a merged ontology was compared with the combined reasoning overhead of its constituent individual ontologies, as discussed in subsection 2 below. This was designed to test if the reasoning overhead of a merged ontology could be predicted given the reasoning overhead of the ontologies that made up that merged ontology.

The third hypothesis was that the mapping operator used had little effect on the reasoning overhead of a merged ontology. Here the operators (but not the selected classes) within mappings were altered and compared, and the results are presented in subsection 3 below.

The findings from examining the fourth hypothesis, which stated that the hierarchical position of the classes used in mappings had little effect on the reasoning overhead of the merged ontology, are discussed in subsection 4 below. Here classes from different positions within the class hierarchy of their individual ontologies were mapped and the reasoning overheads of the resulting merged ontologies were compared.

1. Individual Ontology Reasoning

Figure 5-2 presents the reasoning overhead calculated on the individual ontologies. In Figure 5-2 the ontologies are arranged from smallest to largest (left to right) with the number of statements in each ontology given in parenthesis after its name. As can be seen from the times to load and reason, and runtime query performances, of the different ontologies in Figure 5-2 (and with reference to Table 5-1), the reasoning overhead was greater for the larger ontologies. This confirms that reasoning performance is tied to the number of statements. However it was also observed that, although the wine ontology is not the biggest one in size, it has the largest loadtime reasoning overhead. This is because it has the most complex structure and DL expressivity (see its column in Table 5-1) among tested ontologies. Given these reasoning times and the ontology DL expressivity information, an empirical finding is that both DL expressivity and ontology size impacts on loadtime reasoning performance, not just ontology size.

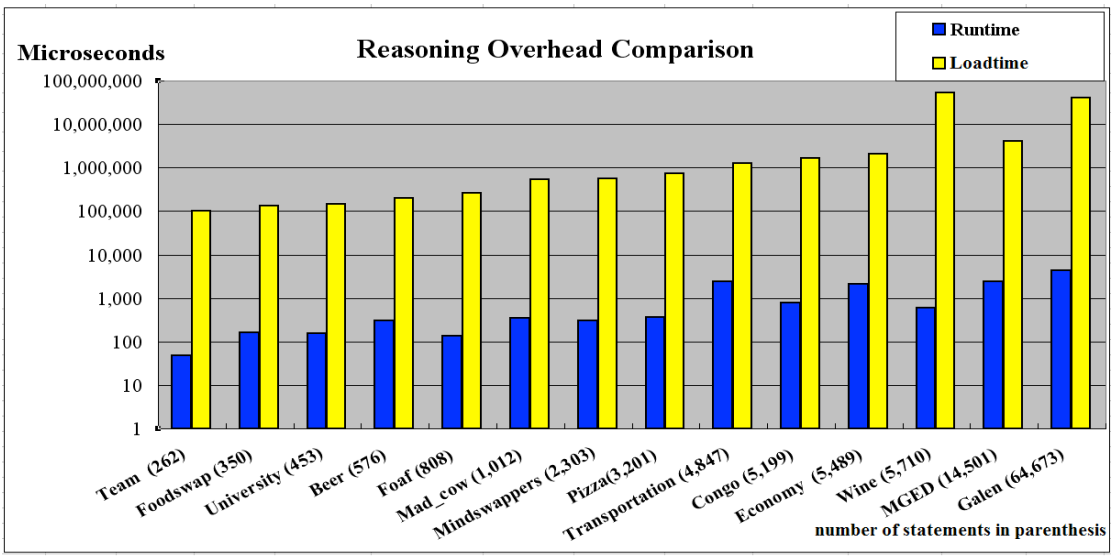


Figure 5-2: The effect of ontology size on reasoning time

2. Merged Ontology Reasoning

The reasoning times of the merged ontologies were compared, as shown in Table 5-2, with the combined (sum) time to reason over the two ontologies that constituted it. As shown in Table 5-5, compared with the combined reasoning overhead of the two constituent individual ontologies in individual merged ontologies, most of the merged ontologies were observed requiring less reasoning time. The **bolded lines** are cases where this is not the case. When the reasoning times were analysed in detail, most of the ontologies showed lower times for some aspects of loadtime reasoning, i.e., loading, classification (TBox), and initial TBox query. The times were very similar for consistency checking, however, the realisation times (ABox) was higher for all merged ontologies. The most likely reason for this increased realisation time is that as the number of individuals increases in the merged ontologies, the reasoner spends more time searching the larger and more complex merged class hierarchical tree to find the proper class that the individuals belongs to. Another finding of this experiment is that the merged ontologies that take significantly longer to reason than their constituents have either a very large number of statements or most complex DL expressivity. For instance, the second largest and most complex merged ontology in our experiment is the one which imported the large economy and complex wine ontologies. This also confirms our previous finding discussed in subsection 1 above on the sensitivity of reasoning performance to both ontology size and DL expressivity. Note from Table 5-2, that the DL expressivity of merged ontologies can be said to be the union of the expressivities of their constituent ontologies. Finally, it was found that the degree of reduction of reasoning overhead on two similarly expressive and sized merged ontologies are comparable.

From our experience of using the KBNImpl in different application scenarios, and experience of its internal algorithms, the occurrence of unknown concepts or properties is much rarer than the number of times that a routing ontology would be queried. Therefore the runtime query time was considered to be an important aspect of KBN performance. Table 5-6 shows the comparison of runtime querying over merged ontologies and their constituent ontologies. Again it was found that the largest ontologies took longer to query.

Table 5-5: Comparing the analysis time of a merged ontology with the reasoning time of its constituent ontologies

Merged Ontology name	Reasoning time: OntA *milliseconds: ms*	Reasoning time: OntB *milliseconds: ms*	Reasoning time: Merged *milliseconds: ms*
FoodUniversity	195	148	249
FoafFood	161	195	234
FoafUniversity	161	148	215
BeerMadcow	199	538	638
FoafMadcow	161	538	600
BeerMindswaper	199	555	595
FoodPizza	195	736	827
FoafPizza	161	736	890
BeerTransport	**199**	**1,265**	**1,655**
EconomyFood	2,051	195	2,263
EconomyBeer	**2,051**	**199**	**2,500**
EconomyTransport	**2,051**	**1,265**	**4,026**
EconomyCongo	2,051	1,695	3,586
EconomyWine	**2,051**	**52,379**	**90,419,878**
EconomyMGED	**2,051**	**4,122**	**9,575,935**

Table 5-6: Comparing the runtime query time of a merged ontology with the runtime query time of its constituent ontologies

Merged Ontology name	Runtime query time: OntA *microseconds: µs*	Runtime query time: OntB *microseconds: µs*	Runtime query time: Merged *microseconds: µs*
FoodUniversity	159	154	254
FoafFood	134	159	172
FoafUniversity	134	154	219
BeerMadcow	304	353	657
FoafMadcow	134	353	304
BeerMindswapper	304	307	456
FoodPizza	159	363	459
FoafPizza	134	363	452
BeerTransport	**304**	**2,383**	**2,803**
EconomyFood	2,158	159	1,950
EconomyBeer	2,158	304	2,205
EconomyTransport	**2,158**	**2,383**	**8,730**
EconomyCongo	2,158	793	2,340
EconomyWine	2,158	610	2,768
EconomyMGED	**2,158**	**2,380**	**8,697**

Coupling the findings from subsections 1 and 2 above, there is confidence that where the number of statements for each ontology is relatively low and where the DL expressivity of these ontologies are not complex it can be generally predicted that the reasoning overhead of a merged ontology will be bounded by the sum of the reasoning overheads of its constituent ontologies. However, a combination of very large number of statements or complex DL expressivity breaks this prediction. It was also found that the reasoning overhead of a merged ontology could be predicted if the reasoning overhead of a similar merged ontology was already known.

3. Impact of Type of Mapping Operators

As discussed in the "experiment method" subsection (Section 5.2.1.3), the experiment here was designed to observe the impact that mapping operators may have on reasoning overhead. Since altering the mapping operator will mainly affect only the structure of TBox classification hierarchy and ABox realisation (as opposed to loading and consistency checking), the classification and realisation were only considered as the measured metrics in this experiment. Recall that the three mapping operators tested here were the *equivalence, sub-class, and super-class* operators, which in Figure 5-3 and Figure 5-4 are represented as EQU, SUB, and SUP respectively.

From the measurement of classification performance (as shown in Figure 5-3) and realisation performance (as shown in Figure 5-4), it is unclear whether any specific operator type has any major impact on reasoning overhead when compared to the other operators. An example from the classification timings shown in Figure 5-3 is the observation that reasoning with *equivalence* operator performs better than the other two operators in the FoafUniversity ontology. However, in the EconomyTransport ontology, the *equivalence* operator requires slightly more reasoning time than *sub-class* operator. As seen in Figure 5-4, three of the ten merged selected ontologies performed slightly better with the *equivalence* operator. Yet, slightly better performance can be seen with some of the other operators in some of the other merged ontologies. Overall from our analysis, it was concluded that there is no direct relation between operator used in a mapping and the time to reason the resulting merged ontology. Thus this candidate characteristic will not be considered as a determining factor influencing mapping strategy selection.

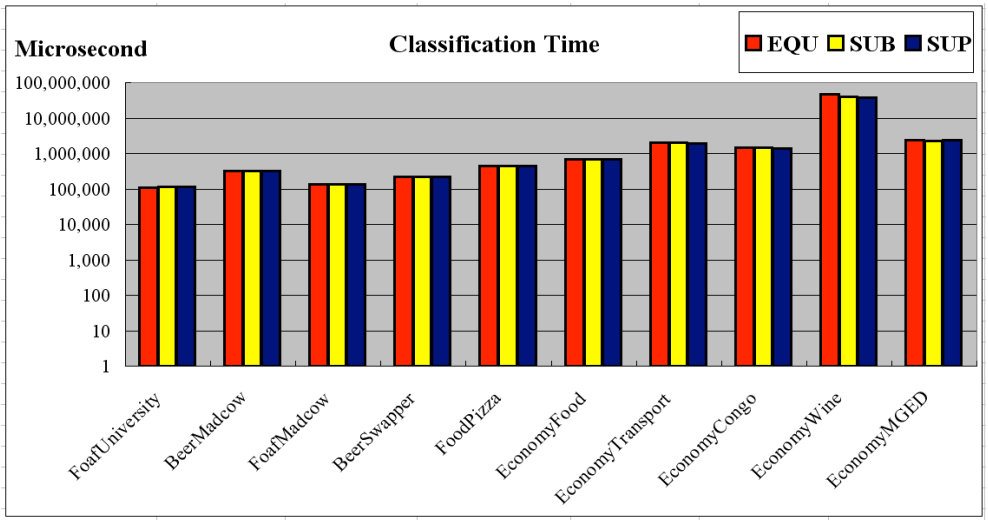


Figure 5-3: Classification performance for different operators

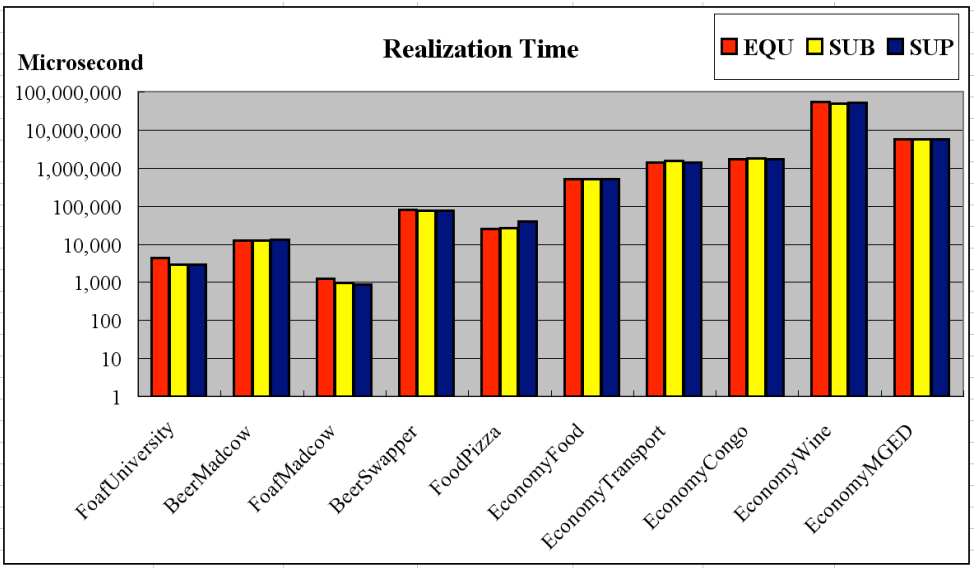


Figure 5-4: Realisation performance for different operators

4. Impact of Position of Concepts used in Mapping

As discussed in the "experiment method" subsection, the experiment here was designed to observe the impact of mapping classes at different positions in their respective class hierarchies. Again, like the previous experiment, since selection of classes at different positions will mainly affect only the structure of TBox classification hierarchy and ABox realisation, again classification and realisation were only considered as the measured metrics in this experiment. It was compared when the same mapping operator (*sub-class*) was applied to two top-level classes (TT), a top-level class and a bottom-level class (TB), two bottom level classes (BB) and two mid-level classes (MM). Recall from the previous experiment that the mapping operator used makes little difference, so only the *sub-class* mapping operator was tested. The classification and realisation performance results are presented in Figures 5-5and Figure 5-6 respectively.

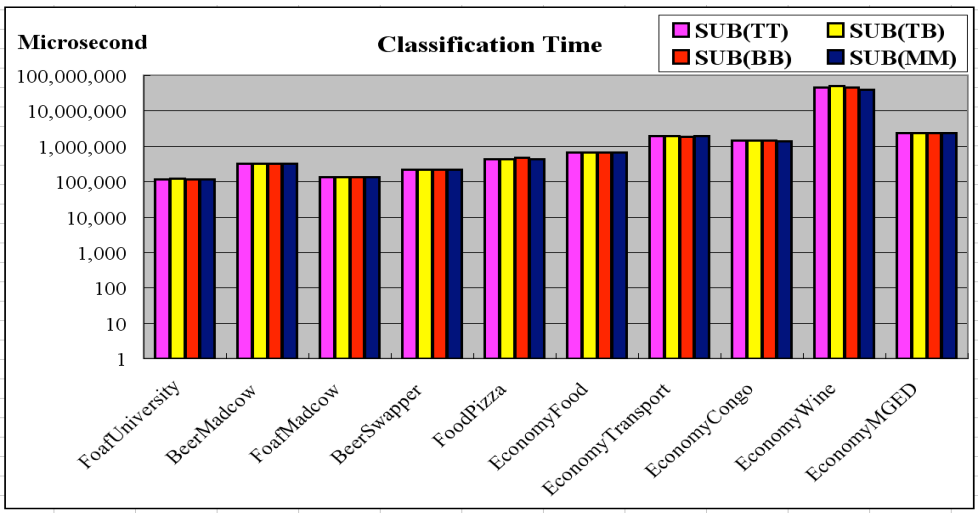


Figure 5-5: Classification performance for different concept positions

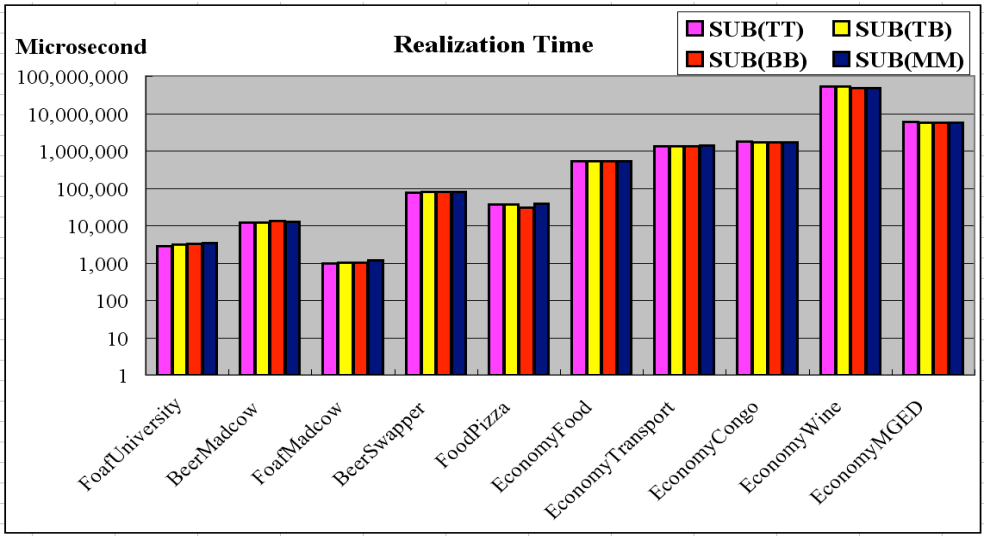


Figure 5-6: Realisation performance for different concept positions

As can be seen from the classification performance (Figure 5-5) of different class positions there is no appreciable consistent difference for the different class positions. Observing the realisation performance (Figure 5-6), the differing reasoning performances on different mapped classes coming from different positions is again not appreciable. For instance, the mapped classes have almost the same overhead in EconomyFood, EconomyTransport, EconomyMGED ontologies, whereas in other ontologies: some TT classes have slightly less overhead (FoodUniversity); some BB classes lead to better performance (FoodPizza). Hence it has been concluded that the concept position in class hierarchical tree is not a reliable characteristic that should be used to influence mapping strategies selection.

5.2.1.8 Conclusions from the Experiment

Observing from the results of the experiments presented in this section, it can be concluded that the combination of count, size and expressivity of ontologies to be loaded and reasoned are the most important *ontology* characteristics to influence the strategy selection. These factors are important as indicators of the potential memory, time and processing overhead involved in merging and reasoning mappings and referenced ontologies into the router's own knowledge base (ontology), and the later overhead of querying these ontologies once loaded. In summary, the ontology-based trigger factors influencing mapping strategies are shown below:

- Routing Ontology Size
- Mapping Ontology Size
- Routing Ontology DL Expressivity
- Mapping Ontology DL Expressivity
- Number of Mapping Ontologies and Associated Imported/Referenced Ontologies

5.2.2 Application-based Trigger Factors

Derived from *application* relevant influential factors identified in Section 3.3.2, the application characteristics that impact strategy selection at an individual KBNMap router are: the rate of publications and subscriptions arriving and the router; the tolerance capability of KBNMap applications to respond gracefully to a false positive / negative match between a subscription and publication that may be due to a mapping being missed; the observed occurrence rates of unknown ontological terms (not expressed by the routing ontology) used in publications and subscriptions. Since the KBNMap network is essentially a middleware, built to facilitate the applications that operate above it, these factors are particularly important in allowing some of the characteristics of the application to inform the semantic interoperability strategy of the brokers/routers in the KBNMap network. The application-based trigger factors are presented in detail as follows:

- **Publication/subscription/unsubscription reception rate:** refers to the rate of messages received by a KBNMap router rather than the rate of messages sent by the clients. This trigger factor is selected for modelling strategy selection, as the rate of messages arriving at a router to some extent determines the rate of unknown data occurring in this router and the router's workload for matching and routing information. Here "unknown data" refers to the ontological terms in subscriptions and publications that are not expressed by the routing ontology. If a router receives more publications and subscriptions than other routers, it is prone to encounter more unknown data than the others. In addition, the high rate of messages received also indicate that the router requires more computing resources for message matching, routing and even executing mapping strategy, which brings higher risk of mismatches between subscriptions and publications and unknown data occurrence. This factor should be taken together with the rest of the application factors.
- **Message false and mismatched tolerance:** this factor also plays a major role of influencing mapping strategy selection. The strategies loading entire mapping files and/or referenced ontologies would reduce the possibility of message mismatching by providing richer knowledge than the strategies which only load individual mappings or mapping files, therefore, the tolerance level assigned by the applications should be lower. Whilst the strategies only loading individual mappings would be used for the environment where the applications are supplied with high level tolerance.
- **Observed unknown data rate:** when a system has been operating in an application domain for a period, the approximate unknown data rate could be observed, so that

the appropriate mapping mechanism could be used for that certain situation. For example, in a small network deployment, if the unknown data rate is observed as "rare" by network administrator, the **ApproInd** mapping strategy is preferable in this case. However considering the dynamic nature of large-scale networking environment, the changes in the rate of unknown data actually occurring in the network would lead to the changing rate of observed unknown data rates. A good mapping service should ensure the system is capable of adapting to this dynamism.

5.2.3 Environment-based Trigger Factors

In this thesis, all network *environment* related trigger factors identified in Section 3.3.2 are considered as trigger factors to impact strategy selection at an individual KBNMap router. It should be noted that each router independently determines the appropriate strategy to use, yet individual (and possibly heterogeneous) routers must operate together as part of a cooperating network of routers to achieve a decentralised knowledge distribution network. Therefore these network environmental factors are required so routers are cognisant of their resources and the topology in which they operate. The environmental trigger factors are illustrated below:

- **Network Scale:** a KBNMap deployment can range from a small deployment, to enterprise scale, to internet-scale. Under certain circumstances, the scale of network has influence on the amount of mapping ontologies stored in a router and the occurrence of unknown data. For example, if KBN is configured for an Internet-scale application scenario (i.e., ubiquitous computing), where the applications are prone to using multiple diverse ontologies for exchanging knowledge. This may lead to a large number of mapping ontologies being injected into each router for the purpose of solving numerous potential occurrences of heterogeneous data produced by the diverse ontologies.
- **Number of Mapping Files and Referenced Ontologies:** refers to the number of mapping ontologies stored at a router, and the number of ontologies imported or referenced by the mappings. In addition, the results of experiment 2 in Figure C-2 in Appendix C also indicates that the number of mapping ontologies and imported ontologies also have significantly negative impact on the performance of mapping strategies. Considering that loading and reasoning an ontology is resource and time consuming, and the **Every** Strategy loads and merges all mapping files and referenced ontologies into the routing ontology, this factor must be taken into consideration for strategy selection.

- **Memory Resources at an Individual Router:** is concerned with the available resources allocated to an individual router. The routers allocated with sufficient resources are capable of dealing with multiple tasks and can be burdened with processing a large volume of information (i.e., processing event matching). The experimental results in (Lewis et. al., 2006) indicated that the loading of new ontologies into a KBNMap router is computationally expensive due to load-time inference and the (re-) loading and (re-)reasoning of ontologies is also expensive. In addition, since the mapping strategies proposed differ in amount of ontologies processing, it is preferable to allow routers to flexibly configure the trade-off between available resources and strategy selected to execute.

5.3 Genesis of the Probabilistic Approach Used

As mentioned in Section 4.2.2, to address the second challenge of "Adaptation to changes", the selection mechanism of mapping strategies for each individual router should rely on the changing states of the key ontology, application, and environment influential factors identified in the previous Section 5.2. In other words, through an analysis of the states of trigger factors which are observed by the KBNMap router, a KBNMap router should predict which mapping strategy is most suitable under that circumstance before it encounters the potential unknown information.

As mentioned in Section 5.2, it can be envisaged that in a small scale scenario that it may be feasible for the network administrator to manually examine the changing states of the network and applications running over the network, to monitor the variation of routing ontology characteristics for each router, in order to determine which strategy is well-suited for which individual router. However, in a large scale deployment, it is would be impractical to examine those changing states and select the appropriate mapping strategy for each router by relying totally upon a human administrator's observation and operation, due to the high traffic and the dynamic nature of the network. Therefore to keep the complexity and uncertainty of observing trigger factors low and to enable efficient, effective and scalable operation of selecting mapping strategy, what is needed is to enable the individual routers to be self-configuring and automatically adapt to the changes of key factors, so that the active strategy of each individual router is tuneable at run time .

Prior to empowering the router's self-managing capability for mapping strategy selection, it is necessary to explicitly express the relations between trigger factors and the selection/prediction of mapping strategies as an influence diagram (Howard et. al., 2005), which is a compact graphical representation of a decision situation for linking trigger factors

and mapping strategies selection, as shown in Fig 5-7. The advantages of using an influence diagram are illustrated below :

1. it exposes the explicit dependencies among influential factors and mapping strategies selection;
2. it models the entire process of selecting strategies in a graphical representation, which is helpful for people to understand the process of selecting mapping strategies;
3. it helps to relieve degree of complexity and uncertainty contained in the process.

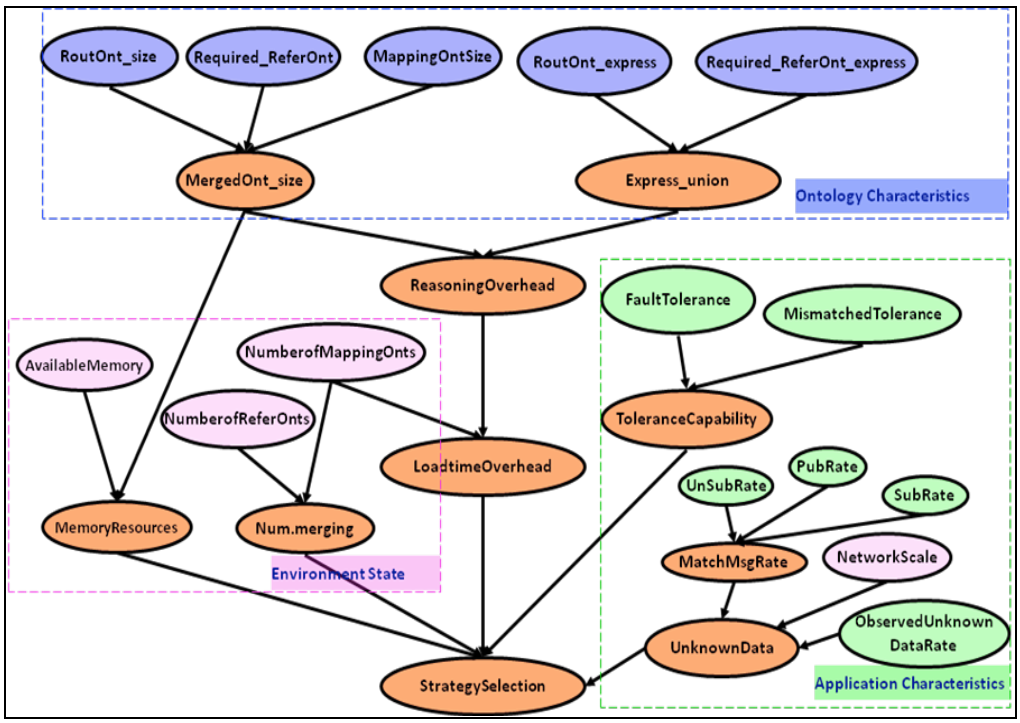


Figure 5-7: Influences of trigger factors on Strategy Selection

As illustrated in Figure 5-7, the leaf nodes represent the 15 identified application-, environment-, and semantic-level strategy selection factors. Intermediate/internal nodes represent combinations and interdependencies in the factors, while the root node represents the combination of factors to influence strategy selection. The direct links between nodes represents the interactions between nodes. For example, the link between **LoadtimeOverhead** and **ReasoningOverhead** indicates that the overhead for reasoning task combined with the number of ontologies has direct influence on the overhead for *loadtime* operation. As the "ontology trigger factor" experiment in Section 5.2.1 indicated that the size and expressivity of ontologies can be categorised into *small/low, medium and large/high* according to their number of statements and DL expressivity, and so each individual ontology-based node in Figure 5-7 is specified with three states: *small/low, medium and large/high*. In addition the other nodes are also specified with *small/low, medium and large/high* states respectively in order to be both in accordance with the specification of ontology-based nodes and maintain simplicity for mapping strategy selection. The final

strategy selection node's three states are specified with the names of three mapping strategies instead. The meaning of each node is presented in the following discussion.

Ontology-based Nodes:

- **RoutOnt_size:** refers to the current size, measured as the number of statements, of the current routing ontology.
- **MappingOnt_size:** refers to the size of mapping ontologies stored at the router.
- **Required_ReferOnt:** refers to the size of the ontologies referenced by the mapping ontologies, again measured as the number of statements.
- **RoutOnt_express:** refers to the reasoning complexity or DL expressiveness of the routing ontology. The expressivity of an ontology can be found by loading the ontology and then querying the ontology reasoner, however, with the aim of minimising the loading of ontologies these values can also be found from lookup tables.
- **Required_ReferOnt_express:** refers to the DL expressivity of the ontologies referenced by the mappings.
- **Express_union:** is the expressivity of a newly merged routing ontology incorporating mappings. This is conditional on the Required_ReferOnt_express and RoutOnt_express nodes. The DL expressivity of a merged ontology is the union of the expressivity of its constituent ontologies.

Application-based Nodes:

- **PubRate**: refers to the average rate at which publications arrive at router.
- **SubRate**: refers to the average rate at which subscription requests arrive at the router.
- **UnSubRate**: refers to the average rate at which unsubscription requests arrive at the router.
- **MatchMsgRate**: is the rate of publication to subscription matching rates will be low, medium or high. This is conditional on the PubRate, SubRate, and UnsubRate nodes.
- **ObservedUnknownDataRate**: refers to the average rate per minute at which an unrecognised ontological concept (or individual, or property) is detected at a KBNMap router.
- **NetworkScale**: refers to the number of brokers comprising a KBNMap network.

- **UnknownData**: is the aggregate rate of unknown data occurrences. This is conditional on the MatchMsgRate, NetworkScale, and ObservedUnknownDataRate nodes.
- **FaultTolerance**: refers to the tolerance the KBNMap clients may have when dealing with false-positive or false-negative publication/subscription matches. It is envisioned that all routers in a KBNMap network might have the same value for this variable.
- **MismatchedTolerance**: refers to the tolerance of the KBNMap clients may have dealing with publication/subscription mismatches due to potentially missed semantic relationships due to the overly conservative loading of mappings. It is envisioned that all routers in a KBNMap network might have the same value for this variable
- **ToleranceCapability**: refers to the composite tolerance of **FaultTolerence** and **MismachedTolerance**.

Environment-based Nodes:

- **NumberofMappingOnts**: refers to the number of mapping ontologies stored at a router.
- **NumberofReferOnts**: refers to the number of ontologies referenced by the mapping ontologies.
- **NumOfStoredOnts**: is the number of ontologies available on a KBNMap router. It is derived from the **NumberofMappingOnts** and **NumberofReferOnts** nodes.
- **AvailableMemory:** refers to the memory allocation given to the KBNMap router application. The value for available memory can be detected from the runtime environment within which the KBNMap router application runs.
- **MemoryResources**: is the amount of memory resources available to the KBNMap router once mappings are loaded. This is conditional on the **AvailableMemory**, **MergedOnt_size** nodes.
- **ReasoningOverhead**: refers to the overhead introduced by the reasoner embedded in the router for reasoning over the new routing ontology. As stated in Chapter 3, a slight change over a semantic model will lead to re-reason over it. In addition, the experimental results indicate that the size and expressivity of ontology are the best indicators of reasoning performance. Therefore, the **MergedOnt_size** and **Express_union** have direct influence on **ReasoningOverhead**.

- **LoadtimeOverhead**: refers to the overhead introduced by the router for performing *loadtime* operation. The *loadtime* is given as the times taken for the different reasoners to load, parse and check the ontology, combined with the time taken to perform TBox classification, perform ABox realisation and an initial query of all concepts (Lewis et. al., 2006). In addition to processing the new routing ontology, the referenced ontologies in the mapping file could also be dynamically loaded and merged by the KBNMap router depending on which strategy is selected. Therefore, the **ReasoningOverhead** and **NumberofStoredOnts** have direct influence on **LoadtimeOverhead**.

Final Strategy Selection Node:

StrategySelection: is used to decide the appropriate mapping strategy for resolving heterogeneous information. Therefore, occurrence rate of unknown data at the router highly determines the strategy selection. In addition, it is known that one of the main differences between mapping strategies is whether to load individual mappings, a required single mapping file, or the entire set of mapping files with their imported ontologies. For example, the **Every** strategy allows for loading all stored ontologies into the routing ontology in order to achieve "once-off" benefit while the **ApproInd** strategy only enables the router to merge the individual mappings into the routing ontology. In addition, loading ontologies is considered an expensive operation, and so the router should take into account not only how many ontologies are stored in the mapping store, but also the memory resources available for supporting the generic operations such as routing and matching/ mapping. Consequently, this node is highly influenced by the combination of its four direct parent nodes. In summary, the **StrategySelection** node is conditional on available memory resources (**MemoryResources)**, ontology loadtime initialisation overhead (**LoadtimeOverhead**), mismatch tolerance (**ToleranceCapability**), and the unknown data occurrence rate at the router (**UnknownData**), each of which continuously change. As an example to illustrate the process of selecting the appropriate mapping strategy, Table 5-7 summarises the possible combinations of states of 11 of the 15 trigger factors that have significant impact on selecting mapping strategies. Therefore, based on this table, the human administrator could select the proper mapping strategy for an individual router through the observation of the combinations of the states of trigger factors. However, selecting mapping strategies purely based on Table 5-7 proves to be a complicated and difficult task due to the following reasons:

- As stated there are 15 individual trigger factors identified and each factor can have three states resulting in up to 3^{15} combinations of factors.
- Not all of the trigger factors have a direct independent impact on the decision

making process, instead, some of the factors are interdependent and cumulative. For example, the network scale, message rate and observed unknown data rate together determine the rate of unknown data occurrences, which then has significant impact on selecting mapping strategies.

- Table 5-7 is overly simplistic since there exists no determining factor to drive strategy selection. Instead, a mechanism to weigh the benefits and deficiencies of each strategy for a given observation of the factors is required.
- In a large scale deployment, as the factors influencing strategy selection change dynamically at *each* broker, it is infeasible to require a human administrator to manually select the appropriate strategy separately for each.

Table 5-7: Which strategy is appropriate for some factors

	Application Trigger Factors				Environment Trigger Factors			Ontology Trigger Factors			
Strategy	*Pub, sub unsub rate/ active (inactive) cycle duration*	*End user tolerance (fault occurrences)*	*End user tolerance (missed matches)*	*Observed Unknown data Rate*	*Network Scale*	*Memory resources allocated in Broker*	*Number of Mapping and reference ontologies*	*Size of an individual mapping ontology (required)*	*Size of routing ontology of the broker*	*Complexity of routing ontology (DL Expressivity)*	*Complexity of mapping ontology (DL Expressivity)*
Every	Large	Small	Small	Large	Small/Medium/Large	Large	Small	Large	Large	Large	Large
ApproOnly	Medium	Medium	Medium	Medium	Small/Medium/Large	Medium	Medium	Medium	Medium	Medium	Medium
ApproInd	Small	Small	Small	Small	Small/Medium/Large	Small	Large	Small	Small	Small	Small

Therefore from the table illustrated above it can be concluded that even though the influence diagram explains the rationale and process of selecting mapping strategy comprehensibly, it is still difficult to select strategies on the basis of this diagram due to the large degree of complexity and uncertainty still existing:

1. First, the influence diagram still lacks essential knowledge such as the states of each node.
2. Second, even given the states to each node, how to specify the boundary values of each state (so that a specific node under which state will be known) for each node is complicated.
3. Third, how to validate and evaluate the credibility of the selected result, how to validate the accuracy and stability of this process model is also difficult.

In order to help the KBNMap router to make an appropriate decision about which mapping strategy to use in a specific circumstance, a probabilistic modelling approach was used for the process of predicting mapping strategies on the basis of the influence diagram. The reason for adopting a probabilistic modelling approach is outlined below:

1. It is the most widely used technique for uncertainty and complexity analysis in all scientific disciplines (Meira 1997; Marvis et. al., 1999; Arabi et. al., 2007).

2. Their use enables the design of rich model of the studied systems, situations, processes that would be otherwise almost impossible to model. Most recently, they were successfully applied directly in the network and service management area (Zhu et. al., 2003; Gonzalez et. al., 2007). For example, the authors in (Gonzalez et. al; 2007) use a probabilistic modelling approach to estimate a near real-time VoIP flows in a large network.
3. Probabilistic modelling approaches drastically reduce the cost of management (i.e., the cost of physical probes, collectors, and high performance data storage) and enable the calculation of the accuracy of the measured data.

In summary, the advantages of probabilistic modelling approaches are considered as follows:

- Accuracy of estimates can be measured.
- Use of statistics improves estimates.
- Realistic communication of uncertainty to decision makers and investors is facilitated.
- Results are immediately usable in network and service management.

In the following subsections, the use of Bayesian Networks as an increasingly popular probabilistic approach of modelling uncertainty and complexity for modelling the process of mapping strategy selection is discussed.

5.4 Using Bayesian Networks for Selecting Mapping Strategy

5.4.1 Motivation of Using Bayesian Network

Bayesian Networks (BN) (Jensen et. al., 2007; Darwiched 2009), also called belief networks, are an increasingly popular probabilistic modelling approaches for modelling uncertainty and complexity in a decision making process in many domains such as text analysis (Dong et. al., 1997), medical diagnoses (Kahn et al., 1997), and evaluation of science evidence (Garbolino et. al., 2002). They are also increasingly used in computer networking area for various purposes (e.g., fault detection and diagnosis) (Learner et. al., 2000; Matsuura et. al., 2004; Sebyala et. al., 2002; Bronstein et. al., 2001).

The Bayesian modelling technique has several advantages in treating uncertain situations and analysing decisions (Jensen et. al., 2007; Darwiched 2009), which drove the decision to

choose this approach to handle mapping strategy selection for KBNMap routers. The main advantages of using BN to model the strategy selection are explained as follows:

1. **Graphical representation**: BNs are mathematical models presented graphically so that as each trigger factor presented as node in the influence diagram (e.g. Figure 5-7), it can also be presented as a node with the directed links forming arcs between them. Therefore, the complexity of inference and can be more easily investigated and efficient inference algorithms can be researched.

2. **Probability distribution:** the information content of each trigger factor (i.e., the possibility of reasoning overhead is conditional on the routing ontology size and complexity) can be represented as one or several probability distribution, which will be used as a measure of uncertainty. BN captures independence and conditional independence between nodes illustrated in Figure 5-7. The probabilistic presentation of the interactions between nodes allows for the estimation of risks and uncertainty.

3. **Deduce causes:** through using BNs, not only the probability distributions of children node (e.g., MatchMsgRate node) given the values of their parents (e.g., *large* PubRate and *large* SubRate) can be calculated, but also the distributions of parents given the values of their children. Thus, it can proceeded not only from causes to consequences, but also deduce the probabilities of different causes given a consequence. For example, observing the inferred weightings of mapping strategies, the KBNMap administrator can deduce the possible states of ontology, application and environment trigger factors.

4. **Explicit treatment of uncertainty and support for decision analysis:** Bayesian Networks can easily be supplemented with variables encoding managerial decisions that in their turn affect the natural variables of the model, and with variables encoding costs and utilities related to these decisions and their outcomes (Jensen et. al., 2007). These models naturally focus on the relationship between actions, knowledge and uncertainty; the consequences of various management decisions can be studied not only from the perspective of expected values, but also with regard to the risks of highly undesirable outcomes. This aspect would be very useful for the KBNMap administrator to manage the overall cost of operating mapping strategies.

5. **Fast responses**: as BNs are solved analytically, they can provide fast responses to queries once the model is compiled (Darwiched 2009). This feature is very important for KBNMap to select the appropriate mapping strategy swiftly in a high traffic environment.

For convenience and to aid understanding, a general introduction of Bayesian Networks is presented in the following section.

5.4.2 Overview of Bayesian Network

A Bayesian Network (BN) (Darwiched 2009) is a form of probabilistic graphical model that represent a set of variables and their probabilistic independences, also called a Bayesian belief network. The best way to understand a Bayesian Network is to imagine trying to model a situation where causality plays a role, but where our understanding of what is really going on is still incomplete and uncertain, so things is needed to describe probabilistically. A Bayesian Network is a directed acyclic graph where there is a set of variables, a set of directed links between variables and a collection of conditional probability tables. Each are now described.

A set of variables

Each variable represents a set of events or possible states. A variable is in exactly one of its states. The observable variables, which provide information, are the quantitative part of the methodology and called as *"information variables"*. The unobservable variables, which the main task of modelling is to identify them, called as *"hypothesis variables"*.

A set of directed links between variables

A set of directed links between variables, with no loops, is a *"directed acyclic graph"(DAG)*. As a definition, a directed graph is "acyclic" when there is no directed path $V_1 \rightarrow \cdots \rightarrow V_n$ s.t. $V_1 = V_n$. When there is a link from V_1 to V_2, then V_1 is parent of V_2, and we write $pa(V_2) = \{V_1\}$, and V_2 is child of V_1. *Marginally independent* of nodes means that there is no influence relation between nodes. The DAGs represent the following connection or causality relations between the variables:

1. *Serial Connection* (Figure 5-8(a)), encodes that $X_1 \perp X_3 \mid X_2$, indicating X_2 is a parent node of X_3, and a child node of X_2. but X_1 and X_3 are not marginally independent, as X_1 has indirect causal influence on X_3.
2. *Diverging Connection* (Figure 5-8(b)), encodes that, although X_1 and X_3 are not marginally independent, as they have the same parent ode. $X_1 \perp X_3 \mid X_2$, means that X_2 is a parent node of both X_1 and X_3, indicating it has direct causal influence on both child nodes.
3. *Converging Connection* (Figure 5-8(c)), encodes that X_1 and X_3 are marginally independent, but they are not independent given X_2.

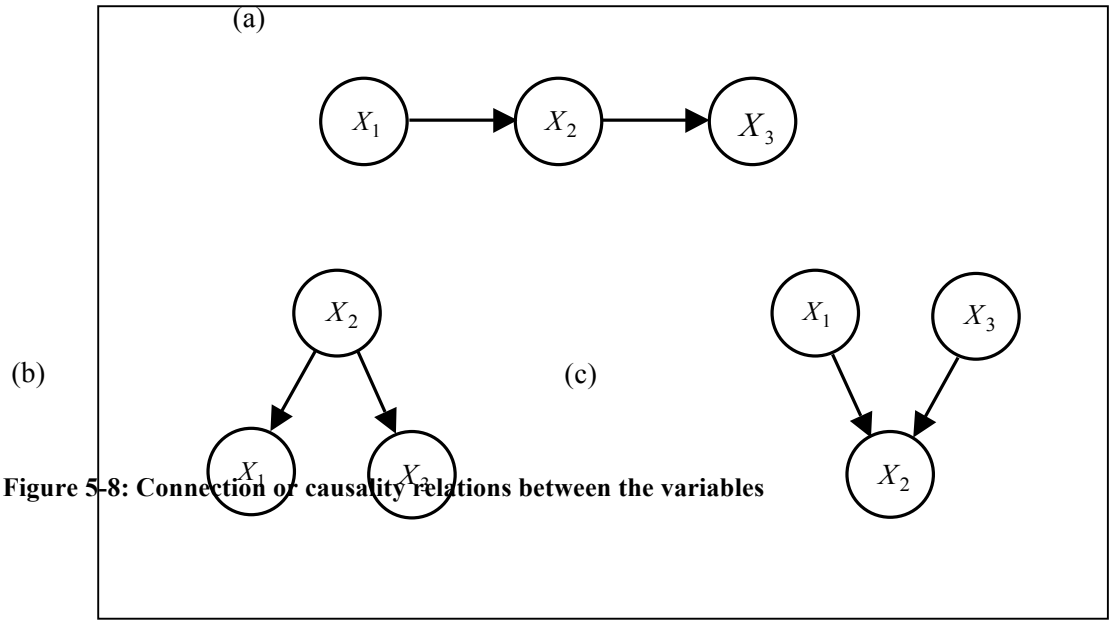


Figure 5-8: Connection or causality relations between the variables

The causality relation between variables helps us to determine how the changes in certainty of one (information) variable may cause changes in certainty of another (hypothesis) variable.

Conditional Probability Tables (CPTs)

For each variable A with parents $V_1, V_2, ..., V_n$, there is a conditional probability such as $P(A \mid V_1, V_2, ..., V_n)$ to provide the quantitative relation between the information variables and hypothesis variables.

The main mathematical point in applying a BN to a system is the rule that is known as *the chain rule*. Assume a BN over the set of $U = \{V_1, V_2, ..., V_n\}$. Then the joint probability distribution $f_U(u) = f(v_1, v_2, ..., v_n)$ can be written as:

$$f_U(u) = f(v_1, v_2, \ldots, v_n) = \prod_i f(v_i \mid pa(v_i)) \quad (1)$$

5.4.3 Methodology: Constructing BN Model for Predicting Appropriate Mapping Strategy

In order to develop a credible and validated model for modelling the mapping strategy selection decision-process, the author followed the guidelines described by (Jensen et. al., 2007; Darwiched 2009) to develop, test and iteratively revise the BN model to avoid potentially spurious or unreliable models. First of all, an initial parameterised BN model (named **alpha-level** model) was developed for mapping strategy decision-making. Secondly,

the BN model was tested by using various combinations of input values and sensitivity analysis to readjust the initial conditional probability tables (CPTs) or structure until it responded reasonably; the new updated model is called **beta-level model**. Finally, the BN model was evaluated with a series of case data to determine the accuracy of **beta-level** model for the purpose of handling missing probabilistic data on some nodes or states, and the eventual model produced is named as a **gamma-level model**. The following subsections describe the development and evaluation of the BN model in detail using this methodology.

5.4.3.1 Alpha-level Mapping Strategy Selection BN Model

By following the guidelines for developing the initial BN model, the structure of the model has several characteristics:

1. The number of parent nodes to any given intermediate node is kept to three or fewer and the number of their states to three. This keeps the associated CPT small enough to be tractable and understandable, since the CPTs are specified by our domain knowledge and empirical assumptions.
2. The input nodes are selected – typically representing ontology, application, and environment state factors, and can be preprocessed or empirically evaluated.
3. According to the influence diagram, the intermediate nodes (e.g, LoadtimeOverhead node) are designed to summarise the major themes that influence on strategy selection.
4. The depth of the model is kept to small (i.e. four here). This is desirable for three reasons: a shallow model with fewer intermediate nodes can avoid unnecessary uncertainty propagation from input to output nodes; the sensitivity of the output node to input nodes cannot be swamped by fewer intermediate nodes; and output nodes in the shallow model is far more sensitive to less distant input nodes.

Figure 5-9 depicts the initial parameterised Bayesian network model designed based on the influence diagram of Figure 5-7. The initial BN for deciding appropriate mapping strategy includes three categories of data input: *ontology, application,* and *environment* characteristics. The *ontology* characteristics inputs (e.g., RoutOnt_size, Required_ReferOnt, MappingOnt_size, RoutOnt_express, Required_ReferOnt_express nodes) were important components of the mapping strategy selection model and worked with the *environment* characteristics inputs (e.g., NumberofMappingOnts, NumberofReferOnts nodes) to predict the overhead cost of KBNMap router's *Loadtime* computation. Some of the *application* characteristics inputs (e.g., PubRate, SubRate, UnsubRate, ObservedUnknownDataRate nodes) and environment characteristics inputs (e.g., NetworkScale node) are used to define

the probability of rate of unknown data occurrences in a router. Some of the application characteristics inputs (e.g., FaultTolerance, MismatchedTolerance nodes) are used to predict the level of applications' robustness against false-positive and false-negative subscription matches due to the overly conservative loading of mappings. Finally, the environment characteristics inputs (AvailableMemory node) and ontology characteristics relevant intermediate node (mergedOnt_size node) are used to determine memory resources available to the router once mapping are loaded. All inputs were summarised to produce the posterior probability values of the final strategy selection node (Strategy_Selection node) for each mapping strategy, in which the strategy with the highest value represents router would use this strategy to deal with heterogeneous information.

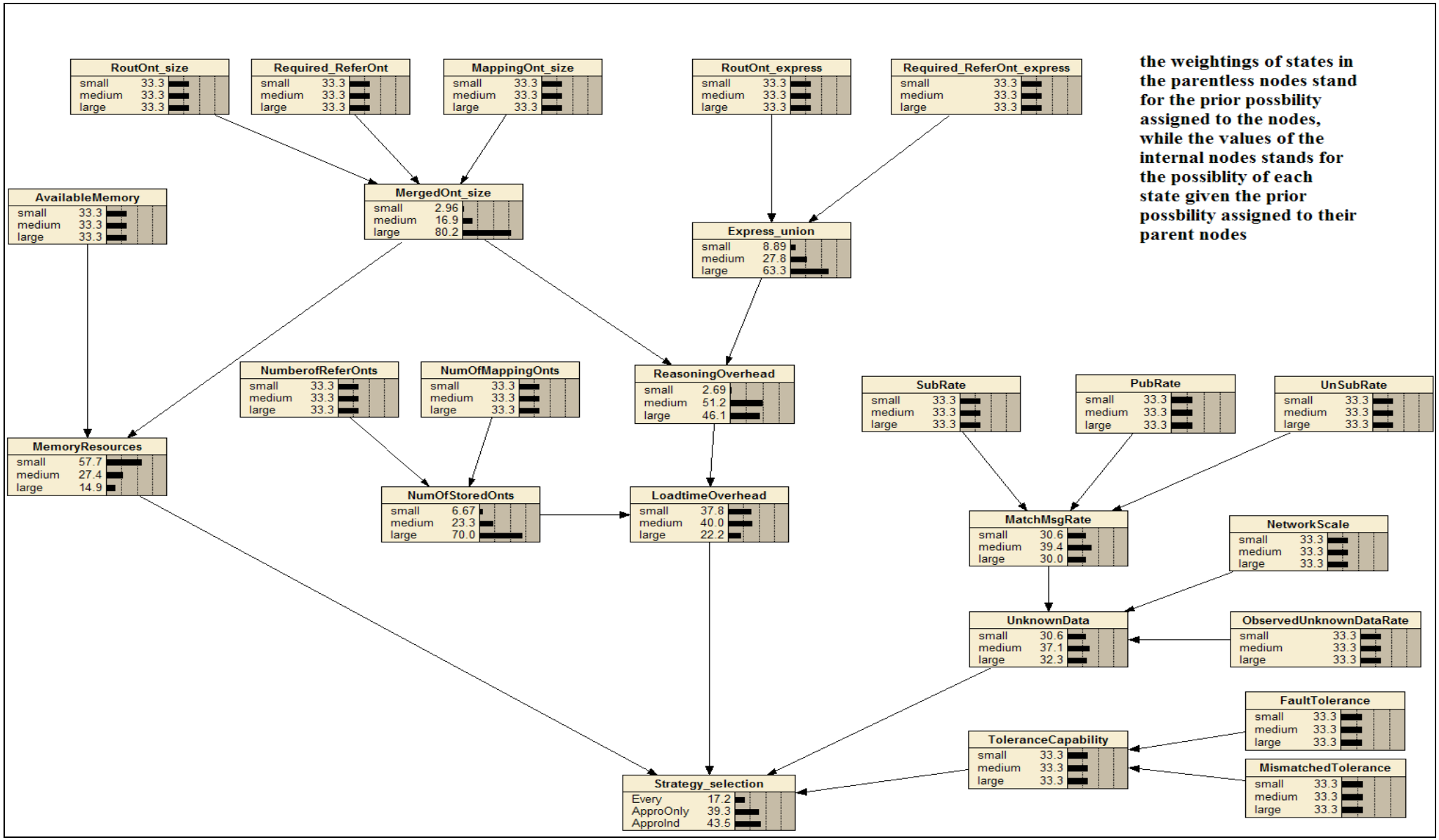


Figure 5-9: BN Model (Alpha_version) illustrating the favourable combinations of influential factors for selecting mapping strategy

5.4.3.2 Node Configuration

The aim of the BN model described above is to promote an understanding of what would happen to mapping strategies selection under different *ontology, application,* and *environmental* trigger factors. For this aim, the specifications of states of each node, the configuration of prior possibility of input node and the population of CPTs for intermediate nodes and output nodes is required to be configured. The following subsections discuss these configurations.

Node State Configuration

It is known that several trigger factors have continuous values such as ontology size, memory resource, publication rate. However, BNs can only deal with continuous variables in a limited manner (Jensen et. al., 2007). The usual solution is to discretise the variables and build the model over the discrete domain (Darwiched 2009). Therefore each node (except the final strategy selection node whose states represent the selection of one mapping strategy, e.g., the **Every, ApproOnly, ApproInd** mapping strategies respectively) in the BN model is specified to have three categorised states: *small/low, medium, and large/high*, which is also in accordance with the specification of node in the influence diagram of Figure 5-7 in Section 5.3. The actual values of those nodes having numeric value (e.g.., RoutOnt_size node has value of 1,234 statements) are converted and categorised by the author into these three states based on the specific boundary values identified according to practical experimentation, knowledge of the implementation of KBNImpl, experience with different reasoning and load configurations, and subjective estimate of each node.

Table 5-8 illustrates the assignment of boundary values for states of each individual node. It should be noted that the threshold values specified below are all *dynamically* customisable for different KBNMap deployments in order to meet the different requirements of various tasks and the different views of individual domain experts. For convenience, based on the description of nodes in the influence diagram of Figure 5-7 (Section 5.3), the description and specification of each node in Table 5-8 is presented in the following subsections.

Table 5-8: The Assignment of Boundary Values for Node State

			State			
Node Name	**Node type**	**Measure indicator[iv]**	**Small**	**Medium**	**Large**	**Source of derived boundary value**
Ontology-based Nodes						
RoutOnt_size	Input	LT+ASE	NS≤1,000	1,000<NS≤5,000	NS>5,000	In Section 5.2.1
Required_ReferOnt	Input	LT+ASE	NS≤1,000	1,000<NS≤5,000	NS>5,000	In Section 5.2.1
MappingOnt_size	Input	LT+ASE	NS≤1,000	1,000<NS≤5,000	NS>5,000	In Section 5.2.1
MergedOnt_size	Internal	Derived	NS≤1,000	1,000<NS≤5,000	NS>5,000	In Section 5.2.1
RoutOnt_express	Input	LT+ASE	DL≤ALC	ALC<DL<SHION	DL=SHION	In Section 5.2.1
Required_ReferOnt_express	Input	LT+ASE	DL≤ALC	ALC<DL<SHION	DL=SHION	In Section 5.2.1
Express_union	Internal	Derived	DL≤ALC	ALC<DL<SHION	DL=SHION	In Section 5.2.1
Application-based Nodes						
PubRate	Input	PC	MH≤10	10<MH≤20	MH>20	Bechmark of KBNImpl (Keeney et. al., 2006b)
SubRate	Input	PC	MH≤10	10<MH≤20	MH>20	Bechmark of KBNImpl (Keeney et. al., 2006b)
UnSubRate	Input	PC	MH≤10	10<MH≤20	MH>=20	Bechmark of KBNImpl (Keeney et. al., 2006b)
MatchMsgRate	Internal node	Derived	MH≤10	10<MH≤20	MH>=20	Derived from its parent nodes
ObservedUnknownDataRate	Input	PC	MH≤3	3<MH≤7	MH>7	Knowledge of the implementation of the KBNMap router application
FaultTolerance	Input	LT+ Admin	small	medium	large	Our subjective hypothesis
MismatchTolerance	Input	LT+ Admin	small	medium	large	Our subjective hypothesis
ToleranceCapability	Internal	Derived	small	medium	large	Derived from its parent nodes
UnknownData	Internal	Derived	small	medium	large	Derived from its parent nodes
Environment-based Nodes						
NumberofMapingOnt	Input	LT+ASE+Admin	NI≤5	5<NI≤10	NI>10	Knowledge of the implementation of the KBNMap router application
NumberofReferOnt	Input	LT+ASE+Admin	NM≤5	5<NM≤10	NM>10	Knowledge of the implementation of the KBNMap router application
NumOfStoredOnt	Input	Derived	NM≤5	5<NM≤10	NM>10	Knowledge of the implementation of the KBNMap router application
NetworkScale	Input	LT+ Admin	NETS≤5	5<NETS≤15	NETS>15	Our subjective hypothesis
AvailableMemory	Input	LT+ PC	MR≤64mb	64mb<MR≤128mb	MR>128mb	Knowledge of the implementation of the KBNImpl router application
MemoryResources	Internal	LT+ PC	MR≤64mb	64mb<MR≤128mb	MR>128mb	Knowledge of the implementation of the KBNImpl router application
ReasoningOverhead	Internal	Derived	small	medium	large	Derived from (Lewis et.al., 2006)
LoadtimeOverhead	Internal	Derived	small	medium	large	Derived from (Lewis et.al., 2006)

[iv] Measure indicator means that the way to measure a particular node's actual value. **LT** indicates the input node is measured at network setup time; **ASE** indicates the input node is measured after strategy execution at runtime; **Admin** indicates the value of input node is specified by network administrator; **PC** indicates periodically Check by system.

A. Ontology-based Nodes

RoutOnt_size: refers to the current size, measured as the number of statements, of the current routing ontology. This is input variable is automatically measured and maintained by the KBNMap router. Ontology models with less than 1000 statements are regarded as *small*, 1000 to 5000 statements are *medium*, and models with greater than 5000 are regarded as *large*. These boundary values are based on the experiments in Section 5.2.1 and Appendix B.

MappingOnt_size: refers to the size of mapping ontologies stored at the router, again measured as the number of statements, ranging in size from *small* (<1000), *medium* (>1000, <5000), and *large* (>5000).

Required_ReferOnt: refers to the size of the ontologies referenced by the mapping ontologies, again measured as the number of statements, ranging in size from *small* (<1000), *medium* (>1000, <5000), and *large* (>5000).

MergedOnt_size: is the probability that a newly merged routing ontology incorporating mappings will be small (<1000), medium (>1000, <5000), or large (>5000). This is conditional on the "RoutOnt_size", "Required_ReferOnt", "MappingOnt_size" nodes.

RoutOnt_express: refers to the reasoning complexity or DL expressiveness of the routing ontology. Ontologies with an expressivity of *ALC* or less are regarded a *low* expressivity, with ontologies ranging from *ALC* to *SHOIN* being regarded as *medium* expressivity, with *SHOIN* ontologies being considered highly *(high)* expressive. These boundary values are based on previous experiments in Section 5.2.1 and Appendix B. Like size, the expressivity of an ontology can be found by loading the ontology and then querying the ontology reasoner, however, with the aim of minimising the loading of ontologies these values can also be found from lookup tables.

Required_ReferOnt_express: refers to the DL expressivity of the ontologies referenced by the mappings, again ranging between *low* (<*ALC*), *medium* (>*ALC*, <*SHOIN*), and *high* (*SHOIN*).

Express_union: is the probability that the expressivity of a newly merged routing ontology incorporating mappings will be low (<*ALC*), medium (>*ALC*, <*SHOIN*), and high (*SHOIN*). This is conditional on the "Required_ReferOnt_express" and "RoutOnt_express" nodes. The DL expressivity of a merged ontology is the union of the expressivity of its constituent ontologies.

B. Application-based Nodes

PubRate: refers to the average rate at which publications arrive at router/broker. A rate of 10 or less publications per minute is considered *low*, 10 to 20 is considered *medium*, while more than 20 is considered *high*. This is a subjective ranging of boundary values, cognisant of the normal operation and loading capability of an average KBNMap router, especially when combined with normal

simultaneous loadings from subscription and unsubscription requests. The observed input value for this variable is calculated and maintained automatically by the KBNMap router application.

SubRate: refers to the average rate at which subscription requests arrive at the router, ranging between *low* (<10/min), *medium* (>10/min, <20/min) and *high* (>20/min).

UnsubRate: refers to the average rate at which unsubscription requests arrive at the router, ranging between *low* (<10/ min), *medium* (>10/min, <20/min) and *high* (>20/min).

MatchMsgRate: is the probability that the rate of publication to subscription matching rates will be low, medium or high. This is conditional on the "PubRate", "SubRate", and "UnsubRate" nodes.

ObservedUnknownDataRate: refers to the average rate per minute at which an unrecognised ontological concept (or individual, or property) is detected at a KBNMap router. Where unknown data can occur in a publication, subscription or unsubscription we estimate that 3 occurrences per minute is *low*, 3-7/minute is *medium*, and more than 7/minute is *high*.

NetworkScale: refers to the number of routers comprising a KBNMap network. It is impossible to define concrete boundary values for this variable since it is inherently subjective. It is estimated that a deployment with 5 routers is *small*, 5 to 15 routers is *medium*, and more than 15 is *large*. However, since this refers to just the number of routers, not the number of clients attached to any routers, 15 routers have the potential to support several thousand clients in a scalable manner.

UnknownData: is the probability that the aggregate rate of unknown data occurrences will be low, medium or high. This is conditional on the "MatchMsgRate", "NetworkScale", and "ObservedUnknownDataRate" nodes.

ToleranceCapability: refers to the tolerance the KBNMap clients may have when dealing with false-positive or false-negative subscription matches due to potentially missed semantic relationships due to the overly conservative loading of mappings. It is further divided into FaultTolerance and MismatchedTolerance nodes. This variable is purely subjective, and attempts to define boundary values to the *low, medium and high* states is out of scope here. The encoding of this variable is delegated to an administrator. It is envisioned that all routers in a KBNMap might have the same value for this variable.

C. Environment-based Nodes

NumberofMappingOnts: refers to the number of mapping ontologies stored at a router. Less than 5 mapping ontologies is considered *small*, 5 to 10 mapping ontologies is *medium*, with more than 10 being considered *large*. These boundary values are based on the empirical analysis and the experience with different reasoning and load configurations in the previous experiments (Section 5.2.1 and Appendix B).

NumberofReferOnts: refers to the number of ontologies referenced by the mapping ontologies, again ranging between *small* (<5), *medium* (>5, <10) and *large* (>10).

NumOfStoredOnts: is the number of ontologies available on a KBN router, ranging between *small* (<5), *medium* (>5, <10) and *large* (>10). It is derived from the "NumberofMappingOnts" and "NumberofReferOnts" nodes.

AvailableMemory: refers to the memory allocation given to the KBN router application. An allocation of 64MB or less is considered *small*, 64MB to 120MB is considered *medium*, with greater than 120MB being considered *large*. These boundary values are based on the knowledge of the implementation of the KBN router application, the experience with different reasoning and load configurations, and the availability of memory resources on mid-range developer workstations. The value for available memory can be detected from the runtime environment within which the KBNMap router application runs.

MemoryResources: is the probability that the amount of further memory resources available to the KBN router once mappings are loaded will be *small* (<64MB), *medium* (>64MB, <120MB), or *large* (>120MB). This is conditional on the "AvailableMemory", "MergedOnt_size" nodes.

ReasoningOverhead: is the probability that the amount of reasoning of a merged ontology will be low, medium or high. This is conditional on the "MergedOntSize", "ExpressUnion" nodes. The weightings and rankings of input nodes to produce probability weighting for this node is derived from our previous work in determining the main factors that influence reasoning overhead (Section 5.2.1, Appendix B).

LoadtimeOverhead: is the probability that the amount of loadtime initialisation and reasoning of a merged ontology will be low, medium or high. This is conditional on the "ReasoningOverhead", "NumOfStoredOnts" nodes. The weightings and rankings of input nodes to produce probability weighting for this node is derived from our previous work in determining the main factors that influence reasoning overhead (Section 5.2.1, Appendix B).

D. Final Strategy Selection Node

The **Strategy_selection** node then provides a set of weightings which ranks the three semantic interoperability strategies, thereby deciding the appropriate mapping strategy. This is conditional on available memory resources ("MemoryResources"), ontology loadtime initialisation overhead ("LoadtimeOverhead"), tolerance ("ToleranceCapability"), and the unknown data occurrence rate at the router ("UnknownData"), each of which continuously change. The rankings between inputs are discussed in Section 5.4.3.4.

5.4.3.3 Prior Possibility and Initial CPTs Population for BN Node

Prior Possibility

The heart of the BN model consists of the set of possibility tables underlying each node. As stated in the previous section, the inputs nodes are structured as three categorical states, each of which is associated with prior probabilities. The prior probability can be assigned according to known frequencies of various states, or based on an assumed statistical distribution. According to the assignment method advised in (Darwiche 2009), a uniform distribution of states of each input nodes (33.333% of each state) is specified, since it represents complete uncertainty and summarises all changing situations of ontology, application and environment trigger factors that a KBNImpl router will encounter. For example, if the prior possibility for each state (*small, medium, large*) of routing ontology size (RoutOntSize node) is specified as 33.333%, it means that there will be equal probability that the KBNMap router will initially use either a *small*, *medium*, or *large* routing ontology.

