An Ontological Framework for Interoperability of Health Level Seven (HL7) Applications: the PPEPR Methodology and System

Ratnesh Nandan Sahay

A Dissertation submitted for the Degree of Doctor of Philosophy

under supervision of
Dr. Axel Polleres and Prof. Dr. Manfred Hauswirth
Digital Enterprise Research Institute (DERI)
National Univerity of Ireland (NUI) Galway,
Ireland

Internal Examiner: Dr. John Breslin
Digital Enterprise Research Institute (DERI)
National Univerity of Ireland (NUI) Galway, Ireland

External Examiner: Prof. Dr. Alberto Fernández Gil
Universidad Rey Juan Carlos (URJC), Spain

External Examiner: Dr. Jeff Z. Pan
Univerity of Aberdeen, United Kingdom (UK)

Chairperson: Prof. Dr. Stefan Decker
Digital Enterprise Research Institute (DERI)
National Univerity of Ireland (NUI) Galway, Ireland

May 21, 2012

The studies presented in this thesis were performed at the Digital Enterprise Research Institute at the National University of Ireland, Galway. The research was financially supported by Science Foundation Ireland Grant No. SFI /02/CE1/I131 (Líon-1), Grant No. SFI/08/CE/I1380 (Líon-2) and by Enterprise Ireland Grant No. CFTD 2005 INF 224.

Abstract

Healthcare applications are complex in the way data and schemas are organised in their internal systems. Healthcare standards have been proposed to reduce heterogeneities between healthcare applications. Widely deployed healthcare standards such as the Health Level Seven (HL7) Version 2 specifications are designed as flexible schemas which allow several optionalities while constructing clinical messages. Such flexibility and optionalities offer choices to meet clinician messaging requirements. However, these optionalities result in a quadratic number of interfaces and alignments between Version 2 applications. To overcome the alignment problem HL7 has proposed Version 3, where a centrally consistent information model controls terminologies and concepts shared by Version 3 applications. However, it is impossible to predefine all clinical possibilities in a central information model. Consequently, Version 3 allows the creation of local vocabularies and constraints, thus further impeding interoperability of HL7 applications.

The goal of this thesis is to reduce the integration burden between HL7 applications. Our hypothesis which we aim to prove with the contributions summarised in this thesis is that "*An ontology-based integration framework can improve healthcare data interoperability by reducing: (i) the integration burden (per message) between heterogeneous healthcare applications; and (ii) the number of alignments between heterogeneous healthcare applications*". We address three main research challenges in the development of an ontology-based integration framework (i) how to build HL7 ontologies by reusing domain artefacts reside in separate global and local spaces; (ii) how to automate alignment of HL7 ontologies with greater accuracy of correspondences retrieved; and finally (iii) how to resolve inconsistencies caused by context-sensitive (or local) healthcare policies while mediating heterogeneous HL7 messages.

We provide (i) a semi-automatic ontology building methodology for the HL7 standard; (ii) a semi-automatic ontology alignment methodology for HL7 ontologies; and (iii) a detailed investigation and a solution path towards realising context-awareness and modularity for local healthcare policies. This thesis offers an ontology-based integration framework called Plug and Play Electronic Patient Records (PPEPR) that reduces the number of alignments and the integration burden between HL7 applications. Further, the PPEPR framework enables interoperation of HL7 Version 2 and Version 3 applications. HL7 ontologies and their alignments are successfully deployed and evaluated under the PPEPR framework.

"Technology feeds on itself. Technology makes more technology possible."

–Alvin Toffler

Acknowledgements

It is a pleasure to thank the many people who made this thesis possible. I am very grateful for all I have received throughout these years. It has certainly shaped me as a person and all these years of PhD studies were full of learning, challenges, chaos, and consolidating the scientific works.

I first want to thank my supervisors Dr. Axel Polleres and Prof. Dr. Manfred Hauswirth. Their inspiration, and their great efforts to explain things clearly and simply, provided major support during this thesis work. Through their wider knowledge and experience, they always guided me about where the work should go and what is necessary to get there. On several occasions I would have been lost without them.

I want to thank my co-supervisor Dr. Antoine Zimmermann. I was privileged to have him as a teacher who taught me to organise the work in a methodological way.

I want to thank my mentor Dr. Ronan Fox. His exceptional abilities to realise high-level thought into real-world solutions made the PPEPR framework successfully adopted by the healthcare industry. His support, guidance, and management helped me to focus on the practicalities of the PPEPR framework.

I am also thankful to Dr. Tomas Vitvar who gave me the opportunity to pursue my PhD studies. I would like to thank Prof. Dr. Alberto Fernández Gil and Dr. Jeff Z. Pan who agreed to serve on my thesis committee as external examiners. Further, I would like to thank Dr. John Breslin who agreed to serve on my thesis committee as internal examiner. My sincere gratitude to our director and chairperson of this thesis, Prof. Dr. Stefan Decker. We the members of DERI, NUIG are flourishing under his leadership and vision that place our works on the world map.

During this work I have collaborated with many colleagues and friends for whom I have great regard, and I wish to extend my warmest thanks particularly to: Dr. Brahmananda Sapkota, Dr. Armin Haller, Dr. Maciej Zaremba, Dr. Sami Bhiri, Dr. James Cooley, Dr. Aidan Hogan, Dr. Renaud Delbru, Dr. Helena Deus, Dr. Matthias Samwald, Waseem Akhtar, Aftab Iqbal, Nuno Lopes, Jürgen Umbrich, Sanaullah Nazir, and all those who have helped during my PhD studies. Thanks also to the administrative staff in DERI, especially, Hilda Fitzpatrick, Claire Browne, Andrew Gallagher, Gerard Conneely, and Dr. Brian Wall, who provided a great environment in excelling our research work.

Finally, I want to thank my family members, especially my wife Shivani, my little daughter Gyani, my brother Amit, and my sister Alka. Without their love and understanding it would certainly have been much harder to finish a PhD.

Dedicated to my grandfather, Rajendra Prasad
Dedicated to my inspiration, Prof. Gustav Roth
Dedicated to my parents, Renu and Abhay

Declaration

I pledge my honor that this thesis represents my own work in accordance with National University of Ireland, Galway regulations. This thesis work has not been submitted for any other degree or professional qualification except as specified.

Ratnesh Nandan Sahay

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In a complex domain like Healthcare and Life Sciences (HCLS) knowledge is represented in clinical information models, repositories (databases), ontologies, terminologies, and vocabularies. Over the past two decades healthcare domain specialists are constantly working toward realising an Electronic Health Record (EHR) system [2] that can enable the sharing of patient information across different healthcare stakeholders. Generally, the EHR system is created and maintained within a healthcare institution, such as a hospital, clinic, or physician office. One of the main purposes of the EHR system is to give patients, physicians, and healthcare providers (*e.g.*, payers, insurers) seamless access to a patient's medical records across different facilities. Considering the impact of this domain, EHR standardisation bodies play a crucial role in defining all entities (*e.g.*, terminologies, codes, vocabularies, information models) related to the construction and exchange of clinical messages.

Health Level Seven (HL7)[1] is the most widely deployed healthcare standard, which defines information models and schemas for constructing and exchanging clinical information across healthcare stakeholders. There are two major HL7 versions, HL7 Version 2 [3] and Version 3 [4]. The HL7 Version 2 standard was created mostly by clinical interface specialists and the majority of HL7 applications comply with Version 2. Version 3 is emerging and advocated by medical informaticians for improving consistency and interoperability within healthcare applications. Unfortunately, heterogeneity exists within HL7 (*i.e.*, between Version 2 and Version 3) and also between other prominent healthcare standards (*e.g.*, openEHR[2], CEN/ISO EN13606[3]). Interoperability of HL7 Version 2 and Version 3 is crucial to bridge the gaps between these two major deployments across the healthcare industry [5]. Furthermore, the involvement of various stakeholders, such as hospitals, healthcare standards, pharmacy, patients, multiplies the integration complexity of this domain.

Healthcare integration solutions introduced in recent years have limitations in dealing with the EHR messaging elements and their alignments: (i) the number of alignments/interfaces between two healthcare applications is quadratic, where $n \times (n-1)$ alignments (for n number of healthcare applications) are required to exchange clinical messages; (ii) the alignment of heterogeneous healthcare applications is a manual process triggered by the domain experts; finally, (iii) syntactic integration solutions are unable to use the concepts behind messaging elements, terminologies, codes, and

[1] http://www.hl7.org/
[2] http://www.openehr.org/
[3] http://www.cen.eu/

constraints associated with a patient record. With the increasing number of interacting healthcare applications, existing integration solutions face challenges of construction and management of the alignments. Our major aim in this thesis is to reduce the number of alignments and consequently the integration burden between pairs of healthcare applications that exchange patient records. Along with reducing the number of alignments our other important aim is to enable interoperability between Version 2 and Version 3 applications.

The presence of different healthcare standards, large scale applicability, and limitations of syntactic integration solutions, motivated healthcare specialists to apply Semantic Web technologies to resolve heterogeneity in a formal and consistent way [6, 7]. Semantic interoperability has been considered in the healthcare domain for the integration of data from heterogeneous sources through semantic mediation [8]. Ontologies, pillars of the Semantic Web framework, are used to annotate message definitions to assign formal understanding and enable mediation between them. Ontology as the symbolic layer is closest to concepts in the real world. An ontology may be defined as the specification of a representational vocabulary for a shared domain of discourse which may include definitions of classes, relations, functions and other objects [9]. In this respect, an ontology can: (i) provide a formally consistent and computable domain model as a reference point for heterogeneous clinical messages, thus reducing the bilateral alignments between healthcare applications, (ii) automate the alignment task for semantically similar domain artefacts, and (iii) apply reasoning about the domain concepts and their relationships. Semantic mediation can be used to convert healthcare messages defined in one standard into another. Such transformation requires the lifting of syntactically defined messages into a semantic format, and then lowering of semantic format into the syntactic counterpart. During transformation, semantic differences are reconciled through ontology alignment mechanisms.

In order to provide semantic interoperability for HL7-based applications, the initial development stage starts with some obvious questions: how to build ontologies for such a vast and complex standard? Is there any methodological support available for building healthcare ontologies? Unfortunately, traditional ontology building methodologies have several limitations in dealing with concrete applications [10, 11, 12]: (i) emphasis is given to build a central ontology and guidelines are provided to generalise or specialise the conceptual hierarchy. In case of the HL7 standard, domain knowledge is arranged into layers, *i.e.*, strict separation is kept between what is global knowledge (shared by all users) and local knowledge (meant for application or institution specific requirements), (ii) great effort is invested at the stage of gathering requirements, *i.e.*, building consensus among the ontology users. However, for HL7, the standard is itself an agreement between HL7 users. Therefore, the priority shifts from requirement gathering to reuse of HL7 resources published in various formats.

HL7 ontologies are built in layers, some of them represent universal concepts such as data types, vocabularies, and information model specific artefacts. Additionally, local ontologies are constructed from existing clinical messages, which contain local concepts and constraints applicable for a particular hospital or department within a hospital. In the context where several hospitals and departments communicate, these different sets of ontologies form a distributed knowledge space where a local system shares a globally available domain knowledge as well as creating its own concepts and constraints. This thesis aims at providing a detailed ontology building methodology for the HL7 standard.

If there are multiple local ontologies, it is important to decide what goes into the local ontologies, and what goes into the global ontology, and how they are related to each other [13, 14]. Is the intent for the global ontology to cover the entire range of concepts that occur in all of the local ontologies? Or, will the global ontology include only a subset for reaching agreement between users. In addition to building global and local ontologies, the next step is to resolve heterogeneity of clinical systems through ontology alignments. For instance, an artefact may be classified in one global ontology only, or in multiple local ontologies, or in both the global and local ontologies. If there is more than one local ontology, questions arise as to whether and how local groups can make use of ontologies from other groups. To do this, alignments are required for a given term and the corresponding term in another ontology [15, 16]. This thesis provides a mechanism to align Version 2 and Version 3 of the HL7 standard. Finally, as said above, the interoperability of local clinical concepts and constraints with globally defined concepts is important for information exchange between different clinical systems. Constraints enforce certain policies aimed at local groups and applications. The current Semantic Web framework has limitations in distinguishing global concepts from locally defined concepts and constraints.

The notion of contextual and modular ontologies directly relates to the problem of integrating global and local ontologies [17, 18]. A module within an ontology is seen as a sub-part of ontology axioms encapsulated within an ontology. Context provides the identification of each module within a knowledge base and additional contextual information may be used to include or avoid certain modules while combining global and local ontologies. We can consider that distinct modules are in fact distinct contextual ontologies. In the following chapters we explore a number of issues related to the creation, and combination of local and global ontologies to achieve the benefits of healthcare standardisation, while meeting local requirements. For each, we investigate in detail the various options and propose possible solutions, then we discuss implications of each choice under different circumstances.

The next section describes the research motivation of this thesis based on how HL7 compliant applications arrange global and local schemas. Firstly, we present the research motivation behind (i) enabling interoperability between HL7 Version 2 and Version 3 applications and (ii) enabling interoperability between globally and locally defined clinical concepts. Secondly, we outline research challenges in order to provide interoperability between HL7 applications. Then, we briefly summarise the thesis contributions, followed by the overall thesis structure.

1.1 Research Motivation

Interoperability of Electronic Healthcare Record (EHR) systems is a key research field in medical informatics over the last two decades [19, 8]. An EHR includes information such as observations, laboratory tests, diagnostic imaging reports, treatments, therapies, drugs administered, patient identifying information, legal permissions, and allergies for a patient. Currently, this information is stored in heterogeneous and proprietary formats through a multitude of medical information systems supported by various vendors. The HL7 standard has been proposed to describe all artefacts needed to construct and exchange patient records. However, the lack of interoperability within healthcare standards adds complexity to interoperability initiatives. This heterogeneity exists between two

versions of same standard (*e.g.*, HL7 Version 2 and Version 3), and also between standards (*e.g.*, HL7, openEHR, CEN/ISO EN13606). This results in severe interoperability problems in the healthcare domain.

Interoperability of HL7 Version 2 and Version 3

Making EHRs interoperable depends much on how the medical vocabularies and message schemas are integrated. The integration should facilitate more effective and efficient patient care by retrieving and processing clinical information about a patient from different sites. It should be noted that the interoperability of IT systems can be assessed on different levels, from the communication protocol and service levels, to data, schema and the behaviour (or functionality) of the system as perceived by the end users. This thesis focuses on conceptual level interoperability which can simplify the interoperability at syntactic level. Conceptual here means the semantics of information exchanged between healthcare applications. In particular, this thesis is centred around the interoperability of Version 2 and Version 3 of the HL7 standard.

Figure 1.1 shows a high level view of alignments required between Version 2 and Version 3 applications. The bottom of Figure 1.1 shows various constraints, that is, different sets of polices within healthcare applications. We notice in Figure 1.1 that two types of alignments are required between the Versions. The first type is a vertical alignment between messages from the same version and the second type is a horizontal alignment between different versions. The rectangular boxes within Figure 1.1 denote healthcare applications containing global and local parts of message schemas.

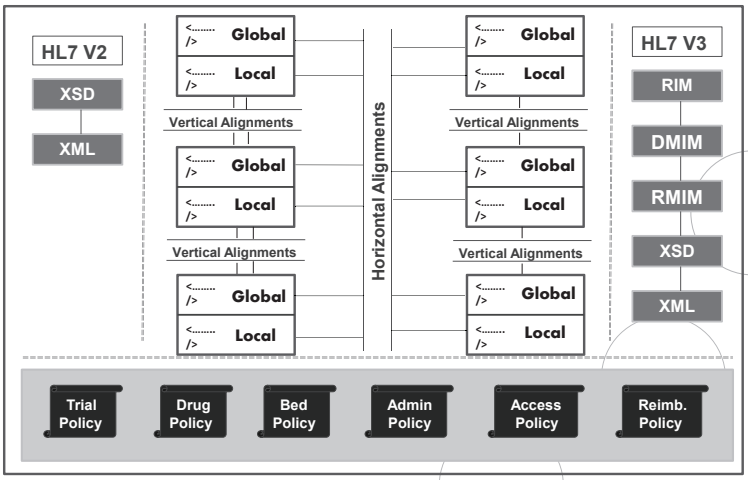

Figure 1.1: Heterogeneity between HL7 Version 2 and Version 3

HL7 Version 2 is based on the Electronic Data Interchange (EDI) format [3] (left side of Figure 1.1) but specifications are also available in XML [20] format. HL7 Version 3 specifications (right side of Figure 1.1) are represented in UML [21] and XML formats. The HL7 Version 3 information modelling process recognises three interrelated types of information models (right side of Figure 1.1): Reference Information Model (RIM), Domain Message Information Model (DMIM), and Refined Message Information Model (RMIM). RIM is a unified model which covers all domains

in healthcare and defines data structures, data types and vocabularies, as well as the relationships between them. RIM structural arrangement is based on an object-oriented paradigm and includes several classes, representing domain models from which all healthcare related messages and documents are built. DMIM is a refined subset of RIM used to create messages for a particular domain (*e.g.*, Lab, Hospital Admin). RMIM is a subset of DMIM and is used to express the information content in a message or set of messages (*e.g.*, Lab-Test Order Message). All three interrelated models use the same notation and have the same basic structure but differ in their information content, scope, and intended use. Each domain model specialises from the upper information model. Finally, at the messaging level, the RMIM model is exported in XML Schema format. HL7 Version 3 is more structured and due to presence of the central RIM, it provides greater consistency than Version 2 across the entire standard and various deployments. Version 2 specifications are flexible, involving optional fields and segments that cause a major interoperability hurdle between clinical systems.

It is important to notice in Figure 1.1 that alignments between healthcare applications (rectangular boxes) either contain double lines or a single line. Double line represents correspondences between global and local schemas of two healthcare applications and a single line indicates correspondences between local schemas. Vertical alignments between Version 3 applications are a single line when compared to double line correspondences required between Version 2 applications. The reason for this is the presence of a centralised information model (*i.e.*, Reference Information Model (RIM)) within Version 3 that globally controls all the vocabularies and terminologies shared by the users. Similarly, horizontal alignments between Version 2 and Version 3 applications require correspondences between global and local schemas. As said above, the core of this thesis is to limit these bilateral correspondences for local schemas only, instead of providing pair-wise correspondences between global and local schemas. In this envisioned situation, global schemas and their alignments are used as a central conceptual model and local schemas represent conceptual models applicable to a specific hospital environment. This thesis uses Semantic Web technologies to represent conceptual models and their alignments. Two major advantages of providing semantics to the HL7 standard are:

- **Formal Conceptual Model:** Version 3 applications have proved that the presence of a central conceptual model reduces heterogeneity within the same version [22]. Along a similar line, Version 2 is required to be lifted to a conceptual level. Enabling a conceptual model for Version 2 will reduce: (i) vertical alignments within Version 2 applications, and (ii) horizontal alignments between Version 2 and Version 3 applications. An ontology based conceptual model will provide a formal, computable, consistent, and flexible means to encode domain knowledge.

- **Conceptual Alignments:** The use of semantics for vertical and horizontal alignments has several advantages: (i) differences are resolved at a conceptual level, in comparison to per-message alignments (which is the case of syntactic integration solutions), (ii) ontology matching tools can automate the alignment task, and (iii) formally defined alignments have the edge over syntactic alignments because consistency and correctness are checked during construction and subsequently the management of alignments. For example, in the case of syntactic integration solutions, consistency and correctness of alignments can only be checked at run-time (*i.e.*, during actual transformation of messages). Conversely, ontology based alignment can use reasoning to ensure the consistency of correspondences during design-time, which may be

helpful for developers to foresee any errors even before any actual transformation of messages.

Apart from the two basic advantages mentioned above, the benefits of providing semantics to the HL7 standard are manyfold. One such benefit is the integration of important bio-medical vocabularies already available as ontologies with the HL7 standard. The HL7 environment uses various bio-medical or life sciences vocabularies to annotate a patient observation. HL7 developers often encounter difficulties in using vocabularies that are outside the HL7 domain because life science vocabularies are described at a different level of granularity.

Interoperability of Local Clinical Concepts and Constraints

Once the ontology-based conceptual model and alignments are constructed, the next issue is how to integrate local constraints and concepts with the global conceptual model. We have discussed in the previous section that the use of semantics has several advantages in resolving differences between healthcare applications that comply with HL7 Version 2 and Version 3 standard. However, semantically-enabled healthcare applications have limitations when dealing with local artefacts in conjunction with a global conceptual model. In order to provide a solution that can use the combined knowledge of global and local models (including both concepts and constraints), it is important to understand the internal information structure of HL7 systems: for instance, how information is arranged within a system and how a system identifies message elements and constraints.

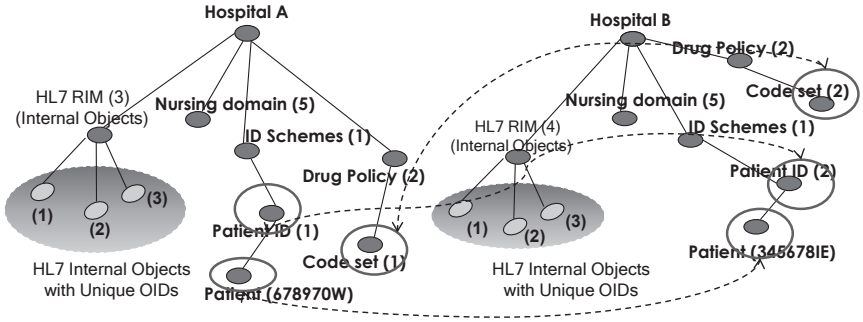

Figure 1.2: Example of HL7 Version 3 Modular Information Structure

Figure 1.2 shows internal information structures of two hospitals complying with HL7 Version 3. HL7 Version 3 clinical information is organised in a modular fashion instead of arranging it within a single space. Modularity here means that subparts of a clinical message (*e.g.*, patient details, lab report) are separated as identifiable modules with a Version 3 information space. The messaging modules are combined as per the requirements during the construction of a clinical message for a patient. HL7 Version 3 uses a special mechanism that assigns the naming and the coding scheme to identify each set of clinical information. All information within a Version 3 application is identifiable and arranged into a modular tree-like structure. Version 3 artefacts use OIDs (Object identifiers, an ISO standard [23]) to identify coding schemes and identifier namespaces. OIDs can be allocated by any organisation using a unique OID root. A single message can use OIDs from various sources and a single scheme can be identified by more than one OID (*e.g.*, by an OID from more than one organisation). We notice in Figure 1.2 that both hospitals have drug policies: *Hospital A* has defined

the local *code set 1* for the drug policy while *Hospital B* has *code set 2* for the policy. Similarly, a patient of the *Hospital A* has been identified locally as 678970W (Social security number), while the same patient admitted to the *Hospital B* is identified by his driving licence number: 345678IE. Apart from local settings, both hospitals share a common vocabulary for nursing domain. The broken lines shown in Figure 1.2 are correspondences between similar entities in different modules of two hospitals exchanging patient records. It is important to note that some advanced subversions of HL7 Version 2 (Version 2.5 onwards) supports OID-based messaging structure.

We argue that the full advantages of semantics for healthcare applications can only be realised when applications are able to use local concepts and constraints according to the local environment. There is a need for a mechanism that can permit clinical systems to share health record data whilst preserving the clinical meaning of locally (or individually) authored content and constraints [24]. We see two major domain-related uses of a mechanism that can enable the handling of local concepts and constraints within ontological frameworks.

- **Clinical Information Exchange**: Healthcare applications consist of different information parts (*e.g.*, medical history, current lab observation, billing information) of a comprehensive patient record. Clinical information uses several vocabularies and constraints to describe their contents. Each of these vocabularies are either described and controlled by domain specific standardisation committees or by local authorities. Therefore, clinical information at care sites can only be well understood when these vocabularies and constraints are combined according to the local environment.

- **Patient-Centric Care**: A patient-specific clinical information spans across different sites, treatment times, diseases, medical situations (*e.g.*, routine, accidents), etc. It is important to arrange them in information modules and instead of combining them all together, integration should be centered according to point-of-care, that is, only those information pieces should be combined or excluded that are appropriate for any treatment situation.

Considering the above domain issues, the next section describes emerging research challenges; addressing each we aim at closing important gaps in the healthcare domain using Semantic Web technologies as an interoperability enabler.

1.2 Research Challenges

Providing semantics to both HL7 versions starts with a basic question: how to build ontologies for these versions? Additionally, alignments between the two versions are not a straightforward task, as each follows different sets of data types, vocabularies, and messaging elements. Furthermore, healthcare institutes can choose to follow a standard or a mix of standard compliant schemas with local changes, or they can opt for open vocabularies like GALEN [25] or SNOMED [26] to model their clinical messages. Regardless of what they choose, the issue of combining global and local information always appears during the integration process. Main challenges are:

1. How to build HL7 ontologies by reusing domain artefacts reside in separate global and local spaces: The traditional ontology building methodologies focus on [10, 12]: (i) consensus building, and (ii) developing, generalising and specialising the central ontology. Recent advancement of technologies and increasing availability of structured non-ontological resources may automate the ontology building process. Use of structured non-ontological resources is addressed at a high-level within existing ontology building methodologies [27, 28, 29, 30]. In our case, considering the impact, use, and scale of the HL7 standard, we need a special ontology building methodology for the standard. It requires: (i) the identification of all development steps (from scope to testing), (ii) the development of ontologies in layers (*i.e.*, global and local), (iii) a greater emphasis on the use of (non-ontological) resources (*e.g.*, UML, XML(S)), and finally (iv) a generalised methodology that can ontologise HL7 in different local environments.

2. How to automate alignment of HL7 ontologies with greater accuracy of correspondences retrieved: The integration and alignment of ontologies is a well-investigated fundamental problem in the development of Semantic Web techniques. Most research in the ontology matching area has focused on (semi-)automatically aligning ontologies using approaches based on combinations of syntactic similarity, graph similarity, and using a third-party upper level ontology [31]. The (semi-)automated approaches of linguistic and structural matchings typically give an incomplete or incorrect set of correspondences between terms. A human must align the remaining terms and check the machine-built alignments to truly complete the alignment process. In the case of domain-specific ontologies, alignment complexities increase because (i) independently built ontologies can vary significantly in the terminology they use and the way they model the same entities; and (ii) domain ontologies focus on a particular domain and use terms in a sense that is relevant to the domain and which are not related to similar concepts in other domains. We investigate to what extent general ontology alignment methods are applicable for Semantic-Web enabled healthcare applications. This thesis places an emphasis on providing alignment mechanism that can improve alignment accuracy and automate ontology matching results for HL7 ontologies.

3. How to resolve inconsistencies caused by context-sensitive (or local) healthcare policies while mediating heterogeneous HL7 messages: Standard ontology modelling and integration approaches assume that everything is global and the local perspective of the domain is largely ignored. The issue of contextual or modular ontologies is directly related to the problem of integrating global and local ontologies [17, 18]. In many cases, when a local vocabulary or constraint (*i.e.*, policies) is used outside of the original publication site, it loses meaning and creates inconsistencies. To avoid such inconsistencies, it is important to understand the origin or context of a local vocabulary or constraint. Two levels of contextual heterogeneity are: (i) intra-contextual heterogeneity, and (ii) inter-contextual heterogeneity. In an ontology-based information system (OBIS [32]) identifying context is crucial, notably to see which ontology, axioms, policies or constraints belong to which context. Similarly, the two broad categorisations of modularity are [18, 33]: (i) syntactic modularity: arranges ontologies in multiple compact modules, where ontology modules are well-controlled for more efficient ontology construction, revision and reuse, and (ii) semantic modularity: allows localised and contextual points of view over autonomous contributors of different ontology modules and distributed reasoning. While a limited form of syntactic modularity has

been supported by the current Semantic Web standards, semantic modularity is an open research problem.

The next section describes the thesis contributions and how each contribution addresses the defined research challenges.

1.3 Research Contribution, Evaluation and Impact

The main aim of this thesis is to enable semantic interoperability between HL7 applications. Our hypothesis:

"***An ontology-based integration framework can improve healthcare data interoperability by reducing: (i) the integration burden (per message) between heterogeneous healthcare applications; and (ii) the number of alignments between heterogeneous healthcare applications***"

requires development of an ontology-based integration framework which can be tested on a use-case highlighting various heterogeneities that exist in real-world healthcare applications.

The possible uses of ontologies in the healthcare domain are manyfold and vary depending on a use-case or application scenario. Our hypothesis attempts to claim two core uses (irrespective of specific application environment) of ontologies for the healthcare data integration. If the hypothesis would indeed hold, it would provide greater insight into the limitations of existing integration solutions and how ontologies could ease integration efforts for the healthcare domain.

The evaluation of the research results presented in this thesis is divided in three main parts: (i) qualitative: methods and tools are compared on the "features" they offer; (ii) quantitative: methods and tools are compared across various "measures"; and (iii) use-case: methods and tools are evaluated against a healthcare use-case demonstrating the potential value of an ontology-based integration framework.

There are three major aspects towards realising an ontology-based integration framework for the healthcare domain: (i) Ontology Building Methodology, (ii) Ontology Alignment Methodology, and (iii) Context-Awareness and Modularity for an Ontology-based Integration Framework. We now discuss the contribution, evaluation and impact of this thesis for each of the three aspects.

1. **Ontology Building Methodology**

   (a) **Contribution:** We propose an ontology building methodology—Plug and Play Electronic Patient Records (PPEPR) Methodology—for the HL7 standard (see Chapter 4). It can be used to ontologise HL7 compliant applications. Ontologies are developed in global and local spaces. The PPEPR methodology provides methods to arrange global and local ontologies into a separate layer and further align them for resolving semantic heterogeneities. We have followed top-down and bottom-up approaches to transform the syntactic HL7 format to a semantic format and vice-versa. We provide an automatic mechanism (see Chapter 5) to transform XML Schema/instances to ontology Schema/instances. The outcomes of the PPEPR Methodology are (i) a HL7 Version 3 global ontology consisting of 1563 classes, 200 object properties, and 100 datatype properties; and (ii) a

HL7 Version 2 global ontology consisting of 1062 classes, 682 object properties, and 50 datatype properties.

(b) **Evaluation:** We provide a feature-based comparison of the prominent ontology building methodologies describing how PPEPR Methodology extends and refines existing methodologies for the HL7 standard. We compare features (*e.g.*, reusability of existing resources, layering of ontologies, and local adaptation of ontologies) that the methodologies support as well as strategy, methods, and output of each methodology (see Chapter 4). We evaluate coverage of HL7 specifications by the ontologies build using the PPEPR Methodology (see Chapter 5). The evaluation results contribute to our first research challenge (*i.e.*, "how to build HL7 ontologies by reusing domain artefacts reside in separate global and local spaces") and shows how the PPEPR Methodology improves automation for ontologising HL7 specifications.

(c) **Impact:** We presented a preliminary methodology at the IEEE European Conference on Web Services (ECOWS) 2008 [34] and at the International Conference on Web Information Systems and Technologies (WEBIST) 2009 [35]. An extended version of the PPEPR Methodology is presented at the International Conference on Multidisciplinary Research and Practice for Business, Enterprise and Health Information Systems (MURPBES) 2011 [36].

2. **Ontology Alignment Methodology**

(a) **Contribution:** We propose a semi-automatic ontology alignment mechanism for HL7 ontologies that significantly improves the ontology matching results through the comparison of existing tools and mechanisms (see Chapter 6).

(b) **Evaluation:** We evaluate automated ontology matching tools for (i) aligning HL7 Version 3 ontologies; and (ii) aligning HL7 Version 2 and Version 3 ontologies. We compare "accuracy" (*i.e.*, precision) and "completeness" (*i.e.*, recall) of matching results at three threshold ranges: minimum range (0.1–0.4), medium range (0.4–0.7), and maximum range (0.7–1) (see Chapter 6). The evaluation results contribute to our second research challenge (*i.e.*, "how to automate alignment of HL7 ontologies with greater accuracy of correspondences retrieved") and shows how the PPEPR Methodology improves accuracy in aligning HL7 ontologies.

(c) **Impact:** We presented an ontology alignment methodology for the HL7 standard at the International Conference on Multidisciplinary Research and Practice for Business, Enterprise and Health Information Systems (MURPBES) 2011 [36].

3. **Context-Awareness and Modularity**

(a) **Contribution:** We provide a use-case driven detailed investigation and an original classification of the formal approaches on their ability to handle heterogeneity in the HCLS domain (see Chapter 8). The investigation resulted into a solution path that can handle undesirable inconsistencies caused by context-sensitive healthcare policies.

(b) **Evaluation:** We provide feature-based comparison of formal approaches on their ability to support: context-awareness, modularity, profile and policy management, correspondence expressiveness, and robustness to heterogeneity in a real-life use-case that requires

exchange of heterogenous patient records (see Chapter 8). The evaluation results contribute to our third research challenge (*i.e.*, "how to resolve inconsistencies caused by context-sensitive (or local) healthcare policies while mediating heterogeneous HL7 messages") that shows how the proposed solution path can enable context-awareness and modularity within an ontology-based integration framework (like PPEPR).

(c) **Impact:** We presented an investigation (Heterogeneity and Context in Semantic-Web-Enabled HCLS Systems) at the International Conference on Ontologies, DataBases, and Applications of Semantics (ODBASE) 2009 [37]. An extended version of the investigation including a solution path is presented in the book chapter (to appear 2012) for: Interoperability in Healthcare Information Systems: Standards, Management, and Technology [38].

The above mentioned three contributions are realised within an integration framework. We propose an ontology-based integration framework, called PPEPR, that resolves heterogeneity between HL7 Version 2 and Version 3 applications (see Chapter 7). We have automated most parts of the design-time operation and have fully automated the run-time portions. We measure the benefits of PPEPR (see Chapter 7). We measure: (i) the resources it takes to model messages and get them operational in PPEPR; (ii) the integration burden (comparing existing integration systems) on three dimensions namely, design-time development effort measured (per message), run-time efficiency of exchanging messages, and deployment methodology comparing traditional software deployment; and (iii) the number of bilateral correspondences created by existing integration framework and the PPEPR framework. We are also cognisant of the benefits of using PPEPR in any environment, where the models created for data mediation can subsequently be used in any HL7 compliant hospitals. The evaluation results contribute to two main objectives of our hypothesis, and shows how the PPEPR framework reduces the integration burden and alignments between heterogeneous HL7 applications. Evaluation presented in this thesis is based on a real-life use-case scenario (see Chapter 3).

We presented an initial version of the PPEPR framework at the Annual ACM Symposium on Applied Computing (SAC) 2008 [39]. We presented evaluation results at the eHealth 2008 Conference [40]. PPEPR is licensed by SlidePath[4] (now known as Leica Microsystems)– a healthcare data integration company. The outcomes of the PPEPR[5] framework contributed to the EU project RIDE[6], especially in the requirement-analysis for a EU-Wide semantically interoperable eHealth Infrastructure. PPEPR is currently used as an HL7 integration engine within a widget-based integration platform, called SQWELCH[7] that allows integrating clinical widgets.

In the rest of this thesis, we will show that our hypothesis holds for the HL7 compliant applications, however, methods and tools developed can be applied to other standards (*e.g.*, openEHR) with minor adjustments. This claim is based on the fact that healthcare standards like openEHR or CEN/ISO EN13606 share many commonalties in the way clinical messages are modeled in the HL7 environment [41].

The next section presents the overall structure of the thesis.

[4]http://www.slidepath.com/
[5]http://www.ppepr.com/
[6]http://www.srdc.metu.edu.tr/webpage/projects/ride/
[7]http://www.sqwelch.com/

1.4 Thesis Outline

Figure 1.3 presents the arrangement of thesis chapters. The remainder of this thesis is organised as follows:

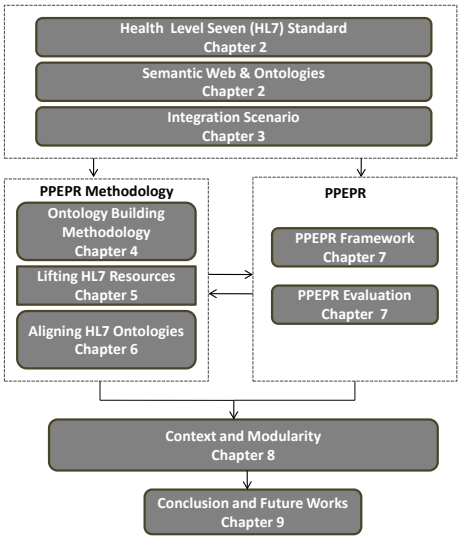

Figure 1.3: Thesis Structure

Chapter 2 presents background information about the Health Level Standard (HL7) standard, Semantic Web technologies, ontology languages, ontology building methodologies, and methods to deal with ontological heterogeneity. The chapter describes how information is arranged and represented within healthcare applications and briefly presents some of the existing integration approaches both at syntactic and semantic levels.

Chapter 3 describes an integration scenario for HL7 compliant hospitals. The scenario presents the case of a patient–*Sean*–who is an acute Type 2 diabetic and is routinely treated at the Galway University Hospital (GUH). The scenario identifies clinical messages exchanged between hospitals and labs which comply with Version 2 and Version 3 of the HL7 standard. The scenario describes all healthcare actors involved, the set of messages to complete a Lab-order for a diabetic Type 2 patient, and interoperability requirements for an integration solution.

Chapter 4 presents an ontology building methodology for the HL7 standard, called the PPEPR Methodology. The methodology identifies all the steps required to ontologise the HL7 standard. It emphasises on reuse of non-ontological resources, layering of ontologies, and the local adaptation of ontologies.

Chapter 5 presents the lifting task in the modelling phase within the PPEPR methodology. The majority of HL7 resources are available in UML and XML formats. The chapter discusses open issues in ontologising the UML and XML-based standard specifications. The chapter presents five guidelines to ontologise HL7 UML models. The chapter proposes an automatic mechanism to ontologise XML(S) specifications to corresponding HL7 ontologies.

Chapter 6 presents the ontology alignment mechanism we developed on these HL7 ontologies. The chapter describes how different data types, vocabularies, terminologies were aligned for HL7

Version 2 and Version 3 ontologies. The chapter discusses several issues raised while aligning concepts and properties with different names and structures. Finally, the chapter presents an evaluation of the proposed semi-automatic ontology alignment mechanism.

Chapter 7 presents the PPEPR framework. The chapter describes how HL7 ontologies are developed by following the PPEPR methodology. The chapter introduces the PPEPR architecture and components that perform transformation and mediation of HL7 messages. The chapter concludes with an evaluation of the PPEPR framework.

Chapters 8 explores the existing formalisms that deal with the difficult issues of representing and using modularity and contextuality within an ontological knowledge base. The chapter investigates formal approaches that relate directly or indirectly to Semantic Web technologies. The chapter concludes some formalisms that can be combined to realise context-awareness and modularity for HL7 ontologies. The chapter proposes a solution path towards allowing the arrangement of knowledge bases into separate layers and into identifiable modules.

Chapter 9 finally discusses open issues that are still not addressed by the presented approaches but are important to the interoperability of healthcare applications. This way we hope to offer a roadmap and directions for future research in the area of Semantic Web-enabled healthcare applications.

Chapter 2

Preliminaries

2.1 Introduction

The problem of combining several information sources has been a challenge in the IT domain for a long time. It gets particulary complicated in the healthcare domain where information models are very complex: the SNOMED clinical terms coding system alone defines more than 300,000 clinical concepts and there are many other coding systems, *e.g.*, Logical Observation Identifiers Names and Codes (LOINC) [42] containing significant amount of concepts. To address the EHR interoperability problem, several standards such as Health Level Seven (HL7), CEN/ISO EN13606 and openEHR have been developed. These standards aim to structure and markup clinical content for the purpose of clinical information exchange.

Information exchange is inevitable between healthcare applications because various parts (*e.g.*, drug, clinical, administrative, demographic) of a patient record exist in applications deployed across departments, hospitals, regions or countries. The average patient often presumes that no matter where he or she goes, every specialists have access to his or her complete medical record. This is not the case. Instead, in a hospital with forty departments there may exist forty specialist electronic patient record systems, some or all of which work in isolation. To enable interoperability between these systems, the hospital IT department typically has to employ a software programmer to develop interfaces between those systems. Consequently, the proliferation of interfaces and alignments may approach $n \times (n-1)$ where n is the number of healthcare applications in a hospital environment. In many cases this programming task is not achievable and the result is the movement of paper files between departments which have functioning IT systems. Healthcare interoperability across heterogeneous information systems shows an unprecedented challenge, and requires integration mechanisms that can meet the demands of distributed healthcare stakeholders.

Healthcare standards have been proposed to reduce the bilateral (*i.e.*, interface to interface) alignments. However, these efforts have resulted in different standards and each of them differ in their information models, overall messaging structure, terminologies, and vocabularies. The existence of several standards and versioning within a standard significantly increases the overall integration complexities. Presence of several heterogenous standards is the situation even worse than clinician changing standard specifications for the local messaging requirements.

The recent research and industrial initiatives to take up the interoperability issues at a semantic level has gained significant attention. The formal annotation of schema, data, and service definitions resolve the heterogeneity issues at the conceptual level. Semantic Web technologies [43] allow to describe and process such conceptual definitions and are considered as key technologies enabling further automation in overall data integration. This chapter first briefly describes Version 2 and Version 3 of the Health Level Seven (HL7) standard. Second, the chapters describes the limitations of existing HL7 integration tools in proving interoperability between Version 2 and Version 3 applications. Third, the chapter provides an overview of the Semantic Web and how the HL7 standard and applications can benefit from Semantic Web technologies. The chapter briefly explains the role of an ontology for the Semantic Web and some of the initiatives where ontologies play a key role in the interoperability of information systems. Fourth, the chapter provides an overview of ontology building methodologies. Finally, the chapter discusses ontological heterogeneity and causes of conceptual differences within ontological knowledge bases.

2.2 Health Level Seven (HL7) Standard

Health Level Seven (HL7) was founded in 1987. It is a non-profit ANSI accredited standard that develops healthcare information models and schemas for storing and exchanging information across the healthcare industry. The HL7 standard covers both clinical and administrative aspects of the healthcare industry, including laboratory, clinical genomic, medical records, patient care, pharmacy, public health reporting, regulated studies, accounts and billing, claims and reimbursement, patient administration, personnel management and scheduling.

There are two major versions of the HL7 standard, Version 2 and Version 3. The HL7 Version 2 messaging standard is the most widely implemented healthcare standard in the world today. However, being HL7 Version 2 compliant does not imply direct interoperability between healthcare systems. This stems from the fact that Version 2 messages have no precisely defined underlying information model; the definitions for many data fields are rather vague and there is a multitude of optional data fields. This vagueness and optionality provides great flexibility, but necessitates detailed bilateral agreements among healthcare systems to achieve interoperability. To remedy this problem, HL7 has developed Version 3, which is based on an object-oriented information model, called the Reference Information Model (RIM) [4]. We now describe both version of the HL7 standard in more detail.

2.2.1 HL7 Version 2

The HL7 Version 2 standard was developed in 1989. HL7 Version 2 is based on Electronic Data Interchange (EDI) format [3] but specifications are also available in XML format. Version 2 was developed with the assumption that an event in the healthcare world, called the "trigger event", causes the exchange of messages between a pair of applications. When an event occurs in a HL7 compliant system, a HL7 message is prepared by collecting the necessary data from the underlying application systems and is passed to the requestor. For example, a trigger event, called $ADT\hat{}A01$—Admission/Discharge/Transfer (admission of an in-patient into a facility)—occurs when a patient

is admitted to a facility and this may cause the data about the patient to be collected and sent to interacting applications.

The Version 2 messaging environment consist of segments, data fields, and data types. Segments are separated by the "Carriage Return" character. Each segment begins with a three-character literal value (*e.g.*, MSH, PID, OBX) that identifies its use within a message. Segments may be defined as required or optional and may be permitted to repeat. Each segment is further divided into data fields that are of variable length and separated by the pipe (|) character. Individual data fields are found in the message by their position within their associated segments. Each data field is associated with a datatype that permits the type of message allowed to be stored and can be further subdivided using the caret (^) symbol. Messaging rules are described on how the various data types are used within a field and when an individual data field may be repeated.

The Listing 2.1 shows an example Version 2 message consists of three segments: (i) MSH: Message Header, (ii) PID: Patient Identification, and (iii) OBX: Observation.

```
MSH|ORL|RemotePatient|PAT−R|Vital−Observation|GUH−L|20080215T0730|GUH−CNTRL−9436|2.5 <cr>
PID|1|353−91−49678|Sean Murphy|John Colleen|19720520|M|13 Rahoon park^Upper Newcastle^Galway|(091)444−534 <cr>
OBX|1|ST|1554−5^GLUCOSE^POST 12H CFST:MCNC:PT:SER/PLAS:QN^LN||182|mg/dl|70−105| <cr>
```

Listing 2.1: HL7 Version 2 Glucose Test Observation: EDI format

The first data field of the MSH segment defines the message type and the type of trigger event that caused the message to be sent. For example, the MSH segment with *ORL* as the first data field value indicates general laboratory order response message. Other data fields of the MSH segment describe that the patient is at a remote location where the message sending application is PAT-R (application that ordered the lab observation). Patient Vital observation has been ordered from the Lab called GUH-L (application that will fulfill the original request). Further, the date of the lab observation (20080215T0730), department specific code (GUH-CNTRL-9436), and subversion (2.5) of Version 2 are mentioned within the MSH segment.

The second segment (PID) describes the patient details, identification scheme (1), patient contact number (353-91-49678), name (Sean Murphy), physician responsible (John Colleen), physician id (19720520), gender (M), address, and the physician contact number. Similarly, the third segment OBX describes, number of observation performed (1), observation value type (ST-String Data), observation identifier (1554-5 described by external vocabulary LOINC) describing clinical parameters (GLUCOSE^POST 12H CFST:MCNC:PT:SER/PLAS:QN^LN) also defined by the LOINC (LN) vocabulary. The outcome of the lab observation is a glucose value of 182 mg/dl where the normal reference range is 70-105 mg/dl.

Listing 2.2 shows a snippet of XML equivalent of the OBX segment from Listing 2.1. A special API, called *HAPI*[1], is developed and supported by Version 2 to convert EDI to XML format.

```
1 <?xml version="1.0" encoding="UTF−8"?>
2 <HL7Message>
3     <OBX>
4         <OBX.1>
5             <OBX.1.1>1</OBX.1.1>
6         </OBX.1>
7         <OBX.2>
8             <OBX.2.1>ST</OBX.2.1>
```

[1]http://hl7api.sourceforge.net/

```
 9          </OBX.2>
10          <OBX.3>
11              <OBX.3.1>1554−5</OBX.3.1>
12              <OBX.3.2>GLUCOSE</OBX.3.2>
13              <OBX.3.3>POST 12H CFST:MCNC:PT:SER/PLAS:QN</OBX.3.3>
14              <OBX.3.4>LN</OBX.3.4>
15          </OBX.3>
16          <OBX.4/>
17          <OBX.5>
18              <OBX.5.1>182</OBX.5.1>
19          </OBX.5>
20          <OBX.6>
21              <OBX.6.1>mg/dl</OBX.6.1>
22          </OBX.6>
23          <OBX.7>
24              <OBX.7.1>70−105</OBX.7.1>
25          </OBX.7>
26      </OBX>
27  </HL7Message>
```

Listing 2.2: HL7 Version 2 Glucose Test Observation: XML format

In order to construct messages as in Listing 2.2, Version 2 provides two sets of schemas: (i) *coreSchemas*: a primary set of schemas containing HL7 message building blocks such as datatypes, segments, and data fields; (ii) *message schemas*: templates provided from HL7, such that hospitals extend them to meet their local messaging requirements. As discussed, Version 2 allows optionalities and flexibilities for the construction of clinical messages. For example, a hospital can decide to use seven data fields for the OBX segment (as in the Listing 2.2) or more, keep some data fields empty (*e.g.*, ⟨OBX.4⟩), or use four sub-data fields instead of one (*e.g.*, ⟨OBX.3⟩). Version 2 has provisions within the specifications to add messages or portions of messages that are local to an institution. This means that there will be a wide variety in the manner in which the standard is applied in different institutions [44]. Also, in XML an element positioning and its identification is limited or sensible within the containing document. The absence of global descriptive identification for XML messaging elements makes difficult integrating messages originating from several institutions. Chapter 5 describes in detail about these two types of schemas (*coreSchemas*, *message schemas*) and other features within the Version 2 schemas allowing such local changes.

2.2.2 HL7 Version 3

HL7 Version 3 development started in 1995 resulting in the initial draft version (first release) in 2003 and standard normative edition in 2005. The main aim of Version 3 is to offer greater consistency and structure in the clinical messages compared to Version 2. The Version 3 specification, as opposed to Version 2 (EDI and XML formats), is based on object-oriented principles. The HL7 Version 3 specifications are centered around a static model, called Reference Information Model (RIM) which covers all domains of the healthcare industry. The RIM defines all data structures, data types and vocabularies, as well as the relationships between them. The RIM is based around six core types of classes and includes rules governing how they relate to each other.

Table 2.1 briefly describes the RIM classes and their relationships. Three of these classes - Act, Entity and Role - are further represented by a set of specialised classes in the domain or local models. For example, specialisations of the Act class include *Observation*, *Procedure* and *SubstanceAdministration*. As can be seen by the nature of the RIM classes, they take an action-centered view (similar to the Version 2 event-based approach) with processes and information in

RIM Classes	Description	Example
Act	an action of interest	observation, referral
Entity	a class or instance of a specific thing capable of participating in Acts	sean, galway university hospital (GUH)
Role	an entity, in a particular Role, can participate in an Act	patient, physician
Participation	an association between an Act and a Role with an Entity playing that Role	author, performer
ActRelationship	an association between a pair of Acts	act composition, act revision
RoleLink	a connection between two roles expressing a dependency	role composition, role dependency

Table 2.1: Reference Information Model (RIM) Classes

healthcare represented primarily in terms of the clinical acts. The scope of RIM is global, which means it is inherited by all Version 3 complying hospitals.

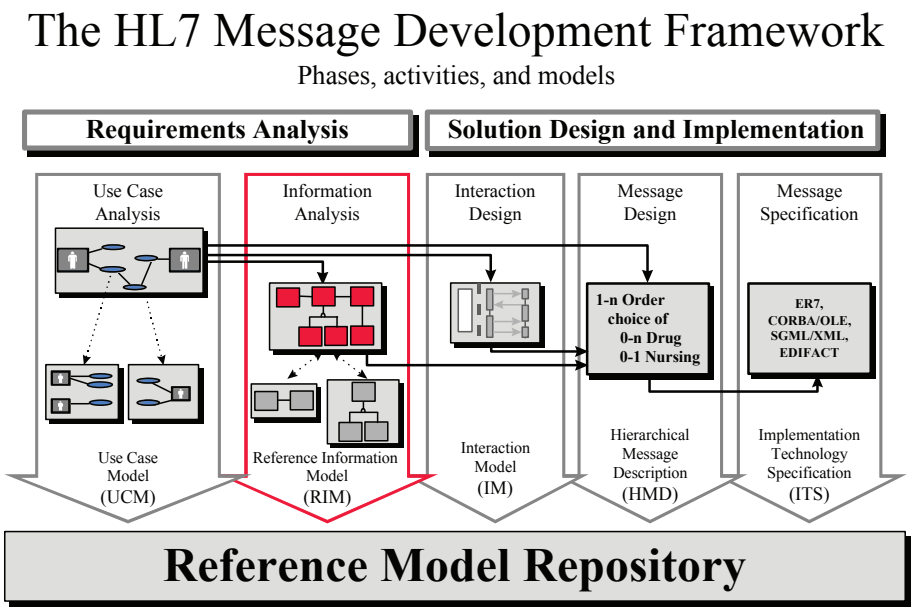

Figure 2.1: HL7 Message Development Framework [1]

In addition to the RIM, two local interrelated types of information models are the Domain Message Information Model (DMIM) and Refined Message Information Model (RMIM). The DMIM is a local model (a refined subset of the RIM) and is used for modelling a particular domain (*e.g.*, Lab, Hospital Admin). The RMIM is a subset of a DMIM and is used to express the information content for a message or set of messages (*e.g.*, Lab Observation Order Message). All three interrelated models use the same notation and have the same basic structure but differ from each other in their information content, scope (context), and intended use.

Figure 2.1 shows the message development framework for HL7 Version 3. The development process starts from the left. The use case scenario, where domain experts describe a storyboard about all related actors, messages exchanged, and applications functionalities. The use case scenario then results in a use case model, which results in two interconnected models, one for information modelling and the other for application interactions (or functionality) modelling. As described, RIM is the centre of all local domain models and each local domain model inherits the RIM classes and

attributes. The Version 3 interaction model is analogous to message exchange patterns (MEPs) in Web services [45]. The interaction model is a step-by-step description of all the clinical events (or trigger events) that fit within the scope of an application scenario. Finally, the message development framework concludes (on the right of Figure 2.1) with the schema level descriptions of the domain information model and the interaction model. In this thesis we focus mainly on the domain information model.

Figure 2.2 shows the three layers of Version 3 specifications. The top layer describes the semantics of healthcare artefacts, the middle layers uses an information modelling language like UML for representing those healthcare artefacts (*e.g.*, RIM, DMIM, RMIM), and finally an implementation technology like XML is used for the actual construction and exchange of clinical messages. Version 3 uses a Formal Data Type Definition Language (FDTL) [46] to express the semantics of vocabularies and data types at the top layer. HL7 provides a set of tools[2] creating these three layers as well as migrating from the middle layer to the bottom layer. For example, the top layer describes the semantics of a postal address which defines that the AD (alias of postal address) class specialises from ANY, LIST⟨ADXP⟩ classes, which means that a list of ADXP classes are specialised by the AD class. The middle layers uses UML-based RIM to represent the AD class. The bottom layer is an XSD snippet for the AD class.

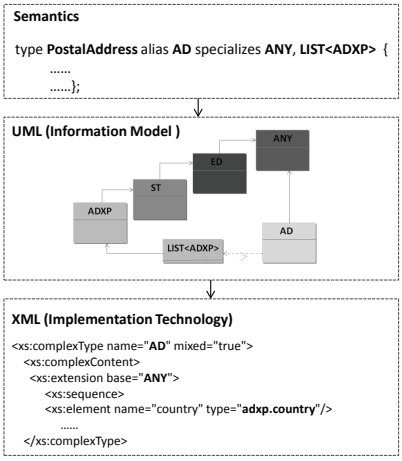

Figure 2.2: HL7 Version 3: Semantics to Implementation Technology

We notice that the use of semantics and conceptual models (top and middle layers) is limited to the construction of message schemas during the design time. If every healthcare institution uses these upper two layers centrally, this will result in a consistent set of messages and a reduction in heterogeneity between Version 3 applications. However, information model (middle layer) itself is never used during the exchange of messages or for integration purposes during runtime. This limits integration efforts at the implementation level (*i.e.*, XML technology).

Figure 2.3 shows a snippet of the RMIM model for an observationEvent class. The observationEvent corresponds to the OBX segment of Version 2. Listing 2.3 shows the OBX equivalent (Listing 2.2) Version 3 message. We notice that the observationEvent is described as a class containing various attributes instead of segments or data fields in Version 2. Version 3 provides tools

[2]http://gforge.hl7.org/

support to generate XSDs from a RMIM model.

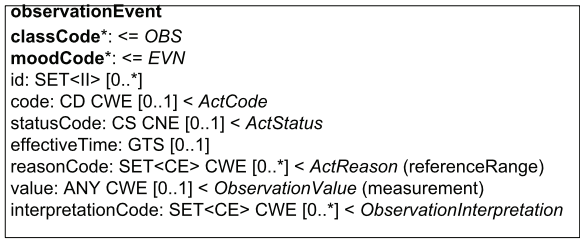

Figure 2.3: observationEvent class within a RMIM

Similar to Version 2, XSDs generated from the Version 3 information models are categorised in two types: (i) *coreSchemas*: XML version of the UML-based RIM describing all data types, vocabularies, and codes; (ii) *message schemas*: XML version of the UML-based RMIM describing local schemas that use the *coreSchemas* (data types, vocabularies, and codes) for a message construction. Chapter 5 describes in detail these schemas and how they can be used for creating formal conceptual models.

```
1  <?xml version="1.0" encoding="UTF−8"?>
2  <observationEvent classCode="OBS" moodCode="EVN">
3     <id root="2.16.840.1.113883.19.1122.4" extension="1045813"
4         assigningAuthorityName="GUH−L"/>
5     <code code="1554−5" codeSystemName="LN"
6           codeSystem="2.16.840.1.113883.6.1"
7           displayName="GLUCOSE^POST 12H CFST:MCNC:PT:SER/PLAS:QN"/>
8     <statusCode code="completed"/>
9     <effectiveTime value="20080215T0730"/>
10    <measurement xsi:type="PQ" value="182" unit="mg/dL"/>
11    <referenceRange>
12          <low value="70" unit="mg/dL"/>
13          <high value="105" unit="mg/dL"/>
14    </referenceRange>
15 </observationEvent>
```

Listing 2.3: HL7 Version 3 Glucose Test Observation: XML format

We notice in the Version 2 message (Listing 2.2) and equivalent Version 3 (Listing 2.3) message that establishing correspondences for the messaging elements is necessary to exchange them between Version 2 and Version 3 applications. For example, OBX corresponds to the observationEvent class, OBX.3.1 corresponds to code attribute, OBX.3.2 and OBX.3.3 (combined) corresponds to displayName attribute, OBX.3.4 corresponds to codeSystemName attribute, etc. It is important to mention that Version 3 is expressive in terms of messaging elements and overall messaging structure, which might lead to a message loss while transforming a Version 3 message to a Version 2 equivalent. For instance, Version 3 global attributes (Listing 2.3) like id, extension, interpretationCode, etc. do not have Version 2 counterparts. On the other hand, if Version 2 compliant hospitals does not use local elements and attributes, Version 2 messages can be transformed completely to Version 3 equivalents. Next, we discuss some important HL7 integration systems that are used in HL7 complying healthcare institutions.

2.3 Healthcare Terminologies and Vocabularies

Healthcare terminologies and vocabularies such as SNOMED, LOINC, or RxNorm [47] describe medical concepts. When these concepts are placed in a clinical record they become associated with an observation (*e.g.*, observation measurement), time (*e.g.*, effective time), policy (*e.g.*, drug policy), and relationships with other records. These associations provide meaning to clinical messaging elements. For example, (i) a clinical *observation* concept (*e.g.*, blood sugar test) has an effective time during which it is valid, and (ii) a diabetic concept placed in a "family history" segment of a patient record does not imply that the individual patient is a diabetic patient.

These terminologies and vocabularies describe specialised sub-domains of the healthcare, for instance, SNOMED and LOINC are preferred for the laboratory domain and RxNorm for the drug domain. Some of these vocabularies contain overlapping definitions for the same concept. For example. In Listing 2.3 the lines 5-7 use LOINC description (1554-5) for glucose blood test, other hospitals can use the SNOMED concept (SNOMED:52302001) for similar purpose. While exchanging patient's records between hospitals their alignment (of overlapping vocabularies) remain the crucial task.

