UNIVERSITAT JAUME I

Departament de Llenguatges i Sistemes Informátics

Logic-based support for Ontology Development in Open Environments

Ph. D. dissertation

Ernesto JIMÉNEZ RUIZ

Supervisors

Dr. Rafael BERLANGA LLAVORI

and Dr. Bernardo CUENCA GRAU

Castellón, 24 de junio de 2010

To my parents and sisters.

Preface

Abstract

The benefits of exploiting the underlying logic of ontologies are theoretically proved, although further evidence of their usability is still needed. The research conducted on this dissertation has been based on the hypothesis that logic formalisms can be useful in practice to avoid, detect and repair errors in evolving ontologies. Current logic-based machinery has been adapted and applied to help ontology developers and domain experts in the resolution of several real use cases. We have designed general principles based on logic formalisms and we have implemented a set of prototype systems to support users in different stages of development of ontologies, namely: reuse of knowledge, concurrent evolution, and integration of independent resources.

Methodology

We have designed general principles based on description logic formalisms and we have implemented techniques following those principles. Then, these techniques have been applied in interesting use cases to show the benefits of using logic based methods.

The designed logic-based principles and implemented techniques aim at being integrated within the correspondent development activities of the ontology lifecycle. Thus, they are complementary to state-of-the-art methodologies and workflows to develop ontologies.

Contributions

The goal of this dissertation has been to provide logic-based support in knowledge reuse, concurrent ontology development and ontology integration. Knowledge reuse has been supported with semantic techniques to adapt domain thesauri and to safely reuse knowledge from ontologies. Concurrent changes in collaborative development scenarios have been assessed enhancing current logic-based debugging techniques.

Finally, ontology integration has been audited applying semi-automatic and automatic methods based on designed logic-based principles and state-of-the-art formalisms.

These logic-based methods and techniques have been integrated in a set of prototype systems. Generated results have also been published and disseminated in prestigious international workshops, conferences and journals.

Proposed methods and techniques are complementary to state-of-the-art technology where we have contributed in different ways. Future research lines will involve the integration of our methods with this technology in order to enrich current operation. Further dedication would be also necessary to provide a precise evaluation of our methods in complex and real use cases.

Keywords: knowledge representation, ontology, thesaurus, OWL, description logics, ontology integration, ontology modularization, concurrent ontology development

Prólogo

Resumen

El sentido filosófico del término *ontología* hace referencia a la esencia misma del ser, a su existencia (onto=ser). Para un sistema inteligente lo que existe es lo que puede representarse. En informática una ontología se ha definido como *una conceptualización formal y compartida de un dominio*. Actualmente las ontologías tienen un rol clave en el desarrollo de la Web Semántica, pero también están siendo usadas en otros dominios como la biomedicina, agricultura, defensa, robótica y astronomía.

En la investigación llevada a cabo en esta tesis nos hemos centrado en el diseño y desarrollo de métodos generales y técnicas basadas en la lógica para dar soporte y respaldar el desarrollo de ontologías en varias de sus fases. La biomedicina ha sido seleccionada como dominio de aplicación debido a la gran cantidad de recursos de conocimiento que hay disponibles. Sin embargo, nuestra tecnología puede ser extendida y aplicada en otros dominios.

Cabe destacar que nuestros métodos y técnicas han sido diseñados para el lenguaje de definición de ontologías OWL (Ontology Web Language) y su lógica de descripción subyacente $\mathcal{SROIQ}$; sin embargo, también pueden ser aplicados sobre otros formalismos de definición de ontologías (ej: OBO), siempre y cuando puedan ser expresados en OWL.

Como prueba de concepto hemos implementado un conjunto de herramientas basadas en los métodos y técnicas diseñados en la tesis. Estas herramientas han sido desarrolladas sobre el interfaz *OWL API* y diseñadas para el editor de ontologías Protégé. No obstante podrían ser extendidas e integradas en otros marcos de trabajo como el *toolkit* de NeOn.

Objetivos

Los casos de uso utilizados en la tesis están basados en el dominio de aplicación del proyecto Health-e-Child[1], concretamente en el desarrollo de una ontología sobre

[1] Health-e-Child: `http://www.health-e-child.org/`

la enfermedad denominada *artritis idiopática juvenil* (siglas en inglés: JIA). A dicha ontología la hemos denominado **JIAO**. Cabe destacar, no obstante, que en esta tesis no hemos centrado nuestros esfuerzos en la propia creación de **JIAO** sino en el diseño e implementación de métodos para facilitar su desarrollo. Por tanto, los métodos propuestos pueden ser aplicados a otros dominios.