Initial CPTs

To populate the CPTs of the intermediate nodes and output node, the conditional possibilities for the state of each intermediate (child) node are specified for all combinations of states of their parent nodes. Normally the size of the CPT of the child node is equal to the number of states of the child node times the product of the number of states of all parent nodes, or with n parents nodes. $S\prod_{i=1}^{n} p_i$ where S is the number of states of the child node and p_i is the number of states of the ith parent nodes. The CPT values are initially specified according to our domain knowledge and empirical assumption.

In line with the specification of boundary values of states for the nodes, the specification of CPTs for the nodes has also been derived from the empirical analyses, experiences, and subjective hypotheses of the author. It should be noted that although the author has calculated default weightings, rankings, influence paths, and boundary values for each of the variables, these configurations are all *dynamically* customisable for different KBNMap deployments.

As an example, Table 5-9 presents the conditional possibility table for the Strategy_Selection node, while the CPTs for the rest of the nodes are presented in Appendix E. It can be observed that the weightings of each strategy indicate which strategy is most appropriate for a given set of values for the input variables.

Table 5-9: Conditional Possibility Table for the Stategy_Selection Node

OutCome States (Strategy_selection)			Parent Node States			
Every	ApproInd	ApproOnly	LoadtimeOverhead	ToleranceCapability	UnknownData	MemoryResources
0.1	0.2	0.7	small	small	small	small
0	0.2	0.8	small	small	small	medium
0	0.1	0.9	small	small	small	large
0.4	0.4	0.2	small	small	medium	small
0.3	0.4	0.3	small	small	medium	medium
0.2	0.3	0.5	small	small	medium	large
0.7	0.3	0	small	small	large	small
0.6	0.3	0.1	small	small	large	medium
0.5	0.3	0.2	small	small	large	large
0.2	0.2	0.6	small	medium	small	small
0.1	0.2	0.7	small	medium	small	medium
0	0.1	0.9	small	medium	small	large
0.5	0.4	0.1	small	medium	medium	small
0.4	0.3	0.3	small	medium	medium	medium
0.3	0.3	0.4	small	medium	medium	large
0.8	0.2	0	small	medium	large	small
0.7	0.25	0.05	small	medium	large	medium
0.6	0.3	0.1	small	medium	large	large
0.1	0.3	0.6	small	large	small	small
0.1	0.2	0.7	small	large	small	medium
0.1	0.1	0.8	small	large	small	large
0.4	0.6	0	small	large	medium	small
0.35	0.55	0.1	small	large	medium	medium
0.3	0.5	0.2	small	large	medium	large
1	0	0	small	large	large	small
0.9	0.1	0	small	large	large	medium
0.8	0.15	0.05	small	large	large	large
0.05	0.45	0.5	medium	small	small	small
0.05	0.35	0.6	medium	small	small	medium
0	0.2	0.8	medium	small	small	large
0.3	0.5	0.2	medium	small	medium	small
0.2	0.5	0.3	medium	small	medium	medium
0.2	0.4	0.4	medium	small	medium	large
0.4	0.6	0	medium	small	large	small
0.35	0.55	0.1	medium	small	large	medium
0.3	0.55	0.15	medium	small	large	large
0.1	0.5	0.4	medium	medium	small	small
0.05	0.45	0.5	medium	medium	small	medium
0.05	0.4	0.55	medium	medium	small	large
0.2	0.6	0.2	medium	medium	medium	small
0.2	0.5	0.3	medium	medium	medium	medium
0.15	0.45	0.4	medium	medium	medium	large
0.35	0.65	0	medium	medium	large	small
0.3	0.6	0.1	medium	medium	large	medium
0.3	0.5	0.2	medium	medium	large	large
0.1	0.5	0.4	medium	large	small	small
0.1	0.45	0.45	medium	large	small	medium
0.1	0.4	0.5	medium	large	small	large
0.3	0.6	0.1	medium	large	medium	small
0.25	0.6	0.15	medium	large	medium	medium
0.2	0.5	0.3	medium	large	medium	large
0.45	0.55	0	medium	large	large	small
0.4	0.5	0.1	medium	large	large	medium
0.35	0.45	0.2	medium	large	large	large
0	0.3	0.7	large	small	small	small
0	0.2	0.8	large	small	small	medium
0	0.1	0.9	large	small	small	large
0.1	0.5	0.4	large	small	medium	small
0.05	0.5	0.45	large	small	medium	medium
0.05	0.45	0.5	large	small	medium	large
0.6	0.4	0	large	small	large	small
0.5	0.5	0	large	small	large	medium
0.35	0.4	0.25	large	small	large	large
0.1	0.3	0.6	large	medium	small	small
0.05	0.25	0.7	large	medium	small	medium
0	0.2	0.8	large	medium	small	large
0.3	0.6	0.1	large	medium	medium	small
0.25	0.55	0.2	large	medium	medium	medium
0.2	0.5	0.3	large	medium	medium	large
0.55	0.45	0	large	medium	large	small
0.5	0.5	0	large	medium	large	medium
0.4	0.5	0.1	large	medium	large	large
0.1	0.2	0.7	large	large	small	small
0.05	0.15	0.8	large	large	small	medium
0	0.1	0.9	large	large	small	large
0.2	0.7	0.1	large	large	medium	small
0.15	0.65	0.2	large	large	medium	medium
0.1	0.6	0.3	large	large	medium	large
0.7	0.2	0.1	large	large	large	small
0.65	0.25	0.1	large	large	large	medium
0.5	0.3	0.2	large	large	large	large

5.4.3.4 Updating Alpha-level BN Model to Beta BN Model

Updating BN Model Structure

There is an obvious need for improving the performance and accuracy of a Bayesian Network, along with the structure of the BN network. As mentioned in (Jenson et. al., 2007; Marcot et. al., 2001; Darwiche 2009), it is necessary to update the BN model structure in order to correct errors in model construction and cater for changes in the dynamics of the domains.

Update 1: Remove Reasoning_overhead intermediate node.

- **Reason**: This node can be replaced and merged by its child node LoadtimeOverhead, since its parent nodes MergedOnt_size and Express_union can be considered having direct causal relations over LoadtimeOverhead. The only difference between reasoning_overhead and LoadtimeOverhead is that the latter is also influenced by the number of ontologies stored in a router (e.g., NumOfStoredOnts node). Furthermore, current depth of the model is five, as the model should be maintained as four or fewer (Darwiche 2009). The reason for this is illustrated in Section 5.4.3.

Update 2: Merge FaultTolerance and MismactchTolerance nodes into ToleranceCapability node

- **Reason:** Both parent nodes refer to robustness capability of applications and are specified manually by the KBNImpl network administrator. Therefore, a single node which refers to the overall robustness capability to represent these two nodes can be used. Furthermore, according to the BN development guideline, it is efficient to keep the depth of network as shallow as possible. The sensitivity of Strategy_selection node to the tolerance branch would be improved, if there is just one ToleranceCapability node left.

Model Analysis

After re-constructing the BN model structure and updating the CPT of nodes, the next step was to verify the correct model structure and parameterisation in order to produce the Beta-level BN model. A *sensitivity analysis* (Jensen et. al., 2007; Darwiche 2009) was used to determine the absolute degree and the rank order of influence of parent variables on each outcome variable in the model. CPT values then were adjusted to make the model's sensitivity conform to our expected behaviors (e.g., the correct strategy that the KBNMap administrator expects to use).

5.4.3.5 Rank of Direct Parent Nodes

It is important to rank which parent nodes play the most important roles on the decision making of Strategy_Selection beforehand with the sensitivity analysis, so that it can be evaluated whether the final strategy selection node has been correctly predicted can be determined, given the mapping

strategy selection output given the input evidences of ontology, application and environment trigger factors. In general, the parent node with the highest entropy variation value normally has a greater influence on strategy selection that the other nodes.

As the BN model is used to facilitate the KBNMap router to select the appropriate mapping strategy in order to resolving heterogeneous data efficiently, the UnknownData node reflecting the rate of unknown data occurrences is ranked in first place for influencing the final strategy selection node. The different characteristics of mapping strategies determine how different mappings are merged into the routing ontololgy. For example, the **ApproInd** strategy enables the router to only load individual mappings into the routing ontology while the **Every** Strategy allows for loading the entire mapping files into the routing ontology. In addition, it is natural that the entire mapping files provide more complete and richer mapping knowledge than individual mappings, which significantly decreases the possibility of missed semantic-relations. Furthermore, it is described that the tolerance of KBNMap applications is used to deal with message mismatches due to potentially missed semantic relationships due to the overly conservative loading of mappings. In other words, it can be said that the tolerance capability of KBNMap applications highly influences the mapping strategy selection. Therefore, the ToleranceCapability node is ranked in second place. Furthermore, the experiment discussed earlier, and described in Appendix B, has revealed that the **Every** and **ApproOnly** mapping strategies are memory resource intensive. The amount of resources available for a router decides which strategy would be configured, so MemoryResources has been ranked by the author as third. Finally, the ontology characteristics exploration experiments in Section 5.2.1 indicated that the size and complexity (DL expressivity) of ontologies has strong connection with the reasoning overhead, which combining with the number of mapping ontologies and referenced ontologies reflects the Loadtime overhead produced by a router. Furthermore, considering the core of mapping strategies is to load appropriate mappings, proper mapping ontologies or all ontologies stored in a router to handle with unknown data, so the LoadtimeOverhead is important in strategy selection. However, as the router with larger resources can deal with large number of ontologies, therefore the LoadtimeOverhead is ranked behinds MemoryResource as fourth.

5.4.3.6 Sensitivity Analysis

As stated in (Jenson et. al., 2007), one of the main difficulties in modelling a decision problem is the elicitation of utilities and probabilities. This makes it desirable to be able to investigate how sensitive the solution is to variations in some utility or probability parameter, and how robust the solution is to joint variations over a set of parameters. *Sensitivity analysis* (Castillo et. al., 1997) is used to analyse how sensitive the conclusions (the probabilities of the hypothesis variables) are to minor changes. The changes might be as a result of variations of the parameters of the model or perhaps from changes of the evidence.

In this thesis, for the Bayesian Network model for strategy selection, *Sensitivity analysis* is used to analyse the sensitivity of the weightings for each of the mapping strategies given in the output node (StrategySelection) to evidence changes of parentless nodes and internal nodes (changes of categorised trigger factors).

Standard *sensitivity analysis* uses calculations of **Variance Reduction** (VR) when dealing with continuous variables and **Entropy Reduction** (ER) when dealing with discrete categorised variables. As all nodes in our model are categorical variables, the **Entropy Reduction** formula is used to calculate the sensitivity of the output node to each input node (Jensen et. al., 2007; Darwiche 2009): **Entropy reduction**, *I*, is the expected reduction in mutual information of the output variable Q having *q* states (measured in information bits) due to a finding at an input variable *F* having *f* states, and is calculated as

$$I = H(Q) - H(Q \mid F) = \sum_q \sum_f \frac{p(q,f)\log_2[P(q,f)]}{P(q)P(f)}$$

Where $H(Q)$ is the entropy of Q before any new findings and $H(Q \mid F)$ is the entropy of Q after new findings at node *F*. That means the higher the entropy reduction, the more sensitive the node *F* is, and the more important it is to influence node *Q*. Furthermore, **Belief Variance** is used to measure how much the beliefs at the output node *Q* will change if a finding is entered at the input node *F*.

The Netica (NeticaV4.0.8) Bayesian Network development software version 4.0.8 was used to conduct sensitivity analysis tests. Netica is a very widely used BN development software and has been broadly used by researchers in various research domains (Learner et. al., 2000; Matsuura et. al., 2004; Marcot et. al., 2006). The results of the *sensitivity analysis* of Strategy_Selection's direct parents and parentless nodes of the BN model are shown in the Table 5-10, and Table 5-11 respectively.

From Table 5-10, it can be seen that the UnknownData has the most influence on the Strategy_Selection node, while the others have less influence, as the UnknownData has greatest **Entropy Reduction** value, and the **Belief Variance** value denotes the belief values of Strategy_Selection changed at rate of 0.0666689, when the findings are entered at UnknownData node. The values of entropy reduction and belief variance showed in the table confirm that the model is correctly representing our expectations and judgements of ranking the direct parent nodes importance on strategy decision making.

Table 5-10: Sensitivity of Strategy_Selection due to findings at its parent nodes

Direct parent node	Entropy Reduction	Percentage	Belief variance	Percentage
UnknownData	0.46461	32.9%	0.0666689	18%
ToleranceCapability	0.02547	1.8%	0.0060490	1.64%
MemoryResources	0.01228	0.869%	0.0017261	0.467%
LoadtimeOverhead	0.00760	0.538%	0.001397	0.378 %

Table 5-11 indicates the influences of parentless nodes on the Strategy_Selection node. The ObservedUnknownDataRate has far greater influence on the output node than do the other input nodes, as it is the most important direct parent node of UnknownData node, which has been identified and verified as the most important trigger factor of the BN model. It can also be seen that the expressivity of ontology (Required_ReferOnt_express and RoutOnt_express) has the weakest impact upon strategy selection. This is due to its lesser impact on the LoadOverhead, of which the number of ontologies and ontology size plays most important roles. It can be also observed that the Strategy_selection node is less sensitive to those nodes that are further away from it. This also confirms the previous decision to remove the ReasoningOverhead node in updating the BN model to shorten the length of the network in order to improve the output node's sensitivity to input nodes.

Table 5-11: Sensitivity of Strategy_Selection due to findings at parentless nodes

Direct parent node	Entropy Reduction	Percentage	Belief variance	Percentage
ObservedUnknownDataRate	0.3531	25%	0.0562365	14.2%
ToleranceCapability	0.02547	1.8%	0.0060490	1.64%
AvailableMemory	0.005867	0.18%	0.0007934	0.214%
NetworkScale	0.002544	0.18%	0.0005673	0.153%
NumberofReferOnts	0.0006636	0.047%	0.000131	0.0354%
NumOfMappingOnts	0.0006636	0.047%	0.000131	0.0354%
MappingOnt_size	0.0002211	0.0156%	0.0000407	0.011%
RoutOnt_size	0.0002211	0.0156%	0.0000407	0.011%
SubRate	0.00008	0.00601%	0.0000170	0.0046%
PubRate	0.00008	0.00601%	0.0000170	0.0046%
UnSubRate	0.00003	0.00242%	0.0000069	0.00186%
Required_ReferOnt_express	0.00000	0.0%	0.0000002	0.0%
RoutOnt_express	0.00000	0.0 %	0.0000002	0.0%

To explicitly illustrate the sensitivity of output node Strategy_Selection to the findings of input nodes, samples of changing the value for the UnknownData node are shown with the corresponding resulting values of the output node are shown in the Figure 5-10.

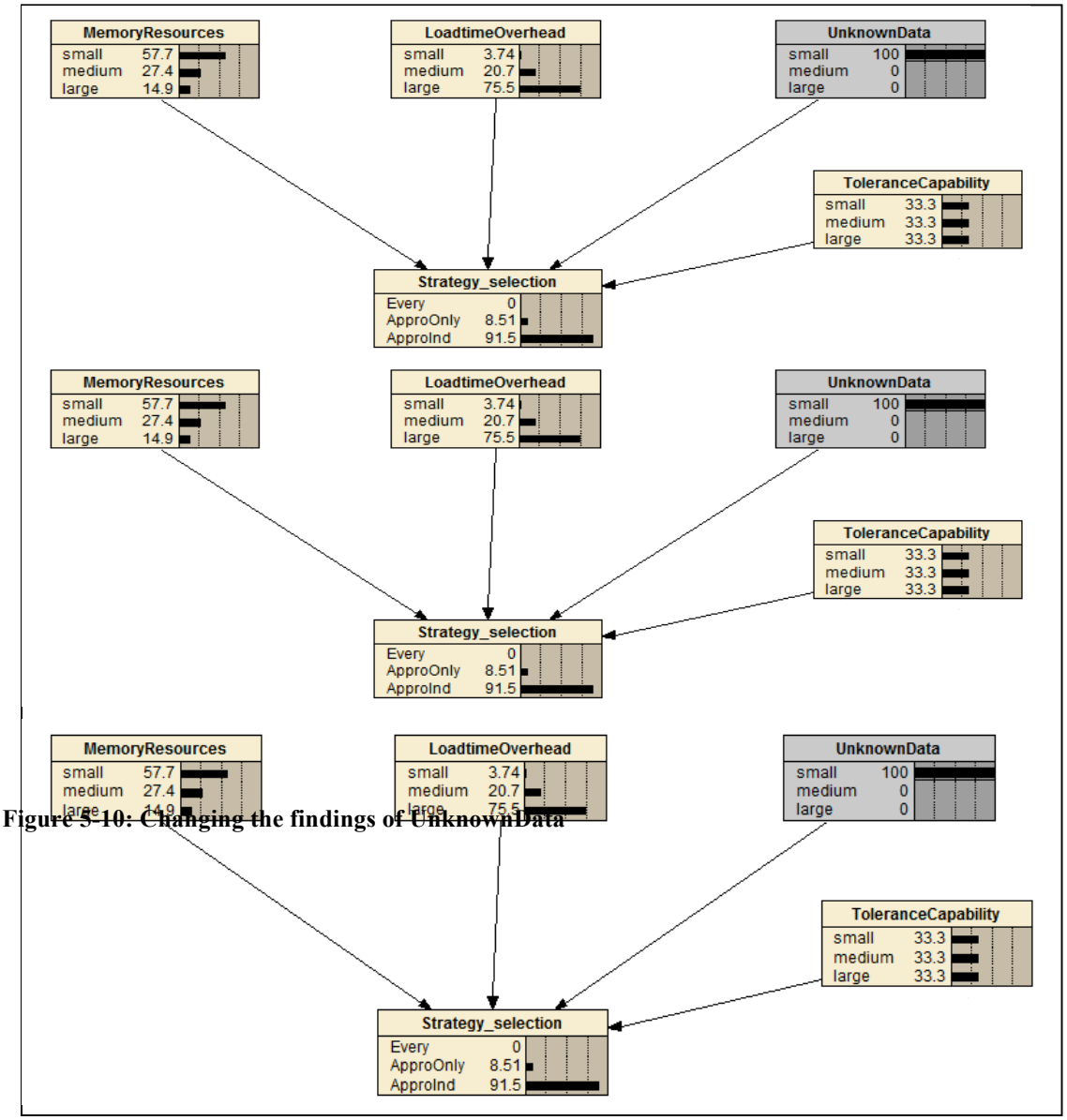


Figure 5-10: Changing the findings of UnknownData

After updating the initial BN Structure, prior possibility and CPTs of each node, and verifying the BN model inference through sensitivity analysis, the Beta-level model was created and shown in Figure 5-11.

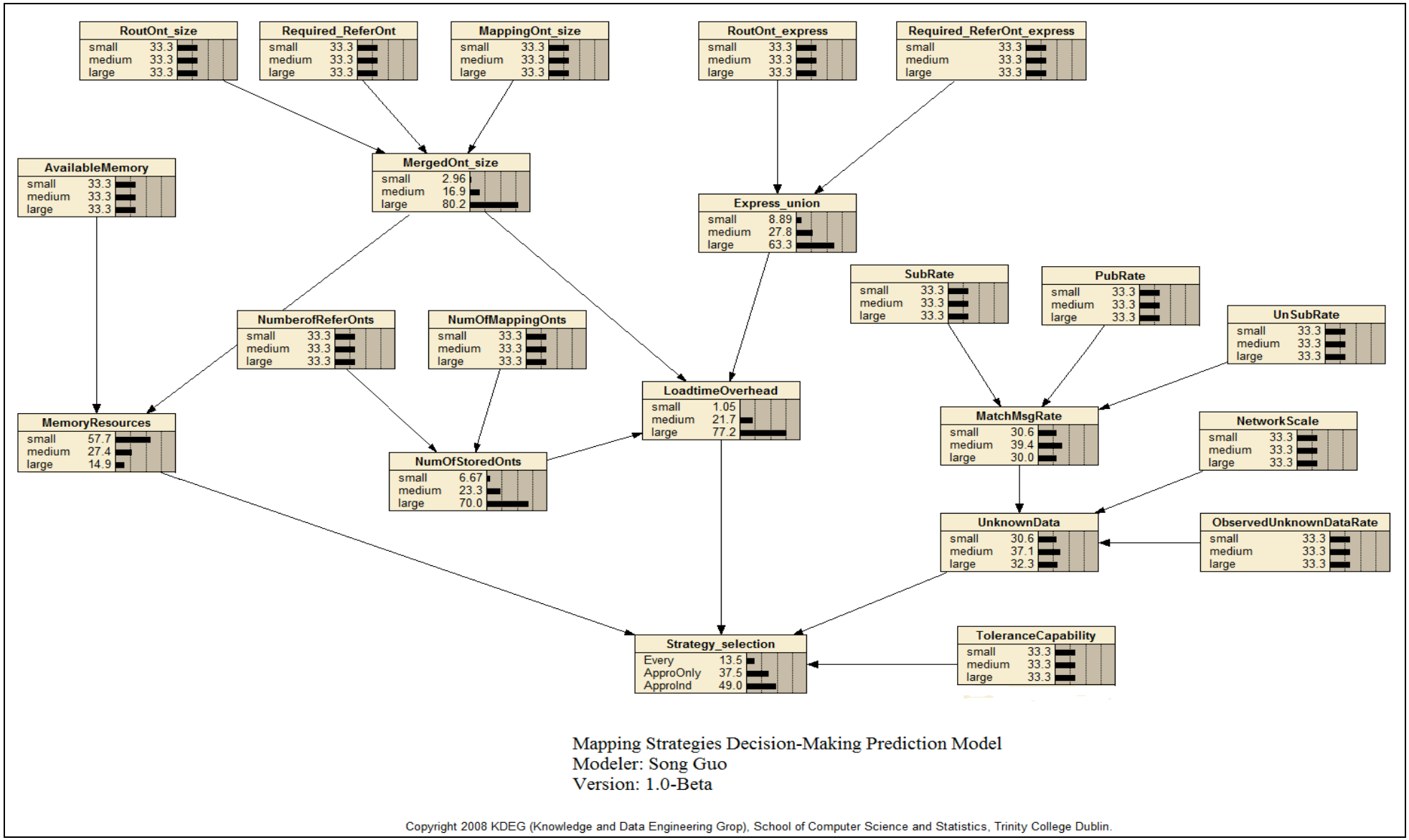


Figure 5-11: Mapping Strategy Decision BN Model (Beta_level)

5.4.4 Evaluating BN Model with Representative Cases

The purpose of this step is to validate the BN model using a set of representative test cases (specified in a test case file) to see how well the calculated predictions of the network model match the expected case results. The basic idea behind this evaluation is that the BN model is divided into two classes of nodes: the *observed* and *unobserved*. In the evaluation, the *observed* nodes are the ontology, application, environment-based input nodes while the *unobserved* node is the final strategy selection node (Strategy_Selection). The observed nodes will be "observed" in the sense that there values will be explicitly specified (i.e. from a case file for this experiment). These observed values are then used to predict the resulting values for the other unobserved node – Strategy_Selection by using Bayesian belief updating. This procedure is iteratively repeated for each case in the case file. For each individual case, the predicated values for the *unobserved* node are compared with those in the case file. All successes and failures are recorded and are used for validating the output of BN model.

The case files in this evaluation were generated with the help of the Netica software. Each case stands for the combination of states of both *observed* and *unobserved* nodes. For comprehensively testing the accuracy of the beta-level model and validating output node, 36 case files was generated which include all possible combinations of states of 14 trigger factors. The number of cases of each case file ranges from 10 to 200000 due to the maximum number of combinations of input nodes is above 150000. Figure 5-12 shows a case file sample:

IDnum	Express	MatchMs	PubRate	Toleran	Observe	Network	MergedO	Mapping	RoutOnt	Require	SubRate	UnSubRa	Unknown	Loadtim	MemoryR	NumOfSt	NumOfMa	Numbero	Availab	Require	RoutOnt_	Strategy_selection
1	large	medium	medium	medium	large	medium	medium	small	medium	large	large	large	large	medium	medium	large	small	medium	medium	small	medium	Every
2	medium	medium	small	small	medium	medium	large	large	small	medium	small	medium	medium	large	small	large	large	small	large	large	large	ApproOnly
3	large	medium	large	large	large	small	large	large	small	large	small	large	medium	large	medium	large	small	large	medium	small	small	ApproInd
4	large	small	medium	small	large	medium	large	small	medium	medium	small	small	large	large	small	large	large	large	medium	medium	large	ApproOnly
5	large	large	small	small	medium	medium	large	large	large	medium	large	small	large	large	medium	medium	small	small	medium	large	large	ApproOnly
6	large	medium	medium	medium	small	small	large	medium	large	small	large	medium	small	large	large	medium	small	medium	large	medium	small	ApproInd
7	medium	medium	large	medium	small	small	large	small	small	medium	large	medium	small	large	small	large	medium	large	large	medium	medium	ApproInd
8	large	large	small	small	small	small	large	medium	small	large	large	medium	small	medium	small	medium	small	small	small	medium	small	ApproOnly
9	large	large	large	small	small	small	medium	small	large	large	small	small	small	medium	medium	large	large	small	medium	medium	small	ApproInd
10	large	medium	small	small	small	medium	large	medium	large	medium	small	medium	small	large	small	large	medium	large	small	large	large	ApproInd
11	medium	large	medium	small	medium	small	large	large	small	medium	large	small	medium	large	small	large	large	large	small	large	small	ApproInd
12	large	large	large	small	large	small	large	medium	medium	large	medium	large	large	large	medium	large	medium	large	medium	medium	large	Every
13	large	large	medium	small	large	small	large	medium	large	medium	large	small	large	large	large	large	small	large	large	large	medium	Every
14	large	large	medium	medium	small	large	large	large	large	large	large	large	medium	medium	medium	medium	medium	small	medium	small	small	ApproInd
15	medium	medium	medium	large	large	large	large	large	medium	small	small	small	large	large	small	medium	medium	small	small	medium	large	ApproOnly
16	large	medium	medium	small	large	small	large	large	large	small	medium	medium	medium	large	medium	large	large	medium	large	large	medium	ApproOnly
17	medium	medium	medium	small	small	medium	large	small	small	medium	small	large	small	medium	medium	small	small	small	medium	medium	small	ApproOnly
18	large	medium	large	small	small	large	large	large	medium	large	small	small	small	medium	small	small	small	small	small	large	large	ApproInd
19	large	large	large	large	large	large	medium	small	large	medium	medium	small	large	large	small	large	large	large	small	small	medium	Every
20	large	large	large	large	large	large	medium	small	large	medium	large	small	large	large	small	large	large	large	medium	medium	medium	ApproOnly
21	small	large	large	small	small	small	large	large	small	small	small	small	small	large	small	medium	medium	medium	small	large	small	ApproInd
22	medium	medium	medium	small	small	large	large	large	medium	small	large	large	small	medium	small	large	medium	small	small	small	large	ApproOnly
23	medium	medium	medium	medium	large	large	large	large	small	medium	medium	small	large	large	small	large	small	medium	medium	small	large	Every
24	medium	medium	large	small	large	small	large	large	medium	small	medium	small	medium	large	small	large	medium	large	medium	medium	small	ApproOnly
25	large	medium	medium	large	medium	large	large	large	large	medium	medium	large	large	medium	small	medium	medium	small	medium	small	small	ApproOnly
26	large	medium	medium	medium	medium	medium	medium	small	large	small	medium	medium	medium	large	medium	large	small	large	medium	medium	small	ApproOnly
27	large	medium	small	large	large	small	medium	small	medium	large	small	small	large	medium	small	large	medium	large	medium	medium	medium	ApproOnly
28	medium	medium	large	large	medium	small	large	medium	small	medium	medium	large	medium	large	small	large	medium	medium	small	large	large	ApproInd
29	large	large	large	medium	medium	medium	medium	medium	large	small	medium	small	medium	large	medium	medium	medium	small	medium	medium	small	ApproOnly
30	large	large	small	small	medium	medium	large	small	large	medium	large	medium	medium	large	medium	large	large	medium	large	large	small	ApproInd
31	large	small	large	small	small	small	large	small	small	large	small	large	small	large	medium	large	large	medium	large	large	small	ApproInd
32	small	medium	small	medium	medium	small	large	small	small	small	small	large	medium	large	small	large	large	small	medium	medium	small	ApproOnly
33	large	small	small	large	small	medium	large	large	small	large	large	large	small	large	large	large	large	small	large	large	large	ApproInd
34	large	medium	small	medium	large	large	large	large	large	medium	medium	small	large	medium	medium	medium	small	medium	medium	large	medium	Every
35	large	medium	small	medium	medium	medium	large	large	large	medium	large	large	medium	large	small	large	small	large	medium	large	large	ApproInd
36	medium	large	medium	medium	medium	large	large	medium	medium	small	large	small	medium	large	medium	large	large	medium	medium	large	medium	ApproOnly
37	small	large	large	small	large	small	medium	small	small	small	medium	large	large	large	small	large	large	small	small	small	medium	ApproOnly
38	large	small	small	medium	medium	large	medium	medium	medium	medium	medium	small	medium	large	medium	medium	medium	medium	large	medium	small	ApproInd
39	large	medium	medium	small	medium	medium	medium	small	medium	large	medium	small	medium	large	small	large	small	large	large	medium	small	ApproInd
40	medium	large	large	large	medium	small	large	small	small	medium	medium	large	small	large	small	large	medium	large	medium	small	large	ApproInd

Figure 5-12: A sample of case file

5.4.4.1 Measurement Metrics

To facilitate validating the output of the BN model, the **Logarithmic** and **Quadratic Loss** (Jensen et. al., 2007; Darwiche 2009) metrics were used to analyse the results collected:

Logarithmic Loss: the logarithmic loss values were used to grade the predictive/diagnostic quality of a net with respect to a certain discrete node (Netica Tutorial). It is defined as: **MOAC [- log (p_c)]**

where **MOAC** stands for the average over all cases (e.g., all cases for which the case file provides a value for the output node: Strategy_Selection), and where **log(p_c)** is the natural logarithm of the probability predicated for the state that turns out to be correct.

Values for logarithmic loss vary from 0 to infinity, with 0 indicating the best performance of a BN model.

Quadratic Loss: it is also known as "Brier score" and is defined as:

$$\textbf{MOAC [1 - 2 * } p_c + \sum_{j=1}^{n} p_j^2 \textbf{]}$$

Where p_c is the probability predicted for the state that turns out to be correct, where p_j is the probability predicted for state j, and where n is the number of states of the node.

Values for quadratic loss vary from 0 to 2, with 0 being a perfect score of the BN model.

The Netica Bayesian Network development software (NeticaV4.0.8) was used to conduct this evaluation. The experiment time of BN node to deal with each case file was specified as 1 minute according to the guidelines of the Netica tutorial (Netica 2008).

5.4.4.2 Analysis of Evaluation Results

The performance results shown in Figure 5-13 and Figure 5-14 (the X-axis corresponds to number of cases, while the Y-axis is value of *logarithmic loss* and *quadratic loss* respectively) are modest, and revealed that the BN model that was designed by the author is stable in inferencing and is accurate enough to select the appropriate mapping strategy given a large number of cases, as the highest *logarithmic loss* value is less than 0.7 and the maximum value of quadratic loss is under 0.5. It is known that the logarithmic loss value less than 1 denotes the better performance of a BN model (Darwiched 2009). As for evaluating the accuracy of the BN model, it was observed that the BN model is sensitive when it dealing with the case files that have smaller number of cases (cases less than 100), whereas it is insensitive when

handling the case files that have larger number of cases (cases greater than 700). This indicates that the designed BN model exhibits better performance for dealing with larger number of cases within a short period. Therfore, it can be envisaged that the KBNMap router would be better capable of dynamically selecting the appropriate mapping strategy in a large scale environment where the triger factors are constantly changing.

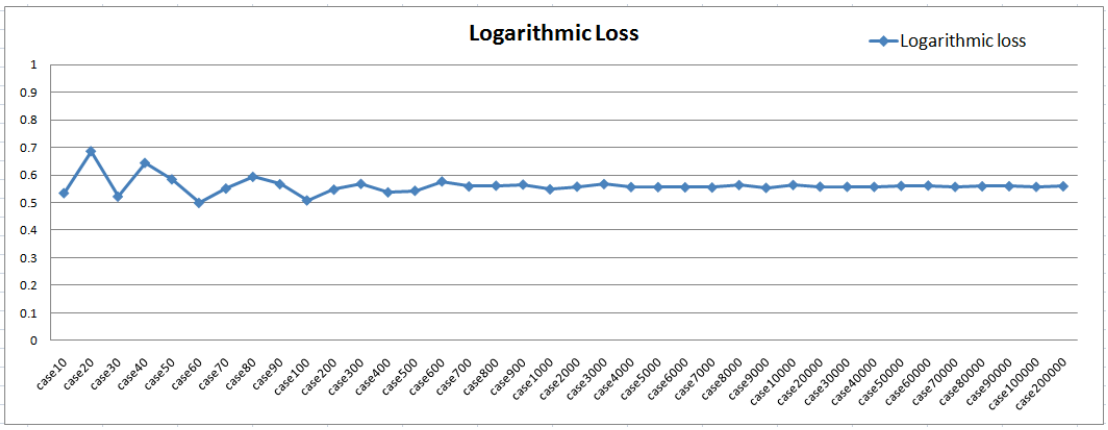


Figure 5-13: the logarithmic loss value of BN model

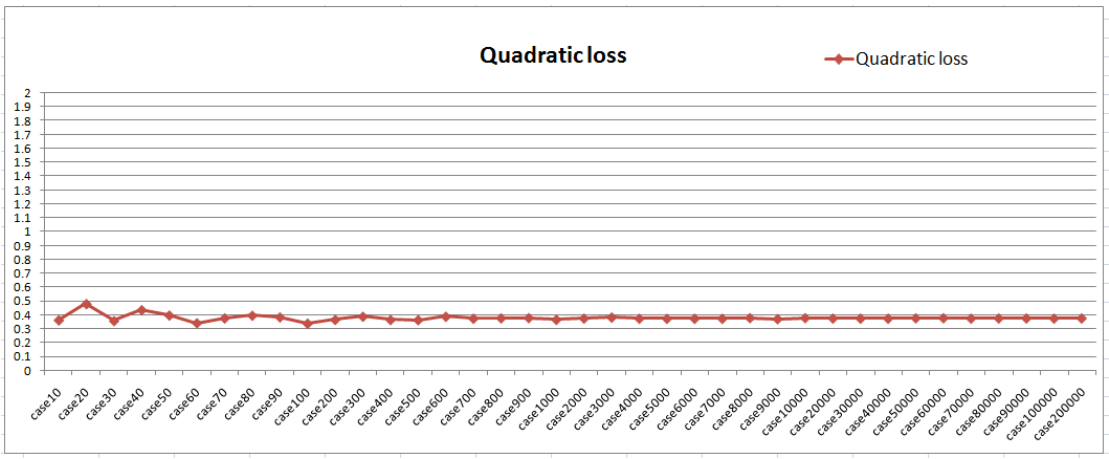


Figure 5-14: the Quadratic loss value of BN model

5.5 Entire Process

Figure 5-15 depicts the entire process of how to measure data, translate the values into evidence, and how the BN model facilitates the mapping enhanced KBNImpl router to select the appropriate mapping strategy for resolving heterogeneous unknown data. This design caters for the challenge of "Adaptation to Changes" in Section 3.6. In other words, the KBNMap system could self-select the mapping strategies and dynamically adapt to changes key trigger factors.

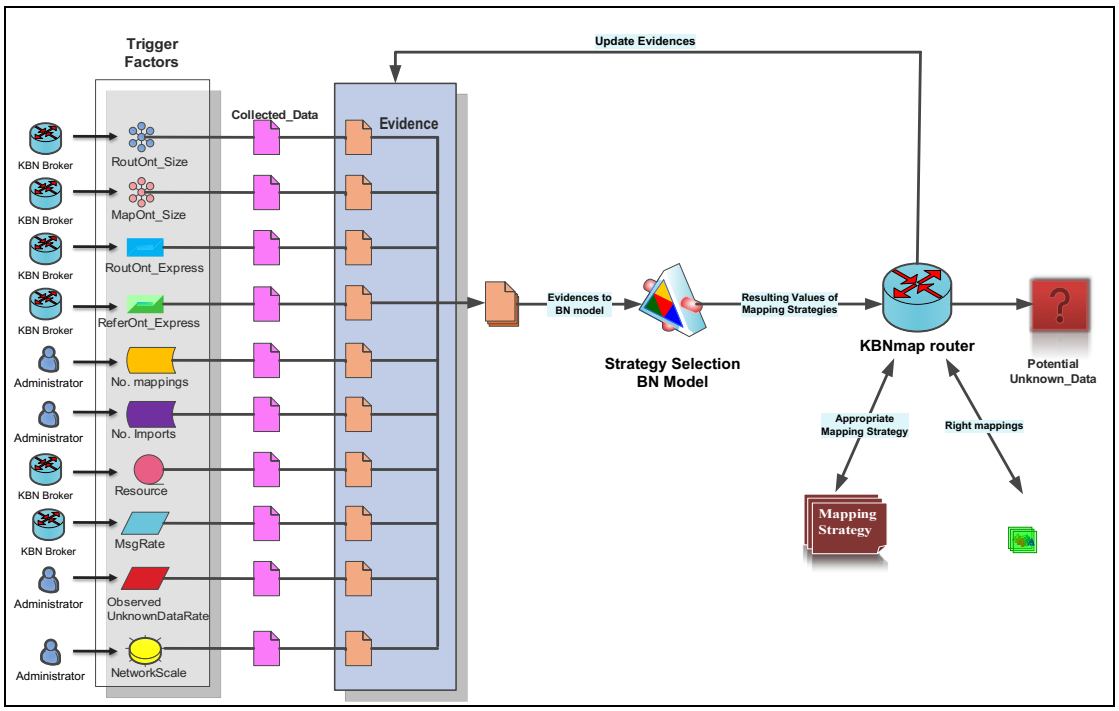


Figure 5-15: Process of adaptively selecting mapping strategy in KBNMap router

In order to address this challenge, a key design goal was to enable the KBNMap router periodically to collect data, translate data into evidence and query the designed Bayesian Network model to select the most appropriate strategy for dealing with potential unknown data. This is a continuous background process in the KBNMap router, e.g., the KBNMap router itself is responsible for periodically monitoring and collecting the values of the majority of the trigger factors, while the values of the other factors are collected and specified by the network administrator. Furthermore, the KBNMap router is also responsible for translating the collected data into relevant evidence, and periodically entering the translated evidence into the BN model to be integrated with the KBNMap router. According to the injected evidence, the BN model will automatically infer the weighting values for each mapping strategy, and forward the inferred results to the KBNMap router. The KBNMap router will then use the mapping strategy with the heaviest weighting if an unknown ontological term is encountered. Therefore, the active strategy of the router can be flexibly and continuously tuned in accordance with the output results for mapping strategies calculated by the BN model. Once the selected strategy is triggered to deal with unknown data, the KBNMap router will again present any new evidence values for the trigger factors it has measured to the evidence translator to trigger the next strategy weighting cycle. Since the BN is relatively small, the weightings calculation (as opposed to strategy execution) is very fast and consumes very little resources in the router, especially where this process executes only periodically as a low priority background task.

5.6 Conclusion

The research undertaken and described in this chapter has helped to clarify and identify which ontology, application and environmental influential factors are important for strategy selection, and which are not. The experimental method of identifying ontology-based influential factors indicates that the combination of size and expressivity of ontologies have significant impact on which strategy is appropriate at any one time. Determining which application and network environment characteristics are important has been more straightforward and has also been outlined in this chapter.

In a small scale or enterprise scale scenario it may be possible to measure the size and complexity of mapping ontologies or referenced ontologies that are chosen to be merged, in order to examine the application running over the KBNMap to *statically* determine which strategy is most appropriate. However, in a large-scale deployment, or where the characteristics of applications using the KBNMap change, then it is necessary to dynamically manage and adapt the selection of the most appropriate strategy. Thus a probabilistic-based mechanism for selecting the most appropriate mapping strategy was designed in this chapter, enabling the KBNMap routers to dynamically configure their mapping strategy based on the characteristics of the ontologies, application and environment.

In summary, in this chapter the mechanism for adaptively selecting mapping strategies has been discussed in detail. First, the difficulties of selecting mapping strategies statically were presented. Then, the adoption of a probabilistic-based modelling approach to facilitate the KBNMap router to make decision with respect to mapping strategy selection was motivated. A detailed discussion of using Bayesian Network approach to model, reason and predict the appropriate mapping strategy according to the changes of trigger factors was then presented, including the methodology for constructing, training and evaluating the BN model. Finally, how the semantic mapping enhanced KBNImplKBNMap router presented in Chapter 4 is enhanced with the designed BN model KBNMaphas been detailed.

In the next chapter, the implementation of KBNMap is presented; especially the implementation of KBNMap's components supporting heterogeneity and adaptive strategy selection..

6 IMPLEMENTATION

In Chapter 4 and Chapter 5, the design of KBNMap is presented and discussed. Those chapters have shown how to support multiple semantic models in a manner that is adaptive to changes of ontology, application, environment influence factors. In this chapter, a prototype implementation of KBNMap's components supporting these functionalities is presented. During the implementation of the design, the following challenges identified in section 3.6 were addressed:

1. The system should support multiple semantic models coexisting in an SBPS system;
2. The system should enable an SBPS system to dynamically configure its semantic interoperability service to adapt to changes in the set influential trigger factors identified in Chapter 3.

The scope of the implementation was to support the KBNMap system to make use of a Bayesian Network model to adaptively select the appropriate mapping strategy, and use the selected mapping strategy to load and merge appropriate semantic mappings/models so that semantic heterogeneity can be addressed.

6.1 Introduction

This chapter describes the implementation of the KBNMap, which is an extended KBNImpl architecture integrated with the designed adaptive semantic mapping service. The architecture of KBNMap which is presented in Figure 4-2 is shown again in Figure 6-1 for reference. KBNMap is implemented in Java and runs as a standalone SBPS router application. KBNMap makes use of external libraries which provides functionality that is not specific to KBNMap. These libraries include the Siena Content-based Network, the Jena library, the Pellet reasoner, and the JavaBayes Bayesian Network library.

The implementation of KBNMap took place over a period of approximately six months. During this period modifications and additions to KBNImpl's code base enabled the integration of KBNMap's additional functionality (e.g., hooks to identify unknown semantic data, the Bayesian inference engine, the KBNImpladaptive semantic mapping service, etc).

The implementation of each component of KBNMap is discussed in the following subsections. Section 6.2 briefly introduces the technologies used in the implementation. The registration interface is described in Section 6.3. This interface is used by network administrator to

register the routing ontologies and mapping files. The URLs of ontologies and mapping files are stored in the model- and mapping-stores respectively. Section 6.4 describes the adaptive semantic mapping service block. This semantic mapping service block consists of an adaptive strategy selection engine, an evidence processor, a Bayesian Network model, and three semantic mapping strategies. The evidence processor is used to translate the continuous values of observed trigger factors into the related states (evidence) to be passed into the Bayesian Network. The adaptive strategy selection engine is used to select the appropriate mapping strategy according to the evidence entering into the BN model. Finally the mapping strategies are used to facilitate the KBNMap router to resolve unknown data. A simple example of using KBNMap system is then demonstrated in Section 6.5.

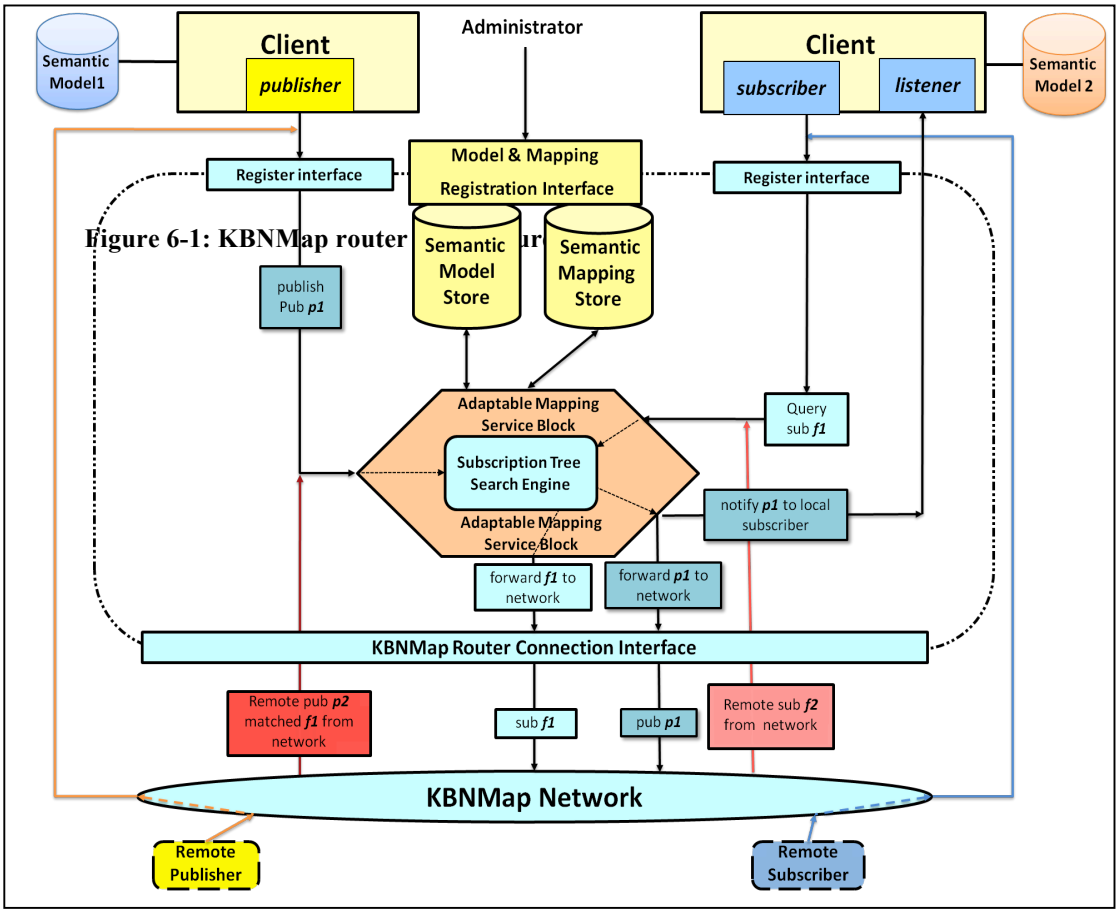


Figure 6-1: KBNMap router

6.2 Technology Used

This section provides an overview of the technologies that were used in the implementation of the KBNMap architecture.

The following open source Java libraries have been used:

- Jena 2.3 is a java framework for building Semantic Web applications (Jena 2005). The KBNMap routers use the Jena library to load the ontologies into a model in memory, so that the KBNMap routers can access and query the ontologies, using various method calls available from the Jena library. A detailed description of Jena and how applications/routers use Jena to access and query ontological information of an ontology can be found in Appendix A.
- Pellet 1.3 is a Java based OWL DL reasoner (Parsia et. al., 2004). The KBNMap routers use the Pellet reasoner to reason, validate, and check the consistency of the loaded ontologies by interfacing with the Jena library. A detailed description of Pellet reasoner can be found in Appendix A.
- Siena 1.5.5 is the Java version of Siena Content-based Network (Carzaniga 2008). The Siena library, which is extended as shown in (Keeney et. al., 2007) to implement KBNImpl, is further extended to create the KBNMap routers/clients, and make connections between clients/routers.
- JavaBayes is a Java based system for the creation and manipulation of Bayesian Networks (JavaBayes 2001). The JavaBayes is used for the KBNMap routers to create a Bayesian Network inference engine, so that the routers can reason, manage and query the BN model developed in Chapter 5. Detailed information about JavaBayes can be found in Appendix A.

The following tools have been used to assist the development:

- Eclipse Platform 3.2 is an integrated development environment for Java (Eclipse 3.2). The core part of KBNMap is developed in the Eclipse.
- Omondo 3.3 is a Java based UML development toolkit (Omondo 3.3). The UML diagrams presented in this Chapter are created by Omondo.
- Swoop is a Semantic Web Ontology editor (Swoop 2004). This tool is used for the author to initially test and edit the collected ontologies, in order to ensure the quality of ontologies used for the experiments, and to measure the DL expressivity of the ontologies.
- Exchanger XML Lite 3.2 is a XML editor (XML_lite 3.2). This tool is used to create and edit the data translation table, configuration table (section 6.4.2), as well as the BN model in XML format (Section 6.4.4).

rtnClient siena = new rtnClient(tcp:127.0.0.1:3000, String ontologyName)

6.3 Registration Interfaces and Model Store

6.3.1 Client Register Interface

KBNMap adopts the same mechanism as the KBNImpl for registering clients with the KBNMap router. KBNMap clients use the ThinClient implementation of KBNImpl/Siena which functions as a connection to the KBNMap router. In the simplest case, a ThinClient uses the URI of the KBNMap router to connect. For example, if the address of a KBNMap router is: "tcp:127.0.0.1:3000", the Client can uses the ThinClient constructor to connectKBNMap.

Once the client connects to the KBNMap router, it can forward subscription queries or publications to the router. The detail of how the client creates subscription/publication and forwards the message to the KBNMap router is given in the Section 6.5.

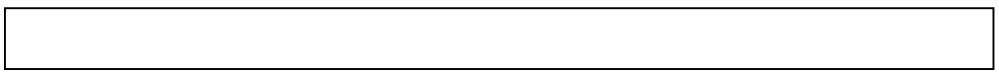
6.3.2 Router Connection Interface

Similar to KBNImpl, KBNMap routers can be connected to form a distributed event service. A distributed topology of routers is constructed incrementally. In practice, every router can connect to another router at startup by using URIs. For example: the following commands create a 3 level router hierarchy with 4 nodes:

1. *start the first router*: on host *foo*, by executing:
   java siena.StartServer -port 2345
2. *start a second router: on host bar, as a sub-router to the one on host foo*
   java siena.StartServer -port 2345 -master tcp:foo:2345
3. *start the third router: on host xyz, as a sub-node to the level-two router on node bar:*
   java siena.StartServer -port 2345 -master tcp:bar:2345
4. *start the fourth router: on host abc, as a sibling node to the third router*:
   java siena.StartServer -port 2345 -master tcp:bar:2345

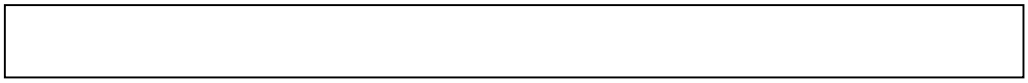
6.3.3 Model Registration Interface

The registration interface is used to store routing ontologies in the semantic model store. The registerRoutingOntology() method stores the routing ontologies then calls the method loadOntology(), which enables the Pellet reasoner embedded in the KBNImpl router to load the ontologies into an ontology model object that is instantiated by Jena (see Appendix A) in memory and reason the registered routing ontologies.



public String registerMapping (String mappingUri, String mappName)
public String listMappingOntology ();

Similarly, the registerMapping () method allows the KBNMap administrator to store the mapping files in the mapping store. It should be noted that for registered mapping ontologies, it is not necessary to call KBNImpl router to load them, as they will be discovered and called by the mapping strategies as required.



In order to display which ontologies and mappings have been loaded by the router, the listRoutingOntologies() and listMappings() methods are supported. These methods return a newline-separated list of routing ontologies and mappings respectively.

6.3.4 Ontology/Mapping Model Stores

The semantic model and mapping stores are implemented simply using any type of text documents, as only the URLs of ontologies and mapping files are recorded in the model- and mapping-stores in a newline-separated list order. These documents (stores) allow the KBNMap router to easily access the model stores without a externally accessible interface, as the model stores do not need to be accessed by any external applications. This saves resources and simplifies the administration of the KBNMap router.

The model stores are implemented as text document and named RoutingOntologyStore.txt and MappingStore.txt respectively. The stores are stored on the KBNMap's local filesystem, allowing the URLs of ontologies and mappings to persist between executions of the KBNMap router. In addition this also enables the network administrator to manually manage the stores.

Strengths/Weaknesses/Limitations

The advantage of storing the URLs of ontologies rather than actual ontologies in the model store is that it allows the KBNMap router to retrieve the appropriate mapping file easily. As the mapping strategies make use of OntModel interface of Jena to only read ontologies (by passing the URL of ontologies) without loading and reasoning, this interface, along with the mapping discovery algorithms implemented by the mapping strategies, enable the KBNMap router to quickly find the appropriate mappings as demonstrated with a series of experiments in this thesis (e.g., initial mapping strategies experiment in Appendix C). Furthermore, storing URLs of ontologies saves resources for KBNMap routers and helps to distribute mappings:

- Considering the nature of a large-scale deployment where the number of mappings and reference ontologies available might be large, storing this large number of

ontologies would inevitably require a large space within the KBNMap routers.

- In addition, as stated in Section 4.3.4, the KBNMap router could itself make use of subscriptions/publications to look for or distribute the URLs of specific mapping files around the network. For example, a router can publish a new mapping file to the network by forwarding a publication: *pub* [new mapping: NewMappingURL] to the network. Once a router gets this publication, it can simply register the URL of the mapping file to the mapping store for future use or use the reasoner to load and merge the new mapping file into the routing ontology, depending on its currently select mapping strategy.

A weakness of this approach to implement the model stores is that it will take time for the KBNMap routers to load the ontologies that are stored remotely. In addition, the router would take extra processing to check the credibility and quality of new ontologies that are provided by the other routers/applications, as the router only knows the URLs of ontologies without knowledge of the actual ontologies. An alternative approach would be to use an XML or RDF database to implement the model stores for the KBNMap routers, which is then deployed in machines with large storage capability, so that the actual ontologies could be stored locally or at least with the KBNMap network. In addition, the KBNMap routers could manage and check the actual ontologies. However, as mentioned above, it will require additional space storage on KBNMap routers, while alternative implementations would not influence the substantive findings of KBNMap evaluations in Chapter 7, since the Pellet reasoner used in the KBNMap router still need to merge and load the ontologies into its own graph model.

6.4 Adaptive Semantic Mapping Service Block

As seen in Figure 6-2, the adaptive semantic mapping service makes use of both the Jena library and the JavaBayes library. This mapping service block allows a KBNMap router to dynamically select the appropriate mapping strategy in order to adapt to the changes of ontology, application, and environment characteristics. The implementation of each component presented in Figure 6-2 and how they interact with each other to provide the adaptive mapping service for the KBNMap router are illustrated in the following subsections.

<?xml version= "1.0" encoding= "UTF-8" standalone= "no" ?>
<validation>
<variable>
<name>RoutOnt_size</name>
<measure type= "Yes" />
</variable>
<variable>
<name>MemoryResources</name>
<measure type= "Yes" />
</variable>

Figure 6-3: Snippet of configuration file for monitoring trigger factors

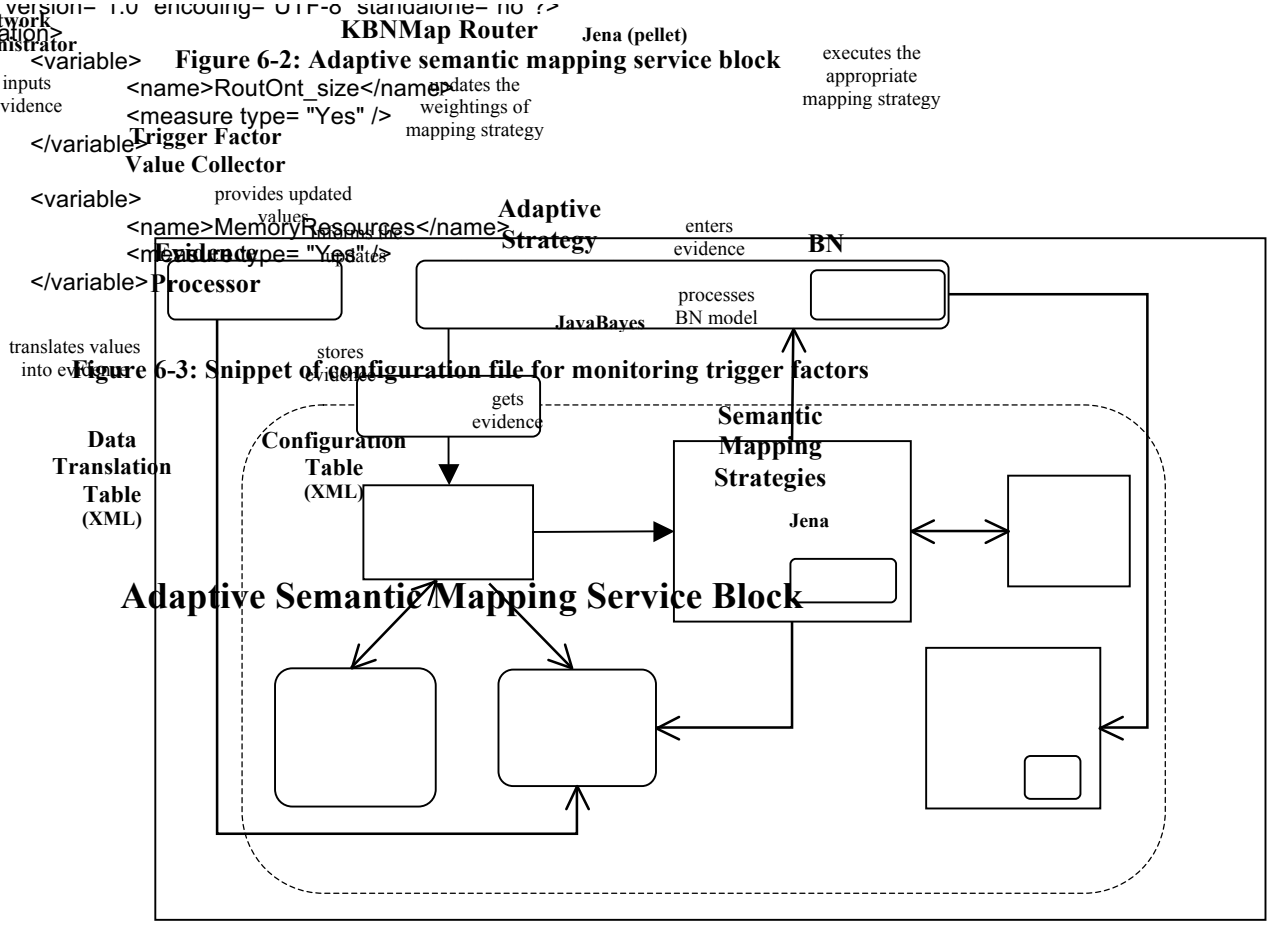
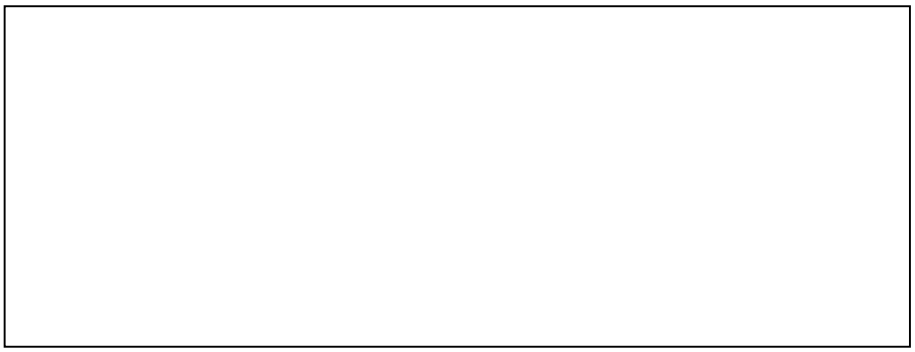


Figure 6-2: Adaptive semantic mapping service block

6.4.1 Trigger Factor Value Collector

When a KBNMap router is initialised, the trigger factor value collector object is created, which spawns a valueMonitor thread to monitor the changes of trigger factors and collect values. The valueMonitor reads a configuration file first in order to get the information of which trigger factors to monitor. Figure 6-3 illustrates the snippet of the configuration file. It can be observed that the "Yes" values in the RoutOnt_size and MemoryResources nodes indicate that valueMonitor needs to measure the variation of routing ontology size and memory resources.



The valueMonitor thread is currently set to monitor the changes of trigger factors periodically (every 200 millisecond, but this is configurable). The trigger factors monitored by the thread can include the size of routing ontology, the

subscription/unsubscription/publication reception rate, the memory resources available, and the number of mappings available in the model store. It should be noted that the network administrator is responsible for collecting the values of some other trigger factors such as the network scale and the tolerance level of applications.

On receiving the updated values of trigger factors, the value collector first analyses the values to see if the new values are the same as the old values from last update check which are stored in a data collection vector. If the updated values are different, the value collector replaces the old values with new values in the vector and sends the updated values to the Evidence Processor.

Strengths/Weaknesses/Limitations

A strength of the Trigger Factor Value Collector is that the trigger factors which need to be measured and the monitoring frequency of valueMonitor, can be easily modified by changing the values of the configuration file, in order to cater for different configurations of network and requirements of applications. Thus the administrators do not need to modify the source code of the Value Collector.

6.4.2 Evidence Processor

The functionality of the evidence processor is implemented by the EvidenceProcessor class. It is responsible for translating the values of trigger factors that are collected by the trigger factor value collector into the relevant evidence states. As described in Figure 6-2, the EvidenceProcessor is connected with two tables: the Data_translation table and the Configuration table. The Data_translation table is used in the Evidence Processor to help translate the collected values into relevant evidence, while the Configuration table is used to store the translated evidence. For example, if the rate of publications is 5 pubs/ min, then this value is passed to the Data_translation Table which will calculate the appropriate evidence state for the **PubRate** BN node. According to the state specification table identified in Chapter 5 (see Table 5-8), value '5' translates into the *small* evidence state. Finally, the evidence processor stores this new evidence state for the **PubRate** node in the Configuration table. Once the EvidenceProcessor stores the evidence, it sends a message to inform the adaptive strategy selection engine of this evidence update.

6.4.2.1 Data Translation Table

The Data Translation Table is implemented on the basis of the boundary values table (Table 5-8) in Chapter 5. It is implemented in an XML file so that it is readable by both the human administrator and the adaptive strategy selection engine of KBNMap router. Each trigger

factor in the file has three states: small, medium and large. And the range of each state is defined by a minimum and maximum boundary value. Figure 6-4 presents a snippet of the data translation table.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<validation>
	<variable>
		<name>RoutOnt_size</name>
		<state type= "small">
			<min>0</min>
			<max>1000</max>
		</state>
		<state type= "medium">
			<min>1001</min>
			<max>5000</max>
		</state>
		<state type= "large">
			<min>5001</min>
			<max>1000000000</max>
		</state>
	</variable>
	<variable>
		<name>Required_ReferOnt</name>
		<state type= "small">
			<min>0</min>
			<max>1000</max>
		</state>
		<state type= "medium">
			<min>1001</min>
			<max>5000</max>
		</state>
		<state type= "large">
			<min>5001</min>
			<max>1000000000</max>
		</state>
	</variable>
```

Figure 6-4: Snippet of Data Translation Table

6.4.2.2 Configuration Table

The Configuration Table is also implemented in an XML file so that it is readable by both the human administrator and adaptive strategy selection engine. Figure 6-5 illustrates the snippet of configuration file.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<Config>
	<variable>
		<name>RoutOnt_size</name>
		<value>medium</value>
	</variable>
		<variable>
		<name>Required_ReferOnt</name>
		<value>null</value>
	</variable>
	<variable>
		<name>MappingOnt_size</name>
		<value>null</value>
	</variable>
	<variable>
		<name>RoutOnt_express</name>
		<value>high</value>
	</variable>
	<variable>
		<name>Required_ReferOnt_express</name>
		<value>null</value>
	</variable>
	<variable>
		<name>AvailableMemory</name>
		<value>large</value>
	</variable>
```

Figure 6-5: Snippet of Configuration Table

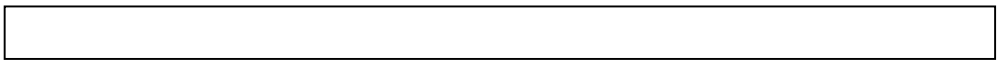
Strengths/Weaknesses/Limitations

A strength of the EvidenceProcessor as designed, is that further translated variables can be easily added or removed into the Data Translation Table and Configuration Table, with the only impact being on the Trigger Factor Value Collector where slight modification is necessary to measure the new added variables. A limitation of the current implementation is that it takes extra time and resources for the KBNMap routers to access the tables and fetch information, which has a minor impact on the process of inferring the mapping strategies, but greatly improves the flexibility and reusability of the system. However, this can be improved by using hard-coded collectors to incorporate pass evidence state values direct to the BN.