As said, standard compliant patient records are constructed from the combination of messages derived from an information model and several terminologies or vocabularies referring to message attributes. However, many healthcare institutes do not use the standard specific information model, rather their messages are constructed using local models and vocabularies.

2.4 HL7 Integration Systems

In recent times several industrial initiatives have provided integration solutions for HL7 applications deployed across hospitals. Some of them provide specialised interfaces for exchanging messages within the same version (primarily Version 2 applications) and only few propose solutions for exchanging messages between Version 2 and Version 3. iWay HL7 Adapter[3], MD link[4], iNTERFACEWARE Iguana[5], CorePointHealth[6], and QUOVADX[7] provide adapters (or interfaces) that allow exchanging messages between Version 2 applications. HL7Connect[8] and Mirth[9] are among the few that offer solutions for exchanging Version 2 and Version 3 messages. Figure 2.4 shows functioning of the Mirth Integration System.

Mirth is an open source cross-platform HL7 interface engine that enables bi-directional sending of HL7 messages between systems and applications over multiple transports. Mirth allows messages to be filtered, transformed, and routed based on user-defined rules. Similar to the Mirth functioning, in general HL7 integration systems take the approach of creating bilateral correspondences between two heterogenous messages, called message-centric approach. For example, in case of Listing 2.2

[3]http://www.iwaysoftware.com/products/adapters/hl7
[4]http://www.mdisolutions.com/products/md-link-integration-engine.html
[5]http://www.interfaceware.com/iguana.html
[6]http://www.corepointhealth.com/products/hl7-analyzer/hl7-analyzer
[7]http://www.quovadx.com/
[8]http://www.hl7connect.com/
[9]http://www.mirthcorp.com/

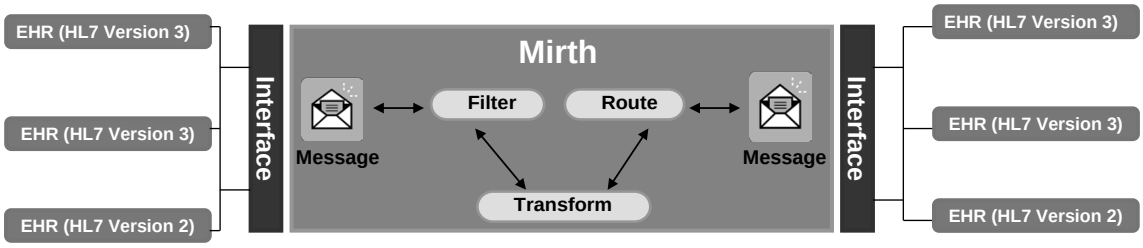

Figure 2.4: Mirth Integration System

(Version 2) and Listing 2.3 (Version 3) messages, Mirth allows domain experts to create custom rules (*i.e.*, bilateral correspondences) describing the similarity of messaging elements (*e.g.*, OBX, observationEvent). This means that conceptual models (*i.e.*, UML models) and the corresponding *coreSchemas* are excluded from the integration process. This exclusion of the conceptual models results in a quadratic size alignments between healthcare applications where dedicated (one-to-one) interfaces between pairs of healthcare applications are created for message exchanges.

Figure 2.5 (a) shows a topology resulting from the quadratic size alignments between healthcare applications. This topology is generally adopted by the existing HL7 integration systems. Figure 2.5 (b) shows an ideal situation where heterogeneity between messages are managed at the central location (circle symbol within Figure 2.5). This centralised approach reduces the alignments to n times.

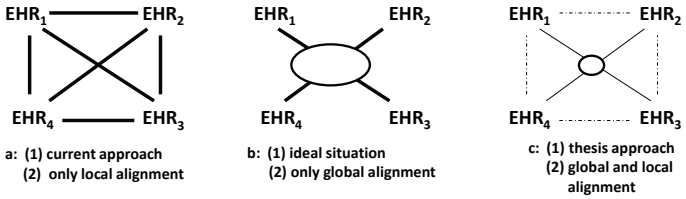

Figure 2.5: Topologies for Healthcare Applications Exchanging Patient Records

Figure 2.5 (c) shows the approach taken in this thesis; a hybrid topology which includes the central integration location and bilateral alignments are limited to the local parts of conceptual models. Considering the diversity of the healthcare domain, the use of local models (schemas) cannot be completely avoided. In other words, local clinical messaging requirements originating from various clinical sites cannot be predefined in a central conceptual model. Therefore, we have established a clear distinction between the global and local parts of healthcare information models. In Figure 2.5 (c) central conceptual models (*e.g.*, RIM and corresponding *coreSchemas*) and their alignments are managed at the central location (the smaller circle in Figure 2.5 (c)). Heterogeneity of local models or message schemas and their alignments are dealt at the local level (the broken lines in Figure 2.5 (c)). This way, we aim to reduce the bilateral correspondences between healthcare applications by delegating the majority of mediation to the central integration location.

Considering that HL7 provides a rich set of semantics for the healthcare domain, it makes sense to use the top and middle layers (Figure 2.2) not only for Version 2 and Version 3 interoperability but integration with the huge set of medical or bio medical vocabularies used within the HL7 standard.

The presence of different healthcare standards, large scale applicability, and limitations of syntactic integration solutions, motivated the healthcare specialists to apply Semantic Web technologies to resolve heterogeneity in a formal and consistent way. Healthcare domain experts have advocated in favor of using Semantic Web technologies as a "common medium" where the middle layer (information models) and lower layer (data) can be engaged with each other [48, 49, 50] during the integration process. The mutual use of Semantic Web technologies as a "common medium" between middle and lower layers would provide computable semantics of the information models, facilitating the model reuse, model harmonisation, and data integration; which is lacking in the existing integration systems. Standardisation groups like the W3C HCLS Interest Group[10] and various research projects have taken initiatives to use Semantic Web technologies to represent healthcare information models and their integration with HCLS terminologies and vocabularies [5, 6].

Next we describe the Semantic Web and how it can benefit in realising interoperable healthcare applications.

2.5 Semantic Web (SW)

The Semantic Web [43] was proposed as an online, interlinked Web of machine-readable information. The vision of the Semantic Web is that of a "global database" containing structured information. In contrast to traditional databases, the information in the Semantic Web does not conform to a single schema and the information can be contradicting, inconsistent or incomplete. Finding and combining information are the main features of the Semantic Web. Additionally, applications must also be able to find and combine services on the Web to make the Web of services scale together with the Web of data.

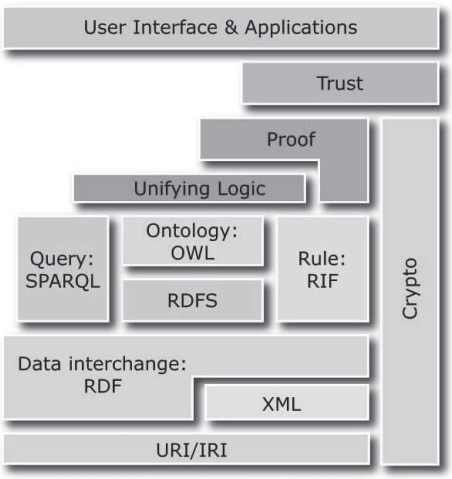

Figure 2.6: The Semantic Web Layer Cake

Figure 2.6 shows the Semantic Web Layer Cake[11] which is a hierarchy of languages and technologies, where each layer exploits and uses capabilities of the layers below. The bottom layer, Uniform

[10] http://www.w3.org/2001/sw/hcls/
[11] http://www.w3.org/2007/03/layerCake.png

Resource Identifier (URI) [51] follows the important features of the existing Web. A URI is a string that is used to uniquely identify Web resources. Internationalized Resource Identifiers (IRI) [52] extend URIs by Unicode [53] support allowing use of characters from a large range of writing systems in identifiers. The Extensible Markup Language (XML) [20] layer with XML namespace and XML Schema definitions ensures a common syntax is used in the Semantic Web. XML is a general purpose markup language for documents containing structured information. An XML document contains elements that can be nested and that may have attributes and content. XML namespaces allow to specify different markup vocabularies in one XML document. XML Schema serves for expressing schema of a particular set of XML documents. HL7 uses the XML layer for describing message schemas and instances.

Healthcare resources and/or resources in general on the Web are distributed. Consequently, the description of resources should be encoded in a way that facilitates integration from a large number of sources. On top of the URI referencing mechanism and XML document exchange mechanism, the Resource Description Framework (RDF) [54] layer provides a graph-structured data format to encode descriptions about Web resources. RDF graphs can be serialised in multiple ways; one of the most commonly used is the XML serialisation. Having interlinked RDF data sets, mechanisms for querying the RDF data are necessary. SPARQL (a recursive acronym for SPARQL Protocol and RDF Query Language) [55] is a declarative query language, which allows for specifying queries against data in RDF. RDF graph based encoding of Web resources covers only parts of the meaning of the data. Often, constructs to model class or property hierarchies provide machines and subsequently humans a more precise understanding of data. To more comprehensively model a domain of interest, ontology languages can be employed. RDF Schema (RDFS) [56] is a language which can be used to express such domain knowledge.

Since complexities of domains may vary and a complex domain like the healthcare may require advanced knowledge constructs, the Web Ontology Language (OWL) [57] can be used for modelling such complex domains. Ontology languages allow for automated inferences, *i.e.*, drawing conclusions based on existing facts. An alternative way to specifying logical inferences is via rules. Often, users require means to express logical rules which transform data or enrich data with additional specifications. The Rules Interchange Format (RIF) [58] allows for encoding and exchanging such logical rules. It is expected that all the semantics and rules will be executed at the layers below Proof and Trust where results will be used to prove deductions. For reliable inputs, cryptography means are to be used for verifying the origin of the sources. On top of these layers, an application with a user interface can be built. The Semantic Web stack is still evolving as the layers are concretised.

The Sematic Web layer cake can improve HL7 interoperability efforts by (i) reducing the gap between HL7 information model (corresponds to SW ontology layer) and data representation (corresponds to SW data interchange layer) technologies; (ii) using HL7 information models and data for intelligent knowledge processing like reasoning, alignments, evaluation, etc; and (iii) providing means to achieve interoperability at the Web scale.

We now provide details on above mentioned Semantic Web technologies employed for the work presented in this thesis. Additionally, we describe the Web Service Modelling Language (WSML) [59] used for building HL7 ontologies in parallel to the OWL version. In the following chapters, we present ontology examples in OWL and WSML versions.

2.5.1 Semantic Data Model- The Resource Description Framework (RDF)

The Resource Description Framework (RDF) [54] is a knowledge representation language for the Semantic Web. It is a simple assertional logical language which allows the specification of binary properties or predicates in the Semantic Web. The assertions, also called triples, are statements expressing that some resource (entity in the Semantic Web) is related to another entity or a value through a predicate. The resources and predicates are denoted by Uniform Resource Identifiers (URIs), which are the natural means to represent and share knowledge in the Semantic Web. RDF has an intuitive graph model and semantics [60] with appropriate notion of entailment. The atomic constructs in RDF are subject (s)-predicate (p)-object (s) statements, a so-called RDF triple where:

- s is the subject node, which can be either a URI or a blank node. Blank nodes are local identifiers to a given graph and cannot be referenced from outside the graph.
- p is the predicate arc, a property identified by a URI.
- o is the object node, which can be either a URI, a blank node or a literal.

A set of RDF triples forms an RDF graph, which can be viewed as a directed labelled graph and consists of nodes and labelled directed arcs that link pairs of nodes. The RDF specification includes a special property *rdf:type* which is used to model instances (*e.g.*, Sean) and classes (*e.g.*, the Patient class). One restriction for RDF triples is that the subject nodes cannot be labeled with literals, plain or typed. Healthcare data can be expressed as a set of RDF triples. Listing 2.4 shows the Version 3 observationEvent (see Listing 2.3) represented in the RDF Turtle format [61].

```
1  @prefix LN: <http://www.loinc.org/lab/codes#>.
2  @prefix v3: <http://www.hl7.org/v3/ontology#>.
3  @prefix owl: <http://www.w3.org/2002/07/owl#>.
4  @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
5
6  <&hospitalA;lab1#observationEvent> v3:assigningAuthorityName "GUH-L";
7    v3:classCode "OBS";
8    v3:code "LN:1554-5";
9    v3:codeSystem "2.16.840.1.113883.6.1";
10   v3:codeSystemName "LN";
11   v3:displayName "GLUCOSE^POST 12H CFST:MCNC:PT:SER/PLAS:QN";
12   v3:effectiveTime "20080215T0730";
13   v3:extension "1045813";
14   v3:measurement "182";
15   v3:moodCode "EVN";
16   v3:referenceRange _:bnode1486314560,
17     _:bnode733772480;
18   v3:root "2.16.840.1.113883.19.1122.4";
19   v3:statusCode "completed";
20   a <http://hl7.org/v3/ontology/ActLabObservation>.
21 _:bnode1486314560 v3:unit "mg/dL";
22   v3:value "105";
23   a v3:high.
24 _:bnode733772480 v3:unit "mg/dL";
25   v3:value "70";
26   a v3:low.
```

Listing 2.4: HL7 Version 3 Glucose Test Observation: RDF Turtle format

In Listing 2.4 the entity observationEvent (line 6) and its relation with a class ActLabObservation (line 20) are described followed by various properties of the observationEvent (*e.g.*, classCode, moodCode, root). The root property (line 18) describes the OID value for the observationEvent entity, which provides an unique identity for the observationEvent with the HL7 information space.

Main advantages of using RDF over XML for representing healthcare data are (i) descriptive global identifiers: healthcare entities can pose global identifers (*i.e.*, URIs) which gives a basic ground to integrate data originating from wide variety of sources, *i.e.*, aggregating disparate data sources as if they came from a single source (ii) common framework: RDF provides a common framework to represent both instance data and the model or schema that describe them. For instance, in Figure 2.5 the HL7 three layers are described in three different formats (FDTL, UML, XML) which makes the situation even more complex on top of the already heterogeneous healthcare domain.

2.5.2 SPARQL Query Language

The SPARQL Query Language for RDF [55] enables the execution of queries against RDF datasets and handling of complex queries over data stored in RDF repositories. The RDF repositories supporting SPARQL must implement querying of the underlying data using a specific syntax and protocol.

```
1 PREFIX rdf: <http://www.w3.org/1999/02/22−rdf−syntax−ns#>
2 PREFIX v3: <http://www.hl7.org/v3/ontology>
3 PREFIX LN: <http://www.loinc.org/lab/codes>
4 PREFIX hospitalA: <http://www.hosA.org/v3/labobservation>
5
6 SELECT ?labobservation
7 WHERE { ?labobservation v3:classCode "OBS" .
8         ?labobservation v3:code LN:1554−5 } .
```

Listing 2.5: SPARQL query for all lab observations that measure glucose level

A SPARQL query consists of parts that define different aspects of the query. For example, Listing 2.5 presents a SPARQL query finding all lab observations that measure glucose level. The code "1554-5" (line 7) is a LOINC code denoting a glucose test. PREFIX is used for abbreviating URIs. The SELECT keyword specifies the information users are interested in. The WHERE clause is the core of SPARQL, where users define the exact graph pattern that has to be matched against the knowledge base. A basic graph pattern consists of triple (subject, predicate, object) patterns which are joined by variables (*e.g.*, ?labobservation), forming a template filled during the matching process. Optionally, FILTER expressions can be used to narrow the returned results only to the structures that fulfill specific criteria.

SPARQL can be implemented over several kinds of graph repositories. Popular repositories include Sesame [62], Jena [63], Virtuoso [64], BigData [65], OWLIM [66] or RDF-3X [67]. Some of the repositories do not only provide storage and query access for RDF graphs, but also support querying with use of inference and rules. With the recently proposed version of SPARQL 1.1 [68], new features are introduced in the language, which include mainly aggregate functions (such as minimum, maximum, sum), sub-queries (nesting of SPARQL queries), and negation.

2.5.3 RDF Vocabulary Description Language (RDFS)

RDF can be used for expressing medical or patient data. However, there is a need for modelling domain knowledge which can provide understanding of the RDF datasets. An ontology provides an explicit, machine-processable conceptualisation of shared and reused knowledge. An ontology has been defined as "a formal, explicit specification of a shared conceptualisation" [9]. The purpose

of an ontology is to provide a common understanding of a given area by both computers and humans. An ontology provides a hierarchy of concepts, their attributes and relationship between these concepts that can be observed in the given area of discourse. Consequently, ontology languages allow more expressive propositions to be expressed, like those that represent general knowledge rather than specific patient data elements. The relationships between heterogeneous healthcare data and knowledge can be formally expressed using ontology constructs.

The RDF Vocabulary Description Language (RDF Schema or RDFS) [56] is an extension of RDF providing a basic type system adapted for the Semantic Web framework. It introduces the notions of class and property and provides mechanisms for specifying class hierarchies, property hierarchies and for defining domains and ranges of properties. A distinguishing feature of the class model of RDF Schema is the separation of the intent of the class from its extent (the set of instances), allowing a class to be a member of its set of instances as well as application of properties to themselves.

The *rdfs:subClassOf* construct can be used to model class hierarchies. For example, ActLabObservation (line 20 of the Listing 2.4) is a subclass of ActObservation, where ActObservation is a subclass of Act class within the RIM structure. Such a construct allows a reasoner to deduce that instances of ActLabObservation are also of type ActObservation and Act. In Listing 2.4, given a fact stating that observationEvent is a ActLabObservation, a reasoner can draw the conclusion that observationEvent is also of type ActObservation and Act and applications that query for all ActObservation or Act also get observationEvent as a query result even if the instance does not explicitly contain that type assertion (as in Listing 2.4). Similarly, RDFS allows property hierarchies using the construct *rdfs:subPropertyOf* as well as defining the domain (*rdfs:domain*) and range (*rdfs:range*) of properties relating two or more classes based on certain properties.

Healthcare resources need expressive constructs for modelling domain artefacts. In Listing 2.4, we observe that properties need to express time (*e.g.*, *effectiveTime*), complex measurements with specific data unit (*e.g.*, *unit*), range (*e.g.*, *high*, *low*) or may be creating an equivalence correspondence for LOINC code (*e.g.*, 1554-5) with other medical vocabularies like SNOMED which might have been used in the patient records of a different hospital. Next, we describe briefly the Web Ontology Language (OWL) which provides additional constructs for creating complex classes and properties.

2.5.4 Web Ontology Language (OWL)

The Web Ontology Language (OWL) [57] is a family of knowledge representation languages for encoding the domain knowledge. The OWL syntax can be encoded in different RDF serialisations (*e.g.*, RDF/XML [69], Turtle [61]) as its exchange syntax. Hence, OWL shares many common features with RDF, such as the use of URIs for the unambiguous reference of Web resources. Compared to RDFS, OWL provides increased expressivity in class and property constructs, equivalence relation, cardinality constraints, increased datatype expressivity, meta-modelling ability, and database-style key identifiers. For example, Listing 2.6 shows a data property definition for the *effectiveTime* (Line 12 of Listing 2.4) property where the range is of `xsd:dateTime` type. OWL support for the majority of XML Schema datatypes allowing the Version 3 *effectiveTime* property to store value of type `xsd:dateTime`. The `xsd:dateTime` type is a sequence of integer-valued year, month, day, hour

and minute values. The Listing 2.4 snippet uses a non-logician syntax, called the OWL Manchester Syntax [70], which produces a less verbose syntax. In the following chapters we use the Manchester Syntax for presenting our examples.

```
1 DataProperty: effectiveTime
2     Domain: ActLabObservation
3     Range: xsd:dateTime
```

Listing 2.6: Extended Data type support (*e.g.*, `xsd:dateTime` in OWL)

OWL support for advanced knowledge constructs such as "Property Chain Inclusion" among others (meta-modelling ability, database-style key identifiers, property qualified cardinality restrictions, etc.) [71] would help domain experts to model some of the complex entities of the healthcare domain. For example, many properties in SNOMED use property chains to define a property in terms of a chain of properties that connect resources [72]. Listing 2.7 shows the SNOMED property "DirectSubstance" which describes the substance on which a lab procedure method directly acts. For example, "injection of Insulin" is a lab procedure where Insulin is a value for "DirectSubstance" that is injected. The property "DirectSubstance" forms a chain with other SNOMED property "HasActiveIngredient". The "HasActiveIngredient" indicates the active ingredient of a drug product, *e.g.*, NOVOLIN NPH 100 UNITS (Insulin drug ingredient). The property chain in Listing 2.7 describes that injection of a DirectSubstance (Insulin) is an injection of an active ingredient (NOVOLIN NPH 100 UNITS).

```
1 ObjectProperty: snomed:DirectSubstance
2
3     Annotations:
4         rdfs:label "Substance on which the lab procedure method directly acts"
5
6     SubPropertyChain:
7         snomed:DirectSubstance o snomed:HasActiveIngredient
```

Listing 2.7: Property Chain Inclusion support in OWL: SNOMED "DirectSubstance" property

As said before, Version 3 applications use various medical terminology and coding systems outside the HL7 specifications. These vocabularies contain overlapping definitions for the same entity. For example, LOINC code 1554-5 is used to describing a glucose test, similarly, other HL7 application may use SNOMED code 52302001 for the glucose test. Listing 2.8 shows a similarity relation using the OWL *SameAs* construct, which specifies that the two resources (LN:1554-5, SNOMED:52302001) are identical.

```
1 Individual: LN:1554−5
2     SameAs: SNOMED:52302001
```

Listing 2.8: SameAs Relation in OWL

This way, the *SameAs* construct would be helpful for applications to consolidate information about a glucose test from multiple sources. Such cross-domain references would be helpful in exchanging patient records across a variety of healthcare applications. The OWL supports three profiles (or sublanguages) [73]:

- **OWL 2 EL** is suited for applications requiring very large ontologies, *i.e.*, ontologies with very large numbers of classes and/or properties. SNOMED a major biomedical ontology is expressible within the OWL 2 EL profile.

- **OWL 2 RL** is suited for applications that require fast and scalable reasoning without sacrificing major expressive power. It is designed as a common layer between OWL and RDFS applications using rule-based reasoning engines.
- **OWL 2 QL** is suited for applications requiring sound and complete query answering for data of significant size with reasonable computation complexity. It provides necessary constructs for expressing conceptual models such as UML class diagrams and ER diagrams. OWL 2 QL is designed to query relational databases by rewriting the original query into a standard SQL query.

OWL provides a special construct, *owl:imports*, to bring information in different OWL ontologies into a single ontology. It offers modularity in a limited sense such that an ontology can be syntactically divided into several files.

2.5.5 Web Service Modelling Language (WSML)

Web Service Modelling Language (WSML) is a family of ontology languages for describing various artefacts related to a Semantic Web services framework called the Web Service Modelling Ontology (WSMO) [74]. WSML brings together the Logic Programming and Description Logic paradigms, and unifies them in a single syntactic framework [59]. It has a human-readable syntax and comes with mappings to RDF(S) and OWL. Figure 2.7 shows the mapping between some basic constructs of the WSML and OWL languages. Interested readers should refer to [75], where the authors describe the entire set of mappings between the languages.

Apart from basic construct correspondences (*e.g.*, concept-class, property-attribute, instance-individual), logical expressions in the WSML language are specified using the *axiom* keyword. WSML supports various datatypes like strings, datetime, integers, decimals, floats and others that correspond to XML Schema primitive datatypes. Furthermore, automated translation from other languages (*e.g.*, OWL, RDF) is possible.

WSML has variant sublanguages with different expressivity and reasoning capabilities. The formalisms underpinning different WSML variants provide different language semantics. The WSML variants and their relation with OWL are defined as follows:

- **WSML-Core**: Based on the intersection of Description Logic and *Datalog* formalisms [59]. It is the common fragment of all WSML variants and the least expressive one. The main language constructs available include: concept, concept hierarchies, attributes, relations, relation hierarchies, instances, and support for datatypes. The recent extension WSML-Core 2.0 is adjusted to align with OWL 2 RL [76].
- **WSML-DL**: Covers the Description Logics fragment which is also a main part of the OWL language. The recent extension WSML-DL 2.0 covers OWL 2 RL, OWL 2 EL and OWL 2 QL [77].
- **WSML-Flight**: An extension of WSML-Core towards a powerful rule language. It adds meta-modelling, constraints and nonmonotonic negation. WSML-Flight is semantically equivalent to *Datalog* with inequality and (locally) stratified negation.

Web Service Modelling Language (WSML)

1. (ontology id
   $header_1$
   ...
   $header_n$
   ontology $element_1$
   ...
   ontology $elementn_n$)

2. (annotations
   id_1 hasValue $value_1$
   ...
   id_n hasValue $value_n$
   endAnnotations)

3. importsOntology id

4. concept id

5. id1 subConceptOf id2

6. id[attribute_id hasValue value]

7. ?x memberOf concept id
   impliedBy
   ?x[attribute_id hasValue ?y])

Web Ontology Language (OWL)

1. Ontology: id
   $header_1$
   ...
   $header_n$

2. Annotations: (id_1 ($value_1$))
   ...
   ...
   ...
   Annotations: (id_n ($value_n$))

3. Import: id

4. **Class**: id

5. id1 SubClassOf: id2

6. Individual: id

7. ObjectProperty: id
   Domain: d1
   Range: r1

Figure 2.7: OWL-WSML Correspondences

- **WSML-Rule**: An extension of WSML-Flight towards more expressive Logic Programming features such as: function symbols, unsafe rules and unstratified negation. The recent extensions WSML-Rule 2.0 and WSML-Flight 2.0 [78] is improved for increased support of Rule Interchange Format (RIF) [58].
- **WSML-Full**: Brings together the expressivity of WSML-DL and WSML-Rule.

Figure 2.8 shows that WSML has two alternative layerings. WSML-Full can be achieved by following along the paths (i): WSML-core → WSML-DL → WSML-Full, and (ii): WSML-core → WSML-Flight → WSML-Rule → WSML-Full. WSML-core is the intersection of Description Logic and *Datalog*. WSML-Full is a very expressive, but undecidable language.

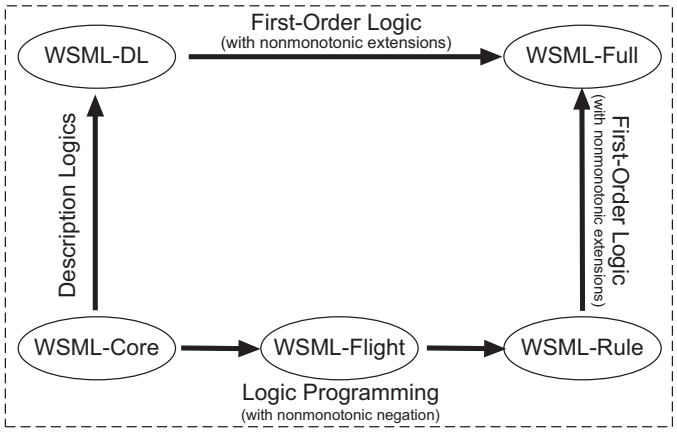

Figure 2.8: WSML Language Variants

The WSML ontology language describes WSMO artefacts: data, schema, goal, and services. In

this thesis HL7 ontologies are developed for two versions using WSML and OWL ontology languages. OWL and WSML have strong correspondences to description logics [79, 59]. Classes (WSML concepts), properties (WSML attributes), and individuals (WSML instances) in OWL correspond to concepts, roles, and individuals in the description logic (DL). OWL class and property hierarchies, property restrictions, cardinality constraints, etc. have semantics described in the DL. Next, we briefly describe DL and semantics behind some of the basic constructs in the OWL and WSML ontology languages.

2.6 Formalism

Description Logics (DL) [80] is a knowledge representation language that can be used to represent the knowledge of an application domain in a structured and formally well-understood way. DL describes the domain in terms of individuals that can be grouped into classes (or concepts) and relate them to each other by binary relationship, called roles (or properties/predicates). DL systems are comparable with traditional database systems, *i.e.*, having the schema and data defined separately and combined in a special way for applying restrictions on the data. The difference lies in the DL formalism and loosely coupled arrangement of schema and data. A DL knowledge base is typically comprised of two components (i) TBox: a TBox contains intensional knowledge in the form of terminologies that describe general properties and concepts; (ii) ABox: an ABox contains extensional (or assertional) knowledge which is specific to the individuals of the domain of discourse. Intensional knowledge is generally considered as fixed whereas extensional is subject to occasional or constant changes. DL generally describes primitive notions such as atomic concepts C, constant concepts (*e.g.*, THING, NOTHING), domain of interest or universe, atomic role R, and individual names n_1, n_2. The description of complex concepts and roles can be constructed with the recursive definition of atomic concepts C and roles R. A DL knowledge base is formally defined as:

Definition 2.6.1 ***(DL knowledge base)*** *A DL knowledge base $\mathcal{K} = (T, A)$ consists of a finite set of TBox T axioms, and a finite set of ABox A assertions.*

A TBox is a vocabulary of an application domain, while an ABox contains assertions about individuals in terms of the vocabulary. In order to provide a formal meaning to DL concepts, DL semantics is defined as:

Definition 2.6.2 ***(DL Interpretation)*** *An interpretation $\mathcal{I}$ of knowledge base $\mathcal{K}$ is a pair $(\Delta^{\mathcal{I}}, (.)^I)$, where $\Delta^{\mathcal{I}}$ is a non-empty set of objects called the domain of $\mathcal{I}$, and $(.)^I$ is an interpretation function which maps every concept C to a subset of the domain $C^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ and every role R to subset $R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$.*

For creating complex concepts semantic conditions are shown in Table 2.2 (third column). The first and second columns of Table 2.2 show commonly used OWL constructs, and corresponding DL syntaxes. One can use these DL (OWL) constructs for building complex concepts. For example, concept constructs such as conjunction ($X \sqcap Y$) and value restriction ($\forall$R.C) can be used to describe classes with multiple superclasses and can apply restrictions on individuals and roles. The first row

of Table 2.2 shows the OWL *SubClassOf* construct, corresponding DL syntax ($\sqsubseteq$) and a semantic condition. In the DL interpretation $\mathcal{I}$, the subsumption relation $C \sqsubseteq D$ holds between concepts C and D, if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ for all interpretations $\mathcal{I}$.

OWL Constructs	**DL Syntaxes**	**Semantics**
C_1 SubClassOf: C_2	$C_1 \sqsubseteq C_2$	$C_1^{\mathcal{I}} \subseteq C_2^{\mathcal{I}}$
C_1 EquivalentTo: C_2	$C_1 \equiv C_2$	$C_1^{\mathcal{I}} = C_2^{\mathcal{I}}$
DisjointClasses: C_1 C_2	$C_1 \sqsubseteq \neg C_2$	$C_1^{\mathcal{I}} \cap C_2^{\mathcal{I}} = \emptyset$
C_1 and C_2	$C_1 \sqcap C_2 .... \sqcap C_n$	$C_1^{\mathcal{I}} \cap C_2^{\mathcal{I}} .... \cap C_n^{\mathcal{I}}$
C_1 or C_2	$C_1 \sqcup C_2 .... \sqcup C_n$	$C_1^{\mathcal{I}} \cup C_2^{\mathcal{I}} .... \cup C_n^{\mathcal{I}}$
not C	$\neg$C	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
R some C	$\exists$R.C	$\{i \in \Delta^{\mathcal{I}} \mid \exists e \in \Delta^{\mathcal{I}} : (i,e) \in R^{\mathcal{I}} \wedge e \in C^{\mathcal{I}}\}$
R only C	$\forall$R.C	$\{i \in \Delta^{\mathcal{I}} \mid \forall e \in \Delta^{\mathcal{I}} : (i,e) \in R^{\mathcal{I}} \Rightarrow e \in C^{\mathcal{I}}\}$
R max n/R min n	$\geq n$R	$\{i \in \Delta^{\mathcal{I}} \mid \mid e \in \Delta^{\mathcal{I}} : (i,e) \in R^{\mathcal{I}} \mid \geq n\}$
	$\leq n$R	$\{i \in \Delta^{\mathcal{I}} \mid \mid e \in \Delta^{\mathcal{I}} : (i,e) \in R^{\mathcal{I}} \mid \leq n\}$

Table 2.2: OWL Constructs, DL Syntaxes and Semantic Conditions

In a TBox one can define the concepts (*ActLabObservation*, *ClinicalTrial*) and a role (*measurement*). An ABox can contain the assertion that, *ClinicalTrial*$\sqcap$*ActLabObservation*(observationEvent), means observationEvent is a *ClinicalTrial*. One can also assert that *measurement*(observationEvent, 182), means observationEvent *measurement* has a value 182. An ABox resembles the database instances. When processing both TBox and ABox together its possible to deduce whether some arbitrary individual is an instance of an arbitrary description. Contrary to the database closed-world semantics, the semantics of DL is an open-world and the language is interpreted against a global domain. DL extended formalisms have been proposed to split the global domain into interpretations of multiple local domains. We will discuss these formalisms in Chapter 8 about local concepts and constraints requiring contextualised or localised interpretation of a domain.

DL systems offer reasoning services for stored terminologies and assertions. Typical reasoning tasks for a terminology are to determine whether a description is satisfiable (*i.e.*, non-contradictory), or whether one description is more general than another one, that is, whether the first subsumes the second. Some of the DL reasoning tasks usually considered in an application environment are:

- **Class satisfiability checking:** the task of checking whether a class C is satisfiable, *i.e.*, testing if there is an individual in an interpretation which is an instance of a class C. For example, if a TBox (of the RDF data in Listing 2.4) T = {ClinicalTrial, Treatment, ActObservation$\sqsubseteq$ $\neg$Treatment, ActLabObservation $\equiv$ ActObservation $\sqcap$ ClinicalTrial}. Then, a new axiom ActLabObservation $\sqsubseteq$ Treatment is unsatisfiable with respect to the TBox T since the disjointness constraint between ActObservation and Treatment is violated.

- **Class subsumption checking:** the task of checking whether a concept C is subsumed by a concept D. For example, if a new axiom Surgery $\equiv$ ClinicalTrial $\sqcap$ Treatment is added to the above TBox T, then new inferences Surgery $\sqsubseteq$ ClinicalTrial and Surgery $\sqsubseteq$ Treatment can be deduced.

- **Consistency checking:** the task of checking if a new concept assertion in an ABox A makes A inconsistent with respect to a TBox T or not. For example, assertions in an ABox A={ActObservation(observationEvent), Treatment(observationEvent)} is inconsistent with respect to the TBox T since an individual (observationEvent) cannot belong to two disjoint classes (ActObservation, Treatment).

- **Instance checking:** the task of checking whether an individual x is an member/instance of a concept C. For example, deciding whether an individual observationEvent is an instance of a concept ActObservation.

As said, one main advantage of applying the Semantic Web Layer cake for healthcare domain is the use of ABox (RDF data in Listing 2.4) and TBox (schema or logical axioms) under a common framework. Also, If these reasoning services described above are tied appropriately with the domain requirements then an obvious benefit is (i) the automation of various services offered by a knowledge-based system; and (ii) assistance in building a knowledge-based system. For instance, reasoning can be applied during the ontology building and alignment processes (i) ontology building: checking concept satisfiability, ontology satisfiability and implied relationships; (ii) ontology alignment: computing integrated ontology concept hierarchy and/or consistency.

2.7 Ontology Building Methodology

Ontology building is a set of activities including ontology development steps, the ontology life cycle, supporting tools, and languages applied coherently for modelling a domain knowledge [81]. Several ontology building methodologies have been proposed in the last two decades. The 1990s have witnessed the growing interest of many practitioners in approaches that support the creation and management, as well as the population of single ontologies built from scratch [10]. Until the mid-1990s, the ontology development process was an art rather than an engineering activity. Each development team usually followed their own set of principles as well as their own design criteria and phases for manually building the ontology [10]. Thus, the absence of common and structured guidelines slowed the development of ontologies within and between teams, the extension of any ontology, the possibility of ontologies of being reused in others, and the use of ontologies in concrete applications [82]. Until now, a large number of ontologies have been developed by different groups, under different approaches, and with different methods and techniques. However, in comparison to the software engineering counterpart, ontological building is still in its infancy.

Furthermore, the advancement of technology and significant improvement in availability of structured information, ontology practitioners with the goal of speeding up the ontology development process, are starting to reuse [83] as much as possible (i) other ontologies such as UMLS [84], GALEN and ontology modules [85] (ii) ontology statements and ontology design patterns [86], and (iii) non-ontological resources [87] such as thesauri, databases, XML schemas, UML models and classification schemas (*e.g.*, LOINC, SNOMED) built by others and which already have some degree of consensus. Developers realised that such distributed ontology networks should not be developed entirely from scratch, but by reusing and possibly reengineering other ontologies, databases, XML schemas, thesauri, UML models, classification schemes, and other knowledge resources, as well as by taking into account good practices in the development process.

A series of methods and methodologies for developing ontologies from scratch have been reported in [88, 10] and these can be summarised as follows: In 1990, Lenat and Guha published the general steps [89] and some interesting points about the Cyc[12] development. Some years later, in 1995, on

[12]http://www.cyc.com/

the basis of the experience gathered in developing the Enterprise Ontology [90], the first ontology building guidelines were proposed in [27]. The methodology METHONTOLOGY [91] appeared at the same time and was extended in the later paper [28]. Some years later, the On-To-Knowledge methodology appeared as a result of the project with the same name [29]. However, all these methods and methodologies do not consider distributed and layered construction of ontologies. In this respect, in 2004, the DILIGENT methodology [30] was proposed. This new methodology was intended to support domain experts in a distributed setting when they need to engineer and evolve ontologies. We will discuss these ontology building methodologies in detail in Chapter 4.

2.8 Ontological Heterogeneity

The global networking paradigm brings with it the challenge of achieving global semantic interoperability among heterogeneous information systems. Most of the work carried out on ontologies for the Semantic Web is on which language or which method to use to build the global ontology in a top-down fashion or to build the global ontology on the basis of the existing local ones in a bottom-up fashion. In recent times, one of the basic problems in the development of techniques for the Semantic Web is the integration of ontologies. In an open or evolving systems, like the Semantic Web, different parties would, in general, adopt different ontologies. Thus, just using ontologies, as when using XML, does not reduce heterogeneity: it raises heterogeneity problems to a higher level.

In this section we discuss ontology based approaches to deal with semantic heterogeneity. First, we describe the standard approach of matching the conceptual differences between ontologies that share similar artefacts though modelled differently. Then, we describe recent research efforts in attaching provenance information to the ontologies or data sources. It is argued that the inclusion of such provenance information will mean a step ahead in the semantic integration and reconciliation of heterogeneous information systems. Finally, we briefly describe advanced issues such as constructing modular knowledge spaces and attaching provenance information to each module. The issues discussed in this section will be addressed later in following chapters.

2.8.1 Ontology Matching and Alignment

Ontology matching is a process of finding correspondences between semantically related entities within the ontologies [31]. The matching task could be manual or (semi)-automatic. Matching correspondences, called alignments, can be used for various tasks, such as ontology merging, query answering, data translation, or for navigation on the Semantic Web. Thus, matching ontologies enables the knowledge and data expressed in the matched ontologies to interoperate with each other. It is to be noted that, to enable bi-directional interoperation between two ontologies, it may require two mappings in each direction (*i.e.*, source, target). Current practice in ontology matching relates to a large number of research fields ranging from machine learning, concept lattices, database schema, and linguistics. Ontology alignment could also be seen as a common layer from which several ontologies could be accessed and hence could exchange information across ontology based information systems (OBISs). Chapter 6 will describe in detail about ontology matching and alignment approaches.

2.8.2 Context and Modularity in Ontologies

Information systems that are deployed across organisations are usually context-dependent. For example, constraints (*e.g.*, profile, policy, preference, temporal, spatial) or knowledge expressed by a particular organisation is usually applicable within the organisation and an appropriate knowledge interpretation is meaningful only when contextual knowledge (*e.g.*, place, event, time) is taken into account [92]. Therefore, real-life or application ontologies and corresponding data produced by individuals or groups in certain settings for specific purposes contain mostly contextual information for their appropriate interpretation. Lacking such contextual information often leads to undesirable inconsistencies when ontologies from various sources are merged to solve a domain problem [17]. Contextual information can be used to resolve such conflicts. It can be used to select relevant consistent parts of the knowledge base which suffice for the task at hand [33]. Contextual information can provide guidance for this selection process, as usually different possibilities exist for resolving an inconsistency. Representing knowledge for contextual ontologies–instead of single global ontologies–opens up new approaches to dealing with contradicting information.

Contextual and modular ontologies conceptually complement each other [33]. One important part of an ontology modularisation formalism concerns mechanisms to create, combine and, more generally, manipulate ontology modules. OWL provides limited support for modular ontologies. An ontology document identified via its ontology URI can be imported by another document using the *owl:imports* statement. The semantics of this import statement is that all definitions contained in the imported ontology document are included in the importing ontology, as if they were defined in the importing ontology document. It is worth mentioning that *owl:imports* is directed (only the importing ontology is affected) and transitive (if ontology A imports B and B imports C, then A also imports the definitions contained in C). Moreover, cyclic imports are allowed (*e.g.*, A imports B and B imports A).

One of the most commonly mentioned weaknesses of the importing mechanism in OWL is that it does not provide any support for partial-import [93]. Even if only a part of the imported ontology is relevant or agreed in the importing ontology, every definition is included. Moreover, there is no logical difference between imported definitions and proper definitions in the importing ontology: they share the same interpretation. Formalisms like Package-based Description Logics (P-DL) [94] and Distributed Description Logics (DDL) [95] have been proposed to support the modularity and context-awareness for ontological knowledge bases. For instance, P-DL improves the importing relation between ontologies by allowing partial modules to be imported and reasoned upon. Unlike importing mechanisms that include elements from some modules into the considered one, Distributed Description Logic (DDL) adopts a linking mechanism, relating the elements of "local ontologies" (called context) with elements of external ontologies. Each context is associated to its own local interpretation. Unfortunately, none of the contextual and/or modular formalisms are deployed within the current Semantic Web framework. We will discuss in Chapter 8 about approaches including formalisms that may support context and modularity for local or context-sensitive healthcare resources.

2.9 Summary

The chapter has presented a brief overview of the HL7 standard and two of its versions widely deployed in the healthcare industry. It describes the evolving Semantic Web layer cake which includes languages and technologies layered upon each other to realise the ultimate vision of the Web as a "global database" and provides an overview of ontology languages and the underlying formalism. The chapter briefly describes ontology building methodology that concerns systematic development of ontologies. Finally, the chapter describes some of the prominent causes of ontological heterogeneities and approaches proposed for resolving such heterogeneities.

Chapter 3

Integration Scenario

3.1 Introduction

Healthcare applications generally require the presence of a well integrated network of information systems that can be used for efficient clinical information exchanges. Common requirements on this integrated network include patient identification, access to demographic data and to information stored in electronic health records (EHRs), including previous medications, lab results etc. Information exchange becomes very important for collaborative care in e-Health scenarios. The scenario we present targets a very commonly encountered clinical case, clinical information exchange, in which medical documents originating from several sources are required to be exchanged and combined as a unified medical summary. Such patient medical summary is crucial in the normal treatment process or in emergency cases.

This scenario is relevant to clinical care providers who wish to access and exchange medical records electronically available throughout the EU for patients for whom they are providing care. The availability of well integrated network systems within the EU region provide a basic infrastructure to envisage unified medical summaries of patients living in the region.

The use case concentrates on cross-lab, hospital, and state information exchange, and the access to medical summaries.

3.2 Scenario: Lab Observation

The scenario describes clinical events in the course of primary medical care provided to a patient suffering from acute chronic diabetes. Clinical events are recorded as a part of patient medical history where two healthcare providers (hospitals) use different healthcare standards to model the electronic health records (EHRs). The scenario highlights the interoperability requirements between healthcare applications due to (i) the heterogeneity of data, information models, terminologies and vocabularies, and (ii) the inability to model and execute local (context-specific) policies on a more general framework such as the Semantic Web.

3.2.1 Patient Case History 1

Dr. Paul Smith is a General Practitioner (GP) in the Galway city. The GP's practice management system is HL7 Version 3 compliant. For lab observation cases, it places lab test order requests to another independent EHR system. Galway University Hospital (GUH) has a pathological laboratory, which is directed by Dr. John Colleen, and who is responsible for approving and releasing all test results. The information system in the GUH laboratory is HL7 Version 2 compliant and able to receive orders and return result reports electronically. GUH has a separate department for diabetic patients, where recruitment requires a physician referral. Usually, patients with advanced symptoms are admitted.

Dr. Paul Smith has a patient, Sean Murphy, who is examined on the 16th of July 2007 because of a poor wound healing, weight loss, and increase in urine production. Dr. Smith draws a blood sample, labels it with a bar code, and sends it to GUH laboratory. Sean Murphy has been identified based on his Irish PPS number: 678970W. The blood sample is identified as 7854698. Dr. Smith fills out the electronic order form in his office system for a Glycosylated hemoglobin (HbA1c) test and sends it to the GUH laboratory.

The GUH laboratory receives the electronic order and returns a message accepting the order with intent to fulfill it on a routine basis. The GUH laboratory performs the requested HbA1c test on the 18th of July 2007, after receiving the sample. The result of the HbA1c test is 9%. The LOINC code for the hemoglobin count is 4548-4 [1]. Dr. John reviews the results of the HbA1c test and notices that the blood sugar is abnormally high (normal reference range 4%-5.9%).

Dr. Colleen authorises the results of the HbA1c lab tests with an indication that Sean's blood sugar level is abnormally high. The order is complete so a notification is sent to Dr. Smith. It is important to mention that, even though the GUH laboratory system is HL7 Version 2 compliant and can electronically exchange information, it requires manual effort and paper work to integrate the patient observations with the GUH specialised diabetic center and the physician's practice management system.

Sean has been diagnosed as an acute chronic diabetic (Type 2), which requires him to frequently test his blood sugar level. GUH has special drug policy for diabetics (Type 2) where patients are treated with either Insulin or Avandia (an insulin equivalent drug), but not both. Sean receives the Insulin therapy and any future abnormal blood sugar level will require immediate and appropriate insulin treatment for him. The nature of his job demands frequent travel within Ireland and his travel constraints bring complexities whenever he is in an emergency situation. Every emergency situation requires him to visit a nearby clinic or hospital diabetic center, and each clinical visit requires repetitive lab tests that may have been performed earlier. Sean original medical records are with GUH laboratory information system and immediate availability of Sean's medical records is crucial in emergency situation.

[1] http://loinc.org/downloads/files/loinc-table-microsoft-access-format

3.2.2 Patient Case History 2

Dr. Gustav Roth is an internal medicine physician in the City hospital, Göttingen, Germany (GOET). The City hospital laboratory is directed by Dr. David Hahn and is the only pathological laboratory available in Göttingen. In addition to HL7 Version 3, GOET laboratory information systems use general vocabularies and terminologies for information modelling such as *GALEN*, *FOAF* [96], and is able to send and receive lab test orders electronically.

On Christmas holidays, Sean went to Göttingen, Germany for a week to meet his friends. On his way back to home he had a major car accident and doctors needed to operate on him as early as possible. Sean is identified based on his international driving licence number 345678IE. He had a year of medical history as an acute chronic diabetic patient and his current medical records are in GUH laboratory and other Irish hospitals where Sean visited frequently. Coincidentally, Galway University Hospital (GUH) and City hospital, Göttingen, collaborate as a part of a common EU healthcare framework and they can share information. In emergency situations like this, where time is critical, the automated integration of patient data can improve the overall quality of primary healthcare services. In this scenario, GUH and GOET lab systems are able to send and receive patient's records electronically but require manual intervention, paper work, and phone calls to make sensible understanding of medical records and Sean was running out of time. City hospital, Göttingen has decided to perform all tests locally without depending on his medical history. Sean was able to speak and provided an informal description of his medical history. Dr. Roth has ordered a diabetic test similar to the one performed on 18th of July 2007 to examine his current blood sugar level before going into any surgical treatment. GOET drug policy on Type 2 diabetics does not suggest any restrictions for Avandia and Insulin treatments, and thus Dr. Roth decides to prescribe Avandia to complement of Sean's insulin treatment.

3.3 Scenario Actors

The scenario has three actors: (i) General Practitioner (GP), (ii) GUH laboratory, and (iii) GOET laboratory. In HL7 applications each actor performs a specific role, *e.g.*, *Order Placer*, *Order Fulfiller*, and *Result Receiver*. Figure 3.1 shows messages required to be exchanged between all the three actors.

- **General Practitioner (GP):** This EHR is HL7 Version 3 compliant and it places a lab test request to another independent EHR system (hospital laboratory). The GP practice management plays the role of *Order Placer* and *Result Receiver*.
- **GUH Laboratory:** This EHR is HL7 Version 2 compliant. The Hospital Laboratory receives the order for the patient's lab test results. The GUH laboratory plays the role of *Order Fulfiller*.
- **GOET Laboratory:** This EHR is HL7 Version 3 compliant and receives lab test results from GUH Laboratory. The GOET laboratory plays the role of *Result Receiver* and *Order Fulfiller*. Clinical laboratories can play two roles in receiving lab results from other independent laboratories and providing lab results performed in their environments.

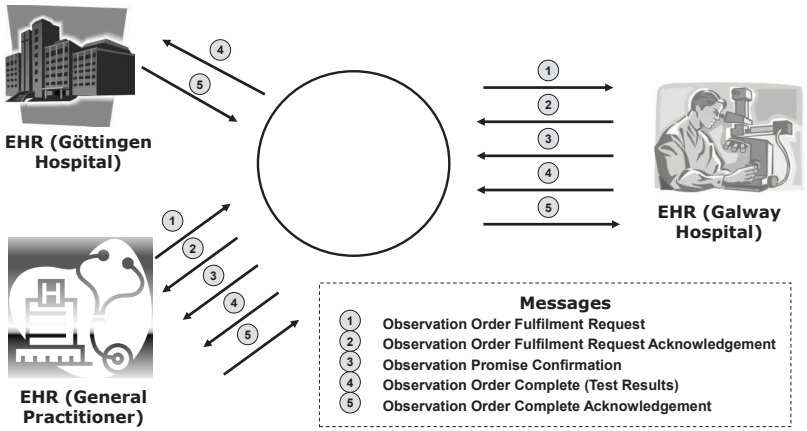

Figure 3.1: Lab Observation Scenario

The use case has five interaction messages with their acknowledgement messages such as:

- **Observation Order Fulfillment Request:** This message is used when a fulfillment request is communicated. If the receiver is the intended *Order Fulfiller* application, it is assumed that the *Order Fulfiller* application will perform the order and provide a completed result that fulfills the order.
- **Observation Order Fulfillment Request Acknowledgement:** An acknowledgement message stating that "Observation Order Fulfillment Request" has been received by the *Order Fulfiller* application.
- **Observation Promise Confirmation:** This message is sent by the receiver (*Order Fulfiller*) to confirm the fulfillment of the original Request (*i.e.*, Observation Order Fulfillment Request).
- **Observation Order Complete (Test Results):** This message states the completion of the result reporting. No other reports will be sent, unless there is a corrected report (*i.e.*, revision in the original Fulfillment Request). The observation order is complete and the result report is final.
- **Observation Order Complete Acknowledgement:** An acknowledgement message from the *Result Receiver(s)* stating that the observation order has been fulfilled.

The *Order Fulfiller* application communicates with a message that a particular clinical event is complete. The completed event may or may not fulfill the related filler order. When the final result event is completed by fulfilling the *Order Placer* order, the lab result message can communicate that the specific event is complete, and the *Order Fulfiller* promise is complete. The *Order Placer* determines if this fulfills the placer order (based upon its own clinical rules). It is one of the responsibilities of the *Order Placer* application to manage its own orders. Only the *Order Placer* application can determine whether *Order Fulfiller* lab results are completed or not. These five interaction messages are part of the PPEPR framework demonstration further described in Chapter 7 (Section 7.7).

Sean's medical records captured by each of the actors (GP, GUH Laboratory, GOET Laboratory) include information that the author (*e.g.*, physician, lab assistants) finds necessary in a specific clinical environment. Parts of (or the whole) record may subsequently be exchanged with others in various patterns that can never all be predefined. For example, GOET performs an independent blood test observation (during the emergency situation) and receives the past lab test result from the GUH laboratory (messages 4 and 5 of Figure 3.1). The widest range of possible scenarios and information flows must therefore be well described.

3.4 Scenario Interoperability Requirements

Table 3.1 shows the summary of the lab order data captured at the two independent laboratories.

Items	**Lab Value (GUH)**	**Lab Value (GOET)**
Lab order requisition number	6560-89	7779-78
Specimen identifier	7854698	89756
The ordering physician	Dr. Paul Smith	Dr. Gustav Roth
Physician health identifier	68374532	6837-4532
Patient	Sean Murphy	Sean Murphy
Patient health identifier	678970W:IrishPPSId	345678IE:Drivinglicence
Patient birth date	15/07/1974	15-07-1974
Date of Lab test Order	16/07/2007	22-12-2008
Lab Test ordered	Glycosylated hemoglobin	hemoglobin A1c
LOINC/SNOMED Lab Test Code	4548-4 (LN)	43396009 (SN)
Priority of Lab Test	Routine	Urgent

Table 3.1: Sean's lab test dated 16/07/2007 (GUH) and 22/12/2008 (GOET)

Healthcare application scenarios that require inter-department or inter-hospital communications increase the integration complexity. The work done by the Integrating the Healthcare Enterprise (IHE) initiative [97, 98] shows that intra-hospital application scenarios are technically very complex and pose interoperability challenges. Thus, a meaningful exchange of Sean's medical records across healthcare applications (GP, GUH, and GOET laboratories) demands interoperability at various levels:

- **Interoperability of standard information models and medical vocabularies:** We discussed that a Reference Information Model (RIM) represents the global artefacts of health record components. Similarly medical vocabularies like SNOMED or LOINC are defined at a global level as a common agreed set of medical terms used between healthcare institutions. Additionally, local terms or resources (*i.e.*, context-specific terms) need to interoperate with global ones and terms originating from other local sites. In distributed settings, global and local artefacts originating from several sources need interoperation among them, such that messages underlying these artefacts could be transformed, and thus, exchanged meaningfully. For example, items or attributes (first column of Table 3.1) such as "patient" or "LOINC/SNOMED Lab Test Code" are defined at the RIM and SNOMED level (*i.e.*, global level), and "Lab order requisition number" or "Specimen identifier" are local to the publishing hospitals. Ideally both the global and local artefacts are available in the UML, XML, or relational database formats.

  Key Requirements: Considering the arrangement of global and local resources in the HL7

and other medical vocabularies, an ontology building method will require three main features: (i) **reusability** of existing non-ontological knowledge sources such as UML specifications and XML schemas; (ii) **layering** of ontological knowledge bases in global and local spaces. As said, HL7 has a globally consistent conceptual model as well as a provision for locally designed messaging schemas that enable the exchange of clinical messages; (iii) **adaptation** of local knowledge sources with an upper or global conceptual model.

- **Interoperability of semantically similar or matching resources:** Healthcare systems need the ability to translate and manipulate a number of different controlled vocabularies. We discussed that domain vocabularies, such as HL7 information model-specific vocabularies, LOINC and SNOMED are used as reference vocabularies to provide meaning to the messaging elements of a patient record. For example, lab concepts 4548-4 and 43396009 (second last row of Table 3.1) are the same concepts which describes "Glycosylated hemoglobin" test. It is important to identify such matching concepts and automate the task of aligning medical vocabularies.
  Key Requirements: Aligning HL7 ontologies and medical vocabularies will require two main features: (i) Structural Similarity: As said, It is common that a same concept is modeled differently in healthcare standards and vocabularies. Which means, it is important to resolve the structural differences of two concepts that describes the same domain entity. For example, the "Patient" item in Table 3.1 are often modeled with different set of properties and descriptions; (ii) Annotation-based Similarity: It also common in the healthcare domain that name of a concept or property pose a coded value and description is provided as an annotation. Therefore, it is important to use such annotations while matching two similar concepts.

- **Interoperability of message instances:** Patient data refer to information model artefacts and domain vocabularies. An efficient instance level transformation should preserve the source and target schemas constraints. This is an important requirement for a successful clinical information exchange. For example, message instances (in any language format) representing the second and third columns of Table 3.1 need to be transformed such that LOINC, SNOMED, or HL7 based applications can process them appropriately.
  Key Requirements: An integration framework is required that can use the message schemas, alignments, and constraints for transforming underlying heterogenous messages.

- **Interoperability of local policies:** Healthcare systems need to process local information, legacy, or constraints (*e.g.*, drug policy, access policy) originating from specific clinical environments. Hospitals use different local concepts and constraints, and context specific treatment will be increasingly important as the scale of these systems increases. For example, we described that GUH has a drug policy that forbids the use of Insulin and Avandia drugs on the same patient. However, GOET is open to both possibilities.
  Key Requirements: The integration framework proposed above needs mechanism to resolve inconsistencies or conflicts caused due to context-specific resources.

We address the first three interoperability requirements within Chapter 4, Chapter 5, Chapter 6, and Chapter 7 which describes the PPEPR framework and how PPEPR framework enables

the exchange of clinical messages between GP, GUH and GOET laboratories. Finally, the fourth interoperability requirement (context-specific local policies) is discussed in Chapter 8.

3.5 Scenario and EU-Wide EHR Harmonisation

Two important topics for European harmonisation of Electronic Health Records (EHRs) in [99] have been defined as: (i) efficient clinical information exchanges, and (ii) quality management of clinical records. In the scope of the RIDE project, we have analysed the interoperability requirements of 36 use-case scenarios [100, 99] that span across: (i) management and exchange of clinical information (*e.g.*, Longitudinal Electronic Health Record, Laboratory Results, Emergency Dataset), (ii) telemedicine applications (*e.g.*, Tele-Diagnosis, Tele-Surgery, Tele-Access to Medical Knowledge), (iii) patient-centric applications (*e.g.*, Personal Consumer Health Record, Health Service Yellow Pages, Patient Support Groups), (iv) public health applications (*e.g.*, Epidemiological Registries, Public Health Surveillance), and (v) secondary healthcare applications (*e.g.*, Reimbursement and Health Insurance Services, Management of Clinical Trials, Cross-enterprise Workflow). These 36 use-cases are collected from healthcare institutes, practitioners, research organisations, and related stakeholders in the EU. The scenario presented in this chapter (*i.e.*, lab observation) covers common interoperability characteristics found in all 36 use-cases. The four interoperability requirements highlighted are key challenges for interoperability enabler technologies such as the Semantic Web.