El conocimiento representado en **JIAO** involucra diferentes granularidades y subdominios como por ejemplo los aspectos genéticos, tratamientos, anatomía, análisis clínicos, etc. Por tanto el desarrollo de ontologías como **JIAO** suele realizarse de forma *colaborativa* (los cambios sobre la ontología son creados, argumentados y reconciliados por un grupo de desarrolladores) y *concurrente* (varios desarrolladores realizan cambios al mismo tiempo). Además, es probable que los desarrolladores no sean expertos en todos los subdominios de **JIAO**, por tanto, siempre que sea posible, es preferible *reutilizar* ontologías y fuentes externas consensuadas por la comunidad. Este conocimiento a reutilizar puede provenir de diferentes ontologías y proyectos independientes, por tanto, tanto el vocabulario utilizado como la conceptualización suelen divergir. La correcta *integración* del conocimiento externo junto con el conocimiento de **JIAO** requiere el establecimiento de correspondencias entre las entidades de las diferentes fuentes a integrar y un análisis de la compatibilidad de las mismas.

Reutilización, *integración* y *concurrencia/colaboración* se presentan como los tres problemas clave para desarrollar ontologías como **JIAO**. En esta tesis partimos de la hipótesis de que la lógica puede ser de gran utilidad para detectar y reparar errores en la evolución de una ontología, y de esta forma dar un soporte lógico al tratamiento de los tres problemas anteriores.

Metodología y desarrollo

Los beneficios del uso de la lógica para dar soporte a la creación y evolución de las ontologías han sido teóricamente probados, sin embargo, una mayor evidencia de su usabilidad en la práctica sigue siendo una de las piedras angulares de este campo.

En esta tesis hemos diseñado una serie de principios generales basados en formalismos lógicos. Estos principios han sido implementados reutilizando y adaptando la tecnología actual. Además, hemos desarrollado un conjunto de sistemas prototipo para dar soporte a los desarrolladores en las diferentes fases de la evolución de una ontología: *reutilización*, *integración* y *colaboración/concurrencia*. Estos prototipos han sido evaluados en varios casos de uso, mostrando los beneficios de usar los métodos y técnicas propuestos.

La tecnología desarrollada tiene como propósito el ser integrada como parte de las actividades del ciclo de vida de una ontología. Por tanto, el trabajo llevado a cabo en esta tesis es totalmente complementario a las metodologías y flujos de trabajo actuales para el desarrollo de ontologías.

Contribuciones

Los métodos y técnicas implementados han sido integrados en el editor de ontologías Protégé. Los resultados generados han sido publicados y divulgados en diversos talleres, conferencias y revistas tanto de ámbito nacional como internacional. Las contribuciones de esta tesis se pueden agrupar en tres grandes grupos: reutilización, desarrollo concurrente e integración.

Contribuciones en la reutilización de conocimiento

El desarrollo de ontologías normalmente requiere reutilizar conocimiento de fuentes externas. La reutilización de vocabulario puede ayudar a proporcionar la etiqueta correcta a un concepto de la ontología . La organización base para los conceptos de una ontología puede ser importada de tesauros como WordNet o UMLS. Finalmente, las ontologías de dominio pueden ser utilizadas como fuente de descripciones lógicas sobre las entidades requeridas.

Nuestra aportación a la reutilización de conocimiento ha sido por partida doble. Por una parte hemos propuesto un método para reutilizar conocimiento de recursos no ontológicos (ej. tesauros). Este método ha consistido en la extracción, adaptación y extensión de porciones de UMLS-Meta, Swissport y Drugbank centradas en el dominio de JIA. Por otra parte, también se ha diseñado e implementado (ProSÉ, una extensión para Protégé) un marco de trabajo para reutilizar ontologías de forma segura y modular. Este marco de trabajo se ha basado en un marco lógico bien fundado, y parte de las siguientes hipótesis: (1) los desarrolladores, en general, no quieren modificar el significado original de las entidades reutilizadas, y (2) la cantidad de conocimiento relevante suele involucrar partes (relativamente) pequeñas de la ontología a reutilizar.

Cabe destacar que ambos métodos no requieren el uso de un razonador y por tanto solo se basan en la estructura del recurso a reutilizar.