6.4.3 Adaptive Strategy Selection Engine

Once the KBNMap router is started, it first instantiates an in-memory Jena OntModel object to access, manage ontologies, which also incorporates the Pellet reasoner. It then creates the connection to the semantic model and mapping stores, allowing KBNMap to parse the URLs of stored ontologies. The KBNMap router then use the OntModel and Pellet to, load and reason all necessary ontologies into the in-memory OntModel in order to form the KBNMap's current knowledge base (routing ontology). Detailed information about how the OntModel and Pellet reasoner work together to access and query the ontological information of the routing ontology is presented in Appendix A.

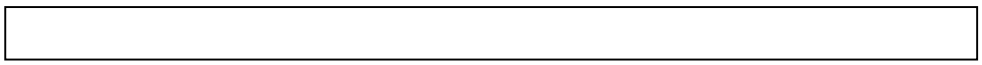
After constructing the routing ontology, the KBNMap router initialises the adaptive strategy selection engine. Once the selection engine is started, it instantiates an in-memory object for the Bayesian Network inference engine from the JavaBayes library and creates connection to the BN model for mapping strategy selection (see Section 6.4.4). An XML-based representation of the BN model (see Figure 6-7) is parsed, which translates each XML-based BN node into the probabilistic-based variable that is understood by the BN Inference engine of JavaBayes (see Section 6.4.3.1). The probability variables (objects) are then stored into a using the registerVariable() method. The variable list is then registered with the JavaBayes inference engine.



As the variable vector is formed, the adaptive strategy selection engine registers itself as a callback object, for when the updated evidence message from EvidenceProcessor is received. After receiving the update message, the adaptive Strategy Selection Engine uses a Java DOM XML parser to parse and analyse the Configuration Table (see Section 6.4.2.2) in order to get the updated evidence. The adaptive strategy selection engine then enters the

public void processBNmodel (String ModelName)

updated evidence into the BN model. When this happens, the Inference engine calls the BN model processing method to process the updated BN Model.



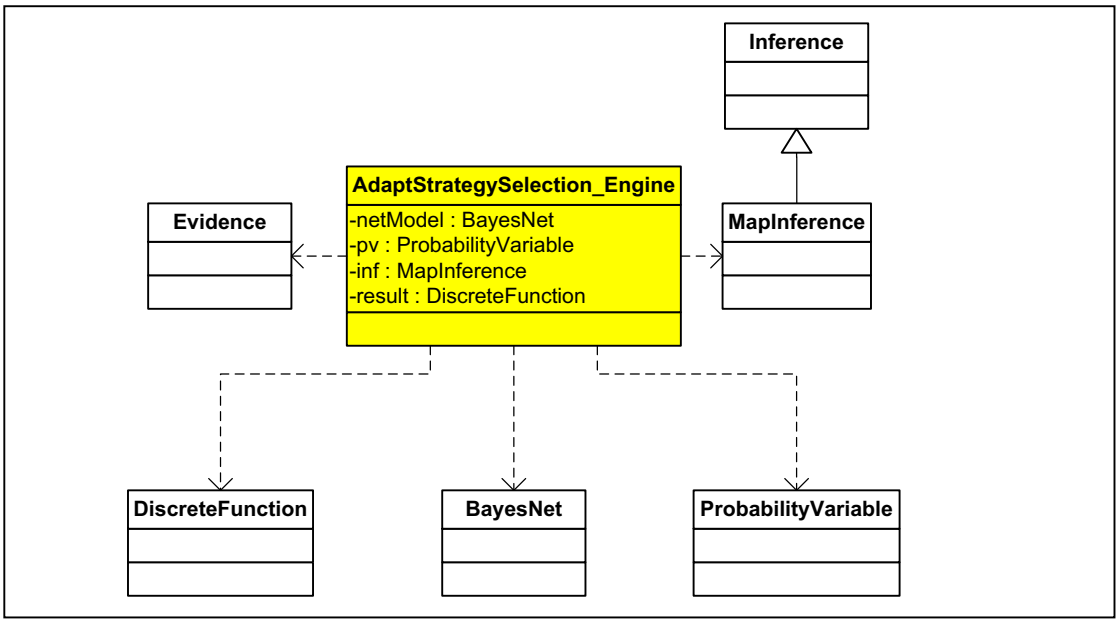
In the following subsection, the detailed implementation of the adaptive strategy selection engine is described.

6.4.3.1 Adaptive Strategy Selection Engine Class

The functionality of the adaptive strategy selection engine is implemented by the Adaptive_StrategySelection_Engine class (see Figure 6-6) that implements the functionality for assigning the updated evidence to the observed variables (parentless nodes) of Bayesian Network model for strategy selection (see Section 6.4.4) and inferring the weightings of mapping strategies in the output variable: StrategySelection of BN model. Each Adaptive_StrategySelection_Engine uses BayesNet (the BN inference Engine implemented in JavaBayes) to initialise Bayesian Network inference algorithms that are implemented in Inference class.Using the JavaBayes library, the BayesNet is implemented as the Bayesian Network inference engine, while the MapInference inherits the main functionalities of Inference class that implements all Bayesian Network inferencing algorithms provide by the JavaBayes library.

The AdaptStrategySelection_Engine iterates through the evidence stored in configuration table and assigns the updated evidence to the relevant observed variables of the BN model. For example, if the new updated evidence of observed unknown data is *small*. The evidence state value *small* is then entered into ObservedUnknownData node. Once the BN model gets the new evidence, the Inference engine (BayesNet) starts to calculate the posterior probabilistic values of influenced nodes by using a predetermined inference algorithm from the mapInference. Based on the diagram of BN model developed in Figure 5-11, this calculation propagates from the UnknownData child node to the output node Strategy_Selection, where ranking weightings for each mapping strategy are produced in the Strategy_Selection node. Finally, the adaptive strategy selection engine sends the updated weightings to the KBNMap router, thereby informing the KBNMap router which strategy it should then select.

Figure 6-6: UML class diagram for adaptive strategy selection engine classes



Strengths/Weaknesses/Limitations

The strength of this implementation is the use of the JavaBayes for building the Bayesian Network inference engine as the JavaBayes library is relatively small (240 Kb) KBNMapand yet complete. KBNMapIn addition, the parsers provided by JavaBayes bring the possibility for the KBNMap router to edit the BN model at runtime and save the BN model in a variety of formats. Furthermore, the multiple JavaBayes inference algorithms, with different characteristics, enable the administrator to flexibly select the inference algorithm for the KBNMap routers, in order to cater for different situations. For example, the "variable elimination algorithm" consumes less memory for inferring while the "bucket tree elimination algorithm" provides a fine-grain inference service (JavaBayes 2001; Jensen et. al., 2007; Darwiched 2009).

It is possible that the KBNMap routers could use a better and more powerful Bayesian Network library to implement the strategy selection engine, however, most Bayesian Network libraries are either closed-source or not freely available. For example, the Netica (NeticaV4.0.8) runtime environment, which was used for designing the BN model in Chapter 5, provides an intuitive and smooth user interface for drawing and editing the networks. However its Java runtime library is not free for developers.

6.4.4 Bayesian Network Model for Strategy Selection

Through use of the JavaBayes Bayesian Network development platform, the Bayesian Network model for selecting the appropriate mapping strategy is encoded and stored in a BIF XML file on the basis of the designed BN model developed in Chapter 5 (Figure 5-11). Each BN node is implemented as an XML-based node with several types of attributes such as the state and the position in the network. The prior probability of each trigger factor and CPTS of each internal node is also encoded into this file. Figure 6-7 (a) gives a snippet of Bayesian Network model for nodes description. For example, observing the first variable "RoutOnt_Size". Its three "outcome" elements represent the three different states of the trigger factor, while the value of its "property" elements: "observed small", indicates that the evidence of this trigger factor entered in the BN model is *small*. In addition, Figure 6-7 (b) gives a snippet of the BN model for prior probability and CPTs. As a concrete example, for the fourth probability node: ("MemoryResources" "AvailableMemory" "MergedOnt_size"), means that it is the CPT of the "MemoryResources" node with two parent nodes: "AvailableMemory" and "MergedOnt_size". The series of digits below are the probabilistic value of each state of the "MemoryResources" given the states of its parent nodes.

(a)

```
<BIF VERSION="0.3">
<NETWORK>
<NAME>Strategy_Selection</NAME>

<!-- Variables -->
<VARIABLE TYPE="nature">
      <NAME>RoutOnt_size</NAME>
      <OUTCOME>small</OUTCOME>
      <OUTCOME>medium</OUTCOME>
      <OUTCOME>large</OUTCOME>
      <PROPERTY>observed small</PROPERTY>
      <PROPERTY>position = (156, 116)</PROPERTY>
</VARIABLE>

<VARIABLE TYPE="nature">
      <NAME>Required_ReferOnt</NAME>
      <OUTCOME>small</OUTCOME>
      <OUTCOME>medium</OUTCOME>
      <OUTCOME>large</OUTCOME>
      <PROPERTY>observed small</PROPERTY>
      <PROPERTY>position = (259, 117)</PROPERTY>
</VARIABLE>

<VARIABLE TYPE="nature">
      <NAME>MappingOnt_size</NAME>
      <OUTCOME>small</OUTCOME>
      <OUTCOME>medium</OUTCOME>
      <OUTCOME>large</OUTCOME>
      <PROPERTY>observed small</PROPERTY>
      <PROPERTY>position = (415, 118)</PROPERTY>
</VARIABLE>

<VARIABLE TYPE="nature">
      <NAME>MergedOnt_size</NAME>
      <OUTCOME>small</OUTCOME>
      <OUTCOME>medium</OUTCOME>
      <OUTCOME>large</OUTCOME>
      <PROPERTY>explanation small</PROPERTY>
      <PROPERTY>position = (243, 241)</PROPERTY>
</VARIABLE>
```

(b)

```
probability (  "Required_ReferOnt_express" ) { //1 variable(s) and 3 values
      table
            0.333 // p(small | evidence )
            0.333 // p(medium | evidence )
            0.333;      // p(large | evidence );
}
probability (  "Express_union"  "RoutOnt_express"  "Required_ReferOnt_express"
      table
            0.8 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.2 0.9 0.0 0.8 0.6 0.0 0.0 0.
}
probability (  "AvailableMemory" ) { //1 variable(s) and 3 values
      table
            0.333 // p(small | evidence )
            0.333 // p(medium | evidence )
            0.333;      // p(large | evidence );
}
probability (  "MemmoryResources"  "AvailableMemory"  "MergedOnt_size" ) { //3
      table
            1.0 1.0 1.0 0.2 0.4 0.6 0.0 0.1 0.2 0.0 0.0 0.0 0.8 0.6 0.4 0.2 0.
}
probability (  "NumOfStoredOnts"  "NumberofMappingOnts"  "NumberofReferOnts" )
      table
            0.6 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.4 0.6 0.0 0.6 0.5 0.0 0.0 0.
}
probability (  "NumberofMappingOnts" ) { //1 variable(s) and 3 values
      table
            0.3333333333333333      // p(small | evidence )
            0.3333333333333333      // p(medium | evidence )
            0.3333333333333333;     // p(large | evidence );
}
```

Figure 6-7: Snippet of Bayesian Network model for strategy selection

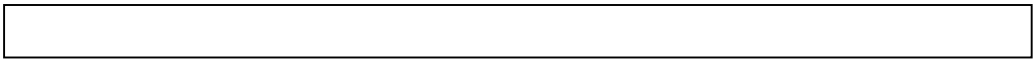
public void [illegible](AttributeConstraint c, AttributeValue av)

Strengths/Weaknesses/Limitations

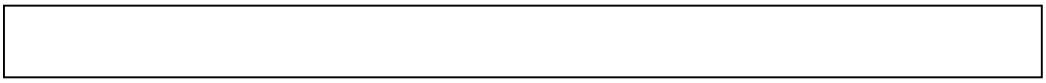
As the BN model is written in XML format, the strength of this implementation is that the BN model can be easily edited. It also provides for useful extensibility opportunities. However, a challenge faced during the implementation is that it becomes troublesome to implement the Conditional Probabilistic Tables for the BN nodes that have three or more parent nodes, as there is a exponentially large number of probabilistic values required as the number of nodes and states grow. Despite the Netica tool providing a convenient way to implement the CPTs by providing an intuitive method for filling in the values, the XML-based BN Model developed by Netica proved to be incompatible to the JavaBayes library. Thus extra effort is needed to convert the Netica version into the JavaBayes version.

6.4.5 Semantic Mapping Strategies

As described in Figure 6-2, after the adaptive strategy selection engine sets the weightings of mapping strategies, the router selects the strategy with the highest weighting through simply comparing the weightings of mapping strategies. The selected strategy is then registered with the router using a strategy registration method. It should be noted that at this stage, the selected mapping strategy is not triggered until the KBNMap router encounters unknown data.



When a KBNImpl subscription/publication arrives, the KBNMap router uses the checkforSemanticInteroperability() method to measure if it is unknown data before the subscription is inserted into the subscription tree or the publication is matched.



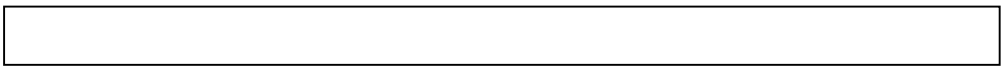
The checkforSemanticInteroperability() method works in two ways. It can be used to check for unknown semantic content contained in either subscriptions (represented as several filter or AttributeConstraints) or publications (represented as several AttributeValues). The AttributeValue av holds the value that will be compared against the filtering constraint. This is taken from publications to match against subscriptions. If this method is being called for subscriptions then av will be null and the semantic values used in the filtering constraint will be used.

The attribute value is then analysed by using the following getOntResource() method, which analyses the extracts any semantic terms from the publication value or subscription filter and checks it against the ontological terms within the knowledge base (routing ontology).

```
public static OntResource getOntResource(OntModel ontmodel, AttrValue attrValue) {
.....
    return ontmodel.getOntResource (attrValue.stringValue ());
    }
```

If the method returns a *null* value, meaning the attribute value is an unknown term. Therefore, the subscription /publication containing the attribute value is measured as unknown data.

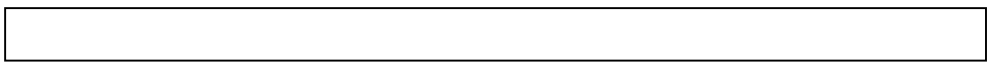
Once the presence of unknown data has been determined, the tryInteropStrategy() method below is called by the router, which uses the registered mapping strategy to deal with the unknown data.



The following subsections illustrate how each mapping strategy designed in Section 4.3.4 is implemented to deal with the unknown data, once it is triggered.

6.4.5.1 The "Every" Strategy

If the **Every** strategy is triggered, the KBNMap router will use the readMapping() method to read the URL of each mapping file stored in the mapping store and call the in memory ontology model that instantiated by the KBNMap router to load and merge all of them into the current routing ontology. Then the entire routing ontology will be re-reasoned and re-loaded by the Pellet reasoner. After this, the validateOntologyModel() is called in order to check the validity of the new merged routing ontology .



If the unknown term still could not be resolved, this **Every** strategy will return a *false* indication to the KBNMap router.

6.4.5.2 The "ApproOnly" Strategy

If the **ApproOnly** strategy is triggered by the router, a new in memory ontology model OntModel object, without wrapping any reasoner, is first instantiated. The reason for this is that the mapping files should not be loaded and reasoned before they are checked by the mapping strategy otherwise it will take time and resources to process these mapping files. The mapping strategy then parses and reads each mapping file available in the mapping store and uses the getOntology() method to test if the processed mapping file is already loaded into the routing ontology. If it is not, then it will be stored in the new ontology model temporarily.

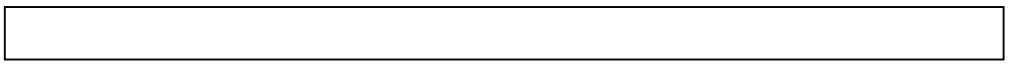
public boolean findIndMapping(AttributeValue av, OntResource r);

Furthermore, the new ontology model also disables the loading of imported/referenced ontologies (enabled by default).

The new ontology model then analyses the temporally stored mapping file to extract an ontological term list, which contains all ontological terms expressed in the mapping file. After that the **ApproOnly** Strategy uses the getOntResource() method to navigate this ontological term list to find the qualified ontological term which is equivalent to the unknown ontological term contained in the subscription or publications. If the unknown ontological term is found, the **ApproOnly** Strategy terminates the search and calls the KBNMap router's OntModel to load and merge this mapping file into the routing ontology. If the term is not found in a mapping file the **ApproOnly** strategy starts to process the next mapping file in the semantic mapping store. If the unknown term contained in the subscription/publication could not be found in any available mapping file in the mapping store, the **ApproOnly** strategy will return *false* indication to the KBNMap router.

6.4.5.3 The "ApproInd" Strategy

The principle of iterating and checking each available mapping file in the mapping store by the **ApproInd** strategy is the same as the **ApproOnly** strategy. However, it uses the findIndMapping() method to allow the OntModel of KBNMap to load the *only* appropriate individual mappings from the mapping file into the routing ontology.



Again if the appropriate individual mapping could not be found in any mapping file in the mapping store, the **ApproInd** strategy will return *false* indication to the KBNMap router.

Strengths/Weaknesses/Limitations

The strength of the implementation is the use of a new in-memory object for the Jena OntModel to discover the appropriate mapping files. The new ontology model without reasoner allows the KBNMap router to access the available mapping files quickly, as it does not need to reason the ontologies. In addition, the methods available in Jena enables easy access for the routers to get the appropriate mapping file.

The key limitation of this implementation is that the current KBNImpl router cannot be extended with the **ApproRefer** mapping strategy (the first option of Section 4.3.4.4), mainly due to the characteristic of this mapping strategy, which highly depends on the combination of mapping relation found, and the operator to be applied to the unknown concept, to load the appropriate reference ontology. KBNImplKBNMapThe KBNMap router that was implemented with this mapping strategy exhibits an very poor performance with respect to

publication/subscription matching during the trial tests. As KBNImpl walks the subscription tree to find the appropriate place to insert a new subscription, or the set of subscriptions that matches a publication, the strategy needs to be executed for each encountered subscription. This takes a considerable amount of time and resources for the KBNMap router to process the mapping strategy (the time and resources used for the **ApproRefer** strategy can be found in Appendix C) every time the publication/subscription matching occurs. As stated in Section 3.2, the efficiency of publication matching is an important key feature for the SBPS system, and the implementation of this strategy was removed. However, it would be worthwhile to test if this mapping strategy is implementable in other SBPS systems where there are alternative mechanisms for event matching/ routing, as the idea of the **ApproRefer** strategy to address heterogeneity is novel.

Furthermore, as stated in Section 4.3.4.4, the current implementation of KBNMap system adopts the second option to simply drop unresolved unknown subscription/publication with a warning, and based on this warning the network administrator is responsible for finding the required mapping files. She can then register new mappings with the KBNMap routers. This approach avoids extra time and resources being expended by the KBNMap router. Otherwise if the other options identified in Section 4.3.4.4 are applied, it will take more extra time and resources for the router to resolve the unknown message. In addition, as stated in Section 4.3.4.5, the other possible options are either highly depending on the other ongoing research (Option 3 in Section 4.3.4.4), or are out scope of the thesis (Option 5).

Finally, due to the fact that KBNImpl mainly supports OWL ontologies, the current implementation of mapping strategies is limited to OWL ontologies only, although it could easily be extended to RDFS and DAML+OIL ontologies due to the capability of Jena. A risk however is the reliance on Jena, as Jena and its reasoner have been found to have poor performance on large OWL datasets (Guo 2004).

6.5 Using the KBNMap System

Suppose there is a knowledge subscriber who expresses specific interests in an ontological event that is published by a respective knowledge publisher. The ontological concept contained in the subscription is expressed *according* to the routing ontology, whereas the ontological event is expressed according to another ontology. The ontological term in the event is semantically similar to an ontological term in the subscription but is referenced differently. Importantly, an mapping ontology containing the mapping relations between the subscriber and publisher ontologies is stored in the ontology store. In the following

subsections, the classes that are necessary to realise this example are described and how the example can run.

6.5.1 Starting a KBNMap Router

As KBNMap is an extension of KBNImpl, the way of starting KBNMap routers is the same as the KBNImpl in the version of (Keeney et. al., 2007). A KBNMap master router is created with the following command:

Java siena.StartServer –port <portnumber> - master [host_name: host_port]

The port option specifies on which port the router listens for incoming connections. If omitted the router will listen on port 1234. If the master option is included, the router tries to connect itself to a parent KBNMap router which resides on the specified host (e.g. –master tcp:localhost:1233) .

6.5.2 Implementing a Subscriber

The ExampleSubscriber testing class (See Figure 6-8) realises a subscriber that is interested in the Event that is semantically equivalent to **region.owl#politicalRegion**, where **region.owl** is the routing ontology loaded in the router. The ExampleSubscriber creates a ThinClient first then creates a subscription that is implemented by a filter. Through the addConstraint method, a filter could be bound to a subscription that expressed interests in the event about "**region.owl#politicalRegion**,". An ExampleSubscriber can be stated by:

Java KBNMap.example.ExampleSubscriber

```
public class ExampleSubscriber {

    public static void main(String[] args) {
        try{
            ThinClient siena;

            /**address is the address of a KBN router**/
            siena = new ThinClient("tcp:127.0.0.1:2000");

            Filter f = new Filter();
            String AttributeValue= "https://www.cs.tcd.ie/region.owl#politicalRegion";
            f.addConstraint("event", new AttributeConstraint(ExtOp.EQUIVAL, AttributeValue));

            /**Notifyme is a simple callback object**/
            NotifyMe myObject = new NotifyMe();

            /**send subscription to the network**/
            siena.subscribe(f, myObject);

            while(true)
                /** the subscriber now waits for matching notification to arrive**/
                Thread.sleep(10000);
        }
        catch (Exception e){
            e.printStackTrace();
        }
    }
}
```

Figure 6-8: A simple Subscriber

6.5.3 Implementing a Publisher

The ExamplePublisher class (Figure 6-9) realises a publisher that is publishing a publication containing an ontological term: "UnknownRegion.owl#Geopolitical-Entity", in which the UnknownRegion.owl, which is not loaded by the router. The ExamplePublisher class first instantiates a ThinClient in order to connect with the KBNMap router. Then it publishes a publication that contains a pair of attributes with name "event" and value "UnknownRegion.owl#Geopolitical-Entity". An ExamplePublisher can be stated by invoking:

Java KBNMap.example.ExamplePublisher

```
public class ExamplePublisher {

    public static void main(String[] args) {

        try{
            ThinClient siena;
            /**address is the address of KBNMap router**/
            siena = new ThinClient("tcp:127.0.0.1:3000"); //address is the address of a KBN router

            Notification n = new Notification();
            String attributeValue = "https://www.cs.tcd.ie/UnknownRegion.owl#Gepolitical-Entity";
            n.putAttribute("event", attributeValue);

            siena.publish(n);
            System.out.println("Sent notification : "+n);
        }
        catch (Exception e){
            e.printStackTrace();
        }
    }
}
```

Figure 6-9: A simple publisher

6.5.4 Configuring Ontologies for a KBNMap Router

First, a routing ontology store (routingontologies.conf in Figure 6-10) is required, which stores the URL of routing ontologies in a serial order. Second, the ontologies stored in the ontology store will be loaded into the router at startup as its routing ontology. The code of the example shows how to load routing ontologies in OntSienaConfig class (Figure 6-10). Note that the actually loading is delegated to a separate class OntCovering.

The initialise () method (see Figure 6-10) will be called in the as the KBNMap router starts up .

With respect to the mapping and reference ontologies, whose URLs are stored in the mapping store that is implemented in another ontology configuration file (ontmapp.conf). The ontmapp.conf file will be read by a specific mapping strategy depending on this mapping strategy is activated by the KBNMap router for tackling with unknown data.

```
public class OntSienaConfig {

    private static final String ONTOLOGIES_CONF_PATH = "routingontologies.conf";

    public static void initialize() throws IOException ,  IFException{
        initialize(ONTOLOGIES_CONF_PATH);
    }

    public static void initialize(String ontologiesConfPath) throws IOException,  IFException{
        System.out.println("Initializing KBNMap");
        System.out.println("Creating ontology model");

        /**check ontology store**/
        List<String> ontologyUrlList = readOntologyUrlList(ontologiesConfPath);

        /**initialise ontology model and Pellet Reasoner**/
        OntModel ontModel = ModelFactory.createOntologyModel(PelletReasonerFactory.THE_SPEC, null);
        OntCovering.setOntologyModel(ontModel);

        /**load and reason ontologies**/
        for (String ontologyUrl : ontologyUrlList) {
            OntCovering.loadOntology(ontologyUrl);
        }
        /**validate loaded Knowledge Base (routing ontology) **/
        OntCovering.validateOntologyModel(ontModel);
    }
```

Figure 6-10: Snippet of OntSienaConfig class.

6.5.5 Implementing a Network Administrator

The Administrator class is a command-line based interface. The basic operations of administrator class is: first, it allows people to load the BN model; second, people can use it to set evidence to the observed variables by using set_ObservedVariable (variableName, evidence) method; thirdly, the final strategy selection node of BN model can be specified by using set_explainVar (variableName) method, in which the explainVar indicates the variable that people expect to infer the outcome. Finally, it can facilitate people to print out the inferred results of the final strategy selection node through print_ExplainVar (variableName) method. An administrator can be started by:

Java KBNMap.example.administrator

6.5.6 Running the example

In the following, it is described how to run the simple example. First, a simple distributed topology of three KBNMap routers (1 master and two sub routers) has to be setup, in order to form a distributed knowledge delivery service. A distributed topology is constructed in sequence. In practice, every KBNMap router can connect to another router at startup:

Start the master router on local host with port number: 1234, execute:

Java siena.StartServer –port 1234

Start the second router that is connecting with the master, with port number: 2000, execute:

Java siena.StartServer -port 2000 -master tcp:localhost:1234

Start the third router that is also connecting with the master, with port number: 3000, execute:

```
Java siena.StartServer -port 3000 -master tcp:localhost:1234
```

At router runtime, the adaptive strategy selection engine periodically executes (see Figure 6-11) updating the resulting values of mapping strategies, depending on the variations of observed characteristics. Then the initial active strategy is in the ready status and will be used by router to deal with the first encountered unknown data.

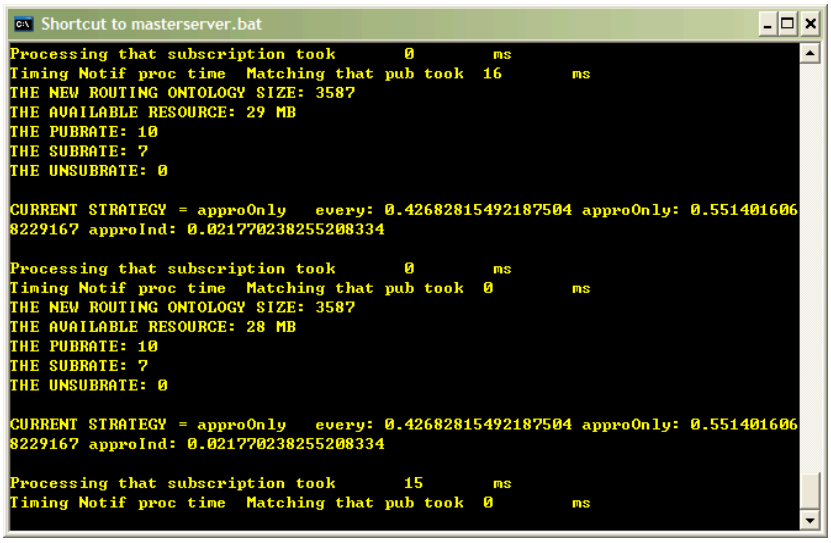


Figure 6-11: Snapshot of master server at network runtime

After configuring the network topology, a subscriber (it connects with the sub router who listens on TCP port 2000) is started using the command:

```
Java KBNMap.example.ExampleSubscriber
```

The subscriber starts to forward subscriptions after a short delay. Finally the publisher is started (it connects with the sub router who listens on TCP port 3000) by:

```
Java KBNMap.example.Examplepublisher
```

When the unknown publication [Event: UnknownRegion.owl#Geopolitical-Entity] published by the publisher arrives at the router who stores the subscription [Event: @= : region.owl#politicalRegion] in its subscription tree, the router would recognise this unknown publication and use its active strategy to deal with it (see Figure 6-12). Recall that "@=" is a shorthand notation for the "semantically equivalent" KBNImpl subscription operator.

Figure 6-12: A snapshot of master router when it deals with an unknown publication

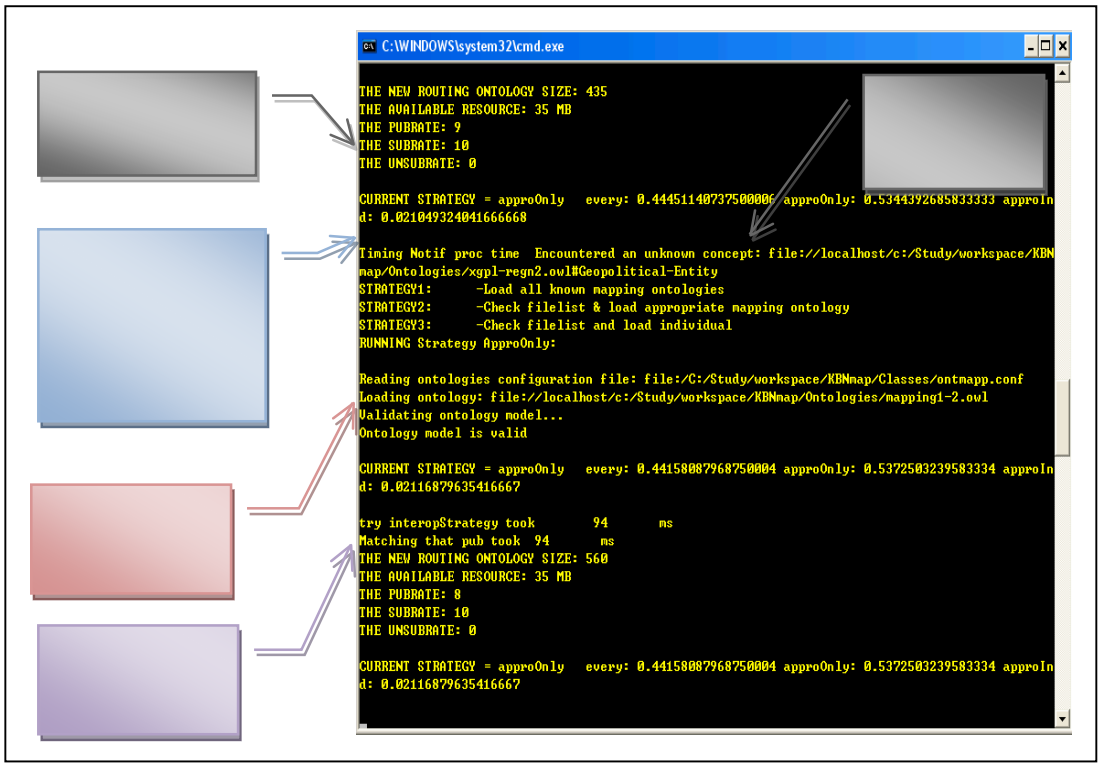
The values of some trigger factors monitored by the KBNMap router

An unknown semantic term is encountered

The selected mapping strategy is **ApproOnly** following with the weightings of mapping strategies, which are inferred by the adaptive mapping strategy selection engine

The appropriate mapping file is loaded by the **ApproOnly** strategy

No warning message indicates that **unknown data** has been resolved



6.6 Conclusion

The implemented knowledge-based networking infrastructure KBNMap is a contribution to the area of Knowledge based Networking systems for several reasons. Firstly, in contrast to most other efforts, it allows multiple diverse ontologies to co-exist among applications and routers, and so can realise a distributed heterogeneous knowledge delivery service. The router makes use of different mapping strategies to deal with heterogeneous ontological information. The implemented mapping strategies include **Every**, **ApproOnly** and **ApproInd** algorithms in order to provide a flexible and configurable mapping service for the routers.

Secondly, rather than selecting a fixed mapping strategy, it is capable of dynamically re-configuring its mapping services in order to adapt to changing states of application, ontology and environment conditions. Moreover, instead of carrying out simulations the prototype also served as the basis for experiments that are described in the next chapter. This increases the validity of the results compared to simulation based results.

Finally, the implemented example shows that an event that is semantically equivalent to a subscription, but instantiated using a different but mapped ontology, can be delivered to the recipients who express specific interest in that knowledge.

KBNMap was implemented based on KBNImpl code, and inherits the advantages of the KBNImpl implementation. For example, with respect to scalability, in addition to the basic hierarchical topology adopted, the KBNMap network can also make use of the semantic clustering mechanisms identified in (Keeney et. al., 2008b) to reduce the processing time for making routing decisions, shorten hops between clients, and reduce the size of routing ontology in a large scale deployment (Section 3.4.7).

With respect to the *adaptive semantic mapping service block* implemented, as discussed previously in this chapter, the design and implementation of BN model and configuration files (e.g., Data Translation Table) allows for extendibility and reconfiguration (XML format), which can be easily support with new evidence variables (trigger factors) and states, or be easily reconfigured with more sophisticated weightings for BN variables (Conditional probability Table (CPT) for intermediate nodes/final nodes and prior probabilities for parentless nodes).

In addition, the adaptive mapping service allows the KBNMap router to use synchronised mechanisms (e.g., Java synchronised method, block) to monitor, collect, and translate the values of measured trigger factors, to ensure the adaptive strategy selection engine has all updated evidence entered into the BN model concurrently. The multi-threaded approach used allows the KBNMap router to continue running even if part of it is blocked. For instance, publication matching and subscription merging can continue to some degree in other threads while the semantic interoperability strategy is executing (except when the core routing ontology is being updated)KBNMap. Furthermore, as the adaptive mapping strategy engine only starts to work when it gets the update message from an evidence processor, memory and resources shared with the other threads can be saved.

In Chapter 7, the performance and applicability of the implemented KBNMap system is evaluated.

7 EVALUATION

In the previous chapter, an implementation of the KBNMap infrastructure was introduced. This chapter presents an evaluation of the implemented KBNMap infrastructure.

7.1 Introduction

This chapter presents an evaluation of the implemented KBNMap infrastructure, giving a detailed insight into its behavior. The aim of this evaluation was to assess the work undertaken in this thesis for addressing the key challenges outlined in Section 3.6.

The evaluation method adopted comprises three experiments. The first experiment aims to examine the extra processing required by each individual mapping strategy to resolve heterogeneous information when each is used by the KBNMap deployment. The second experiment presents three categorized sets of tests for evaluating KBNMap with different ontology, application, and environment related trigger factors. In this way, the effect of the adaptive semantic mapping service on the performance of KBNImpl routers in the context of distributing information in different situations can be evaluated. In addition, the third experiment investigates the correct operation of selecting mapping strategies and the adaptability of KBNMap routers to cater for the changes of key trigger factors in different tests of the three categorized sets..

Section 7.2 presents the experimental goals of this evaluation work. The methodology adopted for evaluation is described in Section 7.3. Section 7.4 presents the key trigger factors used in this evaluation. In addition, the ontologies used for evaluation are described in Section 7.5. Then the general experiment setup is introduced in Section 7.6 and Section 7.7. In Section 7.8, the first experiment for comparing the performance of mapping strategies is described. The second experiment of evaluating the performance of KBNMap is presented in Section 7.9. Finally, the third experiment for investigating the correct operation of selecting strategies and adapting to the changes of trigger factors is discussed in Section 7.10.

7.2 Evaluation Goals

Evaluation Goal 1: to evaluate the effect of the mapping service on runtime performance

The proposed adaptive mapping service enables KBNMap routers to use semantic mappings at runtime to address heterogeneity when they receive unknown information. However, this extra processing will inevitably increase the workload of an individual KBNMap router. It is important to investigate the effect of new mechanisms introduced to see if the extra processing can be managed to a level that is acceptable. Therefore, the *first evaluation goal* was to evaluate the effect of the introduced mechanisms for semantic interoperability support on performance of KBNMap routers for matching publications with subscriptions and delivering publications across the network to the subscribers. The performance of publication matching reflects the scalability of the KBNMap, which is the key feature for the SBPS system (Section 3.3.2.2) meanwhile it is the task of the KBNMap network to effectively reduce the overall, end-to-end delay of publications. In addition, to measure the publication delivery of KBNMap could also reflect how well the system addresses the first main part of research question (Section 1.2): "distributing heterogeneous information". This goal was addressed through experiment 1 and experiment 2.

Evaluation Goal 2: to investigate the adaptability of KBNMap w.r.t influential factors

The adaptability of the mapping service was also considered in order to evaluate KBNMap's support for adaptive semantic interoperability through the use of ontology mappings and probabilistic modelling techniques. The adaptive semantic mapping service provides a mechanism for the KBNImpl routers to self-select the mapping strategies at runtime to adapt to the changes of ontology, application and environment related trigger factors. This is important for distributing heterogeneous information in the highly dynamic situations, where the constant changes in ontologies, applications and environment related states make manual static configuration of mapping strategies impractical. Therefore, the *second evaluation goal* was to first investigate the correct operation of executing the appropriate mapping strategy in KBNMap in different tests, and then to assess the applicability of the KBNMap implementation for adaptively selecting mapping strategies, by evaluating the appropriate weightings of mapping strategies produced by the Bayesian Network model given different tests. This evaluation goal also reflects how well the KBNMap system addresses the second main part of the research question (Section 1.2): "distributing information in an application,

and ontology appropriate manner". This goal was addressed through the combination of experiment 2 and experiment 3.

7.3 Evaluation Design Overview

As described in Chapter 7, each individual KBNMap router is facilitated with a Bayesian Network (BN) model for mapping strategy selection, this BN model is the key component to realize KBNMap's self-configuring functionality, as it is able to automatically select the correct mapping strategy according to the observed evidence entered into the BN model. The evidence entered consists of the observed state values of identified trigger factors through the edge nodes in the BN model. For example, if the "available memory resource" is observed as "small" state by KBNMap router, the evidence value "small" is entered into the AvailableMemory node of BN model. In addition, the evaluation undertaken for KBNMap routers should allow for varying combinations of evidence of identified key factors in order to comprehensively understand the effect of the introduced mechanisms and the capability of KBNMap to adapt to changes of key factors. However, as shown in Figure 5-11, there are 14 trigger factors with each factor having three states in BN model, which leads to a total number of combinations of varying evidence of 3^{14}. Investigating the performance of KBNMap under such a large number of combinations was impractical. Thus, it was decided to identify the key factors that have the most significant impact on selecting mapping strategies, so that the KBNMap routers could be evaluated with a reduced number of combinations of varying evidence. In addition, as KBNMap mainly aims to resolve heterogeneous information, the evidence of selected influential factors need to be grouped into interrelated sets of factors on the basis of combining the observed unknown data factor with other important factors.

Thus the method of evaluating the introduced mechanisms was divided into a number of stages:

- The *first stag*e was to define and categorise and group the most important trigger factors to build a smaller set of tests (detailed in Section 7.4).
- The *second stage* was to provide a routing ontology, a number of mapping files, and referenced ontologies with varying ontology characteristics for use as a testing set by the applications and routers. In particular, each individual mapping file has two constituent ontologies in which one is the routing ontology and the other one is a selected referenced ontology. The mapping files provide mapping relations between the routing ontology and referenced ontologies (detailed in Section 7.5).

- The *third stage* was to design the general network setup, which included 1) synthetically generating a configurable test-set of publications and subscriptions to inject into the KBNMap deployment; 2) designing a hierarchical network topology that would be adopted to build up the router network; 3) designing different types of routers with different ways of using mapping strategies (detailed in Section 7.6).
- The *fourth stage* was to design the experiment setup, particularly to define a set of measurable performance key metrics (detailed in Section 7.7).

The routers were then set up in two different configurations, and their performance measured. Prior to evaluating the effect of the introduced adaptive mapping service on the performance of KBNMap routers it is worthwhile to comprehensively understand the extra overhead introduced by each particular individual mapping strategy. KBNMap was first initialised to employ statically encoded (hard-wired) strategy selection for each of the 3 strategies:

1. **KBNevery** is the router enabled with only the **Every** strategy;
2. **KBNapproOnly** is the router enabled with only the **ApproOnly** strategy;
3. **KBNapproInd** is the router enabled with only the **ApproInd** strategy.

The performances of the three types of network were then compared. These evaluation tests are described in experiment 1 (Section 7.8).

Another configuration was designed called **KBNRan** mainly to compare its performance against that of KBNMap with respect to the effect of the adaptive mapping strategy selection and the capability of KBNMap to adapt to the dynamism of identified key factors. The major difference between KBNRan and KBNMap is that the KBNRan uses a strategy that is randomly selected rather than KBNMap which uses the strategy that is selected by the Bayesian Network based mechanism. The reason to use KBNRan as the reference router in this evaluation is that the random selection process is generally considered as a fast selection process (by using a random number generator); so that the performance of Bayesian Network based selection process could be compared. This experiment was also designed to illustrate the need for a more sophisticated strategy selection mechanism. Thus, the performance of these two different networks were then evaluated and compared. The evaluation work is described in experiment 2 (Section 7.9). Furthermore, the adaptability of the KBNMap to the changes of key trigger factors could also be evaluated through observing the updated weightings of mapping strategies in the tests of experiment 2. This evaluation work is discussed in experiment 3 (Section 7.10).

7.4 Stage 1: Characterisation of Testing Sets

The trigger factors influencing mapping strategy selection are identified and grouped into three categories in Table 5-8. For convenience in illustrating the selection of the most important factors, all influential factors identified in Section 5.4.3.2 are briefly described in the following:

Ontology-based trigger factors refer to the varying characteristics of routing ontology, mapping files and referenced ontologies. Especially the size and DL expressivity of ontologies determine the reasoning performance of KBNMap routers (see Section 5.2.1):

1. **Routing ontology size***:* refers to the number of statements of the routing ontology. The size of routing ontology will be altered when the new mapping files, mapping relations, and referenced ontologies are merged into the routing ontology.
2. **Mapping ontology size:** refers to the size of the mapping file that is loaded and merged into the routing ontology.
3. **Required reference ontology size:** sometimes the ontologies referenced in the appropriate mapping file would be merged into the routing ontology as required.
4. **Routing ontology expressivity:** refers to the DL expressivity of the routing ontology; the DL expressivity of routing ontology will be altered when the new ontologies are added into the routing ontology.
5. **Required referenced ontology expressivity:** according to our findings in Section 5.3.1, the merged ontology expressivity is the union of its constituent ontologies' expressivities. In general, the constituent ontologies comprising mapping file are the routing ontology and other referenced ontologies.

Application-based trigger factors refer to the varying application characteristics (see Section 5.2.2):

6. **Subscription reception rate:** refers to the number of subscriptions received by a router per minute. The varying subscription rate influences the workload of a router for routing messages and matching events, and also the rate of unknown data occurrence**.**
7. **Unsubscription reception rate:** refers to the average rate at which unsubscription requests arrive at the router.
8. **Publication reception rate:** refers to the number of publications received by a router per minute.

9. **Observed unknown data rate:** refers to the number of unknown subscriptions/ publications entering into the network per minute, which is observed by the human administrator.

10. **Mismatch tolerance**: refers to the tolerance capability of KBNMap applications against the message mismatching rates.

Environment-based trigger factors refer to the varying environment characteristics influencing on the mapping strategies selection (see Section 5.2.3):

11. **Network scale:** refers to the number of routers comprising a KBNMap network. The varying scale of network influencing the rate of unknown data occurred.

12. **Number of mapping ontologies:** refers to the number of mapping files stored in the mapping store of a router.

13. **Number of referenced ontologies:** refers to the number of ontologies referenced in the mapping files.

14. **Memory resources:** refers to the computing resources allocated to a router. The mapping strategies are generally considered as resource consuming prone.

7.4.1 Selected Trigger Factors

The trigger factors described above are implemented as the leaf nodes of the BN model in Section 5.4.3 and each node has three states with *small, medium, large* value. The combination of states of any influential factors could stand for a testing scenario for evaluating the KBNMap performance. For example, how would the KBNMap deployment perform if it is evaluated in the scenario where the network size is *large*, unknown rate is *small*, and memory resource allocated is *large*. However, as discussed in Section 73, it is impractical to evaluate the KBNMap deployment under all possible combinations of influential factors. Hence it was decided to select the factors that have most significant influence on selecting mapping strategies. The selection of the most important factors is in accordance with the ranking of direct parent nodes of final strategy selection node in the BN model described in Section 5.4.3.5. The selected factors are listed below:

- **Observed unknown data rate**: this factor is selected as the one that has the most significant influence on selecting mapping strategies since it mainly determines the rate of unknown data that actually received by the router. For example the "*Every mapping file*" Strategy (**Every**) is well-suited for the environment where the unknown data is large, while the "*Appropriate individual mapping*" Strategy (**ApproInd**) is best where the unknown data occurrence is small. The "*Appropriate*

mapping file" Strategy (**ApproOnly**) is a moderate trade-off in between.

- **Subscription and Publication rate**: these two factors are summarised as the message rate. The rate of messages determines the rate of message matching executed by the KBN router where the unknown data are identified. When taken together with the "unknown data rate" the occurrence rate of subscriptions and publications with ontological terms not expressed by the routing ontology can be determined.
- **Tolerance capability**: this factor is selected since a low tolerance of missed mappings, resulting in false-positive or false-negative subscription matches, is a strong indicator that the **ApproOnly** and especially the **ApproInd** strategies may be inappropriate unless other factors outweigh this factor's influence.
- **Available memory resources**: is the last major influencing factor due to the differing resource consumptions of the mapping strategies. The **Every** strategy is preferable when the available resource are sufficiently large, while the **ApproInd** strategy is best-suited when memory is limited since only very targeted individual mappings are loaded.

Of the four main factors, the "Observed unknown data rate" factor is the most influential as shown in weightings in Chapter 5, so we created 3 test cases where each of the other 3 factors were considered in terms of their effect in combination with the "Observed unknown data rate" factor.

In addition, it can be observed that not all ontology based trigger factors are selected, as only one ontology will be used as the routing ontology so that the size and expressivity of original routing ontology is fixed. However, the size and expressivity of the routing ontology will be continuously changing along when new mappings or ontologies are introduced. In addition, it is not possible to predict the required mapping files and referenced ontologies needed to be merged into the routing ontology, as the ontological terms used in the unknown subscriptions and publications are randomly derived from the referenced ontologies. The characteristics of ontologies used for the evaluation are described in Section 7.7.

7.4.2 Three Categorized Testing Sets

The remaining factors were arranged into three testing sets, each which is the varying states of "observed unknown data rate" combining with the varying states of one of the other selected factors. The reason for this is shown below:

1. The design of KBNMap infrastructure is mainly targeting the capability to cope with heterogeneity, the evaluation for KBNMap has to allow for the unknown data

variation, the varying "observed unknown data rate" variable has to be taken into account for designing any case scenario.

2. Even though the number of selected factors has been reduced to 4 from 14, it is still considered to be impractical to evaluate KBNMap given all possible combinations of the states of all selected factors.

Therefore the *observed unknown data* factor was combined with the other selected factor respectively to build the case scenarios. The three categorised sets are described in the following subsections.

7.4.2.1 Testing Set A: Unknown Data Rate with Message Rate

The purpose of this testing set is that the KBNMap router would self-select a well-suited strategy for resolving heterogeneous information in an efficient manner to adapt to the varying states of unknown data and message rate. Even though there are 9 possible cases existing, only three cases are adopted for evaluation in this testing set:

Unkown data rate (State)		**Message Reception Rate (state)**		
		Small	***Medium***	***Large***
	Small	Case 1		
	Medium		Case 2	
	Large			Case 3

The reason for the choice is that the adopted cases cover the practical scenarios when it comes to unknown data occurrence, where the small message rate in the real deployment is likely to introduce rare unknown data while the large number of messages received by the router seems contain more unknown data. In addition, the other cases could be evaluated in the Testing Set B and Testing Set C, as the configuration of the parameters is the same (See Section 7.9.4 and Section 7.9.5). For example, the configuration of the case where the large message rate with small unknown data in this testing set is the same as the case 7 in Testing Set B.

7.4.2.2 Testing Set B: Unknown Data Rate with Tolerance

The purpose of this testing set is to evaluate the capability of the router to cater for the dynamics of the KBNMap applications tolerance level and how the router manages the extra processing of selecting and using strategies in an efficient way. For example, once the tolerance of the KBNMap applications is increased from *small* to *large* under a large unknown data occurring environment, the router should take some actions to adapt to this

change. There are nine cases here that are required to measure the correct operation of KBNMap:

Unkown data rate (State)		Tolerance capability (state)		
		Small	***Medium***	***Large***
	Small	Case 1	Case 2	Cas3
	Medium	Case 4	Case 5	Case 6
	Large	Case 7	Case 8	Case 9

7.4.2.3 Testing Set C: Unknown Data Rate with Memory Resource

The purpose of this set is to evaluate the capability of the router to adapt to the changes of unknown data and resources allocated. Given the situation where the rate of unknown data occurrence observed in the network ranges from ***small*** to ***large,*** the KBNMap might adapt to select different mapping strategies to cater for the changes. In addition, due to the differing resource consumptions of the mapping strategies, the selection of mapping strategy in this situation also needs to take into account the memory resources available in the routers:

Unkown data rate (State)		Memory Resources (state)		
		Small	***Medium***	***Large***
	Small	Case 1	Case 2	Cas3
	Medium	Case 4	Case 5	Case 6
	Large	Case 7	Case 8	Case 9

7.5 Stage 2: The Ontologies Used

Three types of ontologies are used in the evaluation: a **routing ontology** that is already loaded by each router for matching and routing ontological subscriptions or publications; a number of **referenced ontologies** that contain ontological terms which are used by *unknown subscriptions or publications*; and a number of **mapping ontologies** that each contain semantic mappings relating terms between the routing ontology and referenced ontologies. Once a mapping ontology or a referenced ontology is incorporated into the routing ontology, the term **merged ontology** (it is also a new routing ontology) will be used to refer to the resulting ontology.

The characteristics of ontologies used in the evaluation are specified in line with the ontology characteristics specified in the ontology characteristics identification Section (Section 5.3) in Chapter 5 (see Table 5-1). The selection of the ontologies used for evaluation is described in the following subsections.

7.5.1 Dataset Description

A set of ontologies from other researchers were chosen for the experiments rather than creating ontologies from scratch. The criteria used was: first, they should be data sets used for evaluating current ontology matching systems, of relatively high quality; second, ontologies should be independently created by different people. In this sense, the ontologies can be considered as representative of the natural range and diversity of ontologies that can be expected; finally, ontologies should range from small number of statements to large number of statements and from simple expressivity to complex expressivity, so that they can reflect diverse ontology size and complexity, which are important ontology characteristics identified in this thesis.

For these reasons, existing ontologies and their existing mapping files from the ***Conference Track*** in the OAEI website (OAEI 2007) were selected as the dataset, which is a well-known international initiative set up as a contest to evaluate state of the art ontology matching systems.

There are fourteen ontologies collected by this track in the domain of conference organization. The four main features of these ontologies are:

1. General understandable domain as most ontology engineers are familiar with conferences.
2. Independence of ontologies as they were developed independently and organize the conferences using different terminologies.
3. Diversity in the number of classes, properties, individuals, their DL expressivity.
4. Diversity in the underlying sources, because they have been designed according to either an actual conference; or actual software tools for supporting conference organization; or experience of the people who were involved in organizing conferences.

7.5.2 Routing and Referenced Ontologies

As the feature of this data set satisfies the requirements of ontologies identified in Section 5.2.1, the conference track ontologies from OAEI were selected as the routing and referenced

ontologies for the tests. Table 7-1 summarizes all ontologies chosen according to the ontology characteristics of interest (see Table 5-8).

Table 7-1 is formatted in line with Table 5-1, where the "No.Statement" heading of second column indicates the number of statements of an ontology, reflecting the size of ontology (see Section 5.2.1), this value was obtained from Jena DL API (Carroll et. al., 2004) after the Jena API loaded and reasoned each individual ontology. The "Size indicator" heading specification is based on the BN node boundary value table (see Table 5-8) in Section 5.4.3. The "DL Expressivity" heading denotes the presence of description logic features in the ontology, which capture ontology's reasoning complexity (see Section 5.2.1); this value was obtained from the SWOOP (Swoop 2004) OWL Ontology Browser and Editor which . The "related Link" denotes the origin of the ontology. Finally the "note" heading indicates as what kind the ontology will be used in evaluations.

Since many of the mappings that are created by the mapping engineering teams (Svab et. al., 2006; OAEI 2007) in the contest used the *ekaw* ontology as their canonical ontology, this ontology was chosen as the main routing ontology in this evaluation. A detailed description is given in the next subsection.

Table 7-1: Individual Ontology Characteristics

Name	No. statement	size indicator	DL Expressivity	Related Link	Note
Ekaw	**3,587**	**Medium**	**ALCR+HIN**	**http://ekaw.vse.cz**	**Routing ontology**
Sofsem	1,387	Medium	ALC+HIF(D)	http://www.sofsem.cz	Referenced ontology
Sigkdd	682	Small	ALC+I(D)	http://www.acm.org/sigs/sigkdd/kdd2006	Referenced ontology
Iasted	2,981	Medium	ALC+IF(D)R	http://iasted.com/conferences/2005/cancun/ms.htm	Referenced ontology
Micro	852	Small	ALC+OIF(D)	http://www.microarch.org	Referenced ontology
Confious	1,625	Medium	ALCR+HIN	http://www.confious.com	Referenced ontology
Pcs	794	Small	ALC+HIF(D)	http://precisionconference.com	Referenced ontology
OpenConf	1,597	Medium	ALC+OIF(D)	http://www.zakongroup.com/technology/openconf.shtml	Referenced ontology
Conftool	1,252	Medium	ALCR+I(D)	http://www.conftool.net	Referenced ontology
Crs	318	Small	ALC+IF(D)	http://www.conferencereview.com	Referenced ontology
Cmt	859	Small	ALC+IF(D)	http://msrcmt.research.microsoft.com/cmt	Referenced ontology
Cocus	1,041	Medium	ALC+IF(D)	http://cocus.create-net.it/	Referenced ontology
Paperdyne	2,050	Medium	ALC+HIF(D)	http://www.paperdyne.com/	Referenced ontology
Edas	2,873	Medium	ALC+IF(D)	http://edas.info/	Referenced ontology

7.5.3 Mapping Ontologies

As there was a large volume of mapping files provided by six mapping engineering teams (Svab et. al., 2006) in the OAEI contest, a particular set of mapping files from one particular team were selected in order to meet the experimental requirement. The mapping files from the SEMA team (Spiliopoulos et. al., 2007) were selected, since the SEMA team ranked in first place amongst the six teams in the semantic matching contest. SEMA team provided thirteen alignment ontologies by mapping all ontologies to the *Ekaw* ontology, these alignment ontologies were selected as mapping files in the experiments. Thus the *Ekaw* ontology was used as the **routing** ontology (indicated in the column **Note** as routing in Table 7-1) and the other ontologies were used as the **referenced** ontologies (indicated in the column **Note** as reference in Table 7-1). The mapping formats of these alignment ontologies are expressed in alignment format used in the contest (Euzenat 2004). The format is different from the OWL format used in our mapping method, and so it was necessary to translate the referenced alignments ontologies into OWL format. The translated OWL mapping files were obtained using the OISIN mapping translating tools (O'Sullivan 2006) which could keep the semantics of the original mappings. The characteristics of the translated mapping ontologies are shown in Table 7-2.

Table 7-2: Characteristics of Mapping Ontologies

Mapping Name	Number of statements (before reasoning)	DL Expressivity	Constituent Ontologies
Ekaw_sigkdd	1,874	ALCR+HIN	Ekaw & sigkdd
Ekaw_PCS	1,978	ALCR+HOIF(D)	Ekaw & PCS
Ekaw_paperdyne	3,243	ALCR+HOIF(D)	Ekaw & paperdyne
Ekaw_OpenConf	2,836	ALCH	Ekaw & openConf
Ekaw_MICRO	2,066	ALCR+HOIF(D)	Ekaw & Micro
Ekaw_iasted	4,172	ALCR+HIN	Ekaw & iasted
Ekaw_edas	4,085	ALCR+HIN	Ekaw & edas
Ekaw_crs	1,508	ALCR+HIN	Ekaw & crs
Ekaw_confOf	2,438	ALCR+HIN	Ekaw & confTool
Ekaw_confious	2,834	ALCR+HIN	Ekaw & confious
Ekaw_sofsem	1,336	ALCR+HIN	Ekaw & Sofsem
Ekaw_concus	2,238	ALCH	Ekaw & concus
Ekaw_cmt	2,065	ALCR+HIN	Ekaw & cmt

7.6 Stage 3: Design of General Setup

This section describes the specification of the required parameters for setting up the router network, including the network topology, the configurations of routers, the characteristics of subscribers and publishers, which provide a basic guideline to design the testing case scenarios.

7.6.1 Design of KBNMap Network Topology

As stated in Section 3.2.2, the routing topology plays an important role in determining the scalability of a SBPS system. In addition, the router topology that is used has a major impact on any experiments. The authors in (Terpstra et. al., 2003) identified a number of main parameters that can characterize a Pub/Sub router topology. As one of the main evaluation goals is to investigate the performance of the KBNMap implementation with respect to message delivery, the main parameters in (Terpstra et. al., 2003) that characterize a router topology with respect to information delivery are only outlined below:

- the number of routers
- the number of neighbor routers, which may be constant or varying
- the length of path between subscriber and publisher which influence the message delivery time

As the research goal of this thesis is to demonstrate the distribution of heterogeneous information in a large-scale networking environment, the topology adopted in this evaluation concentrates on a simple large hierarchical topology, which consists of 15 nodes running as dedicated KBNMap routers. This number of nodes indicates the KBNMap network is in a *large* scale, according to the boundary values of network scale that are identified in Section 5.4.3.2.

This hierarchical topology, shown in Figure 7-1 arranged the routers, with a publishing client on one side and a subscribing client on the other side, thereby artificially simulating a long route between clients to stress the KBNMap deployment. It should be noted that given this routing topology, there is no practical limits to run the experiments on the computer used for evaluation. In addition, the long path brings more opportunities for combinations of mapping strategies executed in the deployment and potentially introduces more diversities of unknown data occurs in the network. Despite this there are other alternative network topologies that could be deployed for the experiment. For example, the routers can be deployed in a way, which allows several intermediate routers to have two or three sub routers rather than one sub router in Figure 7-1. However, these kind of alternative topologies only makes the path

between clients shorter than the topology adopted. The assumption made here is that if the end-to-end publication delivery time is evaluated efficiently given the topology presented in Figure 7-1, the end-to-end publication delivery time will be evaluated shorter in other alternative topology. Each router was pre-loaded with the *ekaw* ontology as their routing ontology and the large set of mappings and referenced ontologies pre-loaded into their mapping stores. Both of the clients were pre-loaded with the *ekaw* ontology, and 13 other ontologies that were mapped to the *ekaw* ontology. The clients then inject synthesised semantic publications, subscriptions and un-subscriptions drawn from these 14 ontologies into the network.

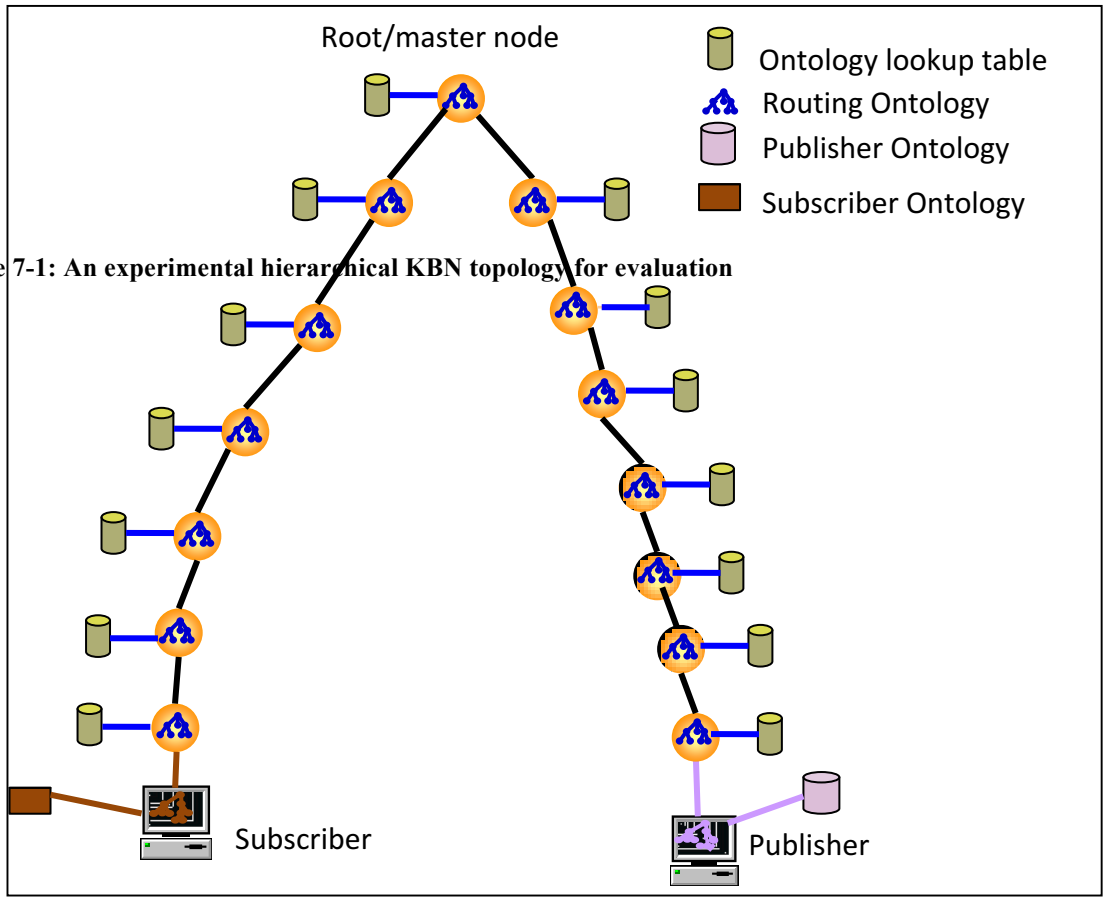


Figure 7-1: An experimental hierarchical KBN topology for evaluation

The type of router has a significant impact on the performance of a network. In this evaluation three different configurations of routers were introduced in order to build different typed KBNImpl networks:

- **KBNMap**: runs the *adaptive strategy selection engine* to **continuously** select an appropriate mapping strategy to dynamically adapt to the changes of identified key influential factors that are monitored.

- **KBNRan**: randomly selects mapping strategies to deal with the encountered heterogeneous information, independent of the changes of the identified key influential factors. This router is mainly used as the reference for evaluating the performance of KBNMap. The random selection of strategies is achieved through a random number generator which randomly generates strategy ID (1 for the Every strategy, 2 for the ApproOnly strategy, and 3 for the ApproInd strategy). Random number generation is generally considered as a fast and efficient process. Therefore, the performance of KBNMap for selecting mapping strategies will be compared to the performance of **KBNRan** in order to observe the effect of dynamic strategy selection.
- **KBNFix**: is a special KBNMap router which was initialised to employ statically encoded (hard-wired) strategy selection for each of the 3 strategies. This type of router is further categorised into three according to different strategies specified: KBNevery, KBNapproOnly, KBNapproInd. These routers are mainly used to evaluate the effect of different mapping strategies on the performance of KBNImpl routers.