Chapter 4

Ontology Building Methodology

4.1 Introduction

This chapter presents an ontology building methodology tailored for the Health Level Seven Standard (HL7). We discussed in Section 2.2 that HL7 has two major versions, namely, Version 2 and Version 3. Version 2 specifications are available as XML Schemas, while Version 3 specifications are combinations of XML Schemas and UML conceptual models. The methodology proposed in this thesis is scenario-based where ontology engineers first identify an application scenario. All resources and entities identified within the application scenario set the guidelines for further development phases. This scenario-based approach makes the proposed methodology application dependent. Our main goal is to provide interoperability between HL7 applications by ontologising resources and aligning ontological concepts of the standard as well as locally deployed applications. HL7 has two types of conceptual models, (i) a static model: this model describes all artefacts required for constructing clinical messages and (ii) a dynamic model: exchanging a message or creating a document is caused (triggered) by an event which leads to an interaction with that message between two or more applications. The dynamic model consists of trigger events that describes how two applications should communicate with each other. This thesis focuses on the static model and covers only the *Laboratory* and *Observation* domains. HL7 ontologies built within scope of this thesis are deployed for the Plug and Play Electronic Patient Records (PPEPR) framework [39] and act as proof of concepts for the PPEPR methodology discussed in this chapter.

While developing the PPEPR methodology, we have considered two general methodological objectives: first, we wanted to provide fine-grained guidance to ontology engineers, healthcare engineers and domain experts so the sequence of development steps are concrete and re-producible. Second, we provide validation of the PPEPR methodology by a case study to show how engineers and domain experts could adapt the proposed development steps in their environments. It is very difficult to provide an abstraction which represents many development processes, onto an instantiated process in detail. Nevertheless, without a reasonable substantiation of the proposed development steps in concrete case studies, a proposal like the PPEPR methodology would be meaningless. At the finest detail for methodological support, we have proposed the PPEPR integration framework, HL7 ontologies, and a concrete case study to investigate the impact of the proposed methodology.

In the previous chapter, we have identified three HL7-specific key requirements (or features) for ontology building methodologies: (i) **reusability** of existing non-ontological knowledge sources such as UML specifications and XML schemas; (ii) **layering** of ontological knowledge bases; and (iii) **adaptation** of local knowledge sources with an upper or global conceptual model. Sections 4.2, 4.3, and Section 4.5.1 describe these required features and compare them with existing methodological support across various dimensions. The features highlighted in this chapter are further discussed and elaborated in the following three chapters:

- Lifting HL7 Resources (Chapter 5): The creation of ontologies from non-ontological HL7 resources, *i.e.*, UML Models and XML Schemas.

- Aligning HL7 Ontologies (Chapter 6): An alignment mechanism improving automation and accuracy in matching HL7 ontologies. The proposed alignment mechanism eventually facilitates interoperability between HL7 Version 2 and Version 3 applications. Moreover, the ontology alignment mechanism discussed in Chapter 6 could also be applicable for aligning other healthcare and bio-medical ontologies.

- PPEPR Framework (Chapter 7): The PPEPR methodology is realised within the PPEPR framework. Chapter 7 describes the architecture and data mediation component of the PPEPR framework. Based on the integration scenario of Chapter 3, Chapter 7 describes the functioning of the PPEPR framework.

The next section summarises and compares well-known methodologies for building ontologies, paying special attention to ontology development activities, reuse of non-ontological resources, and mechanism to integrate local ontologies. It is important to mention that existing methodologies invest significant effort in ontology requirements specification during the initial stages of ontology building. However, in the context of ontologising the healthcare standards, greater consensus is already available in the form of HL7 specifications. Therefore, initial consensus building stage is trivial when dealing with ontologising standard conceptual models and specifications.

Our state of the art investigation is divided into two parts, first we examine support of three identified features within existing ontology building methodologies. Second, we present some recent work which is specific to building HL7 ontologies. We identify contributions made by each methodology and their limitations with respect to ontologising the HL7 standard.

4.2 State of The Art-1: Ontology Building Methodologies

This section includes a chronological brief description of the four most widely used methodologies for building ontologies (Enterprise Ontology, METHONTOLOGY, On-To-Knowledge and DILIGENT). Additionally, at the end of this chapter (Section 4.5), we compare these four methodologies with respect to the already mentioned features required to ontologise the HL7 standard : (i) reuse of non-ontological knowledge sources, (ii) layering of ontologies, and (iii) adaptation of local ontologies.

4.2.1 Enterprise Ontology

The Enterprise Ontology methodology is based on the experiences of modelling enterprise processes [27]. The methodology suggests the following activities to create an ontology:

- **Identify purpose:** it is important to understand the purpose behind building an ontology and its intended uses.
- **Building the ontology**, which is broken down into five steps:
  - **Ontology capture:** (i) identification of the key concepts and relationships in the domain of interest. It is important to focus on the concepts rather than the words representing them, (ii) production of precise unambiguous text definitions for the concepts and relationships, and (iii) identification of terms to refer to such concepts and relationships.
  - **Coding** involves explicitly representing the knowledge acquired in ontology capture in a formal language.
  - **Integration** with other existing ontologies.
  - **Evaluation** with respect to the frame of reference (*e.g.*, associated software environment, domain)
  - **Documentation** recommends that guidelines be established for documenting ontologies, possibly differing according to the type and purpose of the ontology.
- [27] point out three strategies for identifying concepts for the ontology (for the ontology capture task).
  - The **Top-down** strategy proposes that the most abstract concepts be identified first and then specialise them into more specific concepts. This enables knowledge engineers to control the level of detail but creates a risk of creating arbitrary, high-level categories not commonly found in the concrete domain.
  - The **Bottom-up** strategy proposes that the most specific concepts be identified first and then generalise them into more abstract concepts. It results in a very high level of detail which might lead to increased development effort, makes it difficult to spot commonalities between related concepts, and increases the risk of inconsistencies.
  - The **Middle-out** strategy strikes a balance between the previous two strategies. It proposes the core of basic terms be identified first, and then specify and generalise them as required. As a result, detail is added only as necessary and the higher level categories arise more naturally. This leads to less re-work and less overall effort. [27] suggested to apply this strategy for complex domains where knowledge resources have different granularity.

The main Enterprise Ontology contributions are (i) identifying a set of activities for ontology building; (ii) the identification of a middle-out strategy that balances the conceptualisation strategy; and (ii) identifying three strategies for ontology building.

4.2.2 METHONTOLOGY

The METHONTOLOGY methodology [28] includes the identification of the ontology development process, a life cycle based on evolutionary prototyping, and techniques to carry out each activity during the management, development, and support activities. To give technological support to METHONTOLOGY, WebODE [101] was built. However, some other ontology tools and tool suites can also be used to build ontologies following this methodology, *e.g.*, WSMT [102], Protégé [103], etc. METHONTOLOGY is based on the IEEE standard for software development [104] and ontology development activities are organised into three categories: management, development, and support.

Ontology management activities include activities that initiate, monitor, and control an ontology project throughout its life cycle. These activities are:

- The **scheduling** activity, which identifies the tasks to be performed, their arrangement, and the time and resources needed for their completion.
- The **control** activity, which guarantees that scheduled tasks are completed in the manner intended to be performed.
- The **quality** assurance activity, which guarantees that the quality of each and every product output (ontology, software, and documentation) is satisfactory.

Ontology development activities are grouped into pre-development, development and post-development activities:

Pre-development activities include activities that explore and allocate requirements before the ontology development can begin. These activities are

- An **environmental study** is carried out to understand the project environment, the organisation where the ontology will be developed, the participants in the ontology development, etc.
- A **feasibility study** answers questions such as: Is it possible to build the ontology? Is it suitable to build the ontology? etc.

Development activities include activities performed during the development and enhancement of an ontology project. Such activities are

- The **specification activity** states why the ontology is being built, what its intended uses are, and who are the end-users.
- The **conceptualisation activity** structures the domain knowledge as meaningful models at the knowledge level.
- The **formalisation and implementation activities** transform the conceptual model into a formal model using an ontology language.

Post-development activities include the maintenance activity, which updates and corrects the ontology if needed, and the (re)use activity, which refers to the (re)use of the ontology by other ontologies or applications.

Ontology support activities include activities that are necessary to ensure the successful completion of ontology building. This includes a series of activities performed at the same time as the development-oriented activities. Such activities are:

- The **knowledge acquisition** activity whose goal is to acquire knowledge from experts of a given domain or through some kind of (semi)automatic process, which is called ontology learning.
- The **evaluation activity** makes a technical judgment of the ontologies, of their associated software environments, and of the documentation. This judgment is made with respect to a frame of reference (*e.g.*, application, domain) during each stage and between stages of the ontology life cycle.
- The **integration activity** is required when building a new ontology by reusing other available ontologies.
- The **merging activity** consists of obtaining a new ontology from several ontologies on the same domain. The resulting ontology is able to unify concepts, terminology, definitions, constraints, etc., from all the source ontologies. The merge of two or more ontologies can be carried out either at run-time or design time.
- The **alignment activity** establishes different kinds of mappings between the ontologies involved. Hence, this option preserves the original ontologies and does not merge them.
- The **documentation activity** details, clearly and exhaustively, each and every one of the completed stages and products generated.
- The **configuration management** activity records all the versions of the documentation and the ontology code to control the changes.

The main METHONTOLOGY contributions are: (i) identification of the ontology development process, (ii) identification of the life cycle, and (iii) provision of detailed guidelines for building ontologies from scratch.

4.2.3 On-To-Knowledge

The On-To-Knowledge project [29] mainly focused on applying ontologies to electronically available information to improve the quality of knowledge management in large and distributed organisations. On-To-Knowledge developed methodology and tools for intelligent access to large volumes of semi-structured and textual information sources in internet-based environments. The ontologies developed with this methodology are application dependent. The processes proposed by this methodology can be summarised as follows:

- **Feasibility study:** According to On-To-Knowledge, the feasibility study is applied to the complete application and, therefore, should be carried out before developing the ontologies. The feasibility study serves as a basis for the kickoff process.

- **Kickoff:** The result of this process is the ontology requirements specification document that describes the following issues: (i) the domain and goal of the ontology, (ii) design guidelines (for instance, naming conventions), (iii) available knowledge sources (structured or semi-structured), (iv) potential users and use cases, (v) applications supported by the ontology. In the kickoff process developers should look for potentially reusable ontologies already developed. Although this methodology mentions the identification of potential ontologies to be reused, it does not provide detailed guidelines for identifying such ontologies or for reusing them. Apart from that, this methodology does not explicitly provide guides for the reuse and reengineering of non-ontological resources.

- **Refinement:** The goal here is to produce a mature application-oriented "target ontology" according to the specification given in the kickoff process. This refinement process is divided into two tasks:

  - Task 1: Knowledge elicitation process with domain experts. The baseline ontology (the first draft of the ontology obtained in Kickoff task) is refined through interaction with domain experts. As this activity is performed, axioms are identified and modelled. During the elicitation, the concepts are gathered on one side and the terms to label the concepts on the other. Then, two elements are mapped.
  - Task 2: Formalisation of concepts and relations identified in Task 1. The ontology is implemented using an ontology language. The language is selected according to the specific requirements of the application envisaged. To carry out the formalisation, On-To-Knowledge recommends the use of the OntoEdit ontology editor, which automatically generates the ontology code in several languages, but other ontology editors that perform similar functions can also be used.

- **Evaluation:** The evaluation process serves as a proof of the usefulness of the ontologies developed and their associated software environment. The product obtained is called an ontology based application. During this process two tasks are carried out:

  - Task 1: Checking the requirements and competency questions. The developers check whether the ontology satisfies the requirements and "can answer" the competency questions.
  - Task 2: Testing the ontology in the target application environment. Further refinement of the ontology may be needed in this activity. This evaluation process is closely linked to the refinement process. In fact, several cycles are needed until the target ontology reaches the level envisaged.

The main contributions of On-To-Knowledge are (i) using application requirements in the ontology building process; (ii) a top-down approach of applying ontologies for an intelligent information processing.

4.2.4 DILIGENT

The DILIGENT methodology [30] is intended to support domain experts in a distributed setting in order to engineer and evolve ontologies. This methodology is focused on collaborative ontology engineering, and its central issue is to keep track of the change arguments between ontologies developed with different scopes of use (*e.g.*, global and local ontologies). DILIGENT methodology proposes following five main phases for ontology development:

- **Build:** The build phase aims to create an initial version of the ontology quickly, so that the stakeholder can start using the ontology soon. Domain experts, users, knowledge engineers and ontology engineers collaboratively create an initial version of the ontology. Completeness of the initial shared ontology with respect to the domain is not required. In building this initial version of the ontology, DILIGENT does not propose carrying out the ontology requirements specification activity nor does it propose reusing and reengineering available knowledge resources.

- **Local adaptation:** Users locally adapt the ontology according to their own needs to organise knowledge in different layers. Once the shared ontology is made available, users can start using it and locally adapting it for their own purposes, typically due to new organisational requirements or user changes to fit the local requirements. In the local environment, users are free to change the local copy of the shared ontology. However, they are not allowed to directly change the ontology shared by all users. All local changes of the shared ontology are collected by a central control board.

- **Analysis:** In the analysis phase the ontology control board evaluates the changes suggested by the stakeholders. The input from users provides the necessary arguments to underline change requests, whereas the control board analyses the local ontologies and the change requests and tries to identify similarities in users' ontologies. A crucial activity of the board is to decide which changes are going to be introduced in the next version of the shared ontology. Then, a balanced decision is made that takes into account the different needs of the users and meets the user's evolving requirements.

- **Revision:** The board revises the ontology by deciding which changes should be applied to the ontology. The board should regularly revise the shared ontology in order to avoid a larger divergence of the local ontologies from the shared ontology. Therefore, the board should have a well-balanced and representative role of the different types of stakeholders involved in the process, the different needs of the users, and their evolving requirements.

- **Local update:** In the last step the stakeholders update their local ontologies based upon the revised version of the ontology. Once a new version of the shared ontology is released, users can update their own local ontologies to better use the knowledge represented in the new version. Even if the differences are small, users may rather reuse the new concepts, for example, instead of using their locally defined concepts. The shared ontology may contain several of the changes that were introduced in the local adaptation phase; other changes, however, may be missing. Because the control board tries to balance the different users needs, it will not always accept

changes as they are. Thus, even if a previous change made it into the new revision of the shared ontology, it may contain differences.

The main contribution of DILIGENT is the recognition of local ontologies within the ontology development framework. However, local ontologies are reengineered versions of global ontologies. In that sense, DILIGENT does not consider different sources that may provide input separately to global and local ontologies. The central board manages the global and local ontologies, this way, layering is implicit in the DILIGENT methodology.

4.3 State of The Art-2: Ontologising HL7 standard

The first release of the HL7 Version 3 RIM [4] was published in 2003. Since then, ontology engineers have taken initiatives to ontologise Version 3. As far as our knowledge is concerned, the EU Project Artemis [105] was found to be the only significant contribution for ontologising HL7 Version 2. As discussed in Chapter 2, ontologising Version 2 and Version 3 can be categorised into two broad domain requirements (i) semantic mappings with standard bio-medical vocabularies available in ontological format, and (ii) interoperation between Version 2 and Version 3 applications as well as with other prominent EHR standards (*e.g.*, openEHR, CEN/ISO EN13606). For the latter, other EHR standards need to be ontologised. EHR standards such as openEHR share semantic similarities with HL7 artefacts [106] and tool support[1] is available that can transform Archetype Definition Language (ADL) [107] of openEHR to OWL ontologies. We envision that the network of standard ontologies and their alignments would enable the exchange of clinical messages across departments, hospitals, regions or countries. We discuss here four important related works in ontologising HL7 Version 2 and Version 3.

HL7 Version 3 (Bhavana Orgun): Orgun [108] developed a Version 3 ontology using RDFS. The main contribution of this work was to identify the important set of ontological concepts and properties from Version 3 artefacts. The author has developed a small part of a Version 3 conceptual model into an RDFS ontology. A major limitation of this work is the lack of a systematic ontology development methodology. The author provided for a manual transformation of UML constructs to ontologies. Local schemas and layering of ontological knowledge bases are outside the scope of this work.

HL7 Version 2 and Version 3 (Artemis EU Project): One of the goals of the Artemis project was to provide interoperation between the two versions of the HL7 standard. In the context of ontology building, the main contribution of Artemis is to ontologise and align the versions. Artemis has applied a bottom-up approach where ontologies generated from local schemas are generalised to form the global conceptual model [5]. However, the Version 3 specification has a strict separation between global and local models. Therefore, a major limitation of this work is the lack of separation (or layering) between global and local ontologies. Similar to Orgun's work, Artemis lacks a systematic ontology building methodology with detailed steps that can guide the overall ontology development tasks.

[1] http://www.openehr.org/shared-resources/usage/academic.html

HL7 Version 3 (Helen Chan): This work has been published at W3C HCLS IG[2]. Chan's work has improved Orgun's ontology by covering a broader range of artefacts from Version 3. The ontology is implemented in OWL. The approach taken for ontology development is primarily a top-down approach (similar to the top-down strategy proposed in Enterprise Ontology), where Version 3 artefacts are identified first, and each artefact is mapped to a respective OWL class or property. Major contribution of this work is the ontology design which better corresponds with HL7 artefacts (datatypes, concepts, and relations). However, an important limitation is the ignorance of local models and lack of a systematic methodology.

HL7 Version 3 (Alan Rector): The objective of this work [48] is to align the Version 3 artefacts with other standard vocabularies such as SNOMED. Therefore, the focus of this work is different from the issue of interoperability between two versions of HL7 standard and the presence of local applications. However, the author has recognised the importance of domain specific local models and their relationships with a global conceptual model. The work has a major contribution of providing a formal analysis for the problem of aligning Version 3 ontology and standard medical ontology like SNOMED.

We notice that all the approaches mentioned above are about ontologising upper conceptual models of the HL7 standard. None of them consider local applications and the related issues. Above all, works mentioned above lacks detailed steps for ontologising the HL7 artefacts. The PPEPR methodology has two development steps described in the following sections, that deal with the problem of layered knowledge spaces and how local resources could be adapted with an upper conceptual model.

4.4 PPEPR Methodology

The PPEPR methodology is grounded on existing methodologies and domain experiences, as presented in Figure 4.1.

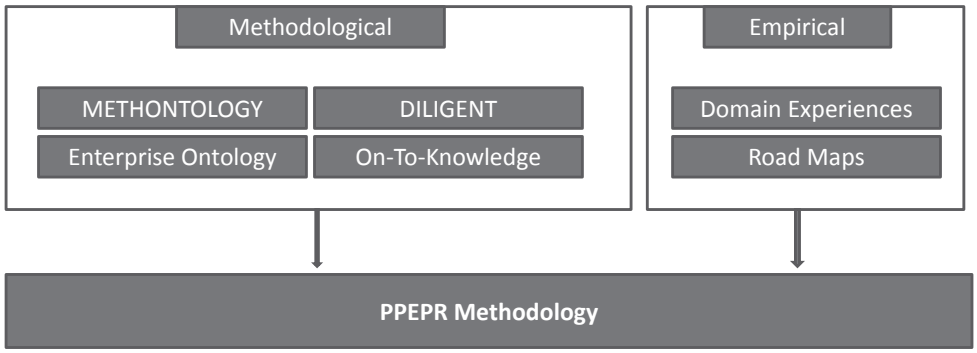

Figure 4.1: Inputs to PPEPR methodology

- **Existing Integration Systems:** Our experiences on existing software (See Section 2.4) aimed at integrating HL7 applications are taken as input to the development of the PPEPR

[2]http://www.w3.org/wiki/HCLS/ClinicalObservationsInteroperability/HL7CDA2OWL.html

methodology. The limitations of existing healthcare integration systems is the main motivation of applying ontologies on top of the HL7 messaging framework.

- **Methodologies and Methods for Building Ontologies:** We have inherited where ever possible from the existing methodologies (Enterprise Ontology, METHONTOLOGY, On-To-Knowledge, and DILIGENT). Additionally, we introduce HL7-specific guidelines for carrying out development phases and steps.

- **Domain Experiences and Roadmaps:** the RIDE[3] consortium has suggested a roadmap for the interoperability of different healthcare standards and systems. The use of semantic technologies is at the core of the suggested roadmap. From our experiences in RIDE, we obtained a preliminary set of requirements and guidelines about the use of semantics and making healthcare applications interoperable. The RIDE guidelines are refined and concretely adapted within the PPEPR methodology.

Figure 4.2 presents the PPEPR methodology which consists of five phases: (i) the *scoping* phase establishes the purpose of ontology building and identifies resources that can support the ontology building process; (ii) the *technology support* phase evaluates Semantic Web languages and supporting tools that can fulfill the requirements of the scoping phase; (iii) the *modelling* phase provides detailed guidelines for constructing ontologies; (iv) the *alignment* phase resolves ontological heterogeneity; finally (v) the *testing* phase ensures consistency and correctness of ontologies with respect to the previous phases and requirements. Particular development steps are allocated to each phase, which indicate the order in which the activities should be performed. The *modelling*, the *alignment*, and the *testing* phases are iterative until the required ontologies and their alignments have been constructed. In the following sections we will describe phase-by-phase details of the PPEPR methodology.

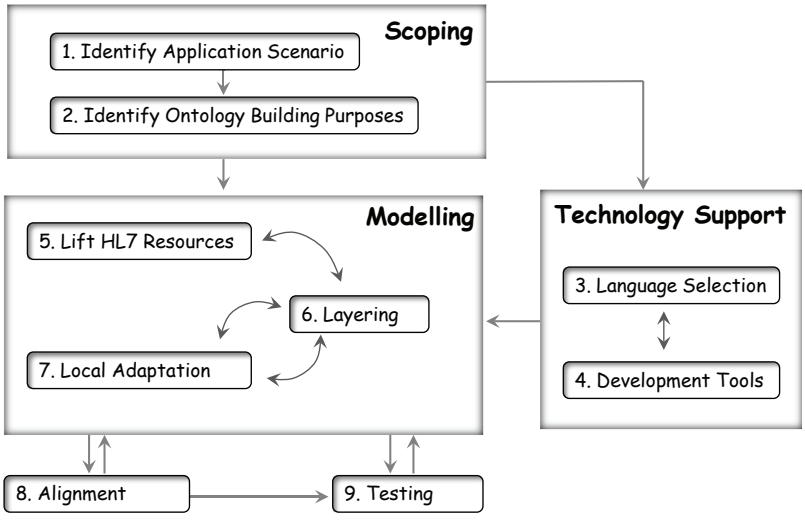

Figure 4.2: PPEPR Ontology Building Methodology

[3]http://www.srdc.metu.edu.tr/webpage/projects/ride/

4.4.1 Scoping

The Scoping phase starts with identifying an application scenario and ends with resources identified within the application scenario and purposes of building HL7 ontologies.

Identify Application Scenario This step takes as input: HL7 UML iteration model; and produces as output: actors, applications, and messages identified within the application scenarios. It is important to first identify an application scenario which drives the need for ontologising the HL7 standard.

For the Step 1 (*i.e.*, Identify Application Scenario), HL7 has a feature to model an application scenario in UML diagrams. Figure 4.3 depicts interactions between the actors of the integration scenario (see Figure 3.1 of Chapter 3) and the sequence of messages exchanged between GP, hospital laboratory applications (*i.e.*, GUH and GOET). In Figure 4.3 the message exchange starts with the interaction event "Observation Order Fulfilment Request" issued by the GP and ends with "Observation Order Complete Acknowledgement" from Galway University Hospital (GUH) and City hospital, Göttingen (GOET). As discussed in the Chapter 3, in HL7 terminology three important entities can be identified from the interaction diagram:

1. **Application Roles:** Order Placer (the application that requests the lab order), Result Receiver (the application that receives the lab order request), and Order Fulfiller (the application that actually performs the lab order).

2. **Messages:** Order Request (a payload envelops within the interaction event "Observation Order Fulfilment Request"). Similarly, the other nine interactions have respective payload messages.

3. **Message Exchange Patterns (MEP)**: it is important to mention that, in context of Version 2 and Version 3 interoperability, applications roles may comply with a particular version (Version 2 or Version 3) or combination of the two versions. For example, the Order Placer (GP) could be Version 3 compliant while the Order Fulfiller (Hospital) has a Version 2 application.

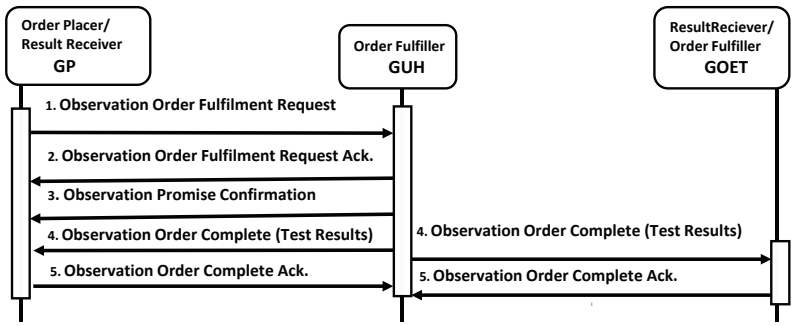

Figure 4.3: HL7 Application Integration Diagram for the Integration Scenario

An identification of a scenario provides the scope for the resources that may be available from existing applications as well as from the standard itself. The next step is to establish the purposes of ontologising the identified scenario and respective resources.

Identify Ontology Building Purposes This step takes as input: resources identified within the Step 1 (*e.g.*, XML Schemas, Messages, UML Models); and produces as output: purposes for building HL7 ontologies. We have categorised four major purposes for ontologising HL7 applications.

- **Information Exchange and Data Integration:** A HL7 ontology can provide formal vocabularies of terms used in HL7 applications. This consistent set of vocabularies will allow the exchange of data between systems using these vocabularies. In addition to describing concepts and relations, a HL7 ontology can also describe constraints and additional constructs (*e.g.*, functional properties) that can make the task of data integration much sensible for application integration scenarios.
- **Vocabulary Sharing:** In certain cases the purpose of an ontology could be to share formal and consistent sets of vocabularies without setting requirements for rigorous data integration or information exchange.
- **Alignment:** The healthcare domain consists of several sub-domains from medical and life sciences area. No single domain covers all the concepts required to describe a patient and clinical treatments. HL7 ontologies define concepts for constructing patient records, while medical ontologies describe clinical and life science concepts. A complete description of a patient record needs both sets of vocabularies. Therefore, the cross-domain alignment of healthcare ontologies will allow access to patient records across multiple domains and sub-domains.
- **Reasoning:** HL7 ontologies can also be used to reason over concepts and relations between HL7 artefacts (*e.g.*, classification, consistency check) as supported by DL reasoners.

The identification of purposes would ultimately influence the next development phases. For example, in the case of vocabulary sharing, it might be suggested to select light-weight ontology language like RDFS. On the other hand, if reasoning is the main purpose of using ontologies then expressive languages like OWL could be a more appropriate candidate. The output of the scoping phase (*i.e.*, identified HL7 resources and ontology building purposes) is input to the Technology Support and modelling phases.

4.4.2 Technology Support

The main goal of the Technology Support phase is to identify list of tools and a language for building HL7 ontologies. The Technology Support phase starts with HL7 resources and ontology building purposes identified within the Scoping phase and ends with list of supporting tools that may automate the overall ontology building process.

Before starting the modelling phase, it is important to analyse relevant ontology languages and tools support. Language selection depends much on constructs supported by the ontology languages

and ontology building purposes identified in the scoping phase. The technology support phase should also analyse tool support for constructing and testing the ontologies.

The selection of ontology languages depends on the richness of axioms that any application scenario may demand. For example, RDFS has limited support to model data types, constraints, and concepts relationships. In the case of modelling advanced data properties and complex relationships between domain concepts, OWL could be a natural choice. OWL supports advanced data properties for modelling complex Healthcare and Life Sciences (HCLS) use cases [57].

Tool support is important for the development, maintenance and usage of ontologies in applications. Different types of ontology tool support are emerging. Some of them are specialised for a specific task, but heavyweight integrated-tools are also available. Tools provide the following support for ontology building activities:

- **Speeding-up** ontology modelling process. User friendly editors support visualisation and ontology construction.
- **Editing** existing ontologies. Updates and corrections are an essential part of the ontology building process. Tools support ontology change management and version control.
- **Exporting/importing** ontology in various encoding formats. It is important to use and publish ontologies for healthcare applications using different ontology languages, *e.g.*, RDFS, WSML, OWL.
- **Advanced usages** such as alignment, merging, reasoning, evaluation of ontologies. Integration and quality enhancements like ontology evaluation is supported by specialised plugins or tools that support integrated ontology building environment.
- **Testing** consistency of the ontologies using reasoners. Ensuring consistency and correctness is at the core of ontology building activities.

Once the ontology language and corresponding tool support are identified, then the next task is the construction of ontologies. The next section describes the modelling phase which includes three ontology building steps.

4.4.3 Modelling

The modelling phase starts with the task of lifting HL7 resources and ends with the local adaptation of ontologies. The overall goal of this phase is to build HL7 ontologies that have conformance with the HL7 standard as well as with deployed applications.

Lifting HL7 Resources The Lifting step takes as input: HL7 resources in XML; and produces as output: HL7 global ontologies (by lifting HL7 *coreSchemas*) and message ontologies (by lifting local message schemas).

We have categorised HL7 resources into two types (see Chapter 2): (i) a conceptual model: artefacts that commonly apply to underlying applications. The Version 3 conceptual model consists

of vocabularies and datatype definitions available as UML and XML Schema specifications. While Version 2 has datatypes, segments and fields specifications published as XML Schemas. Each of the versions has a different set of conceptual models and the reason behind this difference is the approaches taken for modelling clinical messages. Version 3 defines the semantics of datatypes, vocabularies, and messages using UML notations. In Version 2, messages are constructed according to Electronic Data Interchange (EDI) structure. However, equivalent XML Schemas are available; (ii) message schemas: these are generated from local domain models (*e.g.*, DMIM, RMIM for Version 3) or constructed (for Version 2) as per the the clinical messaging requirements. Several semantic similarities exist between the conceptual models of both versions. Ontology engineers can take two possible routes to ontologise HL7 Version 3:

Route 1: Version 3 UML models are stored as Model Interchange Format (MIF) [109] files and HL7 tools export UML models into message schemas (*i.e.*, XML Schema).

- **Conceptual model:** One possibility is to convert MIF repositories to corresponding ontologies. UML and ontologies have similar notions (class, subclass, attribute/property) for knowledge modelling. However, a significant hurdle for this conversion is the presence of extensive system level information within MIF files (*e.g.*, when the model was created, who created it, machine configuration). It is hard to filter out a conceptual hierarchy from MIF files. Some attempts have been made to convert HL7 MIF to OWL ontologies and to be reported a tedious task [110]. To provide automation in this task MIF-OWL/WSML conversion rules can be defined using a transformation language such as XSLT.
- **Message schema:** Once the conceptual model is converted, message schemas and instances can be ontologised (XML(S)→Ontology) using XSLTs.

Route 2: Conceptual models of the two versions are available as XML Schemas (also known as HL7 *coreSchemas*). These *coreSchemas* are arranged in special ways to represent UML notations (class/subclass/attribute) within XML structure.

- **Conceptual model:** *coreSchemas* represent conceptual parts of HL7 messaging framework. Specialised transformation rules (XSD to WSML/OWL) can be defined to lift schema definitions.
- **Message schema:** Similar transformation rules (XSD to WSML/OWL) can be applied to convert message schemas.

We have taken Route 2 where one set of transformation rules (*i.e.*, XML Schema→Ontology) is used to lift both the conceptual and the message schemas without the intermediate transformation to UML/MIF. In the case of Route 1, two sets of transformation rules (*i.e.*, XML Schema→Ontology and MIF→Ontology) are required. However, the advantage of Route 1 is the similarity between UML and ontology languages, where main correspondences between UML and ontological artefacts (*e.g.*, class, subclass) naturally fit together.

Regardless of what route we take, domain and ontology experts are required to sit together for finding and/or verifying correspondences between HL7 artefacts and ontological constructs. Chapter 5 presents an automatic mechanism to convert HL7 *coreSchemas* and message schemas to an

ontological format. Then next section describes the arrangement of the global and local ontologies resulting from core and message schema transformations.

Layering: The Layering step takes as input: HL7 global and message ontologies; and produces as output: layered HL7 global and local ontologies. The two main important approaches for integrating global and local resources are Global-as-View (GaV) and Local-as-View (LaV) [111]. In the GaV approach a global schema is created as a view over source local schemas. On the contrary, the LaV approach creates local schemas as views over the global schema. There have been initiatives to apply the GaV and LaV approaches for ontology-based integration systems [112]. However, in our case both these approaches cannot be applied in a straightforward way because of: (i) Multiple global ontologies: the PPEPR methodology allows for multiple global ontologies (*e.g.*, HL7 Version 2, Version 3, SNOMED) so that local ontologies do not have to commit to one overarching standardised ontology; (ii) Independent global and local ontologies: ontologies are created independently from the different sets of HL7 schemas. Global ontology is created from the schema (*i.e.*, *coreSchemas*) provided by the HL7, while local ontologies generally originate from different groups (hospitals, countries) where each group commits to different standards/policies; and (iii) Independent global and local alignments: alignments at different levels (global and local) enable agreement at the upper layer and the resolution of local differences at a separate level.

The layering task arranges ontologies into global (shared by all HL7 users) and local (used within a particular domain) spaces. Figure 4.4 shows that global ontologies are created in a top-down fashion from HL7 *coreSchemas*, while local ontologies are created from the schemas used in locally deployed HL7 applications.

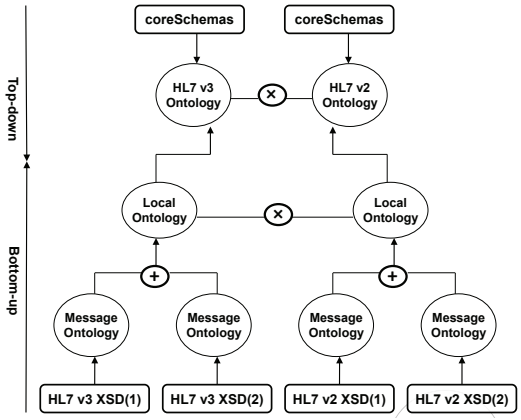

Figure 4.4: PPEPR Methodology: Layering

In Figure 4.4 the $\oplus$ symbol means a merge of ontologies and the $\otimes$ symbol means alignment of ontologies. The alignment task exists between local ontologies as well as between global ontologies. The PPEPR methodology aims to reduce the bi-lateral correspondences between local ontologies by using global ontologies. However, considering the practicalities of HL7 applications and the flexibility that HL7 allows to create local vocabularies and codes, it is impossible to completely avoid bi-lateral correspondences between local ontologies.

In Figure 4.4 local ontologies are created from merging message ontologies. As discussed in the

Lifting phase, XML Schemas of domain messages (*e.g.*, lab, admin, finance) are lifted to message ontologies. For example, it might happen that a particular use-case requires ten XML Schemas to represent all lab messages that should be exchanged between interacting applications. In that case all ten schemas are lifted to corresponding message ontologies. The message ontologies then refer to global ontologies, if matching correspondences are available between them. However, two conceptual ambiguities exist between message ontologies of the same domain: (i) semantically similar concepts are named differently (ii) corresponding concepts are represented at different structural levels. These ambiguities arise because local systems have flexibility and choices in the design of clinical messages.

To deal with both issues (*i.e.*, different naming schemes and structural differences) one option is to provide horizontal alignments between message ontologies. However, maintenance of alignments would be a significant burden. To deal with the problem of several small message ontologies and the maintenance of their alignments, we propose a six substeps process (see Section 7.4) within the Layering step which uses existing alignments to merge semantically similar concepts and properties into a single concept or property. Chapter 6 describes in details about discovering alignments between message ontologies and Chapter 7 (Section 7.4) proposes the six step process that uses alignments to merge message ontologies. This way, ontologies could be maintained at a reasonable size. This is possible as local systems have more control of their message ontologies than the global ontology. An additional task would be to update references within message instances according to the newly created local ontology.

The meeting points of the up (bottom-up) and down (top-down) arrows in Figure 4.4 (left hand side), require extensions within local and global ontologies to find suitable correspondences between them. The next section describes mechanisms and choices for extending both types of ontologies.

Local adaptation The final Local adaptation step within the Modelling phase takes as input: HL7 global and local ontologies; and has as output: Extended global and local ontologies meeting local requirements.

The notion of local adaptation was first proposed by DILIGENT (see Section 4.2.4). DILIGENT local users adapt the global ontology by introducing local changes, and a central board controls the contents of the global and local ontologies. In fact, the HL7 standard works on a similar paradigm where country specific changes are maintained by respective HL7 groups. A central harmonisation committee (like DILIGENT central board) is responsible for introducing local changes within the central RIM conceptual model. On a similar line, PPEPR global and local ontologies originates from two sources, HL7 standard and locally deployed XML Schemas. The local adaptation step is motivated by normal practices where HL7 applications diverge from the standard guidelines by modifying local XML Schemas and by introducing local vocabularies.

The local adaptation phase ensures that: (i) local ontologies are generalised enough to resemble the concepts defined in global ontology, and (ii) global ontologies are specialised enough to resemble the concepts defined in the local ontologies. We categorise the local ontologies into two parts: (i) local concepts, and (ii) local constraints. As said in previous sections, local concepts generally represent the consensus of a local group. Consequently, their understanding is limited to the local environment. Similarly, several types of constraints (*e.g.*, drug policy, access policy) describe restrictions imposed by the local environment. Adaption of local constraints is a much more complex topic because: (i)

it is important to know the origin of a constraint while integrating global and local ontologies, and (ii) ontology alignment or importing allows the transfer of constraints from the original publication site, means that, restrictions would then be applied to other sites. Chapter 8 investigates the issues of context-dependent local concepts and constraints.

```
concept ActObservation
subConceptOf Act

concept ActSpecimenObservation
subConceptOf ActObservation

concept ActGenomicObservation
subConceptOf ActObservation

concept ActClinicalTrial
subConceptOf ActObservation

concept ActCondition
subConceptOf ActObservation

concept ActPolicy
subConceptOf Act

concept ActJurisdictionalPolicy
subConceptOf ActPolicy

concept ActOrganizationalPolicy
subConceptOf ActPolicy

concept ActScopeOfPolicy
subConceptOf ActPolicy

concept ActStandardOfPolicy
subConceptOf ActPolicy
```

Figure 4.5: Snippet of the Global Ontology (RIM): WSML syntax

```
Class: ActObservation
SubClassOf: Act

Class: ActSpecimenObservation
SubClassOf: ActObservation

Class: ActGenomicObservation
SubClassOf: ActObservation

Class: ActClinicalTrial
SubClassOf: ActObservation

Class: ActCondition
SubClassOf: ActObservation

Class: ActPolicy
SubClassOf: Act

Class: ActJurisdictionalPolicy
SubClassOf: ActPolicy

Class: ActOrganizationalPolicy
SubClassOf: ActPolicy

Class: ActScopeOfPolicy
SubClassOf: ActPolicy

Class: ActStandardOfPolicy
SubClassOf: ActPolicy
```

Figure 4.6: Snippet of the Global Ontology (RIM): OWL syntax

The PPEPR methodology deals with adaptation of local concepts, thats is, how locally defined concepts fit together with global ontologies. Figure 4.5 shows a snippet of the global ontology (Version 3 RIM), which describes the top level concepts of local ontologies shown in Figure 4.7 and Figure 4.8. For example, the "ActObservation" class represents all types of clinical observation possible within a clinical environment and "ActGenomicObservation", "ActClinicalTrial", "ActSpecimenObservation", etc. classes are specialisations of "ActObservation". Each of the these specialisations require "lab test order" in different clinical scenarios. That means, a local concept such as "ObservationRequest" (see Figure 4.7) cab be inherited these generalisations ("ActGenomicObservation", "ActClinicalTrial", "ActSpecimenObservation"). In addition to the local adaptation proposed by DILIGENT, the PPEPR methodology suggests three approaches for the adaptation of the local concept "ObservationRequest". These three approaches are refinements of approaches proposed in the Enterprise Ontology (see Section 4.2.1):

Top-Down: Extend the global ontology with more specialised concepts that resemble the concepts defined in local ontologies. For example, each of the concepts "ActGenomicObservation" and "ActClinicalTrial" could be further extended with a lab specific concept like "ActClinicalTrial-

```
Class: ObservationRequest
SubClassOf: ActObservation

Class: SpecimenObservation
SubClassOf: ActObservation

Class: Observer
SubClassOf: RoleClass

Class: DiabeticType2Observation
SubClassOf: SpecimenObservation
```

Figure 4.7: Snippet of Lab Observation Ontology-1: OWL Syntax

```
Class: ObservationOrder.POOB_MT210000UV
SubClassOf: ActObservation

Class: Observer.POOB_MT210000UV
SubClassOf: RoleClass

Class: HemoglobinObservation.POOB_MT210000UV
SubClassOf: ActObservation
```

Figure 4.8: Snippet of Lab Observation Ontology-2: OWL Syntax

LabObservation" or "ActGenomicLabObservation", to represent that all lab tests specific to clinical trials are denoted by "ActClinicalTrialLabObservation".

Bottom-Up: Extend the local ontology with more generalised concepts that resemble the concepts defined in the global ontology. For example, a super class "ActClinicalTrialLabObservation" for the local concept "ObservationRequest" can be added to the local ontology, such that, appropriate inheritance like "ActClinicalTrialLabObservation" subClassOf "ActClinicalTrial" could be established in the local ontology. For instance, DiabeticType2Observation from Figure 4.7 is a local concept and inherited from super class "ActObservation" via "SpecimenObservation".

Middle-Out: As described above, concepts ("ActGenomicObservation" or "ActClinicalTrial") in the global ontology are defined at a higher level of abstraction, which allows a local concept to inherit from all of them. Therefore, instead of specialising or generalising global or local concepts, another approach could be to add a specialised class for the lab test as a sibling of the above two concepts ("ActGenomicObservation" or "ActClinicalTrial"). For example, Alpha fetoprotein (AFP) is an observation for detecting liver cancer, and the RIM (January 2011 Ballot) [4] does not include any specialised classes for cancer diseases. It may be appropriate to add "CancerObservation" on the same level as "ActGenomicObservation" or "ActClinicalTrial", and then extending "CancerObservation" with the "AlphaFetoproteinObservation" concept.

These three approaches could be applied independently or in combinations depending on requirements from different clinical scenarios. Enterprise Ontology suggested the middle-out approach. However, considering the heterogeneities of different clinical scenarios, we intentionally avoid any fit-for-all suggestion. Apart from concepts that construct clinical messages, we notice in Figure 4.5 that specific concepts are included within the RIM model to represent different kinds of policies. Chapter 8 describes issues related with local constraints or policies within healthcare applications.

The next Section describes three approaches for testing the ontologies built in the modelling phase.

4.4.4 Testing

The Testing phase takes as input: HL7 global and local ontologies; and has as output: consistent ontologies.

Once the modelling phase completes, it is important to ensure that ontologies are ready for publishing as well as matching the domain requirements. In the context of HL7 applications, four types of testing can be performed (i) testing the validity and consistency of an ontology (ii) whether the ontology matches the requirements identified in the scoping phase (iii) testing correspondences between local and global ontologies, and finally (iv) HL7 conformance test.

Testing validity and consistency: Validity testing is about ontology correctness, *i.e.*, is serialisation valid for the chosen ontology language. Ontology editors generally parse input ontologies and report errors in the ontology serialisation. Some dedicated online tools are also available for checking the ontology language syntax. This type of testing can be avoided if the ontology editors ensure that its output is valid syntax. Validity testing is important where ontology markup is edited manually in any standard text editor.

Consistency testing is about the coherence of the conceptual model. It checks the consistency of the conceptual model within the representation formalism (*i.e.*, ontology language) used during the modelling phase. The output of consistency checks generally identifies all conflicting concepts and relationships. Some advanced ontology reasoners can also provide further information on debugging and repairing defects in the ontologies.

Requirement Matching: Requirement Matching is applied for verifying whether the ontology matches requirements of the identified scenario during the scoping phase. The techniques for verifying requirements may differ depending on the complexity of the scenario. For example, in the case of well documented HL7 system, it might be sufficient to compare requirements and ontologies side-by-side.

Testing Correspondences: This type of testing is only possible where alignments are stored separately from the ontological definitions. As said above, few ontology editors support this separation. In healthcare scenarios, especially when considering standards like HL7, the amount of correspondences between ontologies is generally high. In such cases it is often hard to find the causes of the conceptual conflicts. Our experience using the PPEPR methodology suggests using ontology consistency testing (with or without correspondences between two more ontologies) to identify the causes of the conflicts. For instance, conflicts may occur due to incorrect correspondences or due to errors in structural definitions. This way, errors are identified at separate levels and corrective measures are applied to the right location.

HL7 conformance: Conformance testing is a post development step. When Version 3 instances are transformed to equivalent Version 2 instances (or vice-versa), and if transformed XML instances validate against standard schemas, then we consider that ontologies and their alignments are correct as well as the mediation of instances are successful.

The next section compares the PPEPR methodology across various dimensions of methodological support and the output of each methodology discussed in this chapter.

4.5 Comparing Ontology Building Methodologies

4.5.1 Features

We have discussed (see Section 4.2) the extent to which existing methodologies (Enterprise ontology, METHONTOLOGY, On-To-Knowledge, DILIGENT) support three features required for ontologising HL7. This section provides a summary, comparing ontology building methodologies and their support for three HL7 specific identified features. In Table 4.1 a plus symbol (+) means feature is present, a minus symbol (-) indicates no support or feature is not present, and a plus minus together (+/-) means a partial support indicating that the methodology has allocated activities/phases to support the feature, but a detailed guideline or mechanism is missing.

	Reusability	Layering	Local Adaptation
Enterprise Ontology	+/-	-	-
METHONTOLOGY	+/-	-	-
On-To-knowledge	+/-	-	-
DILIGENT	+/-	+/-	+
PPEPR	+	+	+

Table 4.1: Ontology Building Methodologies: Feature comparison

Enterprise Ontology method provides a set of steps and activities to build a central ontology from scratch. The method suggests the integration of existing ontologies, but the detailed mechanism and integration steps are missing. Reusability of non-ontological knowledge sources and the adaptation of local knowledge bases are outside the scope of the Enterprise ontology. In our case, the three ontology building strategies proposed (top-down, bottom-up, and middle-out) have been applied during the local adaptation of ontologies. This implies that we generalise the global ontology enough, such that concepts from local ontologies can resemble concepts from the global ontology. But the methodology only considers the specialisation or generalisation of a central ontology where layering or reusability factors are not addressed appropriately.

METHONTOLOGY was proposed to align software engineering and ontology building practices. METHONTOLOGY recognises an ontology as an element of a software development model. The methodology proposes an extensive set of ontology building activities following the standard IEEE guidelines. In standard software engineering practices UML models are the core elements used in software development process. However, the steps required to reuse UML models in the ontology based environment are missing. Also, only a high level description for adapting application specific resource is described. METHONTOLOGY does not consider different levels of granularity during the reuse of ontologies, for example, separate layers of ontologies. Such granular treatment of information modules is normal in software engineering methodologies. When an ontology to be reused has to be modified, METHONTOLOGY proposes to carry out the ontology reengineering activity. However, for this activity, the methodology only mentions the main activities to be carried out and does not provide detailed guidelines.

On-To-knowledge is an application specific methodology. The ontology development process starts with a feasibility study, which requires analysis and the use of the ontology within any application domain. The main outcome of this methodology is the detailed guidelines for ontology refinement and evaluation. However, provision for the reusability and the local adaptation of knowledge sources is still weak in this methodology. In the context of reusing non-ontological resources (UML models and XML Schema) On-To-knowledge does not specify any guidelines.

DILIGENT methodology is close to the two features we have identified for ontologising HL7. The methodology starts with building a core (small) global ontology and then extends it with local changes to suit the application specific needs. In the case of HL7 systems, non-ontological resources at global and local levels are key elements in building HL7 ontologies. DILIGENT does not specifies any detailed guidelines for reusing non-ontological resources (UML models and XML Schema). The main contribution of this methodology is the notion of local adaptation and the local update of ontologies with a mechanism to synchronise them with global ontologies. In the case of HL7 systems, global and local ontologies are developed independently, while in DILIGENT global ontologies themself go through local adaption. The DILIGENT methodology is a close match with our requirements and the PPEPR methodology extends and refines DILIGENT by targeting HL7 applications.

4.5.2 Strategy, Method, and Output

The PPEPR methodology inherits development guidelines from existing methodologies by extending, refining, and introducing steps that are specific to the HL7 framework. The previous section has provided a high level comparison of PPEPR features and how existing methodologies address them.

	L.C[4]	A.D[4]	S.I.C[4]	U.C.O[4]
Enterprise Ontology	No	Independent	Middle-Out	NO
METHONTOLOGY	Evolving prototype	Independent	Middle-Out	On Resource Availability
On-To-knowledge	Evolving prototype and Cyclic	Dependent	Top-down, Middle-out, Bottom-up	On Resource Availability
DILIGENT	Evolving prototype and Cyclic	Independent	Top-down, Middle-out, Bottom-up	Yes
PPEPR	Ordered and Cyclic	Dependent	Top-down, Middle-out, Bottom-up	No

Table 4.2: Ontology Building Methodologies: Comparing Conceptual Construction

However, it is also important to compare the specific offerings or limitations of PPEPR with the existing methodologies. We use three dimensions: (i) ontology construction strategy (ii) similarities/differences of development steps, and (iii) the use of ontologies built by specific methodologies. The comparison framework is based on the one proposed in [10]. Table 4.2 summarises ontology construction strategies adopted by existing methodologies and in PPEPR:

[4]L.C: Life Cycle; A.D: Application Dependency; S.I.C: Strategy to identify concept; U.C.O: Use of core ontology.

1. **Life Cycle Proposal:** The ontology life cycle identifies a set of stages through which ontologies move during their life time. It also describes which activities have to be performed at each stage and how they are related. We observe that METHONTOLOGY, On-To-Knwoledge, DILIGENT has evolving and cyclic life cycle. That means, future changing requirements of users evolve an ontology and the development steps are cyclic or interactive to perform revisions and corrections. The Enterprise Ontology describes a set of development activities without considering the life cycle and order of development steps. PPEPR is ordered and cyclic, where development steps are performed in sequence. The modelling phase, the alignment phase, the testing phase, and the Step6 to Step7 within the modelling phase (see Figure 4.2) are iterative until global and local ontologies are constructed and aligned well together. One main limitation of the PPEPR methodology is the lack of an evolving ontology life cycle.

2. **Application dependency:** This feature is related to the degree of dependency of the ontology with applications using it. Application-dependent means that the ontology is built on the basis of applications that use them, while application-independent indicates that the methodology is totaly independent of any application. On-To-Knowledge requires a feasibility study on an application, which establishes a set of requirements before starting the ontology development. Similarly, PPEPR requires the identification of an application scenario and the scoping of the overall ontology building requirements around the identified scenario.

3. **Strategy to identify concepts:** There are three possible strategies to identify concepts: from the most concrete to the most abstract (bottom-up), from the most abstract to most concrete (top-down), or from the most relevant to the most abstract and most concrete (middle-out, see Section 4.2.1). On-To-Knowledge, DILIGENT and PPEPR have similar strategies in applying bottom-up, top-down, and middle-out mechanisms for developing the conceptual hierarchy. However, PPEPR applies these mechanisms within and between two different sets of ontologies (global and local) rather than a single central ontology.

4. **Use of core ontology:** This indicates the use of a core ontology as a starting point in the development of a domain ontology. DILIGENT recommends using a core ontology as a start-up for further developments phases. The core ontology grows as greater consensus is built. In the case of PPEPR, consensus is implicit within the HL7 standard. Thus, every HL7 resource is a prospective ontological artefact. The challenge here is to build ontologies from HL7 resources.

PPEPR Steps	I.P[5]	I.R[5]	L.S[5]	D.T[5]	L.R[5]	L.[5]	L.A[5]	A.[5]	T[5]
Enterprise Ontology	Described	NP	Described	Described	NP	NP	NP	NP	NP
METHONTOLOGY	Described	NP	Described	Described	NP	NP	NP	General	Described
On-To-knowledge	Described	NP	Described	Described	NP	NP	NP	General	NP
DILIGENT	Described	NP	Described	Described	NP	Described	Described	General	NP

Table 4.3: Ontology Building Methodologies: Comparing Development Steps

[5]I.P: Identify Purposes; I.R: Identify Resources; L.S: Language Selection; D.T: Development tools; L.R: Lifting Resources; L.: Layering; L.A: Local Adaptation; A.: Alignment T.: Testing

Table 4.3 summmarises the ontology building steps in all methodologies. Three types of values are filled in each cell. The value "Described" means that the corresponding methodology provides detailed description of that particular step. The value "General" means that the methodology identifies the step but described at a high level irrespective of any particular domain. The value "Not Proposed (NP)" means that the methodology does not mention the corresponding step.

	O.C[6]	D.[6]	P.[6]
Enterprise Ontology	Enterprise Ontology	Business Enterprises	Enterprise Project
METHONTOLOGY	OntoRoadMap[113], $(KA)^2$Ontologies[114], MK-BEEM ontologies[115]	Environment, Knowledge Management, e-Commerce	OntoWeb
On-To-knowledge	OntoShare[116]	Virtual communities	SemiPort, AIFB portal, SwissLife
DILIGENT	Argumentation Ontology[30]	Tourism, Judicial	SEKT, SWAP
PPEPR	HL7V2, HL7V3	Healthcare	PPEPR

Table 4.4: Ontology Building Methodologies: Comparing Methodological Output

Table 4.4 shows the use of the ontologies built according to different methodologies. First, the major ontologies created by the methodologies are identified. Second, the domain of each ontologies and finally, projects where they are applied show their use and impact within the respective communities.

4.5.3 Tool Support

Tool support is important for speeding-up and quality in the ontology building process as well as the overall maintenance of ontologies. Table 4.5 shows tool support suggested by each of the methodologies. "Not Supported (NS)" means that the methodology does not recommend any specific tool and all phases can be supported by general ontology building tools (*e.g.*, Protégé, WSMT/WSMO Studio). The METHONTOLOGY has extensive tool support for all the development activities. Tools such as WebOde [101] have been specially developed to support all the METHONTOLOGY activities.

The PPEPR methodology does not suggest any specific tool as all development steps can be performed using general ontology building tools like Protégé or WSMT/WSMO Studio. However, we outline each PPEPR development step and how existing tools can be helpful in carrying out the respective tasks:

	Enterprise Ontology	METHONTOLOGY	On-To-knowledge	DILIGENT	PPEPR
Tool Support	NS	WebODE,OntoStudio, Protégé	OntoEdit	NS	NS

Table 4.5: Ontology Building Methodologies: Methodologies and Tool Support

- **Step 1 and Step 2:** These steps target the HL7 messaging environment. Successful completion of the scoping phase requires domain analysis and HL7 documentation tools (described in the next section) could be used in conjunction with tools that support ontology requirements

[6]O.C: Ontology Construction; D.: Domain; P.: Project.

gathering and specification (*e.g.*, OntoKick plugin for OntoPrise[7]). OntoKick plug-in supports the collaborative generation of requirement specifications for ontologies. OntoKick can be used for the description of important aspects of the ontology, such as the domain and goal of the ontology, design guidelines, available knowledge sources (*e.g.*, domain experts, reusable ontologies), potential users, use cases, and applications supported by the ontology.

- **Step 3 and Step 4:** These steps involve the selection of tools and languages to match the scope of an application scenario, which require manual input from the domain experts.
- **Step 5:** Tools support for reusability of ontological resources has been proposed. For example, ProSe[117] is specifically developed for reusing existing ontologies. However, reusability of non-ontological resources is still at an immature stage. The PPEPR methodology does not recommend any tool for this task, but provides an automatic support for the transformation of non-ontological resources to corresponding ontologies and instances.
- **Step 6:** While merging message ontologies (Step3 to Step6 of Figure 7.20) to a local ontology, the Protégé plugin PROMPT-Suite [16], OntoPrise, or other ontology alignment tools could be used.
- **Step 7:** Generalisation or specialisation of global and local ontologies can be performed using general purpose ontology editors.
- **Step 8:** In case of the HL7 ontologies automated ontology alignment tools does not show impressive results, therefore we devised a semi-automatic alignment solution for the HL7 ontologies.
- **Step 9:** Ontology editors generally include reasoning support provided as plugins from reasoning frameworks. Ontologies can be checked within integrated environment to ensure overall consistency.

The next section describes the tool support and various possibilities to provide ontology documentation as well as reusing HL7 generated documents.

4.5.4 Ontology Documentation

Documentation is a general task that applies to all phases of the PPEPR methodology. Existing methodologies (Enterprise ontology, METHONTOLOGY, On-To-Knowledge, DILIGENT) provide general guidelines for documenting ontological entities. In the HL7 framework, documentation is very well integrated across all development stages and specific tools are available to support the entire documentation process. HL7 supports documentation by:

- **HTML Exports:** This is a Microsoft Visio Plugin (RMIM designer[8]) for designing HL7 domain models. This plugin allows developers to construct, update, and store domain models in Visio environment. This plugin has a feature to generate HTML documentation for the stored models. The HTML document generated is similar to the HL7 ballot[9].

[7] http://www.ontoprise.de/
[8] http://gforge.hl7.org/gf/project/visio-rmim-desi/frs/
[9] http://www.hl7.org/v3ballot/html/welcome/environment/index.html

- **V3 Generator:** The V3 Generator[10] (also known as Schema Generator) accepts domain models and, through a series of transforms, generates HL7 artefacts including: static schemas, interaction schemas, HTML views, and Excel views for static models, data types, and vocabulary.

Ontology developers can identify document resources directly after identifying the purposes of ontology building, *i.e.*, at PPEPR Step 2 (identifying HL7 resources). Once these documents are located within the application environment, developers can reuse them for describing ontological artefacts, *i.e.*, at PPEPR Step 5 (Lifting HL7 Resources). There are several ways documentation could be reused or created for ontologies:

- **Integrated Tools:** Ontology editors generally support a documentation function using specialised plugins. The OWLDoc[11] plugin for Protégé exports HTML documents from ontologies. Similarly, TopBraid[12] and OntoStudio[13] allows to create and edit comprehensive sets of ontology documents in HTML format. Some specialised plugins and tools like OWLViz[14] and SWOOP[15] also support visualisation of ontologies and their graphical documentation.
- **Standalone Tools:** Dedicated ontology documentation tools like SpecGen[16] or OntoSpec[17] generate HTML documents from ontology definitions. For example, they identify classes, properties, and other sources to generate templates of definitions and combine them into the main document source.
- **Embedding:** One convenient way could be to use RDFa [118], embedding ontology definitions within a HTML document. HL7 supports comprehensive documents within the messaging environment. HTML documents generated from HL7 tools can easily include parts of ontology definitions rather than creating documentation from scratch. In this case the same document acts as the source for the ontology definition and the specification, thus blending together implementation and documentation activities.

The ontology documentation phase ends with ontology specifications for broad audiences including domain experts, engineers, and non-technical users.

The ontology building comparison presented above shows how the PPEPR Methodology extends and refines three essential features (*i.e.*, reusability of non-ontological knowledge sources, layering of ontologies, and local adaptation of ontologies) to ontologise the HL7 standard. The modelling phase of the PPEPR Methodology is completely automated and specifically contributes to our first research challenge (*i.e.*, "how to build HL7 ontologies by reusing domain artefacts reside in separate global and local spaces"). The next chapter presents the *Lift HL7 Resources* step of the PPEPR's modelling phase. The *Lift HL7 Resources* step describes a set of rules that automates the transformation of syntactic HL7 specifications (*i.e.*, UML, XML) to an ontological format.

[10] http://gforge.hl7.org/gf/project/v3-generator/frs/
[11] http://www.co-ode.org/downloads/owldoc/
[12] http://topquadrant.com/products/TB_Suite.html
[13] http://www.ontoprise.de/
[14] http://www.co-ode.org/downloads/owlviz/
[15] http://code.google.com/p/swoop/
[16] http://forge.morfeo-project.org/wiki_en/index.php/SpecGen
[17] http://moustaki.org/ontospec/

4.6 Summary

In this chapter we have presented and compared methodological approaches for ontologising the HL7 standard. We have discussed several initiatives for ontologising HL7 but a detailed step-by-step methodology to ontologise standard artefacts as well as deployed applications is missing. We identified three features: (i) reusability of non-ontological knowledge sources, (ii) layering of ontologies, and (iii) adaptation of local ontologies as basic requirements for applying semantics on top of HL7 applications. The PPEPR methodology is scenario-based where the application execution environment sets guidelines for all development phases. The PPEPR methodology extends existing methodologies by: (i) a mechanism to improve the reusability of non-ontological knowledge resources, (ii) a mechanism to arrange and use ontologies developed at different layers, and (iii) a mechanism to adapt local ontologies with respect to global ontology. We have compared PPEPR and other existing ontology building methodologies on various dimensions of methodological support by explaining how PPEPR inherits, refines, and extends existing methodologies. The next chapter presents one of the main focuses of this thesis–Step 5 (Lifting HL7 Resources) within the PPEPR methodology.

Chapter 5

Lifting HL7 Resources

5.1 Introduction

This chapter presents a lifting mechanism that reuses HL7 resources for building HL7 ontologies. We have discussed in the previous chapter that reusing non-ontological structured resources is an essential task for ontologising the HL7 standard. The PPEPR methodology describes two possible routes for ontologising HL7 resources available in the MIF (UML serialisation) and XML formats. In HL7 Version 3, messaging schemas are derived from UML models, which means that XSD definitions use special elements and attributes to represent UML constructs like *inheritance* and *association*. The Version 2 schema shares several similarities (*e.g.*, data type, segments) with the Version 3 schema. However, Version 2 lacks the UML specific XSD elements and attributes. In this chapter we present five guidelines that establish correspondences between Version 3 specific UML constructs, and ontological elements. These guidelines are used to formulate rules for transforming an XSD definition to an ontology in an automatic way.

The first step in the task of lifting is to find the common semantics underlying knowledge, schema, and data languages. Once the semantics of the source language constructs and their mappings with the target language are identified then the transformation can be supported in manual or (semi-) automatic ways. UML and ontology languages like OWL and WSML have many common characteristics. For example, both have a notion of a class which can have instances. In this chapter our discussion along ontology languages is focused around OWL and WSML. Similar to UML, XML shares common characteristics such as type system (simple or complex) that may correspond to ontological classes or extension mechanism corresponding with the notion of ontological inheritance.

This chapter is separated in two parts: first we present related works in transforming a UML model to ontology (Section 5.2). Then we describe five guidelines for establishing correspondences between HL7 UML models and ontology languages. The second part of this chapter presents related works in transforming XML(S) to ontology languages (Section 5.3). Then the chapter presents an automatic mechanism for transforming XML(S) to ontology languages. Finally, the chapter presents an evaluation of HL7 ontologies based on their converge of the domain and methods to develop them.

5.2 UML To Ontology

Transforming a UML model to an ontology and vice-versa has been carried out in two categories. First, some of the initial works in this area proposed a conceptual mapping between UML and ontology constructs. These works have revealed that it is not a straightforward task and domain specific customisation is required to preserve the semantics of both conceptual models. The second category of transformation task is creating mappings between UML and ontology serialisation languages. In standard settings, UML models are serialised and stored in the XML Metadata Interchange (XMI) format. However, in the case of HL7 Version 3 serialisation the format is different. HL7 repositories store UML models in the Model Interchange Format (MIF). Though several tools are available to transform (partial) XMI to an ontology language such as OWL, none of them support MIF to OWL conversion.

Java2OWL [119] suggests a transformation from UML to OWL. This program transforms java classes and instances to OWL using XML Schema data types and cardinality restrictions. Similarly, [120] proposes a transformation method from a UML class model to DAML+OIL [121]. [120] classifies the UML *association* to a detailed relationship for representing the semantics of the association more accurately.

Colomb *et al.* [122] proposed a DL centered method for conceptual mappings between UML, Entity Relation (ER), and OWL. In this approach all models should be converted to DL constructs as a first step. However, loss of some semantic information is inevitable. For instance, the *association* and the *attribute* constructs of UML are semantically different elements, but both elements are mapped to the *role* construct in DL, and then transformed to an OWL property.

Dragan *et al.* [123] proposed the use of UML notations for ontology definition. Authors proposed an Ontology UML Profile (OUP) and a transformation method from the OUP-based ontology to an OWL ontology. The OUP defines its own terminologies and stereotypes for ontology representation, so we cannot apply the proposed method to standard UML models.

Several industrial initiatives have also been taken to map XMI and OWL serialisations languages. The Ontology Definition Metamodel (ODM) from the Object Management Group (OMG) has implementations including an Eclipse tool[1]. Similarly, Sandsoft's[2] Visual Ontology Modeller (VOM) is an add-in to IBM's Rational Rose[3] modelling toolkit. VOM uses UML's built in profiling capabilities and Rational Rose extension mechanisms to facilitate ontology modelling, which enables users to leverage pre-existing UML models and other resources as a basis for ontology development.

We observe above that none of the past works relate directly to our problem of transforming HL7 MIF to ontologies. This problem has been recognised by the HL7 standard [109] and healthcare solution developers like Ringholm[4] have started to build tool support for MIF to OWL transformations. However, these efforts are still at a high level and MIF to OWL tool support is expected in the future. In our work, we have concentrated more on a conceptual mapping between HL7 UML models and ontology languages. We propose five guidelines that establish correspondences between HL7 UML and ontological constructs.

[1] http://www.eclipse.org/m2m/atl/usecases/ODMImplementation/
[2] http://www.sandsoft.com/
[3] http://www-01.ibm.com/software/awdtools/developer/rose/
[4] http://www.ringholm.de/

5.2.1 HL7 UML Models

UML and ontology languages share a similar purpose: modelling the domain concepts. UML and ontology languages provide constructs for representing the relationship between domain concepts and enable the expression of constraints that instances must follow in order to share or exchange meaningful data. There are two major model types in UML - structural and behavioural. The structural model represents the static structure of the objects in a system. Elements in a structure model represent the meaningful concepts of an application, and may include abstract, real-world, and implementation concepts. The behavioural model represents the dynamic behaviour of the objects in a system, including their methods, collaborations, activities, and state histories. We focus on the static and structural aspects of a UML model and consider the UML structural model as the main resource for extracting ontological elements. There are six types of diagrams in a UML structural model: class, component, object, composite structure, deployment, and package diagram. HL7 static model such as RIM mainly uses the class diagram. Figure 5.1 shows a class diagram of the HL7 Version 3 RIM. HL7 uses UML extended notions for modelling a clinical scenario.

HL7 Version 3 Global Model: RIM

The RIM is an upper model shared by all users of the Version 3 system. Figure 5.1 shows six core RIM classes:

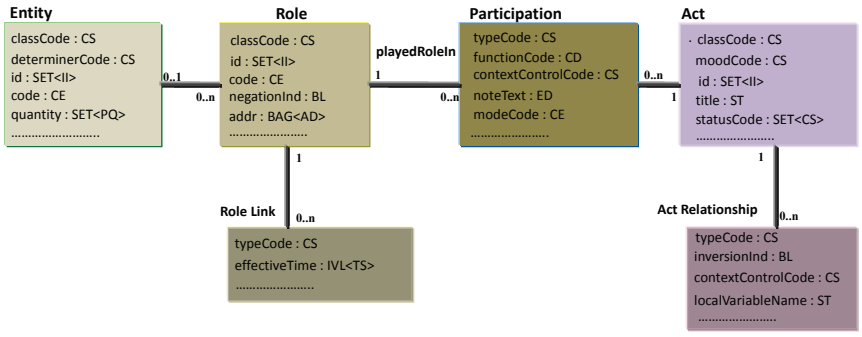

Figure 5.1: HL7 Version 3 RIM Classes

- **Entity:** A physical thing (living and non-living), a group of physical things or an organisation capable of participating in *Acts*, while in a *Role*. The Entity further specialises in classes like, EntityPerson, EntityPlaces, EntityDevices, etc. Each specialised class then introduces additional attributes.
- **Role:** An *Entity* playing the *Role* as identified, defined, or acknowledged by the *Entity* that scopes the *Role*. For example, Figure 5.2 shows the Entity (Sean) playing the *Role* (patient) and participating (as subject) for the clinical *Act* (Lab Observation). The *Role* class further specialises into RoleMember, RoleIngredients, RolePart, etc.
- **Act:** A record of something that is being done, has been done, can be done, or is intended or requested to be done. The *Act* class further specialises into ActObservation, ActClinicalTrial, ActSpecimenOrder, etc.

- **Participation:** An association between an *Act* and a *Role* with an *Entity* playing that *Role*. Each *Entity* (in a *Role*) involved in an *Act* in a certain way is linked to the *Act* by one *Participation*. The *Participation* class further specialises into ParticipationHolder, ParticipationReciver, ParticipationConsultant, etc.

- **RoleLink:** A connection between two Roles expressing a dependency between those Roles.

- **ActRelationship:** A directed association between a source *Act* and a target *Act*.

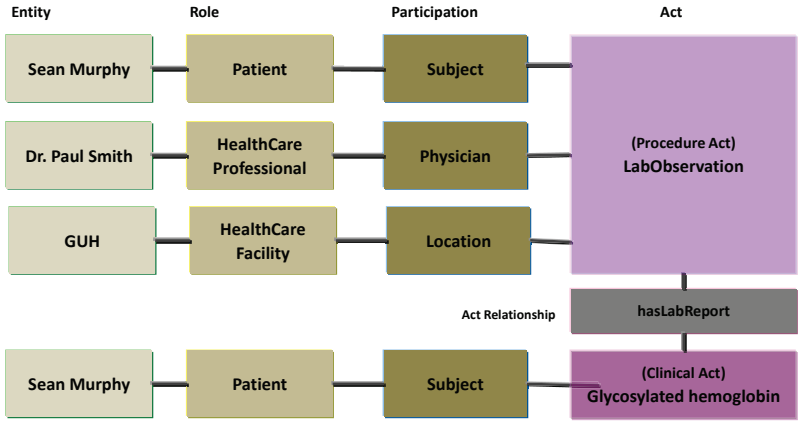

Figure 5.2: Instance of RIM Classes

In addition to the RIM upper model, the Version 3 messaging framework allows us construct a local UML model specific to a healthcare institution. Local UML models inherit from the upper model to specialise the six core classes (*Entity*, *Act*, *Role*, *Participation*, *RoleLink*, *ActRelationship*) for constructing clinical messages.

HL7 Version 3 Local Model: RMIM

The HL7 Version 3 local model is called the Refined Message Information Model (RMIM). The RMIM designer (a Microsoft Visio Plugin) provides a modelling environment and a repository to store upper (RIM) and local models.

Figure 5.3 shows an example RMIM model for the "lab observation". The rectangular boxes with curved arrows are labels and not part of the RMIM model. For example, "ObservationOrder" and "ObservationDefinition" are classes within the "lab observation" model inheriting from the *Act* class in the upper RIM model. Similarly, the author, performer, subject, and recordTarget classes inherit from the *Participation* class. For example, Figure 5.4 shows a Class Refinement Editor of the RMIM designer that describes the local name "ObservationOrder" and the inheritance from the *Act* class. The definition class between the two *Act* classes (ObservationOrder and ObservationDefinition) describes the relationship (ActRelationship) between two clinical activities.

For instance, in this case, ObservationOrder describes the clinical activity of ordering a patient's medical observation and ObservationDefinition is the act of creating clinical readings (*e.g.*, clinical report or medical history). Classes R_AssignedPerson, R_AssignedOrganization, R_Specimen, and

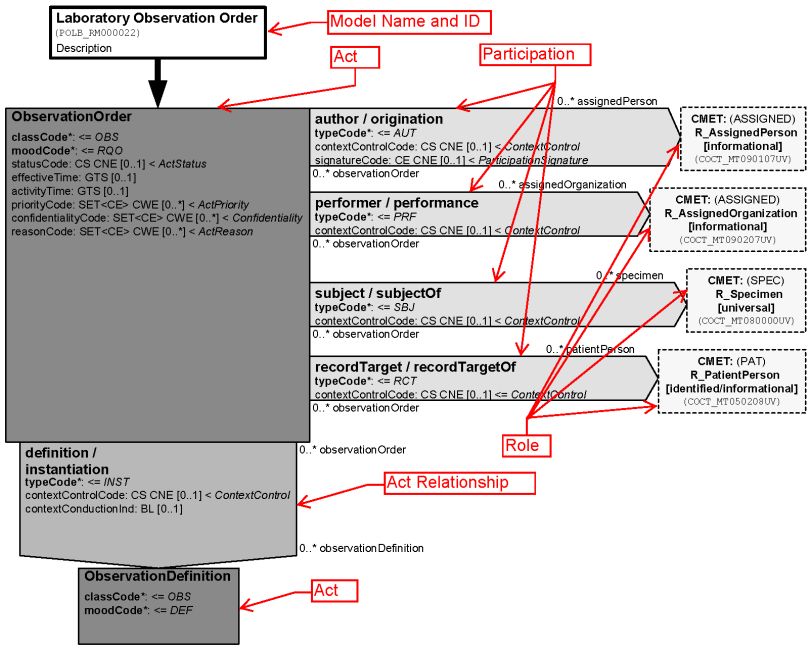

Figure 5.3: Local HL7 UML Model

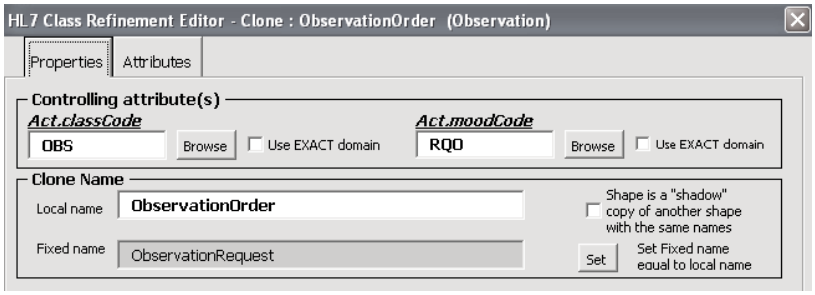

Figure 5.4: Observation Order Inheriting OBS (ActObservation)

R_PatientPerson are specialisations of the *Role* class. Finally, the rectangular box at the top (Laboratory Observation Order) is called the entry-point for this local "lab observation" model. The Entry-point assigns a name and unique id (*e.g.*, POLB_RM000022) for the local RMIM Model.

5.2.2 HL7 UML models to Ontology: Guidelines

Table 5.1 presents detailed correspondences between UML, WSML, and OWL. Based on the mappings presented in Table 5.1, we define five guidelines for transforming a HL7 UML model to WSML and OWL ontologies.

Ontology Class: A class is the most fundamental concept in UML and ontological models. As discussed above, there are some differences between the traditional UML or OO programming class and the ontology class. We consider that, if the type of a UML model element is a class or a component, then the model element corresponds to *owl:Class* or *wsml:concept*.

Guideline 1 *An ontology class can be created from on a UML model element, if the element is a*

UML	WSML	OWL
Class	concept	Class
Attribute	attribute	ObjectProperty|DataProperty
Association	relation	ObjectProperty
Class specialisation	subConceptOf	SubClassOf
Associations specialisation	subAttributeOf	SubPropertyOf
Note	annotations	Annotations@label|comment
Association source class	Domain	Domain
Association target class	Range	Range
import	importsOntology	Import
multiplicity range X..Y	(X..Y)	X = min, Y = max

Table 5.1: Mapping Between UML, WSML, and OWL

class, component, or part.

For example, we can find 4 classes in Figure 5.1, and create ontology classes: Entity, Role, Participation, and Act.

Ontology Properties: OWL and WSML distinguish two kinds of properties -- object and data type properties. Their domain is always a class. The range of an object property is a class, while the range of data type properties is a data type.

Guideline 2 *An ontology object property can be created from a UML model element, if the element is a class attribute and its type is a class.*

For example, the two lower elements (ActRelationship and RoleLink) in Figure 5.1 are UML classes as per the standard specification. However, the semantics of ActRelationship and RoleLink is much closer to an ontology property as their purpose is to relate instances of the Act and Role classes. Guideline 2 suggests this modification to capture the domain specific meaning of ActRelationship and RoleLink as ontology properties. Similarly, for the local model of Figure 5.3 the "definition" (specialisation of ActRelationship) class becomes an ontology property and other elements are ontology classes.

Guideline 3 *An ontology data type property can be created from a UML model element, if the element is a class attribute and its type is a primitive data type, e.g., string, integer.*

For example, in Figure 5.1 the *classCode* attribute within the *Entity* class has a type CS class. The *classCode* is a candidate with others (*e.g.*, determinerCode, code, addr) for an ontology object property. Similarly, the *addr* attribute within the *Role* class has the type AD class. The AD class has attributes (*e.g.*, *city*) of type class (*e.g.*, adxp.city), which makes *city* an appropriate candidate as an object property. Figure 5.5 and Figure 5.6 show snippets of the UML *classCode* attribute as the ontology object property.

```
concept rim:Entity
    rim:classCode impliesType rim:CS
```

Figure 5.5: UML *attribute* and Ontology Property: WSML syntax

```
ObjectProperty: rim:classCode
    Domain: rim:Entity Range: rim:CS
```

Figure 5.6: UML *attribute* and Ontology Property: OWL syntax

Guideline 4 *An ontology object property can be created from a UML model element, if the element is an association.*

A UML *association* represents a relationship between classes. If a property is a part of the ends of an association and the type of the property is a class, then the property is mapped to *owl:ObjectProperty* or *wsml:attribute*. For example, in Figure 5.1, *playedRoleIn* is an association between the *Role* and the *Participation* classes. In this case, the "Domain" and "Range" will be the classes related to the association. Figure 5.7 and Figure 5.8 show a snippet for the *playedRoleIn* ObjectProperty.

```
concept rim:Role
    rim:playedRoleIn impliesType rim:Participation
```

Figure 5.7: UML *association* and Ontology *ObjectProperty*: WSML syntax

```
ObjectProperty: rim:playedRoleIn
    Domain: rim:Role Range: rim:Participation
```

Figure 5.8: UML *association* and Ontology *ObjectProperty*: OWL syntax

Inheritance: UML and OWL both support a generalisation relation. The objects of inheritance in UML are only classes and associations. However, OWL supports class inheritance and property inheritance separately.

Guideline 5 *An ontology inheritance relation can be created from a UML model element, if the element is an association or a class.*

A class inheritance in UML is mapped to the *SubClassOf* element of OWL, and UML *association* inheritance is mapped to the OWL *SubPropertyOf* element. For example, in Figure 5.4 the "ObservationOrder" of Figure 5.3 has a subclass relationship with the "Act" class (Figure 5.1).

Restrictions: UML constraints such as cardinalities and conditions can be represented using the restriction definition for ontology classes or properties. Ontology languages support several types of restrictions, but some of them are outside the scope of UML constructs, *e.g.*, OWL existential quantification on object properties, and defining new classes by restricting OWL datatype properties. In Figure 5.3 the *assignedPerson* association between ObservationOrder and R_assignedPerson classes has a cardinality of minimum 0 and unlimited maximal cardinality denoted by the asterisk symbol. Figure 5.9 and Figure 5.10 show the association restriction of *assignedPerson*. In the OWL Manchester syntax the absence of a maximal cardinality means unlimited occurrence. However, the WSML syntax uses the asterisk symbol for unlimited maximal cardinality.

```
concept rim:ObervationOrder
    assignedPerson impliesType (0 *) R_assignedPerson
```

Figure 5.9: UML cardinalities constraints and Ontology Properties: WSML syntax

```
DataProperty: assignedPerson
        Domain: rim:ObervationOrder
        Range: min 0
```

Figure 5.10: UML cardinalities constraints and Ontology Properties: OWL syntax

An ontology property can be globally declared as a functional or a inverse functional property. A functional property has a maximum cardinality of 1 on its range, while an inverse functional property has a maximum cardinality of 1 on its domain. Thus, if an attribute plays the role of an

identifier, it can be represented as a functional property. Figure 5.11 and Figure 5.12 show the data property *hasId* that defines each patient uniquely. In WSML syntax, a maximal cardinality of 1 makes an attribute/property functional.

```
concept rim:EntityPerson
    rim:hasId ofType (1) xsd:integer
```

Figure 5.11: Ontology functional Properties: WSML syntax

```
DataProperty rim:hasId Characteristics: Functional
  Domain: rim:EntityPerson Range: xsd:integer
```

Figure 5.12: Ontology functional Properties: OWL syntax

As discussed in the beginning of this chapter, even if there are similarities, some foundational differences between the UML and ontology languages exists. The next section summarises these gaps which hinder the smooth transformation of UML models to ontological knowledge bases and vice-versa.

5.2.3 Gaps Between UML and Ontology Languages

One of the major foundational difference between UML and ontology languages is the ontology notion relation, predicate, or property. An ontology property appears, at a first glance, to be the same as the UML *association* or *attribute*. However, properties in an ontology are first-class modelling elements, while the UML *association* or *attribute* is attached to UML classes where they are described. This means the UML *association* or *attribute* cannot exist in isolation or as a self-describing entity defining relationships such as inheritance. More precisely, in an ontology a relation can exist without specifying any classes to which it might relate. In UML, an association is defined in terms of association ends, which must be related to classes, and an attribute is always within the context of a single class.

A namespace or a URI is a basic element of an ontological knowledge base that provides entity identification and link other entities within or outside the knowledge base. UML uses a different kind of namespace mechanism to link its model elements to each other. Instead of URIs, UML uses namespaces to create self contained classes along with attributes and methods that vary depending on the scope of use. For example, each class has its own namespace for its attributes, associations and methods. This is further enriched by the ability to specify private, protected and public scopes. Ontology languages lack any kind of name scoping mechanism. As discuss above, another issue is the relation between UML attributes and ontology properties, which are first-class modelling elements.

UML	**WSML**	**OWL**
No Support	$(C_x..\text{or}..C_y)$	$(C_x..\text{or}..C_y)$
No Support	$(C_x..\text{and}..C_y)$	$(C_x..\text{and}..C_y)$
No Support	equivalent	EquivalentTo
No Support	transitive/symmetric/inverseOf	Transitive/Symmetric/InverseOf

Table 5.2: Some Basic Gaps Between UML and Ontology languages

Table 5.2 shows some differences on how classes are constructed in the UML and ontological models. For instance, ontology classes can be constructed using boolean operations (*union*, *intersection* and *complement*), while similar boolean constructs are missing in the UML model. In addition to boolean operations, a class definition based on property restrictions (*i.e.*, existential/universal

quantifiers) within ontological knowledge bases do not have UML equivalents. Also, ontology constructs such as *equivalentClass* or *equivalentProperty* is outside the scope of the UML model. These differences between UML and ontology languages have also been noted in [124, 125, 126]. Furthermore, the closed-world assumption of UML and an open-world ontological knowledge base is an open research problem within the Knowledge Representation (KR) community [127].

We mentioned in the previous chapter that our automatic approach of lifting HL7 resources follows Route 2 of transforming HL7's XML(S) *coreSchemas* and local message schemas. The five guidelines mentioned above and correspondences expressed in the Table 5.1 provide understanding of elements and attributes used in *coreSchemas* and local message schemas, which are exported from the RMIM designer. Next we present XSD to ontology conversion mechanism.

5.3 XML(S) To Ontology

Several languages have been developed for the mutual conversion between XML to other data formats such as plain text, HTML, and relational databases [128, 129]. Few of them [130, 131] specifically focus on the bi-directional transformation between XML and RDF data formats. biXid [132] is a language proposed for the bidirectional transformation between different XML data formats. XSugar [128] is another language specifically designed for bidirectional conversion between XML and text formats, and emphasising the reversibility of data transformation. There are many attempts to make XML metadata semantics explicit by translating XML constructs to Semantic Web languages and some of them just model the XML tree using the RDF primitives [133]. Others concentrate on modelling the knowledge implicit in XML languages definitions, *i.e.*, DTDs or the XML Schemas, using OWL [134, 135]. There are also attempts to provide unified semantics which brings XML and RDF semantics together [136].

[137] proposed a method for schemata transformation from UML to XML schema. Initiatives like [138, 139] mainly focus on the problem of XML to ontology transformation. While the approach presented in [139] is based on concept mappings defined at the schema level, the approach [138] is based on mappings defined at the data level. [140] proposes an automatic mapping from XML contents to RDFS by using an ontology, which is created from the corresponding XML Schema. This ontology contains the XML Schema information, but has no instances. The XML data will not be mapped to its ontology equivalents. This approach further has been used to generate (X)HTML Web pages with RDF annotations with regard to the constructs defined in the ontology.

[141] describes a direct mapping between XML and an RDF model, without passing through a special serialisation like RDF/XML. The authors assume that an XML Schema is available which guides the mapping process. Similarly, the authors of [142] describe mappings from XML to RDF as well as from XML Schema to OWL, but these mapping are independent of each other. That means, OWL instances do not necessarily match the OWL model, because elements in an XML document may have been mapped to different elements in the OWL model.

[131] proposed a query-based mechanism, called XSPARQL, to transform XML and RDF data instances. The mechanism provides a unified framework that transforms data (XML and RDF) and query languages (XQuery [143] and SPARQL [55]). This query-based mechanism could be a valuable

input to the scenarios of transforming messages instances. However, it lacks the ability to transform XSD definitions.

We notice above that none of the approaches offer a concrete mechanim or tool-support for XSD to ontology conversions. Furthermore, HL7 specific schema elements and attributes need domain specific conversion rules to preserve the semantics within corresponding ontological models. We introduce HL7 specific conversion rules reducing the information loss between standard schemas and ontological format. Next we describe the two types of HL7 specifications (*coreSchemas* and Message Schemas) and how they are transformed to global and local ontologies.

5.3.1 HL7 XML(S) Specifications

An automatic mechanism proposed in this thesis transforms a XML Schema as well as message instances (bi-directionally) to ontologies. The transformation rules are implemented using the XSL Transformations (XSLT) language. The five guidelines defined in Section 5.2.2 for transforming UML to ontology languages are used as an input to use HL7 specific semantics of XSD definitions published by the standard. The *coreSchemas* are transformed first to create global ontologies and message schemas are input for constructing local ontologies. We provide XSD to ontology transformation for the OWL and WSML ontology languages.

HL7 *coreSchemas*

HL7 Version 3 *coreSchemas* are available in three categories: (i) Foundational Data Type (ii) Basic Data Type and (iii) Vocabulary. Similarly, the Version 2 *coreSchemas* are also arranged in three categories: (i) Data Type (ii) Fields and (iii) Segments. Accordingly, Version 3 data types are divided into two groups (Foundational and Basic), where foundational data types are designed to construct the other data types. Basic data types are built by extending foundational data types and one used to create *complexTypes* for clinical message construction. The semantics of Version 3 are expressed using UML notions and any concrete implementation of the HL7 standard uses these built-in data types for their implementation technology. With the UML semantic specification, an Implementable Technology Specification (ITS) provides a mapping between the constructs of implementation technology and the HL7 Version 3 data type semantics. In Version 3, the separation of semantics from implementation technology aids in the consistent use of data types and vocabularies, irrespective of the implementation technology. Listing 5.1 shows the foundational data type SXCM_CD, which is a container of set components that specifies whether the set component is included (union) or excluded (set-difference) from the set, or other set operations with the current set component. We observe in Listing 5.1 that *complexType* SXCM_CD has an extension (specialisation) relation with the data type CD and the data type purpose is described using the annotation element.

```
1 <xs:complexType name="SXCM_CD">
2     <xs:complexContent>
3       <xs:extension base="CD">
4         <xs:attribute name="operator" type="SetOperator" use="optional" default="I">
5           <xs:annotation>
6             <xs:documentation>
7                     A code specifying the set component as constructed from the representation stream.
8                   </xs:documentation>
```

```
9            </xs:annotation>
10         </xs:attribute>
11       </xs:extension>
12     </xs:complexContent>
13 </xs:complexType>
```

Listing 5.1: HL7 Version 3 *coreSchema*: Data type (SXCM_CD)

Listing 5.2 shows a snippet of the vocabulary schema definition for the "ActClassObservation" class. In Line 5 we notice that the attribute *union@memberTypes* defines three classes ("ActClassCondition", "ActClassObservationSeries", "ActClassROI"). According to the UML semantics of the vocabulary definition, these three classes are specialisations of the "ActClassObservation" class. HL7 XSD Implementable Technology Specification (ITS) used the *memberTypes* attribute to represent multiple inheritance. The *enumeration* values of "ActClassObservation" are all allowed values that "ActClassObservation" class should contain. This special way of using *memberTypes* to represent multiple inheritance requires special lifting rules to build the Version 3 vocabulary ontology.

```
1  <xs:simpleType name="ActClassObservation">
2        <xs:annotation>
3           <xs:documentation>specDomain: S11529 (C−0−T11527−S13856−S11529−cpt)</xs:documentation>
4        </xs:annotation>
5        <xs:union memberTypes="ActClassCondition ActClassObservationSeries ActClassROI">
6           <xs:simpleType>
7              <xs:restriction base="cs">
8                 <xs:enumeration value="OBS"/>
9                 <xs:enumeration value="CNOD"/>
10                <xs:enumeration value="CLNTRL"/>
11                <xs:enumeration value="ALRT"/>
12                <xs:enumeration value="DGIMG"/>
13                <xs:enumeration value="INVSTG"/>
14                <xs:enumeration value="SPCOBS"/>
15             </xs:restriction>
16          </xs:simpleType>
17       </xs:union>
18 </xs:simpleType>
```

Listing 5.2: HL7 Version 3 *coreSchema*: Vocabulary (ActClassObservation)

HL7 Version 2 *coreSchemas* are designed entirely using XSD specifications. Listing 5.3 shows the XSD definition for the field "MSH.3.CONTENT" of the segment "MSH.CONTENT" shown in Listing 5.4. "MSH.3.CONTENT" describes the name of a "Sending Application", which initiates any clinical order. Similar to the Version 3 data type and vocabulary, the Version 2 schema uses *xsd:extension* to describe specialisation and annotations are described through *xsd:documentation* and *hl7:LongName* elements. The annotation elements such as *hl7:LongName*, *hl7:Type*, *hl7:Table*, and *hl7:Item* is specific to HL7 applications and requires a special rule to lift to the corresponding ontological element.

```
1  <xsd:complexType name="MSH.3.CONTENT">
2     <xsd:annotation>
3        <xsd:documentation xml:lang="en">Sending Application</xsd:documentation>
4        <xsd:appinfo>
5           <hl7:Item>3</hl7:Item>
6           <hl7:Type>HD</hl7:Type>
7           <hl7:Table>HL70361</hl7:Table>
8           <hl7:LongName>Sending Application</hl7:LongName>
9        </xsd:appinfo>
10    </xsd:annotation>
```

```
11      <xsd:complexContent>
12        <xsd:extension base="HD">
13          <xsd:attributeGroup ref="MSH.3.ATTRIBUTES"/>
14        </xsd:extension>
15      </xsd:complexContent>
16  </xsd:complexType>
```

Listing 5.3: HL7 Version 2 *coreSchema*: Fields (MSH.3.CONTENT)

Listing 5.4 shows an XSD snippet for "MSH.CONTENT" and it is important to notice that the end of *sequence* (Line 8) defines *xsd:any processContents="lax"*, which means, healthcare institutes can extend the XML document with elements not specified by the standard schema and, if a schema cannot be obtained, no errors will occur. This flexility allowed by the Version 2 specification is a significant interoperability hurdle between Version 2 applications as well as between Version 2 and Version 3 applications.

```
1  <xsd:complexType name="MSH.CONTENT">
2      <xsd:sequence>
3        <xsd:element name="MSH.1" minOccurs="1" maxOccurs="1" type="MSH.1.CONTENT" />
4        <xsd:element name="MSH.2" minOccurs="1" maxOccurs="1" type="MSH.2.CONTENT" />
5        <xsd:element name="MSH.3" minOccurs="1" maxOccurs="1" type="MSH.3.CONTENT" />
6        .....
7        <xsd:element name="MSH.20" minOccurs="0" maxOccurs="1" type="MSH.20.CONTENT" />
8        <xsd:any processContents="lax" namespace="##other" minOccurs="0"/>
9      </xsd:sequence>
10 </xsd:complexType>
```

Listing 5.4: HL7 Version 2 *coreSchema*: Segments (MSH.CONTENT)

The *coreSchemas* are designed and maintained by the HL7 standard body, and message schemas are generated by local applications. HL7 publishes guideline schemas for constructing local messages. However, local applications usually divert from the guidelines to meet the local messaging requirements.

HL7 Message Schemas

Listing 5.5 shows a snippet of an observation order schema (Version 3) mentioning patient details. The observation order schema consists of elements and constraints describing a patient. Each element (id, name, administrativeGenderCode, birthTime, and addr) has an associated data type (Instance Identifier (II), Address Datatype (AD), etc.) described in Basic and Foundational data types schemas. An XSD schema for a HL7 message normally contains 90 *complexTypes* and 50 *elements* or *attributes*.

```
1  <xs:complexType name="Patient">
2   <xs:sequence>
3    <xs:element maxOccurs="unbounded" minOccurs="1" name="id" type="II"/>
4    <xs:element maxOccurs="1" minOccurs="1" name="name" type="EN"/>
5    <xs:element maxOccurs="1" minOccurs="1" name="administrativeGenderCode" type="CE"/>
6    <xs:element maxOccurs="1" minOccurs="1" name="birthTime" type="TS"/>
7    <xs:element maxOccurs="unbounded" minOccurs="1" name="addr" type="AD"/>
8    .....
9   </xs:sequence>
10   <xs:attribute fixed="PSN" name="classCode" type="EntityClass" use="optional"/>
11 </xs:complexType>
```

Listing 5.5: HL7 Version 3 Message Schema: Patient details

It is important to notice that the attribute *classCode* (Line 10 of Listing 5.5) has a type "EntityClass" and a fixed value PSN is allocated. In Version 3 messages, the type of *classCode* describes the inheritance relationship, which means, "Patient" is a specialisation of the "Entityclass" described in the Vocabulary schema. This requires a special lifting rule using the *classCode* for the *subclass* definition. Similarly a coded value PSN (in Listing 5.5) is attached to indicate that a person "Patient" is a "LivingSubject". The attribute *classCode* establishes a relation between the *coreschemas* and message schemas, but the definition of it is "optional". That means health care institutes are allowed to create local *complexTypes* within messages without inheriting from the *coreschemas*. Such optionality to construct local elements creates an interoperability hurdle between Version 3 applications.

```
1 <xsd:complexType name="ORU_R01.CONTENT">
2     <xsd:sequence>
3       <xsd:element name="MSH" minOccurs="1" maxOccurs="1" type="MSH.CONTENT" />
4       <xsd:element name="PID" minOccurs="1" maxOccurs="1" type="PID.CONTENT"/>
5       <xsd:element name="OBR" minOccurs="0" maxOccurs="1" type="OBR.CONTENT"/>
6       <xsd:element name="OBX" minOccurs="1" maxOccurs="1" type="OBX.CONTENT"/>
7      </xsd:sequence>
8 </xsd:complexType>
```

Listing 5.6: HL7 Version 2 Message Schema: Observation Request (ORU_R01.CONTENT)

Listing 5.6 shows a schema snippet of the Version 2 ORU_R01 (Observation Request) message that consists of segments: MSH, PID, OBR, and OBX. Segment MSH represents the actual message, PID for a patient identification, PV for a patient verification, OBR for an observation request, and OBX for the actual lab observation. The schema definition for Version 2 messages is straightforward when compared to the Version 3 schema (Version 3 uses special elements and attributes to describe UML notions like generalisation or specialisation). We need transformation rules that can convert both core and message schemas, preserving the HL7 specific semantics of elements and attributes like *classCode*, *membertypes*, etc.

5.3.2 HL7 Schema to Ontology: Rules

We developed two sets of transformation rules for WSML and OWL ontology languages. Table 5.3 presents transformation rules between XML Schema↔OWL↔WSML. In Table 5.3, italicised schema elements are HL7 specific (Rules 5 and 7), while other rules can be applied to general XML Schema↔OWL↔WSML conversions.

Listing 5.7 describes rules for creating correspondences between *xsd:element* and *xsd:attribute*. Listing 5.7 also presents rules for establishing domain, range and cardinality restrictions for ontological properties. We notice in the Line 2 that, a matching pattern is created to extract all the names of XSD elements and attributes. The matching pattern uses a variable *ConceptName* containing name of the parent element (*i.e.*, *xsd:complexType* or *xsd:simpleType* or *xsd:attributeGroup*). Elements and attributes returned by the matching pattern may have complex or primitive types (*e.g.*, string, integer). Line 4 creates a property name and Line 8 checks whether PropertyName itself it not a

Rule	XML Schema	WSML	OWL
1	element|attribute	attribute	ObjectProperty|DataProperty
2	element@substitutionGroup	subAttributeOf	SubPropertyOf
3	element@type	Range	Range
4	complexType|group|attributeGroup	concept	Class
5	5.1 *extension@base—restriction@base* 5.2 *union@memberTypes* 5.3 *attribute@classCode*	subConceptOf	SubClassOf
6	6.1 @maxOccurs 6.2 @minOccurs	maxCard.|minCard.	max|min
7	7.1 *Annotation@appinfo* 7.2 *hl7:LongName|hl7:Type*	annotations	Annotations@label|comment

Table 5.3: XML Schema↔OWL↔WSML transformation rules

type name (or a property range). Line 8 also ensures that the element or attribute is not an XSD URI. If the type is an XSD URI element then attribute is considered as a data property. Otherwise, PropertyName is an object property.

```
1
2  <xsl:for-each select="//xsd:element[(@name=(//xsd:element[ancestor::xsd:element[@name =$ConceptName]]))] and //xsd:attribute[
       @name and (ancestor::xsd:complexType[@name = $ConceptName] or ancestor::xsd:attributeGroup[@name = $ConceptName] or
       ancestor::xsd:simpleType[@name = $ConceptName])"
3
4  <xsl:variable name="PropertyName">
5    <xsl:value-of select="@name"/>
6  </xsl:variable>
7
8  <xsl:if test="(@name and @type ) or (//xsd:element[@name=$PropertyName and not(@type)] and not(XSDUri(@type, namespace::*))
9
10 <xsl:value-of select="$PropertyName"/>
11
12 impliesType
13
14 <xsl:if test="(@maxOccurs and @minOccurs)">
15     <xsl:value-of select="@minOccurs"/>
16   <xsl:value-of select="if(@maxOccurs='unbounded') then '*'
17  else @maxOccurs"/>
18
19 <xsl:when test="@name and @type and not(xo:XSDUri(@type, namespace::*))">
20     <xsl:value-of select="@type"/>
21 </xsl:when>
```

Listing 5.7: Rule 1, 3 and 6 (Table 5.3): WSML attribute, Range, Maximum and Minimum Cardinality

In Line 12 *impliesType* is a WSML keyword that defines the relationship between domain and range of a WSML attribute. Line 14 checks a cardinality restriction (*i.e.*, maxOccurs and minOccurs); and finally Line 19 decides the range of the property names based on whether the range is an XSD URI or a *complexType*. Listing 5.8 is an OWL equivalent rule for creating object and data type properties. In OWL syntax, domain and range keywords are explicitly defined, but for WSML positioning of name and type with *impliesType* describes the domain and range of an attribute.

```
1  <xsl:for-each select="//xsd:element[(@name=(//xsd:element[ancestor::xsd:element[@name =$ConceptName]])) and //xsd:attribute[
       @name and (ancestor::xsd:complexType[@name = $ConceptName] or ancestor::xsd:attributeGroup[@name = $ConceptName] or
       ancestor::xsd:simpleType[@name = $ConceptName])]"
2
3  <xsl:variable name="PropertyName">
4    <xsl:value-of select="@name"/>
5  </xsl:variable>
```

```
6
7 <xsl:if test="(@name and @type ) or (//xsd:element[@name=$PropertyName and not(@type)] and not(XSDUri(@type, namespace::*))
8
9 ObjectProperty: <xsl:value-of select="$PropertyName"/>
10
11 Domain: <xsl:value-of select="$ConceptName"/>
12 Range: <xsl:when test="@name and @type and not(xo:isXsdUri(@type, namespace::*))">
13            <xsl:if test="(@maxOccurs and @minOccurs)">
14          <xsl:value-of select="$PropertyName"/> min <xsl:value-of select="@minOccurs"/>
15              <xsl:value-of select="@type"/>
16          <xsl:if test="(@maxOccurs!='unbounded')">
17               <xsl:value-of select="$PropertyName"/> max <xsl:value-of select="@maxOccurs"/> <xsl:value-of select="@type"
                    />
18     </xsl:if>
19    </xsl:if>
20 </xsl:when>
```

Listing 5.8: Rule 1, 3 and 6 (Table 5.3): OWL ObjectProperty/DataProperty, Range, Maximum and Minimum Cardinality

Listing 5.9 and Listing 5.10 show the Rule 2 transforming *xsd:substitutionGroup* to subAttributeOf/SubPropertyOf relation. A substitution group allows to build a collection of elements that can be specified using a generic element. Since our Rule 1 transforms XML element to an ontology property, substitution group is used to create property inheritance.

```
1 <xsl:template name="Type" match="xsd:complexType|xsd:substitutionGroup">
2     <xsl:if test="@name">
3      <xsl:variable name="$AttributeName">
4       <xsl:value-of select="@name"/>
5        </xsl:variable>
6
7 subAttributeOf <xsl:value-of select="@substitutionGroup"/>
```

Listing 5.9: Rule 2 (Table 5.3): WSML subAttributeOf

```
1 <xsl:template name="Type" match="xsd:complexType|xsd:substitutionGroup">
2     <xsl:if test="@name">
3      <xsl:variable name="$PropertyName">
4       <xsl:value-of select="@name"/>
5        </xsl:variable>
6
7 SubPropertyOf: <xsl:value-of select="@substitutionGroup"/>
```

Listing 5.10: Rule 2 (Table 5.3): OWL SubPropertyOf

Listing 5.11 presents rules for creating an ontological concept/class from an *xsd:complexType*, *xsd:simpleType*, or *xsd:attributeGroup*. Line 1 creates a matching pattern and Line 7 associates the WSML *concept* keyword with a concept name returned by the matching pattern. Listing 5.12 is an OWL equivalent rule for creating ontology classes.

```
1 <xsl:template name="Type" match="xsd:complexType|xsd:simpleType|xsd:attributeGroup">
2     <xsl:if test="@name">
3      <xsl:variable name="$ConceptName">
4       <xsl:value-of select="@name"/>
5        </xsl:variable>
6
7 concept <xsl:value-of select="$ConceptName"/>
```

Listing 5.11: Rule 4 (Table 5.3): WSML concept

Listing 5.13 presents the rule for creating an ontological inheritance (*i.e.*, subconcept/subclass relation). Line 1 of Listing 5.13 is a matching pattern for extracting the *base* name of *xsd:extension* and *xsd:restriction* elements.

```
1 <xsl:template name="Type" match="xsd:complexType|xsd:simpleType|xsd:attributeGroup">
2     <xsl:if test="@name">
3      <xsl:variable name="ConceptName">
4       <xsl:value-of select="@name"/>
5        </xsl:variable>
6
7 Class: <xsl:value-of select="$ConceptName"/>
```

Listing 5.12: Rule 4 (Table 5.3): OWL Class

Line 4 checks whether an extension base carries an XSD URI or another schema element. Similarly, Listing 5.14 shows the OWL equivalent of creating a subclass relationship. In addition to the standard *xsd:extension* or *xsd:restriction* base, Listing 5.15 shows the matching pattern for extracting the value of the *union@memberTypes*.

```
1 <xsl:for-each select="xsd:complexContent/xsd:extension[@base and parent::xsd:complexContent] | xsd:complexContent/xsd:restriction[
      @base and parent::xsd:complexContent] | xsd:restriction[@base and parent::xsd:simpleType]| xsd:simpleContent/xsd:extension[@base
       and parent::xsd:simpleContent] | xsd:simpleContent/xsd:restriction[@base and parent::xsd:simpleContent]"
2
3 SubConceptOf
4 <xsl:if test="@base and not(XSDUri(@base, namespace::*))">
5     <xsl:value-of select="@base"/>
6 </xsl:if>
```

Listing 5.13: Rule 5.1 (Table 5.3): WSML subconcept

```
1 <xsl:for-each select="xsd:complexContent/xsd:extension[@base and parent::xsd:complexContent]| xsd:complexContent/xsd:restriction[
      @base and parent::xsd:complexContent]|xsd:restriction[@base and parent::xsd:simpleType]| xsd:simpleContent/xsd:extension[@base
      and parent::xsd:simpleContent]| xsd:simpleContent/xsd:restriction[@base and parent::xsd:simpleContent]"
2
3 SubClassOf:
4 <xsl:if test="@base and not(XSDUri(@base, namespace::*))">
5     <xsl:value-of select="@base"/>
6 </xsl:if>
```

Listing 5.14: Rule 5.1 (Table 5.3): OWL SubClassOf

Line 1 of Listing 5.15 is a matching pattern for extracting the *union@memberTypes* value for the *xsd:simpleType*. Line 4 splits the *union@memberTypes* value string into individual tokens and creates a subclass relationship. For example, we notice in Listing 5.2 of the HL7 Version 3 vocabulary (Line 5) that memberTypes consists of three substrings and each substring is a subclass name for the class "ActClassObservation".

```
1 <xsl:for-each select="xsd:union[@memberTypes and parent::xsd:simpleType]| xsd:simpleContent/xsd:union[@memberTypes and parent::
      xsd:simpleContent"]
2
3 <xsl:if test="@memberTypes">
4     <xsl:for-each select="tokenize(@memberTypes, '\s')">
5   concept <xsl:value-of select="." />
6   subConceptOf <xsl:value-of select="$ConceptName"/>
7 </xsl:for-each>
```

Listing 5.15: Rule 5.2 (Table 5.3): WSML subConceptOf (memberType)

Listing 5.16 shows the OWL equivalent for the *union@memberTypes* subclass relationship. This *union@memberTypes* subclass relation is a HL7 specific mechanism for representing multiple inheritance.

```
1 <xsl:for-each select="xsd:union[@memberTypes and parent::xsd:simpleType]| xsd:simpleContent/xsd:union[@memberTypes and parent::
      xsd:simpleContent"]
2
3 <xsl:if test="@memberTypes">
4     <xsl:for-each select="tokenize(@memberTypes, '\s')">
5   Class: <xsl:value-of select="." />
6   SubClassOf: <xsl:value-of select="$ConceptName"/>
7 </xsl:for-each>
```

Listing 5.16: Rule 5.2 (Table 5.3): OWL SubClassOf (memberType)

Listing 5.17 shows the subclass relation using the *classCode* attribute of a message schema (see Listing 5.5). The subclass relation creates inheritance between local messaging elements and global *coreSchemas*.

```
1 <xsl:for-each select="xsd:attribute[@name="classCode"]
2
3 <xsl:if test="@name='classCode'">
4       subConceptOf <xsl:value-of select="@type"/>
5 </xsl:for-each>
```

Listing 5.17: Rule 5.3 (Table 5.3): WSML (classCode)

Listing 5.18 is an OWL equivalent rule for the *classCode*-based subclass relation. As said before, the use of *classCode* attribute is optional and local elements without a *classCode* attribute become local ontological concepts without holding any reference to the global ontology. Such local concepts require alignments between local concepts from other healthcare institutions exchanging and sharing patient clinical records.

```
1 <xsl:for-each select="xsd:attribute[@name="classCode"]
2
3 <xsl:if test="@name='classCode'">
4       SubClassOf: <xsl:value-of select="@type"/>
5 </xsl:for-each>
```

Listing 5.18: Rule 5.3 (Table 5.3): OWL SubClassOf (classCode)

In WSML annotations are described as "nonFunctionalProperties" and can be represented using annotation properties such as Dublin Core[5] (*dc:description*), *rdfs:label*, or *rdfs:comment*.

```
1
2 <xsl:variable name="annotation">
3     <xsl:value-of select="xsd:annotation/xsd:documentation"/>
4 </xsl:variable>
5
6 <xsl:variable name="hl7longname">
7     <xsl:value-of select="xsd:appinfo/LongName"/>
8 </xsl:variable>
9
10 nonFunctionalProperties
11 dc#description hasValue "
12 <xsl:if test="($annotation)">
13     <xsl:value-of select="$annotation)"/>
```

[5] http://dublincore.org/

```
14 </xsl:if>
15
16 <xsl:value-of select="$hl7longname"/>
17 endNonFunctionalProperties
```

Listing 5.19: Rule 7 (Table 5.3): WSML annotations

Line 3 of Listing 5.19 extracts the annotation value of *xsd:documentation* and Line 7 extracts the HL7 specific annotation element *LongName*. Listing 5.20 shows the OWL equivalent for the annotation rule.

```
1 <xsl:variable name="annotation">
2     <xsl:value-of select="xsd:annotation/xsd:documentation"/>
3 </xsl:variable>
4
5 <xsl:variable name="hl7longname">
6     <xsl:value-of select="xsd:appinfo/LongName"/>
7 </xsl:variable>
8
9 Annotations:
10 rdfs:label <xsl:if test="($annotation)">
11         <xsl:value-of select="$annotation"/>
12             </xsl:if>
13             <xsl:value-of select="$hl7longname"/>
```

Listing 5.20: Rule 7 (Table 5.3): OWL Annotation

XSLT templates created for XML Schema↔OWL↔WSML conversions are applied separately on each schema (vocabulary, datatype, messages, etc.) and *xsd:include* and *xsd:import* elements, analogous to ontology imports, are used to link ontologies created. The lifting of *coreSchemas* to global ontologies has to be performed once, until HL7 makes amendments in the standard specification. However, lifting of message schemas is required at each hospital location.

5.3.3 Gaps Between XML(S) and Ontology Languages

XML Schema shares similar differences as UML does with ontology languages.

	XML Schema	**Ontology**
Data Type	(i) Supports large number of data types	(i) RDFS has limited support, thanks to OWL/WSML for extended data types support
Structure	(i) Nested data structure (ii) Tree structure (top element is root) (iii) Sequence to describe element order	(i) Concept composition is through properties (ii) Graph based (Any concept could be root) (iii) No ordering of concepts
Relation	(i) Inheritance through Type and Extension (ii) No Support	(i) Multiple Inheritance (ii) Inheritance on properties and logical implications (symmetric, Transitive, etc.)

Table 5.4: XML Schema and Ontology differences

Several efforts for the meaningful transformation of XML(S) to ontology languages have revealed their semantic differences [144, 145, 146, 133]. XML Schema provides rich syntax and structure definitions for data modelling, while an ontology allows more sophisticated semantic modelling of a particular domain. [145] has given a more detailed analysis on the differences between the Ontology Inference Layer (OIL) and XML Schema. [145] also provides a detailed analysis of problems that may be encountered during XML Schema and ontology transformation. Table 5.4 shows the main differences between XML Schema and ontology languages.

These differences might lead to information loss during the transformation from an XML document to an ontology instance or vice versa. Thus they become a major obstacle for an ontology mediated XML data exchange and still an open research problem. In our case we developed HL7 specific rules to preserve the information loss.

5.4 Evaluation

Table 5.5 shows a comparison metrics for HL7 ontologies developed by Bhavana Orgun (see Section 4.3), Helen Chan (see Section 4.3) and the PPEPR Methodology (*i.e.*, using lifting rules of the Table 5.3). The metrics presented in Table 5.5 shows a coverage *i.e.*, extent to which these ontologies cover the HL7 specifications. Ontology metrics is presented in terms of a number of classes, object properties, datatype properties, and individuals.

	Metrics	**HL7 V3**	**HL7 V2**	**Method**
Global Ontology (Bhavana Orgun)	(i) Class (ii) Property	(i) 881 (ii) 82		manual
Global Ontology (Helen Chan)	(i) Class (ii) Object Property (iii) Datatype Property (iv) Individual	(i) 292 (ii) 175 (iii) 50 (iv) 38		manual
Global Ontology (PPEPR)	(i) Class (ii) Object Property (iii) Datatype Property (iv) Individual	(i) 1563 (ii) 200 (iii) 100 (iv) 6582	(i) 1062 (ii) 682 (iii) 50	automatic
Local Ontology (PPEPR)	(i) Class (ii) Object Property (iii) Datatype Property	(i) 30-50 (ii) 33-43 (iii) 10-12	(i) 20-30 (ii) 15-20 (iii) 07-10	automatic

Table 5.5: Evaluating coverage of HL7 ontologies

In Table 5.5 the number of classes for global ontologies indicates the sum of classes within the three ontologies (Foundational Data Type, Basic Data type, and Vocabulary) developed from the HL7 *coreSchemas*. Similarly, object properties, datatype properties, and individuals are summed to

show the complete size of the global ontologies. Individuals within the Version 3 global ontologies are medical codes allowed for a particular class. For example, in Listing 5.2, the vocabulary schema contains code like OBS, CNOD, CLNTRL, etc. which are allowed values for the "ActClassObservation" class. In addition to the Version 3 defined codes, several other medical terminologies like ICD-10, LOINC, etc. are also used as allowed values for Version 3 classes. We treat such codes as ontological individuals.

We see a notable difference in the size of ontologies developed by Bhavana Orgun (see Section 4.3), Helen Chan (see Section 4.3) and the PPEPR Methodology. For example, Orgun's ontologies covers some portions of HL7 specifications (with 881 classes and 82 properties for the Version 3) in the RDFS language, whereas Chan improves the Orgun's ontology in terms of a better design by including individuals (*i.e.*, medical codes), datatype properties, and improving the number of properties. Chan's ontology is developed using the expressive OWL constructs such as functional property, constraints, etc. comparing to Orgun's RDFS based ontology.

There is a significant improvement in the coverage (size) of HL7 ontologies developed by the PPEPR methodology, *i.e.*, 1563 classes, 200 object properties, 100 datatype properties, and 6582 individuals for the Version 3. We cover the entire HL7 specifications for both versions (*i.e.*, Version 2 and Version 3). In addition, PPEPR Methodology introduces a local ontology developed from an application deployed in a specific environment. In the case of local ontologies, the numbers show the range because the actual number of ontological elements depends on local choices made in constructing a clinical message. Based on our experiences we show the minimum and maximum numbers of classes or properties used for creating such local ontologies. We observe that the maximum size for a Version 3 local ontology could be 30 classes, 33 object properties, and 10 datatype properties. Which means, 73 local correspondences (or alignments) are required to exchange a clinical message. This significantly improves the current situation where 140 (90 *complexTypes* and 50 *elements* or *attributes*) local correspondences are required to exchange a clinical message. It is important to notice that global ontologies for Version 2 do not contain individuals. Version 2 schemas primarily use codes from other medical vocabularies (ICD-10, LOINC) which creates a common upper layer from ontologies outside the HL7 system.

The greater coverage and an automatic method (comparing Orgun's and Chan's works) to build HL7 ontologies contributes to our first research challenge (*i.e.*, "how to build HL7 ontologies by reusing domain artefacts reside in separate global and local spaces"). Considering the size of the HL7 ontologies (developed by PPEPR), it is also evident that the Version 3 system is richer in terms of classes, properties and individuals. Such richness comes from well defined models using UML semantics.

5.5 Summary

The chapter describes issues in converting UML models and XSD specifications to ontological formats. We discussed that direct conversion between these languages is not a straightforward task and differences in semantics between the language constructs is the root cause in hindering automatic conversions. The chapter describes the related work in these areas and highlights some of the advantages and disadvantages of each approach. The chapter presents five guidelines to establish correspondences between HL7 UML constructs and ontological elements. The guidelines defined are used as an input to describe rules for the automatic conversion of XSD specifications to ontologies. The rules are designed and implemented using XSLT templates and two sets of templates are created for WSML and OWL ontologies. Finally, the chapter presents an evaluation of HL7 ontologies based on their converge of the domain and methods to develop them.

Chapter 6

Aligning HL7 Ontologies

6.1 Introduction

In the previous chapters we presented our approach in building the HL7 ontologies. We discussed that, even if healthcare institutes follow a particular standard, differences in their data, schema, or information models are inevitable because of local choices. Additionally, complexity multiplies for ontological frameworks, if interoperability is expected between two or more different standards. In recent times, one of the basic problems in the development of techniques for the Semantic Web is the integration of ontologies. In heterogenous and vast domains, as presented by the Healthcare and Life Sciences (HCLS) domain, different parties would, in general, adopt different ontologies. Thus, just using ontologies, like just using XML, does not reduce heterogeneity: it raises heterogeneity problems to a higher level. Analogous to schema matching and mapping techniques, ontology matching methods have been proposed to deal with the ontological heterogeneity.

Ontology matching consists of finding the correspondences (*e.g.*, equivalence, subsumption) between elements of ontologies (*e.g.*, concepts, properties). Usually, ontology matching task is separated into: (i) a (semi-) automated support for discovering ontological correspondences; and (ii) an alignment representing the discovered correspondences. Ontology matching is an important task because it helps restoring interoperability, but a difficult one because it is very hard to find correspondences in many cases: (i) independently built ontologies can vary a lot on the terminology they use and the way they model the same entities; and (ii) domain ontologies focus on a particular domain and use terms in a sense that is relevant to this domain and which is not necessarily related to similar concepts in other domains. Most research in the ontology matching area has focused on (semi-)automatic matching of ontologies using approaches based on combinations of syntactic similarity, graph similarity, constraint checks, and data analysis. Generally, ontology matching methods perform linguistic analysis to obtain a preliminary set of correspondences between the source and target ontologies. Results from linguistic analysis are used as a starting point by other analysis methods for further processing. There are many different approaches for linguistic analysis. The simplest method is to calculate a string similarity between the two elements. Strings are assigned an edit value corresponding to the number of operations to transform it from one string into the other (also known as Edit Distance [147]).

Structural matching of elements can be performed based on the similarity of their data structures, context, adjacent elements, and other structural facets. Structural analysis assumes that if two elements in different ontological models are found to be similar, the structure of the model can provide insights or hints as to which other elements have a high degree of correlation. In cases where two similar concepts have very little or no string similarity, the analysis of their placement within the structure of the ontology is often the only method to correctly align the two concepts to each other. Analysis methods can vary significantly due to placing more or less emphasis on a variety of structural attributes.