7.6.2 Characteristics of Subscriber

The authors in (Terpstra et. al., 2003; Carzaniga et. al., 2002; Keeney et. al., 2006b) state that the characteristics of subscribers have an impact on the performance of a CBN network hence the characteristics of KBNMap subscribers also have influence on the performance of a KBNMap network. Basing on the parameters of subscriber identified in (Carzaniga et. al., 2002; Keeney et. al., 2006b). The main parameters of KBNMap subscriber for this evaluation are listed below:

- The number of subscribers
- The type of the subscriptions
- The number, type of the application ontology being used by the subscriber
- The rate of subscriptions
- The rate of unknown subscriptions

As demonstrated in Figure 7-1, there is only one subscriber connecting with the left-side leaf router. The reason for configuring one subscriber is that the main issue of concern to the KBNMap router is whether the subscriptions received are known data or unknown data rather than the numbers of subscribers connected with it. In addition, the rate of subscription assigned ranges from *small* to *large*, which covers all the possible states of the subscription

rate. Therefore, it is not necessary to configure more subscribers to the router. Moreover, to assign one subscriber allows for straightforward calculation of the subscription forwarding time. As the main goal of this evaluation is to test the router's capability to resolve ontological heterogeneous information, the subscriber only has ontological types of subscriptions. The subscription generated by the subscriber contains only one constraint filter: [attribute_name: ontological_operator: attribute_value]. As the matching publications and subscriptions must have the same attribute name, for convenience, all subscriptions and publications are given with a common attribute name: TEST. In addition, as the *semantic equivalence* is the only mapping relation encoded in the mapping files, the ontological operator used for all subscriptions is Ontological_equivalence (@=). As for attribute values, they are either derived from **routing** ontology (Ekaw) to form the **known** subscriptions or are specified using the ontological terms from the other 13 **referenced** ontologies to form the **unknown** subscriptions. Thus, the basic format of known subscription is: SUB (TEST: @=: any term from **ekaw** ontology) while the unknown subscription is: SUB (TEST: @=: any term from the other 13 **referenced** ontologies). Finally, in line with the specification of subscription rate in Section 7.4.2, the rate of subscriptions and unknown subscriptions are required to be assigned ranging from *small* to *large* value.

7.6.3 Characteristics of Publisher

In line with the characteristics of subscriber, the characteristics of publishers also influence the behavior of a KBNMap network. Hence, based on the specification of publisher in (Terpstra et. al., 2003; Carzaniga et. al., 2002; Keeney et. al., 2006b), the main parameters of KBNMap publisher are shown below:

- The number of publishers
- The type of the publications
- The number, type of the application ontology being used by the publisher
- The rate of publications
- The rate of unknown publications

As demonstrated in Figure 7-1, there is only one publisher connecting with the right side leaf router. The reason for having one publisher is the same as the argument for one subscriber. The publisher also only has ontological types in publications. The basic format of a publication is then given as PUB [attribute_name: attribute_value]. In order to maximize the possibilities to match subscriptions, the attribute name for all publications is also given as "TEST". The attribute value is derived either from the **ekaw** routing ontology to form **known** publications, or other 13 referenced ontologies to form **unknown** publications. Therefore, the basic format of known publication is: PUB (TEST: any term from **ekaw** ontology) while the

unknown publication is: PUB (TEST: any term from 13 **referenced** ontologies). Finally, in line with the specification of publication rate in Section 7.4.2, the rate of publications and unknown publications are required to be assigned ranging from *small* to *large* value.

7.7 Stage 4: Experiment Setup

The KBNMap router implemented with the adaptive strategy selection mechanism discussed in Chapter 6 is used as the basis of the **KBNMap** deployment. The **KBNMap** router with random mapping strategies selection mechanism rather than adaptive mapping strategy selection is used for configuring the **KBNRan** deployment. Finally, a special **KBNMap** router which was initialised to employ statically encoded (hard-wired) strategy selection for each of the 3 strategies was used to configure the **KBNFix** networks (e.g., **KBNevery** network, **KBNapproOnly** network, **KBNapproInd** network). In order to minimise the adverse effect of inconsistent network connection speeds on reasoning performance all tested ontologies and their imported ontologies were cached locally on the machine running the tests[v].

7.7.1 Measured Metrics

The metrics illustrated below were used in this evaluation to compare the performance of the **KBNMap**, **KBNRan** and different **KBNFix** networks:

- **Strategy Execution time:** the time taken by the router to execute an individual mapping strategy. Measuring each individual mapping strategy execution time allows the calculation of how much the mapping strategy affects the performance of the router.
- **Publication matching time**: refers to the overhead of the router for matching a publication it receives to subscriptions covering the received publication. In addition to the time taken for a router to process a publication to search the subscription tree for finding the appropriate subscriptions, it also includes the time taken for a router to execute a specific semantic mapping strategy, if the ontological terms in the received publication are not from the routing ontology. Assume that e represent an event contained in the publication, P_m represents the time taken for the publication

[v] All testing cases in the experiment were run on a Sony VAIO VGN-CR31Z laptop with 2GB of RAM and an Intel Core2 Duo processor T8300 (2.40 GHz and 800 MHz FSB), running Windows Vista Business Service Pack 1. For Java-based tools, Sun's JDK 1.6.0 and Eclipse 3.3.2 was used.

containing e to iterate through subscription tree of a router, S_S represents the strategy selection period, and S_e represents the strategy execution time, then the matching time $M(e)$ for a single publication of an event e in a router is given as follows:

$$M(e) = P_m + S_s + S_e$$

- **Publication end-to-end delay**: also known *as time to delivery*, is defined as the time between a publisher publishing an event and a subscriber that shows interests of this event, receiving a publication containing the same event. It is the task of the overlay network to effectively reduce the overall, end-to-end delay of event publications, in an event router network. If event e represent an event, *M(e)*represents the event matching time, and n is the total number of routers in the path between a publisher and subscriber, then the delay $D(e)$ for a single publication of an event between a publisher and subscriber is given as follows:

$$D(e) = \sum_{i=1}^{n} M_i(e)$$

- **Frequency of strategy usage:** is defined as the number of times a strategy has been triggered by a router during entire network runtime period. Through observing the distribution of each strategy being used, it is helpful to examine the correct operation of KBNMap for selecting mapping strategies given the weightings of mapping strategies produced by the BN model. In addition, it also helps in explaining the impact of mapping strategy execution, as the more times a strategy or several strategies being used, the more overhead will be consumed during the process of publication matching and delivering.

The System.currentTimeMillis () method from the java.lang package in Java Standard Edition 6 JDK[vi] was employed to measure time for the first two metrics, as it is usually used for measuring time in Java. It is important to note that the publications and subscriptions generated are on the same host in order to measure end-to-end delay. In addition, a number counter was employed to measure the frequency of mapping strategies being used. The data for each metric was collected and recorded into different log files separately during the network runtime period.

[vi] http://java.sun.com/javase/6/docs/

The metrics illustrated above were used as part of the evaluation to determine the performance characteristics of different KBNImpl networks.

7.8 Experiment 1: Comparing the Strategies

Prior to evaluating the effect of introducing the adaptive mapping service on the performance of **KBNMap** routers, it is appropriate to examine the extra overhead introduced by each particular individual mapping strategy for resolving heterogeneous information.

7.8.1 Design

This experiment was designed to test the effect of different mapping strategies on the performance of **KBNMap** routers with respect to delivering heterogeneous information.

Three types of **KBNMap** deployments that are composed of different types of **KBNFix** routers described in Section 7.6.2 were configured. The detailed information is presented below:

- **KBNevery Network:** refers to the network that is composed of KBNFix routers equipped with the **Every** strategy.
- **KBNapproOnly Network:** is the network in which all routers are equipped with the **ApproOnly** strategy.
- **KBNapproInd Network:** refers to the network that consists of routers equipped with the **ApproInd** strategy.

The performance of each individual strategy was then tested with respect to different levels of unknown data:

- Where the rate of messages across the network is *high* while the proportion of unknown data contained in messages is relatively *small*.
- Where the rate of messages across the network is *high* while the proportion of unknown data contained in messages is raised to a *medium* level.
- Where the rate of messages across the network is *high* while the proportion of unknown data contained in messages is raised to a *high* level.

Table 7-3 represents the configuration of the parameters in detail.

For the purpose of evaluating networks in high load configurations, according to the specification of boundary values of states of the trigger factors established in Section 5.4.3, the message rates are specified as a *large* level (30 messages/min), while the observed

unknown data rate ranges from a *small* (2 messages/min) to a *large* (12 messages/min) level in different cases. In addition, the memory resources allocated to each router (40 Mb) and tolerance capability of end applications are specified as *small*, so that whether each mapping strategy works in a limited resource deployment configuration.

Table 7-3: Factor configurations to compare semantic interoperability strategies

	Variable Name				
	PubRate (/min)	***SubRate (/min)***	***Unknown Data Rate (/min)***	***Memory (MB)***	***Tolerance***
small unknown data	30	30	2	40	small
medium unknown data	30	30	5	40	small
large unknown data	30	30	12	40	small

The general experiment setup (e.g., network topology, routing and mapping ontologies used, measured metrics) described in Section 7.6 is applied in this experiment. Furthermore, the following metrics are measured: *publication matching time, publication end-to-end delay*, and *strategy execution* time, which are described in details in Section 7.7.1. The experiment ran for over 1 hour, once for each KBNFix deployment.

7.8.2 Results

This section briefly describes the results and analysis of the representative cases and gives an overall summary of the experiment. It is important to note that in all of the figures about **publication matching time** in this chapter, the X-axis corresponds to elapsed time in seconds during which the experiment was running, while the Y-axis is the average time in milliseconds for matching a new publication with the subscriptions stored in the subscription tree in an individual router. the reason for this format is that it is appropriate to record every second of operation of the experiment, so that the entire trend of publication matching times can be clearly demonstrated. In addition, publication matching is considered as a fast process which takes less than 1second in general (Carzaniga et. al., 2002; Keeney et. al., 2006b), thus the milliseconds used can accurately record the time taken for each single publication matching. Furthermore, in all of the figures about **publication end-to-end delay**, the X-axis corresponds to the number of publications received by the subscriber after the transmission, while the Y-axis is the average end-to-end time in milliseconds for delivering a publication from the publisher to the subscriber across the entire router network.

7.8.2.1 Case 1: High Message Rate with Small Unknown Data

This case evaluates the performance of each KBNFix network in the scenario where the rate of messages across the network is high while the proportion of unknown data contained in messages is relatively small.

Results of publication matching time

It can be concluded from Figure 7-2 that the KBNevery deployments outperforms the other networks with unknown data of small, whereas the KBNapproOnly network performs a little worse than the KBNapproInd network. In addition, the KBNapproInd network has similar but more efficient behavior than the KBNapproOnly network.

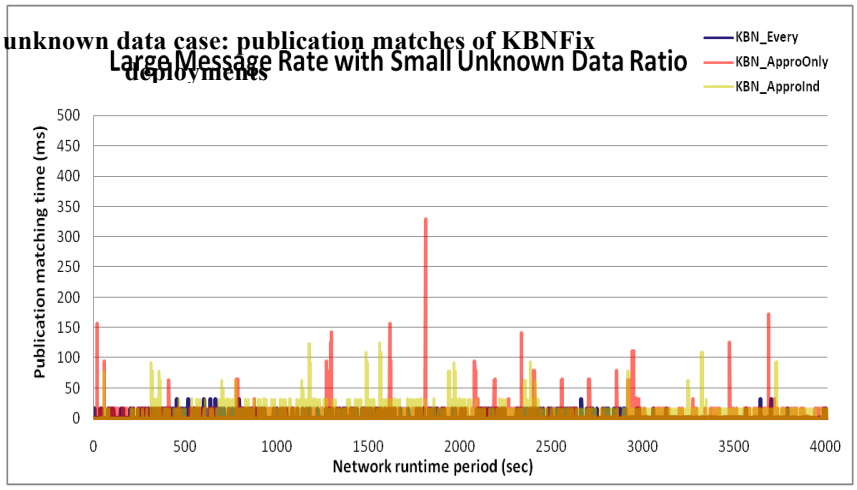


Figure 7-2: Small unknown data case: publication matches of KBNFix deployments

Analysis of publication matching

As the main characteristic of the **Every** strategy is to load all ontologies stored into the routing ontology, it is inevitable that it consumes a considerable amount of overhead. Therefore, the efficient performance of KBNevery with publication matching indicates that all the strategy executions occur in the operation of processing subscriptions prior to publication matching. On the other hand, the nature of the **ApproOnly** strategy is to only load and merge the relevant ontology into the routing ontology, which leads the routers to occasionally consume a moderate amount of overhead for publication matching. Hence, the inefficient performance observed for KBNapproOnly indicates that the **ApproOnly** strategy is triggered several times in the process of publication matching, in addition to being applied in the subscription process. KBNapproInd's characteristic of loading individual mappings into the routing ontology, leads the routers to resolve the encountered unknown data on an individual basis.

Results of publication end-to-end delay

It can be observed from Figure 7-3 that the KBNapproInd deployment has far superior performance than the others with small unknown data occurrence, and the response times for publication matching are well below 1000ms. It can also be clearly observed that both KBNevery and KBNapproOnly have several peaks of publication end-to-end delay, especially,

in KBNevery, the peaks are obviously demonstrated with the maximum value of above 1400ms.

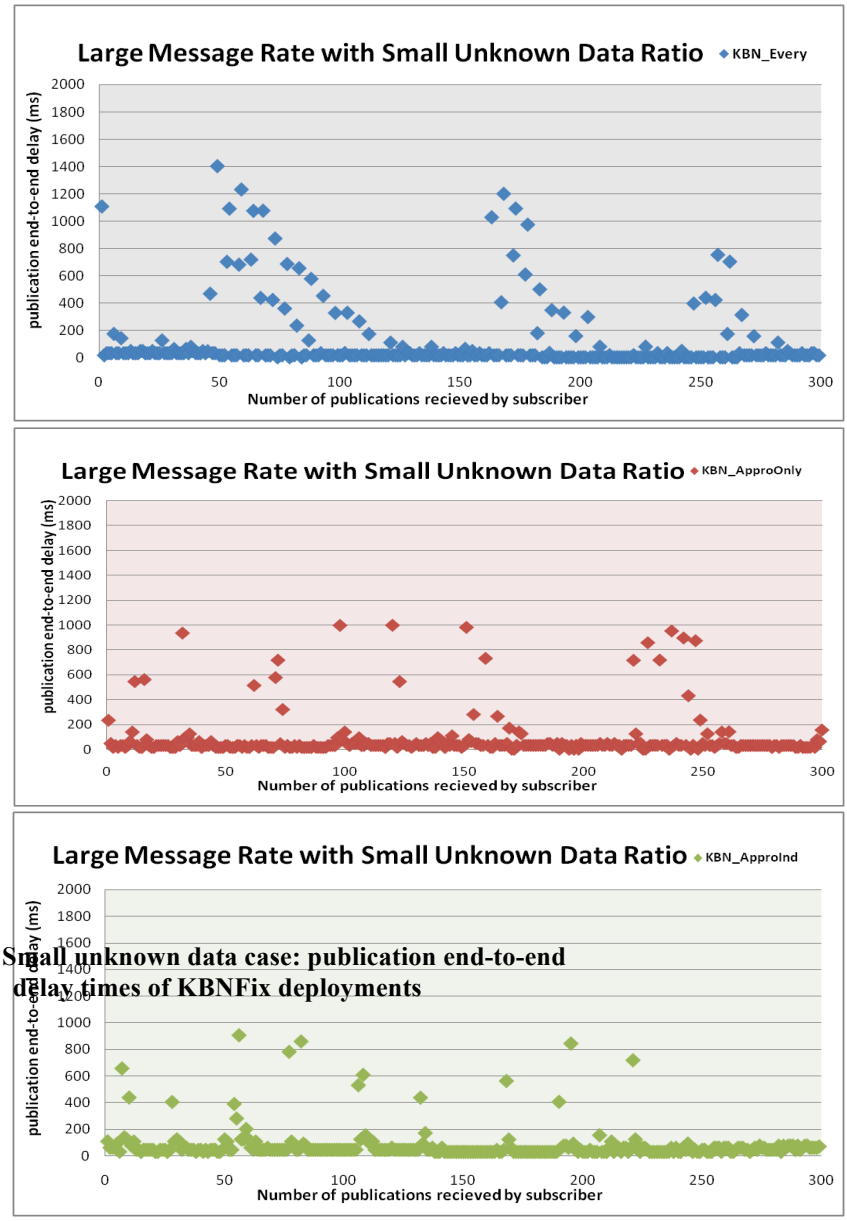


Figure 7-3: Small unknown data case: publication end-to-end delay times of KBNFix deployments

Analysis of publication end-to-end delay

The main reason why the KBNapproInd implementation outperforms the others is that there are more *known* (ontological terms expressed by the routing ontology) publications being delayed by the operation of mapping strategies in the KBNevery and KBNapproOnly networks, and the efficient operation of the **ApproInd** strategy leads the network to having rare *known* publications being delayed. It is important to note that while the router is executing the strategy operation, it will temporarily freeze parts of the subscription and

publication processing algorithm as the knowledge-base (routing ontology) on the router is being updated.

The reasons for the peaks of end-to-end delay can be explained as follows: according to the nature of the network topology adopted in this experiment (Figure 7-1), in KBNevery deployment, once the leaf router that links with the subscriber client encounters the first unknown subscription, it loads all ontologies stored in the mapping store to deal with it by triggering the Every strategy. If this unknown subscription is resolved and it is the most general subscription (according to the covering relation algorithm in Section 3.4.7), the router forwards it to the parent router, which will also perform the same operation as the leaf router. This process continues until the unknown subscription arrives and is resolved in the root router. It should be noted that during the entire Every strategy execution period, there is a flow of publications being delivered to the master router along the left side routers; and there is a publication being involved in the entire strategy execution process, which causes it to have the highest publication delivery time. In addition the publications following this worst publication would have decreasing rates of delivering overhead. As for the KBNapproOnly network, it is the same working principle as that of KBNevery. The only difference with KBNevery is that KBNapproOnly has much shorter strategy execution but the strategy is executed more frequently.

7.8.2.2 Case 2: High Message Rate with Medium Unknown Data

This case evaluates the performance of each KBNFix network in the scenario where the rate of messages across the network is high while the proportion of unknown data contained in messages is raised to a medium level.

Results of publication matching

Comparing with the previous case, the KBNevery network still outperforms the other networks for publication matching and the extra processing required is well below 50ms (See Figure 7-4). The KBNapproOnly deployment performs slightly worse comparing with its performance in the previous case, it also performs slightly worse than the KBNapproInd network in this case. However, it can be observed (Figure 7-4) that the KBNapproInd also requires more matching time than it spent in the previous case.

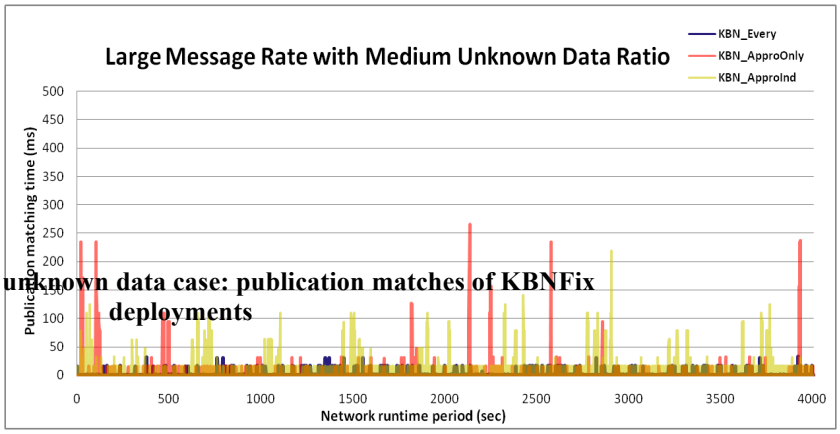


Figure 7-4: Medium unknown data case: publication matches of KBNFix deployments

Analysis of publication matching

It seems that in the KBNevery implementation, all the strategy executions occurred in subscription processing immediately in the routers during network start-up, rather than at the publication matching stage. This leads to no unknown data appearing in the process of publication matching, so that the efficient operation of publication matching is observed. Due to the increased rate of unknown data occurrences in this case, both the KBNapproOnly and KBNapproInd networks require more extra processing to resolve the unknown publications occurring in the process of publication matching.

Results of Publication end-to-end delay

In this case, the KBNapproInd network performs slightly better than the KBNapproOnly network and outperforms KBNevery dramatically (Figure 7-5). It also can be observed that the peaks of end-to-end delay are clearly demonstrated in the KBNevery and KBNapproOnly networks, meanwhile the behavior of the KBNevery network is significantly worse than the previous case.

Analysis of publication end-to-end delay

Compared to the previous case, the reduced performance of the KBNevery network with publication delivery occurs as the increasing unknown data rate leads to more *known* data being delayed by the mapping strategy execution. As for the KBNapproOnly network, the clear peak shapes indicate that as strategy execution time increases this leads to more known data being delayed. Furthermore, those routers whose stored ontologies are not previously merged into the routing ontology will load these due to more unknown data occurrence. Finally, in KBNapproInd, slightly more publications with relatively high delivery time indicate that more unknown data is resolved individually by the routers.

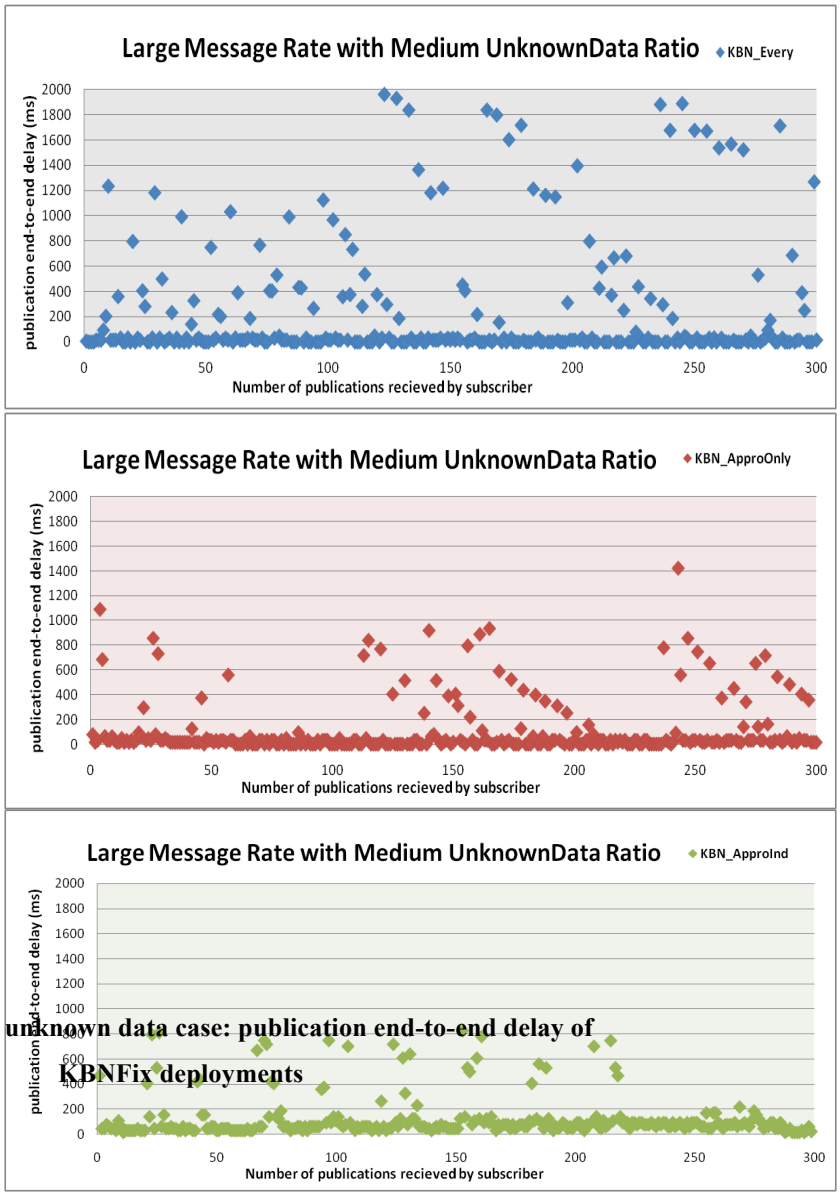


Figure 7-5: Medium unknown data case: publication end-to-end delay of KBNFix deployments

7.8.2.3 Case 3: High Message Rate with Large Unknown Data

This case evaluates the performance of each KBNFix network in the scenario where the rate of messages across the network is high while the proportion of unknown data contained in messages is increased to a high level.

Results of Publication Matching

In this case the KBNevery still outperforms the other networks with respect to publication matching (Figure 7-6). KBNapproOnly performs slightly better than the previous case, in which all expensive overhead is introduced at the relatively earlier stage of network runtime. Finally, the KBNapproInd performs significantly worse than the other networks with respect

to handling high level of unknown data, particularly, the expensive matching times appears more frequently than it did in the previous two cases.

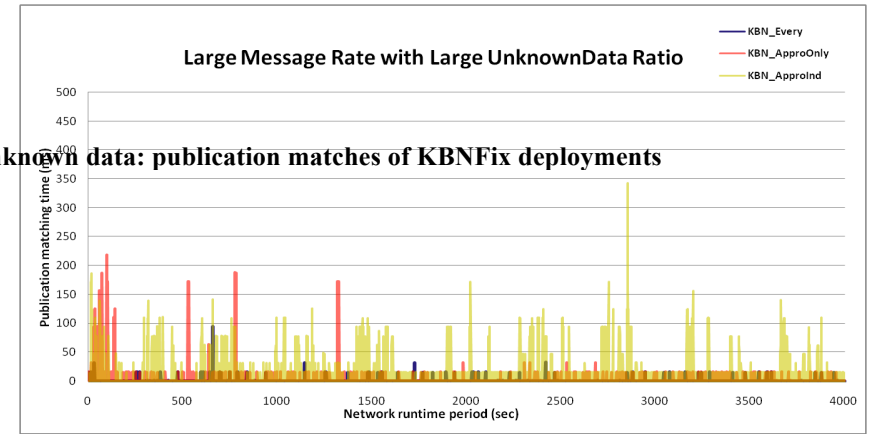


Figure 7-6: Large unknown data: publication matches of KBNFix deployments

Analysis of Publication Matching

In the KBNevery deployment, it seems that all the strategy executions occurs in subscription processing again rather than in the process of publication matching. Interestingly, it can be concluded that routers always execute the **Every** strategy in the process of handling subscriptions in these three test cases (more details of the Every strategy w.r.t subscription processing are demonstrated in **Appendix F.1**). This is due to the experiment setup and the construction of network topology:

1. both the publisher and the subscriber start to forward messages to the network concurrently, which means that the first ***unknown*** publication and subscription are possibly forwarded at the same time;

2. the first ***unknown*** subscription is resolved by the routers prior to the first unknown publication: according to the network topology deployed in this experiment, the first ***unknown*** publication has to be delivered across seven routers (publisher-side routers) to the root router and the routers where the strategy operation occurred, while for the first ***unknown*** subscription, it could be resolved immediately by triggering every strategy once the leaf router has received it.

Results of publication end-to-end delay

In this case, KBNevery and KBNapproOnly still have a considerable number of peaks with respect to end-to-end delay (Figure 7-7). Especially in the KBNevery network, there are a big proportion of publications having high end-to-end delivery times (above 2000ms). It also can be observed that the number of publications that require more delivery time has increased in the KBNapproInd network.

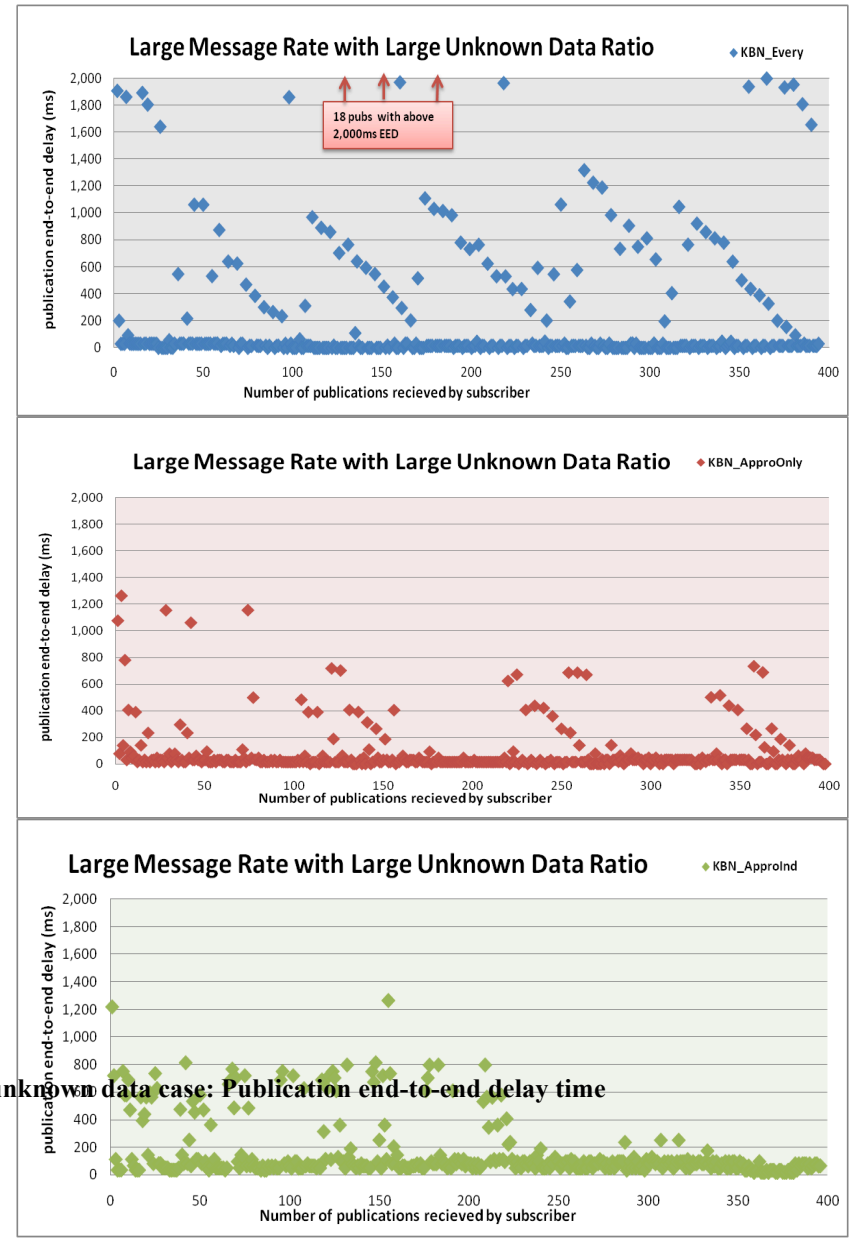


Figure 7-7: Large unknown data case: Publication end-to-end delay time

Analysis of publication end-to-end delay

It is important to note that because of its resource consuming nature, the performance of KBNevery highly depends on the amount of memory resources available in the router. In this experiment, the *small* amount of resources allocated in each router for the purpose of building a limited resource scenario, leads to the KBNevery network performing much worse in the aspect of delivering publications in comparison with the other two networks. In addition, the number of publications requiring more delivery time, suddenly drops off at the middle of runtime in KBNapproInd network. This indicates that all unknown data has been resolved at that point and there is a small portion of known data being delayed by the strategy execution process.

7.8.3 Overall Experiment Findings

The previous cases with different level of unknown data have clearly demonstrated the different impact of mapping strategies on the performance of publication matching and publication delivery in the KBNMap deployment. Figure 7-8 shows the average overhead for each individual strategy execution and the total cost for all the strategy executions across the entire network in the previous three cases.

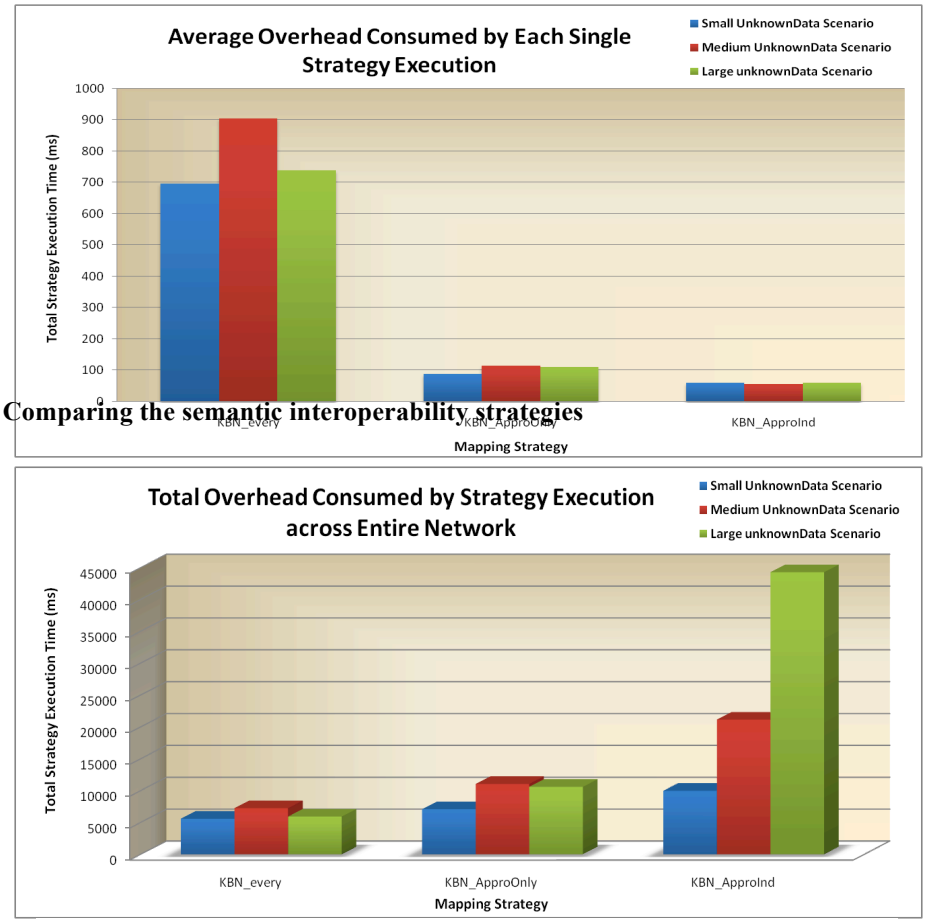


Figure 7-8: Comparing the semantic interoperability strategies

The first graph of Figure 7-8 confirms our observations that the **ApproInd** strategy is efficient for a single execution and is well-suited for the situation where the unknown data is rare, while the **Every** strategy is well suited for the situation where the unknown data occurrence is large due to its high "once-off" cost that enables all potential unknown data becoming known in one operation. Furthermore, considering the combination of characteristics and performance of the **ApproOnly** strategy it is well-suited for the situation where the unknown data occurs at a medium rate.

The second graph in Figure 7-8 shows the sum of all strategy execution times across the entire network over a period of just over 1 hour. The similar overhead introduced by the KBNevery network in each case is due to the fact that the routing ontologies, mapping files,

and referenced ontologies remain the same. It can also be concluded that the KBNevery deployment performs more efficiently than the other deployments, especially in the scenario where the rate of unknown data occurrences remains high, as the Every strategy is only executed once by each individual router. As all unknown data sent by the publisher and subscriber are derived from the stored *referenced* ontologies in this experiment, they all are loaded and merged by the routers into their routing ontology. In other words, no other irrelevant ontologies (unused mappings and ontologies) loaded by the routers. It can be imagined that in a realistic deployment, the router would store multiple diverse mapping files and referenced ontologies, with not every mapping file is useful for a particular unknown data encountered. This would result in extra and wasteful overhead to load the unnecessary mapping files and referenced ontologies in scenarios where there is a small unknown data proportion.

It is clear from the graph that the KBNapproOnly deployment has to execute the **ApproOnly** strategy more frequently in accordance with the generally increased unknown data from case 1 to case 3, resulting in more total extra processing required for the strategy executions. As the **ApproOnly** strategy enables the KBNapproOnly deployment to load and merge the entire appropriate mapping file when unknown data appears, the potential unknown data from the heterogeneous ontology is resolved once and becomes known for the deployment if this ontology's mapping file is loaded. Thus the **ApproOnly** implementation performs more efficiently than the **ApproInd** network in the environment where the rate of unknown data occurrences is high meanwhile the unknown data seemly comes from one source (e.g., an application with its own ontology joins in the network and continuously forwards messages to the network), as the ApproInd deployment has to frequently to execute the ApproInd strategy to resolve the unknown data individually.

Finally, it can be observed from the second graph that, the strategy execution times in the KBNapproInd deployment are also increased along with the increasing rate of unknown data occurrence from case 1 to case 3, since the routers have to execute the ApproInd strategy every time to resolve heterogeneity once they encounter the unknown data. Due to the small individual execution time observed in the first graph, the KBNapproInd deployment should have the smallest overhead for strategy operation in the small unknown data scenario. However, considering the combination of strategy's characteristics (resolving unknown data on an individual basis) and the total amount ***m*** of unknown data originated from unknown data resource, each participating router has to trigger the **ApproInd** strategy ***m*** times. In addition, given the network topology adopted in this experiment, each single unknown data has been sent across the entire network over 15 hops, with result that all routers are involved in executing the strategy to resolve the single unknown data. This results in the KBNapproInd

requiring extra processing than the other networks in case 1. Nevertheless, the overall strategy execution times in the KBNapproInd network highly depend on the average path-length from publisher to subscriber. Given a shorter path under the same unknown data situation, it can be imagined that the total overhead for strategy execution could be reduced dramatically.

In summary, it can be concluded from the experimental results that the **ApproInd** strategy is appropriate for the KBNMap routers to address heterogeneity in the circumstance where the unknown data appears rarely and the average path length between applications for distributing information in the network is small. Otherwise, the overall cost of the **ApproInd** strategy will remains at a high level as Figure 7-8 demonstrated. The KBNMap deployment will benefit from using the **ApproOnly** mapping strategy in the scenarios where the unknown data is constantly derived from one or several heterogeneous ontologies, as in this experiment the **ApproOnly** strategy has been proven to be able to deal with heterogeneity efficiently in this kind of environment. Furthermore, the **Every** strategy is demonstrated to be well-suited for the large KBNMap network where the unknown data is originated from multiple and diverse heterogeneous ontologies, as the **Every** strategy not only brings the "once-off" benefit but also it takes the minimum amount of resources (Figure 7-8) for the deployment to resolve unknown data problem.

However, it must be clear at this stage that the other trigger factors, such as mismatch tolerance of KBNMap applications and memory resources allocation also have a major effect on strategy selection for a specific circumstance/scenario, thereby affirming our assertion that a more intelligent mechanism, such as our Bayesian Belief Network, is required for appropriate strategy selection.

7.9 Experiment 2: The KBNMap Performance

7.9.1 Overview

A Semantic-based Pub/Sub system must process both subscriptions and publications efficiently, with a minimum use of resources, enabling the information to be distributed efficiently to the destinations. In addition, the ontology mappings that allow translation between different semantic models are the vital component in overcoming the heterogeneity between SBPS applications and routers. In KBNMap, the adaptive semantic mapping service provides a mapping mechanism for the KBNMap routers to efficiently and adaptively merge different ontologies/ontological information into their knowledge base (routing ontologies). However, this extra mechanism will inevitably burden the workload of the KBNMap routers. The objective of the experiment is to examine the effect of the adaptive mapping service on

the performance of the KBNMap deployment in the context of distributing heterogeneous information. This experiment is also designed to demonstrate the effect of choosing an incorrect strategy in the **KBNRan** deployment.

The methodology used in this experiment is to evaluate the KBNMap deployment using three different categorized testing sets (Section 7.4.2) where the rate of unknown data occurrence and selected trigger factors are constantly changing. The adaptive KBNMap implementation, incorporating a dynamically updated Bayesian Network was compared to an implementation where the semantic interoperability strategies were selected randomly from the set of 3 strategies (**KBNRan**). The motivation of comparing two types of networks is described in Section 7.3. The performance of KBNMap was evaluated to measure if the extra processing required by the adaptive semantic mapping service can be managed to a level that is acceptable. The publication/subscription matching is regarded as a millisecond process (Carzaniga et. al., 2002; Wang et. al., 2004; Keeney et. al., 2006b), taking into account the inevitable strategy execution time that is explored in Section 7.8, thus the acceptable publication matching time for KBNMap would be less than 1 second in general. On the other hand, as stated in Section 7.7.1, the end-to-end delay time of a specific publication is highly determined by the process of publication matching publication that occurs in the router who received this publication and the number of hops taken for this publication to be sent across to the destination. As the router topology adopted in this experiment is 15 nodes, the acceptable end-to-end delay time would be less than 2 seconds.

7.9.2 Design

In order to evaluate the performance of networks in a large scale environment, the network topology designed in Section 7.6.1 was adopted. The KBNMap deployment composed of developed KBNMap routers was configured. In addition, the **KBNRan** deployment consisting of **KBNRan** routers was configured, so that the performance of KBNMap deployment could be evaluated in comparison with the performance of **KBNRan** deployment

The *ekaw* conference ontology described in Section 7.6.2 was used as the *routing* ontology for both networks and shared among routers while the rest of the other conference ontologies (considered as *referenced* ontologies) and the mapping files providing correspondences between *routing* and *referenced* ontologies are stored in the ontology mapping store of each individual router. It should be noted that, in the router's initial state, the mapping files are not loaded into the routing ontology. In addition, for the purpose of building heterogeneous information distributing environment, the ontological terms in the *unknown* subscriptions and publications are derived from those *referenced* ontologies.

The experiment setup described in Section 7.7 was adopted for setting up this experiment, and the measured metrics (in Section 7.7.1) were collected to evaluate the performance of two networks with respect to *publication matching* and *publication end-to-end delivery*.

The two configured networks were evaluated using the different testing sets described in Section 7.4.2. For convenience, the testing sets are outlined below:

- **Testing Set A:** *Unknown Data Rate with Message Rate***:** this set was to build the testing environment where the message rate and observed unknown data occurrence are varying.
- **Testing Set B:** *Unknown Data Rate Tolerance***:** this set was aimed at building the testing environment where the observed unknown data rate and tolerance level of KBNMap applications are varying.
- **Testing Set C:** *Unknown Data Rate with Memory Resource***:** this set was to build the testing environment where the observed unknown data rate and the resource allocated to the routers are changing.

Each testing set is presented independently in the following subsections. Within each subsection, the specification of parameters used for configuring each individual test is first introduced. It should be noted that for space reasons, the findings and analysis of several representative tests are only given in the subsection, while the results of the other tests can be found in Appendix F.3. Finally, the overall findings of the testing set are given at the end of the subsection.

7.9.3 Testing Set A: Unknown Data Rate with Message Rate

In this testing set the performance of each network was evaluated with respect to different combinations of *the message arrival rate and the unknown data rate*. Three tests of different combinations are outlined as follows:

Unkown data rate (State)		Message Reception Rate (state)		
		Small	***Medium***	***Large***
	Small	Case 1		
	Medium		Case 2	
	Large			Case 3

Case 3 configures the message reception rate and the rate of unknown data occurrence as a *large* level, resulting in the network deployments working in a higher traffic-load

environment than the other two cases. If the KBNMap deployments can work well in this case, it can be observed that it will work better in the other two cases. For reason of space, Case 3 is selected as the representative case in this testing set while the other two cases were also measured and are available in the Appendix F.3.

Table 7-4 summarises the configuration of parameters in detail. For the purpose of evaluating networks in a high load configuration, a large number of *mapping* files and *referenced* ontologies is stored in the ontology mapping store, the message rate ranges from small (5 messages/min) to large (30 messages/min) in different cases, while the unknown data rate also varies from small (2 messages/min) to large (12 messages/min). In addition, the memory resources allocated to each router (40 Mb) and tolerance capability of end applications are specified as small, so that each network works in a resource-limited deployment configuration. The test of each case runs for over 1 hour, once for each implementation.

Table 7-4: Testing set A: Factor configurations to KBNMap and KBNRan deployments

	Variable Name				
Case ID	**Pub Rate (msg /min)**	**Sub Rate (msg/min)**	**Unknown Data Rate (msg/min)**	**Memory Usage (Mb)**	**Mismatch Tolerance**
Case 1	5	5	2	40	small
Case 2	15	15	5	40	small
Case 3 (sample)	30	30	12	40	small

7.9.3.1 Sample Case: Large Unknown Data Rate with Large Message Rate (Case 3)

As a representative case, the scenario where the unknown data rate was high (12 messages /min) and the message rate was high (30 messages /min) is presented. All other trigger factors were set to default high-load configuration (Table 7-4). With the inputs of the selected factors the BN model of KBNMap produced a weighting of 42% for the **Every** strategy, 57% for the **ApproOnly** strategy and 1% for the **ApproInd** strategy. This means that the KBNMap deployment selects the **ApproOnly** strategy to deal with the unknown data.

Despite the memory resources allocated to the networks being set as a small level, it is observed that the **KBNRan** deployment executed the **Every** strategy early on even with the unknown data rate being high, resulting in very high end-to-end publication matching times (Figure 7-9). However, once the mappings were loaded, most of the routers within the **KBNRan** deployment performed well. In contrast, the KBNMap deployment chose the **ApproOnly** strategy, which resulted in lower publication matching times. In addition, at approx. 800 seconds into the experiment, most of the KBNMap routers adaptively switched to

the **ApproInd** strategy. This was due to diminishing memory resources as the routers continuously loaded mappings and merged referenced ontologies, where the **ApproInd** strategy consumes less memory.

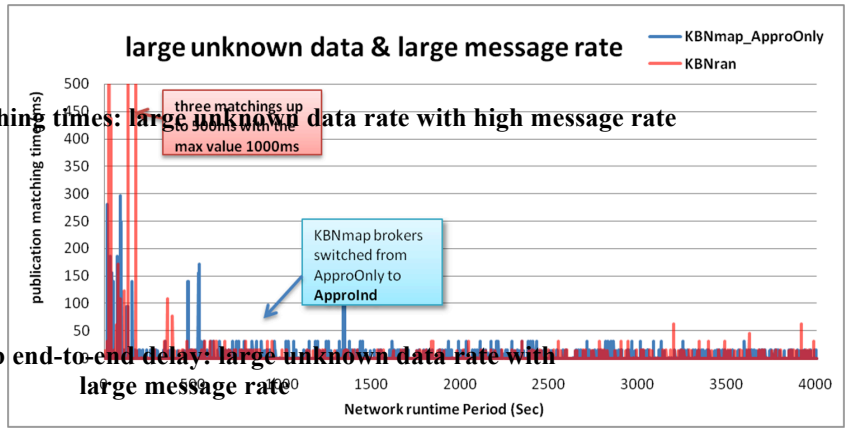
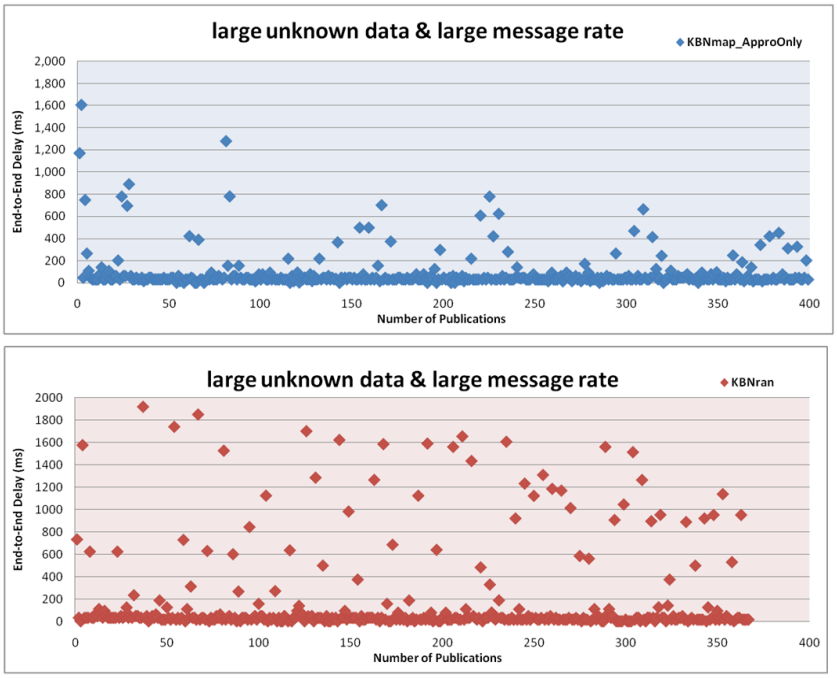


Figure 7-9: Pub matching times: large unknown data rate with high message rate

Figure 7-10: Pub end-to-end delay: large unknown data rate with large message rate

As for the end-to-end delay, Figure 7-10 demonstrated that the subset of the publications matched a subscription were routed across the entire network over 15 hops to the subscriber. Comparing with the previous two cases (Figure F.3-A-2; Figure F.3-A-4), slightly more overhead is introduced by the KBNMap implementation due to the rate of unknown data being increased to a *high* level. However, it still performs much better than the **KBNRan** deployment (Figure 7-10). Moreover as expected, the performance of KBNMap is similar to the KBNFix deployment observed for **KBNapproOnly** in Section 7.4.2.3. As for the **KBNRan** implementation, the results confirm that unpredictable strategy selection leads to more publications with known semantic content being delayed in the process of resolving ***unknown*** data.



7.9.3.2 Overall Summary of Testing Set A

In the situation where the rate of messages arrival and unknown data rate constantly change, the performance of **KBNMap** deployment with respect to publication matching and end-to-end publication delivery was evaluated in comparison with the performance of **KBNRan** deployment. Combining with the results of the other cases in (Appendix F.3.1), the experimental results of the representative case show that the adaptive mapping strategy selection mechanism for KBNMap is clearly superior to the randomly disordered mapping strategy selection mechanisms in this testing set.

By implementing the adaptive semantic mapping service, the **KBNMap** routers selected the appropriate mapping strategy to deal with the unknown data in different cases, with minimum resource consumption. This is because the **KBNMap** routers selected the **ApproInd** strategy for lower rate of unknown data occurrence in first two cases (Figure F.3-A-1, Figure F.3-A-3), whereas the **KBNMap** deployment adaptively used the **ApproOnly** strategy to handle large rate of unknown data in the third case (Figure 7-9). It can be concluded from the three cases that both the publication matching times and end-to-end publication latencies when handling unknown data are significantly reduced in comparison with **KBNRan** deployment. In addition, this also shows randomly selecting a mapping strategy is inappropriate for the **KBNMap** deployment to deal with heterogeneous information.

With respect to the **KBNRan** deployment, the results from the three cases demonstrated that it continuously selected the incorrect mapping strategy in different cases. As an example, the representative case clearly shows that **KBNRan** routers executed the **Every** strategy in the small unknown data scenario, even though as can be seen from the results of experiment 1 (Section 7.8) that the Every strategy is more suitable for large unknown data situations. In addition, using the **Every** strategy allowed the **KBNRan** network to resolve heterogeneity totally at a very initial stage of network runtime in the case scenarios, however, this process seriously diminished memory resources used for distributing publications. Evidence for this can be found in its inefficient performance in the representative case (Figure 7-10) and the other cases (Figure F.3-A-2 and Figure F.3-A-4), particularly in the sense of delivering publications across the entire network of 15 hops.

7.9.4 Testing Set B: Unknown Data Rate with Tolerance

In this testing set the KBNMap and **KBNRan** deployments are evaluated given the combination of two factors: *the mismatch tolerance and the unknown data rate*. Comparing with testing set A, all combinations of the factors were tested, indicating nine different individual case scenarios:

Unkown data rate (State)		Tolerance capability (state)		
		Small	***Medium***	***Large***
	Small	Case 1	Case 2	Cas3
	Medium	Case 4	Case 5	Case 6
	Large	Case 7	Case 8	Case 9

While the sample case 2 in Testing Set A has illustrated the performance of networks in a high traffic-load scenario, it is worthwhile to discover the performance of networks given a low traffic-load scenario, For space reasons, Case 2 describing a scenario where the rate of unknown data occurrences is small is selected for this testing set. However, the results and analysis of the other eight cases of Testing Set B can also be found in Appendix F.3.2.

Table 7-5 presents the factors configuration of this testing set in details. Again all other factors were set to default high-load settings as before: the tolerance was set from low to high level and the unknown data rate ranges from small (2 messages/min) to large (12 messages/min) in different cases. In addition, the message rates are specified as large (30 messages/min), the memory resources was restricted to low (40 Mb), with a large number of *mapping files* and *referenced* ontologies available at each router.

Table 7-5: Testing Set B: factor configurations to KBNMap and KBNRan deployments

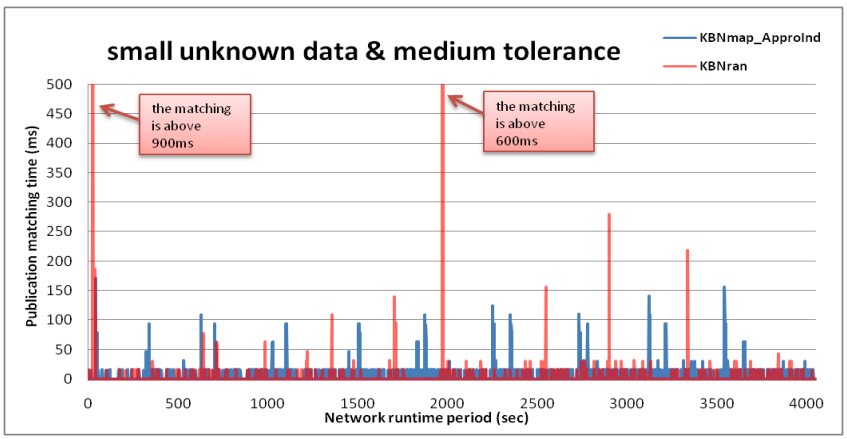
	Variable Name				
Case ID	**Pub Rate (msg /min)**	**Sub Rate (msg /min)**	**Memory Usage (Mb)**	**Unknown Data Rate (msg /min)**	**Mismatch Tolerance**
Case 1	30	30	40	2	Small
Case 2 (sample)	30	30	40	2	Medium
Case 3	30	30	40	2	Large
Case 4	30	30	40	5	Small
Case 5	30	30	40	5	Medium
Case 6	30	30	40	5	Large
Case 7	30	30	40	12	Small
Case 8	30	30	40	12	Medium
Case 9	30	30	40	12	Large

7.9.4.1 Sample Case: Small Unknown Data Rate with Medium Tolerance (Case 2)

As a representative case, this case evaluated the networks in the scenario where the rate of unknown data contained in the messages remains at a relatively small (2 messages /min) level while the mismatch tolerance was set to medium level. All other trigger factors were set to default high-load configuration (Table 7-5). With these inputs the Bayesian Network of KBNMap produced a weighting of 1% for the **Every** strategy, 19% for the **ApproOnly** strategy, and 80% for the **ApproInd** strategy.

Figure 7-11: Pub matching times: small unknown data rate with medium tolerance

Figure 7-11 presents the results for publication matching times. The **KBNRan** implementation incorrectly executed the **Every** strategy early on in several routers, and continued to exhibit poor performance at random intervals during the experiment as the remaining routers also executed the **Every** strategy. As the unknown data rarely occurs in the network, and the mismatch tolerance of applications remains at a higher level, it is wasteful to load and merge a number of unnecessary mapping files into the **KBNRan** routers. By comparison the KBNMap implementation selected the **ApproInd** strategy, which resulted in the strategy being executed more often, but with predictably low overhead for each execution.



As for the performance of distributing publications, it is again clear that KBNMap performs better than **KBNRan**. It is also confirmed from the observations of publication end-to-end delivery (Figure 7-12) that the **KBNRan** network exhibits instability in the aspect of distributing publications, even in the situation where the mismatch tolerance has set to medium level. Since the **KBNRan** routers spend most memory resources allocated in executing the **Every** strategy, there is little resources available for processing the publication delivery, resulting in a large number of publications being delayed to deliver to the subscriber.

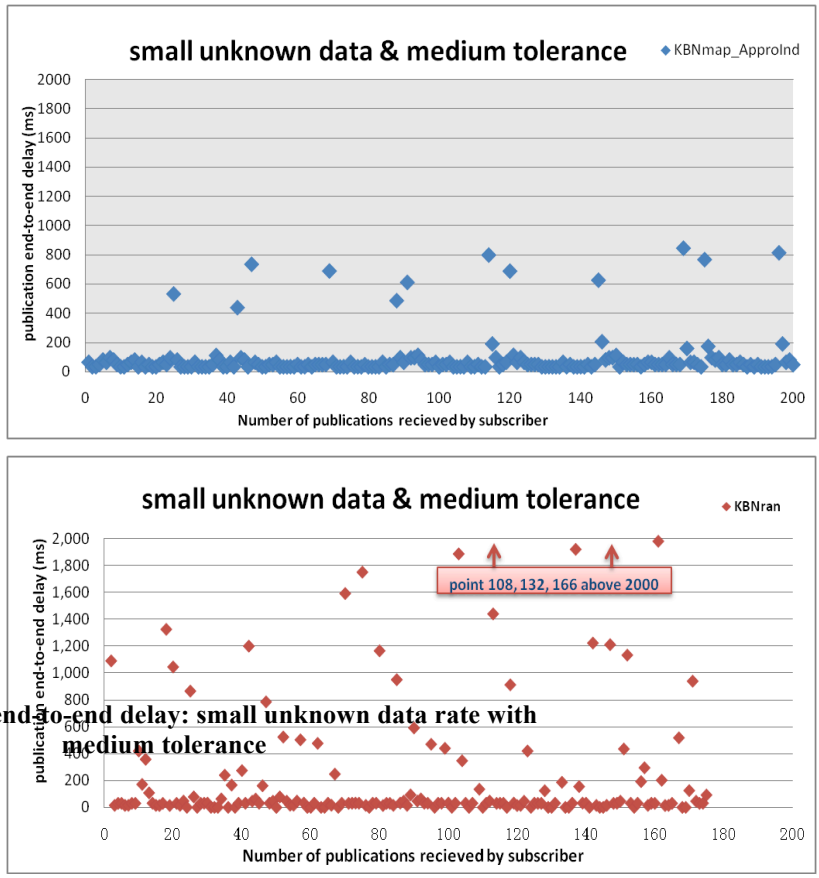


Figure 7-12: Pub end-to-end delay: small unknown data rate with medium tolerance

7.9.4.2 Overall Summary of Testing Set B

In this testing set, the performances of KBNMap and **KBNRan** for publication matching and end-to-end publication delivery were evaluated and compared. Besides the sample case discussed in Section 7.9.4.1, the experimental results of the other cases of this testing set (described in Appendix F.3.2) also clearly demonstrated that in the situation where the rate of unknown data occurrences and mismatch tolerance of applications constantly change, the overall performance of adaptive KBNMap implementation, incorporating a BN model is significantly better than the **KBNRan** deployment who randomly selected the semantic mapping strategies. In addition, it can also be concluded from the experiment results that the KBNMap deployment is able to quickly process publication matches and deliver a large number of publications without consuming large amount of resources. For example, in the case where the unknown data rate is high and tolerance level is medium, the KBNMap deployment used the **ApproOnly** strategy to load and merge the appropriate mapping file more frequently. It can be observed that the network exhibits an efficient performance for both publication matching (Figure F.3-B-13) and end-to-end publication delivery (Figure F.3-B-14) despite the memory resources allocated to the network was set to a small level (40 Mb).

Through the nine cases, the KBNMap deployment exhibits the capability to adaptively choose the appropriate mapping strategy once the combination of the unknown data rate and mismatch rate change. The performance of the KBNMap network using the **ApproInd** strategy in the medium unknown data cases (e.g., case 5: pub matching: Figure F.3-B-7) and high unknown data case (case 9: pub end-to end delay: Figure F.3-B-16) is slightly decreased in comparison with the other cases, but this is due to the small memory resources allocated to the routers. It can be foreseen that allocating more memory resources to the KBNMap routers will help to improve the performance in the future. In addition, the increased memory resources will also induce the Bayesian Network model of KBNMap to select more appropriate mapping strategy in the scenarios where the rate of unknown data is higher. This is evaluated in the next section.

This testing set also shows that it is impractical for a KBNMap deployment to randomly select semantic interoperability strategy without having unpredictable and undesired performance effects. Even though the **KBNRan** implementation gave the opportunities for several routers to use the **Every** or **ApproOnly** strategies in the large unknown data scenario, yielding an efficient performance for matching publications, randomly loading the entire mapping files in the routers in an disordered manner resulted in inefficient performance of the entire network for delivering publications. For example, it could be observed from case 9 (Figure F.3-B-16), there is large number of publications being delayed for distribution in the **KBNRan** network.

7.9.5 Testing Set C: Unknown Data Rate with Memory Resources

As with the previous testing sets, in this testing set we also test the performance of KBNMap and **KBNRan** deployments with respect to different combinations of two trigger factors: *different levels of unknown data and different amount of memory resources*. There are nine different case scenarios evaluated, which are outlined as follows:

Unkown data rate (State)		**Memory Resources (state)**		
		Small	***Medium***	***Large***
	Small	Case 1	Case 2	Cas3
	Medium	Case 4	Case 5	Case 6
	Large	Case 7	Case 8	Case 9

As the sample cases in the previous two testing sets have already presented the performance of the KBNMap deployment which memory resources are allocated as a small (40 Mb) level ,

it is worthwhile to demonstrate the performance of networks in the environment where the memory resources available are set as a medium/large level. Again, for reasons of space, Case 9 in which the memory resources are allocated as a large (120 Mb) level is chosen as the representative case. In addition, the other eight cases were also measured and analysed, which could be found in Appendix F.3.3.

Table 7-6 presents the factors configuration of this testing set in details. Again all other factors were set to default high-load settings as before: the memory resources allocated to the routers ranges from small (40 Mb) to large (120 Mb) as well as the unknown data rate ranges from small (2 messages/min) to large (12 messages/min) in different cases. In addition, the message rates are specified as large (30 messages/min), the mismatch tolerance was restricted to low level, with a large number of *mapping files* and *referenced* ontologies available at each router.

Table 7-6: Testing Set C: factor configurations to KBNMap and KBNRan deployments

	Variable Name				
Case ID	**Pub Rate (msg /min)**	**Sub Rate (msg /min)**	**Mismatch tolerance**	**Unknown Data Rate (msg /min)**	**Memory Usage (Mb)**
Case 1	30	30	Small	2	40
Case 2	30	30	Small	2	80
Case 3	30	30	Small	2	120
Case 4	30	30	Small	5	40
Case 5	30	30	Small	5	80
Case 6	30	30	Small	5	120
Case 7	30	30	Small	12	40
Case 8	30	30	Small	12	80
Case 9 (sample)	30	30	Small	12	120

7.9.5.1 Sample Case: Large Unknown Data Rate with Large Memory Resources Available (Case 9)

In this case the network deployments were evaluated in the situation where the rate of unknown data occurrence was set at a large level (12 messages /min), while the memory resources available in the routers also was raised to a high level (120 Mb). All other factors were set to a default high-load setting as before. With these inputs the Bayesian Network updated a weighting of 62% for the **Every** strategy, 37% for the **ApproOnly** strategy, 1% for the **ApproInd** strategy.

Figure 7-13 demonstrates that the **KBNRan** implementation executed the **Every** strategy early on, introducing highly expensive overhead for the publication matches. And then it exhibited efficient performance during the experiment, as all potential unknown data were resolved after all the mapping files were loaded and merged into the routers. In contrast, at approx. 40 seconds, all the KBNMap routers immediately switched from the **Every** strategy

to the **ApproInd** strategy and exhibited highly efficient performance. The efficient performance also verifies the correct operation of the KBNMap implementation for using the **Every** strategy with sufficient memory resources, which resolved all potential unknown data once by loading all mapping files and merging all referenced ontologies into the routers.