In addition to linguistic and structural matchings, a third approach is to use a higher level ontology as a common layer for two ontologies to be matched. This can typically be an upper-level ontology (*e.g.*, CYC [148] or Suggested Upper Merged Ontology (SUMO) [149]) that is used as external source of common knowledge. Matching each of the ontologies to a common third ontology is an indirect alignment between the two ontologies to be matched. This is easier than the direct matching because: (i) upper level ontologies having a broad coverage are more prone to provide matching super concepts of these ontologies, (ii) there are more chances that these alignments already exist due to the availability of these ontologies. Once the alignments between two ontologies and the upper-level one are obtained, it is easier to partition the ontology matching task into smaller matching problems because there is a high probability that two concepts might have a common super-concept.

The (semi-)automated approaches of linguistic and structural matchings typically give an incomplete or incorrect set of correspondences between terms. A human must align the remaining terms and check the machine built alignments to truly complete the matching process. Although fully automated solutions may be infeasible, there are tools and algorithms that, when combined with human assistance, can greatly aid the matching of large ontologies for which manual matching is impractical. These techniques contain intermediate steps where humans can intervene to manipulate results, parameters, and other data critical to the matching process. Our work places an emphasis on exploiting these steps to provide valuable insight to the matching process and improve accuracy. Meaningful adjustments performed iteratively over the matching process allows a human user to converge on a significantly more accurate alignment.

In our case, we have discussed in the Chapter 3 two important HL7-specific features (*i.e.*, Structural Similarity and Annotation-based Similarity) to determine the correspondences between two matching concepts. We want to investigate and apply automated ontology matching tools for aligning HL7 Version 2 and Version 3 ontologies created in the previous chapter. In this chapter, first, we briefly describe typical ontology matching approaches. Second, we describe the state of the art in ontology matching tools and summarise matching algorithms proposed by the each tool. Third, we analyse the state of the art in ontology matching tools and their respective matching results. Fourth, we introduce a query based ontology matching method that improves the recall and accuracy of the overall matching results. Finally, we evaluate the state of the art in ontology matching tools and the proposed query-based method for aligning the HL7 ontologies.

6.2 Ontology Matching

[150, 151] describe ontology matching as being motivated by four broad challenges. First, the increasing volume of data being dealt with in the typical electronic transaction suggests that it would be impractical in most cases to produce mappings between ontologies manually. Fully or semi-automated means of identifying correspondences and consequent mappings should be developed to facilitate this process. Second, the heterogeneity of information in the information sources will lead to "mismatches" between ontologies. Also, ontologies can be mismatched in the expressiveness of the representational language. This occurs when languages differ in their syntax; when one language has constructs that are not available in another, or the semantics of the same language constructs vary in their implementation. The third source of mismatching occurs at the ontological level where two languages may use the same term to denote different concepts, different terms to denote the same concept; different modelling paradigms, conventions or levels of granularity; using constructs that cover different ranges of the domain. Automated and semi-automated matching systems must be able to identify the variations in concepts as represented by various ontologies and then take appropriate steps to normalise the meanings of those concepts. The final motivation for research in ontology matching lies in the need for autonomous software agents to be able to produce alignment from unknown to known concepts dynamically. In this case, applications must be able to identify and resolve mismatches autonomously, or present the user with information to help resolve discrepancies.

Considering the size and heterogeneity of the Healthcare and Life Sciences (HCLS) domain, many ontologies exist and will be introduced in the future to describe various (sub-)domains (*e.g.*, clinical practices, clinical trials, bio-medical), and many ontologies will exist to describe even similar domains of discourse (*e.g.*, general practitioners and neurologists). A machine supported (semi-) automated ontology matching which can deal with such heterogeneity is essential. To be able to process information within heterogeneous resources, applications must be able to translate resources between ontologies from several sources, which is the essential challenge of any ontology alignment tools.

6.2.1 Ontology Matching: (Semi-)Automated Approaches

[31] provides a classification of ontology matching approaches. The author differentiates between elementary matchers, which include element level and structural level matchers, and combination matchers. Element level matching includes:

- **String Based:** It is based on the idea that the more similar two strings are, the more likely it is that they denote the same concept. Distance functions (*e.g.*, Edit Distance) or variations of distance functions are frequently used. Distance functions map a pair of strings to a real number, where a smaller value of the real number indicates a greater similarity between strings.

- **Language Based:** Natural Language Processing (NLP) techniques use the morphological properties of words to identify important concepts within a source. NLP begins by tokenising an input stream to locate potential words of relevance within a data source. Lemmatisation

then looks at each candidate word and finds all of its inflections (*e.g.*, Observation, Observations). Finally, prepositions, conjunctions and other parts of speech which do not denote concepts are eliminated.

- **Constraint Based:** Evaluations can be made of the internal constraints that exist within an entity, such as data types and cardinality of attributes, to evaluate whether one entity is the same or near another entity.
- **Linguistic Resources:** Common knowledge resources such as a knowledge thesauri maintains information that can be used to ascertain whether two concepts are equal or similar.
- **Alignment Reuse:** Existing mappings between domains can be employed to facilitate the mapping to new domains. The alignment reuse is motivated by the intuition that many ontologies to be matched are similar to already matched ontologies, especially if they are describing the same application domain.
- **Upper Level Formal Ontologies:** Upper level ontologies are a form of external knowledge resource that can be used to ground ontologies in a shared semantic. Suggested Upper Merged Ontology (SUMO) and Descriptive Ontology for Linguistic and Cognitive Engineering (DOLCE [152]) are examples of upper ontologies. The key characteristic of these ontologies is that they are logic-based systems, and therefore, matching techniques exploiting them can be based on the analysis of formal properties.

Structure level matching includes:

- **Graph Based:** This approach take a data source as a labelled graph. It is based on the idea that if nodes from two separate ontologies are related or similar, then the nodes around them are also likely to be similar. Variations of this approach consider child nodes, leaf nodes, and the types of relations between nodes.
- **Taxonomy Based:** This is an extended approach of graph matching that only looks at the specialisation relationship. It works on the idea that *is-a* relations exist through nodes that are already similar, and that neighbours are then also likely to be similar.
- **Repository of Structures:** This approach uses a repository to store ontologies along with some similarity value. When new structures are to be matched, they are first checked for similarity to the structures which are already available in the repository. The idea is to identify structures which are sufficiently similar to be worth matching in more detail, or reusing already existing alignments.

Finally, combination matchers aggregating element and structure level matchers depends on a particular use case.

6.2.2 Ontology Matching: Manual Approaches

Domain experts can choose to construct alignment between domain ontologies. Two manual approaches regarding ontology matching are supported in general by ontology alignment tools; bottom-up and top-down approaches [153]:

- **Top-down:** The top-down approach implies that the matching process starts with complex items (*i.e.*, complex classes, complex properties) and it is usually adopted when a concrete heterogeneity problem has to be resolved. That means that the domain expert is interested only in resolving a particular item mismatch and not in fully aligning the input ontologies.

- **Bottom-up:** The bottom-up approach means that the mapping process starts with the matching of the primitive items and then continues with items having more and more complex descriptions. By this the pairs of primitive items act like a minimal, agreed upon set of correspondences between the two ontologies, and starting from this minimal set more complex relationships could be gradually discovered. This approach is useful when a complete alignment of the two ontologies is desired. The applicability and advantages/disadvantages of each of these approaches depends on the domain requirements and their complexities.

Ontology alignment can also be seen as a third ontology that imports the source and the target ontologies and contains of a set of rules referring to terms from these imported ontologies. From the conceptual point of view, it could be also seen as a "merged ontology" where the two input ontologies were independently put together (the separation is realised by namespaces) and linked by rules. From the technical point of view, the import of the two ontologies can be selective, *i.e.*, only those entities required by the mapping rules are included. By adding the source data to this "merged ontology" and posing queries in terms of the target ontology, the appropriate mapping rules get triggered and results in mediated data.

6.3 Ontology Matching Tools

In recent times several ontology matching algorithms [154] and supported tools have been developed to (semi-)automate the ontology matching task. Some of the matching tools participated in the Ontology Alignment Evaluation Initiative (OAEI[1]) and demonstrated significant matching results are, H-Match [155], Falcon-AO [156], RiMOM [157], BLOOMS [158], and AgreementMaker [159]. We used these five tools for matching our HL7 ontologies and investigated them as the state of the art survey. Matching tools include (semi-)automated algorithms and provide a rich graphical user interface that makes it as easy as possible for a user to "kick-off" the matching process. In addition to (semi-)automated matching tools, general ontology editors (with specialised plugins) such as, TopQuadrant[2], MAFRA[3], WSMT, and Protégé provide human-supported environment for matching source ontologies. These tools have several features including, a reasoner, support for various import or export formats (*e.g.*, RDF, OWL), and a rich GUI.

6.4 Ontology Alignment Languages

To express matching correspondences between ontologies two broad choices have been adopted by the above tools, first, express correspondences in the language which is used for ontologies; second,

[1] http://oaei.ontologymatching.org/
[2] http://www.topquadrant.com/
[3] http://mafra-toolkit.sourceforge.net/

special languages are developed for expressing correspondences separately from ontology languages. The main idea of separating correspondences from ontology definitions is for the reusability of alignment in different ontology formats. Some of the alignment languages used in ontology matching tools and editors are, the Alignment API [160], the Abstract Mapping Language (AML) [161], the Framework for Ontology Alignment and Mapping (FOAM) [162], and a recent extension of the Alignment API called Expressive and Declarative Ontology Alignment Language (EDOAL) [163]. We used the WSMT editor to create the reference alignments for HL7 ontologies. WSMT alignments are stored using the AML syntax, which is formalism neutral, and later grounded to WSML or OWL. In AML there are four types of mapping entities: classes, attributes, relations and instances. They can be simply identified by their URIs or they can be composed within the language in more complex expressions by using boolean constructors (*AND*, *OR* and *NOT*). There are four categories of mappings that can be generated in the WSMT environment:

- **ClassMapping:** two entities being mapped are classes.
- **AttributeMapping:** two entities being mapped are attributes.
- **ClassAttributeMapping:** first entity is a class while the second entity participating in the mapping is an attribute.
- **AttributeClassMapping:** first entity is an attribute while the second entity participating in the mapping is a class.

As discussed, after applying the grounding (*i.e.*, , AML to OWL or WSML), an alignment is seen as a third ontology, which can be used for merging, querying, transformation, or reasoning purposes.

6.5 State of the Art

We investigate five matching algorithms and respective tools (H-Match, Falcon-AO, RiMOM, BLOOMS and AgreementMaker) for their ability to match the HL7 ontologies. Our matching problem is from two perspectives: first, matching ontologies of the same HL7 version; and second, between two versions of the HL7 (*i.e.*, Version 2 and Version 3).

6.5.1 H-Match

H-Match [155] performs ontology matching at different levels of depth by deploying four different matching models spanning from surface to intensive matching, with the goal of providing a wide spectrum of metrics suited for dealing with many different matching scenarios that can be encountered in comparing concept descriptions of ontologies. H-Match takes two ontologies as input and returns the mappings that identify corresponding concepts in the two ontologies, namely the concepts with the same or the closest intended meaning. H-Match mappings are established after an analysis of the similarity of the concepts in the compared ontologies. H-Match performs similarity analysis through affinity metrics to determine a measure of semantic affinity in the range [0, 1]. Semantic affinity is a measure of conceptual similarity between two or more ontological elements.

In H-Match a threshold-based mechanism is enforced to set the minimum level of semantic affinity required to consider two concepts as matching concepts. Given two concepts C_1 and C_2, H-Match calculates a semantic affinity value SA(C_1, C_2) as the linear combination of a linguistic affinity value LA(C_1, C_2) and a contextual affinity value CA(C_1, C_2). The linguistic affinity function of H-Match provides a measure of similarity between two ontology concepts C_1, C_2 computed on the basis of their linguistic features (*e.g.*, concept names). For the linguistic affinity evaluation, H-Match relies on a thesaurus of terms and terminological relationships automatically extracted from the *WordNet*[4] lexical system.

The contextual affinity function of H-Match provides a measure of similarity by taking into account the contextual features of the ontology concepts C_1 and C_2. The context of a concept can include properties, relations with other concepts, and property values. The context can be differently composed to consider different levels of semantic complexity, and four matching models, namely, *surface*, *shallow*, *deep*, and *intensive*, are defined to this end. In surface matching, only the linguistic affinity between the concept names of C_1 and C_2 is considered to determine the conceptual similarity. In the *shallow*, *deep*, and *intensive*, matchings concept similarity is determined by considering both linguistic and contextual affinities. In particular, the shallow matching computes the contextual affinity by considering the context of C_1 and C_2 as composed only by their properties. The *deep* and *intensive* matching extend the depth of concept context for the contextual affinity evaluation of C_1, C_2, by considering also semantic relations with other concepts as well as property values, respectively.

The comprehensive semantic affinity SA(C_1, C_2) is evaluated as the weighted sum of the Linguistic Affinity LA value and the Contextual Affinity CA value, that is:

$$SA(C_1, C_2) = W_{LA} \times LA(C_1, C_2) + (1 - W_{LA}) \times CA(C_1, C_2) \quad \text{(A-1)}$$

where W_{LA} is a weight expressing the relevance to be given for the linguistic affinity in the semantic affinity evaluation process. In terms of usability of each matching model, the surface model is suited for poorly structured ontologies with very simple concept descriptions. The *shallow* and the *deep* models are suited for dealing with schematic ontologies with taxonomic concept descriptions, and the *intensive* model is suited for articulated ontologies with rich concept descriptions. The most accurate and precise results are achieved with the *deep* and *intensive* matching models, provided that the ontology descriptions are detailed enough. However, deep and intensive matching suffer from significant computational delays with larger source ontologies.

H-Match particpated in the 2006 Ontology Alignment Evaluation Initiative (OAEI)[5] covering all types of tests (*e.g.*, anatomy, benchmark, confidence) except test for "retrieval time". The "anatomy" test covers real world ontologies from the biomedical domain, which is close to the healthcare domain discussed in this thesis. H-Match functioning is slow with the larger ontologies [164].

[4] http://wordnet.princeton.edu/
[5] http://oaei.ontologymatching.org/2006/results/

6.5.2 Falcon-AO

Falcon-AO [156] is an ontology matching system for finding, aligning and learning ontologies. Falcon-AO is a similarity-based, generic ontology mapping system. It consists of two elementary matchers: one is a matcher based on linguistic matching for ontologies, called Linguistic Matching for Ontologies (LMO); the other is a matcher based on graph matching for ontologies, called Graph Matching for Ontologies (GMO).

GMO takes the alignments generated by LMO as external input, and outputs additional alignments. The linguistic similarity between two entities relies on their names, labels, comments and other descriptions. LMO combines two different approaches for linguistic similarities: the lexical comparison followed by the statistic analysis. In lexical comparison, LMO calculates the Edit Distance between names of two entities and uses a matching function to capture the string similarity (SS):

$$SS = \frac{1}{e^{\frac{ed}{|s1.len+s2.len-ed|}}} \tag{A-2}$$

In Equation A-2 "ed" denotes the Edit Distance between s1 and s2; s1.len and s2.len denote the length of the input strings s1 and s2, respectively. For statistic analysis, LMO uses VSM (the Vector Space Model) algorithm [165]. In LMO, given a collection of documents, N represents a number of unique terms in the collection. The VSM represents each document as a vector in an N-dimensional space. The components of the vector are the term weights assigned to that document by the term weighting function for each of the N unique terms. The virtual document of an entity consists of "bag of terms" extracted from the entity's names and comments as well as the ones from all neighbors of this entity. The *cosine* similarity between documents (DS) is measured by taking the vectors product: DS = $N.N^t$, where t denotes the number of times where one term occurs in a given document. Falcon-AO combines string similarity (SS) and statistic analysis to calculate the final linguistic similarity by:

$$LinguisticSimilarity = 0.8 \times DS + 0.2 \times SS \tag{A-3}$$

Falcon-AO measures both linguistic comparability and structural comparability of ontologies to estimate the reliability of matched entity pairs with experimental numbers (*e.g.*, 0.8, 0.2). Another important component in Falcon-AO is GMO, which is based on a graph matching approach for ontologies. It uses directed bipartite graphs to represent ontologies and measures the structural similarity between graphs by a new measurement. The main idea of GMO is that: similarity of two entities from two ontologies comes from the accumulation of similarities of involved statements (triples) taking the two entities as the same role (subject, predicate, object) in the triples, while the similarity of two statements comes from the accumulation of similarities of involved entities of the same role in the two statements being compared. Usually, GMO takes a set of matched entity pairs, which are found previously by other approaches (*i.e.*, string similarity (SS)), as external

mapping input to the matching process, and outputs additional matched entity pairs by comparing the structural similarity.

Falcon-AO particpated in the 2007 Ontology Alignment Evaluation Initiative (OAEI)[6] covering all types of test.

6.5.3 RiMOM

RiMOM [157] uses conditional probabilities to model the correspondences between two concepts and based on them defines candidate correspondences. RiMOM considers both metadata and instances. For each possible matching category (*e.g.*, string-based, taxonomy-based) RiMOM discovers the optimal alignment independently of the other categories. Finally, RiMOM combines the results obtained for the different categories of matching, by creating a combined value for each candidate alignment. The RiMOM system can be run in an automatic or semi-automatic mode and uses an iterative process to arrive at a final alignment result. The mapping categories include string-based (edit-distance), lexical (including NLP techniques), constraint-based, linguistic (using a lexical and a statistical similarity dictionaries). As for structure-based techniques, RiMOM uses a taxonomic approach. RiMOM matching task starts with calculating similarity factor between ontological concepts and properties, called a preprocessing step. The higher the similarity of two entities, the more likely the two entities to be aligned. RiMOM denotes the similarity of two entities e_1 and e_2 as $sim(e_1,e_2)$.

$$sim(e_1, e_2) = f(sim\_Meta(e_1, e_2), sim\_Hier(e_1, e_2), sim\_Rest(e_1, e_2), sim\_Inst(e_1, e_2)) \quad \text{(A-4)}$$

The four component similarities (sim_Meta, sim_Hier, sim_Rest, and sim_Inst) represent the similarities of two concepts in metadata, hierarchy, structure, and the instances, respectively. Other matching steps are as follows:

- **Linguistic matching:** it executes multiple linguistic-based strategies and each strategy uses different ontological information and obtains a similarity result $sim(e_1, e_2)$ for each entity pair. These strategies are dynamically selected to be included in the matching task.
- **Similarity combination:** it combines the similarity results obtained by the selected strategies. The weights in the combination are determined by the similarity results.
- **Similarity propagation:** it considers structural similarity and uses three similarity propagation strategies, namely, concept-to-concept, property-to-property, and concept-to-property.
- **Alignment generation:** it fine tunes and outputs the alignment result.

RiMOM participated in the 2007-2009 Ontology Alignment Evaluation Initiatives (OAEI)[7] covering only confidence, benchmarks, anatomy, library, and instance tests.

[6] http://oaei.ontologymatching.org/2006/results/
[7] http://oaei.ontologymatching.org/2009/results/

6.5.4 BLOOMS

BLOOMS [158] ontology matching is based on a bootstrapping approach which uses Wikipedia[8] and the Wikipedia category hierarchy. The BLOOMS system computes alignments with the help of noisy, community generated data available on the Web. BLOOMS has been developed particulary for the Linked Open Data (LOD[9]) cloud. The LOD datasets are well interlinked on the instance level, while they are very loosely connected on the schema level. BLOOMS aims at schema level alignment for the LOD cloud.

The BLOOMS approach constructs a forest (*i.e.*, a set of trees) T_C for each matching candidate class C, which roughly corresponds to a selection of super categories of the class C. Comparison of the forests T_C and T_D for the matching candidate classes C and D then yields a decision whether or not C and D should be aligned. BLOOMS matching steps are as follows:

- **Pre-processing:** input ontologies are pre-processed to (i) remove property restrictions, individuals, and properties; and to (ii) tokenise composite class names to obtain a list of all simple words.

- **Construction of a forest:** a forest is a tree structure hierarchy for each class name C, using information from the Wikipedia. The first step in constructing a forest T_C for class C is to invoke the Wikipedia Web service using the pre-processed class C_p of class C as input. This Web service returns a set of Wikipedia pages WC as results of a search on Wikipedia for the words in the class name C_p. If a returned result is a Wikipedia disambiguation page, it is then removed from WC and replaced by all Wikipedia pages mentioned in the disambiguation page. Such elements are called resulting set WC *senses (c)* for class C.

- **Comparison of forests:** determines which class names are to be aligned. For instance, any concept name C in the one input ontology is matched against any concept name D in the other input ontology. This is done by comparing each $T_s \in T_C$ with each $T_t \in T_D$. A function o is defined, which assigns a real number in the unit interval to each (ordered) pair of forest trees. The value $o(T_s, T_t)$, called the overlap of T_s with T_t. Finally, a comparison of the threshold values for each T_s and T_t determines the types of alignment, that is, *owl:equivalentClass* is used when $T_s = T_t$, and *rdfs:subClassOf* is used for inequalities.

- **Post-processing:** alignment results are processed with the help of the Alignment API. The Alignment API is used to find alignment between the original input ontologies. Alignments returned with a confidence value of at least 0.95 are kept, and added to the results previously obtained. Finally, a reasoner (*e.g.*, Jena) finds inferred alignments.

BLOOMS participated in the 2010 Ontology Alignment Evaluation Initiative (OAEI)[10]. covering only "anatomy" test.

[8] http://en.wikipedia.org/
[9] http://linkeddata.org/
[10] http://oaei.ontologymatching.org/2010/results/

6.5.5 AgreementMaker

AgreementMaker [159] proposes methods that investigate ontological concepts while making a decision on how these concepts should be aligned. User can select one of the following three matching methods:

Base similarity: it is achieved by using a similarity function and applying it to two concepts in the source and target ontologies. If the returned value is greater or equal to a threshold set up by the domain expert, then the two concepts match. The base similarity value is determined with the help of a dictionary like Wordnet. The base similarity $base\_sim(C_1, C_2)$ is measures as:

$$\frac{base\_sim(C_1, C_2) = 2 \times common\_count(L_1, L_2)}{len(L_1) + len(L_2)} \tag{A-5}$$

In Equation A-5, L_1 is a matching string from WordNet dictionary for the concept C_1 and its label (or annotation). $len(L_1)$ is the number of words in the string L_1, and the $common\_count(L_1, L_2)$ is the number of unique common words between L_1 and L_2.

Descendants Similarity Inheritance (DSI): it allows for the parent and in general for any ancestor of a concept to play a role in the identification of the concept. The parent of a concept contributes more to the identity of the concept than its grandparent. The grandparent concept contributes more than the great grandparent, and so on, until the root is reached. DSI is applied in combination with the initial base similarity measurement.

Siblings Similarity Contribution (SSC): it allows siblings of a concept contribute to the identification of the concept. This enhances the quality of the automatic alignment process. Similarly to the DSI method, the SSC method is used in combination with the base similarity measurement.

AgreementMaker participated in the 2009-2011 Ontology Alignment Evaluation Initiatives (OAEI)[11] covering all types of test. AgreementMaker considers label (annotation) of a concept while determining correspondences. This feature may improve the alignment result between HL7 ontologies as annotations are normally used in these ontologies. AgreementMaker has demonstrated significant ontology matching results on the OAEI benchmark.

6.6 Matching HL7 ontologies

As said, we want to investigate how appropriate the generic ontology matching methods are for our relatively straightforward HL7 Version 2 and Version 3 ontologies created in the previous chapters.

In Figure 6.1 and Figure 6.2, *Lab Observation* concepts are modelled differently within the Version 3 ontologies. Similarly, Figure 6.4 and Figure 6.3 show differences in the *AD* datatype of Version 3 and Version 2.

Listing 6.1 shows the XSD for the range class (AD.3.CONTENT) of the AD.3 *objectProperty*. The class AD.3.CONTENT is *equivalent* to the class Adxp.city of Figure 6.4. We notice that Version 3 class names (*e.g.*, Adxp.city, Adxp.country, ObservationRequest) are self-descriptive, while Version 2

[11] http://oaei.ontologymatching.org/2011/results/index.html

```
Class: ObservationRequest
SubClassOf: ActObservation
Class: SpecimenObservation
SubClassOf: ActObservation
Class: Observer SubClassOf: RoleClass
Class: DiabeticType2Observation
SubClassOf: SpecimenObservation
```

Figure 6.1: Version 3: Lab observation 1

```
Class: ObservationOrder.POOB_MT210000UV
SubClassOf: ActObservation
Class: Observer.POOB_MT210000UV
SubClassOf: RoleClass

Class: HemoglobinObservation.POOB_MT210000UV
SubClassOf: ActObservation
```

Figure 6.2: Version 3: Lab observation 2

```
Class: AD
ObjectProperty: AD.1
Domain: AD Range: AD.1.CONTENT
ObjectProperty: AD.2
Domain: AD Range: AD.2.CONTENT
ObjectProperty: AD.3
Domain: AD Range: AD.3.CONTENT
ObjectProperty: AD.4
Domain: AD Range: AD.4.CONTENT
```

Figure 6.3: Version 2 *AD* Ontology Snippet

```
Class: AD SubClassOf: ANY
ObjectProperty: streetAddressLine
Domain: AD Range: Adxp.streetAddressLine
ObjectProperty: state
Domain: AD Range: Adxp.state
ObjectProperty: city
Domain: AD Range: Adxp.city
ObjectProperty: country
Domain: AD Range: Adxp.country
```

Figure 6.4: Version 3 *AD* Ontology Snippet

class names are coded (*e.g.*, AD.1.CONTENT, AD.2.CONTENT) with descriptions attached as annotations (*e.g.*, hl7:Type or hl7:LongName in Listing 6.1).

```
<xsd:complexType name="AD.3.CONTENT">
    <xsd:annotation>
      <xsd:appinfo>
        <hl7:Type>ST</hl7:Type>
        <hl7:LongName>City</hl7:LongName>
      </xsd:appinfo>
    </xsd:annotation>
    ...
```

Listing 6.1: Version 2 XSD Annotations

Matching evaluations are conducted in two categories: (i) finding matches within the same version (*e.g.*, Figure 6.1 and Figure 6.2 of Version 3), and (ii) finding matches between two versions (*e.g.*, Figure 6.3 and Figure 6.4 of Version 2 and Version 3).

```
1  <Alignment><dc:identifier rdf:resource="HL7_order_map"/>
2     <onto1>
3        <uri>http://www.hl7-v3.org#LabObservation1</uri>
4     </onto1>
5     <onto2>
6        <uri>http://www.hl7-v3.org#LabObservation2</uri>
7     </onto2>
8   <map>
9     <Cell id="http://www.hl7-v3.org#ObservationRequesthttp://www.hl7-v3.org#ObservationOrder.POOB_MT210000UV">
10       <entity1> <Class rdf:about="http://www.hl7-v3.org#ObservationRequest"></Class></entity1>
11       <entity2><Class rdf:about="http://www.hl7-v3.org#ObservationOrder.POOB_MT210000UV"></Class></entity2>
12       <measure>1.0</measure><relation>ClassMapping</relation>
13    </Cell>
14  </map>
15 <map>
16    <Cell id="http://www.hl7-v3.org#Observerhttp://www.hl7-v3.org#Observer.POOB_MT210000UV">
17       <entity1> <Class rdf:about="http://www.hl7-v3.org#Observer"></Class></entity1>
18       <entity2><Class rdf:about="http://www.hl7-v3.org#Observer.POOB_MT210000UV"></Class></entity2>
19       <measure>1.0</measure><relation>ClassMapping</relation>
20    </Cell>
21 </map>
22 <map>
23    <Cell id="http://www.hl7-v3.org#DiabeticType2Observationhttp://www.hl7-v3.org#HemoglobinObservation.
          POOB_MT210000UV">
```

```
24      <entity1> <Class rdf:about="http://www.hl7-v3.org#DiabeticType2Observation"></Class></entity1>
25      <entity2><Class rdf:about="http://www.hl7-v3.org#HemoglobinObservation.POOB_MT210000UV"></Class></entity2>
26      <measure>1.0</measure><relation>ClassMapping</relation>
27    </Cell>
28  </map>
29  </Alignment>
```

Listing 6.2: Version 3: Reference Alignment of Fig. 6.1 and Fig. 6.2)

Listing 6.2 and Listing 6.3 show snippets of reference alignments for *Lab Observation* and *datatype* ontologies for Version 2 and Version 3. The reference alignment consists of 800 correspondences, we shows only *Lab Observation* and *Address (AD)* related concepts and properties.

```
1  <Alignment><dc:identifier rdf:resource="HL7_order_map"/>
2     <onto1>
3        <uri>http://www.hl7-v3.org#Datatype</uri>
4     </onto1>
5     <onto2>
6        <uri>http://www.hl7-v2.org#DataType</uri>
7     </onto2>
8   <map>
9     <Cell id="http://www.hl7-v3.org#Adxp.streetAddressLinehttp://www.hl7-v2.org#AD.1.CONTENT">
10      <entity1> <Class rdf:about="http://www.hl7-v3.org#Adxp.streetAddressLine"></Class></entity1>
11      <entity2><Class rdf:about="http://www.hl7-v2.org#AD.1.CONTENT"></Class></entity2>
12      <measure>1.0</measure><relation>ClassMapping</relation>
13    </Cell>
14  </map>
15  <map>
16    <Cell id="http://www.hl7-v3.org#Adxp.countryhttp://www.hl7-v2.org#AD.2.CONTENT">
17      <entity1> <Class rdf:about="http://www.hl7-v3.org#Adxp.country"></Class></entity1>
18      <entity2><Class rdf:about="http://www.hl7-v2.org#AD.2.CONTENT"></Class></entity2>
19      <measure>1.0</measure><relation>ClassMapping</relation>
20    </Cell>
21  </map>
22  <map>
23    <Cell id="http://www.hl7-v3.org#Adxp.cityhttp://www.hl7-v2.org#AD.3.CONTENT">
24      <entity1> <Class rdf:about="http://www.hl7-v3.org#Adxp.city"></Class></entity1>
25      <entity2><Class rdf:about="http://www.hl7-v2.org#AD.3.CONTENT"></Class></entity2>
26      <measure>1.0</measure><relation>ClassMapping</relation>
27    </Cell>
28  </map>
29  <map>
30    <Cell id="http://www.hl7-v3.org#Adxp.statehttp://www.hl7-v2.org#AD.4.CONTENT">
31      <entity1> <Class rdf:about="http://www.hl7-v3.org#Adxp.state"></Class></entity1>
32      <entity2><Class rdf:about="http://www.hl7-v2.org#AD.4.CONTENT"></Class></entity2>
33      <measure>1.0</measure><relation>ClassMapping</relation>
34    </Cell>
35  </map>
36  </Alignment>
```

Listing 6.3: Version 3 and Version 2: Reference Alignment of Fig. 6.4 and Fig. 6.3

Reference alignments are created manually using HL7 specifications that describe similarities between the messaging elements.

6.6.1 Ontology Matching: HL7 Version 3

Matching results presented in this section are the alignments retrieved at the default threshold value of 0.5 used by the five matching tools. Furthermore, the Section 6.8 provides a summary of the alignments retrieved at three threshold ranges: minimum range (0.1-0.4), medium range (0.4-0.7), and maximum range (0.7-1).

The Table 6.1 shows a snippet of Falcon-AO matching results between the *Lab Observation* concepts shown in Figure 6.1 and Figure 6.2. The matching concepts (*e.g.*, ActObservation, RoleClass) are from global ontologies which are shared by the local *Lab Observation* ontologies. Falcon-AO

Lab Observation 1	Lab Observation 2	Threshold
ActObservation	ActObservation	0.5
ObservationRequest	HemoglobinObservation.POOB_MT210000UV	0.5
SpecimenObservation	ObservationOrder.POOB_MT210000UV	0.5
RoleClass	RoleClass	0.5
Observer	Observer.POOB_MT210000UV	0.5

Table 6.1: Falcon-AO: Matching Version 3 Ontologies

finds one correct match (Observer & Observer.POOB MT210000UV) and two incorrect ones (ObservationRequest & HemoglobinObservation.POOB_MT210000UV, SpecimenObservation & ObservationOrder.POOB_MT210000UV) among three heterogeneous concepts as per the reference alignment shown in the Listing 6.2. The reason for three correct correspondences returned by Falcon-AO is the combination of linguistic (basically string matching) and graph-based matchers within the system. The linguistic matcher takes entities names, comments and other descriptions and the results retrieved are then used as input for the graph matcher. For example, the correct correspondence (Observer, Observer.POOB_MT210000UV) is the result of the linguistic matcher where other alignments have been retrieved by the graph matcher.

Lab Observation 1	Lab Observation 2	Threshold
ActObservation	ActObservation	0.5
SpecimenObservation	HemoglobinObservation.POOB_MT210000UV	0.5
ObservationRequest	ObservationOrder.POOB_MT210000UV	0.5
Observer	Observer.POOB_MT210000UV	0.5
RoleClass	RoleClass	0.5

Table 6.2: H-Match: Matching Version 3 Ontologies

Table 6.2 shows H-Match matching results. We notice an improvement in the overall matching results. For instance, the matching result shows two correct correspondences (ObservationRequest & ObservationOrder.POOB_MT210000UV, Observer & Observer.POOB_MT210000UV) and one incorrect correspondences (SpecimenObservation & HemoglobinObservation.POOB_MT210000UV) compared to the reference alignment (Listing 6.2). The reason for two correct correspondences is different levels of matching approaches employed by the H-Match system. In this case, results shown are retrieved by *deep* matching. As described, in *deep* matching properties and their values are taken into account while determining the correspondences between two matching concepts. H-Match relies on *WordNet* for linguistic matching and the unavailability of matching terms within the *WordNet* structure reduces the matching results. The ability to include more thematic structure (*e.g.*, clinical, life-sciences) might improve the H-Match matching results.

Lab Observation 1	Lab Observation 2	Threshold
ActObservation	ActObservation	0.5
ActObservation	RoleClass	0.5
Observer	ActObservation	0.5
RoleClass	RoleClass	0.5

Table 6.3: BLOOMS: Matching Version 3 Ontologies

Table 6.3 shows the BLOOMS matching result. While BLOOMS matching results are quite impressive for the Linked Open Data (LOD) datasets and respective schemas, in our case with HL7, correct matching results are low. We see two prime reasons for the low results: (i) BLOOMS matching relies heavily on the Wikipedia/DBPedia resources. In cases where concept names are

missing in the Wikipedia resources, the matcher is unable to find suitable correspondences.

Lab Observation 1	Lab Observation 2	Threshold
ActObservation	ActObservation	0.5
SpecimenObservation	HemoglobinObservation.POOB_MT210000UV	0.5
ObservationRequest	ObservationOrder.POOB_MT210000UV	0.5
Observer	Observer.POOB_MT210000UV	0.5
RoleClass	RoleClass	0.5

Table 6.4: RiMOM: Matching Version 3 Ontologies

In our case, concepts name like "ObservationOrder.POOB_MT210000UV", might be impossible to find within a general information structure like Wikipedia, and (ii) BLOOMS tokenises complex concept names. For example, names with hyphens, underscores, etc. are separated into independent string tokens. Then the Wikipedia Web service finds all the matching resources for the independent string tokens. Therefore, the concept name like "ObservationOrder.POOB_MT210000UV", is separated into two names (ObservationOrder.POOB, MT210000UV), which results in low matching. We discussed earlier that, in HL7 systems, entity names are either coded or form a complex pattern, which makes BLOOMS ineffective for matching HL7 ontologies.

Lab Observation 1	Lab Observation 2	Threshold
ActObservation	ActObservation	0.5
SpecimenObservation	HemoglobinObservation.POOB_MT210000UV	0.5
ObservationRequest	ObservationOrder.POOB_MT210000UV	0.5
Observer	Observer.POOB_MT210000UV	0.5
RoleClass	RoleClass	0.5

Table 6.5: AgreementMaker: Matching Version 3 Ontologies

Table 6.4 and Table 6.5 show RiMOM and AgreementMaker matching results. Similar to H-Match only one retrieved correspondence is incorrect (SpecimenObservation & HemoglobinObservation.POOB_MT210000UV). We observe that the combination of linguistic and graph-based matching improves the overall matching results. Also, taking other dimensions of matching features (*e.g.*, property, property value, comments) contributes to improved matching results. We suggest that the inclusion of more thematic information structures (instead of general structure like Wikipedia or Wordnet) might be a step forward in improving the effectiveness of the automated ontology matching systems, for domain-specific cases like HL7 systems.

6.6.2 Ontology Matching: HL7 Version 2 - Version 3

The five tools have shown considerable matching results for Version 3 ontologies. We now proceed with matching Version 2 and Version 3 ontologies. Figure 6.6 shows the Falcon-AO matching results. Falcon-AO has shown low matching result. In this case (*i.e.*, Version 2 - Version 3 ontologies) very few concept names have linguistic similarity. Version 2 names are coded and described using annotation attributes, while Version 3 names are self-descriptive.

The last two alignments (AD & AD.1.CONTENT and AD & AD.2.CONTENT) of Table 6.6 are incorrect. The prime reason for this incorrectness is the substring similarities between the concept names (AD, AD.1.CONTENT, and AD.2.CONTENT). False inputs from the linguistic matcher ultimately cause low results from the graph matcher.

V2(AD)	V3(AD)	Threshold
AD	AD	0.5
ST	ST	0.5
AD.1.CONTENT	AD	0.5
AD.2.CONTENT	AD	0.5

Table 6.6: Falcon-AO: Matching Version 2 - Version 3 Ontologies

V2(AD)	V3(AD)	Threshold
AD	AD	0.5
ST	ST	0.5
AD	Adxp.StreetAddressLine	0.5
AD.1.CONTENT	AD	0.5
AD.2.CONTENT	AD	0.5

Table 6.7: H-Match: Matching Version 2 - Version 3 Ontologies

Table 6.7 shows matching results from the H-Match system. We see no major improvement when compared to the Falcon-AO system. For Version 3 ontologies, H-Match system has shown significant improvement compared to the Falcon-AO matching results.

V2(AD)	V3(AD)	Threshold
AD	AD	0.5
ANY	AD	0.5
AD	Adxp.StreetAddressLine	0.5
AD	Adxp.streetNameType	0.5
ST	ST	0.5
ST	Adxp.StreetAddressLine	0.5

Table 6.8: BLOOMS: Matching Version 2 - Version 3 Ontologies

However, H-Match improves with a partially correct correspondence (AD & Adxp.StreetAddressLine) compared to the Falcon-AO system. For example, Adxp.StreetAddressLine class is a true match for the AD.1.CONTENT class and both classes have a super class AD in the respective ontologies.

Table 6.8 shows the BLOOMS matching result. We notice an increase in the correspondences comparing other matching tools. However, the number of correct correspondences (*e.g.*, AD & AD, ST & ST) is similar to the Falcon-AO and H-Match systems. Incorrect correspondences (*e.g.*, AD & Adxp.StreetAddressLine, AD & Adxp.streetNameType, ST & Adxp.StreetAddressLine) result from substring matches.

Table 6.9 and Table 6.10 show RiMOM and AgreementMaker matching results. The matching results are improved by a correct correspondence (AD.1.CONTENT & Adxp.StreetAddressLine) and two incorrect correspondences (AD.1.CONTENT & Adxp.streetNameType, AD.1.CONTENT & Adxp.City). Similarly, for the class AD.2.CONTENT a correct correspondence and several incorrect correspondences are retrieved by the RiMOM and AgreementMaker. Although, RiMOM and AgreementMaker are able to find the correspondences between Version 2 and Version 3 ontologies, the amount of incorrect correspondences is high. This causes problem of filtering the correct alignments among the total matching results.

We observe that the matching results for Version 2 and Version 3 ontologies are significantly poor from all five ontology matching systems. While we suggested that thematic information structure might improve matching results for Version 3 ontologies, in the case of Version 2 ontologies, we need to develop a method that can include: (i) domain-specific matching patterns between ontolog-

V2(AD)	V3(AD)	Threshold
AD	AD	0.5
AD.1.CONTENT	Adxp.StreetAddressLine	0.5
AD.1.CONTENT	Adxp.streetNameType	0.5
AD.1.CONTENT	Adxp.City	0.5
ST	ST	0.5

Table 6.9: RiMOM: Matching Version 2 - Version 3 Ontologies

V2(AD)	V3(AD)	Threshold
AD	AD	0.5
AD.1.CONTENT	Adxp.StreetAddressLine	0.5
AD.1.CONTENT	Adxp.streetNameType	0.5
AD.1.CONTENT	Adxp.City	0.5
ST	ST	0.5

Table 6.10: AgreementMaker: Matching Version 2 - Version 3 Ontologies

ical entities; and (ii) other dimensions (*e.g.*, annotations) for determining correspondence between ontological entities.

6.7 SPARQL Recipes

We investigated five state of the art ontology alignment systems in order to use them to match HL7 ontologies. We observed that the matching results of these systems, especially for Version 2 and Version 3 ontologies, were rather poor, even though they performed well on established benchmarks. We were thus left with finding our own solution to align HL7 ontologies. Our query-based approach, SPARQL Recipes, outperforms state of the art ontology alignment systems for aligning HL7 ontologies. Our approach is to build assistance or guidance recipes that can improve the accuracy of alignment results.

```
CONSTRUCT { ?v3 owl:equivalentClass ?v2 }
WHERE { ?v3 rdf:type owl:Class . ?v2 rdf:type owl:Class .
        ?v2 rdfs:label ?LongName . { FILTER regex(str(?v3), str(?LongName), ''i'')}}
```

Listing 6.4: SPARQL Recipe for Ontology Matching (Classes)

Instead of relying on automated alignments, a simple query-based method for expressing customised and complex alignments between ontologies has been proposed in [166, 167] using the SPARQL query language. Furthermore, this query-based method is extended by using an ontology alignment language and SPARQL language together [167]. We have employed this query-based method specifically to match Version 2 and Version 3 ontologies. Surprisingly, a single, generic SPARQL query that matches concept names of the Version 3 ontology against *LongName* (the label annotations) of the Version 2 ontology outperforms all automatic mapping attempts.

We notice in Listing 6.1 that the annotation of *hl7:LongName* ("City") is a substring of the concept named Adxp.city. This mapping from the annotation in onto the concept name by substrings is a recurring pattern in mappings between Version 2 and Version 3. Similarly, HL7 ontologies contain several other "simple patterns" that can guide us in determining correspondences between ontological elements. A domain expert is required to analyse and identify such "simple patterns". Listing 6.4 shows a "simple pattern" created as a "SPARQL recipe":

V2(AD)	V3(AD)
AD.1.CONTENT	Adxp.StreetAddressLine
AD.2.CONTENT	Adxp.country
AD.3.CONTENT	Adxp.city
AD.4.CONTENT	Adxp.state

Table 6.11: SPARQL Recipe: Matching Version 3 - Version 2 Classes (AD Datatype)

By employing this simple "SPARQL recipe", we observed significant successful matching results for classes that share similarity but are described and modelled differently. The recipe basically applies a "FILTER" expression to match all the Version 3 concepts against Version 2 concepts whose annotations (LongName) value is similar to Version 3 concept names. Table 6.11 shows the "SPARQL recipe" matching result (compared to the reference alignment of Listing 6.3), the accuracy of the correspondences have been improved significantly.

```
CONSTRUCT { ?v3 owl:equivalentProperty ?v2 }
WHERE { ?v3 rdf:type owl:ObjectProperty . ?v2 rdf:type owl:ObjectProperty . ?v2 rdfs:range ?v2range .
        ?v3 rdfs:range ?v3range . ?v2 rdfs:domain ?v2domain . ?v3 rdfs:domain ?v3domain .
        ?v2range owl:equivalentClass ?v3range . ?v2domain owl:equivalentClass ?v3domain };
```

Listing 6.5: SPARQL Recipe for Ontology Matching (Object Properties)

Listing 6.5 shows a SPARQL Recipe that matches object properties of the two versions. We notice that the last two triples *(?v2range owl:equivalentClass ?v3range and ?v2domain owl:equivalentClass ?v3domain)* use the *owl:equivalentClass* correspondences discovered by the first recipe (Listing 6.4). All correspondences discovered by the first recipe are merged within the original ontologies. This helped us to construct additional recipes on top of previously discovered correspondences between source and target ontologies. The greater flexibility and customisation of SPARQL Recipes offer freedom to domain experts on various types of matching they might require in their environment.

V2(AD)	V3(AD)
AD.1	streetAddressLine
AD.2	country
AD.3	city
AD.4	state

Table 6.12: SPARQL Recipe: Matching Version 3 - Version 2 Object Property (Datatype)

For instance, in one case domain experts might be interested in class matching only, or in other cases property matching is required, or in applying classes and properties matching together. Table 6.12 shows matching results for the object properties. Listing 6.6 shows a SPARQL Recipe for matching Data Properties of HL7 ontologies. We notice that data properties have annotation values (?v2fixed or ?v3fixed) provided by the HL7 specifications. These annotation values (*e.g.*, SI, CS, CX) are a set of coded vocabularies which describe the purpose of each data properties. Version 2 and Version 3 specifications share similarities in these fixed annotation values. The "FILTER" expression checks the matching between fixed annotation values from Version 2 and Version 3 ontologies.

```
CONSTRUCT { ?v3 owl:equivalentProperty ?v2 }
WHERE { ?v3 rdf:type owl:DatatypeProperty . ?v2 rdf:type owl:DatatypeProperty . ?v2 rdfs:label ?v2fixed .
        ?v3 rdfs:label ?v3fixed . ?v2 rdfs:domain ?v2domain . ?v3 rdfs:domain ?v3domain .
        ?v2domain owl:equivalentClass ?v3domain . { FILTER regex(str(?v3fixed), str(?v2fixed), ''i'')} };
```

Listing 6.6: SPARQL Recipe for Ontology Matching (Datatype Properties)

As described before, recipes can be created by domain experts after analysing similarities between ontological elements in HL7. As opposed to alignment algorithms that usually just search for simpler alignments, such recipes can express complex correspondences between ontology instances, involving built-in function calls, etc. For example, it can allow one to express complex correspondences such as concatenating classes or properties (*e.g.*, first/last names, dates).

6.8 Evaluation

Table 6.13 shows the evaluation result for matching HL7 ontologies. The second column describes matching between Version 3 ontologies and the third column shows the matching of Version 2 and Version 3 ontologies. In the second column, matching ontologies (*i.e.*, local message ontologies for Version 3) include "common concepts" inherited from the global reference ontology.

Alignment Tool	**H7(V3-V3)** [p-r]	**HL7(V2-V3)** [p-r]	**Threshold**
Falcon-AO	(i) 70%(p)-60%(r) (ii) 70%(p)-50%(r) (iii) 70%(p)-50%(r)	(i) 30%(p)-30%(r) (ii) 30%(p)-20%(r) (iii) 30%(p)-20%(r)	(i) 0.1-0.4 (ii) 0.4-0.7 (iii) 0.7-1
H-Match	(i) 80%(p)-100%(r) (ii) 80%(p)-90%(r) (iii) 80%(p)-90%(r)	(i) 40%(p)-30%(r) (ii) 40%(p)-20%(r) (iii) 40%(p)-20%(r)	(i) 0.1-0.4 (ii) 0.4-0.7 (iii) 0.7-1
BLOOMS	(i) 90%(p)-40%(r) (ii) 90%(p)-30%(r) (iii) 90%(p)-30%(r)	(i) 90%(p)-20%(r) (ii) 90%(p)-10%(r) (iii) 90%(p)-10%(r)	(i) 0.1-0.4 (ii) 0.4-0.7 (iii) 0.7-1
RiMOM	(i) 60%(p)-100%(r) (ii) 70%(p)-90%(r) (iii) 70%(p)-90%(r)	(i) 40%(p)-40%(r) (ii) 40%(p)-40%(r) (iii) 30%(p)-20%(r)	(i) 0.1-0.4 (ii) 0.4-0.7 (iii) 0.7-1
AgreementMaker	(i) 70%(p)-100%(r) (ii) 70%(p)-90%(r) (iii) 70%(p)-90%(r)	(i) 40%(p)-50%(r) (ii) 40%(p)-50%(r) (iii) 30%(p)-20%(r)	(i) 0.1-0.4 (ii) 0.4-0.7 (iii) 0.7-1
SPARQL Recipes	80%(p)-90%(r)	50%(p)-60%(r)	

Table 6.13: Matching Evaluation

The third column shows matching of global ontologies for different versions (Version 2 and Version 3), which means "common concepts" are irrelevant and concepts are different in terms of their names or structure. The percentages (in the second and third columns) denote *precision (p)* and *recall (r)* for evaluating accuracy of matches discovered. Precision indicates the "correctness" and recall measures the "completeness" of matching results. Both parameters are measured against the reference alignment R with alignment A returned by the matching tools, where $p=\frac{|R \cap A|}{|A|}$ and $r=\frac{|R \cap A|}{|R|}$. In our case the two input ontologies with 1563 and 1062 concepts have 800 prospective

matching concepts (*i.e.*, reference alignment R). The fourth column shows three threshold ranges what gave us further insight for the results retrieved by the five matching tools.

We observe in Table 6.13 that matching results are significantly different between the second and third columns. For example, Falcon-AO has higher p=70% and lower r=60% (second column), which indicates that the completeness of alignments is lower. Precision is higher due to obvious matches (the common concepts) from the reference ontology. In the third column, the Falcon-AO recall value is much lower as obvious matches between Version 2 and Version 3 are negligible. H-Match have significantly improved recall for the second column (*i.e.*, 90-100%) for the threshold range (0.1-0.4). H-Match, RiMOM and AgreementMaker precision and recall ratios are best because it has different levels of matchings. In case of H-Match best matches are retrieved by (*intensive* or *deep* matching). However, *intensive* matching shows greater delays with ontologies of size greater than 100 concepts. Unfortunately, the H-Match recall measurement for the third column is quite low. This is due to the naming scheme (coded for Version 2 vs. self-descriptive names Version 3 ontologies). For example, AD from Version 3 matches with the AD, AD.1, AD.1.CONTENT, AD.2.CONTENT concepts of Version 2, which makes *false positives* within the alignment higher, thus, precision lower. BLOOMS recall is comparatively lower in the second and third columns. In the second column, RiMOM and AgreementMaker has similar recall values comparing H-Match, however, precision falls because of more incorrect matching results (*i.e.*, *false positives*). SPARQL recipes matching results are similar to H-Match (*precision* 80%, *recall* 90% for the threshold range 0.4-1) for Version 3 ontologies (second column of Fig. 6.13) and significantly improved *precision (p)* 50% and *recall(r)* 60% for Version 2 and Version 3 ontologies. This improved matching accuracy by SPARQL recipes contributes to our second research challenge (*i.e.*, "how to automate alignment of HL7 ontologies with greater accuracy of correspondences retrieved").

We like to mention that among all the five tools AgreementMaker provides the best interface, starting from ontology selection to evaluation of matching ontologies. User friendly templates are provided for selecting various parameters and different types of matchers (linguistic-based, string-based, graph-based). Also, user can export the alignment format in several types (*e.g.*, plain text, RDF). User friendly interface is a key to involve domain-experts in using knowledge-based systems like the AgreementMaker. In comparison to the automated ontology alignment tools, a major limitation of "SPARQL recipes" is the lack of standard matching measures, such as threshold, similarity coefficient, etc. In "SPARQL recipes" approach the domain knowledge is reflected in matching patterns, which is ultimately responsible for the greater precision and recall values.

6.9 Summary

The chapter presents standard ontology matching approaches. We present state of the art ontology alignment tools for matching HL7 ontologies. We have categorised ontology matching in two types: (i) matching HL7 Version 3 ontologies, and (ii) matching Version 2 and Version 3 ontologies. State of the art ontology alignment tools have shown low matching results, especially when matching Version 2 and Version 3 ontologies. These low matching results motivated us to develop SPARQL recipe based ontology matching. We recognise that various "simple patterns" exist between HL7 ontologies that provide guidance in determining correspondences between ontological elements. Finally, we evaluated the ontology matching results of the five alignment tools and proposed SPARQL recipes which gave more coverage and accuracy.

Chapter 7

PPEPR Framework

7.1 Introduction

We discussed in the previous chapters that HL7 integration systems are used to ease the integration burden between healthcare institutes. These integration systems suffer a number of drawback *e.g.*, quadratic size alignments between deployed applications, inability to separate global and local information sources (see Section 2.4). This chapter introduces a Electronic Healthcare Record (EHR) integration platform, called PPEPR: Plug and Play Electronic Patient Records [39]. PPEPR is designed on the semantic Service-Oriented Architecture (sSOA) principles [168]. In this thesis we concentrate on data interoperability mechanisms.

The chapter first presents the PPEPR approach and the architecture. The chapter then describes the design time and run time phases. Next, the chapter explains the functionality and life cycle of the Adapter Framework responsible for the message transformation. The chapter briefly describes the Data Mediator and shows PPEPR working in a real environment (based on the Integration Scenario described in Chapter 3). Finally, the chapter presents an evaluation of the PPEPR framework based on various parameters.

7.2 PPEPR Approach

PPEPR is software which connects healthcare enterprises. PPEPR can work as a standalone product directly interfacing with EHR systems, or can be used as an add-on to existing HL7 engines. The PPEPR software consists of two parts: the design-time and the run-time. The design-time portions of the system are used when installing PPEPR and configuring the various EHR systems which are to be made interoperable. The run-time portion consists of an Adapter Framework and a Data Mediator. The benefits of PPEPR over existing integration systems are:

- **Semi Automatic:** The work involved in modelling the environment into which PPEPR will work is semi-automatic. The manual effort to be done during the design-time is the construction and alignment of ontologies. The design-time task is done once for every HL7

applications complying with Version 2 and Version 3 specifications. The run-time operation of PPEPR is completely automatic.

- **Flexible:** PPEPR allows the easy addition and modification of models reflecting the changing environment within a hospital. Upgrading an EHR system from HL7 Version 2 to one which uses HL7 Version 3 requires minor changes. Once the models are created, the new system can be incorporated into the hospital without any additional software development.

- **Robust:** With the hub-and-spoke topology inherent in using PPEPR, the system is more robust than a peer-to-peer topology more typical of a system-by-system integration effort. Allied with the hub-and-spoke topology is the suite of models built for use with PPEPR. These models are built at a conceptual level and are more resistant to change than the low-level mapping functionality available with other systems.

- **HL7 Version 2 and Version 3:** As noted above PPEPR seamlessly cater for Version 2 and Version 3 EHR systems.

Figure 7.1 shows the operations of the PPEPR framework, which is divided into two phases: (i) design-time: this schema level phase concerns the modelling and alignment of the healthcare knowledge bases, which is presented in Chapter 4, Chapter 5, and Chapter 6, and (ii) run-time: this instance level phase mediates HL7 messages using the concepts modelled and aligned at the schema level. During run-time PPEPR takes as input message instances and lifts (see Section 7.5.1) them to ontological instances, mediates to required HL7 version (see Section 7.5.2), and lowers (see Section 7.5.3) the ontological instances to corresponding message instances. The run-time message transformation (lifting and lowering) is bi-directional.

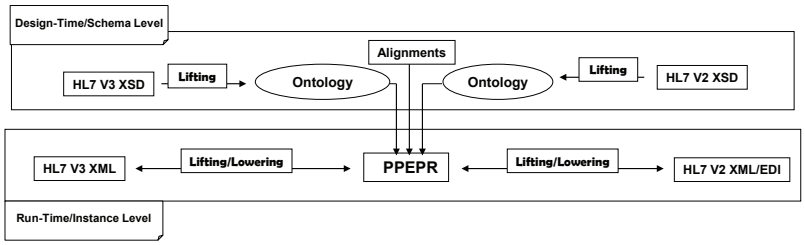

Figure 7.1: PPEPR: Design-Time and Run-Time

The PPEPR framework uses the ontological schema references for bi-directional message transformation. PPEPR's approach of separating design-time and run-time parts within the framework helps delineating the scope and tasks as well as helping to organise, maintain and access the healthcare data and knowledge bases.

7.3 PPEPR Architecture

Figure 7.2 shows the PPEPR architecture. The components within white rectangles (Adapter Framework and Data Mediator) are the main focus of this thesis.

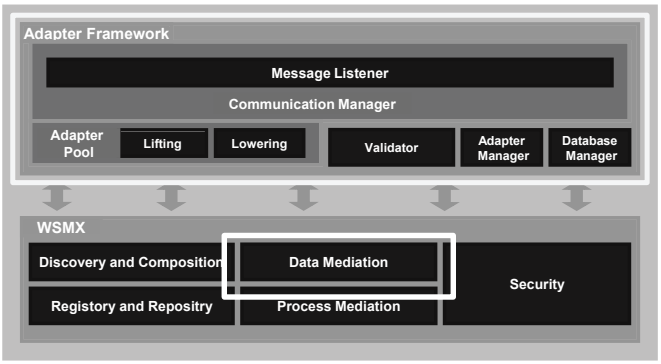

Figure 7.2: PPEPR Architecture

The Adapter Framework is a common platform for enabling communication between systems that use different data formats. It is an application specific component that transforms data from its source to target format. The framework provides an adapter skeleton and allows developers to register (plug-in) their adapters and invokes the target application through one of the registered adapters. Direct invocation is also possible if the input message is described in the target-required format. The adapter's main role is at run-time. Lifting and Lowering rules executed by the adapter are specified at design-time. The main components within the Adapter Framework are:

- **Communication Manager** is a gateway to the adapter framework which can be accessed by a *Message Listener* via a Web Service interface or TCP/IP port (for non-Web Service EHRs). The *Message Listener* waits for any new message incoming from the network. Once requested by a back end application, the Communication Manager interprets the request and invokes the Adapter Manager. If the invocation is successful, it returns a response message which contains either a "success" message or the "response" returned from the target application.
- **Adapter Manager** schedules jobs between adapters and the data format Validator. It updates and refreshes the list of the adapters as they are registered and unregistered.
- **Validator** is responsible for validating incoming and outgoing messages within the adapter framework. An XML message, sent by the back end application, passes through the validation process against the HL7 schema specifications. Message validation confirms compliance of exchanged messages with the standard (HL7) specifications.
- **Adapter Pool** is a "repository or container" of all the registered adapters.
- **Lifting Adapter** is a specialised adapter (HL7 specific) which transforms (lifts) a XML message to an ontological instance.
- **Lowering Adapter** is a specialised adapter which transforms (lowers) an ontological instance to an XML message.
- **Database Manager** manages database connections, exceptions, and utilities.