Figure 7-13: Pub matching time: High Unknown Data Rate with High Memory Resources

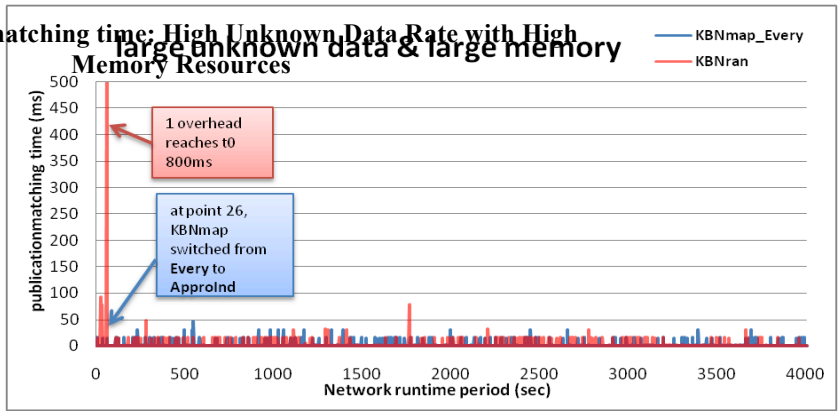
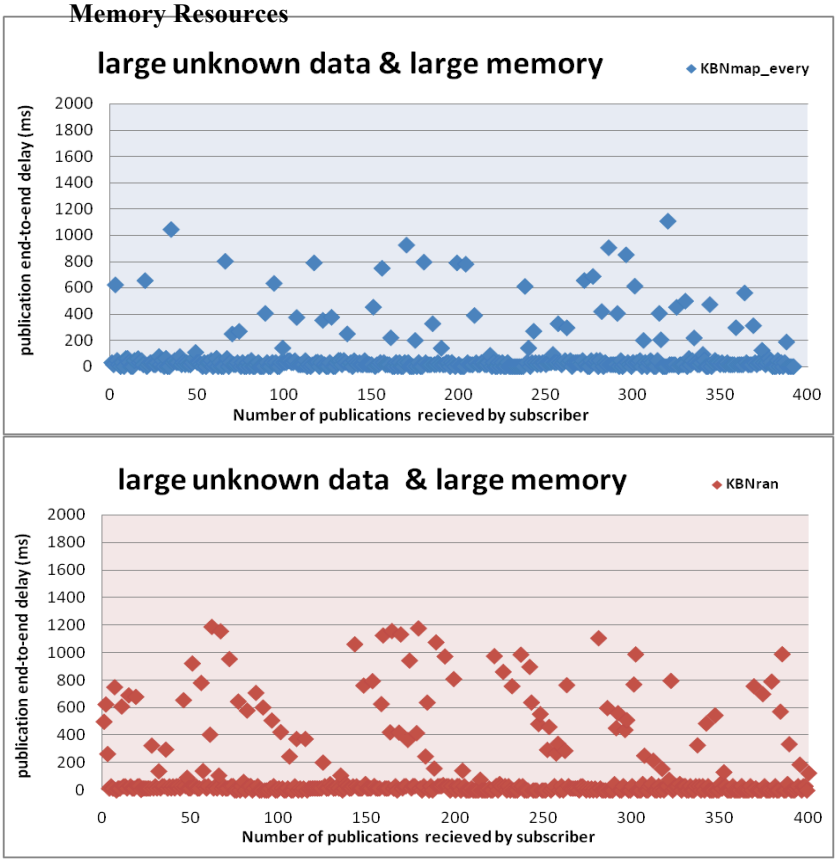


Figure 7-14: Pub end-to-end delay: High Unknown Data Rate with High Memory Resources



With respect to distributing a subset of publications that match a subscription across the entire network to the subscriber, it is demonstrated (Figure 7-14) that the performance of both deployments is improved in comparison with the previous case where the memory resources

were set to a medium level (Figure F.3-C-16), as more sufficient memory resources are allocated to the networks. In addition, the updated weighting of the **Every** strategy indicated that the KBNMap adapted to the changes of the memory resources, which varied from a medium level to a large level.

7.9.5.2 Overall Summary of Testing Set C

This testing set shows that the KBNMap deployment performed better than the **KBNRan** deployment in most tests, particularly in the aspect of distributing a subset of publications that matched a subscription across the entire network over 15 hops to the subscriber. The experiment results also illustrate that the adaptive mapping service could help the KBNMap network to manage the overhead introduced by executing the mapping strategies to a level that is acceptable.

In the *small* unknown data scenarios (case 1 - case 3), which are described in Appendix F.3.3, the KBNMap deployment kept using the **ApproInd** strategy despite the memory resources available were generally updated from a small to a large level, as the small number of unknown data occurrences. However, it can be observed from the Figures of publication end-to-end delay (Figure F.3-C-2, Figure F.3-C-4, Figure F.3-C- 6) that the performance of the network was generally improved in accordance with the increase of the memory resources. Once the rate of unknown data occurrence was set to a *medium* level (case 4 - case 6), the KBNMap implementation exhibited the adaptable and configurable features to tailor to the changes of memory resources. The mechanisms allowed the routers to select the appropriate strategy to resolve the unknown data, with a minimum resource. For example in case 6 where the amount of memory resources available is high, the KBNMap switched to use the **ApproOnly** strategy, resulting in lower publication matching times (Figure F.3-C-11) and smaller proportion of publications requiring longer delivery time (Figure F.3-C-12). Furthermore, comparing with the performance of KBNMap that requires more times for matching (Figure F.3-C-13) and delivering publications (Figure F.3-C-14) in case 7, the sample case combined with case 8 demonstrated that the KBNMap deployment performed efficiently for both publication matching (Figure F.3-C-15) and delivery (Figure F.3-C-16).

Finally, examining the 9 cases, it can be concluded that the increase of the memory resources has significant impact on enhancing the performance of the **KBNRan** deployment. For example, the maximum times required for delivering publications are dramatically reduced from case 1 (Figure F.3-C-2) to case 3 (Figure F.3-C-6) where the memory resource allocation was generally raised to a high level. However, the random mapping strategy selection mechanism still exhibits more inefficient operation of both publication matching and

delivery in comparison with the adaptive mapping strategy selection mechanism, and also makes it is impossible to predict the performance of **KBNRan** network.

7.9.6 Overall Analysis of KBNMap's Performance

The performance experiment shows that the adaptive and self-configuring mapping strategy management mechanisms are clearly superior to the randomly disordered mapping strategy mechanism. In particular the adaptive semantic mapping service could help the KBNMap routers to quickly resolve the unknown data without consuming large amounts of resources in different situations, with both the publication matching times and end-to-end publication latencies when handling unknown data are significantly reduced.

Looking at the results for the publications matching, we see that the time taken to resolve the unknown data depends on the type of mapping strategy executed, as different mapping strategies allow the routers to load different amounts of mappings and referenced ontologies. For example, the routers executing the **ApproInd** strategy only load the appropriate individual mapping each time to resolve a specific unknown data, while the **Every** strategy has to load the entire mapping files and referenced ontologies with a consequential large consumption of resources. Therefore, reasonably and adaptively selecting the appropriate mapping strategy in a specific scenario is a key step to reduce the extra processing required to support ontology mappings.

Analysing the results for the end-to-end delay of distributing a number of publications that matched a subscription across the entire network, it can be concluded that adaptively selecting a mapping strategy did help manage the overhead introduced by distributing an unknown data less than 2 seconds, which is an acceptable level identified in Section 7.8.2. It was observed over all the tests that the maximum end-to-end delay time taken by the KBNMap deployment to deliver an unknown publication to the subscriber was approx. 1600 ms (e.g., case 8 of Testing Set C in Appendix F) and the majority of delay times are well below 200ms. In contrast, in the three testing sets, the **KBNRan** deployment spent more than 1000ms to deliver most unknown publications, in which the maximum time was observed to be over 2000ms for several cases (i.e., Case 1 of Testing Set A; Case 2, Case 4 of Testing Set B, Case 4, Case 7 of Testing Set C in Appendix F).

In summary, the results of three testing sets demonstrated that this experiment was designed sufficiently to examine the effect of mapping strategy selection mechanism on the runtime performance, indicating the first evaluation goal (Section 7.2) in this chapter has been achieved. It has been demonstrated that the extra processing required to execute mapping strategies for addressing heterogeneity is inevitable in the **KBNImpl** deployment which is

integrated with the adaptive semantic mapping service. However, by using the appropriate and adaptive strategy selection mechanism, the **KBNImpl** deployment can still work efficiently for matching publications and distributing heterogeneous information across a large network deployment in a variety of situations where environmental factors are constantly changing. In addition, it should be noted that network topology adopted (Section 7.6.1) in the experiment was a high-load configuration with the longest path (15 hops) between the applications. However, as stated in Section 7.6.1, the other alternative network topologies of 15 routers are also possible in a real-world deployment, but it is likely that the delivery path between the applications are shorter than 15 nodes, and so it can be foreseen that the time taken to distributing the publications would be significantly reduced.

Without a doubt, the scenarios investigated in this experiment cannot cover all possible situations that the KBNMap would encounter in the practical world. Hhowever, as stated in Section 5.3, it is impossible to statically select the mapping strategies with 14 influential factors, it is also impractical to design all testing cases for this experiment given 14 influential factors. In addition, the categorised testing sets designed for the experiment have covered the most important influential factors, under which the networks were evaluated. Furthermore, it should be noted that one issue not directly measured in this experiment is the messages mismatch tolerance of KBNMap applications, as tolerance capability is inherently subjective, so it must be directly specified by higher layers/applications/users. However, the lack of this information would not have a large impact on investigating the effect of the adaptive mapping service, as it is integrated inside the KBNMap routers rather than the higher layer or applications side.

7.10 Experiment 3: the Adaptability of KBNMap

7.10.1 Overview

The second evaluation objective in this chapter was to evaluate the adaptability of the probabilistic-based mapping strategy selection mechanism to changes in ontology, application, and environment-based trigger factors. The adaptive mapping strategy selection mechanism allows the KBNMap routers to self-select the appropriate mapping strategy to cater for different situations, without requiring a human to manually choose a strategy for the routers.

This evaluation part is a complement of the second experiment (Section 7.9), mainly concentrating on the adaptability of the mapping strategy selection mechanism when the KBNMap deployment was tested in different testing sets in the second experiment. Therefore the methodology used in this experiment, the experiment design, and the result data with

respect to the publication matching and end-to-end delay for both KBNMap and **KBNRan** networks are the same as the previous experiment. It only presents a different view on the second experiment performed above. It should be noted that in each individual case of the testing sets, with the inputs of the observed factors (the combinations of two selected factors, and the other factors that were set to high-load settings as demonstrated in Table 7-4, Table 7-5, and Table 7-6). As stated in Chapter 6, once the KBNMap routers start to work, the adaptive mapping strategy selection engine initialize the Bayesian Network model for mapping strategy selection. Given the factors configured above, the Bayesian network model of KBNMap produced a weighting for each mapping strategy. The KBNMap router then chose the strategy with the highest weighting to handle unknown data.

7.10.2 Results

In the following subsections, the summary of weightings of the mapping strategies produced by the Bayesian Network in each testing set is given, and then a systematic analysis of the adaptability based on the summarised results is described.

7.10.2.1 Overall Analysis of KBNMap's Adaptability in Testing Set A

Table 7-7 summarises the weightings of mapping strategies produced by the Bayesian Network in the KBNMap deployment for the different tests. The strategy with the highest weighting in each case (the highlighted cell) caused the KBNMap implementation to choose that strategy for dealing with unknown data.

Table 7-7: Testing Set A: weightings of mapping strategies in KBNMap

Case Name	Strategy Weighting (%)		
	The "Every" Strategy	*The "ApproOnly" Strategy*	*The "ApproInd" Strategy*
1. small unknown data rate with small message rate	2	27	71
2. medium unknown data rate with medium message rate	3	43	54
3. large unknown data rate with large message rate	42	57	1

It is clear from Table 7-7 that the KBNMap deployment consecutively chose the **ApproInd** strategy for dealing with unknown data in the first two cases. However, the increased weightings of the **Every** and **ApproOnly** strategies indicate that the KBNMap implementation reconfigured its mapping strategies to adapt to the changes. The reason that the **ApproInd** strategy was continuously selected is that the *small* memory resources allocated in the KBNMap routers were not sufficient to trigger the **ApproOnly** strategy. Finally in the high unknown data scenario, the KBNMap deployment adaptively used the **ApproOnly** strategy, which allowed the appropriate mapping file to be entirely loaded into

the routers, resulting in the large amount of unknown data to be resolved at one time. The efficient performance of the KBNMap deployment demonstrated in Section 7.9.3 verified that the selection of the mapping strategies by the Bayesian Network was correct.

7.10.2.2 Overall Analysis of KBNMap's Adaptability in Testing Set B

Table 7-8 summarises the weightings of mapping strategies produced by the Bayesian Network in the KBNMap deployment in the situations where the rate of unknown data and the level of messages mismatch tolerance were constantly changing.

Table 7-8: Testing Set B: weightings of mapping strategies in KBNMap

Case Name	Strategy Weighting (%)		
	The "Every" Strategy	*The "ApproOnly" Strategy*	*The "ApproInd" Strategy*
1.small unknown data rate with low tolerance	2	27	71
2.small unknown data rate with medium message tolerance	1	19	80
3.small unknown data rate with large tolerance	1	7	92
4.medium unknown data rate with low tolerance	4	44	52
5.medium unknown data rate with medium message tolerance	3	42	55
6.medium unknown data rate with large tolerance	2	24	75
7.large unknown data rate with low tolerance	42	57	1
8.large unknown data rate with medium message tolerance	32	50	18
9. large unknown data rate with large tolerance	11	44	45

The highlighted cells demonstrate that the adaptive mapping strategy selection mechanism reacted to the changes of unknown data and mismatch tolerance level. For example, the consecutively increasing weighting of the **ApproInd** strategy in *small* unknown data scenarios (case1-case3) indicates that the KBNMap deployment is more preferable to using the **ApproInd** strategy, as the mismatch tolerance level raises from *small* to *high*. While in the large unknown data scenario, the strategy selection mechanism enabled the KBNMap routers to switch to the **ApproOnly** strategy for resolving unknown data, especially when the mismatch tolerance level of applications is lower. In addition the slightly higher weighting of the **ApproInd** strategy in case 9 indicates that the reuse of the **ApproInd** strategy in the KBNMap deployment is only due to the mismatch tolerance being increased to a high level. Thus it empowered the applications to be more tolerant to publications mismatches because of potentially missed semantic relationships due to the overly conservative loading of mappings. Despite the experiment results of KBNMap in case 9 where the unknown data rate is high illustrates that frequently executing the **ApproInd** strategy would introduce expensive overhead for both publication matching (Figure F.3-B-15)and delivery (Figure F.3-B-16). It should be noted that the weightings of BN model for mapping strategy selection is easily

reconfigurable due to the graphical conditional possibility table (CPT) specified to the BN nodes (See Section 5.4). This easier reconfiguration of the CPT helps support finer tuning to the **ApproOnly** strategy for the deployment whose configurations are similar to the Case 9, but highly requires the unknown data to be resolved promptly even though the tolerance of applications is high.

7.10.2.3 Overall Analysis of KBNMap's Adaptability in Testing Set C

Table 7-9 summarises the weightings of mapping strategies produced by the Bayesian Network in the KBNMap deployment in the situations where the rate of unknown data and the level of amount of memory resources allocated to the routers were constantly changing. As can be seen in case 2 and case 3, despite sufficient memory resources being allocated to the KBNMap deployment, the continual selection of the **ApproInd** strategy indicates that the KBNMap implementation recognized that it was not necessary to expend more memory resources in loading the entire mapping file, as the unknown data occurrence still remains relatively rare.

Table 7-9: Testing Set C: weightings of mapping strategies in KBNMap

Case Name	Strategy Weighting (%)		
	The "Every" Strategy	*The "ApproOnly" Strategy*	*The "ApproInd" Strategy*
1. small unknown data rate with low memory resources	2	27	71
2. small unknown data rate with medium memory resources	2	28	70
3. small unknown data rate with large memory resources	3	23	74
4. medium unknown data rate with low memory resources	4	44	52
5. medium unknown data rate with medium memory resources	5	50	45
6. medium unknown data rate with large memory resources	6	60	34
7. large unknown data rate with low memory resources	42	57	1
8. large unknown data rate with medium memory resources	50	49	1
9. large unknown data rate with large memory resources	62	37	1

While in the medium unknown data scenarios, it is clear that the adaptive mapping strategy selection mechanism dynamically switched to the ApproOnly strategy once more memory resources were provided to the KBNMap routers. The more efficient performance (Figure F.3-C-15, Figure F.3-C-16) of KBNMap in case 8 of Testing Set C demonstrates that it was correct to allow the KBNMap deployment to be more confident in using the ApproOnly strategy in case 6. Furthermore, the weightings in case 8 indicates that the strategy selection mechanism sensed that the advantage of using the Every strategy (50) over the ApproOnly strategy (49) was not quite obvious, as it is better to assign more memory resources to the KBNMap routers for loading all the mapping files and merging referenced ontologies. Thus the experiment results in case 8 demonstrates that the KBNMap deployment introduces more

overhead for distributing a subset of publications to the subscriber. By comparison the updated weightings of mapping strategies in case 9 indicates that the KBNMap implementation was more confident in selection of the Every strategy to deal with a large rate of unknown data occurrences, as more sufficient resources were allocated to the network deployment. The experiment results in Section 7.9.5.1 have demonstrated that the performance of the KBNMap deployment was improved in comparison with case 8.

7.10.3 Overall Analysis of KBNMap's Adaptability

The weightings of mapping strategies automatically produced by BN model in the three testing sets above demonstrates that the KBNMap network has the ability to dynamically reconfigure the mapping strategies in order to adapt to the changes of key trigger factors. In addition, the frequency of mapping strategies usage in each case was measured and the results are presented in Appendix F.2 for space reason. The experimental results demonstrated that the KBNMap network correctly executed an appropriate mapping strategy to address heterogeneity, as inferred by the BN model. For example, as the BN model inferred the **ApproInd** strategy (with the highest weighting 71%) as the appropriate strategy for **KBNMap** in case 1 of Testing Set A (Table 7-7), indicating the other mapping strategies should not be triggered by the KBNMap routers. The frequency of each mapping strategy usage in this case (where only the ApproInd strategy is executed 146 times (See Table F-1)) has verified that the **KBNMap** implementation correctly operated the mapping strategy. Furthermore, observing through the cases of three categorised testing sets, the results of the appropriate mapping strategy inferred by the BN Model is in line with the findings of experiment 2 (Section 7.9. This shows that the adaptive mapping strategy selection mechanism operated as expected, as shown by the efficient performance of KBNMap network demonstrated in experiment 2.

As stated in Section 7.9, it is also impossible to fully evaluate the adaptability of KBNMap and the correct operation of mapping strategy selection in the experiment given the combinations of all trigger factors identified in Chapter 5. However, due to the easily reconfigurable nature of Bayesian Network model, it brings a convenient way to flexibly reconfigure the Conditional Possible Tables of BN nodes in order to cater for different requirements on the mapping strategies selection (e.g. different network administrators perspectives and goals).

Furthermore, as the authors stated in (Jensen et. al., 2007; Darwiched 2009), the Bayesian Network model can also be used to automatically predict the possibilities of the observed nodes (the states of trigger factors in the BN model of KBNMap) given the results of the output nodes (the weightings of mapping strategies in the final node). Thus monitoring the

trend of weightings of mapping strategies could become a predictor of mapping strategy changes and thus acts as a threshold or policy for more automatic management of the KBNMap network. For example, from case 9 to case 8 in Table 7-9, the decreased weightings of the **Every** strategy (varied from 62% in case 9 to 52% in case 8) could be used to specify a policy about "allocate more memory resources to the routers in order to ensure efficient information distribution". If a KBNMap router recognises the weighting changes of the **Every** Strategy, it can make use of BN model to infer the changes of "memory resources available" first, then it can send a message about "asking for more resources" to the network administrator or the other routers.

7.11 Conclusion

It is important to note that during all of the simulations, the KBNMap did not encounter a situation in which it blocked evaluating a message (publication/ subscription) for a period long enough to cause a bottleneck at the publication matching and publication delivery. Even during the times in which a large numbers of messages with unknown data was received within one second, the publication matching still operated efficiently and the publications are still distributed to the destinations within 2 seconds. On the other hand, **KBNRan** got blocked occasionally when it dealt with the situation in which the messages arrival rate and the rate of unknown data occurrence are relatively high.

For convenience, the evaluation goals described in Section 7.2 are listed below, in order to help to illustrate how well the three experiments undertaken in this chapter address the evaluation goals:

Evaluation Goal 1: *was to evaluate the effect of the introduced mechanisms for semantic interoperability support on performance of KBNMap routers for matching publications with subscriptions and delivering publications across the network to the subscribers.* The performance of publication matching reflects the scalability of the KBNMap, which is the key feature for the SBPS system (Section 3.3.2.2). Meanwhile it is the task of the KBNMap network to effectively reduce the overall, end-to-end delay of publications. In addition, measuring the publication delivery of KBNMap could also indicate how well the system addresses the first main part of research question (Section 1.2): "distributing heterogeneous information".

To address this goal, the results of the first experiment of *comparing the mapping strategies* described in Section 7.8 have demonstrated that the effects of executing different mapping strategies on the performance of KBNMap are significantly different. Which mapping strategy is appropriate for which scenario in KBNMap network was also investigated by the

first experiment. Therefore, selecting the appropriate mapping strategy to deal with different scenarios was evaluated by undertaking the second experiment "the KBNMap performance", which was presented in Section 7.9. The experimental results have demonstrated that both the publication matching and overhead of handling unknown data are significantly reduced by using the appropriate mapping strategy, which is selected by the *adaptive strategy selection engine* in KBNMap router given three categorised testing sets. In addition, the effect of using a wrong mapping strategy is also evaluated by measuring the random strategy selection method. The inefficient performance of KBNRan deployment illustrated that the importance of selecting the appropriate mapping strategy from another perspective.

Evaluation Goal 2: *was to first investigate the correct operation of executing the appropriate mapping strategy in KBNMap in different tests, and then to assess the applicability of the KBNMap implementation for adaptively selecting mapping strategies, by evaluating the appropriate weightings of mapping strategies produced by the Bayesian Network model given different tests*. This evaluation goal also reflects how well the KBNMap system addresses the second main part of the research question (Section 1.2): "distributing information in an application, and ontology appropriate manner".

The experimental results in the second experiment of Section 7.9 has shown that for the investigated cases the adaptive and self-configuring mapping strategy selection mechanisms are clearly superior to randomly choosing a mapping strategy. The KBNMap routers are demonstrated to be capable of using the appropriate mapping strategy to deal with different cases, resulting more efficient operation of publication matching and delivery than **KBNRan**. The results of the **KBNRan** evaluations show that **KBNRan** is inflexible to the changing environment. The characteristic of randomly selecting mapping strategies in **KBNRan** resulted in the instability of its operation in publication matching and inefficiency of publication delivery. Especially in the perspective of delivering publications, it was demonstrated that it always introduce extremely high cost of processing the publications that are involved in unknown data resolution, regardless of whether the unknown data occurrence is high or not. In addition the results of experiment 3 are evident that KBNMap has the capability to select the appropriate mapping strategy dynamically to adapt to changing contexts in which the ontology, application, environmental factors vary from a state to another state.

The evaluation results of the three experiments have demonstrated that the KBNMap was implemented well enough to address the evaluation goals, and in turn to address the thesis objective that is outlined Section 1.2. Of course, over time the KBNMap implementation could be optimised to operate publication matching more efficiently and deliver the publications to the destinations within a shorter period. However, the better implementation

should only improve the experiment data values while the findings of the experiments should remain similar.

In the thesis, the adaptive semantic mapping service was designed to allow KBNMap routers to use the appropriate ontology mappings only when the unknown data appear in the routers, rather than the other alternative ways such as loading available mappings as soon as the routers are initialized or the routers get the new mappings. It is worthwhile for the KBNMap routers to "wait and see", as the conditions of unknown data they are experiencing and the network environment they are working in are different in various scenarios due to the dynamic nature of networks. If the routers use ontology mappings without the knowledge of states of influential factors identified in Chapter 3, it would potentially waste unnecessary time and consume a considerable amount of unnecessary resources, e.g., if no unknown data arrived in publications or subscriptions. In addition, the experimental results of KBNRan have also shown that inefficient performance for matching and delivering publications in the routers, through executing the wrong mapping strategies in the scenarios.

As a representative system of SBPS systems (See Section 3.4), the KBNImpl is integrated with the adaptive semantic mapping service and was evaluated to be capable of distributing heterogeneous information in an ontology, application and environment appropriate manner. It is envisaged that the adaptive mapping service would also be helpful for other SBPS systems to address the key challenges identified in Section 3.6.

In addition, the designed experiments can also be used to evaluate and compare other SBPS systems that have incorporated an adaptive semantic mapping service. The results of experiments with respect to publication matching and delivery would be different from KBNMap, due to the different SBPS systems being extended, however, the findings arising from the experiments should remain similar.

Finally, while the "**ApproRefer**" mapping strategy was designed but not sufficiently implementable in the current KBNMap prototype, it can be foreseen that the extra processing by the routers for executing this strategy would be similar to the **ApproOnly** Strategy, as both of the mapping strategies are designed to load the appropriate ontology. The main difference is that the **ApproOnly** is used to load the appropriate mapping file while the **ApproInd** is used to load the appropriate reference ontology. In addition, the performance of the other mapping strategies would not be influenced by implementing the new strategy, as they are used to deal with different situations respectively. Furthermore, with the addition of further potential mapping strategies, only the CPTs of final node in the BN model are required to be re-implemented, as the newly added mapping strategies only influence the possibilities distribution of the StrategySelection node rather than the other nodes.

In conclusion, it has been demonstrated that the KBNMap routers are capable of utilizing ontology mappings to resolve heterogeneous information in an efficient manner. . In addition, through making use of Bayesian Network-based mapping strategy selection mechanism, the evaluation has shown that KBNMap can distribute information in an ontology, application, environment appropriate manner, in dynamic networking environments where the variables are constantly changing.

The next chapter concludes this thesis, and further discusses how well the KBNMap design and prototype implementation has addressed the key challenges identified in Section 3.6 and how well the thesis objectives identified in Chapter 1 have been achieved. It then presents the contributions of this thesis.

8 CONCLUSIONS

In the previous chapter, the performance of the KBNMap implementation was evaluated. The applicability and correct operation of KBNMap to the changes of identified factors were verified. This chapter summarises the research undertaken, presents the contributions of this work and outlines how the objectives identified in Chapter 1 have been achieved

8.1 Introduction

This chapter first evaluates the design of KBNMap with respect to the current state of the art of SBPS systems and the criteria for semantic interoperability detailed in Section 3.6To do this, Section 8.2 positions the KBNMap approach with respect to the semantic interoperability taxonomy for SBPS systems (identified in Section 3.3), and with respect to key challenges arising out the state of the art analysis (described in Section 3.6). This chapter then presents how well the objectives were achieved (Section 8.3), summarises the contributions made (Section 8.4), describes some ongoing work and future work (Section 8.5), and finally concludes some final remarks.

8.2 State of the Art Evaluation

Selected state of the art systems were analysed with respect to the taxonomy introduced in Section 3.3 and key challenges identified in Section 3.6. In Table 8-1, the state of the art comparison table from Section 3.5.2 is extended with the addition of a row reflecting how KBNMap addresses the key criteria and key challenges. The classification in Table 8-1 shows that none of the existing SBPSs have the full ability to configure semantic interoperability mechanisms in order to adapt to changes of the influential trigger factors identified.

As can be seen from Table 8-1, only Cashua and Elvin (O) uses OWL mappings to resolve the heterogeneity problem derived by different ontological models. They explore provided mapping files to translate items expressed by one ontology into the related items expressed by another ontology. However, they can only realize mapping at the edge of network rather than inside the router. KBNMap can flexibly use one of several mapping strategies depending on the unknown data occurrence. In addition, both Cashua and Elvin (O) provide a fixed mapping service and cannot reconfigure their semantic interoperability service in accordance with the changes of influential factors, whereas in contrast KBNMap can adapt to changes of ontological, environmental and application based factors.

S-ToPSS has a mapping function to correlate attribute-value pairs to semantically related attribute-value pairs. However, the mapping relations between different items are created at network runtime, which means that the accuracy and efficiency of mappings remains uncertain. Comparing with S-ToPSS, KBNMap uses mapping ontologies that are provided in advance, and has been demonstrated that it is possible to cater for heterogeneous data efficiently.

CQS provides limited semantic interoperability, adding new publications into a router's knowledge base. It also employs a consistency checking algorithm to reduce the overhead of the reasoning task necessary for update of the Knowledge Base. However, a slight change in an ontology will lead to the entire ontology being re-reasoned again.

As for the other systems, no effort is made towards supporting semantic interoperability. In contrast, KBNMap can make use of an adaptive semantic mapping strategy selection mechanism to maximally diminish the negative effect of mapping strategies on the performance of routers, and the results of our evaluations have confirmed that the performance of KBNMap has been managed efficiently to address heterogeneity.

An additional discussion of how well KBNMap meets the criteria and address the challenges is presented in Section 8.3.

Table 8-1: Comparison of KBNMap with State of the Art SBPSs

	Semantic Mapping Service				Adaptive Configuration of mapping service
System	**The way to express mapping:** *OWL, RDF or other typed ontology/language*	**The way to use mapping:** *mapping, merging or translating, or other ways*	**Trigger Place:** *where and when the mappings are used*	**Flexibility:** *have a single or multiple mapping mechanism*	*how the system dynamically adapts to the changes of three influential factors(**semantic model, application, network environment**)*
KBNImpl	No mappings are provided.***note:*** all routers and applications have the same and common ontology	No mappings are provided.	**within the network:** inside router When the router execute event matching	No mapping mechanisms are provided.	Does not have the capability to adapt to the changes of key triggers
S-ToPSS	No concrete ontology language is used. ***note:*** Uses mapping function to correlate different attribute value pairs	Mapping: map different attribute value pairs.	When the router gets new information	No mapping mechanisms are provided	Does not have the capability to adapt to the changes of influential factors
OPS	Does not have mappings ***note***: all routers and applications use a RDF ontology	Limited to **merge** new information into the RDF model but not use mappings	No mappings are provided.	No mapping mechanisms are provided	Does not have the capability to adapt to the changes of influential factors
OBPS	Does not have mappings all routers and clients have the same ontology	No mappings are provided.	**within the network:** inside router When the router meet new pub	No mapping mechanisms are provided	Does not have the capability to adapt to the changes of influential factors
CQS	Does not have mappings ***note***: A single local Knowledge base (KB) in a centralized router	Supports **merging** new publications to into the KB.	**Edge of network:** triggers at generic gateway when the subscriptions are sent into the publisher's side	No mapping mechanisms are provided	Does not have the capability to adapt to the changes of influential factors
Elvin(O)	OWL language Comment: a OWL mapping file for CIM and SMI is provided	Uses OWL mappings to **translate** CMI/SMI into SMI/CMI format	No mappings are provided.	No mapping mechanisms are provided	Does not have the capability to adapt to the changes of influential factors
SPS-P2P	No mappings are provided.***note:*** all routers and applications have the same and common ontology	No mappings are provided.	No mappings are provided.	No mapping mechanisms are provided	Does not have the capability to adapt to the changes of adaptation triggers
Cashua	OWL language Comment: mappings related to different models are expressed in OWL	Uses OWL mappings to translate different context models	**Edge of network:** triggers at CSNs when clients receiving different ontological model.	Automatically distributing mappings	Does not have the capability to adapt to the changes of adaptation triggers
KBNMap	Uses OWL to express mappings	Uses mappings to load multiple diverse ontologies; uses merging to merge new ontologies with routing ontology	Whenever the KBNMap router encounters the unknown subscription or publication, it will use mappings	Having three alternative mapping strategies	Self-selecting the appropriate mapping strategy at runtime to dynamically adapt to changes of influential factors

8.3 Objectives and Achievements

The main goal of this thesis is to answer the question of ***how to enable the management of a Semantic-based publish/subscribe system to distribute heterogeneous information, in an application and ontology appropriate manner***. In order to address this research question, three objectives were derived. A summary of what has been achieved with respect to each objective is provided in italics.

1. The first objective was to identify the key features of Semantic-based Pub/Sub systems, semantic heterogeneous information, and criteria to influence how to handle heterogeneous information in SBPS systems.
    - *A taxonomy was established with respect to the key features for classifying and characterising SBPS systems. A comprehensive literature review was undertaken (Chapter 3), which shows that there is a gap in the state of the art with respect to support for semantic interoperability in SBPS systems. It also identifies a number of factors that influence how SBPS systems should operate in order to deal with heterogeneous semantic information.*
2. The second objective was to present a comprehensive framework to resolve heterogeneous information in SBPS systems in an appropriate manner.
    - *This thesis proposes an adaptive semantic mapping service comprising multiple semantic mapping strategies (Chapter 4) and an adaptive semantic mapping strategy selection mechanism (Chapter 5) as an extension to SBPS systems. As a representative SBPS system, the KBNImpl infrastructure extended with the proposed mapping service (called KBNMap) successfully addresses the key challenges of SBPS systems identified in Section 3.6. The mapping strategies provide multiple choices for the KBNMap router to flexibly load and merge the required mappings/ontologies at runtime, so that the heterogeneous information received can be resolved efficiently. In addition, the KBNMap routers make use of Bayesian Network model developed in Chapter 5 to dynamically select the appropriate mapping strategy to cater for dynamics of networks. Furthermore, due to the extendible nature of the BN model, it is easy to reconfigure the mapping strategy selection mechanism to introduce more candidate mapping strategies and more influence factors. In summary, the KBNMap infrastructure supports flexible and adaptive semantic interoperability. It is extendible and adaptable and provides flexible mapping services. It also supports dynamic configuration of mapping services to adapt to changing states of ontology, application and environment*

conditions. Currently there is no other SBPS infrastructure that meets all of these features. Furthermore, to the author's knowledge, no other proposal provides support for both flexible semantic interoperability and the adaptive configuration of the level of semantic interoperability.

3. The final objective was to implement the framework; evaluate the potential effect of the designed semantic interoperability framework on the performance of SBPS routers; and verify whether the adaptive aspect supported management of the SBPS appropriately.
    - *The first part of this objective was achieved through the successful comprehensive prototype implementation of KBNMap (KBNImpl extended with the both multiple - mapping strategies, and an adaptive strategy selection mechanism). The second part of this objective has been achieved through experimentation that evaluates the performance and verifies the correct operation of the KBNMap infrastructure via the implemented prototype. The results of the KBNMap performance experiments demonstrate that the extra processing required by routers to address semantic heterogeneity can be reconciled effectively using the adaptive semantic mapping service. In addition, the importance of using the appropriate mapping strategy was also demonstrated through examination of the performance of routers that may use a fixed or inappropriate mapping strategy to handle unknown data. Furthermore, the results of KBNMap adaptability tests show that KBNMap routers are able to dynamically select the appropriate mapping strategy to adapt to the dynamics of ontology application, and environment characteristics. In summary, the results derived from the experiments indicate that the KBNMap infrastructure is useful and beneficial.*

The rest of this section describes how well these objectives were achieved in more detail.

Objective 1: Identify the key features for SBPS systems and criteria w.r.t supporting heterogeneity

The state of the art review presented in Chapter 3 details the research that was undertaken in the achievement of the first objective. Given that there is no work identifying the key features of SBPSs, a survey was conducted of existing SBPS approaches, in order to investigate the key features that of a SBPS system. A basic taxonomy with respect to key features was identified.

Both initial and recent literature reviews has revealed some effort in the development of Semantic-based Pub/Sub infrastructures using a single common semantic model for sharing and distributing knowledge, but the area is still maturing. However, very little effort has been expended on the development of a SBPS infrastructure allowing for multiple, diverse

ontologies to coexist among applications, so that heterogeneity between application domains can be supported. In addition, as the SBPS systems would support heterogeneous applications, this survey also explored a number of criteria and identified an advanced taxonomy with respect to semantic interoperability support in SBPS systems.

State of the art SBPS systems were also compared using the identified taxonomies with respect to both key features and semantic interoperability support, which indicated that current SBPS systems elected to focus on either the issue of improving expressivity and scalability, but did not provide a mechanism or solution to adaptively support semantic interoperability so that allow heterogeneity between applications.

Objective 2: Establish a framework for supporting heterogeneity in SBPSs

The second objective was to establish a framework to be used by a SBPS system for resolving heterogeneous information in decentralized networking environments. In response this thesis proposes a comprehensive approach which comprises: support for multiple mapping strategies to enable semantic mappings to be flexibly incorporated into the SBPS router's routing ontology (See Chapter 4); and a probabilistic-based adaptive mapping strategy selection mechanism (see Chapter 5). The degree of success in reaching this objective can be seen through the examination of how the proposed KBNMap infrastructure meets the key challenges outlined in Section 3.6. For each challenge that has been identified, a brief review of how the key challenges have been addressed in KBNMap is presented in the following discussion.

Challenge 1: Support for multiple semantic models

As yet, there has been little previous effort made in the direction of allowing multiple diverse ontologies to co-exist amongst applications and routers in SBPS systems. Through making use of ontology mappings, KBNMap infrastructure has the ability to load and merge the heterogeneous individual mappings or entire heterogeneous ontologies into its knowledge base (routing ontology). In Chapter 4, the integration of an adaptive semantic mapping service into the matching/routing engine of KBNImpl router was discussed, whereby,SBPS routers could efficiently use ontology mappings to address heterogeneity only when the heterogeneous information occurs. This saves time, resources and also avoids loading and merging unnecessary ontologies into the routing ontology.

The main strategies presented to address heterogeneity in the adaptive semantic mapping service were the **Every** strategy, the **ApproOnly** strategy, and the **ApproInd** strategy. The **Every** Strategy was designed to load all available ontologies in the ontology repository into

the router's routing ontology. This strategy maximises the exploration of mappings to tackle the heterogeneity problem and reduces the probability that further unknown concepts will be encountered at a later stage, at the expense of a high once-off cost. The **ApproOnly** Strategy enables the router to search mapping ontologies stored in the router in order to load the appropriate mapping ontology that contains at least one concept used by the conflicting subscription or publication. This strategy is beneficial when the KBNMap router stores a large number of mapping ontologies while the rate of unknown data remains at a moderate level. The **ApproInd** strategy enables the KBN router to efficiently load the appropriate individual mapping. This strategy is beneficial when there is rare unknown data.

Furthermore, in Chapter 4, a number of other candidate mapping strategies were also outlined in order to support the KBNMap routers to deal with the unknown data that cannot be resolved by these three mapping strategies. Similar to KBNImpl, KBNMap uses OWL itself to express ontology mappings. However, the ontology mappings can be expressed in many different knowledge sharing formats. Thus if some ontologies mappings are expressed by other languages, they will be not interpretable to the KBNMap routers and the network administrator. In order to use these mappings, the administrator or router will need to undertake extra effort to transform these ontologies into an appropriate machine interpretable format. However, if one of these formats begins to emerge as the one most commonly used, this transformation work may disappear over time. In addition, as the mapping service is designed to address heterogeneity inside the router, the extra time and resources that are required for the routers to use the mapping service, will inevitably influence the efficiency of SBPS routers operation for matching and routing information. The impact of mapping strategies on the KBNMap router operation was evaluated so that the negative impact of different strategies could be understood and traded-off, and similar evaluations would be required for other SBPS systems which follow this approach.

Challenge 2: Adaptation to Changes

It is argued that different mapping strategies designed in Chapter 4 should be flexibly configured in KBNMap routers to cater for dynamic changes of ontologies, the type of application operating over the KBN as well as the network environment state. Based on the influential factors identified in Chapter 3, 15 ontology, application, environment based trigger factors that have significant impact on reasoning performance of KBNMap were identified in Chapter 5. These identified factors provide a baseline for KBNMap to dynamically evaluate which is the appropriate mapping strategy. For example, the mapping strategy can be selected in accordance with the changes of rate of unknown data occurrences (one application-based trigger factor identified).

However, attempting to select the appropriate strategy based on a static rule-based table (Table 5-7) in Chapter 5, it was clear that it is unrealistic to statically or manually select the appropriate mapping strategy on the basis of 15 identified trigger factors. To address this challenge, a Bayesian Network-based probabilistic approach was employed to build an extendable Bayesian Network (BN) model, in which the identified trigger factors were implemented as observed BN nodes, while the mapping strategies were implemented into the final/output decision node of BN model. Therefore, a KBNMap router, enhanced with the developed BN model, has the capability to dynamically select mapping strategies to cater for dynamic changes of the trigger factors, where the BN model can automatically infer the appropriate mapping strategy given the values of observed trigger factors. In addition, the implementation of the BN model allows for easy extendibility and reconfiguration, which can be easily extended with more strategies, new variables (trigger factors) or be easily reconfigured with more sophisticated strength weightings for BN variables (Conditional probability Table (CPT) for intermediate nodes/final nodes and prior probability for parentless nodes). However, the difficulty with implementing the BN model lay in the design of how the various strengths would be computed to cater for various kinds of tasks and requirements from different KBNMap experts. Currently there are no common standard/specification/algorithm that can be referenced in this regard, and it is likely that KBNMap deployment experience will need to be harnessed in order to tune the implementation.

Objective 3: Implement, evaluate, and verify the SBPS systems extended with the proposed adaptive mapping approach

The first part of the final objective of implementing the SBPS system extended with the adaptive semantic mapping service was achieved through implementing KBNMap as an extension of the KBNImpl system, as described in Chapter 6. KBNImpl was extended with the KBNMap framework of multiple mapping strategies and an adaptive mapping strategy selection engine, making use of the BN model for dynamically selecting the appropriate mapping strategy.

The second part of the this objective of evaluating the effect of introduced semantic interoperability service on the performance of SBPS routers, was achieved through the execution of two experiments described in Chapter 7. The "*comparing the mapping strategies"* experiment demonstrated that the effects of executing different mapping strategies on the performance of KBNMap network are significantly different. In addition the second experiment "*the KBNMap performance*" showed that both the publication matching and overhead of handling unknown data are significantly reduced by using the appropriate mapping strategy selected by the BN-based strategy selection method. In addition, the effect

of using an inappropriately selected mapping strategy was also evaluated by measuring the performance of a special version of KBNMap implementation which is facilitated with a random strategy selection method. The inefficient performance of the KBNRan deployment illustrated that the importance of selecting the appropriate mapping strategy from another perspective.

The third part of the final objective involved verifying the applicability and correct operation of the KBNMap at runtime to dynamically select the appropriate mapping strategy according to the changes of ontology, application and environment characteristics. This was achieved through the combination of experiments 2 and 3 in Chapter 7. The second experiment demonstrated that the KBNMap implementation is capable of using the appropriate mapping strategy to deal with different cases, resulting in more efficient operation of publication matching and delivery than **KBNRan (**that is the KBNMap network facilitated with random strategy selection mechanism). The results of the **KBNRan** evaluations in experiment 2 also showed that the **KBNRan** deployment is inflexible for changing environments. In addition, the reasonable variation in weightings of mapping strategies in the third experiment "the adaptability of KBNMap" make it evident that KBNMap has the capability to adapt to changing contexts in which the ontology, application, environmental factors vary from a state to another state.

8.4 Contributions

1. Design of a multi-strategy mapping approach for SBPS systems to deal with heterogeneous semantic information

The first contribution of the thesis is to present a design of a multi-strategy mapping approach for any SBPS systems to deal with semantic heterogeneity. To our knowledge, such a mapping approach has not been published before. The designed mapping approach allows diverse semantic applications to communicate with each other over a single SBPS network, using their own application semantic models. This is important, as in order to be practical, the application semantic models for different domains will likely be designed differently from the common semantic model of SBPS systems. Evidence of this contribution is given in experiments 1 and 2 in Chapter 7 which demonstrated and evaluated the correct operation and performance of the prototype KBNMap implementation.

The design of the mapping framework supports a number of mapping strategies, which enables multiple semantic models or a scalable semantic model to exist within the SBPS. This support is currently lacking in the state of the art. For SBPS systems relying on a single fixed

semantic model, this approach will help their developers also to build a common scalable semantic interoperability mechanism, so that they can enlarge and enrich their knowledge base at runtime to serve more applications. In particular, this work stresses the important of a flexible multi-strategy approach, rather than advocating a "one size fits all" approach. A multi-strategy service significantly reduces the efforts required in order to support communication between SBPS applications and systems with different domain semantic models, as it provides multiple choices for the SBPS systems to load and merge their required semantic models at runtime without human intervention or shutting down the systems to re-load a new large domain semantic model. In addition, it can be envisaged that this multi-strategy mapping service can also benefit the Semantic Web community in general. For example, heterogeneous semantic web services could be composed by making use of multiple mapping policies in accordance with different requirements of service composition.

Initial descriptions of the multi-strategy semantic mapping framework have been published in:

> S. Guo, J. Keeney, D. O'Sullivan, D. Lewis, "Adaptive Semantic Interoperability Strategies for Knowledge Based Networking", in proceedings of the 3rd International Workshop on Scalable Semantic Web Knowledge Base Systems (SSWS '07) at OTM 2007, Vilamoura, Algarve, Portugal, 27-29 November, 2007.
>
> *This paper presents how the multi-strategy mapping approach facilitates a SBPS system (KBNImpl) to support multiple routing ontologies. In particular, it illustrates that by using ontology mappings, a heterogeneous policy distribution service can be achieved for the dynamic management of radio spectrum.*

2. A comprehensive literature review of existing SBPS systems and a taxonomy of key features for evaluating SBPS systems

The literature review of existing SBPS systems contributes to a taxonomy with respect to key features of SBPS systems. To our knowledge, there is no survey carried out for the state of the art SBPS systems; and also no effort so far in the direction of identifying key features of Semantic-based Publish/Subscribe systems. This literature review provides a guideline for people/organizations who are willing to use a SBPS system to provide a knowledge distribution service for their applications to select the appropriate SBPS system. as the literature review surveyed all existing SBPS systems. In addition, the identified taxonomy not only allows the SBPS experts to evaluate their SBPS systems, but also provides a reference for the designers to design and develop new SBPS systems in accordance with the key features identified. In addition, the literature review and advanced taxonomy presented in this thesis also contributes to a set of criteria for evaluating and comparing SBPS systems with respect to semantic interoperability support. The set of criteria is useful for the SBPS

community, when the experts decide to support multiple semantic models in their SBPS systems which are experiencing heterogeneous information. As the problem of semantic interoperability is not new, This taxonomy/criteria is also beneficial to the non-Pub/Sub semantic middleware. For example, in semantic-query based P2P networks, because the network is decentralised, each peer may have its own peer ontology and interact with other nodes by submitting queries and by replying with relevant knowledge. This taxonomy helps to categorise similar systems, according to the criteria of semantic interoperability.

The literature review of state of the art SBPS systems, and the associated taxonomies, have been published:

> J. Keeney, D. Jones, S. Guo, D. Lewis, D. O'Sullivan, "Knowledge Based Networking", in Handbook of Research on Advanced Distributed Event-Based Systems, Publish-Subscribe and Message Filtering Technologies, [ed.s A. Hinze and A. Buchman], Information Science Reference (IGI Global) ,New York, 2009, in print.
>
> *This publication discusses Knowledge-based networking, which is a subclass of SBPS systems. It also presents a version of KBNImpl. The taxonomy and literature review presented here was used to evaluate and compare KBNImpl with existing semantic-aware Pub/Sub systems.*

3. Identification of key factors that influence provision of a semantic mapping service in a SBPS system

A further contribution is the identification of the ontology, application, and environment-based characteristics that influence support for semantic interoperability in SBPS systems, and in particular their influence on the ontology reasoning performance of SBPS system. This is important, as it has been demonstrated that the appropriate management of semantic interoperability strategies/policies depend on these influential factors. Evidence of this contribution is given in "ontology characterisation" experiment in Chapter 5 and in "DL Reasoner Comparison " experiment in Appendix B which demonstrated and evaluated the significant impact of ontology and application factors on the reasoning performance. It can be envisaged that this study is also beneficial to non Pub/Sub semantic middleware systems. For example, some parameters of semantic web services/applications can be used as the semantic indicators, where these semantic indicators would influence how semantic web service application servers might choose an appropriate mapping strategy to address unknown semantic data passed to semantic web services. However, it should be noted that the influential factors may be different for different systems and paradigms, but the motivation of them is the same. The study of influential factors affecting semantic interoperability support has been published at two conferences:

S. Guo, J. Keeney, D. O'Sullivan, D. Lewis, "Coping with Diverse Semantic Models when Routing Ubiquitous Computing Information", in proceedings of The 5th International IEEE Workshop on Managing Ubiquitous Communications and Services (MUCS2008) at NOMS 2008, Salvador, Bahia, Brazil, 11 April 2008,

In addition to presenting the designed multi-strategy approach extending an SBPS system (KBNImpl) to cope with heterogeneous context models, where KBNImpl is deployed to route context information in the ubiquitous computing environments, this paper also identified and categorised a set of factors that have significant impact on the reasoning performance of SBPS systems.

D. Lewis, J. Keeney, D. O'Sullivan, S. Guo, "Towards a Managed Extensible Control Plane for Knowledge-Based Networking", in proceeding of the 17th IFIP/IEEE International Workshop on Distributed Systems: Operations and Management (DSOM 2006), Dublin, Ireland, October 2006.

While this paper primarily motivates the clustering of KBNImpl routers, the author evaluated the performance issues involved in attempting to perform ontology-based reasoning for knowledge-based routing, and identified the key issue of how loading unnecessary ontological information into the routing knowledge base, and then fully reasoning it, was a major performance factor in KBNImpl and must be avoided where possible, thereby motivating the need to carefully select which ontologies are loaded.

4. Provide an adaptive mapping strategies selection approach, based on the identified influential factors above

Another important contribution of the thesis is to provide an extendable adaptive approach to selecting an appropriate mapping strategy, based on the identified influential factors discussed above. This empowers SBPS systems the adaptability of selecting the appropriate mapping strategy in order to cater for the changes of influential factors. Evidence of this contribution is given in experiments 2 and 3 in Chapter 7 which demonstrated and evaluated the adaptability of the prototype KBNMap implementation. This adaptive selection approach can be used whether the determination of what is a mapping strategy to be undertaken by an expert user, or by the SBPS system itself. The adaptive mapping strategy selection approach is applicable to many different SBPS deployments in many different situations, as it is extendable for other influential factors and other mapping strategies. Again, this probabilistic-based adaptive approach provides a reference for the experts of semantic web community to define their service provision/configuration mechanisms in accordance with the changes of business goals and applications which can be themselves also be considered as influential factors. This work has been recently published at a special issue of Journal of Software (JSW) called *Semantic Extension on Middleware*:

S. Guo, J. Keeney, D. O'Sullivan, D. Lewis, "Adaptive Semantic Interoperability Strategies for Knowledge Based Networking", Journal of Software (JSW, ISSN 1796-217X), Special Issue on Semantic Extensions to Middleware, 2009, accepted.

This paper summaries and categorises the influential factors that have impact on the semantic mapping strategies provision of a SBPS system (KBNImpl). It also presented that combined with the Bayesian Network-based mapping strategy selection mechanism, the KBNImpl routers can adaptively and efficiently provides semantic interoperability services for resolving heterogeneous data in the large-scale network environments.

5. Provide a reference implementation and a benchmark for evaluation.

Finally, a minor contribution is the implementation of KBNMap infrastructure and the construction of the evaluation experiments. The methods used to implement KBNMap can be used as the guideline for the experts/designers to implement their own semantic interoperability service for the SBPS systems. The methodology adopted, metrics used and testing scenarios built in the evaluation provided a benchmark for evaluating other SBPS systems, allowing comparison with the KBNMap system presented in this thesis.

8.5 Future Work

The first step in extending KBNMap is to take more applications in large-scale network computing environments running over the KBNMap in order to expand experience gained with the practicality of the system and its performance with respect to distributing heterogeneous information. Using more diverse applications with different semantic models will allow KBNMap to be improved in such a way. More diverse applications will have more different and complex semantic models, indicating more unknown data will be encountered by the KBNMap routers. These larger unknown data will exercise the functionality of KBNMap routers more fully, allowing insight into any difficulties they might experience. However KBNMap was designed to be application independent so it is envisioned that testing the approach with more diverse applications should further validate the approach.

Allowing for more efficient usage of semantic mappings, it is envisaged that the SBPS system could itself be used for distributing semantic mappings among KBNMap enabled routers. Work is ongoing to investigate how mappings can be dynamically distributed around the network as the knowledge bases of clients joining and leaving the network affect the spread of knowledge across the network.

Ongoing work is also focussing on how the semantics of the knowledge in the network can be exploited to cluster nodes that focus on semantically similar knowledge (Lewis et. al., 2006b).

In this way the interoperability of clusters' knowledge bases can be localised to edge of clusters, thereby localising the overhead required for semantic interoperability and mapping.

KBNMap's implementation may also be rewritten to be more processor and memory efficient, providing greater performance and allowing KBNMap routers to run on more resource constrained devices. Other SBPS systems integrated with the designed mapping services may also be investigated, in order to assess changes in performance that this would bring.

With respect to evaluation, the general setup (e.g., ontologies used, network topology) and the testing sets identified for this experiment were sufficient to evaluate the performance of KBNMap for the purposes of this thesis and so validate the approach. However if more time and resources are available, it would be beneficial to extend the experiments to evaluate the KBNMap implementation in further deployments (e.g., more diverse, complex ontologies, more diverse network topology) and more applicable testing sets (such as providing the scenarios which take into account combining more application-specific trigger factors). These larger experiments could provide more information about how to streamline and manage KBNMap to more efficiently distribute information in an efficient and adaptive manner.

8.6 Final Remarks

KBNMap's integration of Knowledge-based Networking and an adaptive semantic mapping service provides support for some of the challenges posed by managing Semantic-based Pub/Sub systems to deliver heterogeneous information to applications in an ontology, application, environment appropriate manner. While Semantic-based Publish/Subscribe paradigm has not yet come to full fruition, there is increasingly a need for such platforms. The semantic models used for communication in the distributed networking environments will be heterogeneous, this requires SBPS systems to be put in place to allow multiple semantic models to coexist, to support communication between applications which using different application models, enhancing the knowledge distribution service efficiently. Given the contributions in the thesis, the author believes that the adaptive, multi-strategy mapping approach can achieve the coexistence of multiple semantic models in SBPS systems, and enable the SBPS routers to adaptively and effectively distribute heterogeneous information.

BIBLIOGRAPHY

(Agoulmine et. al., 2006) Agoulmine, N., Balasubramaniam, S., Botvich, D., Strassner, J., Lehtihet, E., Donnelly, W. (2006), "Challenges for Autonomic Network Management", In Proceedings of 1st IEEE Workshop on Modelling Autonomic Communication Environments (MACE06), Dublin, Ireland

(Akyidiz et. al., 2006) Akyildiz, I.F., Lee, W., Vuran, M.C., Mohanty, S., (2006), "Next Generation/Dynamic Spectrum Access/Cognitive Radio Wireless Networks: a Survey", In the International Journal of Computer and Telecommunications Networking, Issue 13, 2127-2159.

(Antoniou et. al., 2004) Antoniou, G. Van Harmelen, F., (2004), "A Semantic Web Primer": MIT Press, 2004.

(Arabi et. al., 2007) Arabi, M., Govindaraju, R. S., Hantush, M. M., (2007), "A Probabilistic Approach for Analysis of Uncertainty in the Evaluation of Watershed Management Practices", In Journal of Hydrology, 333(2-4), 459-471.

(Bates et., al., 1998) Bates, J., Bacon, J., Moody, K., Spiteri, M., (1998), "Using Events for the Scalable Federation of Heterogeneous Components," In Proceedings of the 8th ACM SIGOPS European workshop on Support for composing distributed applications, pp. 58-65.

(Bechhofer et. al., 2003) Bechhofer, S., Lord, P., Volz, R., (2003), "Cooking the Semantic Web with the OWL API," In Proceedings of ISWC2003, Sanibel Island, Florida, October 2003.

(Belecheanu et. al., 2004) Belecheanu, R., Jawaheer, G., Hoskins, A., McCann, J.A., Payne, T., (2004), "Semantic web meetings autonomic ubicomp" In proc. Workshop on Semantic Web Technology for Mobile and Ubiquitous Applications, Hiroshima, Japan, 2004

(Berners-Lee et. al., 2001) Berners-Lee, T., Hendler, J., Lassila, O., (2001), "The Semantic Web", Scientific American, May 2001.

(Bowers et. al., 2000) Bowers S. and Delacambre L., (2000), "Representing and Transforming Model-based Information", In Proceedings of Workshop on the Semantic Web at the Fourth European Conference on Digital Libraries, Portugal, 2000.

(Brasethvik et. al., 2001) Brasethvik, T., Atle, J., Gulla, (2001), "Natural Language Analysis for Semantic Document Modeling", In Journal of Data & Knowledge Engineering, vol. 38, pp. 45-62, 2001.

(Bronstein et. al., 2001) Bronstein, A., Das, J., Duro, M., Friedrich, R., Kleyner, G., Muller, M., Singhal, S., Cohen, I., "Self-aware Services: using Bayesian Networks for Detecting Anomalies in Internet-based Services", Technical Report, HP Labs Technical Reports HPL-2001-23 (R.1), 2001

(Burcea et. al., 2003) Burcea, I., Petrovic, M., Jacobsen, H-A., (2003), "I know what you mean: Semantic Issues in Internet-scale Publish/Subscribe Systems," In

Proceedings of the International Workshop on Semantic Web and Databases (SWDB03), Berlin, Germany, 2003.

(Carroll et. al., 2004) Carroll, J., Dickinson, I., Dollin, C., "Jena: Implementing the Semantic Web Recommendations", in Proceedings of the 13th International World Wide Web Conference, (WWW 2004), New York, 17-22 May, 2004. http://jena.sourceforge.net/

(Carzaniga 2008) Carzaniga, A., "Siena - Software", (2008), Available at: http://www.inf.unisi.ch/carzaniga/siena/software/index.html, Last Visit: September 2008.

(Carzaniga et. al., 1999) Carzaniga, A., Rosenblum, D., Wolf, A.L., (1999), "Challenges for Distributed Event Services: Scalability vs. Expressiveness", In Proceedings of Engineering Distributed Objects (EDO'99), ICSE 99 Workshop, Los Angeles CA. May 1999.

(Carzaniga et. al., 2000) Carzaniga, A., Rosenblum, D., Wolf, A.L., (2000), "Achieving Scalability and Expressiveness in an Internet-Scale Event Notification Service", In Proceedings of the Nineteenth Annual ACM Symposium on Principles of Distributed Computing, Portland, Oregon, United States, 2000.

(Carzaniga et. al., 2001) Carzaniga, A., Rosenblum, D.S., and Wolf, A.L., (2001), "Design and Evaluation of a Wide-Area Event Notification Service", ACM Transactions on Computer Systems, 2001.

(Carzaniga et. al., 2002) Carzaniga, A., Wolf, A.L, (2002), "A Benchmark Suite for Distributed Publish/Subscribe Systems", Technical Report CU-CS-927-02, Department of Computer Science, University of Colorado, April, 2002.

(Castano et. al., 2005) Castano, S., Ferrara, A., Montanelli, S., (2005), "Ontology-based Interoperability Services for Semantic Collaboration in Open Networked Systems", In Proceedings of the First International Conference on Interoperability of Enterprise Software and Applications (INTEROP-ESA), pp. 135-146., 2005.

(Castillo et. al., 1997) Castillo, E., Gutierrez, J.M., Hadi, A.S., (1997), "Sensitivity Analysis in Discrete Bayesian Networks", IEEE Transactions on Systems, Man and Cybernetics, 27(4), 412–423, 1997

(Chakraborty et. al., 2005) Chakraborty, D, Joshi, A., Finin, T., Yesha, Y., (2005), "Service Composition for Mobile Environments", Journal of Mobile Networks and Applications, 10(4), 435-451, 2005

(Chalupsky et. al., 2000) Chalupsky, H., (2000), "OntoMorph: A Translation System for Symbolic Knowledge," Principles of Knowledge Representation and Reasoning, vol. 185, 2000.

(Chan et. al., 2006) Chan, C., Ni, Y., (2006), "Content-based Dissemination of Fragmented XML Data", In Proceedings of International Conference on Distributed Computing Systems, ICDCS 2006

(Chirita et. al., 2006) Chirita, P.A., Idreos, S., Koubarakis, M., Nejdl, W., (2006), "Semantic Web and Peer-to-Peer: Decentralized Management and Exchange of Knowledge

and Information," In Proceedings of Semantic Web and Peer-to-Peer, Berlin Heidelberg, 2006.

(Choi et. al., 2006) Choi, N., Song, I., Han. N., "A Survey on Ontology Mapping", In Proceedings of SIGMOD Rec., 35(3):34–41, 2006.

(Cian 2001) Cian, J., (2001), "Planning Improvements in Natural Resources Management: Guidelines for Using Bayesian Networks to Support the Planning and Management of Development Programs in the Water Sector and Beyond", Center for Ecology and Hydrology, Wallingford, Oxon., UK. 2001.

(CIM v2.21) Common Information Model, Available at http://www.dmtf.org/standards/cim/, Last visited: September 2007.

(Cross 2003) Cross, V., (2003), "Uncertainty in the Automation of Ontology Matching", In Proceedings of 4th International Conference on Uncertainty Modelling and Analysis, ISUMA '03, IEEE Computer Society, 2003

(Cugola et. al., 2001) Cugola, G., Nitto, E.D., Fuggetta, A., (2001),"The JEDI Event-Based Infrastructure and Its Application to the Development of the OPSS WFMS," 2001

(Cui et. al., 1999) Cui, Z., Tamma, V. A. M., Bellifemine, F. (1999), "Ontology Management in Enterprises", BT Technology Journal, 17(4), 98-107

(DAML 2005) DAML Ontology Library, Available at: http://www.daml.org/ontologies/, Last visited: October 2005.

(DAML Model 2001) "A Model-Theoretic Semantics for DAML+OIL", Available at: http://www.w3.org/TR/daml+oil-model, Last visited: October 2005.

(Darwiched 2009) Darwiche, A., (2009), "Modeling and Reasoning with Bayesian Networks", Cambridge University Press, ISBN-13:9780521884389.

(Decker et. al., 2000) Decker, S., Melnik, S., Van Harmelen, F., Fensel, D., Klein, M., Broekstra, J., Erdmann, M., Horrocks, I., (2000), "The Semantic Web: The Roles of XML and RDF," 2000.

(Deng et. al., 2007) Deng, Y.H., Wang, F., (2007), "Opportunities and Challenges of Storage Grid Enabled by Grid Service", ACM SIGOPS Operating System Review, 41(4), 79-82, 2007

(Diao et. al., 2003) Diao, Y., Altinel, M., Franklin, M.J., Zhang, H., Fischer, P., (2003), "Path Sharing and Predicate Evaluation for High-Performance Xml Filtering", ACM Transactions on Database Systems (TODS), 28(4):467–516, 2003

(Ding 2004) Ding, L (2006), "Web of Belief Ontology", Available at http://daml.umbc.edu/ontologies/webofbelief/0.8/wob.owl, Last visited: June 2006

(Doan et. al., 2002) Doan, A., Madhavan, J., Domingos, P., Halevy, A., (2002), "Learning to Map between Ontologies on the Semantic Web", In Proceedings of the 11th international conference on World Wide Web 2002 (WWW02), pages 662–673, New York, NY, USA, 2002.

(Dong et. al., 1997) Dong, A., Agogino, A.M., (1997), “Text analysis for constructing design representations”, Journal of Artificial Intelligence, Eng. 11, 65–75, 1997

(Dou 2004) Dou, D., (2004), “Ontology Translation by Ontology Merging and Automated Reasoning”, PhD thesis, Yale University, New Haven, CT, USA, 2004.

(Dou et. al., 2003) Dou, D., McDermott, D., Qi, P., (2003), "Ontology Translation on the Semantic Web," In Proceedings of International Conference on Ontologies, Databases and Applications of Semantics, pp. 952-969.

(Eclipse 3.2) Eclipse IDE for Java Developer, available at: http://www.eclipse.org/downloads/, Last visit: September 2008.

(Ehrig et. al., 2004) Ehrig, M., Staab, S., (2004), “QOM - Quick Ontology Mapping”, In Proceedings of International Semantic Web Conference (ISWC 2004), LCNS3298, pages 683-697, Springer, Berlin, Germany, 2004.

(Eugster et. al., 2001) Eugster, P., Guerraout, R., (2001), “Content-based Publish/Subscribe with Structural Reflection”, In Proceedings of the 6th Usenix Conference on Object-oriented Technologies and Systems (COOTS’01), 2001

(Eugster et. al., 2003) Eugster, P., Felber, A., Guerraout, R., Kermarre, A.-M., (2003), “The Many Faces of Publish/Subscribe”. Journal of ACM Computing Surveys 35(2) 114-131, 2003

(Euzenat 2001) Euzenat, J., (2001), “Towards a Principled Approach to Semantic Interoperability”, In Proceedings of workshops on ontologies and information sharing, IJCAI01, Seattle, USA, 2001

(Euzenat 2004) Euzenat, J., (2004), “An API for ontology alignment”, In Proceedings of 3rd International Semantic Web Conference (ISWC2004), Hiroshima, Japan, 7-11 November, 2004

(Exchanger 3.2) A XML Editor, Available at: http://www.freexmleditor.com/, Last visit: June, 2008.

(eXist 1.2.5) eXist: Open Source Native XML Database, Available at: http://exist.sourceforge.net/download.html#N1028B, Last Visit: September, 2008.

(Fensel 2000) Fensel, D., (2000), “The semantic web and its languages”, Journal of IEEE Computer Society 15, 6 (November/December), 67−73, 2000

(Fensel 2003) Fensel, D., (2003). “Spinning the Semantic Web: Bringing the World Wide Web to Its Full Potential”, MIT Press, 2003.

(Fensel et. al., 2000) Fensel, D., Harmelen, F. V., Klein, M., Akkermans, H., Broekstra, J., Fluit, C., van der Meer, J., Schnurr, H. P., Studer, R., Hughes, J., (2000), "On-To-Knowledge: Ontology-based Tools for Knowledge Management," In Proceedings of the eBusiness and eWork, 2000

(Fensel et. al., 2000b) Fensel, D., Crubezy, M., Harmelen, F. V., Horrocks, I., (2000), "Oil & Upml: A Unifying Framework for the Knowledge Web", In Proceedings of ECAI 2000, Berlin, Germany, 2000.

(Fonseca et. al., 2002) Fonseca, F., Egenhofer, M., Agouris, P., Camara, G., (2002), "Using Ontologies for Integrated Geographic Information Systems", Transactions in GIS, − (6):3 in print, 2002

(Galen) "The Galen Ontology", Available at http://www.cs.man.ac.uk/~horrocks/OWL/Ontologies/galen.owl, Last visited: June 2006

(Garbolino et. al., 2002) Garbolino, P., Taroni, F., (2002), "Evaluation of Scientific Evidence Using Bayesian Networks", Forensic Sci, Int. 125, 149–155, 2002

(Gardoso 2007) Gardoso, J., "The Semantic Web Vision: Where are We? ", Journal of IEEE Intelligent System, Sept-Oct 2007, pp.22-26, 2007.

(Giunchiglia et. al., 2003) Giunchiglia, F., Shvaiko P., (2003), "Semantic Matching", In Proceedings of IJCAI 2003 Workshop on Ontologies and Distributed Systems, pages 193-146, Acapulco, Mexico, 2003.

(Giunchiglia et. al., 2004) Giunchiglia F., Shvaiko P., Yatskevich M., (2004), "S-Match: an Algorithm and an Implementation of Semantic Matching", In Proceedings of ESWS, pages 61-75, 2004.

(Goksel et. al., 1999) Goksel, A., Dennis, M., (1999), "Semantic Heterogeneity Resolution in Federated Databases by Metadata Implantation and Stepwise Evolution", Journal of Very Large Databases, 8(2), 120-132, 1999

(Gonzalez et. al., 2007) Gonzalez Prieto, A., Stadler, R., (2007), "A-GAP: An Adaptive Protocol for Continuous Network Monitoring with Accuracy Objective", IEEE TNSM, vol. 4, no. 1, June 2007.

(Gruber 1993) Gruber, T. R., (1993), "A Translation Approach to Portable Ontology Specifications," Knowledge Acquisition, vol. 5, pp. 199-220. 1993

(Gryphon 2007) Gryphon IBM Research (1997-2007). Available at: http://www.research.ibm.com/distributedmessaging/gryphon.html, Last Visit: July 2007

(Guarino et. al., 1999) Guarino, N., Masolo, C. ., Vetere, G., (1999)."OntoSeek: Content-Based Access to the Web," 1999.

(Guo 2004) Guo, Y., Heflin, J., Pan, Z., (2004), "An Evaluation of Knowledge Base Systems for Large OWL Datasets", Technical Report, CSE Dept., Leigh University, 2004

(Guo 2005) Guo, Y., Heflin, J., (2005), "LUBM: A Benchmark for OWL Knowledge Base Systems", Journal of Web Semantics, Volume 3, Issue 2, 2005.

(Guo et. al., 2008) Guo, S., Keeney, J., O'Sullivan, D., Lewis, D., (2008), "Coping with Diverse Semantic Models when Routing Ubiquitous Computing Information", The Workshop on Managing Ubiquitous Communications and Services (MUCS2008) at NOMS 2008, Salvador, Bahia, Brazil, 7-11 April 2008.

(Gupta et. al., 2003) Gupta A., Suciu. D., (2003), "Stream Processing of Xpath Queries with Predicates". In Proceedings of 2003 ACM SIGMOD International Conference on Management of data pages 419–430, 2003.

(Haarslev et. al., 2001) Haarslev, V., Moller, R. (2001), “RACER System Description”, In Proceedings of IJCAR 2001, volume 2083 of LNAI, 701–706, Siena, Italy, 2001

(Hakimpour et. al., 2001) Hakimpour, F., Geooert, A., (2001), “Resolving Semantic Heterogeneity in Schema Integration”, In Proceedings of the international conference on Formal Ontology in Information Systems, Ogunquit, Maine, USA, 2001

(Halaschek-Wiener et. al., 2006) Halaschek-Wiener, C., Parsia, B., Sirin, E., (2006), “Description Logic Reasoning with Syntactic Updates”, In Proceedings of International Conference on Ontologies, Databases, and Applications of Semantics, 2006

(Halaschek-Wiener et. al., 2007) Halaschek-Wiener, C., Hendler, J., (2007), "Toward Expressive Syndication on the Web," In Proceedings of the 16th international conference on World Wide Web, Banff, Alberta, Canada: ACM, 2007

(Harrison et. al., 1997) Harrison, T.H., Levine, D.L., Schmidl, D.C., (1997), “The Design and Performance of a Real-time CORBA Event Service”, in Proceedings of the ACM SIGPLAN Conference on Object-Oriented Programming Systems, Language and Applications (OOPSLA-97), pages 184-200. ACM press/

(Heflin et. al., 2000) “Semantic Interoperability on the Web”

(Heimbigner 2001) Heimbigner, D., (2001), "Adapting Publish/Subscribe Middleware to Achieve Gnutella-like Functionality," Proceedings of the 2001 ACM symposium on Applied Computing, pp. 176-181, 2001

(Hendler 2001) Hendler, J., (2001), “Agents and the Semantic Web”, Journal of IEEE Intelligent Systems, Vol.5, no.2, pages 199-220, 2001

(Horrocks et. al., 2002) Horrocks, I., Patel-Schneider, P.F., Van-Harmelen, F., (2002), “Reviewing the Design of Daml+Oil: an Ontology Language for the Semantic Web”, In Proceedings of AAAI20002, pages 792-797, 2002.

(Howard et. al., 2005) Howard, R. A., Matheson, J. E. (2005), “Influence diagrams”, Journal of Decision Analysis, Volume 2, Issue 3:127-143, 2005.

(Interoperability 2006) “Achieving Semantic Interoperability and Integration Using RDF and OWL”, January 2006, Available at http://cmenzel.org/w3c/SemanticInterop.html , last visited: December 2008.

(JavaBayes 2001) Bayesian Networks in Java, available at http://www.cs.cmu.edu/~javabayes/Home/index.html, Last visit: January, 2009

(Jena 2005) “Jena - A Semantic Web Framework for Java”, Available at http://jena.sourceforge.net/, Last visited: August 2008.

(Jensen et. al., 2007) Jensen, F.V., Nielsen, T.D., (2007), “Bayesian Networks and Decision Graphs”, Springer-Verlag, New York, ISBN: 0-387-68281-3, 2007

(JMS 1.1) “Java Message Service”, Available at: http://java.sun.com/products/jms/, Last visit: April, 2008

(Kahn et al., 1997) Kahn Jr., C.E., Roberts, L.M., Shaffer, K.A., Haddawy, P., (1997), "Construction of a Bayesian Network for Mammographic Diagnosis of Breast Cancer", Computers Biol. Med. 27 (1), 19–29, 1997

(Kaiser et. al., 1999) Kaiser, J., Mock, M., (1999), "Implementing the Real-Time Publisher/Subscriber Model on the Controller Area Network (CAN)," In Proceedings of the 2nd IEEE International Symposium on Object-Oriented Real-Time Distributed Computing, 1999

(Kalfoglou et. al., 2003) Kalfoglou, Y., Schorlemmer, M., (2003), "Ontology mapping: the state of the art", The Knowledge Engineering Review, Volume 18, Issue 1, pages 1-31, 2003

(Keeney et. al., 2006a) Keeney, J., Lewis, D., O'Sullivan, D., Roelens, A., Wade, V., Boran, A., Richardson, R., (2006), "Runtime Semantic Interoperability for Gathering Ontology-based Network Context", In Proceedings of International Conference on Network Operations and Management Symposium (NOMS 2006), Canada. April 2006.

(Keeney et. al., 2006b) Keeney, J., Lewis, D., O'Sullivan, D., (20069b), "Benchmarking Knowledge-based Context Delivery Systems", In Proceedings of the International Conference on Autonomic and Autonomous Systems (ICAS 06), Silicon Valley, USA, July 19-21, 2006.

(Keeney et. al., 2006c) Keeney, J., Lynch, D., Lewis, D., O'Sullivan, D., (2006c), "On the Role of Ontological Semantics in Routing Contextual Knowledge in Highly Distributed Autonomic Systems", Tech. Report (TCD-CS-2006-15), Dept of Computer Science, Trinity College Dublin, 2006

(Keeney et. al., 2007) Keeney, J., Lewis, D., O'Sullivan, D., (2007), "Ontological Semantics for Distributing Contextual Knowledge in Highly Distributed Autonomic Systems", Journal of Network and System Management, Vol 15, March 2007

(Keeney et. al., 2008a) Keeney, J., Roblek, D., Jones, D., Lewis, O'Sullivan, D., (2008), "Extending Siena to support more expressive and flexible subscriptions," in The 2nd International Conference on Distributed Event-Based Systems (DEBS 2008), Rome, Italy., 2008.

(Keeney et. al., 2008b) Keeney, J., Jones, D., Roblek, D., Lewis, D., O'Sullivan, D., (2008), "Knowledge-based Semantic Clustering," In Proceedings of ACM Symposium on Applied Computing, Fortaleza, Brazil, 2008.

(Klein 2001) Klein, M., (2001), "Combining and Relating Ontologies: an Analysis of Problems and Solutions," In Proceedings of the 17th international joint Conference on Artificial Intelligence (IJCAI-01), Workshop: Ontologies and Information Sharing, Seattle, USA. 2001

(Kremen et. al., 2008) Kremen, P., Sirin, E., (2008), "SPARQL-DL Implementation Experience", In Proceedings of 4th OWL Experiences and Directions Workshop (OWLED-2008 DC), Washington, DC, USA, 2008

(Langheinrich et. al., 2000) Langheinrich, M., Mattern, F., Romer, K., Vogt, H., (2000), "First Steps Towards an Event-Based Infrastructure for Smart Things," In Proceedings of Ubiquitous Computing Workshop, PACT, 2000

(Learner et. al., 2000) Lerner, U., Parr, R., Koller, D., Biswas, G., (2000), "Bayesian Fault Detection and Diagnosis in Dynamic Systems", In Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence, pages 531-537, AAAI Press / The MIT press, 2000.

(Lefort et. al., 2006) Lefort, L., Taylor, K., Ratcliffe, D., (2006), "Towards Scalable Ontology Engineering Patterns: Lessons Learned from an Experiment based on W3C's Part-whole Guidelines ", In Proceedings of the 2nd Australasian Workshop on Advances in Ontologies, (AOW 2006), Hobart, Australia, 5 December, 2006.

(Lewis et. al., 2005) Lewis, D., O'Sullivan, D., Power, R., Keeney, J., (2005), "Semantic Interoperability for an Autonomic Knowledge Delivery Service", In Proceedings of the 2nd IFIP TC6 International Workshop on Autonomic Communication (WAC 2005), Vouliagmeni, Athens, Greece, October 2005.

(Lewis et. al., 2006a) Lewis, D., O'Sullivan, D., Keeney, J., (2006a), "Towards the Knowledge-Driven Benchmarking of Autonomic Communications", in Proceedings of the Second International IEEE WoWMoM Workshop on Autonomic Communications and Computing (ACC 2006), Niagara-Falls / Buffalo, New York, USA, 26 June 2006.

(Lewis et. al., 2006b) Lewis, D., Keeney, J., O'Sullivan, D., Guo, S., (2006), "Towards a Managed Extensible Control Plane for Knowledge-Based Networking", In Proceedings of Distributed Systems: Operations and Management Large Scale Management, (DSOM 2006), at Manweek 2006, Dublin, Ireland, 23-25 October 2006

(Lewis et. al., 2006c) Lewis, D., O'Sullivan, D., Feeney, K., Keeney, J., Power, R., (2006b), "Ontology-based Engineering for Self-Managing Communications", In Proceedings of the 1st IEEE International Workshop on Modelling Autonomic Communications Environments (MACE 2006), at Manweek 2006, Dublin, Ireland, 23-25 October 2006.

(Liebig et. al., 1999) Liebig, C., Boesling, B., Buchmann, A., (1999), "A Notification Service for Next-Generation IT Systems in Air Traffic Control," In Proceedings of GI-Workshop: Multicast-Protocol and Anwendungen, 1999

(Lópezdevergara et. al., 2002) López de Vergara, J.E., Villagrá, V.A., Berrocal, J, (2002), "Semantic Management: Advantages of Using an Ontology-based Management Information Metamodel", In Proceedings of HP-OVUA'2002, distributed videoconference, 11-13 June 2002

(Lynch et. al., 2006) Lynch, D., Keeney, J., Lewis, D., O'Sullivan, D., (2006), "A Proactive Approach to Semantically Oriented Service Discovery". In Proceedings of

Innovations in Web Infrastructure (IWI 2006), at World-Wide Web Conference, Edinburgh, Scotland, May 2006.

(Madhavan et. al., 2001) Madhavan J., Bernstein P., Rahm E., "Generic Schema Matching with Cupid", Proceedings of International Conference on Very Large Databases (VLDB), 2001.

(Maedche et. al., 2002) Alexander Maedche, Boris Motik, Nuno Silva, and Raphael Volz. Mafra – amapping framework for distributed ontologies. In Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web: 13th International Conference, EKAW 2002, Siguenza, Spain, page 235, 2002.

(Maedche et. al., 2003) Alexander Maedche, Boris Motik, Ljiljana Stojanovic, Rudi Studer, and Raphael Volz. Ontologies for enterprise knowledge management. IEEE Intelligent Systems, 18(2):26–33, 2003.

(Mahambre et. al, 2007) Mahambre, S. P., Madhu Kumar, S. D., Bellur, U., (2007), "A Taxonomy of QoS-Aware, Adaptive Event-Dissemination Middleware", In Journal of IEEE Internet Computing, 35-44, 2007

(Mahambre et. al., 2007) Mahambre, S.P., Madhu-Kumar, S.D., Bellur, U., (2007) "A Taxonomy of QoS-Aware, Adaptive Event-Dissemination Middleware," IEEE Internet Computing, vol. 11, no. 4, pp. 35-44, July/Aug. 2007,

(Marcot et. al., 2006) Marcot, B.G., Steventon, J.D., Sutherland, G.D., McCann, R.K., (2006), "Guidelines for Developing and Updating Bayesian Belief Networks for Ecological Modeling", Can J. For. Res. 36. This Issue, 2006

(Marvis et. al., 1999) Mavris, D. N., Bandte, O., & Delaurentis, D. A. (1999). Robust design simulation: A probabilistic approach to multidisciplinary design. Journal of aircraft, 36(1), 298-307, 1999

(Masuoka et. al., 2004) Masuoka, R., Labrou, Yannis, Parsia, B., Sirin, E., (2004) "Ontology-Enabled Pervasive Computing Applications", IEEE Intelligent Systems, Sep-Oct 2004, pp 68-72., 2004

(Matsuura et. al., 2004) Matsuura, P., Yoneyama, T., J., "Learning Bayesian Networks for Fault Detection", in Machine Learning for Signal Processing, 2004, In Proceedings of the 2004 14th IEEE Signal Processing Society Workshop, Pages 133-142, 2004.

(McCann et. al., 2006) McCann, R., Marcot, B.G., Ellis, R. (2006), "Introduction: Bayesian belief networks: application in natural resource management". Can. J. For. Res. 36. This issue, 2006

(McGuinness et. al., 2000) McGuinness, D. L., Harmelen, F. van., (2004), "OWL Web Ontology Language Overview," W3C Recommendation, vol. 10, pp. 2004-03, 2004.

(Meier et. al., 2005) Meier, R., Cahill, V., (2005), "Taxonomy of Distributed Event-Based Programming Systems", The Computer Journal, vol 48, no 5, pp 602-626, 2005

(Meira 1997) Meira, D. M., (1997), "A Model For Alarm Correlation in Telecommunications Networks", Computer Science, Institute of Exact

Sciences (ICEx) of the Federal University of Minas Gerais, Belo Horizonte, Brazil, 149. 1997

(Melnik et. al., 2000) Melnik S., Garcia-Molina H., Rahm E., (2002), “Similarity Flooding: A Versatile Graph Matching Algorithm”, ICDE 2002.

(Milenko et. al., 2003) Milenko, P., Burcea, I., Jacobsen, H.A., (2003), "S-ToPSS: semantic Toronto publish/subscribe system," in Proceedings of the 29th international conference on Very large data bases - Volume 29 Berlin, Germany: VLDB Endowment, 2003.

(Mindswap) “The Mindswappers Ontology”, Avaialbe at http://www.mindswap.org/2004/owl/mindswappers, Last visited: June 2006

(Mitra et. al., 2002) Mitra, P., Wiederhold, G., (2002), “Resolving Terminological Heterogeneity in Ontologies”, In Proceedings of ECAI-02 Workshop, CEUR-WS 64, Ontologies and Semantic Interoperability, 2002

(Motik et. al., 2006) Motik, B., Sattler, U., (2002), “Practical DL Reasoning over Large Aboxes with KAON2”, found at: http://www.fzi.de/KCMS/kcms_file.php?action=link&id=580, expected publication 2006.

(Muhl 2002) Muhl, G. (2002), “Large-scale Content-based Publish/Subscribe Systems”, PhD Thesis, Computer Science, Darmstadt University of Technology, 2002

(Mühl et. al., 2006) Mühl, G., Fiege, F., Pietzuch, P., (2006), “Distributed Event-Based Systems”. Springer-Verlag, 2006

(Neapolitan 2004) Neapolitan, R.E., (2004), “Learning Bayesian Networks”, Prentice Hall, 2004

(Nejdl et. al., 2002) Nejdl, W., Wolf, B., Qu, C., Decker, S., Sintek, M., Naeve, A., Nilsson, M., Palm, M., Risch, T., (2002), "EDUTELLA: a P2P networking infrastructure based on RDF," In Proceedings of the 11th international conference on World Wide Web, 2002, pp. 604-615, 2002

(Netica 2008) “Netica Tutorial” Available at: http://www.norsys.com/tutorials/netica/nt_toc_A.htm, Last Visit: November 2008.

(NeticaV4.0.8) Netica Bayesian Network Software, Available at: http://www.norsys.com/

(Noy 2004) Noy, N., (2004), “Semantic Integration: A Survey of Ontology-Based Approaches”, Special Issue on Semantic Integration, SIGMOD Record, Volume 33, Issue 4, pages 65-70, December 2004.

(Noy et. al., 2000) Noy, N., Musen, M., (2000), “Algorithm and Tool for Automated Ontology Merging and Alignment”, In Proceeding of Seventeenth National Conference on Artificial Intelligence (AAAI-2000), 2000

(Noy et. al., 2001) Noy, F.N., McGuinness, D. L., (2001), “Ontology Development 101: A Guide to Creating Your First Ontology”, Stanford Knowledge Systems Laboratory Technical Report KSL−01−05 and Stanford Medical Informatics Technical Report SMI−2001−0880, March, 2001

(Noy et. al., 2003) Noy, N., and Musen M., (2003), “The PROMPT Suite: Interactive Tools for Ontology Merging and Mapping”, International Journal of Human-Computer Studies, Volume 59, Issue 6, pages 983-1024, 2003.

(O’Sullivan 2006) O’Sullivan, D., (2006), “The OISIN framework: Ontology Interoperability in support of Semantic Interoperability”, PhD thesis, Computer Science, Trinity College, Dublin, 2006

(O’Sullivan et. al., 2004) O’Sullivan, D., Lewis, D., Wade, V., (2004), “Enabling Adaptive Semantic Interoperability for Pervasive Computing”, In Proceedings of the 2nd International Workshop on Managing Ubiquitous Communications and Services, December 2004.

(OAEI 2007) “Ontology Alignment Evaluation Initiative”, Available at http://oaei.ontologymatching.org/2007/, Last visited: October 2008.

(Omelayenko 2002) Omelayenko, B., (2002), "Integrating Vocabularies: Discovering and Representing Vocabulary Maps," In Proceedings of the First International Semantic Web Conference (ISWC-2002), Sardinia, Italy, June 9-12, 2002

(OMG 2000) Object Management Group. “CORBA Event Service Specification, version 1.0.”, OMG Document formal/2000-06-15, 2000

(OMG 2001) O.M.G. (OMG). Complete UML 1.4 Specifications, 2001.

(Omondo 3.3) UML Toolkit for Eclipse, Available at: http://www.eclipsedownload.com/index.html, Last Visit: September 2008.

(OntoFarm 2007) “OntoFarm Project”, Available at http://nb.vse.cz/~svatek/ontofarm.html, Last visited: October 2008.

(Orgun et. al., 2006) Orgun, B., Dras, M., Nayak, A., James, G., (2006), "Approaches for Semantic Interoperability between Domain Ontologies," In Proceedings of the second Australasian workshop on Advances in Ontologies - Volume 72 Hobart, Australia: Australian Computer Society, Inc.

(Ouksel et. al., 1999) Ouksel, A. M., and Sheth, A., (1999), "Semantic interoperability in global information systems," ACM SIGMOD Record, vol. 28, pp. 5-12, 1999.

(OWL 2004a) “OWL Web Ontology Language, Use Cases and Requirements”, W3C Recommendation, 10 February 2004, http://www.w3.org/TR/2004/REC-webont-req-20040210/, Last visited: August 2008.

(OWL 2004b) “OWL Web Ontology Language Overview”, W3C Recommendation, 10 February 2004, Available at: http://www.w3.org/TR/2004/REC-owl-features-20040210/, Last visited: August 2008.

(OWL 2006) “Achieve Semantic Interoperability and Integration Using RDF and OWL”, Available at http://cmenzel.org/w3c/SemanticInterop.html, Last visited: June 2006

(OWL-S Mapping) “OWL-S Map Ontology”, Available at http://ontology.ihmc.us/SemanticServices/KAoS-OWL-S-Mapping.owl, Last visited: June 2006

(Pan 2005) Pan, Z., (2005), “Benchmarking DL Reasoners Using Realistic Ontologies”, In Proceeding of the International Workshop on OWL: Experience and Directions, (OWLED2005), Galway, Ireland. 11-12 November, 2005.

(Parsia et. al., 2004) Parsia, B., Sirin, E., (2005), “Pellet: An OWL-DL Reasoner”, Poster at ISWC 2004, Hiroshima, Japan, 2004.

(Patrick et. al., 2006) Patrick, P., Andrew, P., Eduard, H. (2006), “Matching and Integration across Heterogeneous Data Sources”, In Proceedings of the 2006 International Conference on Digital Government Research, San Diego, California: ACM. 2006

(Pellet 2003) “Pellet Performance”, (200V), http://www.mindswap.org/2003/pellet/performance.shtml

(Petrovic et. al., 2003) Petrovic, M., Burcea, I., Jacobsen, H. A., (2003), "S-ToPSS: Semantic Toronto Publish/Subscribe System," In Proceedings of the 29th international conference on Very large data bases (VLDB03), Berlin, Germany, 2003, pp. 1101-1104, 2003

(Pflugel et. al., 1993) Pflugel M., Siegel, A., Skeen, D., (1993), “The Information Bus – an Architecture for Extendible Distributed System”, In Proceedings of the 14th Symposium on Operating Systems Principles, pages 58-68, Asheville, NC, United States, Dec. 1993. ACM press, 1993

(Pietzuch et. al., 2002) Pietzuch, P., Bacon, J., (2002), "Hermes: A Distributed Event-Based Middleware Architecture," In Proceedings of International Conference on Distributed Computing Systems, 2002.

(Power 2008) Power, R., (2008), “Cashua: Integrating Semantic and Content-Based Networking for Context Distribution”, PhD thesis, Computer Science, Trinity College, Dublin, 2008

(RDF 2004) "RDF/XML Syntax Specification (Revised)", W3C, February 2004, Available at: http://www.w3.org/TR/rdf-syntaxgrammar/, Last visited: September 2008.

(RDFS 2004) "RDF Vocabulary Description Language 1.0: RDF Schema", W3C Recommendation, February 2004, http://www.w3.org/TR/rdf-schema/, Last visited:October 2008.

(Robert et. al., 2004) Robert, J.H., Andrea, Z., (2004), “Model Interchange and Integration for Web Services”, SIGSOFT Software, Eng. Notes, 29(5), 1-11, 2004

(Roblek 2006) Roblek, D., (2006), "Decentralized Discovery and Execution for Composite Semantic Web Services", M.Sc. Thesis, Computer Science, Trinity College Dublin, Ireland, 2006.

(Rowstron et. al., 2001) Rowstron, A., Druschel, P., (2001),"Pastry: Scalable, Decentralized Object Location, and Routing for Large-Scale Peer-to-Peer Systems," Lecture Notes in Computer Science, vol. 2218, 2001

(Schlosser et. al., 2002) Schlosser, M. Sintek, M., Decker, S., Nejdl, W., (2002), “HyperCuP — Hypercubes, Ontologies and Efficient Search on P2P Networks”, In

Proceedings on the International Workshop on Agents and Peer-to-Peer-Systems, Bologna, Italy, 2002

(Sebyala et. al., 2002) Sebyala, A.A., Olukemi, T., Sacks, L., (2002), "Active Platform Security through Intrusion Detection using Naïve Bayesian Network for Anomaly Detection", In Proceedings of the London Communications Symposium 2002.

(Segal et. al., 2000) Segall, B., Arnold, D., Boot, J., Henderson, M., Phelps, T., (2000), "Content-Based Routing in Elvin4", In Proceedings of AUUG2K, Canberra 2000

(SID 2007) Shared Information/Data Model, Available at: http://www.tmforum.org/SharedInformationData/968/home.html, Last visited: September 2007.

(Skovronski et. al., 2006) Skovronski, J., Chiu, K., (2006), "Ontology Based Publish Subscribe Framework". In Proceedings of International Conference on Information Integration and Web-based Applications Services, 4-6 December 2006, Yogyakarta, Indonesia, 2006

(SMI v2) Structure of Management Information, Available at: http://tools.ietf.org/html/rfc2580, Last visited: September 2007

(Spiliopoulos et. al., 2007) Spiliopoulos, V., Valarakos, A.G., Vouros, G.A., Karkaletsis, V., (2007), "SEMA: Results for the Ontology Alignment Contest OAEI 2007". In Proceedings of Ontology Matching Workshop, OAEI, Busan, Korea, 2007

(Strassner et. al., 2006) Strassner, J., Agoulmine, N., Lehtihet, E., (2006), "FOCALE-A Novel Autonomic Computing Architecture", LAACS, 2006

(Strassner et. al., 2007) Strassner, J., O'Sullivan D., David, L. (2007), "Ontologies in the Engineering of Management and Autonomic Systems: A Reality Check", Journal of Network and Systems Management, 15(1), 5-11, 2007

(Starlab 2003) Starlab 2003. Systems Technology and Applications Research Laboratory home page. Faculty of Sciences, Department of Computer Science, Vrije Universiteit Brussel. Available at: http://www.starlab.vub.ac.be/default.htm, Last visit: January 2006

(Su 2004) Su, X., (2004), "Semantic Enrichment for Ontology Mapping", PhD thesis, Computer Science, Norwegian University of Science and Technology, Norway, 2004

(Sun 1998) Sun Microsystems, Inc. Distributed Event Specification, 1998.

(Sure et. al., 2000) Sure, Y., Maedche, A., Staab, S., (2000), "Leveraging Corporate Skill Knowledge-From ProPer to OntoProper," In Proceedings of the Third International Conference on Practical Aspects of Knowledge Management. Basel, Switzerland, October, pp. 30-31, 2000

(Svab et. al., 2007) Svab O., Svatek V., Stuckenschmidt H., (2007), "A Study in Empirical and Casuistic' Analysis of Ontology Mapping Results", In proceedings of 4th European Semantic Web Conference (ESWC2007), Innsbruck, 2007

(Swartout et. al., 1996) Swartout, B., Patil, R., Knight, K., Russ, T., (1996), “Toward Distributed Use of Large−Scale ontologies”, In Proceedings of the Tenth Knowledge Acquisition for Knowledge− Based Systems Workshop, (KAW ’96), November 9−14, Banff, Alberta, Canada,1996

(Swoogle 2008) Semantic Web Search Engine, Available at: http://swoogle.umbc.edu/, Last visit: August, 2008

(Swoop 2004) Swoop OWL ontology editor, Available at: http://www.mindswap.org/2004/SWOOP/, Last visit: September 2007.

(Tempich et. al., 2003) Tempich, C., Volz, R., (2003), “Towards a Benchmark for Semantic Web Reasoners - an Analysis of the DAML Ontology Library”, in Proceedings of Evaluation of Ontology-based Tools, (EON2003), at 2nd International Semantic Web Conference (ISWC 2003), Florida, USA, 20-23 October, 2003.

(Terpstra et. al., 2003) Terpstra, W.W., Behnel, S., Fiege, L., Zeidler, A., Buchmann, A.P., (2003), “A Peer-to-Peer Approach to Content-Based Publish/Subscribe”, In Proceedings of DEBS 2003, ACM Press 2003

(TIBCO 1996) TIBCO, Inc. TIB/Rendezvous, White Paper, (1996), Found at http://www.rv.tibco.com

(Uschold et. al., 2004) Uschold, M., Gruninger, M., (2004), "Ontologies and Semantics for Seamless Connectivity", Special Issue on Semantic Integration, SIGMOD Record, Volume 33, Issue 4, pages 58-64, December 2004.

(Valente et. al., 1999) Valente, A., Russ, T., MacGrecor, R., Swartout, W., (1999), “Building and Reusing an Ontology for Air Campaign Planning”, Journal of IEEE intelligent system, 14(1): 27-36, 1999

(Vergara et. al., 2002) Vergara, J.E. Lopez De, Villagra, V.A., Berrocal, J., (2002), “Semantic Management: Advantages of Using an Ontology-based Management Information metamodel”, In Proceedings of the HP Open view University Association Ninyh Plenary Workshop (HP – OVUA’2002), distributed videoconference, 11-13 June 2002.

(W3C, 2003) W3C: “The Wine Ontology” (2003), available at: http://www.w3.org/TR/owl-guide/wine.rdf. Last Visit: June 2006

(Wang et. al., 2004) Wang, J., Jin, B., Li, J., (2004), “An ontology-based publish/subscribe system”. In Proceedings of ACM/IFIP/USENIX International Conference on Middleware, 2004

(Wine) “W3C: The Wine Ontology” Available at http://www.w3.org/TR/2003/PR-owl-guide-20031209/wine, Last visited: August 2006

(Wong et. al., 2005) Wong, A. K. Y., Ray, P., Parameswaran, N., Strassner, J. (2005), “Ontology mapping for the interoperability problem in network management”, in IEEE Journal of Selected Areas in Communications, 23(10), 2058-2068, 2005

(Wun et. al., 2007) Wun, A., Cheung, A., Jacobsen, H-A., (2007), “A Taxonomy for Denial of Service Attacks in Content-based Publish/Subscribe Systems”, In

Proceedings of the 2007 inaugural international conference on Distributed event-based systems (DEBS 2007), Toronto, Canada, June 20th-22nd, 2007.

(XG-policy 2004) DARPA XG Working Group “DARPA XG Policy Language Framework, Request for Comments”, version 1.0, prepared by BBN Technologies, Cambridge MA, USA, April 16 2004

(XML 2000) "Extensible Markup Language (XML) 1.0 (Second Edition)",W3C, October 2000, Available at:http://www.w3.org/TR/2000/REC-xml-20001006, Last visited: September 2008.

(XMLSchema 2004) “XML Schema Part 0: Primer Second Edition", W3C, October 2004, Available at: http://www.w3.org/TR/xmlschema-0/,Last visited: September 2005.

(Yahia et. al., 2006) Yahia, B.G.I, Bertin, E., Crespi, N., (2006), “Next/New Generation Networks Services and Management”, In Proceedings of International Conference on Networking and Services (ICNS 2006), July 16-19, Silicon Valley, USA, 2006

(Zhao et. al., 1999) Zhao, Y., Sturman, D., Bhola, S. (1999), “Matching events in a content-based subscription system”, In Proceedings of the ACM Symposium on Principles of Distributed Computing, 1999

(Zhu et. al., 2003) Zhu, S., Xu, S., Setia, S., & Jajodia, S, (2003), “Establishing Pair-wise Keys For Secure Communication in Ad Hoc Networks: A Probabilistic Approach” In Proceedings of the 11th IEEE International Conference on Network Protocols, Washington, DC, USA, 2003

APPENDICES

The appendices present support information for this thesis.

- Appendix A provides an overview of the technology used in the thesis.
- Appendix B presents the DL reasoner experiments to support Chapter 4 and Chapter 5.
- Appendix C presents an initial evaluation of mapping strategies designed in Chapter 4.
- Appendix D presents the links of ontologies used in the thesis.
- Appendix E provides the CPTs of nodes for Bayesian network model for mapping strategy selection which was developed in Chapter 6.
- Appendix F presents supporting information for the KBNMap evaluation in Chapter 7.

A. TECHNOLOGY USED IN KBNMAP

This appendix presents the technologies used in the implementation of KBNMap. The Jena ontology environment used by the KBNMap router for processing and managing routing ontologies is presented in Section A.1. The Pellet reasoner used by the router is presented in Section A.2. A description of Netica and JavaBayes used for processing and editing the BN model is used in Section A.3.

A.1 Jena

A.1.1 Overview

Jena is a Java framework for building Semantic Web (Jena 2005) applications, and allows processing and manipulation of RDF, RDFS and OWL ontologies. It also includes an inference engine using a variety of reasoners and SPARQL query support. Jena was developed as part of HP Labs Semantic Web Research[7], and was released under an open source license in order to encourage its adoption.

[7] http://www.hpl.hp.com/semweb/

```
ExtendedIterator iter = ontModel. listClasses ();
while (iter . hasNext ()) {
        OntClass ontClass = ( OntClass ) iter. next ();
        String classURI = ontClass . getURI ();
        conceptList .add ( classURI );
}
```

KBNMap makes extensive use of Jena's ontology API, which is language-neutral, using calls which are not specific to particular ontology languages. This allows Jena applications to concentrate on the manipulation of the ontology, rather than the specifics of the ontology language used. Jena provides a consistent programming interface to the developer, independent of which ontology language is being used. The features of ontology languages are expressed using profiles, which when bound to an ontology model allow for querying of the information stored in the underlying RDF model, using the facilities supported by the ontology language.

```
OntModel ontModel = ModelFactory.createOntologyModel();
ontModel.read ("file:ont1" + entFile);
```

A.1.2 Ontology API

The KBNMap router makes use of the Jena library to load the representations of the routing ontologies, mapping files/referenced ontologies into a model in memory to form a knowledge base for querying and routing information. This is achieved using the following method calls that first instantiates an in-memory object for the model and then reads the ontology file into that object.

This model allows access to the ontology statements via a collection of RDF data represented as a graph. The model is then queried by the routers, using various method calls available from the Jena library. The methods that are possible to invoke on the model depend on both the asserted statements in the underlying RDF graph, and the statements that can be inferred by the OWL reasoner being used. For example, the method listClasses below lists all ontological classes in the ontology:

```
OntClass ontClass = ontModel. getOntClass ( ontClassURI );
String comment = ontClass . getComment ( null );
```

The method getOntClass below allows the applications to access a specific ontological class given the URI of that class in the ontology.

A.2 Pellet Reasoner

A.2.1 Overview

The following description is drawn from the Pellet documentation (Pellet 2003). Pellet is an open-source Java based OWL DL reasoner. It can be used in conjunction with both Jena and OWL API libraries and also provides a DIG interface. The Pellet API provides functionalities for the species validation, check consistency of ontologies, classify the taxonomy, check entailments and answer a subset of RDQL queries (known as ABox queries in DL terminology).

Pellet is an OWL DL reasoner based on the tableaux algorithms developed for expressive Description Logics. It supports the full expressivity OWL DL including reasoning about nominals (enumerated classes). Therefore, OWL constructs owl:oneOf and owl:hasValue can be used freely. Currently, Pellet is the first and only sound and complete DL reasoner that can handle this expressivity (Parsia et. al., 2004). There are two typical Pellet structural reasoners available: one is the "`OWLReasoner`" designed to provide an Jena-like interface to applications using Jena to access ontologies by translating the results of common calls to provide Jena data structures; the second "`OWLAPIReasoner`" reasoner is designed to support applications using the OWL API framework (Bechhofer et. al., 2003) to query and manipulate ontologies.

In current KBNMap implementation, the Pellet reasoner accessed through the Jena framework as the default reasoning configuration for the KBNMap router, as the Pellet reasoner provide sound and complete reasoning services for OWL-based applications to manage their OWL ontologies properly.

OntModel ontModel = ModelFactory.*createOntologyModel*(PelletReasonerFactory.*THE_SPEC*, null);

A.2.1 API Overview

The Pellet reasoner is registered with the ontology model of Jena:

A.2 Bayesian Network Modelling Platforms

A.2.1 Netica Overview

The following description is drawn from the Netica documentation (NeticaV4.0.8). Netica is a powerful, easy-to-use, complete platform for working with belief networks and influence diagrams. It has an intuitive and smooth user interface for drawing the networks, and the relationships between variables may be entered as individual probabilities, in the form of equations, or learned from data files (which may be in ordinary tab-delimited form and have "missing data").

Netica can use the networks to perform various kinds of inference using the fastest and most modern algorithms. Given a new case of which a limited knowledge is available, Netica will find the appropriate values or probabilities for all the unknown variables. Netica can use influence diagrams to find optimal decisions which maximize the expected values of specified variables. Netica can construct conditional plans, since decisions in the future can depend on observations yet to be made, and the timings and inter-relationships between decisions are considered. In addition, Netica can be used to transform a network in a number of ways. Variables that are no longer of interest may be removed without changing the overall relationships between the remaining variables (technically, the probabilities are "summed out" when we don't know the variable's value, and a more complex operation is used when we do). Probabilistic models may be explored by such operations as reversing individual links of the network, removing or adding causal influences, optimizing one decision at time, etc.

In the thesis, Netica was used to construct, compile, and evaluate the BN Model for selecting the appropriate mapping strategy, as it is the world’s most widely used BN development software and has been broadly used by researchers in various research domains (Learner et. al., 2000; Matsuura et. al., 2004; Marcot et. al., 2006)

A.2.1 JavaBayes Overview

The JavaBayes (JavaBayes 2001) system is a set of tools for the creation and manipulation of Bayesian networks. The system is composed of a graphical editor, a core inference engine and a set of parsers. The graphical editor allows you to create and modify Bayesian networks in a friendly interface. The parsers allow you to import Bayesian networks in a variety of formats. The engine is responsible for manipulating the data structures that represent Bayesian networks. The engine can produce:

- The marginal probability for any variable in a Bayesian network. In the thesis, the marginal probability refers to the weightings of mapping strategies in the final node: StrategySelection.
- The expectations for univariate functions (for example, the expected value of a variable).
- Configurations with maximum a posteriori probability.

JavaBayes can produce marginal distributions and expectations using two different algorithms: variable elimination and bucket tree elimination. In the first case, inferences are generated from scratch for each query; in the second case, a data-structure (bucket tree) is generated once and several queries can be generated directly from the bucket tree. Variable elimination consumes less memory, but it may take longer if several queries are made to the same network with the same collection of observations.

In the thesis, the JavaBayes was chosen to be integrated with the KBNMap router, acting as the Bayesian network inference engine for calculating the weightings of mapping strategies. This is due to not only the JavaBayes is an open source software, but also it has a small library that consumes a little resource for integration.

B. DL REASONER EXPERIMENT

This experiment was undertaken to compare OWL Description Logic (DL) reasoner implementations, in order to determine the extent to which ontology reasoner implementations differed with respect to initial loading and subsequent querying times.

B.1 Experiment Goal

Recent exploratory experimentation by the authors in (lynch06 et. al., 2006; keeney et. al., 2006) has evaluated the integration of ontological equivalence and subsumption into Siena CBN event/subscription matching yielding the following results. Firstly, the loading of new ontologies into a reasoner embedded in a KBN node is computationally expensive due to load-time inference, so the frequency of changes to the ontological base of a given KBN node must be minimised since changes will need to be distributed to each of the nodes in the network. Secondly, ontological reasoning is memory intensive and memory usage is proportional to the number of concepts and relationships loaded into the reasoner so reasoning latency can be controlled by limiting this number in any given KBN node. However, once loaded and reasoned over, the querying of such an ontological base is relatively efficient, with performance relative to size of the ontological base. Based on these initial observations, that the (re-)loading and (re-)reasoning of ontologies is expensive and that such operations will greatly affect the scalability of the network, it was considered possible that such axioms could form the basis of semantic clustering policies for partitioning the routing mechanism, thereby localising the effect of such operations and minimizing the ontological base at each node and so improving routing performance.

In order to determine the extent to which ontology reasoner implementations differed with respect to initial loading and subsequent querying times, it was decided to undertake an experiment to compare OWL Description Logic (DL) implementations. Three typical DL implementations are measured in our experiment: Racer (Haarslev et. al., 2001), the Jena framework (Jena 2005), and Pellet (Parsia et. al., 2004). Racer implements a highly optimized tableau calculus for very expressive description logic and offers reasoning service for

multiple TBoxes and for multiple ABoxes. Pellet is an open source Java based OWL DL reasoner and supports the full expressivity of OWL DL including reasoning about nominals (enumerated classes). Two Pellet structural reasoners were tested: one is the "`OWLReasoner`" designed to provide an Jena-like interface to applications using Jena to access ontologies by translating the results of common calls to provide Jena data structures; the second "`OWLAPIReasoner`" reasoner is designed to support applications using the OWL API framework (Bechhofer et. al., 2003) to query and manipulate ontologies. Both Pellet and Racer implement tableaux optimization techniques to improve their description logic reasoning performance. Finally, the Jena framework supports a number of different inbuilt reasoning modes, four of which are discussed here: The first, the "OWL_MEM_RDFS_INT" reasoner, incorporated directly into Jena, performs RDFS entailment reasoning. The second "OWL_MEM_TRANS_INF" reasoner, also supplied with Jena, performs transitive reasoning. The third reasoner configuration is where Pellet is plugged directly into Jena to perform full OWL DL reasoning and is accessed directly from within Jena like one of Jena's own reasoners. The fourth mode "OWL_MEM", is where Jena undertakes no reasoning, and is included for comparison.

B.2 Experiment Metrics

A thorough and fair comparison of all aspects of these implementations is difficult, because they are implemented in different programming languages with the reasoning algorithms varying considerably. However for our purposes this is not important, rather we were interested in the performance of the implementations given a certain expected pattern of usage. This pattern of usage for knowledge routing is where the ontology is loaded and reasoned over initially at KBN start up time (which we call the loadtime stage) and then the ontology would be subsequently queried on a repetitive and ongoing basis (which we call the runtime stage) to facilitate the KBN routing.

Thus in the experiment, for a number of ontologies, loadtime performance is given as the times taken for the different reasoners to load, parse and check the ontology, combined with the time taken to perform TBox classification, perform ABox realisation and an initial query

of all concepts. The time taken to perform subsequent queries for the set of concepts in those ontologies was measured as the runtime performance of the reasoners.

B.2.1 Testing Ontologies

For the purpose of investigating the impact of the complexity of ontologies upon the reasoner implementations, five ontologies were chosen for test. These were chosen in order to reflect a range of ontology complexity, from simple ontologies with instances through to complex ontologies with no instances. First, a large but relatively simple subset of DMTF's CIM (Common Information Model) ontology with 167 classes and 733 individuals was used in the test. This ontology contains the set of CIM concepts required to manage printing devices. (A plug-in for the Protégé editor from the Universidad Politécnica de Madrid (UPM) (lópezdevergara et. al., 2002) was used for the conversion of the CIM object-oriented model of classes and properties to an ontology.) Secondly, we tested a more extensive ontology that represents the entire DMTF CIM Core Model, with 1215 classes and 5734 individuals. This ontology is very large by comparison to most ontologies, but relatively simple in terms of content. Next, the wine ontology (Wine) from the W3C community contains a classification of wines, with only 138 classes of which 61 are imported from a food ontology, and 206 individuals of which 45 are also imported. The nominals (individuals in class expression) and advanced DL constructors (e.g. disjunctions and equality) used in wine ontology makes it very complex to reason over, and is used as the de-facto stress test for OWL DL tools. The fourth test data set is the Galen ontology (Galen). It is a medical terminology ontology with a very large and complex class and property model (2749 classes) but without instances. It has traditionally been used as a benchmark for terminological reasoning. Finally, the Mindswap ontology (Mindswap) is a relatively small but detailed ontology containing the student and staff details and listings of the Semantic Web Research Group in the University of Maryland. It contains 49 classes, of which 37 are imported, and 122 individuals, of which only 6 are imported.

B.3 Experiment Setup

All tests were performed on a desktop computer with Dual 3.2 GHz Intel Xeon processors, 2GB of RAM, running Windows XP Service Pack 2. For Java-based tools, Sun's JDK 1.5.0 was used. All the timings presented in this Section are computed as the average of numerous independent timings.

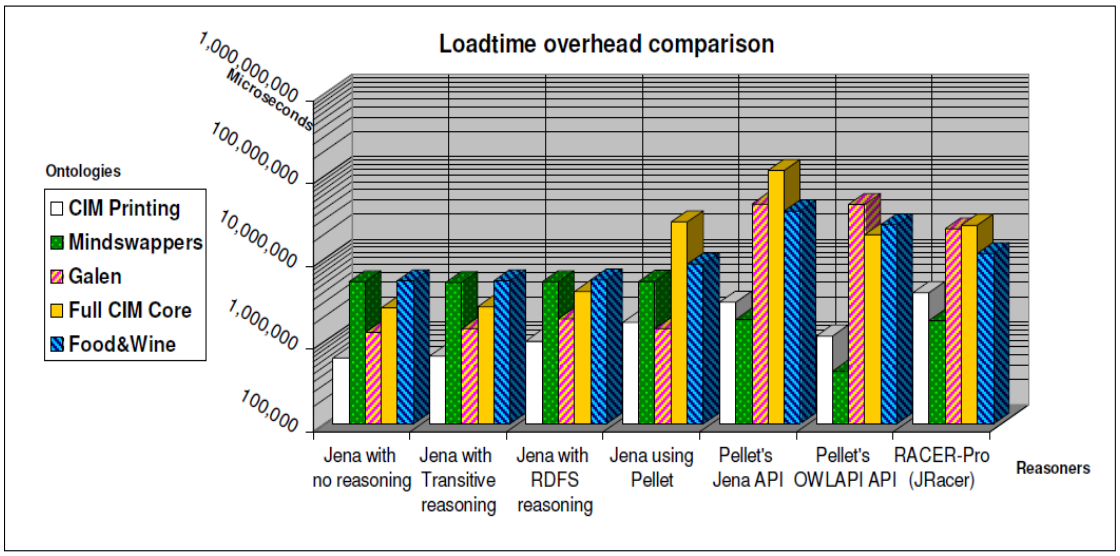


Figure B-1: LoadingTime Performances of DL Reasoners

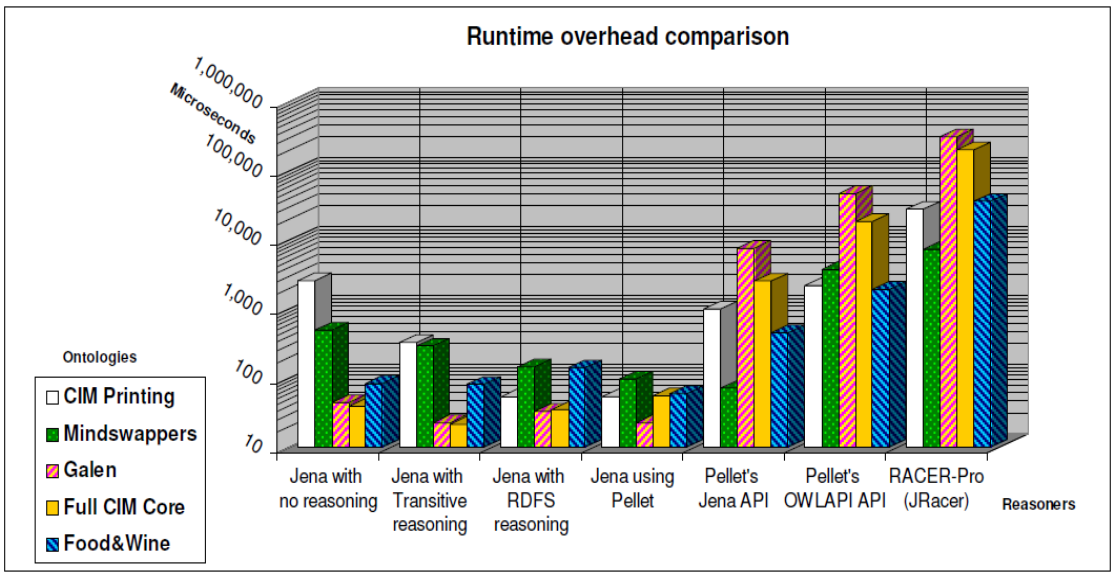


Figure B-2: Runtime Performances of DL Reasoners

B.4 The Experiment Result and Analysis

As can be seen from the loadtime performance (Figure B-1) of the different reasoners, all of the reasoners perform similarly with the small ontologies, with the ones accessed through Jena performing a little worse than Racer and Pellet when accessed directly. However for the large and complex ontologies the Jena based reasoners perform substantially better than the others. When the operation of the other Jena based reasoners is compared to the Jena configuration performing no reasoning it is clear that much of the Jena loadtime overhead is independent of the level of reasoning required. Of all of the reasoners, only Pellet, whether or not through Jena, performed an adequate level of reasoning on the complex Wine ontology. For this reason, of the reasoners tested, the Pellet reasoner configurations seems to perform best, perhaps using its own Jena style interface or its OWL API interface for small ontologies, but definitely embedded in Jena for large and complex ontologies.

From the runtime performance graph (Figure B-2) it is clear that all of the reasoners accessed using the Jena framework perform far superior to the others. This seems to be achieved by the clever use of model caching for second and subsequent queries within Jena. When compared to the other reasoner configurations, the use of Pellet through the Jena framework would appear to be the best overall choice for runtime performance. However, further investigation may be required to explain the counterintuitive poor performance of Jena's own reasoners with the smaller ontologies.

In the event that an off-the-shelf generic reasoner is required, taken together, the results point to the adoption of the Pellet reasoner accessed through the Jena framework as the default reasoning configuration. However, in situations where very good runtime performance is required, such as in network routing, and this performance requirement outweighs the need for complete and correct reasoning, the use of cut-down or application specific reasoners may prove beneficial. It is also envisioned that in a large scale network different reasoners will be used by different nodes in accordance with their role in the clustered network's routing scheme.

C. INITIAL STRATEGIES EVALUTATION

In order to determine which of the implemented strategies suit different application scenarios, it was decided to undertake an experiment to compare performances of each of the strategies and the impact of different numbers of mapping ontologies and imported ontologies on the strategies' performances.

C.1 Performance Metrics

Several metrics are measured in my experiments: **Loading time** which is given as the times taken for the reasoner to load, parse and check the ontology, combined with the time taken to perform TBox classification, perform ABox realisation and an initial query of all concepts. **Matching time** which is defined as the time taken to perform subsequent queries for the set of concepts in those test ontologies (Lewis et. al., 2006). The time taken to **check mapping files list** into the mapping store, in order to find correct mappings, was measured as checklist time. As well as the timing issue, I also measured number of ontology statements (that is the total number of classes, properties and individuals of all eventually loaded ontologies) in order to measure the **memory usage** by different strategies.

C.2 Domains and Mapping Ontologies

The first experiment compared the strategies by using two xg-regn ontologies and their mapping ontology (as shown in Appendix D). The xg-regn1 ontology is stored in the application ontology store in a KBN router. The mapping store of KBN router contains the region mapping ontology, whereas the xg-regn2 ontology is used as a source of unknown concepts for the experiment.

As for the second experiment, the xg-regn1 ontology is again used as the application ontology and region mapping ontology is stored in mapping store, another three mapping ontologies were chosen for testing: firstly, I created another region mapping ontology which was generated from the xg-regn2 ontology and a third xg-regn3 ontology. The second mapping data set is the owl-s mapping ontology (owl-s-mapping). It is used for policy management in

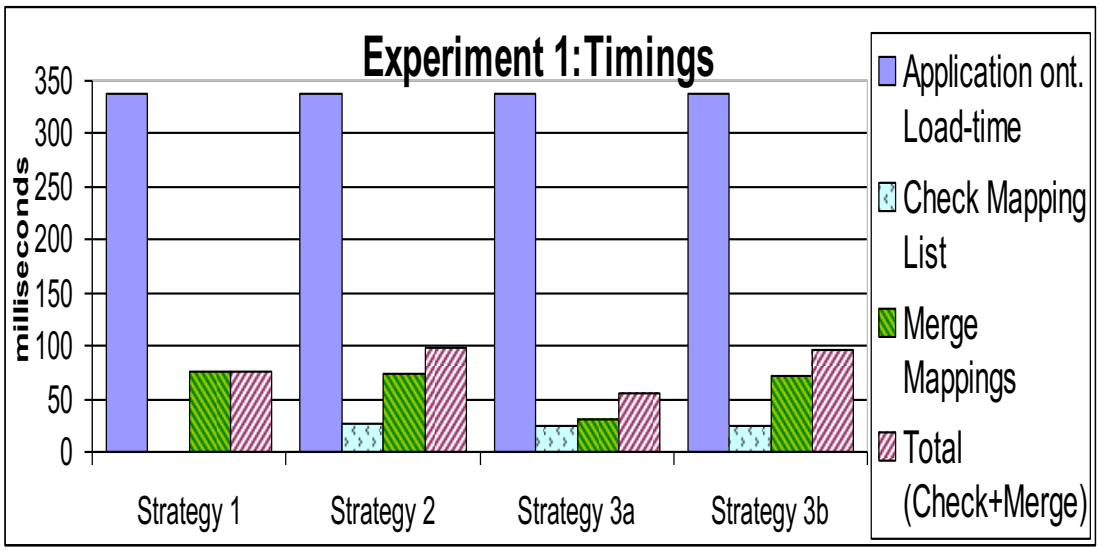
semantic web service with a large number of imported ontologies. Finally, the wob ontology is relative small mapping ontology without imported ontologies (Ding 2004).

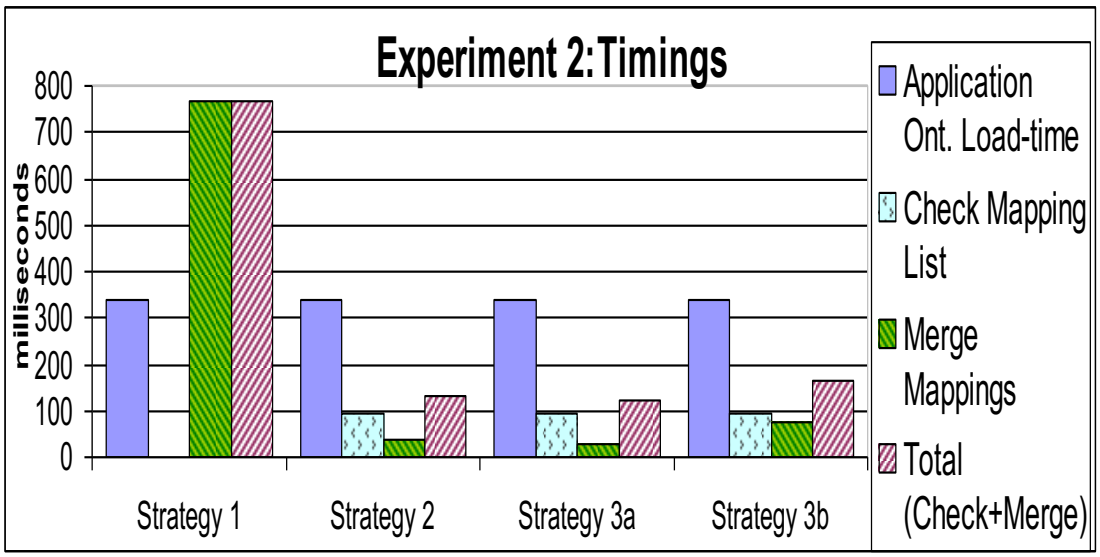
C.3 Experimental Setup

All tests were taken on a dell inspiron 9300 laptop with 1.73 GHz intel processor, 1GB of RAM, running Windows XP Service Pack 2. For java-based tools, sun's JDK 1.6.1 was used.

Figure C-2. Experiment 2: The two aggregate ontologies with multiple mapping ontologies

C.4 The Experiment Results and Analysis



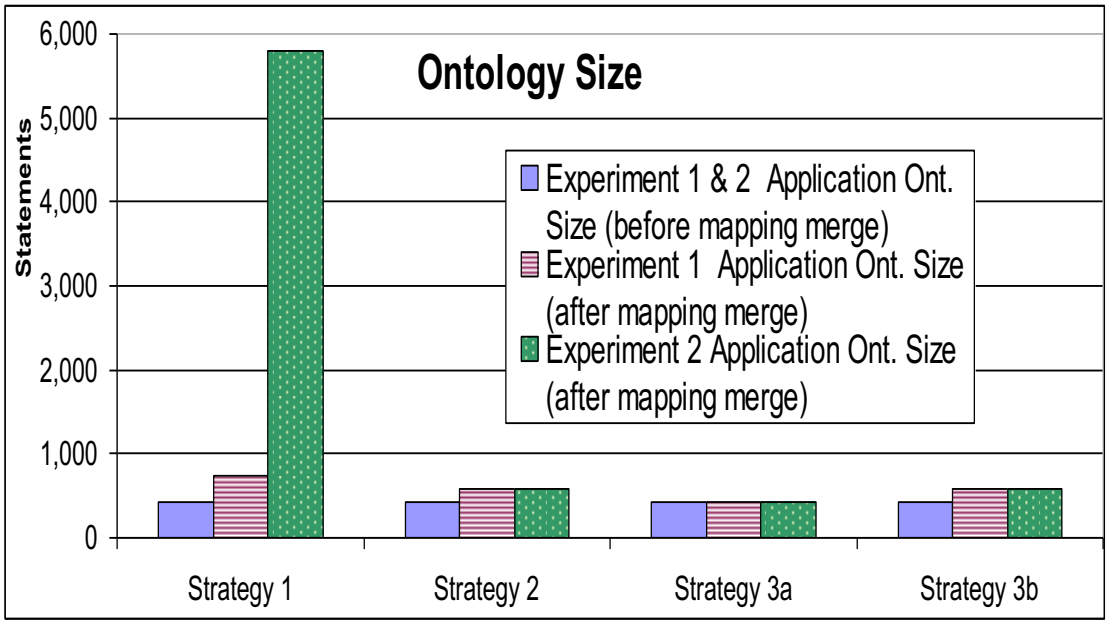


As can be seen from Figure C-1 and Figure C-2, the time taken for a KBN router to load the main application ontology at network set-up time same for each strategy.

In Experiment 1, with only one mapping file, strategy 1 performed only a little worse than strategy 3a, since unlike strategy 3a the *xg-regn2* ontology that is imported by the mapping ontology was also loaded and merged. By using strategies 2, 3a and 3b the KBN router only sometimes needed to load the referenced ontology. Compared to other strategies, strategy 1 did not check the mappings to select which mappings to merge, since all available mappings and referenced ontologies are loaded, and so strategy 1 outperformed strategies 2 and 3a. Strategies 2, 3a and 3b performed similarly for the "check mapping list" time.

Figure C-3: Experiment 1 & 2: Ontology Size

In Experiment 2 we see that as the number of mapping files in the KBN router increases, the performance of strategy 1 for mapping load time worsens dramatically. This is due to it having to load and merge more mapping ontologies and their imported ontologies. Strategy 3b performs a bit worse than strategy2 and strategy3a, because it merges referenced ontology into application ontology, when the direct mapping relations between known and unknown data are found in mapping file. We can also conclude that the increased number of mappings does not have a major effect on the performance of strategies 2, 3a and 3b, just increasing the "check mapping list" time.



From Figure C-3 we can see the dramatic effect on resources when strategy 1 is used. For this reason it is obvious that strategy 1 should not be used where there is a large number of mapping ontologies, with a large number of imported ontologies, especially where the ontologies are particularly large or resources are scarce. With respect to size, strategies 2, 3a and 3b are much more efficient as only the correct mapping ontology is loaded. Strategy 3a performs best since the mapping ontology is not merged, but just the individual mapping relationship. Strategy 3b performs very similar to strategy 2, except depending on the mapping relationship and operator used an imported ontology may also need to be loaded.

Overall, across all of the experiments, strategy1 seems to have performed worst, with each of the other strategies performing similar to each other. However, it must be remembered that for strategy 1, despite the once off cost of loading all of the mappings and referenced ontology, the KBN is much less likely to encounter unknown concepts in the future, thereby reducing the need to incorporate mappings at a later time. Similarly, although strategy 3b performs slightly worse than strategy 2 or strategy 3a, strategy 3b is more likely to load a complete referenced ontology. Overall, strategy 3a slightly out-performs strategy 2 and strategy 3b since it performs the least amount of ontology loading, querying and merging. This strategy is preferable where the occurrence of unknown concepts is rare and localised. Another point to remember is that the strategies that do not load the imported or referenced ontologies can result in the case where the unknown concept remains unknown after checking the mappings, and so the publication or subscription cannot be handled locally in a satisfactory manner.

D. LINKS OF USED ONTOLOGIES

This appendix presents the web links of ontologies used in experiments in the thesis.