The bottom of Figure 7.2 shows the Web Service Execution Environment (WSMX) system. WSMX is a Semantic Execution Engine used to solve the data and process heterogeneity. A detailed

description of WSMX [169] is outside the scope of this thesis. Therefore we explain WSMX from the PPEPR perspective, mainly focusing on data meditation. The ontologies and alignments created are processed within the WSMX environment as follows:

- **Mapping Rules Creator:** It is used during runtime for transforming the mappings into mapping rules. The main distinction between mappings and mapping rules is that the mappings are language independent (using Abstract Mapping Language (AML)) while the mapping rules are expressed in the formal language Flora-2 [170]. The mapping rules express the mappings in an executable way. Also, mappings express the similarities between the two ontologies, while the mapping rules describe how these similarities are used in order to transform instances of the first ontology to instances of the second ontology. This separation in mappings and mapping rules ensures a high degree of flexibility: the mapping rules can be expressed in the appropriate language suited for the available execution environment like WSMX.
- **Rules Execution Environment:** This environment is invoked during the mediation process, and performs the execution of mapping rules. During runtime, the mapping rules are received from the mapping rules generator and executed inside the Rules Execution Environment.

The PPEPR architecture is separated into self-contained components, where each component has a specific task assigned during the overall mediation of heterogeneous healthcare messages. Such a "separation-of-concern" helps delineating the mediation tasks and maintenance of the system.

7.4 PPEPR Design Time

We discussed in the Section 7.2 that the design time phase concerns with the modelling and alignment of the healthcare knowledge bases. We now present a walk-through using examples presented in previous chapters showing building, alignment, and finally execution of the HL7 ontologies.

```
<addr use="HP">
 <streetAddressLine partType="SAL"> 22 Dangan Heights <
       streetAddressLine/>
 <country partType="CNT"> Ireland <country/>
 <state partType="STA"> Galway <state/>
 <city partType="CTY"> Galway <city/>
 <postalCode partType="ZIP"> 99 <postalCode/>
 ....
</addr>
```

Figure 7.3: Patient Address: XML Instance

```
<xs:complexType name="AD" mixed="true">
 <xs:complexContent>
  <xs:extension base="ANY">
   <xs:sequence>
    <xs:choice minOccurs="0" maxOccurs="unbounded">
    <xs:element name="streetAddressLine" type="adxp.
          streetAddressLine"/>
    <xs:element name="country" type="adxp.country"/>
    <xs:element name="state" type="adxp.state"/>
    <xs:element name="city" type="adxp.city"/>
    <xs:element name="postalCode" type="adxp.postalCode"/>
    ....
</xs:complexType>
```

Figure 7.4: Patient Address (AD): XSD Specification

Figure 7.3 and Figure 7.4 show an XML instance and the schema of the HL7 Version 3 AD datatype. The AD datatype represents the mailing, home, or office addresses of healthcare entities (*e.g.*, patient, physician).

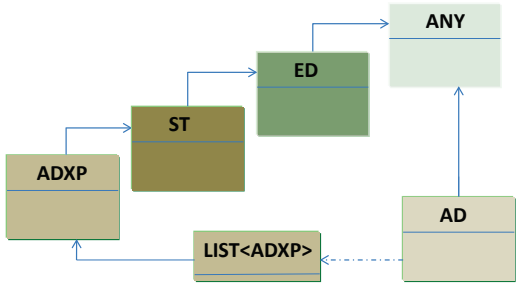

Figure 7.5: Simplified UML Hierarchy of the AD Datatype

Figure 7.5 shows a simplified conceptual hierarchy of AD from the RIM specification [4]. The fixed arrows indicate a *subclass* (is-a) relationship and the broken arrow represents a *part-of* relation. In this case AD inherits from the ANY class (similar to *owl:Thing*) and made of list (similar to *rdf:List*) of components from *ADXP* class.

```
concept Datatype#AD subConceptOf Datatype#ANY
streetAddressLine impliesType Datatype#Adxp.streetAddressLine
country impliesType Datatype#Adxp.country
state impliesType Datatype#Adxp.state
city impliesType Datatype#Adxp.city
postalCode impliesType Datatype#Adxp.postalCode
```

Figure 7.6: Version 3 AD Datatype: WSML Syntax

A postal address (AD) has a sequence of address parts (ADXP), such as street or post office

box, city, postal code, country, etc. Similarly, Figure 7.4 shows *complexType* AD extends the base ANY and has a sequence of elements describing each address parts (or components). AD is the most commonly used element in clinical message exchanges. For instance, messages ("Observation Order Fulfilment Request" and "Observation Order Complete") in Figure 4.3 contain the postal address of a patient.

```
Class: Datatype:AD SubClassOf: Datatype:ANY
ObjectProperty: streetAddressLine Domain: AD Range: Datatype:Adxp.streetAddressLine
ObjectProperty: country Domain: Datatype:AD Range: Datatype:Adxp.country
ObjectProperty: state Domain: Datatype:AD Range: Datatype:Adxp.state
ObjectProperty: city Domain: Datatype:AD Range: Datatype:Adxp.city
ObjectProperty: postalCode Domain: Datatype:AD Range: Datatype:Adxp.postalCode
```

Figure 7.7: Version 3 AD Datatype: OWL Syntax

Figure 7.6 and Figure 7.7 show ontology snippets transformed from the example shown in Figure 7.4. This transformation is based on the mechanism proposed and described in Chapter 5. The proposed transformation rule defines an XSD base extension as a *subclass/subconcept* relation. According to this rule, the transformation results into *concept AD subConceptOf Datatype#ANY* (Figure 7.6).

```
<xsd:complexType name="AD">
  <xsd:sequence>
    <xsd:element name="AD.1" type="AD.1.CONTENT"/>
    <xsd:element name="AD.2" type="AD.2.CONTENT"/>
    <xsd:element name="AD.3" type="AD.3.CONTENT"/>
    <xsd:element name="AD.4" type="AD.4.CONTENT"/>
    <xsd:element name="AD.5" type="AD.5.CONTENT"/>
    .....
  </xsd:sequence>
</xsd:complexType>
```

Figure 7.8: Version 2 AD Datatype: XML Schema

Figure 7.8 shows an XML Schema snippet for the Version 2 AD datatype. In this case, AD does not extend any base type and each of the AD elements belong to certain type (*e.g.*, *AD.1.CONTENT*).

```
Class: AD
ObjectProperty: AD.1 Domain: AD Range: AD.1.CONTENT
ObjectProperty: AD.2 Domain: AD Range: AD.2.CONTENT
ObjectProperty: AD.3 Domain: AD Range: AD.3.CONTENT
ObjectProperty: AD.4 Domain: AD Range: AD.4.CONTENT
ObjectProperty: AD.5 Domain: AD Range: AD.5.CONTENT
....
```

Figure 7.9: Version 2 AD Datatype: OWL Syntax

According to the transformation rules defined in Chapter 5, any *complexType* will be transformed to ontological class, and an element with a custom type will be transformed to *objectProperty*. Figure 7.9 shows the AD ontology snippet for Version 2.

We notice that the properties of both AD ontologies from Version 2 and Version 3 have different naming schemes for similar concepts (*e.g.*, *Adxp.streetAddressLine* and *AD.1.CONTENT*). Such

```
Class: Adxp.streetAddressLine
EquivalentTo: AD.1.CONTENT

Class: Adxp.state
EquivalentTo: AD.2.CONTENT

Class: Adxp.city
EquivalentTo: AD.3.CONTENT
```

Figure 7.10: Aligning Version 2 - Version 3 AD Datatypes: OWL Syntax

heterogeneity requires the alignment of datatype ontologies. Chapter 6 presented the SPARQL Recipes to align such differences. For example, figure 7.10 shows a snippet of alignment result returned by the SPARQL Recipes and figure 7.11 shows these alignments in the AML format used within the WSMX environment.

```
<Alignment><dc:identifier rdf:resource="HL7_order_map" />
   <onto1>
      <uri>http://www.hl7-v3.org#Datatype</uri>
   </onto1>
   <onto2>
      <uri>http://www.hl7-v2.org#DataType</uri>
   </onto2>
 <map>
   <Cell id="http://www.hl7-v3.org#Adxp.streetAddressLinehttp://www.hl7-v2.org#AD.1.CONTENT">
     <entity1> <Class rdf:about="http://www.hl7-v3.org#Adxp.streetAddressLine"></Class></entity1>
     <entity2><Class rdf:about="http://www.hl7-v2.org#AD.1.CONTENT"></Class></entity2>
     <measure>1.0</measure><relation>ClassMapping</relation>
   </Cell>
 </map>
<map>
   <Cell id="http://www.hl7-v3.org#Adxp.countryhttp://www.hl7-v2.org#AD.2.CONTENT">
     <entity1> <Class rdf:about="http://www.hl7-v3.org#Adxp.country"></Class></entity1>
     <entity2><Class rdf:about="http://www.hl7-v2.org#AD.2.CONTENT"></Class></entity2>
     <measure>1.0</measure><relation>ClassMapping</relation>
   </Cell>
</map>
<map>
   <Cell id="http://www.hl7-v3.org#Adxp.cityhttp://www.hl7-v2.org#AD.3.CONTENT">
     <entity1> <Class rdf:about="http://www.hl7-v3.org#Adxp.city"></Class></entity1>
     <entity2><Class rdf:about="http://www.hl7-v2.org#AD.3.CONTENT"></Class></entity2>
     <measure>1.0</measure><relation>ClassMapping</relation>
   </Cell>
</map>
</Alignment>
```

Figure 7.11: Aligning Version 2 - Version 3 AD Datatypes: AML Syntax

Similarly, in case of local messages they need to be lifted and aligned like the *coreSchemas*. Figure 7.12 shows a snippet of a lab test order schema (*i.e.*, Observation Order Complete (Test Results)) generated from HL7 local UML models.

Lines 1-10 present a *complexType* (ObservationRequest) containing sequence of elements needed to construct the lab test order message. Line 3 is an id element of type instance identifier (II) which contains object identifier (OID) as a value. In Version 3 each message, and their parts, are uniquely identified by an OID.

Line 9 shows an important attribute (*classCode*) with a type "ActObservation". As discussed

```
1  <xs:complexType name="ObservationRequest">
2   <xs:sequence>
3    <xs:element maxOccurs="unbounded" minOccurs="1" name="id" type="II"/>
4    <xs:element maxOccurs="unbounded" minOccurs="1" name="Specimen" type="SpecimenObservation"/>
5    <xs:element maxOccurs="1" minOccurs="1" name="recordTarget" type="Patient"/>
6    <xs:element maxOccurs="1" minOccurs="1" name="healthCareProviderObserver" type="Observer"/>
7    .....
8   </xs:sequence>
9    <xs:attribute default="OBS" name="classCode" type="ActObservation" use="optional"/>
10 </xs:complexType>
11 <xs:complexType name="SpecimenObservation">
12  <xs:sequence>
13   <xs:element maxOccurs="unbounded" minOccurs="1" name="id" type="II"/>
14   <xs:element maxOccurs="unbounded" minOccurs="1" name="specimen" type="DiabeticType2Observation"/>
15   .....
16  </xs:sequence>
17   <xs:attribute fixed="OBS" name="classCode" type="ActObservation" use="optional"/>
18 </xs:complexType>
19 <xs:complexType name="Observer">
20  <xs:sequence>
21   <xs:element maxOccurs="unbounded" minOccurs="1" name="id" type="II"/>
22   .....
23  </xs:sequence>
24   <xs:attribute fixed="PROV" name="classCode" type="RoleClass" use="optional"/>
25 </xs:complexType>
26 <xs:complexType name="Patient">
27  <xs:sequence>
28   <xs:element maxOccurs="unbounded" minOccurs="1" name="id" type="II"/>
29   <xs:element maxOccurs="1" minOccurs="1" name="name" type="EN"/>
30   <xs:element maxOccurs="1" minOccurs="1" name="administrativeGenderCode" type="CE"/>
31   <xs:element maxOccurs="1" minOccurs="1" name="birthTime" type="TS"/>
32   <xs:element maxOccurs="unbounded" minOccurs="1" name="addr" type="AD"/>
33   .....
34  </xs:sequence>
35   <xs:attribute fixed="PSN" name="classCode" type="EntityClass" use="optional"/>
36 </xs:complexType>
37 <xs:complexType name="DiabeticType2Observation">
38  <xs:sequence>
39   <xs:element maxOccurs="unbounded" minOccurs="1" name="id" type="II"/>
40   .....
41  </xs:sequence>
42   <xs:attribute fixed="SPCOBS" name="classCode" type="SpecimenObservation" use="optional"/>
43 </xs:complexType>
```

Figure 7.12: HL7 Version 3 (XSD1): Lab Test Order

in Chapter 5 Version 3 has the special mechanism to describe inheritance by using the *classCode* attribute. The *classCode* for a *complexType* describes its immediate super class, in other words, *complexType* (ObservationRequest) is a specialisation of "ActObservation" class defined in the vocabulary conceptual model.

HL7 uses domain specific elements and attributes to transform UML notions into XML structure. In Figure 7.12, we observe that each *complexTypes* (SpecimenObservation, Observer, Patient, DiabeticType2Observation) has a *classCode* attribute and a default value indicating a coded term for the immediate super class.

Line 32 shows the AD datatype for describing the patient home address. From an ontology design perspective, "ObservationRequest" is a core class inherited from "ActObservation" and with object properties (id, specimen, recordTarget, healthcareProviderObserver) describing several observation components. Similarly, Figure 7.13 shows an alternative schema for a lab order message. One major difference is the use of codes (*e.g.*, POOB_MT210000UV) for *complexType* names (*e.g.*, ObservationOrder.POOB_MT210000UV, Observer.POOB_MT210000UV). Codes describe the unique identification of a schema within the HL7 messaging space and it carries a specific meaning.

For example, ObservationOrder is a label for the POOB_MT210000UV message where POOB

```
1  <xs:complexType name="ObservationOrder.POOB_MT210000UV">
2   <xs:sequence>
3    <xs:element maxOccurs="unbounded" minOccurs="1" name="id" type="II"/>
4    <xs:element maxOccurs="unbounded" minOccurs="1" name="Specimen" type="HemoglobinObservation.POOB_MT210000UV"/>
5    <xs:element maxOccurs="1" minOccurs="1" name="recordTarget" type="Patient.POOB_MT210000UV"/>
6    <xs:element maxOccurs="1" minOccurs="1" name="healthCareProviderObserver" type="Observer.POOB_MT210000UV"/>
7    .....
8   </xs:sequence>
9    <xs:attribute default="OBS" name="classCode" type="ActObservation" use="optional"/>
10 </xs:complexType>
11 <xs:complexType name="HemoglobinObservation.POOB_MT210000UV">
12  <xs:sequence>
13   <xs:element maxOccurs="unbounded" minOccurs="1" name="id" type="II"/>
14    .....
15   </xs:sequence>
16    <xs:attribute fixed="SPCOBS" name="classCode" type="ActObservation" use="optional"/>
17 </xs:complexType>
18 <xs:complexType name="Observer.POOB_MT210000UV">
19  <xs:sequence>
20   <xs:element maxOccurs="unbounded" minOccurs="1" name="id" type="II"/>
21   .....
22  </xs:sequence>
23   <xs:attribute fixed="PROV" name="classCode" type="RoleClass" use="optional"/>
24 </xs:complexType>
25 <xs:complexType name="Patient.POOB_MT210000UV">
26  <xs:sequence>
27   <xs:element maxOccurs="unbounded" minOccurs="1" name="id" type="II"/>
28   <xs:element maxOccurs="1" minOccurs="1" name="name" type="EN"/>
29   <xs:element maxOccurs="1" minOccurs="1" name="administrativeGenderCode" type="CE"/>
30   <xs:element maxOccurs="1" minOccurs="1" name="birthTime" type="TS"/>
31   <xs:element maxOccurs="unbounded" minOccurs="1" name="addr" type="AD"/>
32   .....
33  </xs:sequence>
34 </xs:complexType>
```

Figure 7.13: HL7 Version 3 (XSD2): Lab Test Order

indicates Observation Order, MT stands for Message Type, the number shows the order of message exchanges, and UV is used to denote universal scope.

There are country specific codes as well, in that case, UV changes to GB for Great Britain. For instance, our experiences suggest that developers use local conventions to name *complexType*. For example, some use only labels, others use codes, and many use a combination of both such as ObservationOrder.POOB_MT210000UV. Another difference is the hierarchy of observation classes. Line 16 of Figure 7.13 defines HemoglobinObservation.POOB_MT210000UV as a direct specialisation of ActObservation, while in the Figure 7.12 there is a subclass relation between SpecimenObservation and DiabeticType2Observation classes. All diabetic tests require hemoglobin test. However, DiabeticType2Observation is a special test for Diabetic Type 2 patients. Its up to local practices and

```
concept ObservationRequest
subConceptOf ActObservation

concept SpecimenObservation
subConceptOf ActObservation

concept Observer
subConceptOf RoleClass

concept DiabeticType2Observation
subConceptOf SpecimenObservation
```

Figure 7.14: Lab observation (XSD1): WSML Syntax

```
Class: ObservationRequest
SubClassOf: ActObservation

Class: SpecimenObservation
SubClassOf: ActObservation

Class: Observer
SubClassOf: RoleClass

Class: DiabeticType2Observation
SubClassOf: SpecimenObservation
```

Figure 7.15: Lab observation (XSD1): OWL Syntax

```
concept ObservationOrder.POOB_MT210000UV
subConceptOf ActObservation

concept Observer.POOB_MT210000UV
subConceptOf RoleClass

concept HemoglobinObservation.POOB_MT210000UV
subConceptOf ActObservation
```

Figure 7.16: Lab observation (XSD2): WSML Syntax

```
Class: ObservationOrder.POOB_MT210000UV
SubClassOf: ActObservation

Class: Observer.POOB_MT210000UV
SubClassOf: RoleClass

Class: HemoglobinObservation.POOB_MT210000UV
SubClassOf: ActObservation
```

Figure 7.17: Lab observation (XSD2): OWL Syntax

```
axiom v2v3_1 definedBy (?x memberOf ObservationRequest_s
equivalent ?x memberOf
      ObservationOrder.POOB_MT210000UV_t)

axiom v2v3_2 definedBy (?x memberOf Observer_s
equivalent ?x memberOf Observer.POOB_MT210000UV_t)

concept DiabeticType2Observation_s
subConceptOf HemoglobinObservation_t
```

Figure 7.18: Vertical alignments: WSML Syntax

```
Class: ObservationRequest_s EquivalentTo:
      ObservationOrder.POOB_MT210000UV_t

Class: Observer_s EquivalentTo:
      Observer.POOB_MT210000UV_t

Class: DiabeticType2Observation_s SubClassOf:
      HemoglobinObservation.POOB_MT210000UV_t
```

Figure 7.19: Vertical alignments: OWL Syntax

clinicians how they prefer to model any observation.

Figure 7.14 and Figure 7.16 show snippets of WSML message ontologies transformed from the schema definitions in Figure 7.12 and Figure 7.13. Figure 7.15 and Figure 7.17 show the OWL equivalent of the WSML ontologies. Figure 7.14 and Figure 7.16 show that the DiabeticType2Observation concept is more specialised than HemoglobinObservation. However, a correspondence describing the relationship between the two concepts is required to exchange messages.

```
Step1: Local Ontology (LO) = Merging all concepts from source and target message ontologies;
Step2: If alignments are available then iterate Step3 to Step6;
Step3: If equivalentClass (Class_source, Class_target) then
       merge{Class_source, Class_target};
Step4: If equivalentProperty (Property_source, Property_target) then
       merge{Property_source, Property_target};
Step5: If subClassOf(Class_source, Class_target) then
       add subClassOf(Class_source, Class_target) to LO;
Step6: If subProperty(Property_source, Property_target) then
       add subProperty(Property_source, Property_target) to LO;
```

Figure 7.20: Layering: Six substeps for merging message ontologies

Figure 7.18 and Figure 7.19 show alignments between concepts from the message ontologies. Figure 7.20 presents the substeps required to merge semantically similar concepts and properties. Currently, the steps outlined in the listing are semi-automatic. In Figure 7.20 Step1 to Step6 can be achieved using general ontology management tools such as the Protégé plugin PROMPT-Suite [16]. We have the SPARQL Recipes to create ontology alignments (described in Chapter 6) shown in Figure 7.18 and Figure 7.19. When the steps of Figure 7.20 are applied to message ontologies and their alignments, then Figure 7.21 shows the resulting local ontology.

```
concept ObservationRequest
subConceptOf ActObservation

concept Observer
subConceptOf RoleClass

concept SpecimenObservation
subConceptOf ActObservation

concept HemoglobinObservation.POOB_MT210000UV
subConceptOf ActObservation

concept DiabeticType2Observation
subConceptOf SpecimenObservation
      HemoglobinObservation.POOB_MT210000UV
```

Figure 7.21: Merged Local Ontology: WSML syntax

```
Class: ObservationRequest
SubClassOf: ActObservation

Class: Observer
SubClassOf: RoleClass

Class: SpecimenObservation
SubClassOf: ActObservation

Class: HemoglobinObservation.POOB_MT210000UV
SubClassOf: ActObservation

Class: DiabeticType2Observation
SubClassOf: SpecimenObservation
HemoglobinObservation.POOB_MT210000UV
```

Figure 7.22: Merged Local Ontology: OWL syntax

We observe in Figure 7.21 that concepts ObservationRequest and ObservationOrder.POOB_MT210000UV are merged to a single concept ObservationRequest. Similarly, the concepts Observer and Observer.POOB_MT210000UV are merged to the Observer concept. In case of structural differences, concept DiabeticType2Observation specialises from concepts SpecimenObservation and HemoglobinTest, such that DiabeticType2Observation is represented at the appropriate level of the hierarchy. Local ontologies are mapped onto the global ontologies thus avoiding the potentially large number of bi-lateral local mappings.

7.5 PPEPR Run Time

As described above, the main role of the Adapter Framework is at Run-Time. The Adapter Framework provides interfaces for developing and invoking adapters. The adapter framework follows a design pattern similar to Java Connector Architecture (JCA)[1]. The Integration Scenario described in the Chapter 3 requires five messages to be exchanged between all three actors. We now describe the Lifting and Lowering tasks of the Adapter Framework.

7.5.1 Lifting Instances

Figure 7.23 shows an instance of the schema (of Figure 7.12) describing values the "Patient" element. According to the Version 3 specification, the "Patient" *complexType* element (line 1 of Figure 7.23) describes the name of the patient. The "nameuse" and "given" attributes values (L, Sean Murphy) specify the type of name format "L (Long Name)" and the patient name (Sean Murphy).

```
1 <Patient>
2 ......
3  <id root="2.16.840.1.113883.19.3.2409" extension="1−976−245" displayable="true"/>
4    <nameuse>L</nameuse>
5    <given>Sean Murphy</given>
6 ......
7 </Patient>
```

Figure 7.23: GUH Lab observation (HL7 Version 3): Observation Order Complete (Test Results) of the Figure 3.1

Figure 7.24 shows the XSLT lifting rules for transforming message instances to corresponding ontological instances, which refer to concepts and attributes described in the respective schema. Line 2 of Figure 7.24 takes an instance name (*e.g.*, patient) from the "Observation Order Complete (Test Results)" message and associates it with a concept (*e.g.*, EN_Patient) using the "memberOf" (similar to *rdf:type*) WSML keyword. Figure 7.25 shows the resulting snippet of the "Observation Order Complete (Test Results)" ontological instance.

```
1 <xsl:if test=" $domain_concept != '' and $domain_instance != '' ">
2    <instance name="{concat($nsURI,$domain_instance)}">
3      <memberOf>
4        <xsl:value−of select="concat($nsURI, $domain_concept)"/>
5      </memberOf>
6 <xsl:if test="$attribute1 != '' and $primitiveValue1 != '' ">
7      <attributeValue name="{concat($nsURI,$attribute1)}">
8        <xsl:choose>
9          <xsl:when test="$primitiveValue1 != '' ">
10           <value type="{$dataType1}">
11             <xsl:value−of select="$primitiveValue1"/>
12           </value>
13         </xsl:when>
14       </xsl:choose>
15     </attributeValue>
16 </xsl:if>
```

Figure 7.24: Lifting Rule: XML to WSML Instance

[1] http://java.sun.com/j2ee/connector/

The Adapter Framework keeps a log of all messages and lifting rules. Each message and lifting rule is uniquely identified as per the standard specification. An incoming message with a unique identification results in the automatic invocation of the responsible lifting rule.

```
1 <instance name="http://www.hl7−v3.org#patient">
2  <memberOf>http://www.hl7−v3.org#EN_Patient</memberOf>
3    <attributeValue name="http://www.hl7−v3.org#nameuse">
4     <value type="http://www.wsmo.org/wsml/wsml−syntax#string">L</value>
5    </attributeValue>
6    <attributeValue name="http://www.hl7−v3.org#given">
7      <value type="http://www.wsmo.org/wsml/wsml−syntax#string">Sean Murphy</value>
8    </attributeValue>
9 </instance>
```

Figure 7.25: WSML Instance: Observation Order Complete (Test Results)

7.5.2 Data Mediator

Figure 7.26 shows input and output of the WSMX Data Mediator. The Data Mediator takes as input, source and target ontologies, ontology alignments, a source message, and results a target message.

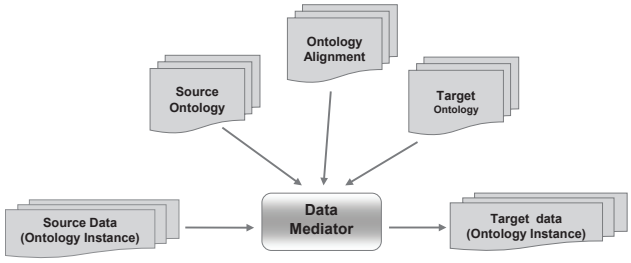

Figure 7.26: PPEPR Data Mediator

Figure 7.27 shows a snippet of alignments between Version 2 and Version 3 ontologies. Alignments are expressed in the Abstract Mapping Language (AML). As said, it is a formalism-neutral syntax for ontology alignments.

Line 3 and Line 6 in Figure 7.27 describe the source and target ontologies. Line 9 and Line 14 describe the matching elements. Line 12 and Line 17 describe the type of relation between them. In this case it is "AttributeMapping". When the Data Mediator takes the source and target ontologies (Version 2 and Version 3 ontologies developed in the previous chapters), ontology alignments (Figure 7.27), and a source instance (Figure 7.25) then results into a target instance (Figure 7.28).

In addition to instance transformation, the Data Mediator is also used for testing the correctness of the alignments between source and target ontologies. During design-time we have used the Data Mediator as a test module. Therefore, alignments are developed incrementally (*i.e.*, checking the correctness with each new set of alignments) instead of testing the complete alignments finally at the run-time instance transformation.

```
1  <Alignment><dc:identifier rdf:resource="HL7_order_map" />
2     <onto1><formalism name="WSML" uri="http://www.wsmo.org/wsml" />
3        <uri>http://www.hl7−v3.org#POLB_IN002120_Ontology</uri>
4     </onto1>
5     <onto2><formalism name="WSML" uri="http://www.wsmo.org/wsml" />
6        <uri>http://www.hl7−v2.org#OML_021_Ontology</uri>
7     </onto2>
8   <map>
9     <Cell id="http://www.hl7−v3.org#givenhttp://www.hl7−v2.org#XPN.2">
10      <entity1> <Attribute rdf:about="http://www.hl7−v3.org#given"></Attribute></entity1>
11      <entity2><Attribute rdf:about="http://www.hl7−v2.org#XPN.2"></Attribute></entity2>
12      <measure>1.0</measure><relation>AttributeMapping</relation>
13    </Cell>
14    <Cell id="http://www.hl7−v3.org#nameusehttp://www.hl7−v2.org#XPN.7">
15      <entity1> <Attribute rdf:about="http://www.hl7−v3.org#nameuse"></Attribute></entity1>
16      <entity2><Attribute rdf:about="http://www.hl7−v2.org#XPN.7"></Attribute></entity2>
17      <measure>1.0</measure><relation>AttributeMapping</relation>
18    </Cell>
19  </map>
20 </Alignment>
```

Figure 7.27: Alignment Input to Data Mediator: Abstract Mapping Language (AML)

7.5.3 Lowering Instances

Figure 7.28 is a Version 2 equivalent of the Version 3 "Observation Order Complete (Test Results)" instance. As described, "Observation Order Complete (Test Results)" describes patient details with an observation of the patient medical condition. The "PID" element of the Version 2 specification is used to describe the patient identification, where "PID.5" (line 1 of Figure 7.28) element describes the patient name and the type of name format.

```
1 <instance name="http://www.hl7−v2.org/PID.5">
2    <memberOf>"http://www.hl7−v2.org#PID_5_Content"</memberOf>
3       <attributeValue name="http://www.hl7−v2.org#XPN.2">
4         <value type="http://www.wsmo.org/wsml/wsml−syntax#string">Sean Murphy</value>
5       </attributeValue>
6       <attributeValue name="http://www.hl7−v2.org#XPN.7">
7          <value type="http://www.wsmo.org/wsml/wsml−syntax#string">L</value>
8       </attributeValue>
9 </instance>
```

Figure 7.28: WSML Instance (Version 2): Observation Order Complete (Test Results)

"PID_5_Content" is the conceptual reference for the "PID.5" instance. The attributes "XPN.2" and "XPN.7" describe the patient name (Sean Murphy) and the name format (L for long name). Figure 7.29 shows the lowering rules, where lines 2, 4 and 9 describe XML elements from the ontological instances. Figure 7.30 shows the resulting snippet of the Version 2 "Observation Order Complete (Test Results)".

As described in Chapter 5, a major hurdle in the transformation between an XML (tree-based structure) and an ontology (graph-based structure) is the preservation of the ordering of XML elements. In the case of lifting (tree-based structure to graph-based structure) this issue is trivial as graphs are not restricted to the ordering of ontology elements. This problem is apparent during lowering of ontology instances as the ordering information is lost during the lifting process. Thankfully, HL7 specifications use numbering of elements in their names (*e.g.*, MSH.1, MSH.1.1, PID.5) and they appear in a certain defined order within messages. Therefore, we have used ontology instance names (*e.g.*, MSH.1, MSH.1.1, PID.5) to describe the order of XML elements while constructing the

```
<xsl:if test="$XPN_2_PID_5 != '' or $XPN_7_PID_5 != ''">
 <xsl:element name="PID.5">
   <xsl:if test="$XPN_2_PID_5 != '' ">
     <xsl:element name="XPN.2">
  <xsl:value-of select="$XPN_2_PID_5"/>
     </xsl:element>
    </xsl:if>
   <xsl:if test="$XPN_7_PID_5 != '' ">
     <xsl:element name="XPN.7">
      <xsl:value-of select="$XPN_7_PID_5"/>
     </xsl:element>
   </xsl:if>
 </xsl:element>
</xsl:if>
```

Figure 7.29: Lowering Rule: Observation Order Fulfilment Request

```
<OML_O21>
  <MSH>
  ....
  </MSH>

  <PID>
   ...
   <PID.5>
     <XPN.2>Sean Murphy</XPN.2>
     <XPN.7>L</XPN.7>
   </PID.5>
   ...
  <PID>
<OML_O21>
```

Figure 7.30: Lab Observation Message: Observation Order Fulfilment Request

lowering XSLTs.

7.6 Adapter Life Cycle

As described above, the Adapter Framework is a container for application specific adapters. Each application specific adapter within the framework has a lifecycle, which makes this framework flexible to accommodate factors like future changes, upgrades, and the maintenance of the PPEPR system, *e.g.*, if any changes are required in a deployed adapter, the first step would be to undeploy, second reconstruct and then redeploy to continue to send and receive messages with the improved version of the adapter. Operations within the adapter life cycle are divided into four main parts.

- **Deploy adapter:** The deployment of an adapter within the framework is simple. The framework provides a method, *deployAdapter* that takes the adapter name and the source of the adapter to deploy to the adapter framework and returns a security key. Once deployed, the adapter is stored in the adapter pool of the framework. The security key is required to *undeploy* the previously deployed adapter and keep track of the adapter identity within the framework. Therefore applications can invoke only those adapters that are identified with the security key, which makes the overall framework secure from unidentified or malicious applications that may try to transform or change sensitive data within a healthcare organisation.
- **Send message:** EHRs can invoke and send a message by sending the adapter name, message, and security key of a particular adapter to the adapter framework. When the framework

receives this request, the communication type is scanned (synchronous or asynchronous), the security key and message are verified by the Validator component. After verification, the framework generates an internal request for invoking appropriate adapter (*e.g.*, XML2WSML, XML2OWL, WSML2XML) from the adapter pool. The adapter performs the message transformation and forwards the transformed message to the WSMX Data Mediator.

- **Receive message:** Once the adapter framework receives the mediated message from the Data Mediator it forwards the message to a specific adapter for transformation.
- **Undeploy adapter:** Adapters can be undeployed by sending a request to the adapter framework. The request message consists of the adapter name and the security key attached to it during deployment.

The Adapter Framework provides interfaces to the Data Mediator. Every incoming or outgoing messages from the Data Mediator is transformed via the Adapter Framework.

7.7 PPEPR in Action

This section demonstrates the PPEPR[2] system showing how it is used in a real environment. Figure 7.31 shows the General Practitioner (GP) EPR system which sends the request (for a lab report) to GUH laboratory. This corresponds to the first message "Observation Order Fulfilment Request" of the lab observation use case (Figure 3.1).

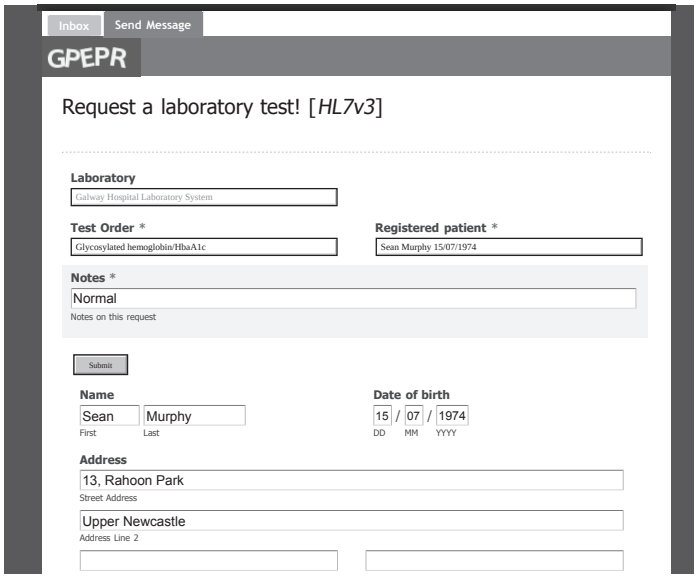

Figure 7.31: Lab Observation Request

The request form shown in Figure 7.31 describes the laboratory responsible for receiving the request (multiple laboratory options could be available to GP), patient details (*e.g.*, name, address, date of birth), with the type of test (*e.g.*, Glycosylated hemoglobin/HbaA1C). The GP system generally keeps a log of all incoming (inbox) and outgoing lab messages.

[2]http://www.ppepr.com/epr/ppepr/monitor/

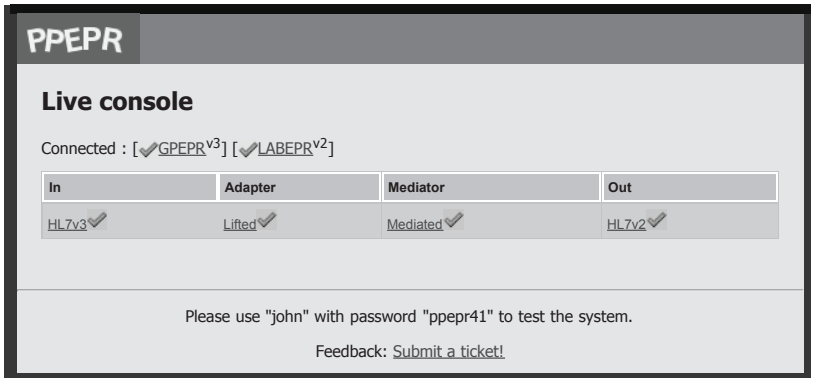

Figure 7.32: PPEPR Live Console

Figure 7.32 shows the PPEPR Live Console which keeps track of all the incoming and outgoing messages within the PPEPR system. This includes: (i) types of message (Version 2 or Version 3), and (ii) message format after the lifting or lowering process and the results of the Data Mediator for each transformation. Figure 7.32 also shows the connected clinical entities (GP and Lab) for this particular interaction. Keeping log of all exchanged messages helps audit and maintain the PPEPR system.

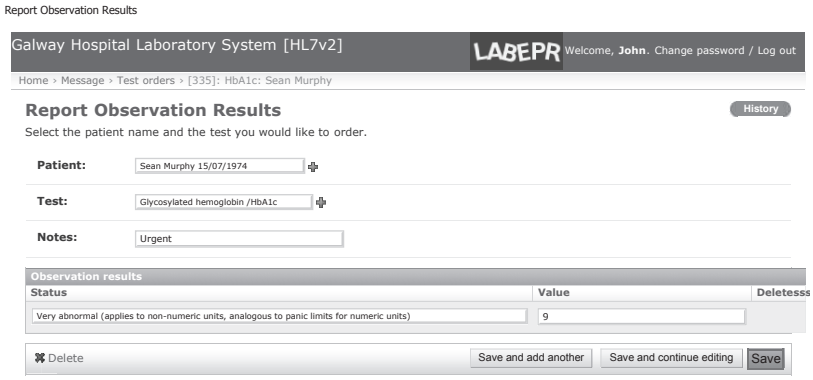

Figure 7.33: GUH Laboratory Test Report

Figure 7.34 shows the lab observation form (GUH laboratory). Lab clinicians perform the lab observation and report on the clinical parameters requested. We observe the original form (from GP) contains the note as "Normal", while lab observation mentions it "Urgent" based on the HbaA1C level measured for Sean Murphy. Similarly, the HbaA1C finding value (9 point) and the status (Abnormal) is reported back to the GP system.

7.8 Evaluation

PPEPR is evaluated to measure the integration burden between heterogenous healthcare applications. The integration burden is measured across three dimensions: (i) design-time development effort measured (per message), (ii) run-time efficiency of exchanging messages, and (iii) deployment methodology comparing traditional software deployment.

Figure 7.34 shows the time consumed working on two separate messages within HL7 Version 2 and

Activity	Time Spent (Message 1 - Days) OUL-21	Time Spent (Message 2 - Days) POLB_IN003130	Current Time
1. V2 XSD->OWL	Automated	Automated	N/A
2. V2 Manual Modification of OWL	**2**	**1**	N/A
3. V2 OWL->WSML	Automated	Automated	N/A
4. V3 XSD->OWL	Automated	Automated	N/A
5. V3 Manual Modification of OWL	**2**	**1**	N/A
6. V3 OWL->WSML	Automated	Automated	N/A
7. Discovery of Mapping Specifications	**3**	**1**	N/A
8. Create Mapping/Checking Accuracy	**1**	**1**	**0.5**
9. XSLT for Lifting (Instances)	**2**	**1**	N/A
10. XSLT for Lowering (Instances)	**2**	**1**	N/A
Totals	**12 (4)Days**	**6 (2) Days**	**0.5 (0.5) Days**

Automated
From XSD -> OWL\WSML

Figure 7.34: PPEPR Evaluation

HL7 Version 3. The figures represent the time taken by the development team, familiar with the HL7 standards, and with the HL7 development framework. The measurement is based on development-recorded observations. The "Totals" (the last row of Figure 7.34) shows the time spent during the construction of the ontological models and their alignments (within brackets). For example, the second column (Figure 7.34) "Totals" is 12(4), which says 12 days were required for the ontology development (per message) and 4 days for their alignments. One day include 8 working hours.

The following parameters are used to measure the design-time and run-time integration efforts comparing existing HL7 integration solution such as Mirth (see Section 2.4):

1. **Design-Time**

   (a) *Modelling HL7 message:* Currently, the time taken for modelling HL7 ontologies, creating transformation rules (*e.g.*, XSLT), and mapping definitions takes 0.5 days, where the tasks of lifting/lowering and alignment are automated. Initially (*i.e.*, message 1 Figure 7.34) the development of ontologies from the local schemas took 4 days (*i.e.*, Activity 2 and 5 of the second column in Figure 7.34). Currently the lifting process is completely automated. A typical HL7 engine takes 3-4 days (discovery of mappings plus creating mapping specification) to align two HL7 messages (*i.e.*, Activity 7 and 8 of the second column in Figure 7.34). Currently, the alignment process is semi-automated and reduced to 0.5 days (*i.e.*, 4 hours per message), which is required to check the correctness of the resulting alignments via SPARQL recipes (see Chapter 6). This measurement is based on developers-recorded observations with a good level of knowledge in HL7 and semantic technology tools. An XSD schema for a HL7 message normally contains 90 *complexTypes* and 50 *elements* or *attributes*. That means, 140 mapping rules are required to align two heterogenous messages of a same type. Consequently, integration system like Mirth would create 1400 mappings rules for 10 HL7 messages and their counterparts. Similarly, a local ontology (without involving global ontologies) generated from an XSD schema includes 90 concepts and 50 properties, and mappings result in 140 mapping rules. The number of mapping rules grows significantly with each heterogenous message exchanged between interacting HL7 applications. Including a global reference ontology reduces the local mappings (*i.e.*, bilateral correspondences) to 73 mapping rules. A significant reduction in the time required to model and align the second message can be seen in the third column of

Figure 7.34. With the top down methodology (*i.e.*, including a global reference ontology) the number of bilateral correspondences reduced by 50 percent.

(b) *Syntactic vs. Semantic Mapping:* Syntactic mapping is predominantly based around the XML level of expressivity. Due to the inherent nature of XML, mappings are more at an implementation level which causes a significant increase in the number of mappings. In PPEPR, mappings are at the semantic (ontological) level which by nature maps two equivalent elements (concepts) at a higher level. The results have shown that the number of mappings are reduced (compared to existing integration systems) by 50 percent–PPEPR's major achievement.

2. **Run-Time**

(a) *Execution-time:* The total message exchange time (message transformation, mediation and transmission) measured between two HL7 applications on a typical broad-band connection is 1.5 seconds. This is comparably better than the existing integration systems.

(b) *Transformation:* Each message transformation is validated against the standard schema. The purpose of this validation is to ensure that the transformation (lifting/lowering) process is not losing the original message content.

(c) *Stability:* In the last 2 years more than a thousand messages have been exchanged on the PPEPR system with 100 percent success rate. These measurements are based on an in-house evaluation of the PPEPR framework and use of PPEPR within the SQWELCH environment [171]. SQWELCH is a widget-based integration platform allowing creation and integration of clinical widgets.

PPEPR's deployment methodology can be compared to the "traditional" route of software development as:

- Once messages are modelled and alignments are developed in PPEPR, that work never has to be done again for the standards under consideration. All that needs to be done at each location is to configure the various EHRs to point at PPEPR as an intermediary step. Contrast that with a traditional software development (or integration tools), which will have to be modified for each application used in a specific environment.

- PPEPR uses declarative methods to model the messages and the mappings between them. This implies that the testing of the artifacts produced by PPEPR will be easier and implementation more immediate than a syntactic software program.

- Since PPEPR holds mappings at a standard specification level rather than an implementation level, changes will be less frequent. Even as standards do evolve, given that PPEPR is using declarative methods, the changes will be easier to implement.

The evaluation result confirms achieving the two main objectives of our hypothesis, *i.e.*, PPEPR (an ontology-based integration framework) reduced: (i) the integration burden between heterogeneous healthcare applications; and (ii) the number of alignments between heterogeneous healthcare applications. Including global ontologies and their alignments (which is performed only once for

the standard in consideration) limited the alignment task for local heterogeneities. Consequently, the number of alignments required for each heterogenous message is now reduced to 50 percent and total time invested for the alignment is reduced to 0.5 days comparing 4 days needed by existing integration solutions. Further, the automated methods proposed in this thesis (*i.e.*, lifting and alignment mechanisms) have significantly reduced the involvement of developers during ontologising and aligning the HL7 applications. Finally, the run-time performance (*i.e.*, 1.5 seconds per message transformation) is comparably better than existing integration solutions. PPEPR evaluation has been conducted over 5 years by improving each stage of using semantics within the overall integration process.

7.9 Summary

This chapter presented the healthcare integration platform PPEPR build on the PPEPR Methodology proposed in the Chapter 4. The PPEPR platform connects healthcare institutions complying with HL7 Version 2 and Version 3 specifications. The chapter provides details on the functioning of the Adapter Framework and Data Mediator components with the overall integration framework. The chapter shows an example demo based on the lab observation integration scenario presented in the Chapter 3. Finally, the chapter presents an evaluation, measuring PPEPR on three dimensions that shows PPEPR strengths over existing healthcare integration solutions.

Chapter 8

Context & Modularity

8.1 Introduction

We have shown in previous chapters how healthcare standards are now used in, *e.g.*, hospitals for representing information models, clinical repositories, ontologies for terminologies and vocabularies, and patient records. Although standards have improved the management of very complex data, they show their limits when integrating and exchanging data between different systems. The PPEPR framework uses standard ontologies to ease the integration and mediation of HL7 messages. However, PPEPR has revealed certain limitations of ontological frameworks in dealing with local constraints such as policies *i.e.*, the fourth interoperability requirement of the Integration Scenario (see Section 3.4). High level ontologies that are shared by all systems cannot describe all possible subdomains that may be needed by local clinics. Therefore, they must extend common knowledge with domain ontologies, as well as define internal policies, rules and vocabularies that are specific to their context. Even globally shared knowledge may be used differently according to the context. Interestingly, the very same problems have been identified on the Semantic Web, where the notion of context and heterogeneity have not been sufficiently tackled in existing standards (viz., RDF and OWL) [172].

We discussed in the previous chapters that Ontology Based Information Systems (OBISs) received attention for their flexibility and automation in maintaining, updating, sharing schemas and underlying data. This strength is due to the fact that schemas and data are loosely-coupled (as compared to cohesively-weak schema and data in traditional databases) and both are represented and integrated in a logical fashion. However, current Semantic Web technologies have limitations in providing a unified framework for all the varieties of applications and subdomains of Healthcare and Life Sciences (HCLS). Semantics and the reasoning mechanisms behind ontological knowledge bases are centralised in the way data and schemas are accumulated and processed. Therefore, when an OBIS experiences use-cases where data, schema, and applications are heterogenous and distributed then that impedes the expected integration results. This limitation has roots in the formalism and corresponding reasoning mechanism underlying ontology-based knowledge bases. Ontologies are good in describing general invariant concepts and mappings or relation among those concepts. When ontologies are applied for information systems, then they describe information which is at-

tached to multiple parameters, for example, information that is local and specific to some domain, time-dependent, and policies like constraints that are tied to context specific entities (*e.g.*, patient, physician, hospitals).

This chapter surveys formal approaches (logical formalisms as well as other theoretical framework) and assess their adequacy towards managing context and heterogeneity over distributed HCLS systems. We show the implications of each approach on the integration scenario presented in Chapter 3, which realistically represents a plausible scenario in the HCLS domain. While other related formal comparisons exist in the literature [173, 174, 175], we are not aware of an existing survey that presents and analyses as many formal approaches and apparently no other such survey concretely illustrates the consequences of using different techniques.

Our contribution, in addition to the analysing of the state of the art, is an original classification of the approaches according to requirements identified from the study of concrete HCLS systems and the goal of the HCLS task force[1]. Formalisms are classified according to the following requirements:

- Ability to identify and use context (*i.e.*, "context-awareness"). This enables systems to distinguish between local and global KBs.
- Ability to process internal policies or profiles distinctively.
- Ability to modularise ontologies (reusing multiple ontologies or parts of ontologies).
- Ability to relate heterogeneous knowledge, either within or between contexts (in particular via more or less expressive *ontology alignments*).
- Robustness with respect to heterogeneity (*i.e.*, ability to tolerate incompatibilities or inconsistencies within or between contexts).

We will show that there is a continuum between strongly and loosely connected knowledge. Strong connection (*e.g.*, ontology axioms) easily leads to many inconsistencies but enhances cross-context inferences, while loose connection avoids inconsistencies but decreases knowledge transfer between contexts. We finally show how to combine several approaches to satisfy more requirements. The features described above are based on our experiences in developing and maintaining the PPEPR framework.

We start the chapter with highlighting the importance of context-awareness for our integration scenario (Chapter 3). We show how to apply various contextual formalisms to this scenario in four sections overviewing the state of the art: first, we present two general theories of reasoning with context (Section 8.4.1); second, we detail some instantiation of one of the model of context (Section 8.4.2 and Section 8.4.3); third, we present more concrete formalisms for the description of context on the Semantic Web (Section 8.4.4); fourth, we provide other formal approaches built on top of semantic technologies that deals with the identified problems of our scenario (Section 8.4.5). After this extensive state of the art, we provide a summary and analysis of the studied approaches (Section 8.5). Finally, we present a solution path that combines existing approaches to better deal with the issues of context and modularity enabling treatment of context-specific local policies (Section 8.6).

[1] `http://esw.w3.org/topic/HCLS/ClinicalObservationsInteroperability`

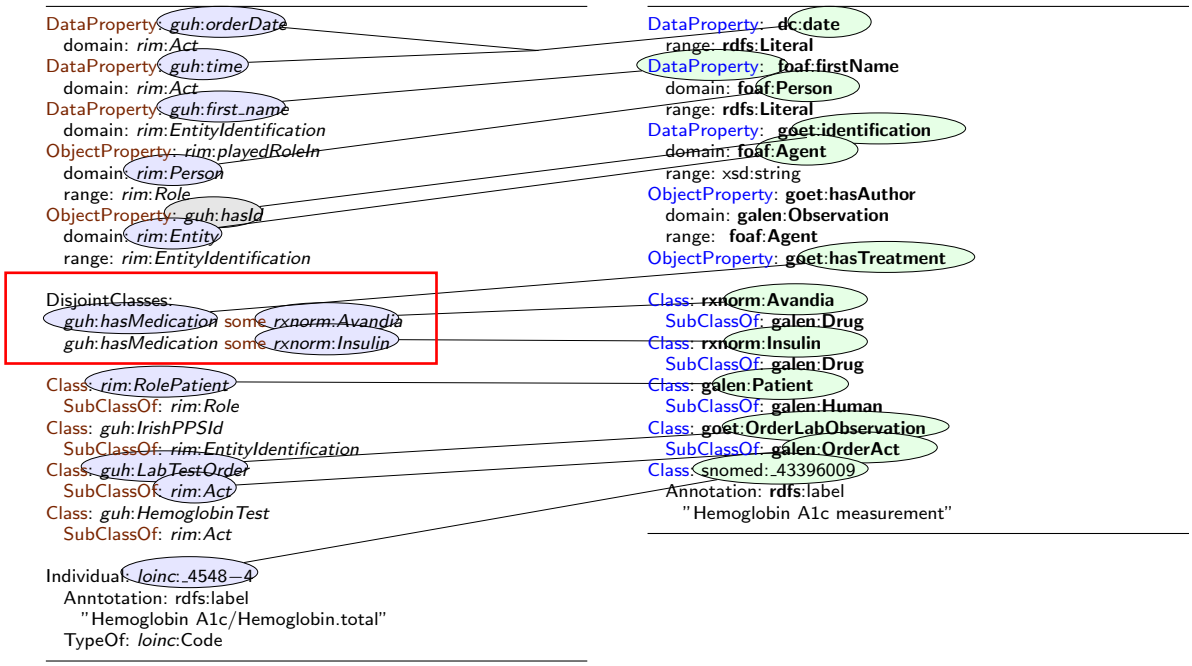

Figure 8.1: Extract of GUH (left) and GOET (Right) ontologies. Correspondences are informally represented by dotted lines. The rectangle shows the local policy axiom.

8.2 Integration Scenario And Context-Awareness

In the scenario of Chapter 3, GUH and GOET lab systems are Semantic-Web-enabled and are able to send and receive patient's records electronically. Figure 8.1 shows a snippet of the ontologies used in GUH (left) and GOET (right). Figure 8.2 shows the data equivalent to Table 3.1 described semantically, *i.e.*, RDF triples. GUH is using ontologies that are strongly influenced by HL7 standard while GOET is using more Semantic Web oriented ontologies (*e.g.*, *GALEN*, *FOAF*) in addition to the HL7 standard, which better ensures interoperability with other Linked Data (*e.g.*, other administrative systems may take advantage of *FOAF* data). GUH and GOET model medical, clinical and patient knowledge using ontologies, while reusing well established ontologies such as *SNOMED* and *GALEN*. They extend them to represent local domain knowledge and internal policies.

GUH and GOET model the same domain of knowledge in a common format (RDF and OWL). These systems define medical knowledge according to two different contexts in a heterogeneous way. There are two levels of contextual heterogeneity:

- *intra-contextual heterogeneity*, which occurs within a context. As seen in the example, several vocabularies are used for describing different types of information (*e.g.*, *RxNorm* [47] for drugs, LOINC for types of tests). Multiple terminologies must be integrated in a modular way. Local systems also have to deal with knowledge of differing natures, *e.g.*, axioms about a domain of

```
:sean a rim:Person ;
  rim:playedRoleIn [a rim:RolePatient] ;
  guh:hasID :678970W .
:678970W a guh:IrishPPSId ;
  rdfs:label "Sean Murphy's Irish PPS number" ;
  guh:first_name "Sean" ;
  guh:last_name "Murphy" ;
  guh:hasMedication [a rxnorm:Insulin] .
:paul a rim:Person ;
  rim:playedRoleIn [a rim:RolePhysician] ;
  guh:hasID :68374532 .
:68374532 a guh:PID ;
  rdfs:label "Paul Smith's professional Id" ;
:6560−89 a guh:LabTestOrder ;
  guh:orderedBy :paul ;
  guh:hasPatient :sean ;
  guh:orderDate "2007−07−16" ;
  guh:time "13:07:07" ;
  guh:orders :Test743890 .
:Test743890 a guh:HemoglobinTest ;
  rim:effectiveTime "M0717173034" ;
  guh:specimen :7854698 ;
  rim:measures loinc:_4548−4 .
```

```
:345678IE a galen:Patient ;
  goet:identification "Irish Driving licence" ;
  foaf:family_name "Murphy" ;
  foaf:firstName "Sean" ;
  goet:hasTreatment [a rxnorm:Avandia] .
:6837−4532 a galen:Doctor ;
  goet:identification "Professional Id" ;
  foaf:family_name "Roth" ;
  foaf:firstName "Gustav" .
:7779−78 a goet:OrderLabObservation ;
  goet:hasAuthor :6837−4532 ;
  goet:hasPatient :345678IE ;
  galen:orders :Test777767 .
:Test777767 a galen:BloodSugarTest ;
  dc:date "2008−12−22T00:30:00" ;
  goet:hasCode [a snomed:_43396009] ;
  goet:hasSpecimen :89756 .
:s89756 a galen:Specimen .
```

Figure 8.2: Extract of Lab-Test orders at GUH (left) and GOET (right).

discourse (*e.g.*, an hemoglobin test is a kind of act) and internal policy (a patient cannot be treated with both Insulin and Avandia).

- *inter-contextual heterogeneity* occurs between contexts. If the terminologies are different, the systems cannot interoperate unless some relations are asserted between the domain ontologies or a translation mechanism exists. Correspondences between local ontologies are informally presented in Figure 8.1 using dotted lines. Besides, corresponding concepts of distinct ontologies can be modelled in very different ways. For instance, GUH here uses an object property for patient's identification, while GOET is using a datatype. Thus, systems should be able to tolerate such heterogeneity. Finally, we can see that identifying context is crucial, notably to see which policy belongs to which context.

The goal of this thesis is not to establish a universal definition for context but to investigate and examine Semantic Web relevant formal approaches (logical formalisms as well as other theoretical framework) for their ability to manage context and heterogeneity over distributed Health Care and Life Science (HCLS) systems. A natural choice to identify a context on the Web is to use the fundamental entity of the Web, namely, URIs. However, specialised mechanisms are required to represent, reason, and exchange contextual knowledge in conjunction with globally shared knowledge bases.

Outline of the state of the art. The following sections detail the existing formal approaches that were devised to deal with the problem of heterogeneity in multi-contextual knowledge systems. In Section 8.4.1, we describe foundational work on modelling context, especially focusing on McCarthy's and Guha's approach on the one hand, and Giunchiglia *et al.* on the other hand. The latter approach has been more successful in its adoption and has led to several instantiations of the approach, that we discuss in Section 8.4.2. While these formalisms essentially focus on reasoning across contexts

or modules, they offer a limited infrastructure for "reifying" the notion of context, describing it and constraining it. These aspects are better covered by the development of Guha *et al.*'s work that we present in Section 8.4.4. Finally, we show that other essential concerns of interoperable HCLS systems, such as modelling constraints, policies, rules, are also tackled by proposed extensions of semantic formalisms that are quite independent of the handling of heterogeneity and context.

8.3 Context - Formalisms & Reasoning

The notion of context has a very long history within several research communities, leading to vast studies about how to define a context, how to take into account information coming from the context, how to contextualise knowledge and information, etc. Despite all this study, a clear and unifying definition of the notion of context is still missing. Probably the most simple and concrete definition of context comes from linguistics. WordNet defines a linguistic context as the "discourse that surrounds a language unit and helps to determine its interpretation"[2]. [176] considers that a sentence in a knowledge base is context dependent if its meaningfulness and its truth rely on some assumptions. Making explicit the context dependencies, *i.e.*, reifying the context, corresponds to making explicit these assumptions. [177] defines context-aware computing as "Context is any information that can be used to characterise the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and application themselves". Judging by the many definitions of context in different disciplines, the notion of context is itself context-sensitive, and it is hard to point out the specific characteristic that distinguishes context from background, background knowledge and/or the multiplicity of implicit facts and assumptions that are simply taken for granted, unnoticed, left out, or suppressed as too obvious to mention. This is reflected in some important papers on context to actually define it, such as John McCarthy's [178].

8.4 State of the Art

8.4.1 State of the Art-1 : Context Formalisms

Logic for Context

In artificial intelligence, the notion of context generally concerns the representation and use of information. The notion is used to account for phenomena such as the context of validity of information [179] and the efficiency of reasoning in narrower contexts [176]. An influential attempt to formalise context dependency was made by John McCarthy [178]. The idea was to mark the dependency of propositions on context and track this dependency through changes of context and stages of reasoning. The basic step was to move from a (simple) proposition to the (meta) proposition that the proposition in question is true in a context. The syntax for such meta-propositions was `ist(c,p)` for proposition `p` and context `c`. Contexts thus entered the theory as objects, enabling the theory to express changes of context and the effects of such changes on propositions. Changes of context

[2] http://wordnet.princeton.edu/perl/webwn?s=context

typically involve making or dropping assumptions, so the syntax also allows compound, functional terms for contexts such as `assuming(p, c)` (the context obtained from `c` by assuming `p`). With such terms, the theory can cover logically interesting consequences of context change, expressed as `ist(c, p)` $\Rightarrow$ `ist(d, q)` and similar formulas, called lifting axioms. McCarthy's ideas were later developed by others, and found their way into a working AI system (Cyc, in the form of micro-theories applied for medical domain [180]). McCarthy's work made contexts as first class objects in the domain that can be quantified, be the range of functions, and so on. All this allows one to write very expressive formulae that lift axioms from one context to another. While this is very convenient, it also makes it quite difficult to provide an adequate model theory and extremely difficult to compute with [173].