D.1 Ontology Characterisation Experiment (Section 5.2.1)

Web ontologies:

1. Beer: http://www.purl.org/net/ontology/beer
2. CongoService: http://www.daml.org/services/owl-s/1.1/CongoService.owl
3. Economy:http://reliant.teknowledge.com/DAML/Economy.owl
4. Foaf: http://xmlns.com/foaf/0.1/index.rdf
5. Foodswap: http://www.mindswap.org/dav/ontologies/commonsense/food/foodswap.owl
6. Galen: http://www.cs.man.ac.uk/~horrocks/OWL/Ontologies/galen.owl
7. Mad_cow: http://www.cs.man.ac.uk/~horrocks/OWL/Ontologies/mad_cows.owl
8. MGED: http://mged.sourceforge.net/ontologies/MGEDOntology.daml
9. Mindswapper: http://www.mindswap.org/2004/owl/mindswappers
10. Pizza: http://www.co-ode.org/ontologies/pizza/pizza_20041007.owl
11. Teams: http://owl.man.ac.uk/2005/sssw/teams
12. Transpotation: http://reliant.teknowledge.com/DAML/Transportation.owl
13. University: http://www.mindswap.org/ontologies/debugging/university.owl
14. Wine: http://www.w3.org/2001/sw/WebOnt/guide-src/wine

Mapping ontologies:

1. BeerMadcow: https://www.cs.tcd.ie/~gsong/ontologies/test/beermadMapping_Test.owl
2. BeerMindswap: https://www.cs.tcd.ie/~gsong/ontologies/test/beermandswapperMapping_Test.owl
3. BeerTransport: https://www.cs.tcd.ie/~gsong/ontologies/test/beertransportmapping_test.owl
4. EconomyBeer:: https://www.cs.tcd.ie/~gsong/ontologies/test/EconomyBeerMapping_Test.owl
5. EconomyCongo: https://www.cs.tcd.ie/~gsong/ontologies/test/EconomyCongoMapping_Test.owl
6. EconomyFood: https://www.cs.tcd.ie/~gsong/ontologies/test/EconomyFoodMapping_Test.owl
7. EconomyMGED: https://www.cs.tcd.ie/~gsong/ontologies/test/EconomyMGEDMapping_Test.owl
8. EconomyTransport: https://www.cs.tcd.ie/~gsong/ontologies/test/EconomyTransportationMapping_Test.owl
9. EconomyWine: https://www.cs.tcd.ie/~gsong/ontologies/test/EconomyWineMapping_Test.owl
10. FoafFood: https://www.cs.tcd.ie/~gsong/ontologies/test/foaffoodMapping_Test.owl
11. FoafPizza: https://www.cs.tcd.ie/~gsong/ontologies/test/foafPizzaMapping_Test.owl

12. FoafUniversity: https://www.cs.tcd.ie/~gsong/ontologies/test/foafuniversityMapping_Test.owl
13. FoodPizza: https://www.cs.tcd.ie/~gsong/ontologies/test/FoodswapPizzaMapping_Test.owl
14. FoodUniversity: https://www.cs.tcd.ie/~gsong/ontologies/test/UniversityFoodMapping_Test.owl

D.2 DL Reasoner Experiment:

1. Galen: http://www.cs.man.ac.uk/~horrocks/OWL/Ontologies/galen.owl
2. Mindswapper: http://www.mindswap.org/2004/owl/mindswappers
3. Wine: http://www.w3.org/2001/sw/WebOnt/guide-src/wine
4. CIM: http://www.dmtf.org/standards/cim/cim_schema_v2101
5. FullCIM Core: http://www.mindswap.org/2004/owl/mindswappers

D.3 XG-policy Distribution Scenario (Section 2.4)

XG ontologies:

1. XG-regn1: https://www.cs.tcd.ie/~gsong/ontologies/test/xgpl-regn1.owl
2. XG-regn2: https://www.cs.tcd.ie/~gsong/ontologies/test/xgpl-regn2.owl
3. Mapping1-2: https://www.cs.tcd.ie/~gsong/ontologies/test/Mapping1-2.owl

D.4 Evaluation (Chapter 7)

Mapping ontologies:

1. Ekaw_Sigkdd: https://www.cs.tcd.ie/~gsong/ontologies/test/Ekawsigkdd.owl
2. Ekaw_PCS: https://www.cs.tcd.ie/~gsong/ontologies/test/EkawPCS.owl
3. Ekaw_paperdyne: https://www.cs.tcd.ie/~gsong/ontologies/test/Ekawpaperdyne.owl
4. Ekaw_OpenConf: https://www.cs.tcd.ie/~gsong/ontologies/test/EkawOpenConf.owl
5. Ekaw_Micro: https://www.cs.tcd.ie/~gsong/ontologies/test/EkawMicro.owl
6. Ekaw_iasted: https://www.cs.tcd.ie/~gsong/ontologies/test/Ekawiasted.owl
7. Ekaw_edas: https://www.cs.tcd.ie/~gsong/ontologies/test/Ekawedas.owl
8. Ekaw_Crs: https://www.cs.tcd.ie/~gsong/ontologies/test/EkawCrs.owl
9. Ekaw_ConfTool: https://www.cs.tcd.ie/~gsong/ontologies/test/EkawConfTool.owl
10. Ekaw_Confious: https://www.cs.tcd.ie/~gsong/ontologies/test/EkawConfious.owl
11. Ekaw_Sofsem: https://www.cs.tcd.ie/~gsong/ontologies/test/EkawSofsem.owl
12. Ekaw_Concus: https://www.cs.tcd.ie/~gsong/ontologies/test/EkawConcus.owl
13. Ekaw_Cmt: https://www.cs.tcd.ie/~gsong/ontologies/test/EkawCmt.owl

E. CPTS FOR BN NODES

This appendix presents the Conditional Possibility Tables (CPTs) of internal nodes of the Bayesian Network for mapping strategy selection, which was developed in Chapter 5.

Table E-1: Conditional Possibility Table of the "MergedOnt_size" Node

OutCome States (MergedOnt_size)			Parent Node Sates		
small	medium	large	Required_ReferOnt	MappingOnt_size	RoutOnt_size
0.8	0.2	0	small	small	small
0	0.7	0.3	small	small	medium
0	0	1	small	small	large
0	0.7	0.3	small	medium	small
0	0.5	0.5	small	medium	medium
0	0	1	small	medium	large
0	0	1	small	large	small
0	0	1	small	large	medium
0	0	1	small	large	large
0	0.7	0.3	medium	small	small
0	0.5	0.5	medium	small	medium
0	0	1	medium	small	large
0	0.5	0.5	medium	medium	small
0	0.4	0.6	medium	medium	medium
0	0	1	medium	medium	large
0	0	1	medium	large	small
0	0	1	medium	large	medium
0	0	1	medium	large	large
0	0	1	large	small	small
0	0	1	large	small	medium
0	0	1	large	small	large
0	0	1	large	medium	small
0	0	1	large	medium	medium
0	0	1	large	medium	large
0	0	1	large	large	small
0	0	1	large	large	medium
0	0	1	large	large	large

Table E-2: Conditional Possibility Table of the "Express_union" Node

OutCome States (Express_union)			Parent Node Sates	
small	medium	large	Required_ReferOnt_express	RoutOnt_express
0.8	0.2	0	small	small
0	0.8	0.2	small	medium
0	0	1	small	large
0	0.8	0.2	medium	small
0	0.6	0.4	medium	medium
0	0	1	medium	large
0	0	1	large	small
0	0	1	large	medium
0	0	1	large	large

Table E-3: Conditional Possibility Table of the "MemoryResources" Node

OutCome States (MemoryResources)			Parent Node Sates	
small	medium	large	MergedOnt_size	AvailableMemory
1	0	0	small	small
0.2	0.8	0	small	medium
0	0.1	0.9	small	large
1	0	0	medium	small
0.5	0.5	0	medium	medium
0.3	0.5	0.2	medium	large
1	0	0	large	small
0.6	0.4	0	large	medium
0.333333	0.333333	0.333333	large	large

Table E-4: Conditional Possibility Table of the "ToleranceCapability" Node

OutCome States (ToleranceCapability)			Parent Node Sates	
small	medium	large	MissMatchedTolerance	FaultTolerance
0.9	0.1	0	small	small
0	0.9	0.1	small	medium
0	0.1	0.9	small	large
0.2	0.7	0.1	medium	small
0	0.5	0.5	medium	medium
0	0.2	0.8	medium	large
0	0.4	0.6	large	small
0	0.2	0.8	large	medium
0	0	1	large	large

Table E-6: Conditional Possibility Table of the "MatchMsgRate" Node

OutCome States (MatchMsgRate)			Parent Node Sates		
small	medium	large	SubRate	PubRate	UnSubRate
0.5	0.5	0	small	small	small
0.6	0.4	0	small	small	medium
0.7	0.3	0	small	small	large
0.5	0.5	0	small	medium	small
0.6	0.4	0	small	medium	medium
0.7	0.3	0	small	medium	large
0.2	0.3	0.5	small	large	small
0.3	0.25	0.45	small	large	medium
0.4	0.2	0.4	small	large	large
0.5	0.5	0	medium	small	small
0.6	0.4	0	medium	small	medium
0.7	0.3	0	medium	small	large
0.3	0.7	0	medium	medium	small
0.4	0.6	0	medium	medium	medium
0.6	0.4	0	medium	medium	large
0	0.5	0.5	medium	large	small
0.2	0.4	0.4	medium	large	medium
0.3	0.4	0.3	medium	large	large
0.2	0.3	0.5	large	small	small
0.3	0.25	0.45	large	small	medium
0.4	0.2	0.4	large	small	large
0	0.5	0.5	large	medium	small
0.2	0.4	0.4	large	medium	medium
0.3	0.4	0.3	large	medium	large
0	0	1	large	large	small
0.1	0.3	0.6	large	large	medium
0.2	0.3	0.5	large	large	large

Table E-5: Conditional Possibility Table of the "LoadtimeOverhead" Node

OutCome States (LoadtimeOverhead)			Parent Node Sates		
small	medium	large	MergedOnt_size	Express_union	NumOfStoredOnts
1	0	0	small	small	small
0.8	0.15	0.05	small	small	medium
0.5	0.3	0.2	small	small	large
0.9	0.1	0	small	medium	small
0.75	0.2	0.05	small	medium	medium
0.45	0.25	0.3	small	medium	large
0.8	0.2	0	small	large	small
0.6	0.3	0.1	small	large	medium
0.2	0.4	0.4	small	large	large
0	1	0	medium	small	small
0	0.7	0.3	medium	small	medium
0	0.6	0.4	medium	small	large
0	0.9	0.1	medium	medium	small
0	0.65	0.35	medium	medium	medium
0	0.55	0.45	medium	medium	large
0	0.85	0.15	medium	large	small
0	0.6	0.4	medium	large	medium
0	0.52	0.48	medium	large	large
0	0.2	0.8	large	small	small
0	0.15	0.85	large	small	medium
0	0.1	0.9	large	small	large
0	0.15	0.85	large	medium	small
0	0.1	0.9	large	medium	medium
0	0.05	0.95	large	medium	large
0	0.02	0.98	large	large	small
0	0.01	0.99	large	large	medium
0	0	1	large	large	large

E.2

F. ADDITIONAL EVALUATION RESULTS

This appendix presents the detailed information of several experiments that are not described in Chapter 7. However, they provide the supportive information for the evaluation chapter.

F.1 Subscription processing Test

In order to help illustrating the process of publication matching in the KBNMap deployments in Chapter 7, it is worthwhile to understand the situation of mapping strategies executed in the routers. This Appendix describes the situation of subscription processing operated by the routers, when they are handling subscriptions. This process can be described as follows:

Once a router receives a new subscription, it first examines whether this subscription is already existed in its subscription tree or not. If the subscription is purely new one, the router second examines whether this subscription not covered by an existing subscription in the tree. If it is not, the router inserts it in the subscription tree; otherwise, the subscription will be stored firstly then is forwarded to the router's parent router, which may propagate up the router hierarchy in the deployment described in Chapter 7. If it get passed to the leaf nodes parent, then it will propagate to the root since the subscription cannot be covered by any other subscriptions from elsewhere in the network as there exists only one subscriber.

As it is the supplementary experiment for the experiment in Chapter 7, the application scenario, setup and experiment running environment is the same as the one in Section 7.7. Due to the principle of forwarding subscription in KBNImpl (forwarding the most general subscription to the master router), some routers in our network topology (see Figure 7-1) may not receive all the subscriptions forwarded by the subscription client. Therefore, in this experiment, we only measure the subscription processing in subscription-leaf router (the leaf router on the right side in Figure 7-1). The following subsections provide detailed discussion of each individual strategy executed by the leaf router in different scenarios.

F.1.1 The "Every Mapping Files" Strategy

Once the first unknown subscription arrives in the router, the **Every** strategy will be used only once to allow the router to load all the mapping files and reference ontologies stored in the ontology mapping store. As all unknown ontological terms contained in the subscriptions are derived from the reference ontologies, triggering the **Every** strategy once will permanently resolve the unknown data problem, which makes it unnecessary to measure the entire working process of the router. In order to comprehensively understand the situation of **Every** strategy executed in subscription processing, it was decided to run the test for twenty times to record

the frequency of **Every** strategy being used in subscription process rather than to test the time required for the subscription processing.

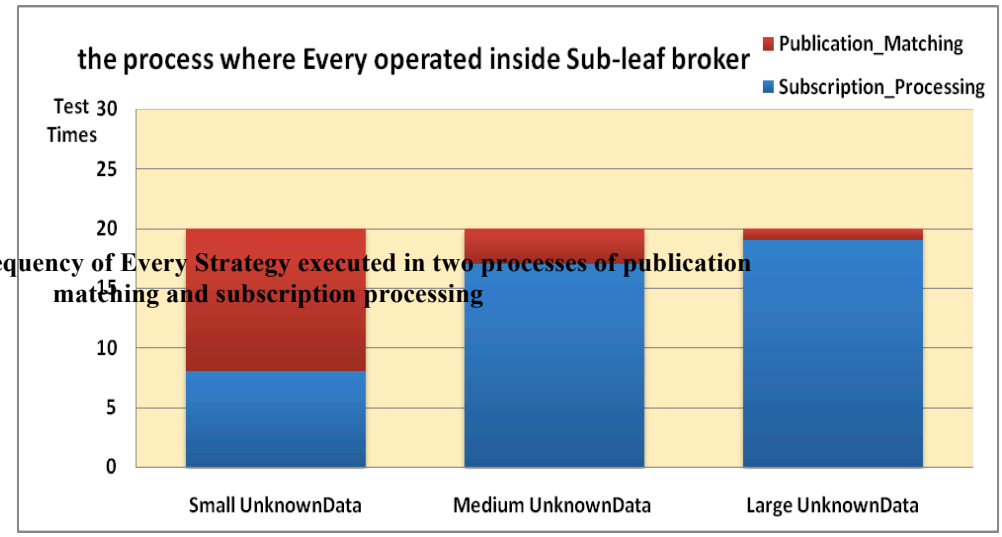


Figure F-1: the Frequency of Every Strategy executed in two processes of publication matching and subscription processing

Figure F-1 demonstrates the number of times the **Every** strategy executed in two processes inside the leaf router in three different unknown data cases. In the case of *small* unknown data, it can be observed that for twenty running tests, the **Every** strategy is used by the router to deal with unknown subscription in the process of subscription processing for eight times while it is used for dealing with unknown publication in the process of publication matching for 12 times. It is worthwhile to note that for twenty tests, the unknown subscription or publication randomly arrives at the leaf router. In other words, it is unpredictable to tell unknown data is received by the leaf router in a subscription or in a publication. In the case of *medium* unknown data, it can be observed that for twenty running tests, **Every** strategy is used by the router to deal with unknown subscription in the process of subscription processing for seventeen times while there are only three times that **Every** strategy is used in the publication matching. As for *large* unknown data case, **Every** strategy is triggered nineteen times in subscription processing for dealing with unknown subscription while only triggered once in the process of publication matching.

The results derived from Figure F-1 confirmed our previous hypothesis in Section 7.8 that the **Every** strategy is prone to be executed by KBN deployment in subscription processing rather than publication matching in the environment where the rate of unknown data occurrence is *large*.

F.1.2 The "Appropriate Mapping File Only" Strategy

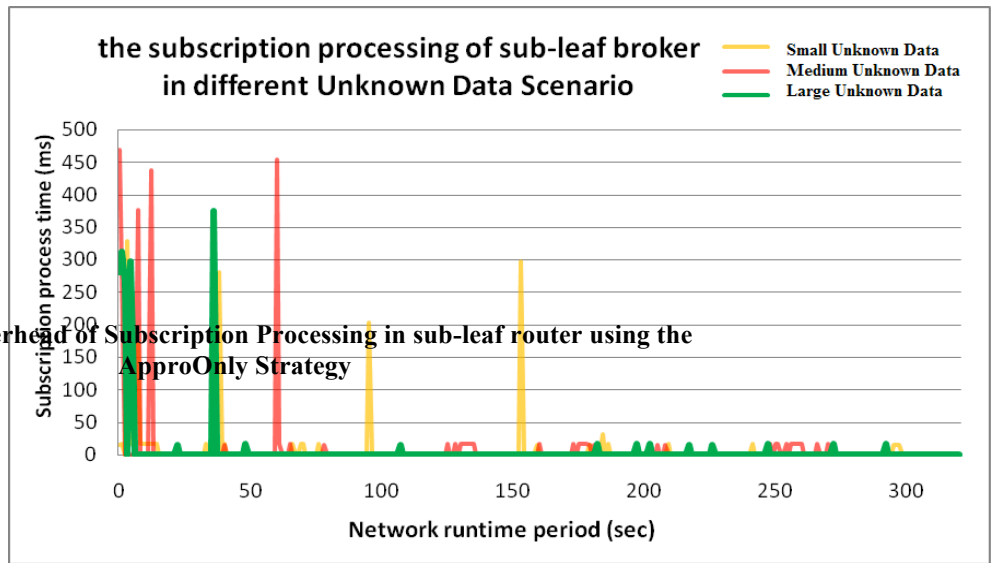


Figure F-2: the Overhead of Subscription Processing in sub-leaf router using the ApproOnly Strategy

Figure F-2 demonstrates that overhead introduced by processing subscriptions in sub-leaf router when it using the **ApproOnly** strategy**.** For small unknown data scenario, it can be observed that three relatively expensive extraprocessing required for subscription processing occurred, indicating that the router encountered an unknown subscription and used the the **ApproOnly** strategy to load the appropriate mapping file to deal with the unknown data. Whilst in medium/high unknown data scenario, executing the **ApproOnly** strategy frequently earlier on in the experiment indicated that the possibility of encountering the unknown subscriptions in this case is more frequently than small unknown data case. Therefore, it loads the appropriate mapping files more frequently by using **ApproOnly** strategy. These results can also confirm that our analysis of publication matching in Section 7.8: where (in publication matching or subscription processing) the **ApproOnly** being triggered highly depend on whether the router encounters the unknown subscription or publication.

F.1.3 The "ApproInd" Strategy

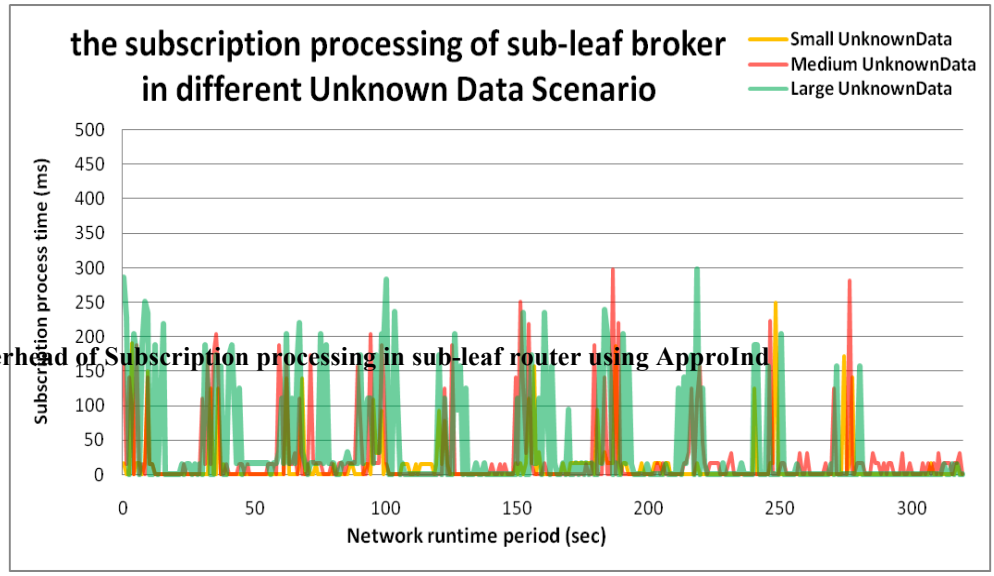


Figure F-3: the Overhead of Subscription processing in sub-leaf router using ApproInd

Figure F-3, demonstrates the time consumed by subscription processing in sub-leaf router using **ApproInd** strategy in three different unknown data rate scenarios. Again, the relatively expensive overhead indicates that the router is executing **ApproInd** to deal with the unknown subscription. The results showing in this graph associated with the publication matching performance of KBN_ApproInd in Section 7.8 confirms that where (in publication matching or subscription processing) the **ApproInd** strategy being triggered highly depend on what type of unknown data the router encounter firstly.

F.2 Frequency of Mapping Strategy Usage

Table 0-1: Frequency of Mapping Strategy Usage in Three Categorised Testing Sets

Case Name	Frequency of Mapping Strategies Used		
	The "Every" Strategy	*The "ApproOnly" Strategy*	*The "ApproInd" Strategy*
Testing Set A			
1.small unknown data rate with small message rate	0	0	146
2.medium unknown data rate with medium message rate	0	0	269
3.large unknown data rate with large message rate	0	96	0
Testing Set B			
1.small unknown data rate with low tolerance	0	0	150
2.small unknown data rate with medium message tolerance	0	0	149
3.small unknown data rate with large tolerance	0	0	156
4.medium unknown data rate with low tolerance	0	0	306
5.medium unknown data rate with medium message tolerance	0	0	297
6.medium unknown data rate with large tolerance	0	0	316
7.large unknown data rate with low tolerance	0	96	0
8.large unknown data rate with medium message tolerance	0	96	0
9. large unknown data rate with large tolerance	0	0	747
Testing Set C			
1.small unknown data rate with low memory resources	0	0	145
2.small unknown data rate with medium memory resources	0	0	147
3.small unknown data rate with large memory resources	0	0	152
4.medium unknown data rate with low memory resources	0	0	289
5.medium unknown data rate with medium memory resources	0	96	0
6.medium unknown data rate with large memory resources	0	96	0
7.large unknown data rate with low memory resources	0	96	0
8.large unknown data rate with medium memory resources	8	0	0
9.large unknown data rate with large memory resources	8	0	0

F.3 The KBNMap Performance

This subsection presents the detailed description of tests in the Testing sets which are discussed in Chapter 7.

F.3.1 Testing Set A: Unknown Data Rate with Message Rate

Case 1: Small Unknown Data Rate with Small Message Rate

As a representative case, the performance of two networks is evaluated in the scenario where the rate of messages across the network remains a small (5 messages/min) level while the proportion of unknown data contained in the messages is also relatively small (2 messages/min). All other trigger factors were set to default high-load configuration (Table 7-2). With these inputs the Bayesian Network of KBNMap produced a weighting of 0.47% for the **Every** strategy, 25.60% for the **ApproOnly** strategy, and 74% for the **ApproInd** strategy.

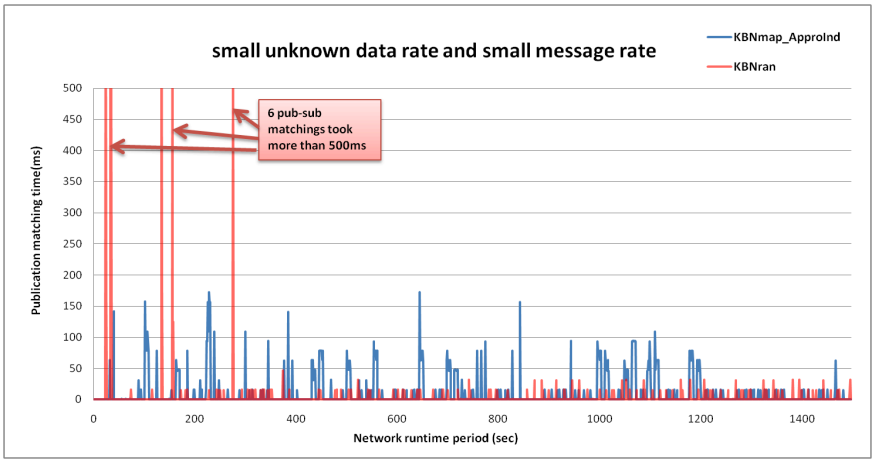


Figure F.3-A-1: Pub Matching Time: small unknown data with small message rate

As unknown data rarely occurs in this scenario and also the memory resources allocated in the network remains a small level, it is clear from Figure F.3-A-1 that KBNRan deployment incorrectly executes the **Every** strategy at the earlier stage of the test, resulting in very high publication matching times (above 500 ms). However, once all the mapping files were loaded in most of the routers the KBNran deployment exhibits efficient performance. In contrast, the KBNMap deployment selected the **ApproInd** strategy that drives the routers to load and merge a single mapping into the **routing ontology** rather than the entire mapping file, which leads KBNMap routers to gradually resolve individual unknown data on an individual basis.

As for the end-to-end delivery of the subset of the publications that matched a subscription and so were routed across the entire network over 15 hops. From the diagram Figure F.3-A-2, it is clear that KBNMap implementation substantially outperforms the KBNRan implementation with respect to delivering publications. In addition, due to the ApproInd

strategy being only selected in KBNMap, the performance of KBNMap is much alike as the KBNFix deployment: KBNapproInd in Section 7.4.2.1.

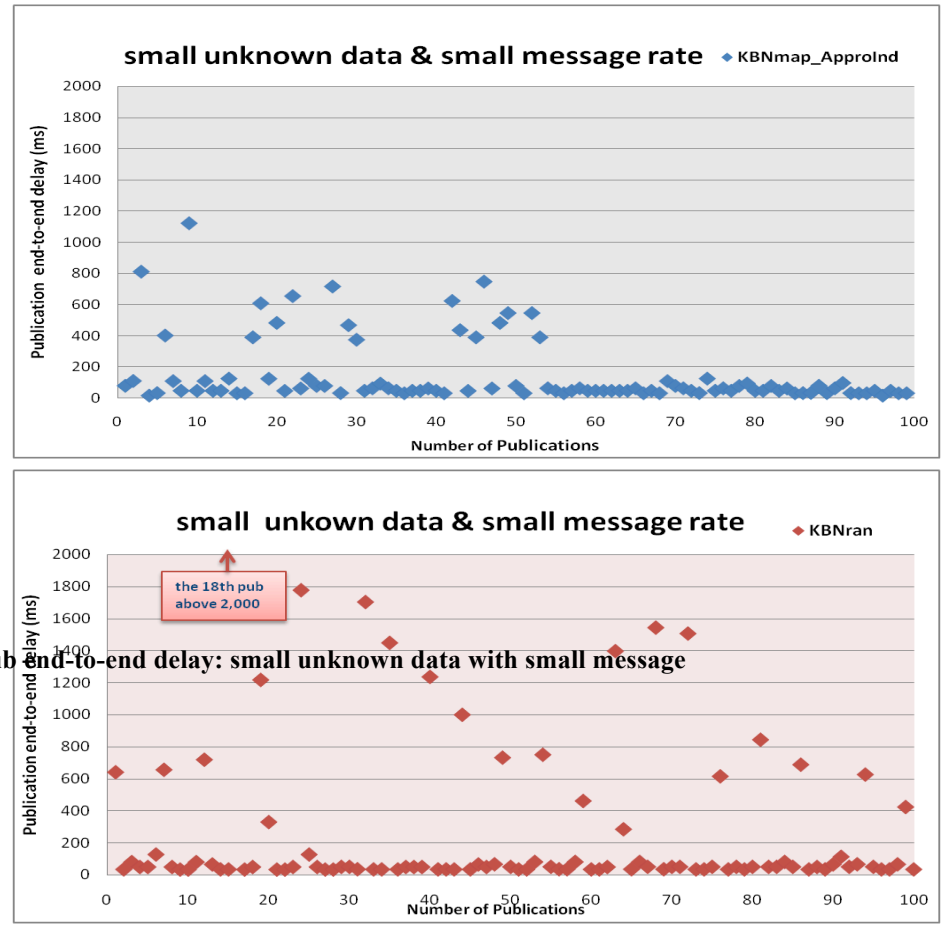


Figure F.3-A-2: Pub end-to-end delay: small unknown data with small message

Case 2: Medium Unknown Data Rate with Medium Message Rate

This case examines the network deployments in the situation where rates of both message and unknown data raised to a medium level. These two inputs combining with the inputs of the other factors (Table 7-2), the Bayesian Network of KBNMap produced a weighting of 3% for the **Every** strategy, 43 % for the **ApproOnly** strategy, and 54% for the **ApproInd** strategy.

It is clearly demonstrated that loading semantic matches during publication processing for KBNMap deployment are gradually increased in comparison with case 1 (F.3-A-3). This is because KBNMap deployment executed the **ApproInd** strategy more frequently to handle the increased rate of unknown data occurrence. On the other hand, the KBNRan implementation executed the **Every** strategy early on in several routers, and continued to exhibit poor performance at random intervals during the experiment as the remaining routers also executed the **Every** strategy or **ApproOnly** Strategy.

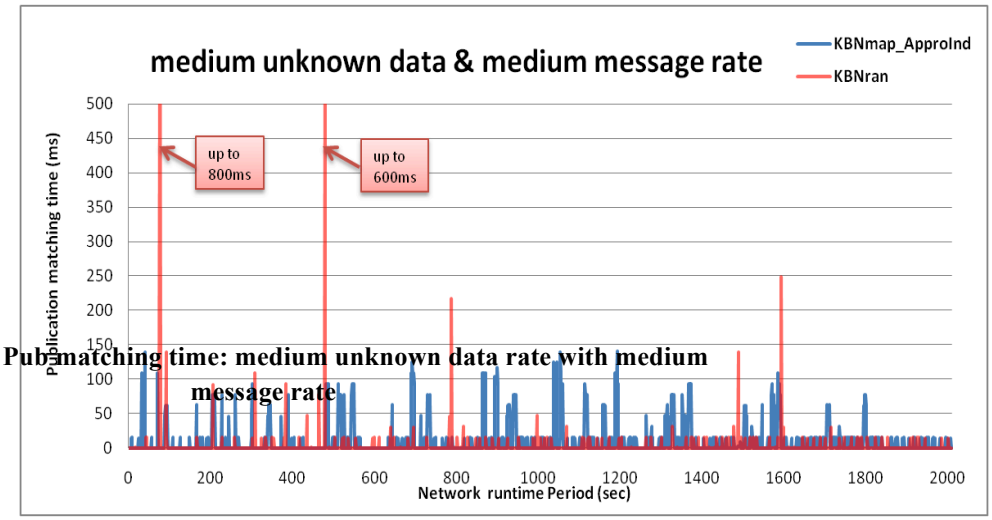


Figure F.3-A-3: Pub matching time: medium unknown data rate with medium message rate

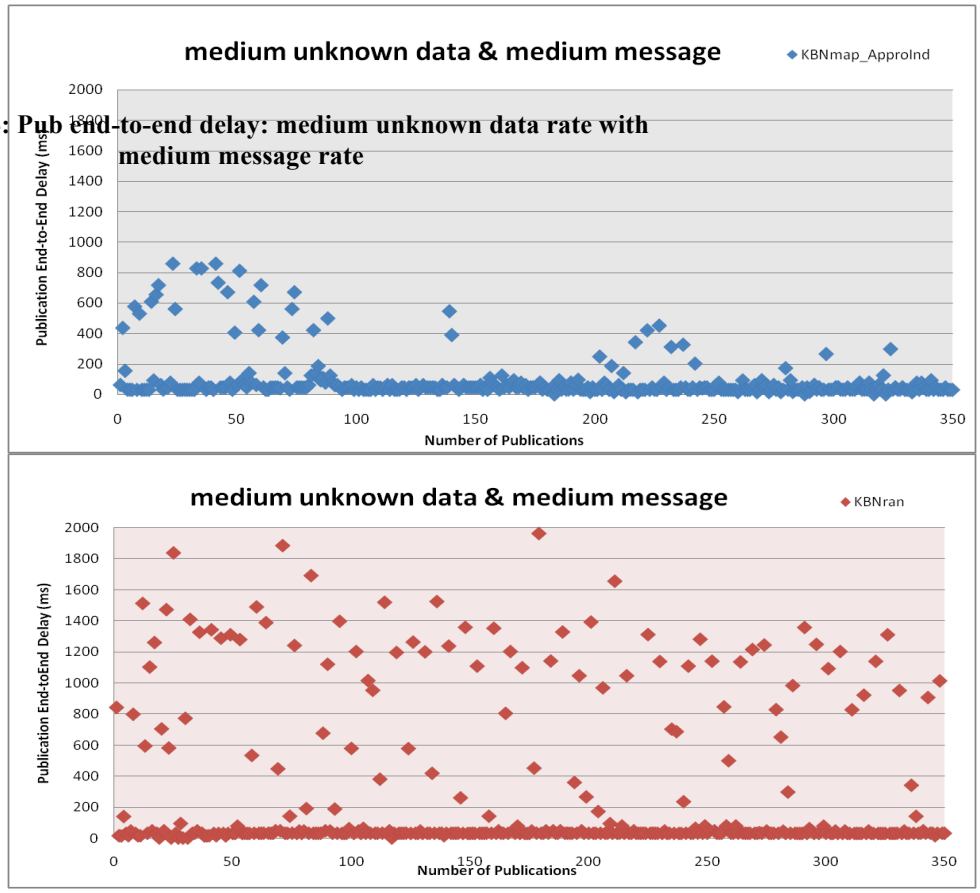


Figure F.3-A-4: Pub end-to-end delay: medium unknown data rate with medium message rate

With respect to the performance of end-to-end publication delivery (Figure F.3-A-4), the results demonstrate that KBNMap deployment exhibit more efficient performance than KBNRan deployment even in the situation where the message rate is increased to medium level with more unknown data occurrences. Again the performance of KBNMap is similar to KBNapproInd implementation (Section 7.4.2.2) as the **ApproInd** strategy was only used in

KBNMap routers. For KBNran, its inefficient performance can be easily explained that the mapping strategies are disorderly executed by the routers.

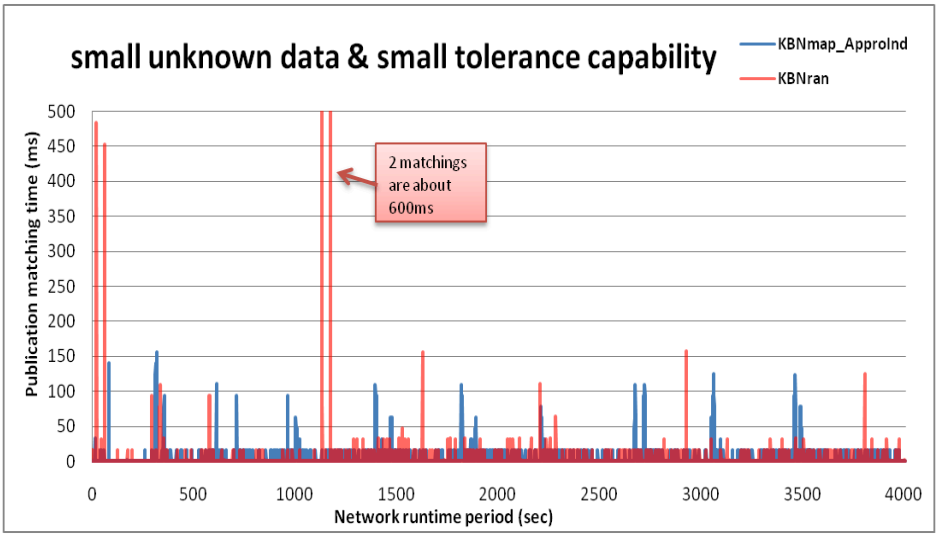
F.3.2 Testing Set B: Unknown Data Rate with Mismatch Tolerance

Case 1 of 9: Small Unknown Data Rate with Small Tolerance

In this case we evaluated the performance of two networks in the scenario where the rate of unknown data contained in the messages is relatively small (2 messages/min) and the mismatch tolerance was small. All other trigger factors were set to default high-load configuration (Table 7-3). With these inputs the Bayesian Network of KBNMap produced a weighting of 2% for the **Every** strategy, 27% for the **ApproOnly** strategy, and 71% for the **ApproInd** strategy.

Figure F.3-B-1: Pub matching time: small unknown data rate with small tolerance

Figure F.3-B-1 presents the publication matching time for each network. The response times for matching in KBNMap are relatively small and range from under 30ms to just over 100ms, depending on when the network occasionally encounter unknown data. The matching overhead consumed by KBNRan routers is still unpredictable and varies from very small value (under 30ms) to large value (a bove 600 ms).



With respect to the end-to-end delay times for distributing publications, it is verified again from Figure F.3-B-2 that the KBNMap network substantially performs far superior to the KBNRan deployment for delivering the publications that matched a subscription across the network to the subscriber, as the KBNMap implementations performs smoothly and stead for delivering large number of publications with small portion of unknown data.

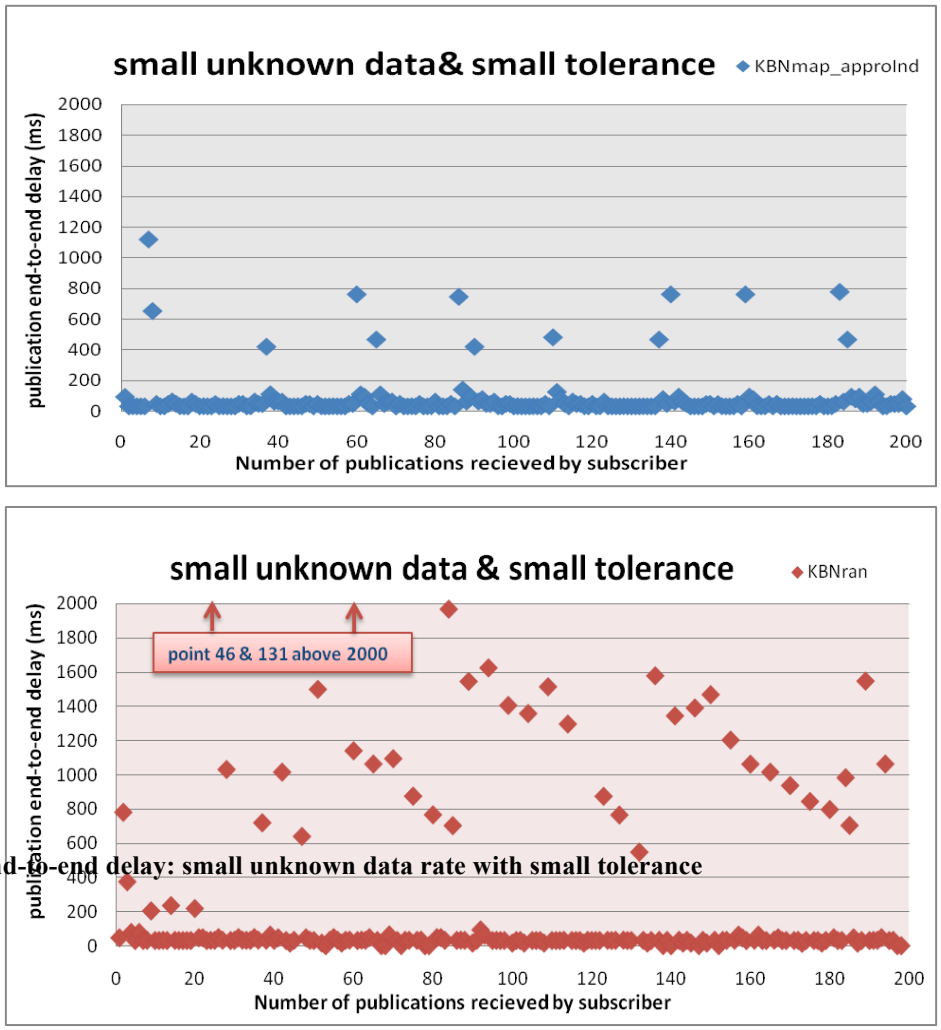


Figure F.3-B-2: Pub end-to-end delay: small unknown data rate with small tolerance

Case 3 of 9: Small Unknown Data Rate with Large Tolerance

In this case we evaluated the combinations of two factors: the rate of unknown data contained in the messages is still relatively small (2 messages /min) but the mismatch tolerance was set to high level in this case. All other trigger factors were set to default high-load configuration (Table 7-3). With these inputs the Bayesian Network of KBNMap produced a weighting of 1% for the **Every** strategy, 7% for the **ApproOnly** strategy, and 92% for the **ApproInd** strategy.

The empirical results in this scenario show that KBNMap performs relatively efficiently for both publication matching (Figure F.3-B-3) and publication delivery (Figure F.3-B-4) in the situation where the rate of messages arrival is high but the unknown data is rare. Thus, it can be concluded that the **ApproInd** strategy is well-suited in such a situation if the high level of tolerance is assigned to the applications. When taking into account the strategy selection aspect, it is also confirmed that KBNMap has made an appropriate decision of selecting the **ApproInd** strategy, since it enables routers to use individual mappings to resolve unknown

data, requiring the minimum resources. If taking insight into the adaptability of the KBNMap, the Bayesian Network automatically increased the weightings of the **ApproInd** strategy in three cases, indicating the KBNMap deployment is capable of tailoring to the changes of mismatch tolerance level.

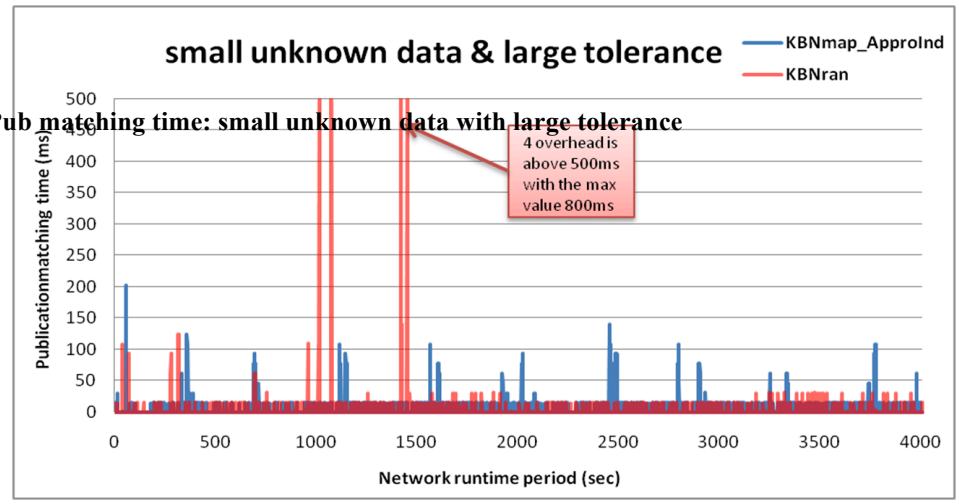
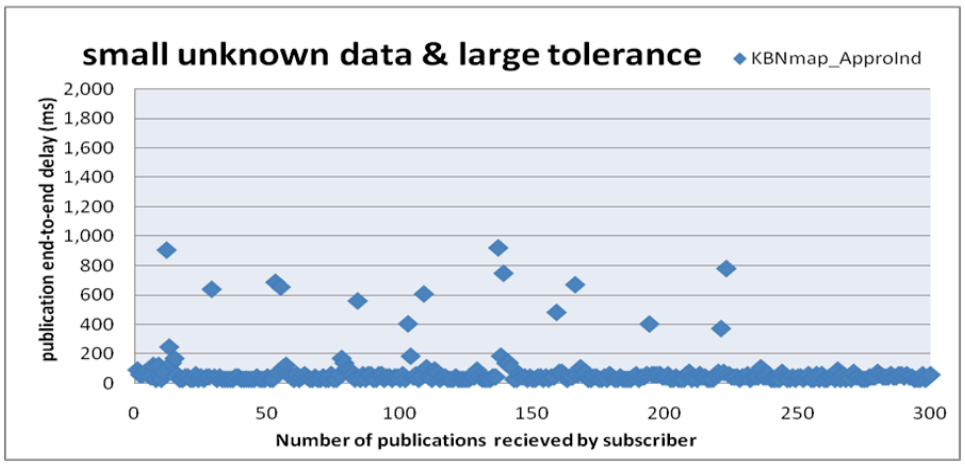
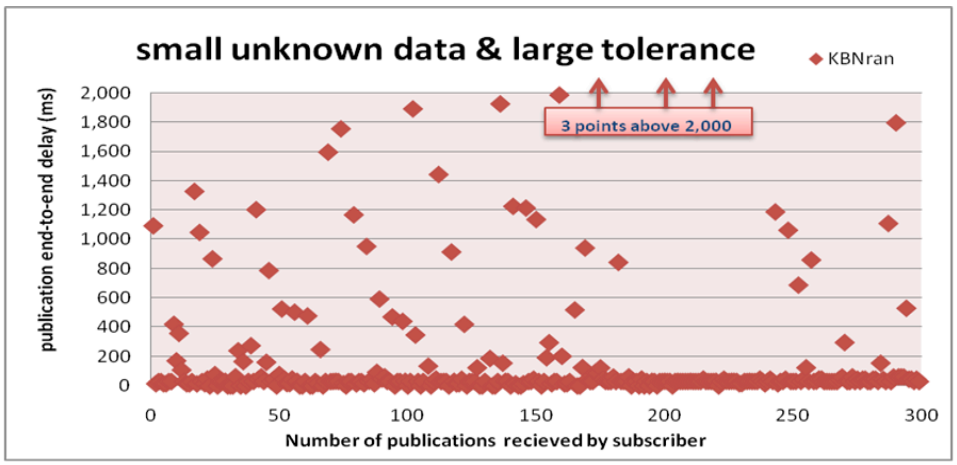


Figure F.3-B-3: Pub matching time: small unknown data with large tolerance

Figure F.3-B-4: Pub end-to-end delay: small unknown data rate with large tolerance

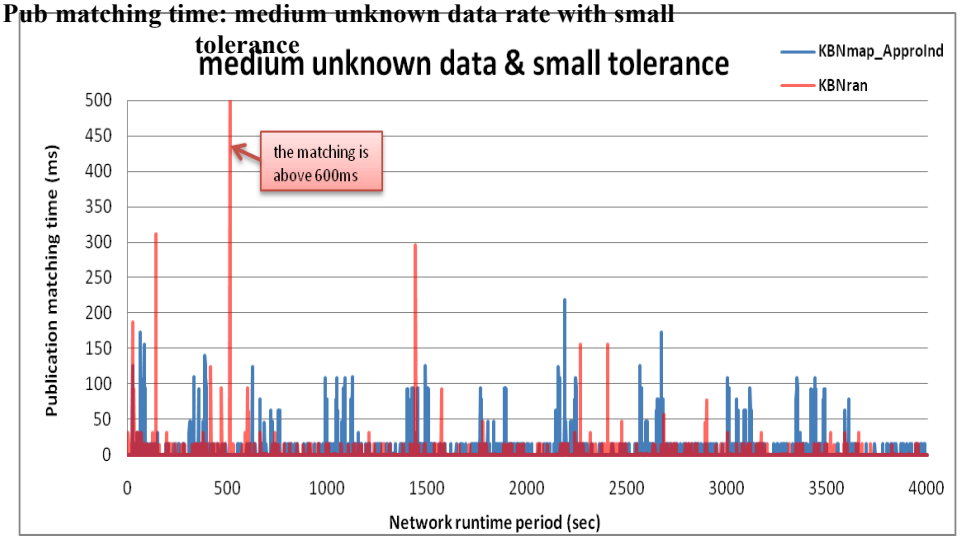




Case 4 of 9: Medium Unknown Data Rate with Small Tolerance

In this case we evaluated the combinations of two factors: the rate of unknown data contained in the messages is increased to a medium level (5 messages /min) but the mismatch tolerance was set back to low level in this case. All other trigger factors were set to default high-load configuration (Table 7-3). With these inputs the Bayesian Network of KBNMap produced a weighting of 4% for the **Every** strategy, 44% for the **ApproOnly** strategy, and 52% for the **ApproInd** strategy.

Figure F.3-B-5: Pub matching time: medium unknown data rate with small tolerance



The evaluated results of publication matching (Figure F.3-B-5) indicate that KBNMap routers continuously selected the **ApproInd** strategy in the medium unknown data situation, resulting in slightly higher publication matching times. Keeping choosing the **ApproInd** strategy by the Bayesian Network is primarily driven by the limited (small) memory resources allocated in the KBNMap routers. However, the updated weightings of mapping strategies indicated that the BN model has dynamically adapted to the changes of the unknown data occurrence. By comparison the KBNRan implementation alternatively executed the **Every** and **ApproOnly** strategies at random intervals during the testing, exhibiting a certain level of efficiency for processing publications.

On the other hand, it is clear that the KBNMap deployment is evaluated far more efficiently than the KBNRan deployment with respect to distributing a big subset of publications across the entire network over 15 hops (Figure F.3-B-6), as the **ApproInd** strategy that consecutively used in KBNMap only requires small resources for execution. By comparison, the poor performance of KBNran is due to that the strategies executed seriously diminished memory resources for continuously loading mapping files and merging *referenced* ontologies, resulting in inadequate resources used for delivering publications.

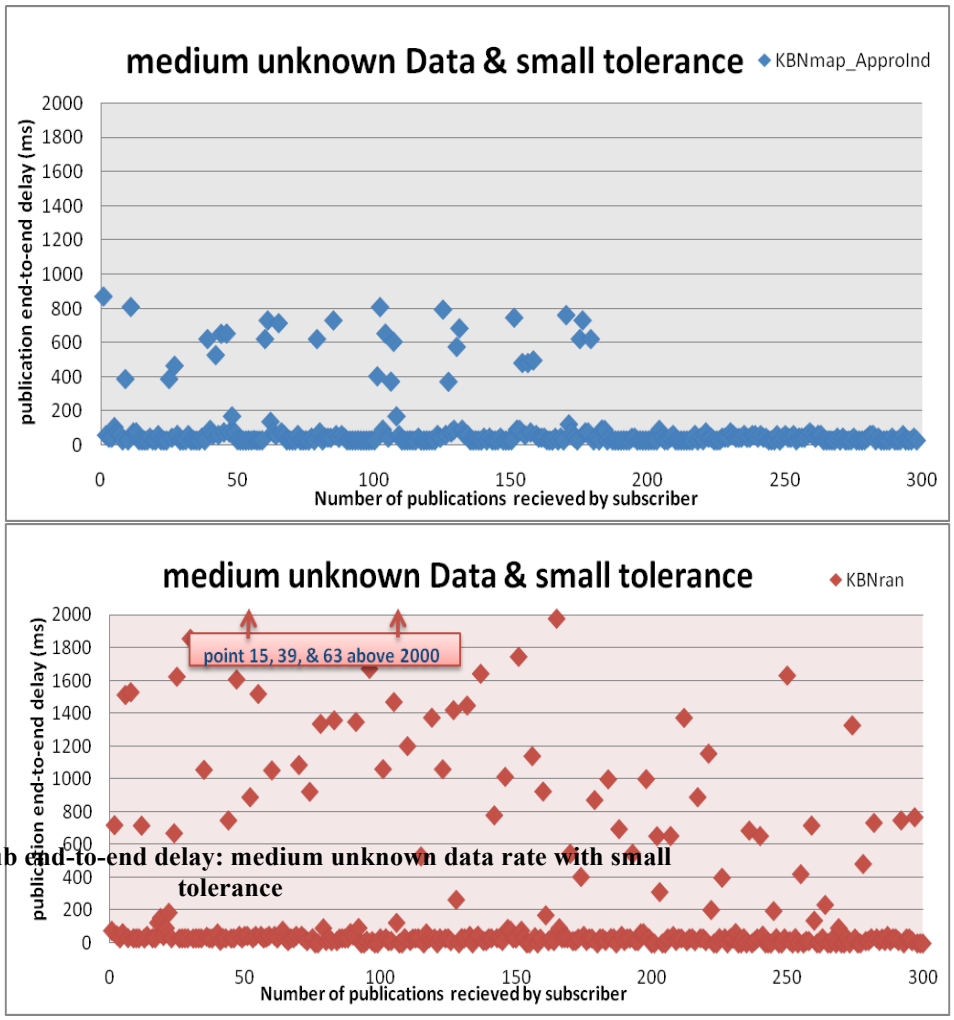


Figure F.3-B-6: Pub end-to-end delay: medium unknown data rate with small tolerance

Case 5 of 9: Medium Unknown Data Rate with Medium Tolerance

In this case we compared the KBNMap deployment with the KBNRan deployment in the scenario of where the rate of unknown data still remains a medium level (5 messages /min) but the mismatch tolerance was raised to medium level. With the inputs of selected trigger factors the Bayesian Network automatically updated a weighting of 3% for the **Every** strategy, 42% for the **ApproOnly** strategy, and 55% for the **ApproInd** strategy.

It is clear from (Figure F.3-B-7, Figure F.3-B-8) that the KBNMap deployment performed as similarly as it did in the previous case due to no changes of unknown data occurrence. In addition, the increased weighting of the **ApproInd** strategy indicated that the KBNMap implementation is more likely to choose this strategy, as the applications are more tolerant to the publication mismatch due to potentially missed semantic relations due to the overly conservative loading of mappings. In contrast, the KBNRan deployment incorrectly executed the **Every** strategy for matching publications early on, resulting in insufficient resources left.

This significantly influenced the performance of KBNRan with respect to distributing publications across the network to the destinations.

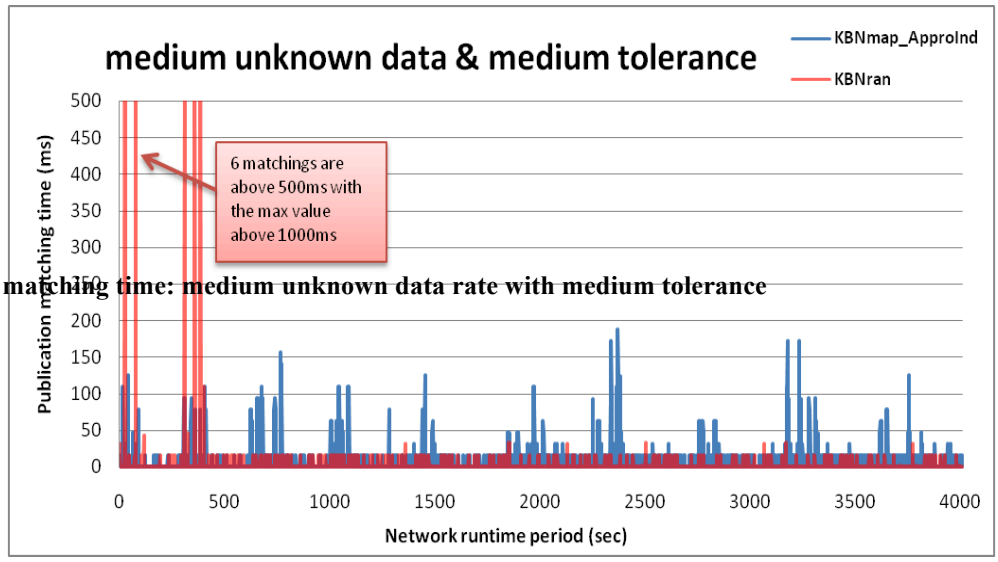
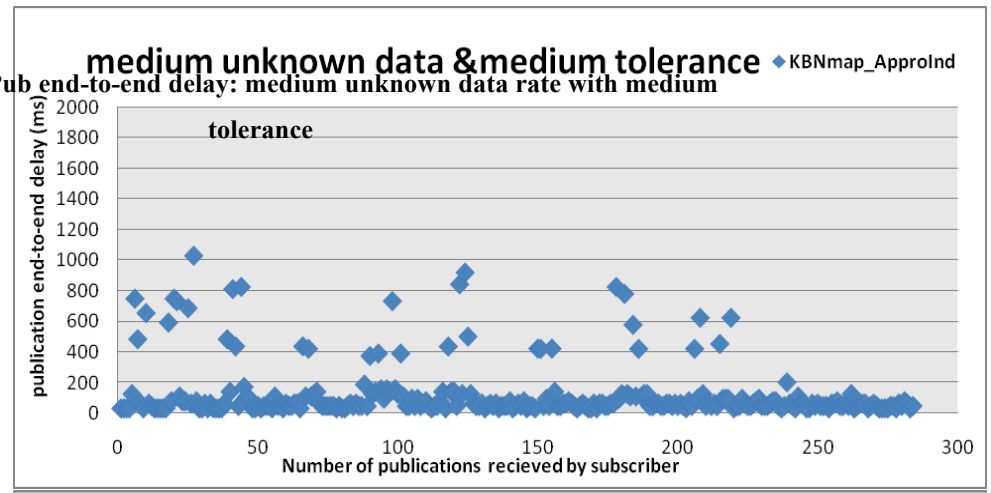


Figure F.3-B-7: Pub matching time: medium unknown data rate with medium tolerance



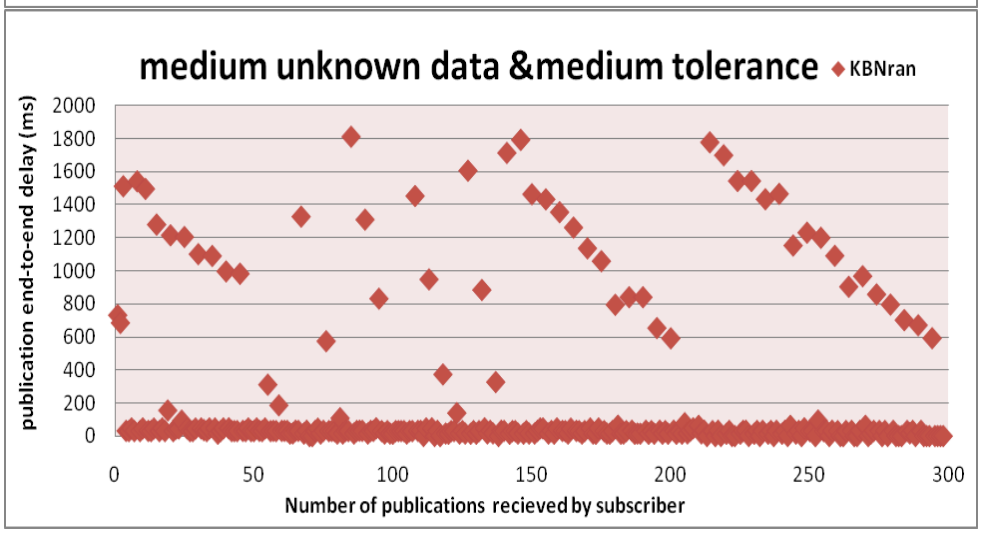


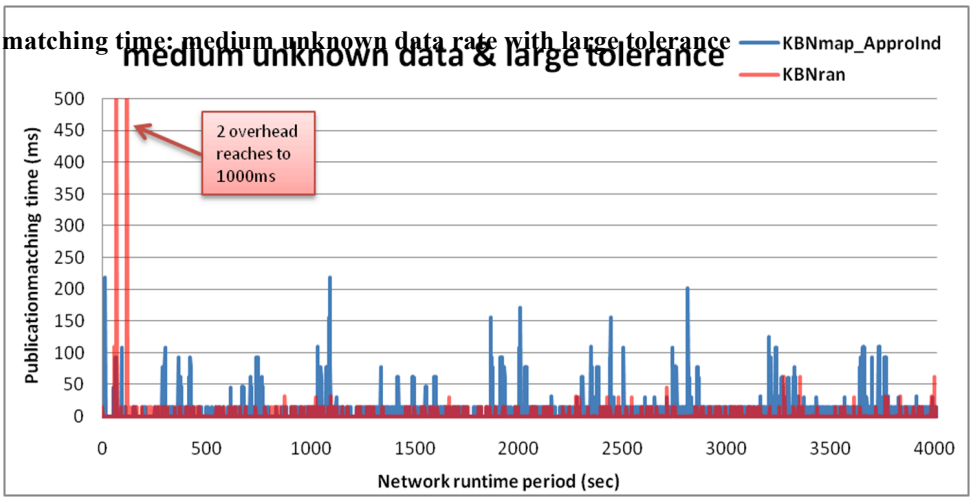
Figure F.3-B-8: Pub end-to-end delay: medium unknown data rate with medium

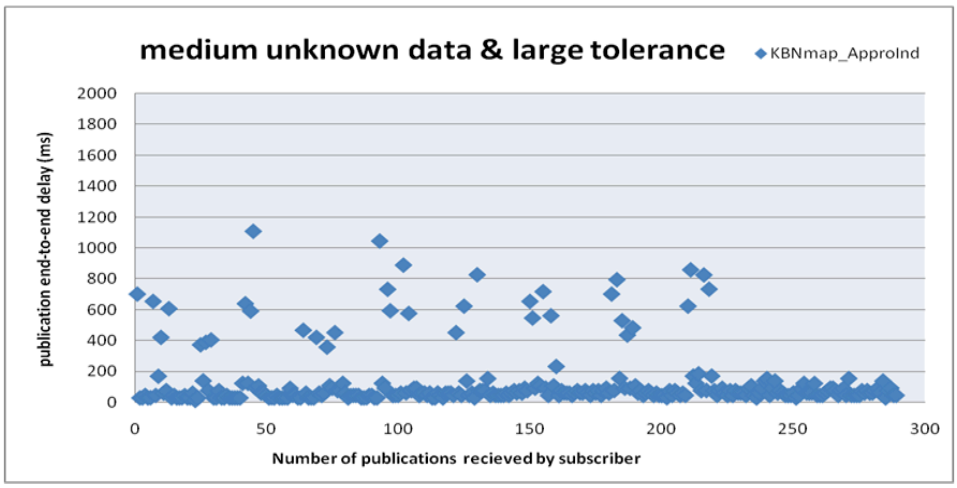
Figure F.3-B-10: Pub end-to-end delay: medium unknown data rate with large tolerance

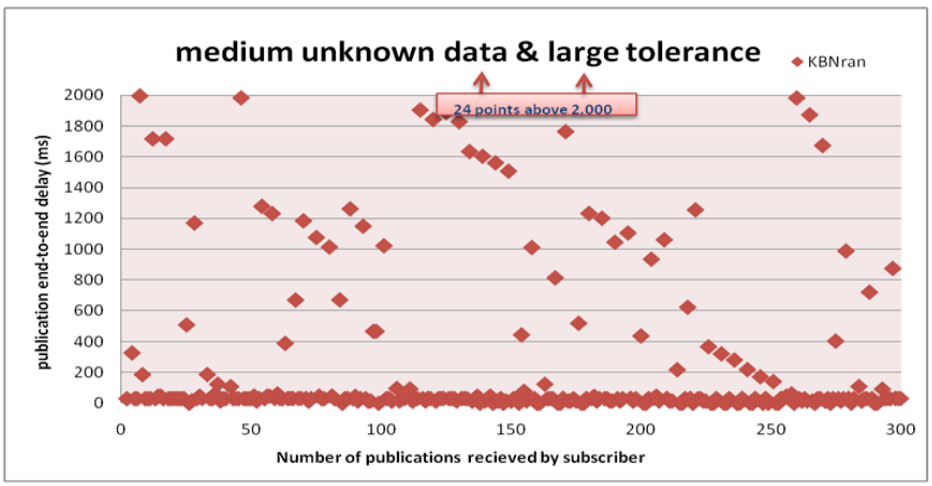
Case 6 of 9: Medium Unknown Data Rate with Large Tolerance

This case presents the scenario where the unknown data rate still remained a medium level (5 messages /min) while mismatch tolerance was set up to high level. With the inputs of the selected trigger factors, the Bayesian Network updated a weighting of 1% for the **Every** strategy, 24% for the **ApproOnly** strategy, and 75% for the **ApproInd** strategy.

Figure F.3-B-9: Pub matching time: medium unknown data rate with large tolerance





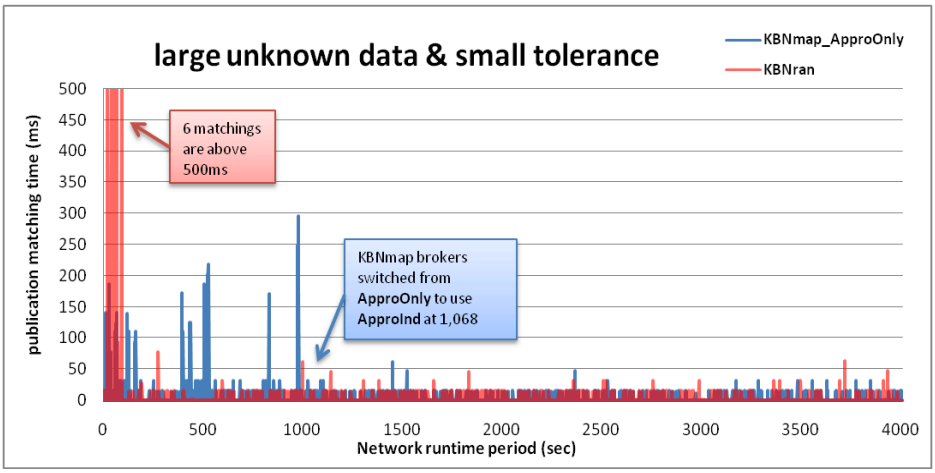


The results of both publication match and delivery (Figure F.3-B-9, Figure F.3-B-10) clearly demonstrated that the KBNMap network performed similar as the previous two cases, exhibiting a stable performance for dealing with the medium rate of unknown data occurrence. The increased weighting of the **ApproInd** strategy indicated that the KBNMap deployment is more confident to use this strategy with the updated level of mismatch tolerance. On the other hand, even though the several KBNRan routers executed the **Every** strategy early on, benefiting this implementation performed efficiently for matching publications later, again, the **Every** strategies heavily diminished memory resources for distributing the publications.