Contextual Reasoning - Local Model Semantics

In the early nineties, a different line of thought (compared to McCarthy's and Guha's works on context) was proposed—Local Model Semantics (LMS)—by Fausto Giunchiglia [181, 13], who argued that people do not use all of the knowledge in an attempt to solve a problem. Instead, they construct a "local theory" where each independent theory (or context) is related to some particular domain knowledge. Giunchiglia's formalisation of context is motivated by the problem of locality where the reasoning process uses only a subset of the world knowledge. Therefore, in LMS, context is a theory of the world that encodes an agent's perspective of it and that subset is used during a given reasoning process. While reasoning, the user can switch from one context to another in case the original context is not adequate to solve the problem. Giunchiglia's approach is more towards formalising contextual reasoning than McCarthy's work of formalising context as a first class object.

Serafini *et al.* then proposed refinements of LMS to concretely realise a distributed reasoning framework. In Section 8.4.2, we discuss those refinements, namely Distributed Description Logics (DDL) [95] and a contextualised form of OWL, called C-OWL [182]. For instance, DDL is an underlying formalism behind the Contextualized Web Ontology Language (C-OWL) that allows separate ontological models for a domain and correspondences between entities of different models are formalised via domain relations which are in turn used to interpret the bridge rules[3]. Reasoning will then be carried out locally with respect to a single context and is shared only via explicitly specified connections, following the principle of locality and compatibility. Same authors [17] have attempted to align and map medical ontologies (*e.g.*, *GALEN*, *UMLS* [84], *TAMBIS* [183] using three proposed features of contextual ontologies, namely, directionality of information, local domain, and context based ontology mappings.

8.4.2 State of the Art-2 : Context and Constraints for OBIS

Standard Approach - OWL

The representation and reasoning of contextual knowledge is outside the scope of OWL semantics [184, 185]. When reasoning is performed across different ontologies, then these ontologies share a single and global interpretation domain. The mappings between two ontologies become part of the overall model which is interpreted globally. In OWL, the ability to combine ontologies is restricted

[3]bridge rules are cross-ontology assertions in DDL

to the import of the complete ontology and to the use of the imported elements by direct reference. Thus, a set of local ontologies is globalised in a unique shared model, via the import mechanism. OWL also supports mapping constructs that implicitly exist in terms of the mutual use of statements across ontologies (*e.g.*, *subClass*, *sameAs*, *equivalentClass*, *equivalentProperty*, *differentFrom*, and *AllDifferent* axioms).

Although we have already raised some doubts in the informal discussion above on the suitability of current Semantic Web technologies with respect to heterogeneity and context, we can explore what OWL can offer to overcome these problems. This language partially addresses modularity and expressing correspondences between various ontologies.

Integration Scenario: OWL Solution

In the integration scenario, several well identified terminologies are reused. OWL provides an import feature thanks to the property `owl:imports` which helps to modularly integrate several domain ontologies. By stating that the GUH ontology imports RIM and LOINC, the axioms of these ontologies are automatically made part of the GUH ontology.

In the integration scenario, concepts and properties of the two hospitals are modelled differently, but correspondences can be identified and expressed in OWL. In Listing 8.1, we present the axioms that should be added to make the two systems interoperate. Notice that these mappings relating terms in two different contexts cannot be distinguished from mappings between terms of imported ontologies within one context.

```
Class: ( foaf:Person ) SubClassOf: ( rim:Entity )
Class: ( galen:OrderAct ) SubClassOf: ( rim:Act )
Class: ( rim:playedRoleIn some rim:RolePatient ) EquivalentTo: ( galen:Patient )
Class: ( guh:LabTestOrder ) EquivalentTo: ( goet:OrderLabObservation )
Class: ( guh:HemoglobinTest and (rim:measures some loinc:_4548−4) )
    EquivalentTo: ( galen:BloodSugarTest and (goet:hasCode some {snomed:_43396009}) )

EquivalentProperties: ( guh:first_name ) ( foaf:firstName )
EquivalentProperties: ( guh:hasMedication ) ( goet:hasTreatment )

Instance: ( guh:sean ) sameAs: ( goet:345678IE )
```

Listing 8.1: Extract of OWL supported mapping definitions

This approach is the only possible way of dealing with heterogeneity which fully complies with established Semantic Web standards. However, these standards are clearly not enough to solve the important issues presented in Section 8.2.

Integration Scenario: OWL Limitations

First, while a form of modularity is offered by OWL, its import statement can only handle the reuse of full ontologies without being able to specify subparts of them. This is particularly problematic with large ontologies like *SNOMED*, which have to be fully integrated, even when only a small subdomain is needed.

Second, not all relevant mappings can be expressed in OWL. For example, (i) the *ObjectProperty* `guh:hasId` and the *DatatypeProperty* `goet:identification` are designed for a similar purpose (identify the person) but OWL semantics does not allow to map between *ObjectProperty* and

DatatypeProperty; (ii) OWL does not support operations on attributes, *e.g.*, the concatenation of two *DatatypeProperties* (*e.g.*, `guh:orderDate`, `guh:time`) into a single *DatatypeProperty* (*e.g.*, `dc:date`). Other examples include unit or currency conversion.

Third, OWL does not include any feature for distinguishing between universal facts (*e.g.*, a patient is a person) and local policy or profile (*e.g.*, people should not be treated with both Insulin and Avandia). Additionally, OWL does not permit identifying the context of an axiom or term. The implication of these two limitations is that policies have to be represented as DL axioms, and these axioms are affecting all contexts identically. In our scenario, according to GUH, Sean is treated with Insulin. When he goes to GOET, the record indicates that he has been given Avandia. Thanks to the aforementioned mappings, the terms *hasMedication* and *hasTreatment* can be interpreted interchangeably, so that GOET can understand GUH record automatically. But it leads to a contradiction with the GUH policy because Sean has now both treatments. Yet, it should not be the case because GOET does not have this policy, and therefore should not detect an inconsistency. Note that undesired interactions can be reduced by using subsumption instead of equivalence in mappings, but the problem remains.

Fourth, OWL is not tolerant to diverging modelling of a knowledge domain. Different points of view can equally well describe a domain of interest, while being partially incompatible. Interpreting all axioms and assertions as equally true, in all contexts, may easily lead to inconsistency or nonsensical entailments.

Distributed Description Logic (DDL)

Distributed Description Logics (DDL) [95] is a formalism which was developped to formalise contextual reasoning with Description Logic ontologies. Indices $i \in I$ are used to determine from which context an ontology or an axiom comes from. Given, for instance, an axiom $C \sqsubseteq D$ from an ontology O_i, DDL uses the prefixed notation $i{:}C \sqsubseteq D$ to highlight the context of the axiom. Moreover, cross-context formulas can be defined to relate different terminologies. These particular formulas are called bridge rules and written either $i:C \xrightarrow{\sqsubseteq} j:D$ or $i:C \xrightarrow{\sqsupseteq} j:D$ where i and j are two different contexts, and C and D are terms from the contextual ontologies O_i and O_j respectively. A bridge rule $i{:}C \xrightarrow{\sqsubseteq} j{:}D$ (resp. $i{:}C \xrightarrow{\sqsupseteq} j{:}D$) should be understood as follows: from the point of view of O_j (*i.e.*, in the context j), C is a subclass (resp. superclass) of D.

In terms of model-theoretic semantics, this is formalised by assigning a distinct Description Logic interpretation $\mathcal{I}_i$ to each contextual ontology O_i, instead of having one single global interpretation. Thus, there are as many domains of interpretation as there are contexts. Additionally, cross-context relations are made explicit by so-called domain relations, that is set-theoretic binary relations between each pairs of contexts (formally, $r_{ij} \subseteq \Delta^{\mathcal{I}_i} \times \Delta^{\mathcal{I}_j}$). Two contextual interpretations $\mathcal{I}_i$ and $\mathcal{I}_j$ satisfy a bridge rule $i{:}C \xrightarrow{\sqsubseteq} j{:}D$ (resp. $i{:}C \xrightarrow{\sqsupseteq} j{:}D$) iff $r_{ij}(C^{\mathcal{I}_i}) \subseteq D^{\mathcal{I}_j}$ (resp. $r_{ij}(C^{\mathcal{I}_i}) \supseteq D^{\mathcal{I}_j}$).[4]

The advantage of this approach is the identification of context, a better robustness with respect to heterogeneity, improved modularity. However, it still misses some of the requirements that were identified in the integration scenario.

[4]For a set S, $r_{ij}(S) = \{x \in \Delta^{\mathcal{I}_j} \mid \exists y \in S, \langle x, y\rangle \in r_{ij}\}$.

Integration Scenario: Solution in DDL.

In the scenario, ontologies would be related thanks to C-OWL [182] bridge rules, which instantiates DDL for the Description Logic of OWL. A P2P reasoning system called Drago [186] implements a fragment of C-OWL and could be used in each hospital. Each peer manages its own context by reasoning with its internal ontology and "incoming" bridge rules. Messages are sent to neighbour peers according to a distributed algorithm involving bridge rules in order to take advantage of knowledge from other contexts.

In our healthcare use case, GUH and GOET may implement a Drago reasoner. GOET expresses the correspondences by way of bridge rules, as shown with a few examples in the Listing 8.2.

```
guh:( rxnorm:Insulin ) ⟹ goet:( rxnorm:Insulin )
guh:( rxnorm:Avandia ) ⟹ goet:( rxnorm:Avandia )
guh:( rim:playedRoleIn some rim:RolePatient ) ⟹ goet:( galen:Patient )
guh:( guh:hasMedication ) ⟹ goet:( goet:hasTreatment )
guh:( guh:sean ) ⟼ goet:( goet:345678IE )
```

Listing 8.2: Extract of DDL bridge rules

Because of the semantics of bridge rules, no inconsistency can be derived in this case. So DDL reduces the problem of diverging policies. In fact, DDL decreases interactions between different ontologies, which in turn decreases the chance of inconsistency.

Integration Scenario: DDL Limitations.

Bridge rules are not able to represent mappings between object and datatype properties, nor can they express operations on datatypes. Besides, C-OWL uses the same import mechanism as OWL. Additionally, the non-standard semantics of DDL may be counter intuitive, sometimes. Neither disjointness nor cardinality constraints are "transferred" from an ontology to the other via bridge rules. That is, if Insulin and Avandia are disjoint in GUH, and there are the bridge rules above, it cannot be inferred that Insulin and Avandia are disjoint in GOET. However, a variant of DDL has been defined to treat this specific problem [187]. Finally, the problem of policy is not completely solved. By adding the bridge rules of Listing 8.3, the GOET system can infer that a patient must not be treated with both Avandia and Insulin, which is what we tried to avoid.

```
guh:( guh:hasMedication some rxnorm:Insulin ) ⟹ goet:( goet:hasTreatment some rxnorm:Insulin )
guh:( not guh:hasMedication some rxnorm:Avandia ) ⟹ goet:( not goet:hasTreatment some rxnorm:Avandia)
```

Listing 8.3: Other possible bridge rules

While this example may seem a bit artificial, it shows that some restriction would have to be made on bridge rules to avoid undesired inferences. These restrictions are not, by themselves, defined by the formalism.

8.4.3 Other contextual reasoning formalisms

Contextual reasoning formalisms are characterised by a non-standard semantics where several ontologies are assigned distinct interpretations. Apart from DDL, this family of formalisms includes $\mathcal{E}$-connections, Package-based Description Logics and Integrated Distributed Description Logics.

Package-based Description Logics.

In package-based Description Logics (P-DL [18]), each ontological axiom is associated with an identifier of the ontology, similarly to DDL. Moreover, as in other contextual formalisms, a distinct interpretation is assigned to each ontology in a network of ontologies. However, cross-ontology knowledge can only take the form of *semantic imports* of ontological terms. The reason behind this is that this formalism was essentially designed to compensate the drawbacks of the OWL import mechanism and improve modularity of Web ontologies.

As an example, the GOET ontology only uses a few terms from the *GALEN* ontology. Since the *GALEN* ontology is extremely big and highly expressive, its complete import would result in a very complex, hard to manage ontology. However, while this helps building local ontologies in a modular way, it does not very much help expressing cross-context knowledge such as the correspondences that needed to bridge GUH and GOET. As an ontology alignment formalism, semantic imports have a very limited expressiveness. To express a complex correspondence such as, for instance, the one of Listing 8.4, one would have to first import the terms `guh:HemoglobinTest`, `rim:measures` and `loinc:_4548-4`, then add a *local* axiom in standard DL using the same approach as in Section 8.4.2.

```
Class: ( guh:HemoglobinTest and (rim:measures some loinc:_4548−4) )
    EquivalentTo: ( galen:BloodSugarTest and (goet:hasCode some {snomed:_43396009}) )
```

Listing 8.4: An example of correspondence that cannot be represented with semantic imports.

Integrated Distributed Description Logics.

Integrated Distributed Description Logics (IDDL [188]) is a formalism that address similar issues as DDL but take a different paradigm than other contextual frameworks. Usually, cross-ontology assertions (*e.g.*, bridge rules in DDL, links in $\mathcal{E}$-connections, semantic imports in P-DL) define knowledge from the point of view of one ontology. That is to say that the correspondences are expressing the relations "as witnessed" by a local ontology. On the contrary, IDDL asserts correspondences from a "third party"'s point of view which encompasses both the ontologies in relation. One consequence of this approach is that correspondences can be manipulated and reasoned with independently of the ontologies, allowing operations like inversing or composing ontology alignments, as first class objects.

In terms of model theory, this is represented by using an additional domain of interpretation to the whole network of ontologies, as if it was a single ontology. The local domains of interpretation, assigned to every ontologies, are then related to the global domain by way of the so-called *equalizing functions* (ε_i). These functions map the elements of local domains to elements of the global domain. Formally, a correspondence $i{:}C \xleftrightarrow{\sqsubseteq} j{:}D$ from a concept C of ontology O_i to concept D of ontology O_j is satisfied whenever $\varepsilon_i(C^{\mathcal{I}_i}) \subseteq \varepsilon_j(D^{\mathcal{I}_j})$.

Furthermore, a reasoning procedure for this formalism has been defined [189], where a central system detaining the correspondences can determine global consistency of a network of ontologies, by communicating with local reasoners of arbitrary complexity. This formalism is useful for federated reasoning systems, while the interactions between local ontologies are rather weak. By separating local reasoning and global reasoning, it better prevents interactions between contexts, thus being quite robust to heterogeneity.

In our example, let us assume that the correspondences of Listing 8.5 are defined to make the two systems interoperate.

guh:(*rxnorm*:*Insulin*) $\overset{\equiv}{\longleftrightarrow}$ **goet**:(*rxnorm*:*Insulin*)
guh:(*rxnorm*:*Avandia*) $\overset{\equiv}{\longleftrightarrow}$ **goet**:(*rxnorm*:*Avandia*)
guh:(*rxnorm*:*Avandia*) $\overset{\perp}{\longleftrightarrow}$ **goet**:(*rxnorm*:Insuline)
guh:(*rim*:*playedRoleIn* some *rim*:*RolePatient*) $\overset{\equiv}{\longleftrightarrow}$ **goet**:(**galen**:**Patient**)
guh:(*guh*:*hasMedication*) $\overset{\equiv}{\longleftrightarrow}$ **goet**:(**goet**:**hasTreatment**)
guh:(*guh*:sean) $\overset{=}{\longleftrightarrow}$ **goet**:(**goet**:345678IE)
guh:(*guh*:*hasMedication* some *rxnorm*:*Insulin*) $\overset{\equiv}{\longrightarrow}$ **goet**:(**goet**:**hasTreatment** some *rxnorm*:*Insulin*)
guh:(not *guh*:*hasMedication* some *rxnorm*:*Avandia*) $\overset{\equiv}{\longrightarrow}$ **goet**:(not **goet**:**hasTreatment** some *rxnorm*:*Avandia*)

Listing 8.5: Extract of IDDL correspondences

A policy of the form $C \sqsubseteq \neg D$ would only influence another ontology if a disjointness is asserted at the alignment level. One can see the similarity of these correspondences with bridge rules. Yet the resulting inferences differ. No inconsistency will arise from these correspondences. However, thanks to the third correspondence, the system would detect an inconsistency if a medicine is asserted to be Avandia and Insulin at the same time. While this formalism decreases undesired interaction of knowledge, especially with respect to policies, its drawback is the possible missing inferences at the local level. Moreover, correspondences are not more expressive than in DDL.

$\mathcal{E}$-connections.

$\mathcal{E}$-connections is another formalism for reasoning with heterogeneous ontologies [190]. Again, different ontologies are interpreted distinctly but formally related using particular assertions. Instead of expressing correspondences of ontological terms, an ontology can connect to another by using special terms (called *links*) which can be combined in conjunction with terms from another ontology. The semantics of links is very similar to the semantics of roles in Description Logics, except that instead of relating elements from the same domain of interpretation, they relate two different domains. In principle, $\mathcal{E}$-connections serve to relate ontologies about very different domains of interest. For instance, an ontology of laboratories in GUH could be connected to an ontology of medical staff used in GOET. To do this, one can define the link $\langle$hasDirector$\rangle$ and use it in GUH ontology as in 8.6.

guh:Laboratory $\sqsubseteq \exists\langle$ hasDirector $\rangle$**goet**:StaffMember

Listing 8.6: An axiom of an ontology using a *link* in the $\mathcal{E}$-connections formalism.

Thus, $\mathcal{E}$-connections are particularly useful for ontology design by modularly reusing and connecting existing blocks. However, one of the main focus of this thesis is on relating existing ontology systems on overlapping domains. So, although $\mathcal{E}$-connections is a relevant formalism for the management of heterogeneity, its applicability to the type of scenario we are interested in is weak.

8.4.4 State of the Art-3 : Context in Semantic Web Technologies

Models of Provenance

RDF model-theory [60] provides reification as a mechanism for making statements about statements. There are significant differences between reification and contexts both in what they are intended

for and in their structure. Reification is intended to enable statements about potential statements (which may or may not be true). They can be useful for making statements about provenance [191, 192]. Named graphs are also used for making statement about provenance [193]. In named graphs triples become quadruples where the fourth element is ID(URI), which may describe the origin of a graph. In addition to the reification mechanism, named graphs, N-Quads [194], and the system/language CWM/N3 [195] supports a construct called contexts. This notion of context is not substantially different from reification. Since these approaches to include additional information about the graph have no coupling with the truth of the triple that has been reified, they cannot be used to relate the truth of a triple in one graph to its truth in another graph. However, the main strength of the approach like Named graphs is its compatibility with the standard semantics.

Contextual RDF(S)

Guha *et al.* [172] proposed an extension of RDF(S)—Context Mechanism—to incorporate contextual knowledge within the RDF model theory. A simpler version of OWL is assumed to be interoperable with the proposed context mechanism. The most basic change in RDFS model-theory introduced—by the addition of contexts—is that the denotation of a resource is not just a function of the term and the interpretation (or structure), but also of the context in which that term occurs. Most importantly, the proposed context mechanism allows RDF statements to be true only in their context. The goal of this RDFS extension is to aggregate triples that are true in the graphs being aggregated, and because of the close coupling between truth and contexts [196, 197, 198, 199], an extension cannot be introduced at the RDF Vocabulary level but in the internals of the model theory *i.e.*, in the definition of an interpretation and satisfaction.

Guha's context mechanism updates the standard RDF(S) satisfaction by allowing set of context-dependent graphs instead of a single graph. Context mechanism allows a ground graph G_1 in a context C_1 to be entailed by a set of graph-context pairs $\langle G_i, C_i \rangle$ if $\langle G_1, C_1 \rangle$ is true under every interpretation under which ($\langle G_i, C_i \rangle$) is true. The proposed context mechanism may lead to non-monotonic aggregation—depending on the expressivity of *lifting rules*[5] [172]—in the following sense. A graph G_1 might imply φ but the aggregation of this graph (including lifting rules) with other graphs might not imply φ.

In our integration scenario, if the contents at URLs `http://wwww.example.guh.ie/guh.rdf` and `http://www.example.goet.de/goet.rdf` are available as RDF then we can have a context corresponding to these URLs and the contents of an URL is said to be true in that context. Furthermore, lifting rules can be defined to *import* all or part of the contents from data sources. For example, if we assume that an extended version of RDFS could express the *disjoint* axiom in GUH drug policy, then the truth of this axiom can be scoped to the context `http://wwww.example.guh.ie/guh.rdf`. The truth of GUH drug policy is applicable only within the GUH context and when GUH and GOET records are aggregated then the GUH drug policy could be easily ignored (thus avoiding inconsistency) by using an appropriate lifting rule. Similarly, when Sean is back in Galway, the GOET drug policy would not influence his further treatment in Galway or any other places. However, one major limitation of this proposed RDFS extension is the significant changes required in the standard RDFS semantics.

[5] lifting rules are basically normal *imports* that brings contents of one context to another

8.4.5 State of the Art-4 : Other formal handling of heterogeneity

Database Style Integrity constraints (IC) for OWL

This approach is motivated by data-centric problems in DL/OWL based applications. [200] established the relationship between the role of Integrity Constraints (IC) in databases, *i.e.*, (i) data reasoning (*e.g.*, in checking the integrity of a database) and schema reasoning (*e.g.*, in computing query subsumption), and (ii) DL/OWL knowledge bases (*e.g.*, schema (*TBox*) reasoning and data (*ABox*) reasoning). In this approach an additional *TBox* is introduced to model constraint axioms, which results in the knowledge base containing two *TBoxes* and an *ABox*. In *TBox* reasoning, constraints behave like normal *TBox* axioms, and for *ABox* reasoning they are interpreted as constraints in relational databases. This approach is very relevant in solving profile and policy issues in our integration scenario. For example, to avoid inconsistency due to hospital specific drug policy, the axiom:

$$\exists \text{guh:hasMedication.rxnorm:Avandia} \sqsubseteq \neg \text{guh:hasMedication.rxnorm:Insulin}$$

can be placed in the *TBox* for constraints and when *ABox* reasoning is performed constraint axioms can act as Integrity Constraints. To some extent, it helps formalising policies but since it does not identify the context of these constraints, their utility for this purpose is limited. Moreover, as standard OWL, robustness to heterogeneity is poor.

Modular Web Rule Bases

Although this approach is not based on current Semantic Web standards, it is relevant to this survey. The framework proposed in [14] makes the distinction between global knowledge, local knowledge and internal knowledge. The framework is based on a rule-based language rather than Description Logics and provides an approach to express and reason with modularity on top of the Semantic Web. In this framework each predicate in a rule base is constrained with "uses" and "scope", which in turn determine the reasoning process. The framework also treats different forms of negation (weak or strong) to include Open-World Assumption (OWA) as well as Closed-World Assumption (CWA) [201]. This rule-based framework provides a model-theoretic compatible semantics and allows certain predicates to be monotonic and reasoning is possible with inconsistent knowledge bases. This framework addresses a few issues of our integration scenario because rules can express some DL axioms and can be exchanged with certain restrictions (private, global or local). For example, the drug policy rule of our integration scenario:

$$\mathcal{F} \leftarrow \mathit{hasMedication}(?x,?y),\ \mathit{Avandia}(?y),\ \mathit{hasMedication}(?x,?z),\mathit{Insulin}(?z)$$

can be expressed and treated appropriately. However, the main problem we observe is how DL-based ontologies (as the majority of HCLS ontologies are DL ontologies) and rules can work together. The integration of DL with rules is still an open research problem [202]. Moreover, this framework is not concerned about the heterogeneity of the knowledge model, and does not provide an expressive way of relating contextual ontologies.

Query-based Data Translation

The query-based approach translates data from one knowledge source to another, and is close to the problem of expressing complex correspondences of the integration scenario. In this approach mappings between ontologies are first expressed in an expressive alignment language [203] and then grounded and executed to a combined query language and engine, SPARQL++ and PSPARQL, called PSPARQL++ [167]. Listing 8.7 shows how (i) two ontology entities (guh:orderDate, guh:time) could be concatenated to a single entity (dc:date) and (ii) a conversion is possible between an object property and a datatype property by using a proposed cast-function that converts *xsd:string* to RDF resource. Expressive correspondences between ontology instances can be constructed using "SPARQL CONSTRUCT" to create additional datasets and query upon them.

```
(a) CONSTRUCT { ?X dc:date fn:concat(?Date,"T",?Time). }
    WHERE { ?X guh:orderDate ?Date . ?X guh:time ?Time . }

(b) CONSTRUCT { ?X guh:hasId rdf:Resource(fn:encode−for−uri(?Id)) . }
    WHERE { ?X goet:identification ?Id . }
```

Listing 8.7: Mappings expressed as a SPARQL CONSTRUCT query

This approach allows one to express complex correspondences like concatenating attributes or even datatype to object properties and one can avoid some undesired interactions between knowledge of various sources. However, one major limitation is that the query result depends on how the correspondences are written and the knowledge in the domain ontologies are largely unexploited. Other than query-based approach, complex correspondences can also be expressed in Rule Interchange Format (RIF), which offers a rich set of built-in functions (*e.g.*, *string* manipulations), as well as a formal semantics for interoperating with RDF and OWL knowledge bases [204]. RIF has been recently approved as W3C's standard recommendation and can be considered as a prominent candidate to express rules (like our integration scenario) over OWL knowledge bases.

Reasoning with Inconsistencies

Robustness to heterogeneity is an important aspect in healthcare integration scenarios. One of the most problematic consequences of heterogeneity is the occurrence of undesired inconsistencies. Therefore, we believe it useful to investigate formal approaches for handling inconsistencies. There are two main ways to deal with inconsistent ontologies. One is to simply accept the inconsistency and to apply a non-standard reasoning method to obtain meaningful answers in the presence of inconsistencies. An alternative approach is to resolve the error, that is, to repair the ontology, whenever an inconsistency is encountered.

Repairing or revising an inconsistent ontology is, in principle, a possible solution for handling inconsistency. However, one major pragmatic issue we observe is that healthcare institutes may not expose and/or allow repair of their knowledge bases due to various legal constraints. Also, in a typical Semantic Web setting, importing ontologies from other sources makes it impossible to repair them, and if the scale of the combined ontologies is too large as in the case of HCLS ontologies then repair might appear ineffective. Other work focus on revising mappings only [205], but they are meant to be used at alignment discovery time.

Reasoning with inconsistencies is also possible without revision of the ontology. One effective way of tolerating inconsistencies consist of using paraconsistent logics [206]. Paraconsistent logics use a "weaker" inference system that entails less formulas than in classical logics. This way, reasoning can be done in the presence of inconsistency. A paraconsistent extension of OWL was proposed in [207]. Alternatively, defeasible argumentation [208] and its implementation Defeasible Logic Programs (DeLP [209]) have been introduced to reason and resolve inconsistencies. In this case, the TBox is separated into 2 subsets, one being *strict*, which means that it must always be used in reasoning, the other being *defeatable*, which means that an argumentation process may defeat them and nullify them for a particular reasoning task.

While we want to tolerate inconsistency when reasoning with an ontology defined in another context, it is not desirable to tolerate local inconsistencies in a HCLS system. The system should have a strict logical framework when it only treats local data, that are existing in a unique and well understood context. Unfortunately, the approaches mentioned here are not able to distinguish local knowledge and external knowledge. They do not allow specification of the types of mappings we need, and are not capable of treating policies.

8.5 Comparison of Formal Approaches

While the previous sections described the state of the art horizontally by merely describing approaches one by one, the present section analyses the features of formal approaches vertically, that is, each characteristic is compared separately throughout the extent of formal approaches.

Table 8.1 shows a brief, synthetic summary of what is detailed here. The columns can be seen either as a feature of a formal approach or as an issue related to heterogeneity. Consequently, the content of a cell can be read as "this formal approach possess this feature" (if a + is present) or not (if a − is present) or it can be read as "this issue is addressed by this formal approach" (+) or not (−). In the case of the last two columns, the issue defines a continuum between "this issue is not addressed at all" and "this issue is fully addressed", but all approaches are only partially addressing them. Thereby, we order the extent to which they address the issue in the following way:

	C.A.[6]	M.[6]	P.& P.M.[6]	C.E.[6]	R.H.[6]
DL/OWL	-	+/-	-	Good	Very limited
DDL/C-OWL	+	+	-	Very good	Good
P-DL	+	+	-	Very limited	Limited
DDL Revisited	+	+	-	Very good	Medium
$\mathcal{E}$-connections	+	+	-	Medium	Very good
IDDL	+	+	-	Good	Very good
Contextual RDF(S)	+	+/-	-	Good	Good
DeLP/Paraconsistent	-	-	+/-	Good	Good
Query-based	+/-	-	-	Very good	Excellent
Modular Rule bases	+	+	+/-	Limited	Limited
OWL/IC	-	+/-	+/-	Good	Limited

Table 8.1: Formal Approaches towards Heterogeneity and Context

[6]C.A.: Context-awareness; M.: Modularity; P.& P.M.: Profile and policy management; C.E.: Correspondence expressiveness; R.H.: Robustness to heterogeneity.

Very limited < Limited < Medium < Good < Very good < Excellent

The first column represents *context awareness* (C.A. in the table) which is the ability or possibility to identify the context in which some information or knowledge is described. The second column shows the possibility of *modularising* ontologies (M. in the table). This is considered fully addressed if subparts of ontologies can be reused rather than reuse of complete ontologies. The third column (P.&P.M. in the table) shows whether the formalism can be used to *manage profiles and policies*. The fourth column gives an indication of *correspondence expressiveness* (C.E. in the table), relatively to the other formalisms. We use here a loose notion of "expressiveness" and our classification is partly based on informal arguments rather than an authentic logical proof. Finally, the fifth column shows the *robustness with respect to heterogeneity* (R.H. in the table). This corresponds to the capability to exploit knowledge from independently designed sources while keeping coherence and relevance of inferences.

Context awareness can be enabled by clearly separating the axioms and facts asserted in distinct ontologies or from distinct provenance. Such formalisms as DDL, P-DL, $\mathcal{E}$-connections and IDDL assign an identifier to each of the ontologies such that an axiom $i : C \sqsubseteq D$ can be associated with an ontology O_i. This also ensures the distinction between local axioms and cross-ontology correspondences, such as bridge rules $i : C \stackrel{\sqsubseteq}{\rightarrow} j : D$. However, these formalisms do not allow one to describe or give a type to contexts. The contextual RDF framework of [172] improves on this by treating context identifiers as any other RDF resource identifiers: they are URIs that can be dealt with as first class objects. The case of query-based transformations is a bit particular because the transformation must say from which and to what datasets the translation occurs, but the transformation could actually be used independently of the context. This explains the $+/-$ sign for this formal approach. Modular rule bases uses a mechanism similar to the one of DDL etc but the underlying formalism is not based on Semantic Web languages.

Modularity is almost always associated with context awareness. This is because in order to reuse modules, one must be able to identify them, that is, have a mean of identifying the provenance of knowledge. Moreover, formalisms such as DDL, $\mathcal{E}$-connections, P-DL, IDDL are designed to enable the reuse of external knowledge through the use of cross-context assertions like imports, bridge rules, links, correspondences. Therefore, what identifies a context can be thought of as a module. To a lesser extent, OWL allows some modularisation but is limited to the full import of complete ontologies which results—from a logical point of view—in the same ontology as a complete merge of all imported knowledge. OWL/IC has the same behaviour as OWL in terms of modularity. Finally, the contextual RDF approach allows to identify contexts and relate them, it is harder to separate knowledge into modular blocks since contexts can overlap and, as first class objects, contexts themselves can be part of a context. These intertwined elements make modularisation in contextual RDF(S) more difficult.

Policies and profiles has not been often addressed as a knowledge representation issue. Most of the solutions to this problem are using ad hoc implementations or algorithm that are not explicitly tackled by the underlying formalism. Of course, it is possible to use existing languages—such as RDF and OWL as in [210]—to represent policies but OWL cannot formally distinguish between ontological axioms and policy axioms. This is true for most of the formalisms presented here. Notable exceptions are Defeasible Logic Programs for which there is a distinction between "strict"

knowledge, which is always true, and "defeatable" knowledge which can be canceled by way of an automatic argumentation process. Policies are typically defeatable by stronger policies (*e.g.*, from GUH's point of view, GOET's policy axioms would be defeated by contradicting local policies). As an alternative, OWL/IC offers a way to separate the usual ontological knowledge from database-style constraints. Unfortunately, both approaches are unable to distinguish the context in which appear the axioms or policies. Thereby, they can only separate the totality of the ontological axioms from the totality of the policy constraints. Finally, Web Rule base approach has a similar aptitude to distinguish different types of knowledge by assigning different reasoning schemes to different sets of rules. Additionally, it separates knowledge from different sources in a similar way as contextual formalisms do. However, it is not complying to Semantic Web technologies and it would hardly be possible to integrate it with description logics systems.

Correspondence expressiveness varies a lot depending on the formalism being used. Most of the correspondences given in the integration scenario are directly expressible in OWL due to its high expressiveness. However, OWL constructs still have some limitations which appear in practical cases. In formalisms that allow delimiting contextual knowledge, the expressiveness of correspondences is defined by the types of relations that can be asserted between contexts. The case of the query-based approach to transforming data is notable. This approach allows for defining very fine grained correspondences in the form of a data transformation. Basing the approach on queries decreases the exploitation of reasoning but enables all kinds of structural and functional transformations.

Robustness to heterogeneity is the ability to make consistent and useful inferences in spite of the variations of points of view, modelling approaches and contexts. There are two extreme cases: one is the classical logic approach, the other is the context separation approach. In classical logic, all statements are treated equally in a theory and can equally interfere with any other statement. Therefore, incompatible views are very likely to produce inconsistencies or nonsense. This results in a formalism very vulnerable to heterogeneous knowledge. On the opposite side, it is possible to simply separate all conclusions drawn from a context from any other context. This is very robust to heterogeneity since, granted that each context is self consistent, it cannot produce incompatible inferences. However, this way, no context can take advantage of information coming from a different context. Context aware formalisms can be ordered in their ability to tolerate heterogeneity: P-DL $<$ modular rule bases $<$ DDL revisited $<$ DDL $<$ IDDL $<$ $\mathcal{E}$-connection $<$ Query-based. The reason why the query-based approach is so robust is that it does not actually reason with a source ontology to produce new data according to the target knowledge base. Moreover, the transformation can be designed in such a way that it conforms to the destination ontology.

Conclusion of the state of the art. Although the presented formalisms are significantly different from each others, there are common aspects to them and some of them are not incompatible. First, in all these approaches, we can recognise the need to delimit parts of the knowledge to assign them either provenance information, a type (policy, general knowledge) or a distinct reasoning scheme (OWA, CWA). Thereby, a better solution to the problem of heterogeneity should possess this ability to identify subparts of the knowledge and apply different status to them. Second, context-aware formalisms are usually adequate approaches to modularisation thanks to their ability to relate distinct ontologies. Additionally, the separation of knowledge makes them more tolerant to heterogeneous modelling. However, these approaches have very little addressed issues of policies and

profiles. Yet, handling policy-like information differently from ontological knowledge is critical to the management of electronic patient records. Improving the so-called *modular ontology languages* with policies, or making policy-aware formalisms more context sensitive is an important aspect to achieve interoperability of HCLS systems. For this reason, we present a preliminary solution path for enabling a multi-policy System. Since this is by no mean a complete solution to the vast problem of heterogeneity, we thereafter discuss more briefly the other critical issues that should be covered by a semantic formalism to represent the knowledge involved in this type of system.

8.6 Towards a framework for handling context and modularity in a Multi-Policy System

We now attempt to propose a general framework for separating policies from general knowledge bases and adding a context-aware formalism. While a semantics must be chosen to allow context aware reasoning, it can be extended with different formal approaches. We have discussed in the previous sections that semantics for a multi-policy system involves several layers, all of which needed to be extended in a manner that policies originated from several sources should be stored, retrieved and reasoned upon distinctively. Two basic extensions required are: (i) modular knowledge structure that allows separating global and local knowledge bases in distinct modules (or in contexts); and (ii) semantics that formally describes the properties of a multi-policy system. In this thesis we define the first extension (modular knowledge structure). The second extension (semantics for multi-policy systems) is a future task layered on top of the modular knowledge structure.

We propose the combination of the formalisms that identify the context of axioms with a mechanism for handling constraints that do not lead to undesired interactions. We start with the notion of separating knowledge bases which originates from [200] and discussed in Section 8.4.5, where axioms are separated into two T-Boxes, one for ontological axioms, the other for integrity constraints. Although, the approach presented in [200] is more about validating RDF data with constraints expressed in OWL, we extend this notion of separating knowledge bases for policies and contextual reasoning.

We now introduce a formalism for a multi-policy system. We provide a multi-policy knowledge definition in the standard DL sense and discuss how this could be realised using the DDL framework.

Formalism

We start with our general idea of separating knowledge bases into global and local parts. DDL [17] described in Section 8.4.2 proposes global and local interpretations of distributed TBoxs where each local TBox can be shared by interested parties via bridge rules. In result, local knowledge is opaque to external systems and only exchange possible between interacting systems are bridge rules. We extend this notion by separating the local TBox in two parts, one for global domain knowledge base (D) and the other for policy-like axioms (P). A policy-like axiom is a constraint placing a certain restriction on the use of individuals and their relationships within a knowledge base. Figure 8.3 depicts the general idea behind the multi-policy system.

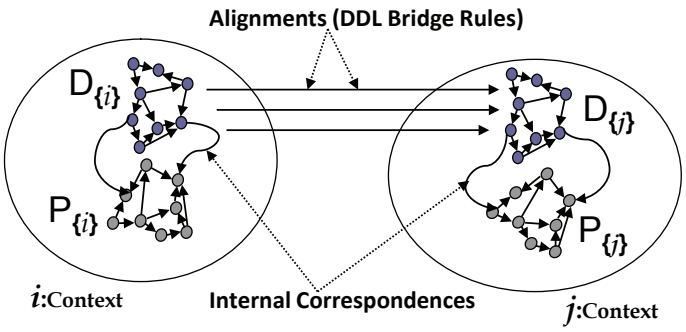

Figure 8.3: Multi-Policy Systems

Definition 8.6.1 ***(Multi-Policy Knowledge Base)*** *Let I be a set of indices of URIs representing contexts. A multi-policy knowledge base is a triple $(D_i, P_i, A_i)_{i\in I}$ where D_i is a distributed TBox, P_i is a distributed policy box (PBox), and A_i is a distributed ABox. A global knowledge space $(\bigcup_{i\in I} D_i)$ is a combination of shared domain ontologies. PBox is a local TBox which contains policy-like axioms only.*

We define a local knowledge base as a pair $\langle D_i, P_i\rangle$, where D_i describes a domain ontology, and P_i represents the internal policies. If several local ontologies and constraints exist, the overall knowledge is a distributed system ($\langle D_i, P_i\rangle$). To ensure interoperability, ontology alignments (a_{ij}) are added to the system to bridge the gaps between different terminologies. Note that these alignments could be simple DL axioms, DDL bridge rules, P-DL semantic imports, or IDDL correspondences. In cases where alignments are expressed by bridge rules, the resulting system (*i.e.*, multi-policy knowledge base) is a pair of tuples $\langle(\langle D_i, P_i\rangle), (a_{ij})\rangle$. In DL/OWL alignments are supported by various constructs *e.g.*, *subClass*, *sameAs*, *equivalentClass*, *equivalentProperty*, *differentFrom*, and *AllDifferent*. However, in DDL, alignments are primarily restricted to the *subClass* relation.

Definition 8.6.2 ***(Entailment wrt Policy** i**)*** *Let $K = (D_i, P_i, A_i)_{i\in I}$ is a multi-policy knowledge base. Let α be an axiom in the vocabulary of a global knowledge space $(\bigcup_{i\in I} D_i)$. K P_i-entails α (or $K \models_{P_i} \alpha$) iff $(\bigcup_{i\in I} D_i) \cup P_i$ entails α.*

The semantics of a multi-policy system depends on the type of alignments in place. For example, in the DDL framework, a simple pair of OBIS (as in our integration scenario) is $\Omega = \langle(\langle D_{\text{guh}}, P_{\text{guh}}\rangle, \langle D_{\text{goet}}, P_{\text{goet}}\rangle), a_{\text{guh,goet}}\rangle$. The entailment wrt policy i ($\models_{P_i}$) is the distributed entailment relation for the multi-policy system. Thereby, $\langle(\bigcup_{i\in I} D_i), (a_{ij})\rangle \models_{P_i} k : C \sqsubseteq D$ means that the global knowledge space $(\bigcup_{i\in I} D_i)$ and alignment (a_{ij}) distributively entails the axiom $C \sqsubseteq D$ in ontology D_k. A multi-policy entailment $\models_{P_i}$ is defined over distributed ontologies (in DDL) Ω as follows:

For a given local OWL axiom $\alpha = \text{guh} : x \sqsubseteq y$ in the terminology of the ontology of GUH, $\Omega \models_{P_i} \alpha$ if and only if α is distributively entailed by the system composed of $D_{\text{guh}} \cup P_{\text{guh}}$, $\{D_{\text{goet}}\}$ and $A_{\text{guh,goet}}$ (and vice versa if the axiom belongs to the ontology of GOET). In other words, only the policy axioms of the ontology which is asking for an entailment is used. In our scenario, it means

that if GUH is reasoning, it will take its drug policy into account but not the one of GOET, while GOET would not consider the Avandia-Insulin counter indication of GUH. The very same approach can be easily adapted to P-DL or IDDL.

Next step in realising a multi-policy system is to create a context-sensitive data model that may allow the arrangement of knowledge bases into separate contexts or modules. In DLL (and other context-aware formalisms discussed in Sections 8.4.2-8.4.3) context is abstractly identified by an identifier, associated to an ontology and to the axioms of the ontology. In DLL implementation, the reasoners rely on in-memory data structures which may not be exchangeable. For instance, in Drago (a DDL reasoner) [186], an ontology is part of a context if it has been loaded explicitly by a peer which identify the context. So context is an instance of a Drago peer reasoner. However, we want to exchange information about context, notably by publishing it on the Web, we only want to identify context, leaving the actual definition to each application that uses context. We can do that by providing a universal identifier (a URI) to a context and we can offer a minimal RDF vocabulary describe the context. On top of the context-sensitive data model, the policy entailment can be realised by selecting relevant parts of the knowledge bases, thus, avoiding undesirable conflicts. A natural consequence of the policy entailment is that inferences are drawn from the selected modules. Naturally, different selection of modules might infer different consequences. Currently, we are developing (i) a context-sensitive data model; and (ii) a vocabulary describing various types of context-dependent policies (*e.g.*, admin, data-access, security, reimbursement).

The comparison of formal approaches presented in the Section 8.5 and a solution path for realising a multi-policy system contribute to our third research challenge (*i.e.*, "how to resolve inconsistencies caused by context-sensitive (or local) healthcare policies while mediating heterogeneous HL7 messages"). The work presented in thesis is a preliminary step towards a significant complex problem of handling context, modularity, and policies together by an ontology-based integration framework. A concrete realisation of a multi-policy system would significantly improve the functioning of ontology-based integration frameworks like PPEPR. A multi-policy system would allow arranging healthcare ontologies (*e.g.*, HL7) and policies in a context-sensitive data model and reason locally upon them.

8.7 Summary

In this chapter, we discussed the issue of context and modularity within ontological frameworks. A context-aware and modular system is crucial in the appropriate treatment of local knowledge including institutional policies. We focused on formal aspects of knowledge representation formalisms, assessing their ability to effectively model the information commonly needed in electronic health records (EHRs). While the discussion was essentially on theoretical aspects, we discussed the concrete consequences of applying these formalisms to the HL7 based scenario. The strengths and limitations identified for each formalism became the motivation for proposing a solution path enabling a context-aware and modular knowledge structure. We argued that an approach to the problem of heterogeneity—or in general interoperability of healthcare systems—relies on the combination of several integration models. However, complexities of different magnitude may arise from this combination. In the future, we aim to investigate formal properties concerned with compatibility of policies, inter-module dependencies, exploit context relations, and resolution of distributed policies in multi-policy systems. These formal properties will be supported by (i) an extensive evaluation by including various types of polices (data access, security, administrative, etc.) in the healthcare environment; and (ii) an implementation of a multi-policy system.

Chapter 9

Conclusion and Future Works

In this thesis, we have presented how healthcare applications can benefit from semantic technologies to improve interoperability. The thesis focus was centred around the most widely deployed Health Level Seven (HL7) standard. We have shown that a patient record consists of cross-domain vocabularies and terminologies (*e.g.*, SNOMED, LOINC, RxNorm), thus, an integration framework must support mediation of resources originating from multiple related domains. Throughout this thesis, we argued that the reconciliation of local knowledge bases with the global domain knowledge is a key issue in delivering an ontology-based integration framework for the healthcare domain. Therefore, methodologies that may allow the engagement of local resources (bottom-up) and resources provided by standards (top-down) would ultimately be able to deliver a unifying framework for interoperable healthcare applications. The methods presented in this thesis are based on a real-life integration scenario which highlights the requirements of intra-hospital and lab message exchanges. We give a summary of the main contributions of this thesis presented in Chapters 4–8.

- **Ontology Building Methodology:** We proposed the PPEPR methodology which describes a step-by-step ontology development process for HL7 applications. The PPEPR methodology is semi-automatic where the core step—Lifting HL7 Resources—is completely automated.
- **Ontology Alignment:** We proposed a semi-automatic alignment method based on "SPARQL Recipes" that improves the overall accuracy of the alignment results for HL7 ontologies. "SPARQL Recipes" are based on domain specific matching patterns, which are formulated by the domain experts.
- **Integration Framework:** We proposed the PPEPR integration framework that automates the execution of ontologies, alignments, and run-time instances between HL7 Version 3 and Version 2 applications. PPEPR has been successfully evaluated and licensed by a company providing a healthcare integration solution.
- **Context-Awareness and Modularity:** We investigated formalisms concentrating on what relates to context, modularity, policies, and the overall heterogeneity of healthcare applications. We proposed a solution path enabling a multi-policy system that may allow context-sensitive local policies to be treated according to a local environment. We are aware that the issues of

context, modularity, and local policies are vast research problems, as such, and the proposed solution path is a preliminary step in addressing these issues.

We now review the central hypothesis of this thesis as originally introduced in Section 1.3.

9.1 Assessing The Hypothesis

Our hypothesis:

"***An ontology-based integration framework can improve healthcare data interoperability by reducing: (i) the integration burden (per message) between heterogeneous healthcare applications; and (ii) the number of alignments between heterogeneous healthcare applications***"

attempts to create a value proposition of an ontology-based integration framework in resolving the heterogeneities of healthcare applications. We now discuss how the presented contributions contribute to the two main objectives: (i) reducing the integration burden between heterogeneous healthcare applications; and (ii) reducing the number of alignments between heterogeneous healthcare applications.

Reducing Integration Burden

We measured the integration burden on an ontology-based integration framework on three dimensions (see Section 7.8): (i) design-time development effort measured (per message), (ii) run-time efficiency of exchanging messages, and (iii) deployment methodology comparing traditional software deployment. Since healthcare resources are separated into global and local parts, we found that including global ontologies and their alignments is the prime reason in reducing the bilateral correspondences between a pair of healthcare applications. Our approach of separating global and local ontological knowledge-bases (see Section 4.4) resulted in delineating the scope of a heterogeneous resource, thus improving the overall interoperability of HL7 applications. We measured a significant reduction in the time-effort allocated for mediating two heterogenous messages (see Section 7.8). At present, we require 0.5 days per message to resolve the heterogeneities compared to 12 days needed by existing integration solutions. The run-time efficiency (or performance) of exchanging a HL7 message within 1.5 seconds is comparably better than existing integration solutions. The deployment of any existing or proposed HL7 application (or EHR) is a straightforward job where only end-points need to be configured within the PPEPR Framework. PPEPR automatically detects the heterogeneities and resolves them through the ontologies and alignments developed at the design-time. This impressive reduction in the integration effort is achieved due do the automated methods (see Section 5.4 and Section 6.8) proposed in this thesis.

Reducing Alignments

We measured the number of alignments in cases where all of the resources are local and cases where resources are separated into global and local parts. Existing integration solutions consider all

heterogeneous resources as local and encourage one-to-one alignment between a pair of heterogenous HL7 applications, thus resulting a quadratic size alignment (see Section 2.4). We measure that a typical HL7 message (90 *complexTypes* and 50 *elements* or *attributes*) requires 140 mappings in a one-to-one alignment approach as is usually adopted by existing integration solutions (see Section 5.4). Our approach of resolving the majority of heterogeneities at the global level (*i.e.*, ontologies and alignments) resulted in 73 local mappings between a pair of HL7 applications (see Section 7.8). This is a 50 percent reduction in the bilateral correspondences between a pair of HL7 applications. The reduction of alignments is further supported by an automated method (*i.e.*, SPARQL Recipes) for discovering matching correspondences between heterogeneous resources (see Section 6.8). SPARQL Recipes limit the developer's involvement in discovering matching correspondences. Any future changes introduced in the HL7 standard specifications will be handled at the global level (*i.e.*, updating global ontologies and their alignments) and local differences will be resolved with the help of methods (*i.e.*, lifting and alignment) proposed in this thesis.

Our hypothesis has been tested over 5 years and we received consistent evaluation results that are presented in this thesis. Nonetheless, the diversity of the healthcare field is so vast that we did not cover some relevant issues that we only sketched to provide a roadmap for further research. We believe that these open research issues require the attention of a broader research community.

9.2 Open Issues and Future Directions

This section presents critical issues of semantic-enabled healthcare systems that have not been addressed by this thesis. Since the breadth of this domain is so wide, this section can only discuss those problems succinctly, but we hope to provide directions for future research on the key obstacles to better interoperability.

9.2.1 Ontology Building Methodology: Complexity and System Behaviour

We discussed that ontology building methodologies are in their early stage when compared to software engineering methodologies. Software engineering methodologies are well matured in guiding the development process, which might include distributed teams or resources, independently owned modules of large projects, complex and flexible workflows, reusability of existing resources, etc. The full potential of Semantic-Web enabled applications (domain-dependent or general) requires the large-scale and distributive adoption and use of ontologies to share knowledge and resources. In integration scenarios as presented in this thesis, instead of a single, centralised ontology, it is natural to have multiple distributed ontologies that cover different, perhaps partially overlapping, domains (*e.g.*, healthcare, biology, medicine, pharmaceutical). The PPEPR methodology presented a step-by-step development process that focuses on reusability of existing resources and alignment of knowledge bases separated into global and local spaces.

The PPEPR methodology needs to be extended in directions that contribute to the maturity of the methodology by: (i) analysing the complexity of the ontologies built, which would be helpful in deciding, the deployment and the requirements on the healthcare information systems; (ii) the PPEPR methodology has covered the static part of the HL7 information model. HL7 Version 3

also provides behavioural models (*i.e.*, HL7 Interaction Model) which describe workflows between distributed HL7 applications. It is important to extend the PPEPR methodology to cover the behavioural semantics [211] of interacting applications. Such extension would contribute to the meaningful discovery, composition and orchestration of Web service-enabled HL7 applications.

9.2.2 Lifting Resources: Seamless Transformation

The recent advancement of Web-based technologies has resulted in the greater availability of structured information resources. We discussed that the transformation of syntactic format (XML(S)) to declarative resources (RDF or OWL) is hindered by gaps in their foundational structure (tree *vs.* graph). In PPEPR we used domain-specific rules that perform the lifting and lowering of HL7 resources. However, a general transformation framework is required that domain experts can use as a black box system, *i.e.*, without involving themselves into the internals of language transformation. Considering the size and availability of structured resources in the healthcare and related domains (*e.g.*, medical, biology, life sciences), the PPEPR lifting approach needs to be extended either towards a general transformation framework, or to include transformation-rules from other related domains.

9.2.3 Ontology Alignment: Accuracy and Automation

We discussed that the accuracy of ontology alignments is crucial in the healthcare domain. SPARQL recipes proposed in thesis covers only the HL7 standard. We are aware that other sub-domains in Healthcare and Life Sciences (HCLS) have similar features to HL7 where annotation descriptions are richer, which may play a major role in deciding correspondences between ontological elements. SPARQL recipes can be extended by identifying a list of matching patterns per sub-domain and to develop a unified framework where domain experts are allowed to select patterns from their domain of interest and execute SPARQL recipes for source ontologies. Other than SPARQL recipes, it would be also interesting to extend the existing automated alignment systems by including domain-specific thematic structures instead of general information structures like WordNet, Wikipedia, DBpedia, etc.

9.2.4 Context and Modularity: Rule Based Approach

On many occasions, data integration use cases advocate hybrid formalisms that can use open world semantics of ontologies with closed world semantics of rules under a single framework. There are strong arguments from knowledge representation specialists in favour of using closed world semantics for constraints (or policies) like axioms [212, 213, 210, 214]. Rules have been thoroughly studied in the knowledge representation and logic programming communities [215, 216]. In our integration scenario, we dealt with knowledge bases shared across applications and various application-specific constraints resulting in conflicts when used or referred outside their origin. The shared part of a knowledge base, expressed in DL, fits perfectly with the open world semantics. However, application-specific constraints always exist within applications, which basically makes sensible use of the shared knowledge bases. For example, in Listing 9.1 an insulin dose policy is described in DL. This policy

enforces that insulin dose can only be given to diabetic Type 1 or Type 2 patients and not to patients suffering from both. Patients suffering from both are a special case and they require other drug policy for them. In this policy, for Sean to have insulin therapy, it is not sufficient that Sean is an instance of $DiabeticType2$, Sean also has to prove that he is not DiabeticType1.

$$\mathsf{InsulinPolicy} \sqsubseteq \mathsf{DrugPolicy}$$

$$\mathsf{InsulinPolicy} \equiv (\mathsf{DiabeticType1} \sqcup \mathsf{DiabeticType2}) \sqcap \neg\, (\mathsf{DiabeticType1} \sqcap \mathsf{DiabeticType2})$$

Listing 9.1: Insulin Dose Policy for Diabetic Patients in DL

Equivalently, in Listing 9.2 (rule equivalent of the Insulin Policy) a knowledge base assertion *DiabeticType2(Sean)* is sufficient to conclude that Sean will get the Insulin therapy. Since it is not provable that *DiabeticType1(Sean)*, CWA simply assumes that Sean is not a $DiabeticType1$. In this case the behaviour of CWA appears to be more reasonable than its OWA counterpart in Listing 9.1.

```
DrugPolicy(X) ← InsulinPolicy(X)
InsulinPolicy(X) ← (DiabeticType1(X); DiabeticType2(X)), NOT (DiabeticType1(X), DiabeticType2(X)).
```

Listing 9.2: Insulin Dose Policy for Diabetic Patients (Rules)

The rule-based approach should be based on our general notion of separating the knowledge base and applying a context-aware formalism on top of separated (global and local) knowledge bases. The idea is to switch between closed and open world semantic by executing the application specific constraints under CWA while shared part of the knowledge base will receive the OWA treatment. Such hybrid approaches has already been initiated where all (or part) of the DL knowledge is converted and executed in rule based engines [217, 218], or vice versa. To realise such a solution we need a Semantic Web language that can easily be executed under rule engines. RDF(S) fits such a requirement. The reason is three folds (i) rule support in RDF(S) is already a practical reality [195, 219, 220], (ii) several initiatives have been taken for RDF(S) to rules conversion [221, 222] and, (iii) many DL reasoner already support the rule fragment (*e.g.*, Pellet, Jena).

9.2.5 Temporal aspects of healthcare systems

Temporal information is everywhere in patient records. For example, Lab observations (*e.g.*, blood test, blood pressure) have a duration of validity, vaccines must be renewed after some time, medicines have an effect on the body for a certain time (thus, counter-indications are valid beyond the duration of the treatment). One aspect of implicit, contextual information is its temporal component. The ability to identify, represent and reason about time-dependent information is important for various applications, especially for healthcare systems. With respect to temporal information one is often faced with the problem of implicit time [223]. For example, patient information and laboratory observations are true at the time of publication, with the temporal qualification left implicit. Equally often, this information does not get updated when it no longer holds. Even worse, such implicitly temporally qualified data is often mixed with data that is not temporally qualified. In this thesis, we have not directly addressed the time-dependency issue but we assume that a framework that enables context-awareness within the knowledge bases can act as a ground work for enabling and processing temporal information.

9.2.6 Practicality of formal approaches

The implementation of methodologies and the proposed formalisms themselves would lead to even further research issues that are at least equally important and challenging as theoretical aspects.

First, the more expressive and powerful the formalism is, the more difficult it is to use and to model knowledge with. This can be overcome by building intuitive interfaces that help the user in these tasks. The problem occurs on the side of the knowledge engineer building ontologies and for the practitioner who needs to understand what the system is reporting. For practitioners, explaining inference results is often a complex task.

Second, it is sensible to assume that not all systems will ever use a unique data model. It is thus important to rely on data conversion. In this thesis, we focused on XML-based legacy systems, however, other significant resources are available in relational databases. Initiatives like RDB2RDF[1] are working in such direction that will help taking advantage of legacy-databases within the framework of the Semantic Web.

9.3 Concluding Note

Researchers and practitioners in HCLS domain have been facing heterogeneity problems for decades already, whereas Semantic Web technologies are a comparably new solution they started looking at. One crucial aspect of the interoperability solutions so far was that the domain experts have taken a top-down approach to create and use agreed-upon vocabularies. The recent W3C initiatives like Linking Open Drug Data (LODD)[2] and BioRDF[3] are taking a application-driven bottom-up approach by focussing on lower levels of the Semantic Web layer cake. However, onus is still on publishers linking healthcare resources and mechanisms enabling user privacy and security on top of the linked resources. We envision that the Semantic Web can play a crucial role in the convergence of healthcare standard specifications published in a top-down manner and legacy or application-dependent resources originating from the user community. The thesis advances and advocates the use of Semantic Web technologies as a key interoperability enabler for healthcare applications.

[1] http://www.w3.org/2001/sw/rdb2rdf/
[2] http://www.w3.org/wiki/HCLSIG/LODD
[3] http://www.w3.org/wiki/HCLSIG_BioRDF_Subgroup

Bibliography

[1] G. W. Beeler, S. Huff, W. Rishel, A. M. Shakir, M. Walker, C. Mead, and G. Schadow, "HL7 Version 3 Message Development Framework," December 1999. [Online]. Available: http://www.hl7.org/library/mdf99/mdf99.pdf

[2] I. Iakovidis, "Towards personal health record: current situation, obstacles and trends in implementation of electronic healthcare record in Europe," *International Journal of Medical Informatics*, vol. 52, pp. 105–115, 1998.

[3] M. Henderson, "HL7 Version Messaging Standard Version 2.5: An Application Protocol for Electronic Data Exchange in Healthcare Environment," 2003.

[4] G. W. Beeler, J. H. Duteau, G. Grieve, L. McKenzie, and R. Natarajan, "HL7 Version 3 Standard," May 2011, Version 3 Ballot. [Online]. Available: http://www.hl7.org/v3ballot2011may/html/welcome/environment/index.html

[5] V. Bicer, O. Kilic, A. Dogac, and G. B. Laleci, "Archetype-Based Semantic Interoperability of Web Service Messages in the Health Care Domain," *International Journal of Semantic Web and Information Systems*, vol. 1, no. 4, pp. 1–22, 2005.

[6] A. L. Rector, R. Qamar, and T. Marley, "Binding Ontologies & Coding Systems to Electronic Health Records and Messages," in *KR-MED 2006, Formal Biomedical Knowledge Representation, Proceedings of the Second International Workshop on Formal Biomedical Knowledge Representation: "Biomedical Ontology in Action" (KR-MED 2006), Collocated with the 4th International Conference on Formal Ontology in Information Systems (FOIS-2006), Baltimore, Maryland, USA, November 8, 2006*, ser. CEUR Workshop Proceedings, O. Bodenreider, Ed., vol. 222. Sun SITE Central Europe (CEUR), 2006.

[7] A. L. Rector and S. Brandt, "Why do it the hard way? The Case for an Expressive Description Logic for SNOMED," *Journal of the American Medical Informatics Association*, vol. 15, no. 6, p. 744, Aug. 2008.

[8] I. Iakovidis, A. Dogac, O. Purcarea, G. Comyn, and G. B. Laleci, "Interoperability of eHealth Systems - Selection of Recent EU's Research Programme Developments," in *Proceedings of CeHR: International Conference 2007, eHealth: Combining Health Telematics, Telemedicine, Biomedical Engineering and Bioinformatics to the Edge*, Regensburg, Germany, 12 2007.

[9] T. R. Gruber, "Towards Principles for the Design of Ontologies Used for Knowledge Sharing," in *Formal Ontology in Conceptual Analysis and Knowledge Representation*, N. Guarino

and R. Poli, Eds. Kluwer Academic Publisher, 1993. [Online]. Available: http://citeseer.ist.psu.edu/gruber93toward.html

[10] M. Fernández-López and A. Gómez-Pérez, "Overview and analysis of methodologies for building ontologies," *Knowledge Engineering Review*, 2002.

[11] D. W. Lonsdale, D. W. Embley, Y. Ding, L. Xu, and M. Hepp, "Reusing ontologies and language components for ontology generation," *Data Knowledge Engineering*, vol. 69, no. 4, pp. 318–330, 2010.

[12] M. Hepp, "Possible Ontologies: How Reality Constrains the Development of Relevant Ontologies," *IEEE Internet Computing*, vol. 11, no. 1, pp. 90–96, 2007.

[13] C. Ghidini and F. Giunchiglia, "Local Models Semantics, or contextual reasoning=Locality+Compatibility," *Artificial Intelligence*, vol. 127, no. 2, pp. 221–259, 2001.

[14] A. Analyti, G. Antoniou, and C. V. Damásio, "A Principled Framework for Modular Web Rule Bases and Its Semantics," in *Principles of Knowledge Representation and Reasoning: Proceedings of the Eleventh International Conference, KR 2008, Sydney, Australia, September 16-19, 2008*, G. Brewka and J. Lang, Eds. AAAI Press, 2008, pp. 390–400.

[15] D. Calvanese, G. D. Giacomo, and M. Lenzerini, "A framework for ontology integration," in *Proceedings of the 2001 International Semantic Web Working Symposium (SWWS 2001)*. IOS Press, 2001, pp. 303–316.

[16] N. F. Noy and M. A. Musen, "The PROMPT suite: interactive tools for ontology merging and mapping," *International Journal of Human-Computer Studies*, vol. 59, no. 6, pp. 983–1024, 2003.

[17] P. Bouquet, F. Giunchiglia, F. van Harmelen, L. Serafini, and H. Stuckenschmidt, "Contextualizing ontologies," *Journal of Web Semantics*, vol. 1, no. 4, pp. 325–343, 2004.

[18] J. Bao, D. Caragea, and V. G. Honavar, "On the Semantics of Linking and Importing in Modular Ontologies," in *Proceedings of the 5th International Semantic Web Conference, ISWC 2006, Athens, GA, USA, November 5-9, 2006*, ser. Lecture Notes in Computer Science, I. F. Cruz, S. Decker, D. Allemang, C. Preist, D. Schwabe, P. Mika, M. Uschold, and L. Aroyo, Eds., vol. 4273. Springer-Verlag, Nov. 2006, pp. 72–86.

[19] P. Szolovits, *Artificial Intelligence in Medicine*. Boulder, CO, USA: Westview Press, Inc., 1982.

[20] T. Bray, J. Paoli, C. M. Sperberg-McQueen, E. Maler, and F. Yergeau, "Extensible Markup Language (XML) 1.0 (Fifth Edition), W3C Recommendation 26 November 2008," World Wide Web Consortium (W3C), Tech. Rep., Nov. 11 2008. [Online]. Available: http://www.w3.org/TR/xml/

[21] S. S. Alhir, *UML in a Nutshell*. O'Reilly, 1998.

[22] A. R. Mori, "Standardization Efforts for Providing Semantic Interoperability in eHealth Domain," Tech. Rep., May 2006, RIDE Project Deliverable D.2.2.1. [Online]. Available: http://www.srdc.metu.edu.tr/webpage/projects/ride/deliverables/RIDE-D2.2.1-standards-09.doc

[23] G. Grieve, W. T. Klein, G. W. Beeler, J. Landgrebe, H. Solbrig, A. Honey, L. McKenzie, D. Nelson, C. Parker, and I. Singureanu, "Core Principles and Properties of HL7 Version 3 Models," May 2008. [Online]. Available: http://www.hl7.org/v3ballot2011may/html/infrastructure/coreprinciples/v3modelcoreprinciples.html

[24] K. A. Spackman, "SNOMED CT Style Guide: Situations with Explicit Context, Style Guide 14 January 2008, Version 1.03," SNOMED CT, Tech. Rep., Jan. 2008. [Online]. Available: http://www.ihtsdo.org/fileadmin/user_upload/Docs_01/Copenhagen_Apr_2008/SNOMED_CT_Style_Guides/IHTSDO_Modeling_StyleGuide-Situations-20080114_v1-03.pdf

[25] A. L. Rector, J. E. Rogers, and P. A. Pole, "The GALEN High Level Ontology." IOS Press, Jan. 1996, pp. 174–178.

[26] K. A. Spackman, K. E. Campbell, and R. A. Cote, "SNOMED RT: A reference terminology for health care," in *Journal of the American Medical Informatics Association*, 1997, pp. 640–644.

[27] M. Uschold, "Building Ontologies: Towards a Unified Methodology," in *16th Annual Conference of the British Computer Society Specialist Group on Expert Systems*, 1996, pp. 16–18.

[28] M. Fernández-López, A. Gómez-Pérez, and N. Juristo, "Methontology: from ontological art towards ontological engineering," in *Proceeding Symposium on Ontological Engineering of AAAI*, 1997.

[29] S. Staab, R. Studer, H.-P. Schnurr, and Y. Sure, "Knowledge Processes and Ontologies," *IEEE Intelligent Systems*, vol. 16, pp. 26–34, January 2001.

[30] H. S. Pinto, S. Staab, and C. Tempich, "DILIGENT: Towards a fine-grained methodology for Distributed, Loosely-controlled and Evolving Engineering of Ontologies," in *16th European Conference on AI (ECAI)*. IOS Press, 2004, pp. 393–397.

[31] J. Euzenat and P. Shvaiko, *Ontology Matching*. Heidelberg (DE): Springer-Verlag, 2007.

[32] I. Horrocks, "Scalable ontology-based information systems," in *Invited Talk at 13th International Conference on Extending Database Technology (EDBT)*, 2010, p. 2. [Online]. Available: http://www.edbt.org/Proceedings/2010-Lausanne/edbt/papers/p0002-Horrocks.pdf

[33] J. Bao, "Representing and reasoning with modular ontologies," Ph.D. dissertation, Iowa State University, Ames, Iowa (USA), 2007. [Online]. Available: http://www.cs.iastate.edu/~baojie/acad/JieBaoDissertation.pdf

[34] R. Sahay, R. Fox, and M. Hauswirth, "Ontologising Interaction Behavior for Service-Oriented Enterprise Healthcare Integration," in *ECOWS 2008 - Proceedings of the Sixth European Conference on Web Services, 12-14 November, Dublin, Ireland*. IEEE Computer Society, 2008, pp. 165–174.

[35] R. Sahay, W. Akhtar, and R. Fox, "Semantic Service-oriented Design and Development Methodology for Enterprise Healthcare Integration," in *WEBIST 2009 - Proceedings of the Fifth International Conference on Web Information Systems and Technologies, Lisbon, Portugal, March 23-26*. INSTICC Press, 2009.

[36] R. Sahay, R. Fox, A. Zimmermann, A. Polleres, and M. Hauswirth, "A Methodological Approach for Ontologising and Aligning Health Level Seven (HL7) Applications," in *ARES 2011 - Proceedings of the Availability, Reliability and Security for Business, Enterprise and Health Information Systems, Vienna, Austria, August 22-26*. Springer LNCS Series, 2011, pp. 102–117.

[37] A. Zimmermann, R. Sahay, R. Fox, and A. Polleres, "Heterogeneity and Context in Semantic-Web-Enabled HCLS Systems," in *ODBASE 2009 - The 8th International Conference on Ontologies, DataBases, and Applications of Semantics, Vilamoura, Portugal, November 1-6*, ser. Lecture Notes in Computer Science, R. Meersman, T. Dillon, and P. Herrero, Eds., vol. 5871. Springer, Nov. 2009, pp. 1165–1182.

[38] R. Sahay, A. Zimmermann, R. Fox, A. Polleres, and M. Hauswirth, *A Formal Investigation of Semantic Interoperability of HCLS Systems*, M.-A. Sicilia and P. S. Balazote, Eds. IGI Global (To appear), 2012.

[39] R. Sahay, W. Akhtar, and R. Fox, "PPEPR: Plug and Play Electronic Patient Records," in *Proceedings of the 2008 ACM Symposium on Applied Computing (SAC), Fortaleza, Ceara, Brazil, March 16-20, 2008*, R. L. Wainwright and H. Haddad, Eds. ACM Press, Mar. 2008, pp. 2298–2304. [Online]. Available: http://doi.acm.org/10.1145/1363686.1364232

[40] R. Fox, R. Sahay, and M. Hauswirth, "PPEPR for Enterprise Healthcare Integration," in *Proceedings of the eHealth 2008 Conference*. City University, London EC1: Springer CCIS, 9 2008.

[41] P. Schloeffel, T. Beale, G. Hayworth, S. Heard, and H. Leslie, "The relationship between CEN 13606, HL7, and openEHR," in *Health Informatics Conference*, Sidney, Australia, 2006. [Online]. Available: http://www.oceaninformatics.com/Media/docs/Relationship-between-CEN-13606-HL7-CDA--openEHR-2ba3675f-2136-4069-ac5c-152139c70bd0.pdf

[42] A. W. Forrey, C. J. McDonald, G. DeMoor, S. M. Huff, D. Leavelle, D. Leland, T. Fiers, L. Charles, B. Griffin, F. Stalling, A. Tullis, K. Hutchins, and J. Baenziger, "Logical Observation Identifier Names and Codes (LOINC) database: a public use set of codes and names for electronic reporting of clinical laboratory test results." *Clinical chemistry*, no. 1, pp. 81–90, January 1996.

[43] T. Berners-Lee, J. Hendler, and O. Lassila, "The Semantic Web," *Scientific American*, vol. 284, no. 5, pp. 34–43, 2001. [Online]. Available: http://www.scientificamerican.com/article.cfm?id=the-semantic-web

[44] J. Quinn, "HL7 Version Messaging Standard Version 2.5: An Application Protocol for Electronic Data Exchange in Healthcare Environment(Chapter 1))," 2003. [Online]. Available: http://www.hl7.org/implement/standards/v2messages.cfm

[45] G. Alonso, F. Casati, H. Kuno, and V. Machiraju, *Web Services: Concepts, Architecture and Applications*. Springer Verlag, 2004.

[46] G. Schadow, P. Biron, L. McKenzie, G. Grieve, and D. Pratt, "Data Types - Abstract Specification," May 2011, Version 3 Ballot. [Online]. Available: http://www.hl7.org/v3ballot2011may/html/infrastructure/datatypes/datatypes.html

[47] S. Liu, W. Ma, R. Moore, V. Ganesan, and S. J. Nelson, "RxNorm: Prescription for Electronic Drug Information Exchange," *IT Professional*, vol. 7, no. 5, pp. 17–23, 2005.

[48] A. L. Rector, R. Qamar, and T. Marley, "Binding ontologies and coding systems to electronic health records and messages," *Applied Ontologies*, vol. 4, no. 1, pp. 51–69, 2009.

[49] S. Heymans, M. McKennirey, and J. Phillips, "Semantic validation of the use of SNOMED CT in HL7 clinical documents," *Journal of Biomedical Semantics*, vol. 2, no. 1, p. 2, 2011.

[50] C. Tao, G. Jiang, W. Wei, H. R. Solbrig, and C. G. Chute, "Towards Semantic-Web Based Representation and Harmonization of Standard Meta-data Models for Clinical Studies," *AMIA 2011*, vol. 18, pp. 59–63, 2011.

[51] T. Berners-Lee, "A Unifying Syntax for the Expression of Names and Addresses of Objects on the Network as used in the World-Wide Web," Tech. Rep., 1994. [Online]. Available: http://tools.ietf.org/html/rfc1630

[52] M. Duerst and M. Suignard, "Internationalized Resource Identifiers (IRIs)," W3C and Microsoft Corporation, Tech. Rep., Jan. 2005. [Online]. Available: http://tools.ietf.org/html/rfc3987

[53] The Unicode Consortium, "The Unicode Standard," Unicode Consortium, Mountain View, CA, Tech. Rep. Version 6.0.0, 2011. [Online]. Available: http://www.unicode.org/versions/Unicode6.0.0/

[54] F. Manola and E. Miller, "RDF Primer W3C Recommendation 10 February 2004," World Wide Web Consortium (W3C), Tech. Rep., Feb. 10 2004. [Online]. Available: http://www.w3.org/TR/2004/REC-rdf-primer-20040210/

[55] E. Prud'hommeaux and A. S. (eds.), "SPARQL Query Language for RDF," Tech. Rep., Jan. 2008, W3C Recommendation. [Online]. Available: http://www.w3.org/TR/rdf-sparql-query/

[56] D. Brickley and R. V. Guha, "RDF Vocabulary Description Language 1.0: RDF Schema, W3C Recommendation 10 February 2004," World Wide Web Consortium (W3C), Tech. Rep., Feb. 10 2004. [Online]. Available: http://www.w3.org/TR/2004/REC-rdf-schema-20040210/

[57] P. Hitzler, M. Krötzsch, B. Parsia, P. Patel-Schneider, and S. Rudolph, "OWL 2 Web Ontology Language Primer, W3C Recommendation," World Wide Web Consortium (W3C), Tech. Rep., Oct. 27 2009.

[58] H. Boley, G. Hallmark, M. Kifer, A. Paschke, A. Polleres, and D. Reynolds, "RIF Core Dialect," Tech. Rep., 2010, W3C Recommendation. [Online]. Available: http://www.w3.org/TR/rif-core/

[59] J. de Bruijn, H. Lausen, A. Polleres, and D. Fensel, "The Web Service Modeling Language WSML: An Overview," Digital Enterprise Research Institute, Tech. Rep., Jun. 16 2005. [Online]. Available: http://www.wsmo.org/wsml/wsml-resources/deri-tr-2005-06-16.pdf

[60] P. Hayes, "RDF Semantics, W3C Recommendation 10 February 2004," World Wide Web Consortium (W3C), Tech. Rep., Feb. 10 2004. [Online]. Available: http://www.w3.org/TR/2004/REC-rdf-mt-20040210/

[61] D. Beckett and T. Berners-Lee, "Turtle - Terse RDF Triple Language, W3C Team Submission 14 January 2008," World Wide Web Consortium (W3C), Tech. Rep., Jan. 14 2008. [Online]. Available: http://www.w3.org/TeamSubmission/2008/SUBM-turtle-20080114/

[62] J. Broekstra, A. Kampman, and F. van Harmelen, "Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema," in *Proceedings of the first International Semantic Web Conference (ISWC 2002)*, ser. Lecture Notes in Computer Science, vol. 2342. Sardinia, Italy: Springer Verlag, May 2002, pp. 54–68.

[63] B. McBride, "Jena: Implementing the RDF Model and Syntax Specification," Tech. Rep., 2001.

[64] Virtuoso, "Virtuoso Universal Server." [Online]. Available: http://virtuoso.openlinksw.com/

[65] BigData, "Bigdata RDF database." [Online]. Available: http://www.bigdata.com/

[66] OWLIM, "OWLIM Semantic Repository." [Online]. Available: http://www.ontotext.com/owlim

[67] T. Neumann and G. Weikum, "The RDF-3X engine for scalable management of RDF data," *The VLDB Journal*, vol. 19, pp. 91–113, February 2010.

[68] W3C, "SPARQL New Features and Rationale," July 2009. [Online]. Available: http://www.w3.org/TR/sparql-features/

[69] M. Dean and G. Schreiber, "OWL Web Ontology Language Reference, W3C Recommendation 10 February 2004," World Wide Web Consortium (W3C), Tech. Rep., Feb. 10 2004. [Online]. Available: http://www.w3.org/TR/2004/REC-owl-ref-20040210/

[70] M. Horridge and P. F. Patel-Schneider, "OWL 2 Web Ontology Language Manchester Syntax, W3C Working Group Note," World Wide Web Consortium (W3C), Tech. Rep., Oct. 27 2009. [Online]. Available: http://www.w3.org/TR/2009/NOTE-owl2-manchester-syntax-20091027/

[71] C. Golbreich and E. K. Wallace, "OWL 2 Web Ontology Language New Features and Rationale, W3C Recommendation," World Wide Web Consortium (W3C), Tech. Rep., Oct. 27 2009. [Online]. Available: http://www.w3.org/TR/owl-new-features/

[72] K. A. Spackman, "An Examination of OWL and the Requirements of a Large Health Care Terminology," in *Proceedings of the CEUR Workshop in OWLED*, C. Golbreich, A. Kalyanpur, and B. Parsia, Eds., vol. 258. CEUR-WS.org, 2007. [Online]. Available: http://ceur-ws.org/Vol-258/paper31.pdf

[73] B. Motik, B. C. Grau, I. Horrocks, Z. Wu, A. Fokoue, and C. Lutz, "OWL 2 Web Ontology Language Profiles, W3C Recommendation," World Wide Web Consortium (W3C), Tech. Rep., Oct. 27 2009. [Online]. Available: http://www.w3.org/TR/owl2-profiles/

[74] J. de Bruijn, C. Bussler, J. Domingue, D. Fensel, M. Hepp, U. Keller, M. Kifer, B. König-Ries, J. Kopecky, R. Lara, H. Lausen, E. Oren, A. Polleres, D. Roman, J. Scicluna, and M. Stollberg, "Web Service Modeling Ontology (WSMO)," Jun. 2005, W3C Member Submission. [Online]. Available: http://www.w3.org/Submission/WSMO/

[75] N. Steinmetz and J. de Bruijn, "WSML/OWL Mapping," 1 2008, WSML Working draft D37 version 0.1. [Online]. Available: http://www.wsmo.org/TR/d37/v0.1/

[76] F. Fischer and B. Bishop, "Defining the Features of the WSML-Core v2.0 Language," Tech. Rep., 2009, SOA4All Deliverable D3.1.2. [Online]. Available: http://www.soa4all.eu/docs/D3.1.2%20Defining%20the%20features%20of%20the%20WSML-Core%20v2.0%20language.pdf

[77] B. Bishop, F. Fischer, P. Hitzler, M. Krötzsch, S. Rudolph, Y. Trimponias, and G. Unel, "Defining the Features of the WSML-DL v2.0 Language," Tech. Rep., 2009, SOA4All Deliverable D3.1.3. [Online]. Available: http://www.soa4all.eu/docs/D3.1.3%20Defining%20the%20Features%20WSML-DL.pdf

[78] I. Toma, B. Bishop, and F. Fischer, "Defining the features of the WSML-Rule v2.0 language ," Tech. Rep., 2009, SOA4All Deliverable D3.1.4. [Online]. Available: http://www.soa4all.eu/docs/D3.1.4%20Defining%20the%20features%20of%20the%20WSML-Rule%20v2.0%20language.pdf

[79] I. Horrocks, P. F. Patel-Schneider, and F. V. Harmelen, "From SHIQ and RDF to OWL: The Making of a Web Ontology Language," *Journal of Web Semantics*, vol. 1, p. 2003, 2003.

[80] F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel-Schneider, Eds., *The Description Logic Handbook: Theory, Implementation, and Applications.* Cambridge University Press, 2003.

[81] A. Gómez-Pérez, O. Corcho, and M. F. López, *Ontological Engineering : with examples from the areas of Knowledge Management, e-Commerce and the Semantic Web. First Edition (Advanced Information and Knowledge Processing).* Springer, Jul. 2004.

[82] M. F. López, A. Gómez-Pérez, and M. D. R. Amaya, "Ontology's Crossed Life Cycles," in *Proceedings of the 12th European Workshop on Knowledge Acquisition, Modeling and Management*, ser. EKAW '00. London, UK: Springer-Verlag, 2000, pp. 65–79.

[83] E. Simperl, "Reusing ontologies on the Semantic Web: A feasibility study," *Data Knowledge Engineering*, vol. 68, pp. 905–925, October 2009.

[84] D. Lindberg, B. Humphreys, and A. McCray, "The Unified Medical Language System," *Methods of Information in Medicine*, vol. 32, no. 4, pp. 281–291, 1993.

[85] B. C. Grau, I. Horrocks, Y. Kazakov, and U. Sattler, "Just the right amount: extracting modules from ontologies," in *Proceedings of the 16th International Conference on World Wide Web*, ser. WWW07. New York, NY, USA: ACM, 2007, pp. 717–726.

[86] V. Presutti and A. Gangemi, "Content Ontology Design Patterns as Practical Building Blocks for Web Ontologies," in *Proceedings of the 27th International Conference on Conceptual Modeling*, ser. ER '08. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 128–141.

[87] A. J. Yepes, E. J. Ruiz, R. B. Llavori, and D. R. Schuhmann, "Reuse of terminological resources for efficient ontological engineering in Life Sciences," *BMC Bioinformatics*, vol. 10, no. 10, pp. S4+, 2009.

[88] O. Corcho, M. Fernández-López, and A. Gómez-Pérez, "Methodologies, tools and languages for building ontologies: where is their meeting point?" *Data Knowledge Engineering*, vol. 46, pp. 41–64, July 2003.

[89] D. B. Lenat and R. V. Guha, *Building Large Knowledge-Based Systems; Representation and Inference in the Cyc Project*, 1st ed. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1990.

[90] M. Uschold and M. King, "Towards a Methodology for Building Ontologies," in *In Workshop on Basic Ontological Issues in Knowledge Sharing, held in conjunction with IJCAI-95*, 1995.

[91] M. Fernández-López and A. Gómez-Pérez, "Towards a method to conceptualize domain ontologies," in *Proceedings of the ECAI Workshop on Ontological Engineering*, 1997.

[92] R. V. Guha, "Contexts: a Formalization and Some Applications," Ph.D. dissertation, Stanford University, Stanford, CA (USA), 1991. [Online]. Available: http://www-formal.stanford.edu/guha/guha-thesis.ps

[93] J. Pan, L. Serafini, and Y. Zhao, "Semantic Import: An Approach for Partial Ontology Reuse," in *Proceedings of the 1st International Workshop on Modular Ontologies, WoMO'06, co-located with the International Semantic Web Conference, ISWC'06 November 5, 2006, Athens, Georgia, USA*, ser. CEUR Workshop Proceedings, P. Haase, V. G. Honavar, O. Kutz, Y. Sure, and A. Tamilin, Eds., vol. 232. Sun SITE Central Europe (CEUR), Nov. 2007.

[94] J. Bao, D. Caragea, and V. G. Honavar, "A Tableau-Based Federated Reasoning Algorithm for Modular Ontologies," in *2006 IEEE / WIC / ACM International Conference on Web Intelligence (WI 2006), 18-22 December 2006, Hong Kong, China*. IEEE Computer Society, 2006, pp. 404–410.

[95] A. Borgida and L. Serafini, "Distributed Description Logics: Assimilating Information from Peer Sources," *Journal on Data Semantics*, vol. 1, pp. 153–184, 2003.

[96] D. Brickley and L. Miller, "The Friend Of A Friend (FOAF) vocabulary specification," Tech. Rep., November 2007. [Online]. Available: http://xmlns.com/foaf/spec/

[97] S. Kande, N. Jain, T. Briks-Fader, and K. Witting, "IHE IT Infrastructure Technical Framework," Tech. Rep., Jul. 2005, Volume I-II, Revision 2.0. [Online]. Available: http://www.ihe.net/Technical_Framework/upload/IHE_ITI_Suppl_HPD_Rev1-1_TI_2010-08-10.pdf

[98] D. Clunie, "IHE Radiology Technical Framework," Tech. Rep., Jul. 2005, Vol. IIV, Revision 6.0. [Online]. Available: http://www.ihe.net/Technical_Framework/upload/IHE-RAD_TF_Suppl_Basic_Image_Review_2009-06-21.pdf

[99] M. Eichelberg, "Requirements Analysis for the RIDE Roadmap," Tech. Rep., Dec. 2007, Deliverable D2.3.1. [Online]. Available: http://www.srdc.metu.edu.tr/webpage/projects/ride/deliverables/RIDE-D2.3.1-2006-10-13_final.pdf

[100] A. Dogac, "Vision for a Europe-wide Semantically Interoperable eHealth Infrastructure," Tech. Rep., Dec. 2007, Deliverable D3.2.1. [Online]. Available: http://www.srdc.metu.edu.tr/webpage/projects/ride/deliverables/RIDE-D_3_2_1Vision-v1.5final.doc

[101] J. C. Arpírez, O. Corcho, M. Fernández-López, and A. Gómez-Pérez, "WebODE: a scalable workbench for ontological engineering," in *Proceedings of the 1st International Conference on Knowledge Capture*, ser. K-CAP '01. New York, NY, USA: ACM, 2001, pp. 6–13.

[102] M. Kerrigan, A. Mocan, M. Tanler, and D. Fensel, "The Web Service Modeling Toolkit - An Integrated Development Environment for Semantic Web Services," in *The Semantic Web: Research and Applications*, ser. Lecture Notes in Computer Science, E. Franconi, M. Kifer, and W. May, Eds. Springer Berlin / Heidelberg, 2007, vol. 4519, pp. 789–798.

[103] H. Knublauch, R. W. Fergerson, N. F. Noy, and M. A. Musen, "The Protégé OWL Plugin: An Open Development Environment for Semantic Web Applications," in *ISWC 2004*. Berlin, Heidelberg: Springer Berlin / Heidelberg, 2004, vol. 3298, ch. 17, pp. 229–243.

[104] D. J. Schultz, D. E. Nickle, S. M. Burgess, and J. W. Horch, "IEEE Standard for Developing Software Life Cycle Processes," Tech. Rep., 1997, IEEE Standard number 1074-1997. [Online]. Available: http://standards.ieee.org/findstds/standard/1074-1997.html

[105] V. Bicer, G. B. Laleci, A. Dogac, and Y. Kabak, "Artemis message exchange framework: Semantic Interoperability of Exchanged Messages in the Healthcare Domain," *SIGMOD Rec.*, vol. 34, pp. 71–76, September 2005.

[106] P. Schloeffel, T. Beale, G. Hayworth, S. Heard, and H. Leslie, "The relationship between CEN 13606, HL7, and openEHR," in *Health Informatics Conference*, 2006.

[107] T. Beale and S. Heard, "The openEHR Archetype Model: Archetype Definition Language (ADL)," OCEAN Informatics, Tech. Rep., Salvatore T. March 2007. [Online]. Available: http://www.openehr.org/releases/1.0.2/architecture/am/adl2.pdf

[108] B. Orgun, "Interoperability in Heterogeneous Medical Information Systems Using Smart Mobile Agents and HL7 (EMAGS)," Master's thesis, Macquarie University, Australia, 2003.

[109] L. McKenzie, G. W. Beeler, G. Grieve, W. T. Klein, and R. Hamm, "Model Interchange Format (Version 2.2.0)," May 2011, Release 1, Version 3 Ballot. [Online]. Available: http://www.hl7.org/v3ballot/html/infrastructure/mif/mif.html

[110] F. Oemig and B. Blobel, "An Ontology Architecture for HL7 V3: Pitfalls and Outcomes," in *World Congress on Medical Physics and Biomedical Engineering*, 2009.

[111] J. D. Ullman, "Information Integration Using Logical Views," in *Database Theory - ICDT '97, 6th International Conference, Delphi, Greece, January 8-10, 1997, Proceedings*, ser. Lecture Notes in Computer Science, F. N. Afrati and P. G. Kolaitis, Eds., vol. 1186. Springer-Verlag, 1997, pp. 19–40.

[112] I. F. Cruz and H. Xiao, "Ontology Driven Data Integration in Heterogeneous Networks," in *Complex Systems in Knowledge-based Environments*. Springer, 2009, vol. 168, pp. 75–98.

[113] A. G. Perez, "OntoWeb: Ontology-based Information Exchange for Knowledge Management and Electronic Commerce," UPM, Tech. Rep., 2002. [Online]. Available: http://www.cs.toronto.edu/nernst/papers/onto-evaluation-EC.pdf

[114] R. V. Benjamins, D. Fensel, S. Decker, and A. Gomez-Perez, "(KA)2: Building Ontologies for the Internet: a mid-term report," *International Journal of Human-Computer Studies*, vol. 51, no. 3, pp. 687–712, Sep. 1999. [Online]. Available: http://dx.doi.org/10.1006/ijhc.1999.0275

[115] A. Lèger, G. Michel, P. Barrett, S. Gitton, A. Goméz-Pérez, A. Lehtola, K. Mokkila, S. Rodrigez, J. Sallantin, T. Varvarigou, and J. Vinesse, "Ontology Domain Modeling Support for Multi-lingual services in E-Commerce : MKBEEM," in *ECAI'00 Workshop on Applications of Ontologies*, 2000, p. 25.

[116] J. Davies, A. Duke, and Y. Sure, "OntoShare: a knowledge management environment for virtual communities of practice," in *Proceedings of the 2nd International Conference on Knowledge Capture*, ser. K-CAP '03. New York, NY, USA: ACM, 2003, pp. 20–27.

[117] E. Jimnez-Ruiz, B. Grau, U. Sattler, T. Schneider, and R. Berlanga, "Safe and Economic Re-Use of Ontologies: A Logic-Based Methodology and Tool Support," in *The Semantic Web: Research and Applications*, ser. Lecture Notes in Computer Science, S. Bechhofer, M. Hauswirth, J. Hoffmann, and M. Koubarakis, Eds. Springer Berlin / Heidelberg, 2008, vol. 5021, pp. 185–199. [Online]. Available: http://krono.act.uji.es/people/Ernesto/safety-ontology-reuse

[118] B. Adida and M. Birbeck, "RDFa Primer: Bridging the Human and Data Webs," W3C, Tech. Rep., 2008, W3C Working Group Note. [Online]. Available: http://www.w3.org/TR/xhtml-rdfa-primer/

[119] M. Dean, "Java2owl," 2003. [Online]. Available: http://www.daml.org/2003/10/java2owl/

[120] K. Falkovych, M. Sabou, and H. Stuckenschmidt, "UML for the Semantic Web: Transformation-Based Approaches," in *In Knowledge Transformation for the Semantic Web*. IOS Press, 2003.

[121] D. Connolly, F. van Harmelen, I. Horrocks, D. L. McGuinness, P. F. Patel-Schneider, and O. Stepánková, "Annotated DAML+OIL Ontology Markup, W3C Note 18 December 2001," World Wide Web Consortium (W3C), Tech. Rep., Dec. 18 2001. [Online]. Available: http://www.w3.org/TR/daml+oil-walkthru/

[122] R. Colomb, A. Gerber, and M. Lawley, "Issues in Mapping Metamodels in the Ontology Development Metamodel Using QVT," in *The 1st International Workshop on the Model-Driven Semantic Web*, Hong Kong, 2004.

[123] G. Dragan, D. Dragan, D. Vladan, and D. Violeta, "Converting UML to OWL ontologies," in *Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters*. New York, NY, USA: ACM, 2004, pp. 488–489.

[124] S. Cranefield, "Networked Knowledge Representation and Exchange using UML and RDF," *Journal of Digital Information*, vol. 1, no. 8, February 2001.

[125] H. Knublauch, D. Oberle, P. Tetlow, E. Wallace, J. Z. Pan, and M. Uschold, "A Semantic Web Primer for Object-Oriented Software Developers," Salvatore T. March 2006, W3C Working Group Note. [Online]. Available: http://www.w3.org/TR/sw-oosd-primer/

[126] P. Tetlow, J. Z. Pan, D. Oberle, E. Wallace, M. Uschold, and E. Kendall, "Ontology Driven Architectures and Potential Uses of the Semantic Web in Systems and Software Engineering," 2006, W3C Working Draft. [Online]. Available: http://www.w3.org/2001/sw/BestPractices/SE/ODA/

[127] S. Cranefield, "UML and the Semantic Web," in *The Emerging Semantic Web*, I. F. Cruz, S. Decker, J. Euzenat, and D. L. McGuinness, Eds., vol. 75. IOS press, 2001.

[128] H. Hosoya and B. C. Pierce, "XDuce: A statically typed XML processing language," *ACM Transaction Internet Technology*, vol. 3, pp. 117–148, May 2003.

[129] C. Brabrand, A. Møller, and M. I. Schwartzbach, "Dual syntax for XML languages," *Information System*, vol. 33, pp. 385–406, June 2008.

[130] J. J. Carroll and P. Stickler, "TriX: RDF Triples in XML," HP Labs, Tech. Rep. HPL-2004-56, 5 2004. [Online]. Available: http://www.hpl.hp.com/techreports/2004/HPL-2004-56.pdf

[131] W. Akhtar, J. Kopecký, T. Krennwallner, and A. Polleres, "XSPARQL: traveling between the XML and RDF worlds - and avoiding the XSLT pilgrimage," ser. ESWC'08. Springer-Verlag, 2008, pp. 432–447.

[132] S. Kawanaka and H. Hosoya, "biXid: A bidirectional transformation language for XML," *SIGPLAN Not.*, vol. 41, pp. 201–214, September 2006.

[133] M. Klein, "Interpreting XML Documents via an RDF Schema Ontology," in *Proceedings of the 13th International Workshop on Database and Expert Systems Applications*. IEEE Computer Society, 2002, pp. 889–894.

[134] B. Amann, C. Beeri, I. Fundulaki, and M. Scholl, "Ontology-based integration of XML Web resources," in *Proceedings of the International Semantic Web Conference (ISWC)*, 2002, pp. 117–131.

[135] I. F. Cruz, H. Xiao, and F. Hsu, "An Ontology-Based Framework for XML Semantic Integration," in *Proceedings of the International Database Engineering and Applications Symposium*. Washington, DC, USA: IEEE Computer Society, 2004, pp. 217–226.

[136] P. Patel-Schneider and J. Siméon, "The Yin/Yang web: XML syntax and RDF semantics," in *Proceedings of the 11th international conference on World Wide Web*. New York, NY, USA: ACM, 2002, pp. 443–453.

[137] L. Feng, E. Chang, and T. S. Dillon, "Schemata transformation of object-oriented conceptual models to XML," *Computer System Science and Engineering*, vol. 18, no. 1, pp. 45–60, 2003.

[138] H. Bohring and S. Auer, "Mapping XML to OWL Ontologies," in *Leipziger Informatik-Tage*. GI, 2005, pp. 147–156.

[139] T. Rodrigues, P. Rosa, and J. Cardoso, "Mapping XML to Existing OWL Ontologies," in *International Conference WWW/Internet 2006*, P. Isaías, M. B. Nunes, and I. J. Martínez, Eds., 2006, pp. 72–77.

[140] G. Reif, H. Gall, and M. Jazayeri, "WEESA: Web engineering for semantic Web applications," in *Proceedings of the 14th International Conference on World Wide Web*. New York, NY, USA: ACM, 2005, pp. 722–729.

[141] S. Battle, "Round-tripping between XML and RDF," in *Proceedings of the International Semantic Web Conference (ISWC)*. Springer, 2004.

[142] M. Ferdinand, C. Zirpins, and D. Trastour, "Lifting XML Schema to OWL," in *Proceedings of the 4th International Conference on Web Engineering (ICWE)*. Springer Heidelberg, July 2004, pp. 354–358.

[143] S. Boag, D. Chamberlin, M. F. Fernández, D. Florescu, J. Robie, and J. Siméon, "XQuery 1.0: An XML Query Language - W3C Recommendation 23 January 2007," World Wide Web Consortium (W3C), Tech. Rep., Jan. 23 2007. [Online]. Available: http://www.w3.org/TR/2007/REC-xquery-20070123/

[144] S. Decker, S. Melnik, F. v. van Harmelen, D. Fensel, M. C. A. Klein, J. Broekstra, M. Erdmann, and I. Horrocks, "The Semantic Web: The Roles of XML and RDF," *IEEE Internet Computing*, vol. 4, no. 5, pp. 63–74, 2000.

[145] M. Klein, D. Fensel, F. van Harmelen, and I. Horrocks, "The relation between ontologies and XML schemas," in *Electronic Transactions on Artificial Intelligence (ETAI)*, 2001.

[146] P. Lehti and P. Fankhauser, "XML Data Integration with OWL: Experiences and Challenges," in *Symposium on Applications and the Internet (SAINT)*, 2004, pp. 160–170.

[147] V. Levenshtein, "Binary Codes Capable of Correcting Deletions, Insertions and Reversals," *Soviet Physics Doklady*, vol. 10, p. 707, 1966.

[148] D. B. Lenat, "CYC: a large-scale investment in knowledge infrastructure," *Communications of the ACM*, vol. 38, no. 11, pp. 33–38, 1995. [Online]. Available: http://doi.acm.org/10.1145/219717.219745

[149] N. Ian and P. Adam, "Towards a standard upper ontology," in *Proceedings of the International Conference on Formal Ontology in Information Systems*. ACM, 2001.

[150] M. Klein, "Combining and relating ontologies: an analysis of problems and solutions," in *Workshop on Ontologies and Information Sharing, IJCAI'01*, A. Gomez-Perez, M. Gruninger, H. Stuckenschmidt, and M. Uschold, Eds., Seattle, USA, 2001.

[151] N. F. Noy, "Semantic Integration: A survey of ontology-based approaches," *SIGMOD*, vol. 33, pp. 65–70, December 2004.

[152] A. Gangemi, N. Guarino, C. Masolo, A. Oltramari, R. Oltramari, and L. Schneider, "Sweetening Ontologies with DOLCE," in *Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management (EKAW)*. Springer, 2002, pp. 166–181.

[153] A. Mocan and E. Cimpian, "An ontology-based data mediation framework for semantic environments," *International Journal on Semantic Web and Information System*, vol. 3, no. 2, pp. 69–98, 2007. [Online]. Available: http://www.sti-innsbruck.at/fileadmin/documents/articles/ITJ3696_MEDcZfOZcI.pdf

[154] J. Euzenat, C. Meilicke, H. Stuckenschmidt, P. Shvaiko, and C. T. dos Santos, "Ontology Alignment Evaluation Initiative: Six Years of Experience," *Journal on Data Semantics*, vol. 15, pp. 158–192, 2011.

[155] S. Castano, A. Ferrara, and S. Montanelli, "Matching Ontologies in Open Networked Systems: Techniques and Applications," *Journal on Data Semantics*, vol. 3870(V):25–63, 2006.

[156] W. Hu and Y. Qu, "Falcon-AO: A Practical Ontology Matching System," *Journal of Web Semantics*, vol. 6, no. 3, pp. 237–239, 2008.

[157] J. Li, J. Tang, Y. Li, and Q. Luo, "RiMOM: A Dynamic Multistrategy Ontology Alignment Framework," *IEEE Transaction on Knowledge and Data Engineering*, vol. 21, no. 8, pp. 1218–1232, 2009.

[158] P. Jain, P. Hitzler, A. P. Sheth, K. Verma, and P. Z. Yeh, "Ontology Alignment for Linked Open Data," in *Proceedings of the 9th International Semantic Web Conference (ISWC)*. Springer-Verlag, 2010, pp. 402–417.

[159] I. F. Cruz, F. P. Antonelli, and C. Stroe, "AgreementMaker: efficient matching for large real-world schemas and ontologies," *Proceedings of the VLDB Endowment*, vol. 2, pp. 1586–1589, August 2009. [Online]. Available: http://dl.acm.org/citation.cfm?id=1687553.1687598

[160] J. Euzenat, "An API for Ontology Alignment," in *Proceedings of the International Semantic Web Conference (ISWC04)*. Springer-Verlag, 2004, pp. 698–712.

[161] F. Scharffe and J. de Bruijn, "A Language to Specify Mappings Between Ontologies," in *Proceedings of the 1st International Conference on Signal-Image Technology and Internet-Based Systems, SITIS 2005, November 27 - December 1, 2005, Yaounde, Cameroon*, R. Chbeir, A. Dipanda, and K. Yétongnon, Eds. Dicolor Press, 2005, pp. 267–271. [Online]. Available: http://www.u-bourgogne.fr/SITIS/05/download/Proceedings/Files/f142.pdf

[162] F. Scharffe and D. Fensel, "Correspondence Patterns for Ontology Alignment," in *Proceedings of the 16th International Conference on Knowledge Engineering: Practice and Patterns*, ser. EKAW '08. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 83–92.

[163] J. David, J. Euzenat, F. Scharffe, and C. T. dos Santos, "The Alignment API 4.0," *Semantic Web*, vol. 2, no. 1, pp. 3–10, 2011.

[164] S. Castano, A. Ferrara, and S. Montanelli, "H-MATCH: an Algorithm for Dynamically Matching Ontologies in Peer-based Systems," in *In the first International Workshop on Semantic Web and Databases (SWDB), Co-located with VLDB 2003*, 2003, pp. 231–250. [Online]. Available: http://www.cs.uic.edu/~ifc/SWDB/papers/Castano_etal.pdf

[165] V. V. Raghavan and S. K. M. Wong, "A critical analysis of vector space model for information retrieval," *Journal of the American Society for Information Science*, vol. 37, no. 5, pp. 279–287, 1986.

[166] A. Polleres, F. Scharffe, and R. Schindlauer, "SPARQL++ for mapping between RDF vocabularies," in *ODBASE07*. Springer, 2007, pp. 878–896.

[167] J. Euzenat, A. Polleres, and F. Scharffe, "SPARQL Extensions for Processing Alignments," *IEEE Intelligent Systems*, vol. 23, no. 6, pp. 82–84, Nov. 2008.

[168] T. Vitvar, A. Mocan, M. Kerrigan, M. Zaremba, M. Zaremba, M. Moran, E. Cimpian, T. Haselwanter, and D. Fensel, "Semantically-enabled service oriented architecture: concepts, technology and application," *Service Oriented Computing and Applications*, vol. 1, no. 2, pp. 129–154, 5 2007.

[169] A. Haller, E. Cimpian, A. Mocan, E. Oren, and C. Bussler, "WSMX - A Semantic Service-Oriented Architecture," in *Proceedings of the 3rd International Conference on Web Services*. Orlando, Florida, USA: IEEE Computer Society, 2005, pp. 321 – 328.

[170] G. Yang, M. Kifer, H. Wan, and C. Zhao, "Flora-2 : Users Manual," Tech. Rep., 2008. [Online]. Available: http://flora.sourceforge.net/docs/floraManual.pdf

[171] R. Fox, J. Cooley, and M. Hauswirth, "Creating a Virtual Personal Health Record Using Mashups," *IEEE Internet Computing*, vol. 15, no. 4, pp. 23–30, 2011.

[172] R. V. Guha, R. McCool, and R. Fikes, "Contexts for the Semantic Web," in *Proceedings of the International Semantic Web Conference (ISWC04)*. Springer-Verlag, 2004, pp. 32–46.

[173] V. Akman and M. Surav, "Steps Toward Formalizing Context," *AI Magazine*, vol. 17, no. 3, pp. 55–72, 1996.

[174] L. Serafini, H. Stuckenschmidt, and H. Wache, "A Formal Investigation of Mapping Languages for Terminological Knowledge," in *IJCAI-05, Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence*, August 2005, pp. 576–581. [Online]. Available: http://www.dit.unitn.it/~p2p/RelatedWork/Matching/lucianoijcai05.pdf

[175] B. Cuenca-Grau and O. Kutz, "Modular Ontology Languages Revisisted," in *SWeCKa 2007: Proceedings of the IJCAI-2007 Workshop on Semantic Web for Collaborative Knowledge Acquisition, Hyderabad, India, January 7, 2007*, V. G. Honavar, T. Finin, D. Caragea, D. Mladenic, and Y. Sure, Eds., 2007.

[176] R. Guha, "Contexts: a Formalization and Some Applications," Stanford Computer Science Department, Stanford, California, Tech. Rep. STAN-CS-91-1399, 1991.

[177] A. K. Dey, "Understanding and Using Context," *Personal Ubiquitous Computing*, vol. 5, no. 1, pp. 4–7, 2001.

[178] J. McCarthy, "Generality in Artificial Intelligence," *Communications of the ACM*, vol. 30, no. 12, pp. 1029–1035, 1987.

[179] J. D. Kleer, "An Assumption-Based TMS," *Artificial Intelligence*, vol. 28, no. 2, pp. 127–162, 1986.

[180] D. B. Lenat, "CYC: a large-scale investment in knowledge infrastructure," *Communications of the ACM*, vol. 38, no. 11, pp. 33–38, 1995. [Online]. Available: http://doi.acm.org/10.1145/219717.219745

[181] F. Giunchiglia, "Contextual Reasoning," *Epistemologica*, vol. 16, pp. 345–364, 1993.

[182] P. Bouquet, F. Giunchiglia, F. van Harmelen, L. Serafini, and H. Stuckenschmidt, "C-OWL: Contextualizing Ontologies," in *The Semantic Web - ISWC 2003, Second International Semantic Web Conference, Sanibel Island, FL, USA, October 20-23, 2003, Proceedings*, ser. Lecture Notes in Computer Science, D. Fensel, K. P. Sycara, and J. Mylopoulos, Eds., vol. 2870. Springer-Verlag, Oct. 20 2003, pp. 164–179. [Online]. Available: citeseer.ist.psu.edu/bouquet03cowl.html

[183] R. Stevens, P. Baker, S. Bechhofer, G. Ng, A. Jacoby, N. Paton, C. Goble, and A. Brass, "TAMBIS: Transparent Access to Multiple Bioinformatics Information Sources," *Bioinformatics*, vol. 16, no. 2, pp. 184–186, 2000. [Online]. Available: http://www.cs.man.ac.uk/~stevensr/papers/application-note.ps

[184] I. Horrocks, B. Parsia, and U. Sattler, "OWL 2 Web Ontology Language Direct, Semantics W3C Recommendation," World Wide Web Consortium (W3C), Tech. Rep., Oct. 27 2009. [Online]. Available: http://www.w3.org/TR/2009/REC-owl2-direct-semantics-20091027/

[185] M. Schneider, "OWL 2 Web Ontology Language RDF-Based Semantics W3C Recommendation 27 October 2009," World Wide Web Consortium (W3C), Tech. Rep., Oct. 27 2009. [Online]. Available: http://www.w3.org/TR/2009/REC-owl2-rdf-based-semantics-20091027/

[186] L. Serafini and A. Tamilin, "DRAGO: Distributed Reasoning Architecture for the Semantic Web," in *The Semantic Web: Research and Applications, Second European Semantic Web Conference, ESWC 2005, Heraklion, Crete, Greece, May 29 - June 1, 2005, Proceedings*, ser. Lecture Notes in Computer Science, A. Gomez-Perez and J. Euzenat, Eds., vol. 3532. Springer-Verlag, May 2005, pp. 361–376.

[187] M. Homola, "Distributed Description Logics Revisited," in *Proceedings of the 20th International Workshop on Description Logics DL'07*. Bolzano University Press, Jun. 2007. [Online]. Available: http://ceur-ws.org/Vol-250/paper_51.pdf

[188] A. Zimmermann, "Integrated Distributed Description Logics," in *Proceedings of the 20th International Workshop on Description Logics DL'07*. Bolzano University Press, Jun. 2007, pp. 507–514. [Online]. Available: http://ftp.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-250/paper_37.pdf

[189] A. Zimmermann and C. LeDuc, "Reasoning on a Network of Aligned Ontologies," in *Web Reasoning and Rule Systems, Second International Conference, RR 2008, Karlsruhe, Germany, October/November 2008, Proceedings*, ser. Lecture Notes in Computer Science, D. Calvanese and H. Lausen, Eds., vol. 5341. Springer-Verlag, Oct. 2008, pp. 43–57.

[190] O. Kutz, C. Lutz, F. Wolter, and M. Zakharyaschev, "$\mathcal{E}$-connections of abstract description systems," *Artificial Intelligence*, vol. 156, no. 1, pp. 1–73, 2004.

[191] E. Watkins and D. Nicole, "Named graphs as a mechanism for reasoning about provenance," in *Frontiers of WWW Research and Development - APWeb 2006, 8th Asia-Pacific Web Conference, Harbin, China, January 16-18, 2006, Proceedings*, ser. Lecture Notes in Computer Science, X. Zhou, J. Li, H. T. Shen, M. Kitsuregawa, and Y. Zhang, Eds., vol. 3841. Springer-Verlag, 2006, pp. 943–948.

[192] H. Stoermer, P. Bouquet, I. Palmisano, and D. Redavid, "A Context-Based Architecture for RDF Knowledge Bases: Approach, Implementation and Preliminary Results," in *Web Reasoning and Rule Systems, First International Conference, RR 2007, Innsbruck , Austria, June 7-8, 2007, Proceedings*, ser. Lecture Notes in Computer Science, M. Marchiori and J. P. and, Eds., vol. 4524. Springer-Verlag, Jun. 2007, pp. 209–218.

[193] J. J. Carroll, C. Bizer, P. Hayes, and P. Stickler, "Named graphs, provenance and trust," in *Proceedings of the 14th International Conference on World Wide Web, WWW 2005, Chiba, Japan, May 10-14, 2005*, and, Ed. ACM Press, May 2007, pp. 613–622.

[194] R. Cyganiak, A. Harth, and A. Hogan, "N-Quads: Extending N-Triples with Context," DERI, NUI Galway, Ireland and AIFB, University of Karlsruhe, Germany, Tech. Rep., 2009, public Draft. [Online]. Available: http://sw.deri.org/2008/07/n-quads/

[195] T. Berners-Lee, "Closed World Machine," 2000. [Online]. Available: http://www.w3.org/2000/10/swap/doc/cwm.html

[196] C. Lutz and U. Sattler, "Mary Likes all Cats," in *Proceedings of the 2000 International Workshop on Description Logics (DL2000), Aachen, Germany, August 17-19, 2000*, ser. CEUR Workshop Proceedings, F. Baader and U. Sattler, Eds., vol. 33. Sun SITE Central Europe (CEUR), Aug. 2000, pp. 213–226.

[197] F. M. Donini, D. Nardi, and R. Rosati, "Description logics of minimal knowledge and negation as failure," *ACM Transaction on Computational Logics*, vol. 3, no. 2, pp. 177–225, 2002.

[198] G. Wagner, "Web Rules Need Two Kinds of Negation," in *Principles and Practice of Semantic Web Reasoning, International Workshop, PPSWR 2003, Mumbai, India, December 8, 2003, Proceedings*, ser. Lecture Notes in Computer Science, F. Bry, N. Henze, and J. Maluszynski, Eds., vol. 2901. Springer-Verlag, Dec. 2003, pp. 33–50.

[199] A. Analyti, G. Antoniou, C. V. Damásio, and G. Wagner, "Negation and negative information in the W3C resource description framework," *Annals of Mathematics, Computing and Teleinformatics*, vol. 2, pp. 25–34, 2004.

[200] B. Motik, I. Horrocks, and U. Sattler, "Adding Integrity Constraints to OWL," in *Proceedings of the OWLED 2007 Workshop on OWL: Experiences and Directions, Innsbruck, Austria, June 6-7, 2007*, ser. CEUR Workshop Proceedings, C. Golbreich, A. Kalyanpur, and B. Parsia, Eds., vol. 258. Sun SITE Central Europe (CEUR), Jun. 2007.

[201] A. Analyti, G. Antoniou, C. V. Damásio, and G. Wagner, "Extended RDF as a Semantic Foundation of Rule Markup Languages," *Journal of Artificial Intelligence Research*, vol. 32, pp. 37–94, 2008.

[202] T. Eiter, G. Ianni, T. Krennwallner, and A. Polleres, "Rules and Ontologies for the Semantic Web," in *Reasoning Web, 4th International Summer School 2008, Venice, Italy, September 7-11, 2008, Tutorial Lectures*, ser. Lecture Notes in Computer Science, C. Baroglio, P. A. Bonatti, J. Maluszynski, M. Marchiori, A. Polleres, and S. Schaffert, Eds., vol. 5224. Springer-Verlag, Sep. 2008, pp. 1–53.

[203] J. Euzenat, F. Scharffe, and A. Zimmermann, "Expressive alignment language and implementation," Knowledge Web NoE, Deliverable D2.2.10, 2007. [Online]. Available: ftp://ftp.inrialpes.fr/pub/exmo/reports/kweb-2210.pdf

[204] J. de Bruijn, "RIF RDF and OWL Compatibility, W3C Recommendation," World Wide Web Consortium (W3C), Tech. Rep., June 2010. [Online]. Available: http://www.w3.org/TR/rif-rdf-owl/

[205] C. Meilicke, H. Stuckenschmidt, and A. Tamilin, "Supporting Manual Mapping Revision using Logical Reasoning," in *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence, AAAI 2008, Chicago, Illinois, USA, July 13-17, 2008*, D. Fox and C. P. Gomes, Eds. AAAI Press, Jul. 2008, pp. 1213–1218.

[206] J.-Y. Béziau, W. Carnielli, and D. M. Gabbay, *Handbook of Paraconsistency.* College Publications, 2007.

[207] Z. Huang, F. van Harmelen, and A. ten Teije, "Reasoning with Inconsistent Ontologies," in *IJCAI-05, Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence*, August 2005, pp. 454–459.

[208] C. I. Chesñevar, A. G. Maguitman, and R. P. Loui, "Logical Models of Argument," *ACM Computing Survey*, vol. 32, no. 4, pp. 337–383, 2000.

[209] A. J. García and G. R. Simari, "Defeasible Logic Programming: An Argumentative Approach," *Theory and Practice of Logic Programming*, vol. 4, no. 1–2, pp. 95–138, 2004.

[210] V. Kolovski, B. Parsia, Y. Katz, and J. A. Hendler, "Representing Web Service Policies in OWL-DL," in *The Semantic Web - ISWC 2005: 4th International Semantic Web Conference, Galway, Ireland, November 6-10, 2005, Proceedings*, ser. Lecture Notes in Computer Science, Y. Gil, E. Motta, V. R. Benjamins, and M. A. Musen, Eds., vol. 3729. Springer-Verlag, Nov. 2005, pp. 461–475.

[211] T. Vitvar, J. Kopecký, J. Viskova, and D. Fensel, "WSMO-Lite Annotations for Web Services," in *The Semantic Web: Research and Applications, 5th European Semantic Web Conference, ESWC 2008, Tenerife, Canary Islands, Spain, June 1-5, 2008, Proceedings*, S. Bechhofer, M. Hauswirth, J. Hoffmann, and M. Koubarakis, Eds., vol. 5021. Springer-Verlag, Jun. 2008, pp. 674–689. [Online]. Available: http://cms-wg.sti2.org/doc/ESWC2008-VitvarKVF.pdf

[212] E. Sirin and J. Tao, "Towards Integrity Constraints in OWL," in *Proceedings of the 5th International Workshop on OWL: Experiences and Directions (OWLED)*. CEUR-WS, 2009.

[213] E. Sirin, M. Smith, and E. Wallace, "Opening, Closing Worlds - On Integrity Constraints," in *Proceedings of the Fifth OWLED Workshop on OWL: Experiences and Directions, collocated with the 7th International Semantic Web Conference (ISWC-2008)*. CEUR-WS, 2008.

[214] K. Clark, B. Parsia, M. Grove, E. Sirin, M. Smith, M. Stocker, B. Bulka, H. Pérez-Urbina, P. Oliveira, and P. Klinov, "Pellet Integrity Constraints: Validating RDF with OWL," 2010. [Online]. Available: http://clarkparsia.com/pellet/icv

[215] C. Baral and M. Gelfond, "Logic programming and knowledge representation," *Journal of Logic Programming*, vol. 19/20, p. 73148, 1994.

[216] J. W. Lloyd, *Foundations of logic programming; (2nd extended ed.)*. New York, NY, USA: Springer-Verlag New York, Inc., 1987.

[217] G. Antoniou, C. V. Damsio, B. Grosof, I. Horrocks, M. Kifer, J. Maluszynski, and P. F. Patel-Schneider, "Combining Rules and Ontologies. A survey." Institutionen för datavetenskap, Linköpings Universitetet, deliverable I3-D3, 2005. [Online]. Available: http://idefix.pms.ifi.lmu.de:8080/rewerse/index.html#REWERSE-DEL-2005-I3-D3

[218] D. Wlodzimierz, E. Thomas, I. Giovambattista, K. Thomas, L. Thomas, and M. Jan, "Hybrid reasoning with rules and ontologies," in *Semantic techniques for the web: the REWERSE perspective*. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 1–49.

[219] M. Sintek and S. Decker, "TRIPLE: A query, inference, and transformation language for the Semantic Web," in *Proceedings of the 1st International Semantic Web Conference (ISWC)*. London, UK: Springer-Verlag, 2002, pp. 364–378.

[220] J. D. Roo, "Euler proof mechanism," 2007. [Online]. Available: http://www.agfa.com/w3c/euler/

[221] H. J. ter Horst, "Completeness, decidability and complexity of entailment for RDF Schema and a semantic extension involving the OWL vocabulary," *Journal of Web Semantics*, vol. 3, no. 2-3, pp. 79–115, 2005.

[222] T. Eiter, G. Ianni, R. Schindlauer, and H. Tompits, "dlvhex: A system for integrating multiple semantics in an answer-set programming framework," in *Proceedings 20th Workshop on Logic Programming and Constraint Systems (WLP06)*, 2006, pp. 206–210.

[223] D. I. Moldovan, C. Clark, and S. M. Harabagiu, "Temporal Context Representation and Reasoning," in *IJCAI-05, Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence, Edinburgh, Scotland, UK, July 30-August 5, 2005*, L. P. Kaelbling and A. Saffiotti, Eds. Professional Book Center, May 2005, pp. 1099–1104.

"I have made this letter longer than usual, only because I have not had the time to make it shorter."

–Blaise Pascal