Figure F.3-B-11: Pub matching time: large unknown data rate with small tolerance

Case 7 of 9: Large Unknown Data Rate with Small Tolerance

This case presents a scenario where the rate of unknown data occurrence was increased to a high level (12 messages /min) but the mismatch tolerance of applications was set back to the low level. All other factors were set to default high-load setting as before. With these inputs the Bayesian Network updated a weighting of 42% for the **Every** strategy, 57% for the **ApproOnly** strategy, 1% for the **ApproInd** strategy.



It can be observed from (Figure F.3-B-11) that the KBNMap deployment chose the **ApproOnly** strategy instead, resulting in slightly high end-to-end publication matching times. However, once the appropriate mapping file was loaded in most of the routers the KBNMap performed efficiently. Switching to the **ApproOnly** strategy also indicated that the KBNMap depolyment dynamically tailored to the changes of unknown data occurrence. In addition, at approx. 1000 seconds into the experiment, most of KBNMap routers switched to the **ApproInd** strategy for efficiently using resources. In contrast, the KBNRan deployment also exhibit efficient performance in the experiment after executing the **Every** strategy.

In terms of end-to-end publication delivery delay (Figure F.3-B-12), the KBNMap network is shown that it still performed more efficiently than the KBNRan deployment. However, the

slightly worse performance of KBNMap can be explained that the **ApproOnly** strategy consumed more memory resources to allow the routers to load the entire mapping file, which slightly affect the operation of delivering publications.

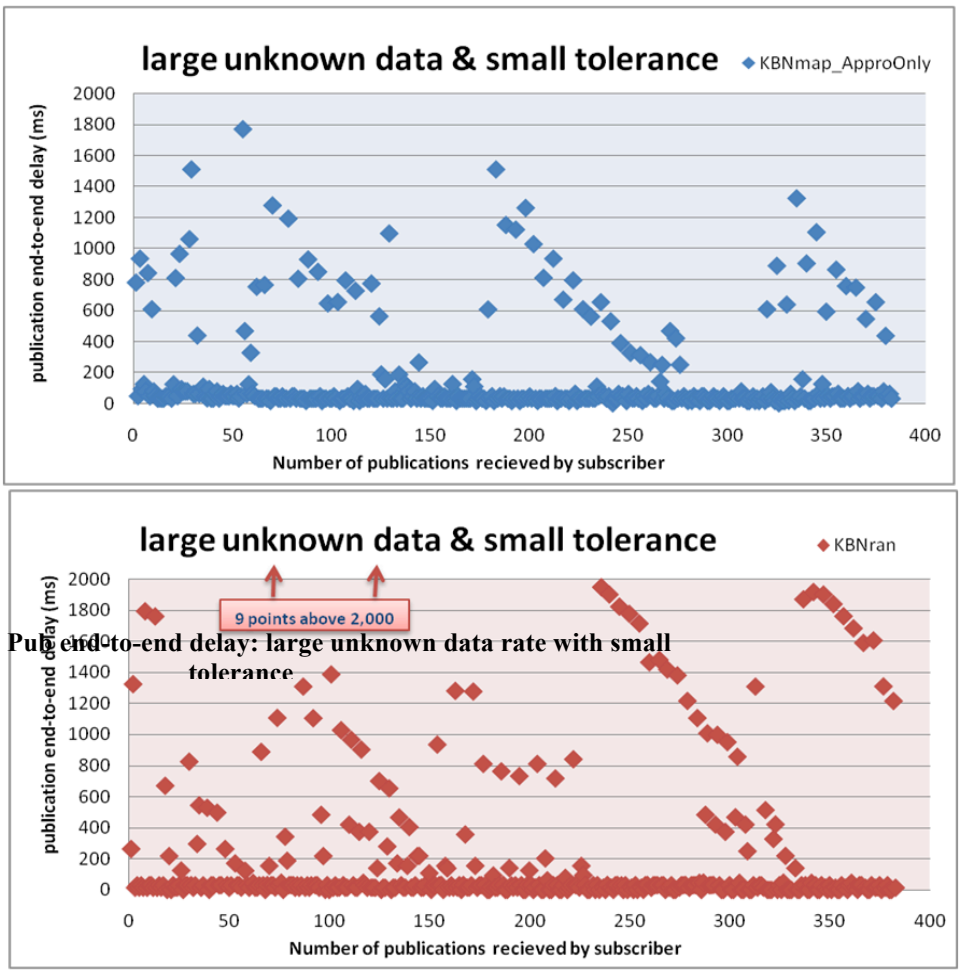


Figure F.3-B-12: Pub end-to-end delay: large unknown data rate with small tolerance

Case 8 of 9: Large Unknown Data Rate with Medium Tolerance

This case presents a scenario where the rate of unknown data occurrence still remained a high level (12 messages /min) but the mismatch tolerance of applications was raised to the medium level. All other factors were set to default high-load setting as before. With these inputs the Bayesian Network updated a weighting of 32% for the **Every** strategy, 50% for the **ApproOnly** strategy, 18% for the **ApproInd** strategy.

The empirical results observed from the publication matching (Figure F.3-B-13) and publication delivery (Figure F.3-B-14) demonstrated that the KBNMap deployment performed similarly as it did in the previous case. In addition, the decreased weighting of the **ApproOnly** strategy indicated that the KBNMap has dynamically adapted to the increased applications tolerance level. By comparison the efficient operation for processing publication matches indicated that the KBNRan implementation randomly executed the strategies in the

subscription processing. The inefficient processing for distributing publications across the network clearly exhibited that randomly executing the strategies in the subscription processing did not have a positive impact on the performance of the KBNRan deployment.

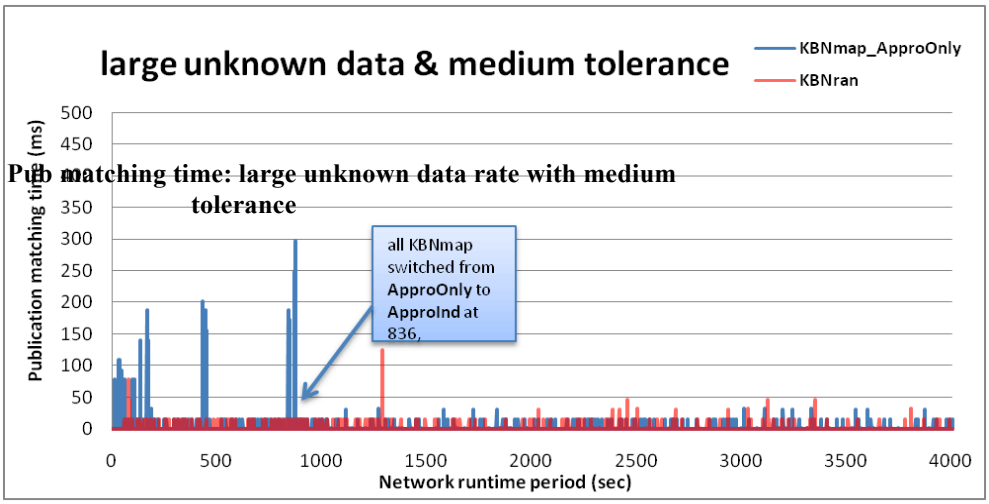
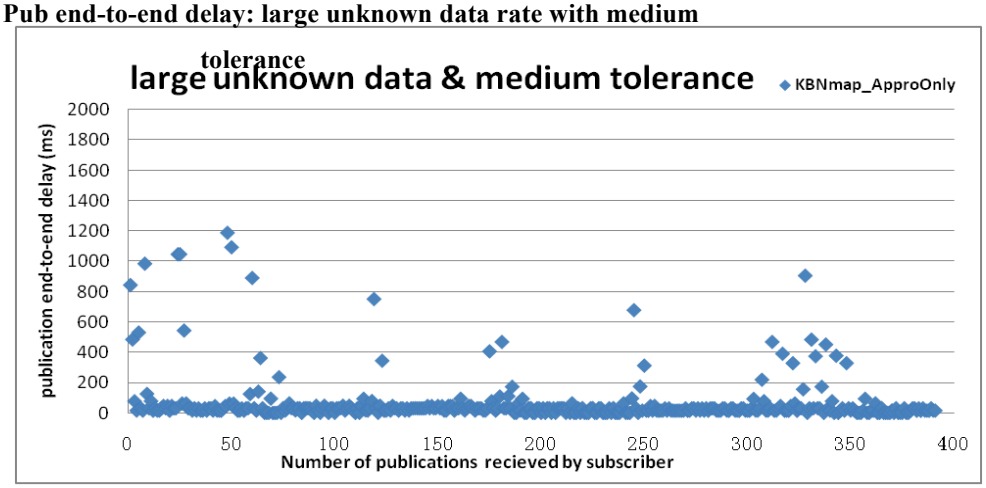
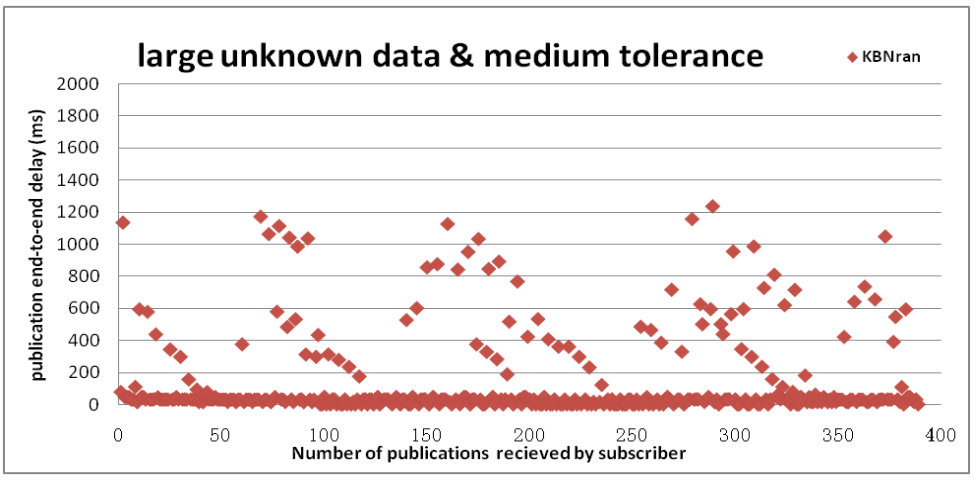


Figure F.3-B-13: Pub matching time: large unknown data rate with medium tolerance

Figure F.3-B-14: Pub end-to-end delay: large unknown data rate with medium tolerance





Case 9 of 9: Large Unknown Data Rate with Large Tolerance

This case evaluated the combination of two factors: the rate of unknown data occurrence still remained a high level (12 messages /min) but the mismatch tolerance of applications was set up to the large level. With the inputs the Bayesian Network updated a weighting of 11% for the **Every** strategy, 44% for the **ApproOnly** strategy, 45% for the **ApproInd** strategy.

Figure F.3-B-15: Pub matching time: large unknown data rate with large tolerance

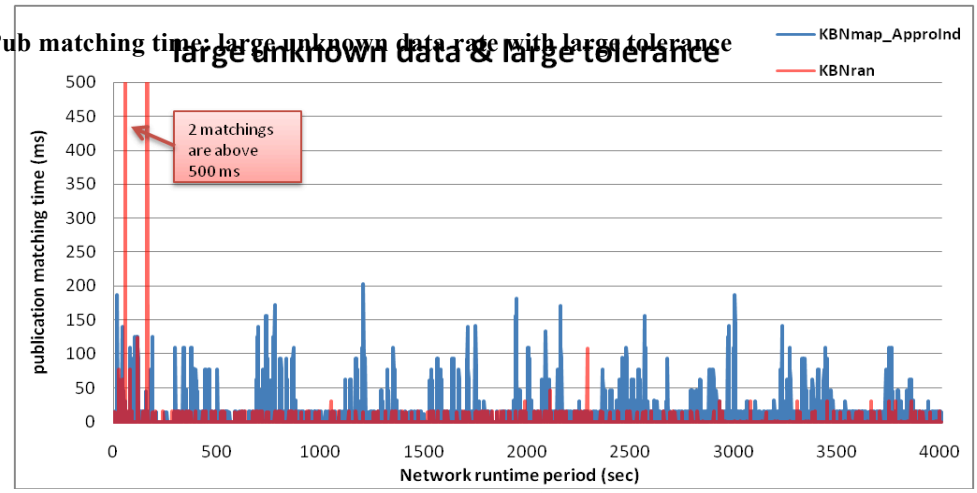
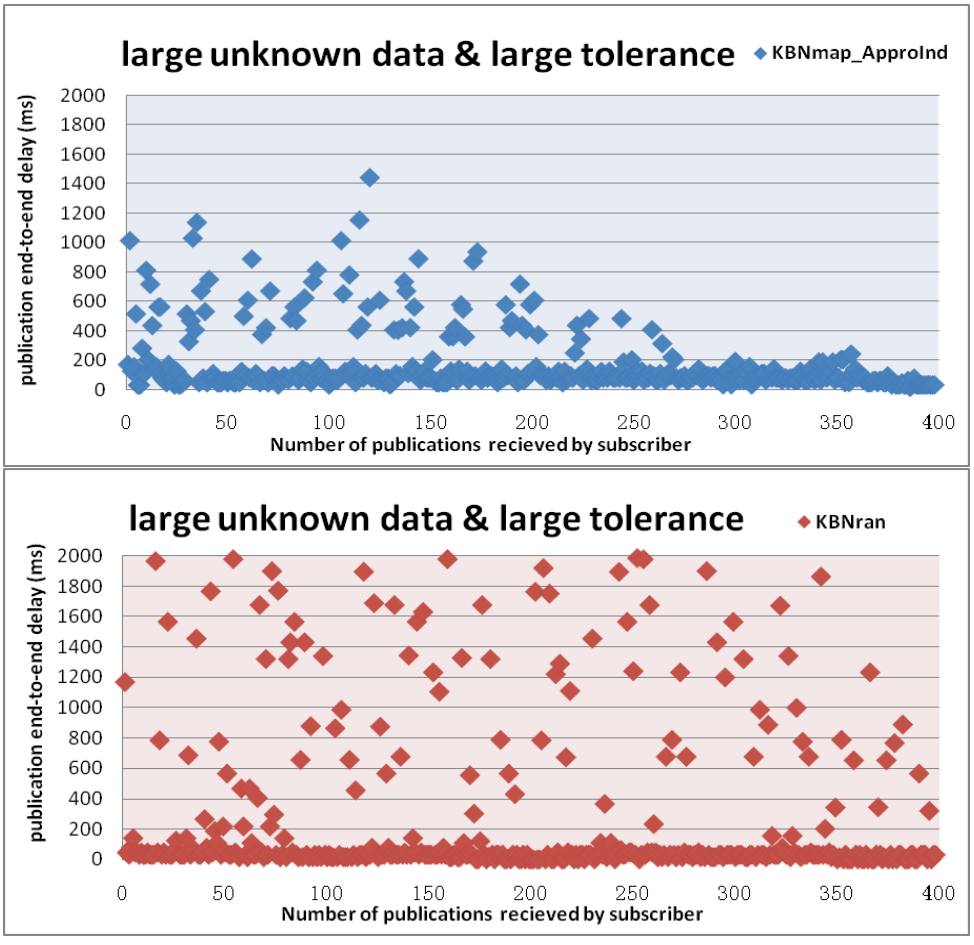


Figure F.3-B-16: Pub end-to-end delay: large unknown data rate with large tolerance



It is clear from (Figure F.3-B-15) that the KBNMap deployment executed the **ApproInd** strategy high frequently, resulting in inefficient publication matching times. As the mismatch tolerance updated to high level, switching to the ApproInd strategy indicated that the total memory resources consumed in the publication matching could be reduced in order to preserve more resources for distributing a subset of publications across the network to the destinations. This can be verified by the more efficient performance of distributing publications (Figure F.3-B-16) in the KBNMap network in comparison with the KBNRan deployment.

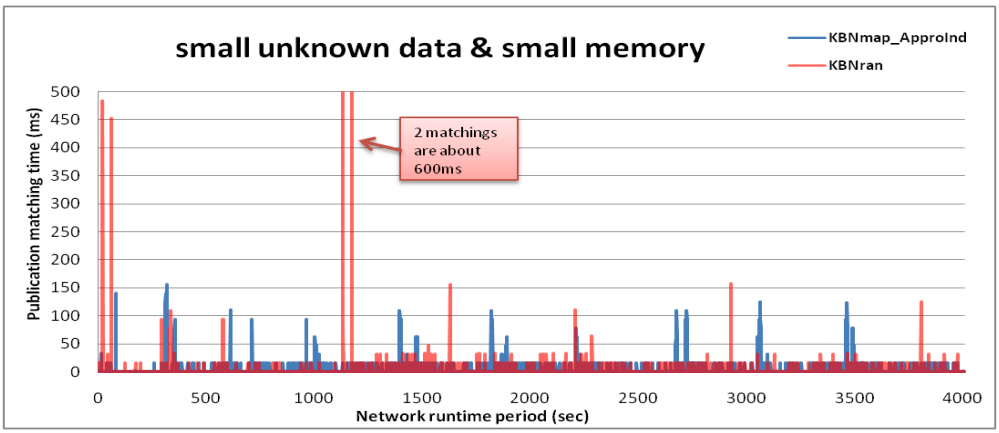
Figure F.3-C-1: Pub matching time: small unknown data rate with small memory resources

F.3.3 Testing Set C: Unknown Data Rate with Memory Resources

Case 1 of 9: Small Unknown Data Rate with Small Memory Resources Available

This case evaluated the network deployments in the situation where the rate of unknown data occurrence remained a small level (2 messages /min) while the memory resources available in the routers was allocated to a small level (40 Mb). All other factors were set to default high-load settings as before. With these inputs the Bayesian Network of KBNMap produced a weighting of 1.68% for the **Every** strategy, 27.20% for the **ApproOnly** strategy, and 71.10% for the **ApproInd** strategy.

It can be observed that the weightings of mapping strategies in this case is the same as case 1 of Testing Set B (Section 7.9.4.1), as the factor configurations of network deployments are the same in these two cases. This resulted in that the KBNmap deployment exhibited similar efficient performance as it did in case 1 of Testing Set B. In contrast, the KBNRan implementation still performed inefficiently with respect to both matching and distributing publications (Figure F.3-C-1, Figure F.3-C-2).



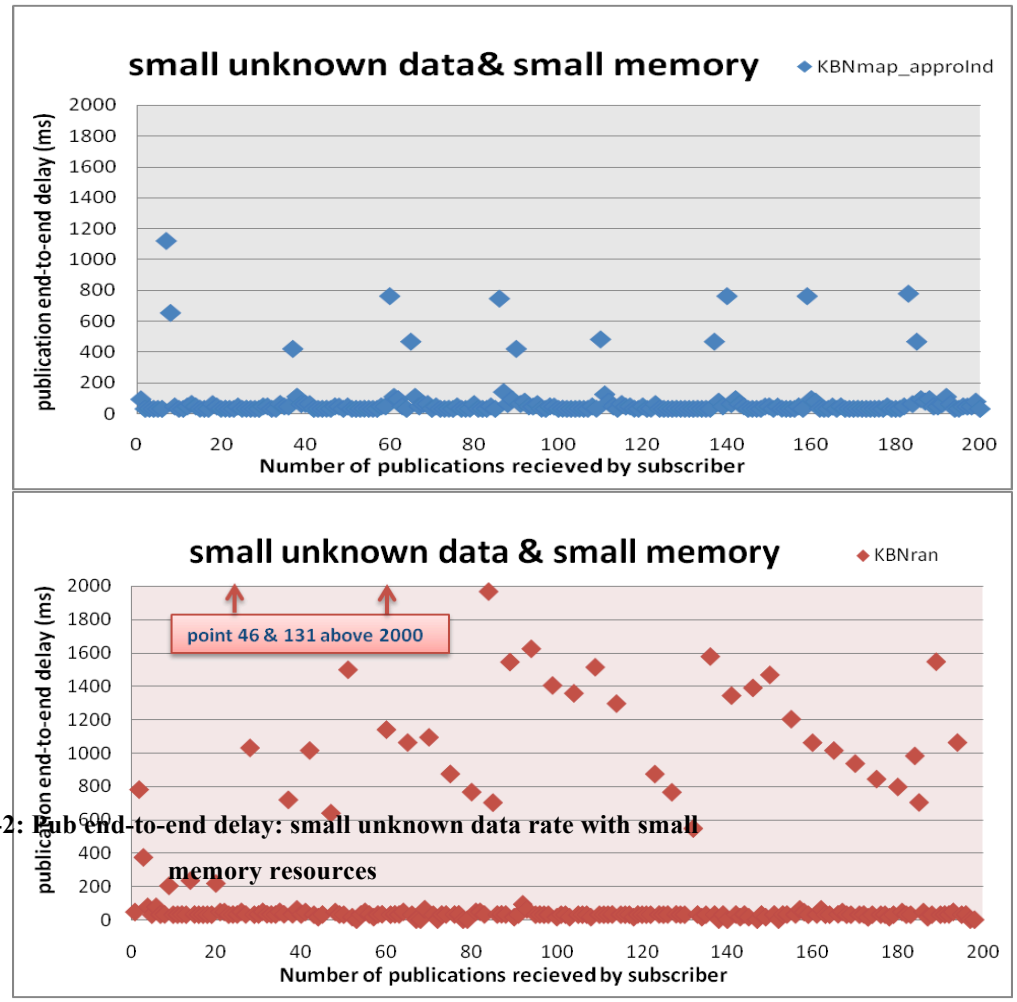


Figure F.3-C-2: Pub end-to-end delay: small unknown data rate with small memory resources

Case 2 of 9: Small Unknown Data Rate with Medium Memory Resources Available

This case presents a scenario where the unknown data rate still remained a small level (2 messages /min) while the available memory in the routers was raised to a medium level (80 Mb). With the inputs of the selected trigger factors the Bayesian Network produced a weighting of 2% for the **Every** strategy, 28% for the **ApproOnly** strategy, and 70% for the **ApproInd** strategy.

It is observed from (Figure F.3-C-3) that again the KBNRan network incorrectly executed the **Every** strategy early on, resulting in highly expensive overhead for matching publications, and then performed similarly as the KBNMap deployment. Whereas the KBNMap consecutively used the **ApproInd** strategy mainly due to the low number of unknown data observed despite the memory resources was raised enough to execute the **ApproOnly** strategy. Again this predictably low overhead was introduced for each execution.

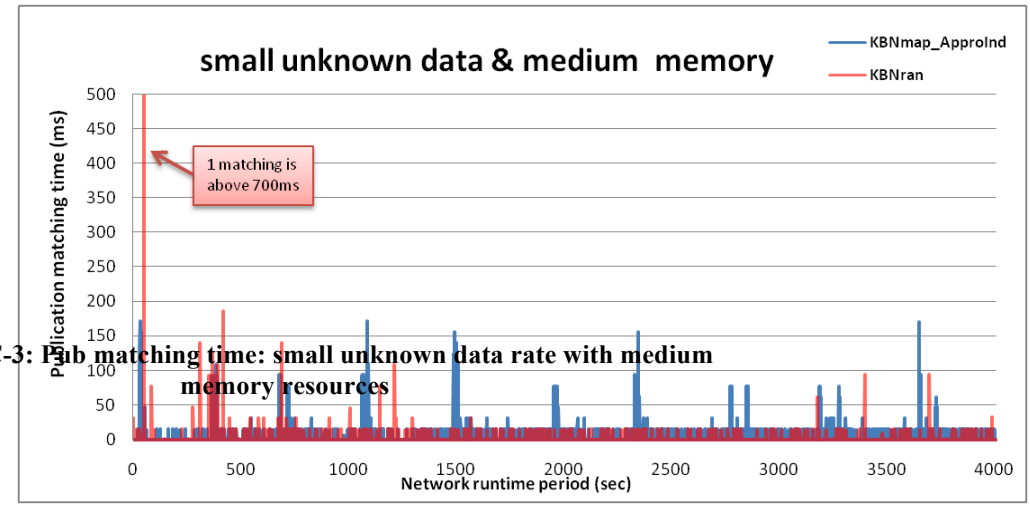


Figure F.3-C-3: Pub matching time: small unknown data rate with medium memory resources

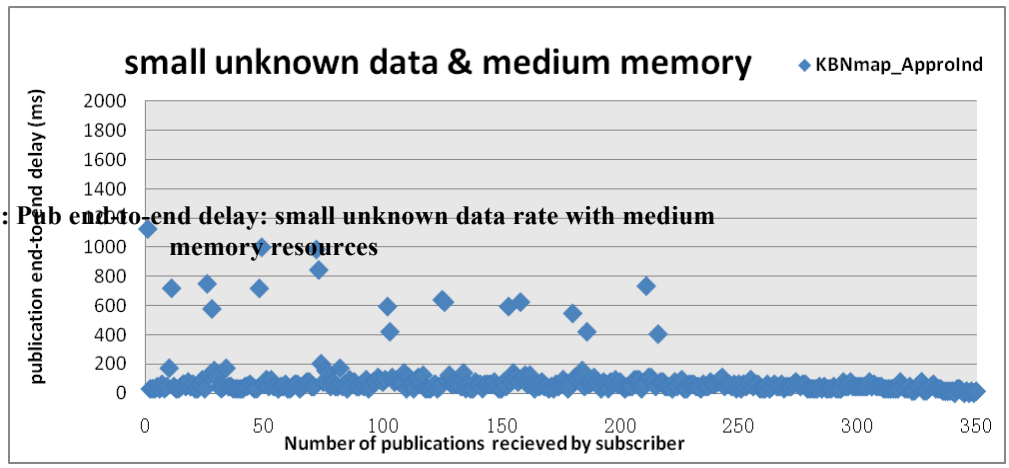
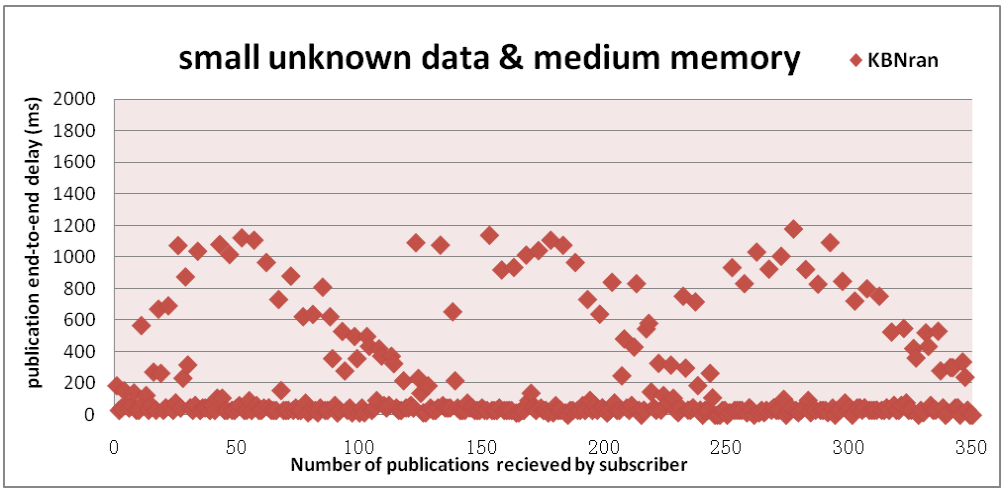


Figure F.3-C-4: Pub end-to-end delay: small unknown data rate with medium memory resources



With respect to the end-to-end delay of a subset of publications that matched a subscription and so were routed across the network over 15 hops to the subscriber, it is clear from (Figure F.3-C-4) that the KBNMap network performed far superior to the KBNRan deployment. In

addition, both implementations exhibited better performance in comparison with the pervious case, as more memory resources are allocated to the routers for operation.

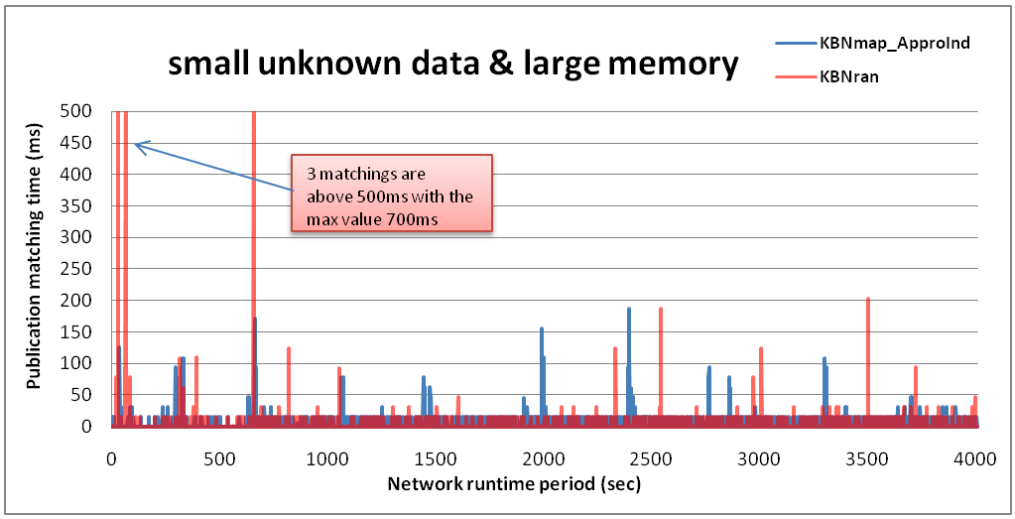
Case 3 of 9: Small Unknown Data Rate with Large Memory Resources Available

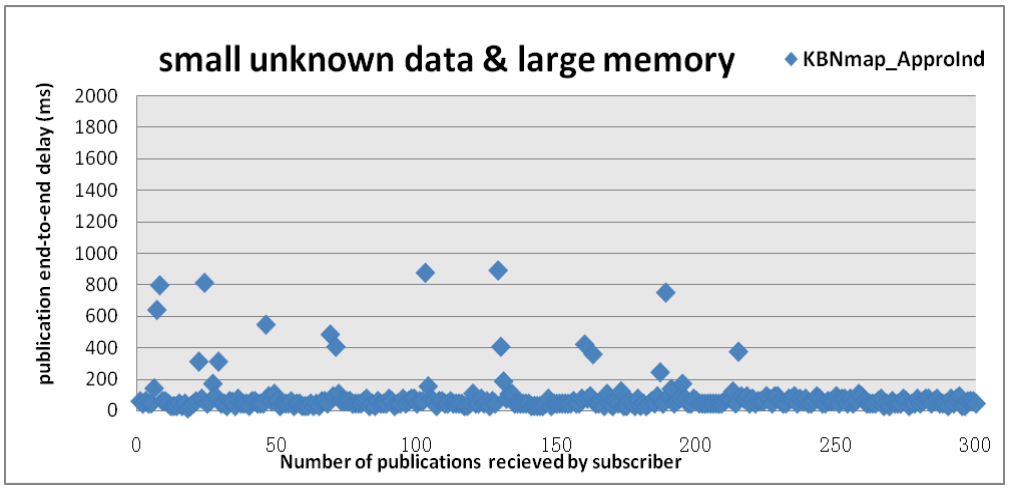
This case presents a scenario where the rate of unknown data occurrences again remained a small level (2 messages /min), but the memory resources was allocated up to a high level (120 Mb). Again all other factors were set to default high-load settings as before (see Table 7-4).

Figure F.3-C-5: Pub matching time: small unknown data rate with large memory resources

With the inputs of the selected trigger factors the Bayesian Network produced a weighting of 3% for the **Every** strategy, 23% for the **ApproOnly** strategy, and 74% for the **ApproInd** strategy.

The evaluated results in this case indicated that KBNMap clearly outperformed KBNRan in both publications matching (Figure F.3-C-5) and delivering (Figure F.3-C-6). The KBNRan implementation exhibited unpredictably high overhead early on in the experiment and then settled down to perform in a manner similar to KBNMap. Meanwhile KBNMap implementation again selected the **ApproInd** strategy, mainly due to the low number of occurrences of unknown data despite the availability of enough memory to execute the **Every** strategy. Again this resulted in the strategy being executed more often, but again with predictably low overhead for each execution. Summarising with the previous 2 cases, it is confirmed that the performance of both networks is enhanced with the increase of memory resources allocated.





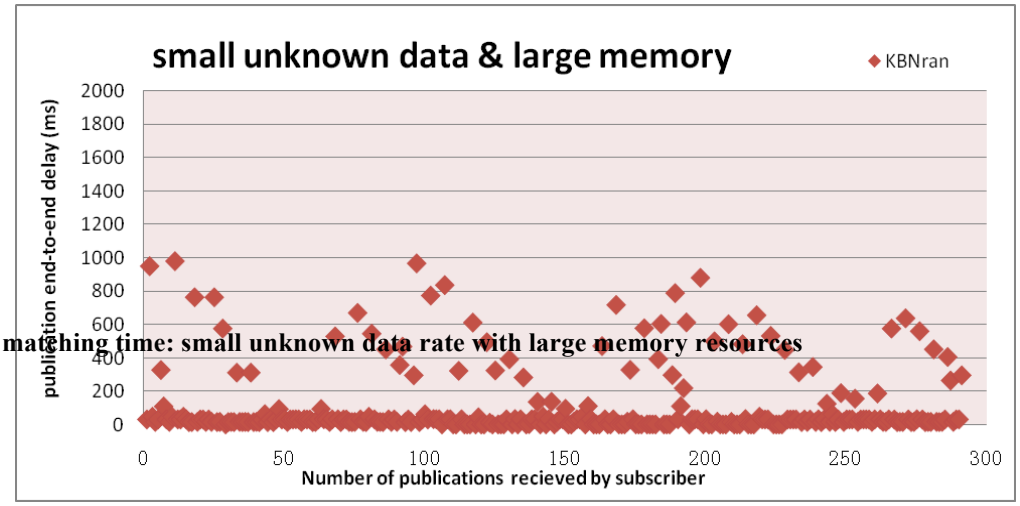


Figure F.3-C-6: Pub matching time: small unknown data rate with large memory resources

Case 4 of 9: Medium Unknown Data Rate with Small Memory Resources Available

This case evaluated the network deployments in the situation where the rate of unknown data occurrence was increased to a medium level (5 messages /min) while the memory resources available in the routers was set back to the small level again (40 Mb). All other factors were set to default high-load settings as before. With these inputs the Bayesian Network of KBNMap produced a weighting of 4% for the **Every** strategy, 44% for the **ApproOnly** strategy, and 52% for the **ApproInd** strategy.

It can be observed that the weightings of mapping strategies in this case is the same as case 4 of Testing Set B (Section 7.9.4.4), mainly due to the same factor configurations applied. Again, this resulted in that the KBNmap deployment performed similarly as it did in case 4 of Testing Set B. On the other hand, the KBNRan implementation did not exhibit similar performance even though given the same factors configuration, mainly due to the unpredictable and incorrect executions of mapping strategies. And it still performed

Figure F.3-C-8: Pub end-to-end delay: medium unknown data rate with small memory resources

inefficiently with respect to both matching and distributing publications (Figure F.3-C-7, Figure F.3-C-8).

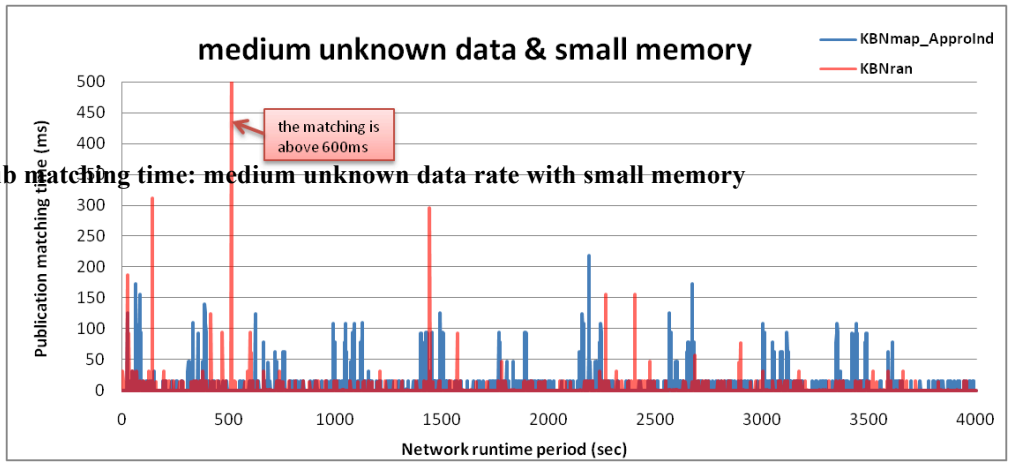
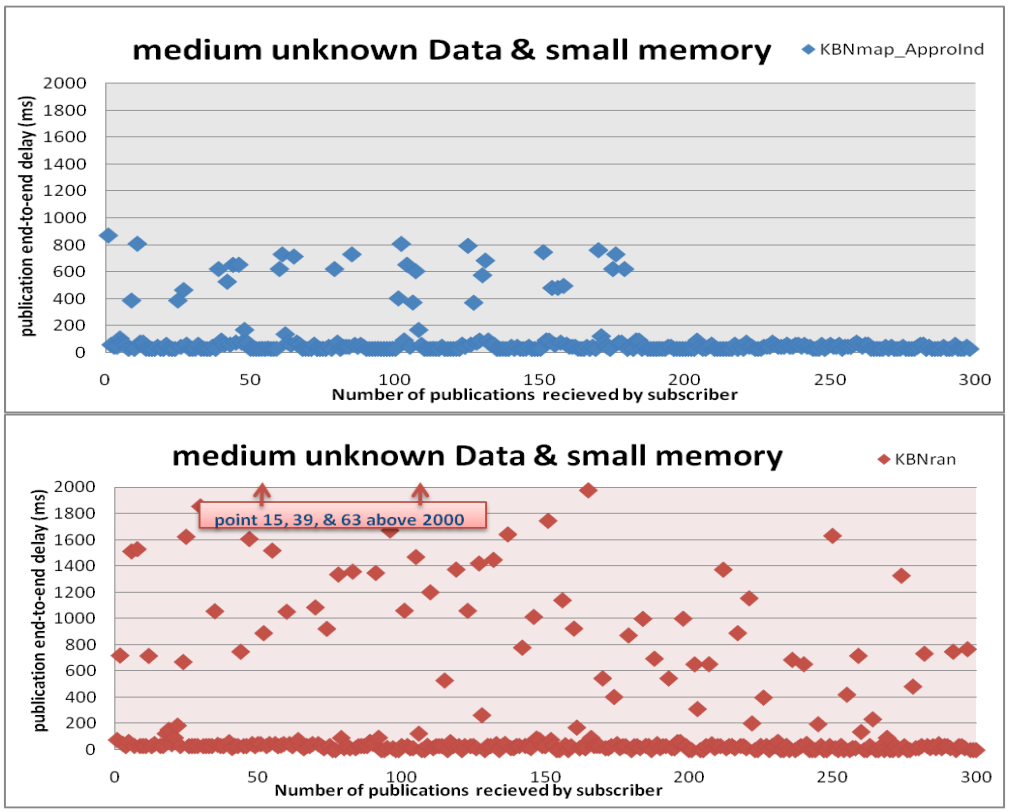


Figure F.3-C-7: Pub matching time: medium unknown data rate with small memory



Case 5 of 9: Medium Unknown Data Rate with Medium Memory Resources Available

This case presents a scenario where the unknown data rate still remained a medium level (5 messages /min) while the available memory resources were raised to a medium level (80 Mb). With the inputs of the selected trigger factors the Bayesian Network produced a weighting of

5% for the **Every** strategy, 50% for the **ApproOnly** strategy, and 45% for the **ApproInd** strategy.

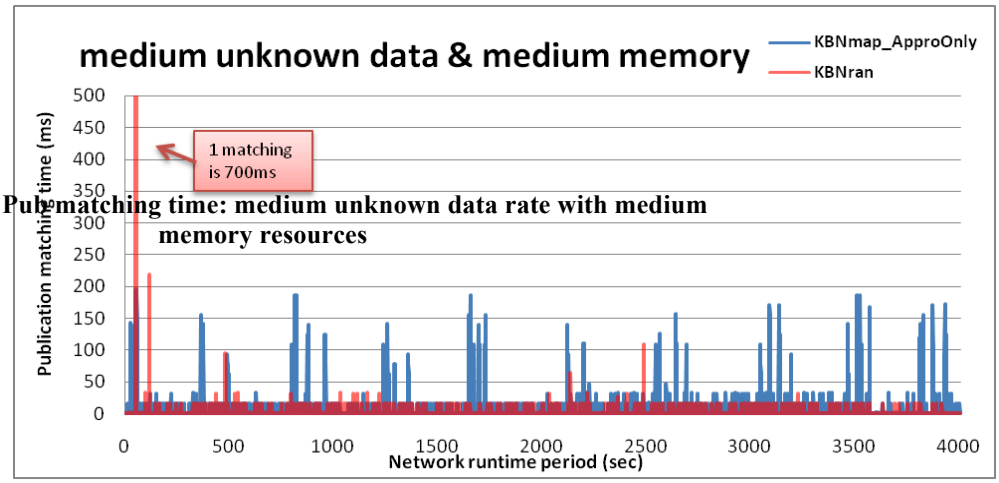


Figure F.3-C-9: Pub matching time: medium unknown data rate with medium memory resources

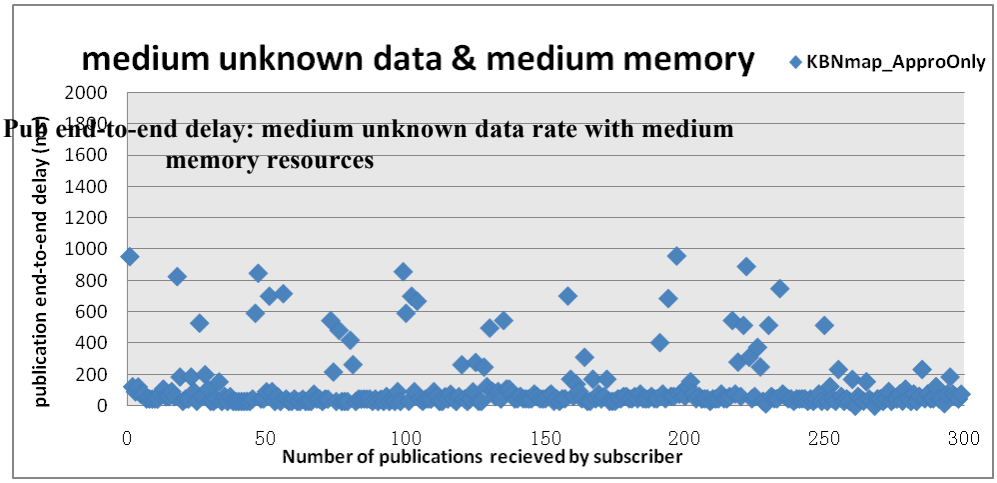
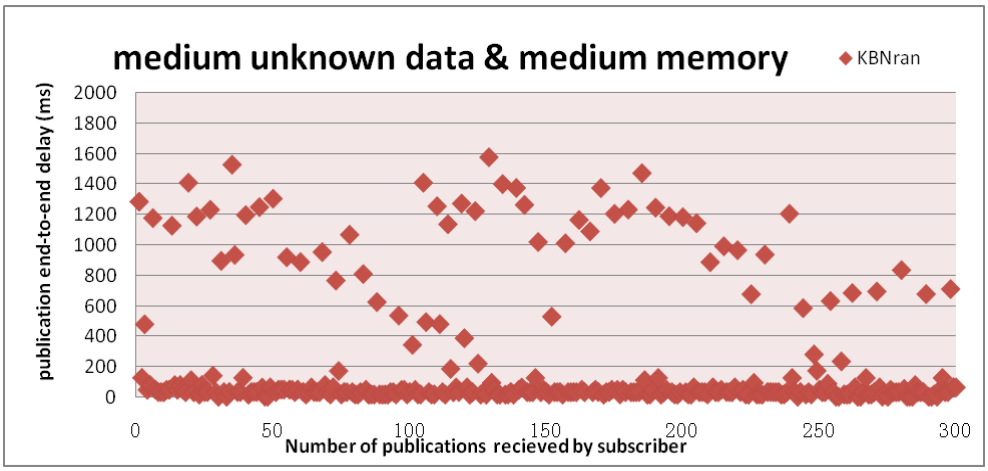


Figure F.3-C-10: Pub end-to-end delay: medium unknown data rate with medium memory resources



As shown in (Figure F.3-C-9), the **KBNRan** implementation required highly expensive extraprocessing executing the **Every** Strategy, and then the network matched publications

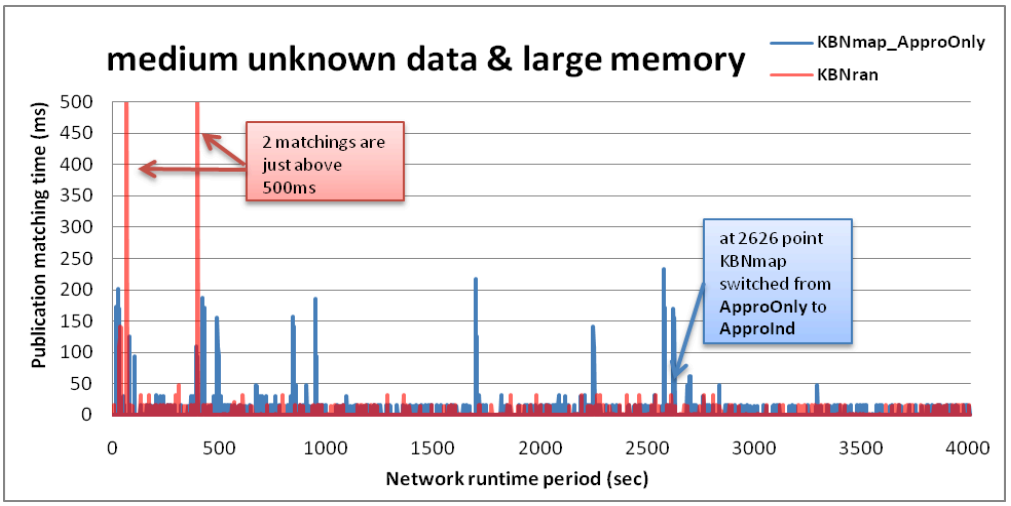
efficiently. By comparison the overhead introduced by the publication matches in the KBNMap deployment has moderately increased in contrast with the previous case, as most KBNMap routers executed the **ApproOnly** strategy several times during the experiment. This led the KBNMap deployment to spend more resources in consecutively loading and merging the appropriate mapping files.

With respect to the performance of distributing publications to the subscriber (Figure F.3-C-10), despite the **ApproOnly** strategy executed, the KBNMap deployment still performed better than the KBNRan deployment. However, comparing with the previous case, the diminished extraproccessing required for publication delivery in KBNRan indicated that the sufficient allocation of memory resources could be beneficial to the performance of publication delivery.

Figure F.3-C-11: Pub matching time: medium unknown data rate with large memory resources

Case 6 of 9: Medium Unknown Data Rate with Large Memory Resources Available

This case presents a scenario where the unknown data rate still remained a medium level (5 messages /min) but the memory resources available to the routers was assigned to a high level (120 Mb). With the inputs of the selected trigger factors the Bayesian Network produced a weighting of 6% for the **Every** strategy, 60% for the **ApproOnly** strategy, and 34% for the **ApproInd** strategy.



It can be observed from Figure F.3-C-11 that the KBNMap deployment continuously executed the **ApproOnly** strategy; however the publication matching times are lower in comparison with the previous case. This is mainly due to that sufficient memory resources were allocated. As for KBNRan, in addition to the most expensive overhead introduced in the

network startup period, the network performs efficiently for processing the majority of publication matching.

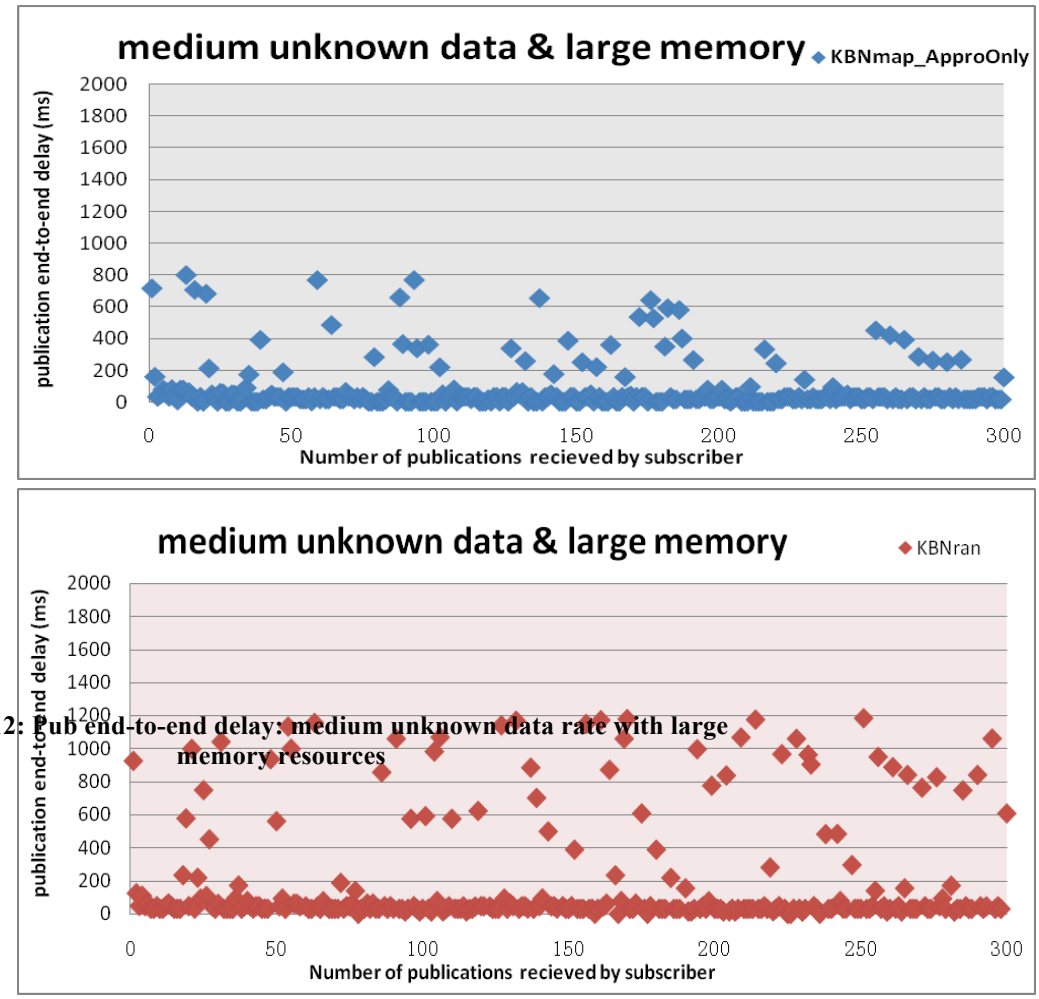


Figure F.3-C-12: Pub end-to-end delay: medium unknown data rate with large memory resources

Again it is clear from Figure F.3-C-12 that the KBNMap implementation performed far superior to the KBNRan deployment in this case. In addition, both networks performed more efficiently compared to the previous case. This further confirmed that the increased memory resources allocated to the routers is helpful to enhance the performance of KBNMap network. Moreover, the updated weighting of the **ApproOnly** strategy indicated that the BN model made a correct decision of selecting this strategy, as the efficient performance of KBNMap was achieved.

Case 7: Large Unknown Data Rate with Small Memory Resource

This case evaluated the network deployments in the situation where the rate of unknown data occurrence was increased to a large level (12 messages /min) while the memory resources available in the routers was set back to the small level again (40 Mb). All other factors were

set to default high-load setting as before. With these inputs the Bayesian Network updated a weighting of 42% for the **Every** strategy, 57% for the **ApproOnly** strategy, 1% for the **ApproInd** strategy.

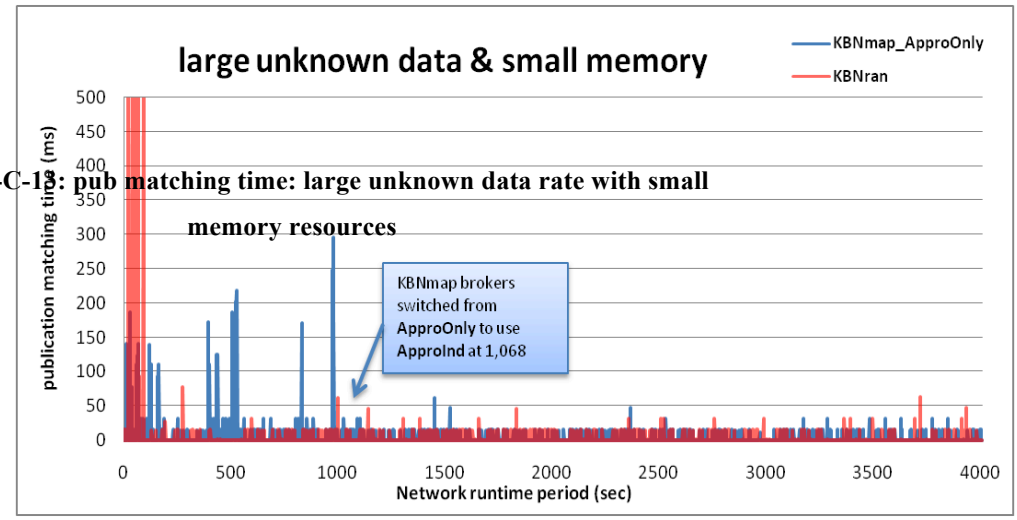


Figure F.3-C-13: pub matching time: large unknown data rate with small memory resources

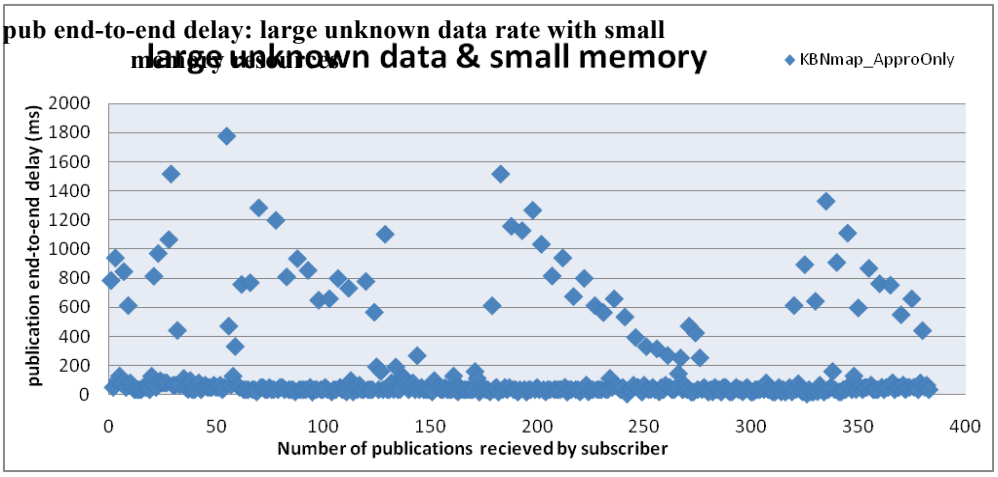
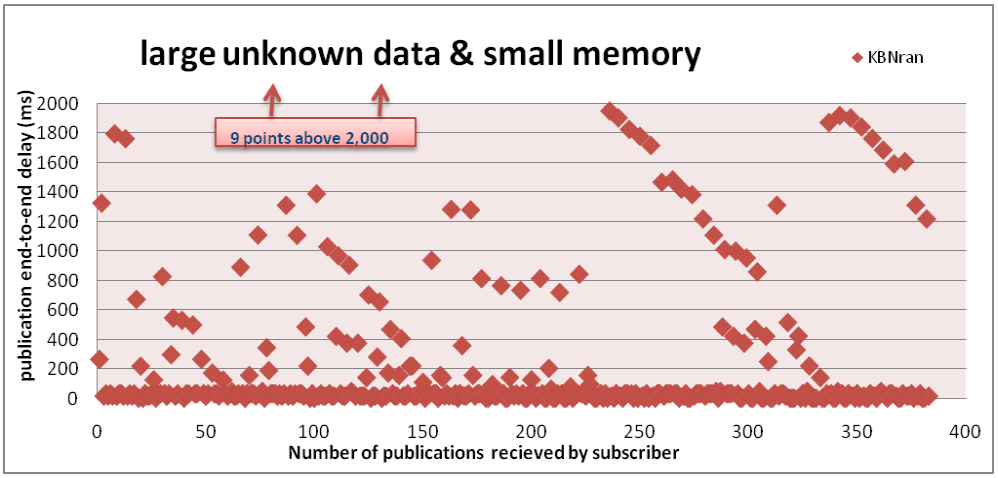


Figure F.3-C-14: pub end-to-end delay: large unknown data rate with small memory resources

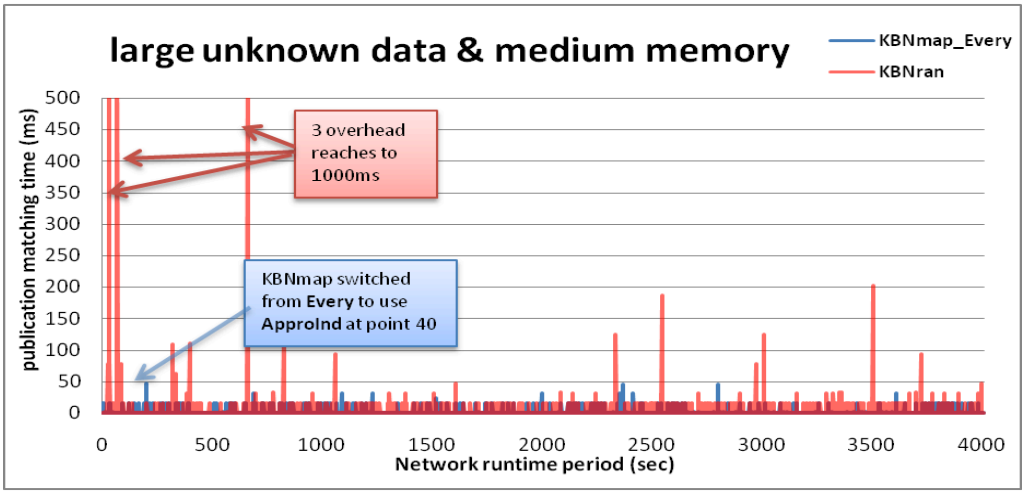


Again it can be observed that the weightings of mapping strategies in this case is the same as case 7 of Testing Set B (Section 7.9.4.7), mainly due to the same factor configurations applied. Again, this resulted in that the KBNmap deployment performed similarly as it did in case 7 of Testing Set B. On the other hand, the KBNRan implementation did not exhibit similar performance even though given the same factors configuration, mainly due to the unpredictable and incorrect executions of mapping strategies. And it still performed inefficiently with respect to both matching and distributing publications (Figure F.3-C-13, Figure F.3-C-14).

Figure F.3-C-15: Publication matching time: large unknown data rate with medium memory resources

Case 8 of 9: Large Unknown Data Rate with Medium Memory Resources Available

This case presents a scenario where the rate of unknown data occurrence still remained a high level (12 messages /min) but the memory resources available to the routers was assigned to a medium level (80 Mb). With the inputs of the configured trigger factors the Bayesian Network produced a weighting of 50% for the **Every** strategy, 49% for the **ApproOnly** strategy, and 1% for the **ApproInd** strategy.



As shown in Figure F.3-C-15, the KBNRan deployment executed the **Every** strategy early on in several routers, resulting in very high end-to-end publication matching times. In addition, other routers also executed the ApproOnly strategies at random intervals during the experiment as poorer performance was exhibited. In contrast, at approx. 40 seconds, all the KBNMap routers immediately switched from the **Every** strategy to the **ApproInd** strategy and exhibited highly efficient performance. It also verified the correct operation of the KBNmap implementation for using the **Every** strategy, which resolved all potential unknown data once by loading all mapping files and merging all referenced ontologies into the routers.

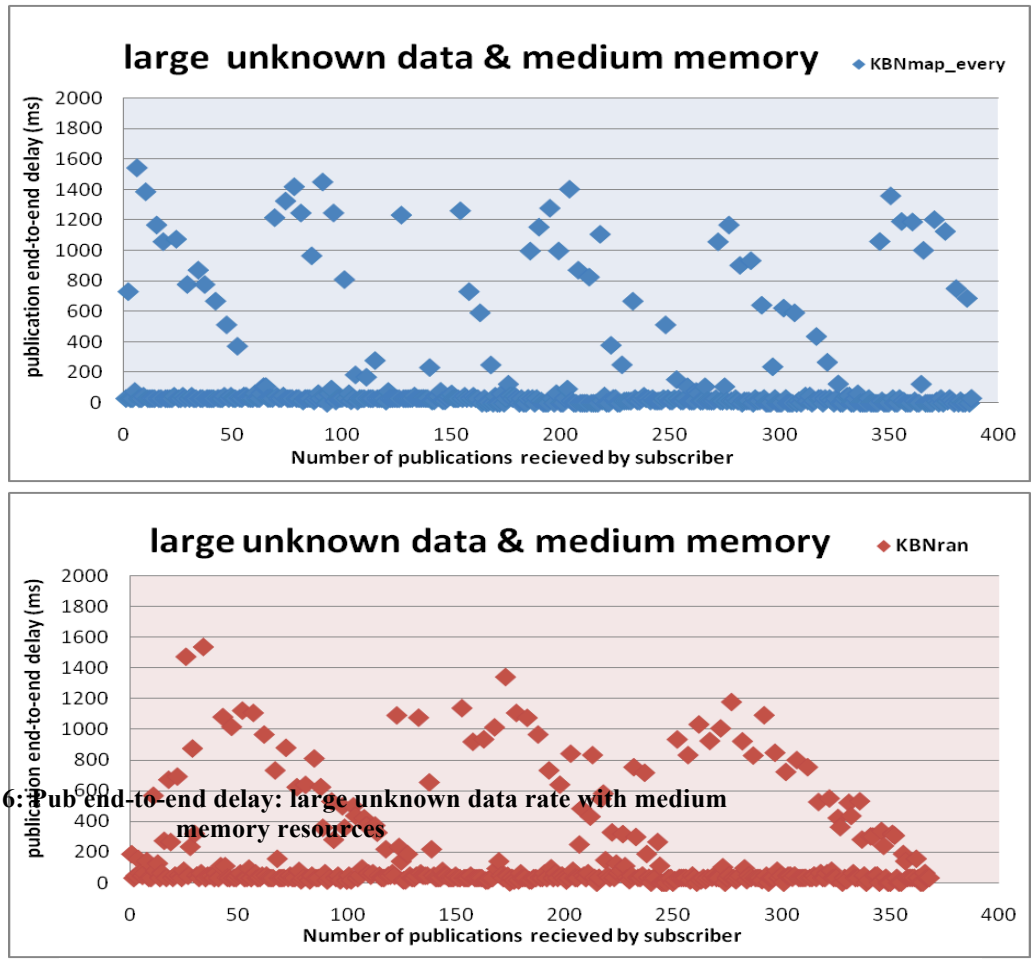


Figure F.3-C-16: Pub end-to-end delay: large unknown data rate with medium memory resources

With respect to distributing a subset of publications that matched a subscription across the entire network to the subscriber, it can be observed from Figure F.3-C-16 that the KBNMap deployment did not perform as well as it did previously, as executing the **Every** Strategy occupied the routers a large amount of memory resources. However, from another standpoint, it could be used to explain that why the BN model yielded so close weightings for the **Every** (50%) and the **ApproOnly** (49%) strategy.