Contribuciones en el desarrollo concurrente de ontologías

El control de la evolución de una ontología debe involucrar una gestión de los cambios realizados y sus efectos (propagación a las entidades dependientes). En la literatura podemos encontrar varios trabajos que intentan abordar este problema desde distintos enfoques: representación formal de los cambios, detección y reparación de inconsistencias, y desarrollo colaborativo y concurrente.

Según demuestran ciertos estudios de usuario, los expertos de dominio buscan herramientas funcionales, fáciles de manejar, y que proporcionen técnicas para facilitar la comunicación y discusión entre los diferentes desarrolladores. Adicionalmente también se interesan por herramientas que prevengan, o reparen si es necesario, conflictos y consecuencias lógicas no deseadas causados por cambios concurrentes.

Nuestra contribución a la comunidad ha sido el diseño de un marco de trabajo y la implementación de ContentCVS (un plugin para Protégé) para auditar las consecuencias de cambios realizados de forma concurrente en un entorno de desarrollo colaborativo. Actualmente, ContentCVS permite una edición asíncrona e implementa un método de versionado pasivo. Por tanto, las versiones deben ser comparadas para detectar conflictos potenciales. Las técnicas para la detección de cambios y conflictos están basadas en las nociones de diferencia estructural y semántica (ej. consecuencias lógicas) entre ontologías. ContentCVS requiere del razonador para obtener la diferencia semántica y para utilizar técnicas de reparación basadas en la lógica subyacente de las ontologías. Estas técnicas de reparación guían de forma iterativa la aceptación y rechazo de cambios y consecuencias lógicas no deseadas.

Contribuciones en la integración de ontologías

Actualmente existe un número elevado de técnicas para identificar correspondencias entre ontologías desarrolladas independientemente. No obstante, la integración de las ontologías junto con las correspondencias puede derivar en consecuencias no deseadas y por tanto debe ser auditada. Estas consecuencias no deseadas pueden deberse a correspondencias erróneas o a incompatibilidades en las descripciones, respecto a las entidades alineadas, de las ontologías fuente.

En esta tesis hemos implementado técnicas semiautomáticas y automáticas para evaluar y auditar la integración semántica de ontologías. Las técnicas semiautomáticas han sido creadas para guiar, apoyándose en el razonador, al desarrollador en el análisis y comprensión de las consecuencias lógicas derivadas de la integración. Las técnicas automáticas están basadas en principios lógicos (requieren del razonador pero en menor medida) y tienen como objetivo reducir el número de correspondencias conflictivas y las consecuencias lógicas que son claramente erróneas. En resumen, las técnicas automáticas permiten reducir la cantidad de información a revisar por parte del desarrollador; mientras que las técnicas semiautomáticas permiten guiar la reparación de errores no obvios que necesiten la intervención del desarrollador.

Conclusiones y líneas abiertas

Nuestra tecnología presenta notables puntos fuertes, pero también un conjunto de limitaciones que se plantean como líneas abiertas de investigación. Además, aunque las técnicas y métodos propuestos son complementarios a la tecnología actual, una comparación más detallada respecto a esta tecnología junto con un análisis de las posibilidades de integración formaría también parte del trabajo futuro.

La reutilización de ontologías de forma segura puede ser muy restrictiva cuando los requerimientos de la aplicación implican tanto la especialización como la generalización de la ontología a reutilizar. Por tanto una dirección interesante en el trabajo futuro sería el análisis de otras estrategias de reutilización para complementar y dotar de mayor flexibilidad a la actual.

Otra dirección de investigación interesante implicaría la evaluación del impacto del conocimiento reutilizado sobre el conocimiento local y así analizar si el conocimiento reutilizado es el apropiado. Las técnicas a utilizar serían análogas a las utilizadas en el análisis de cambios concurrentes y en la evaluación de la integración de ontologías.

El marco de trabajo propuesto para el análisis de cambios concurrentes sobre una ontología también podría ser extendido y mejorado. Actualmente no se consideran políticas de acceso ni restricciones en los cambios. Estas características dotarían al sistema de un mayor control sobre la evolución concurrente. Adicionalmente, los planes de reparación de consecuencias lógicas no deseadas podrían ser extendidos para permitir también la modificación de axiomas ya que actualmente sólo sugieren el borrado de los mismos.

Las técnicas para la reparación automática de errores en la integración de ontologías también presentan ciertas limitaciones en lo que respecta al número de correspondencias descartadas. Además, deberían analizarse nuevas heurísticas para poder detectar y borrar el menor conjunto de correspondencias posible.

Respecto a la escalabilidad de nuestros métodos, las técnicas de reutilización de conocimiento, como ya se ha comentado, no necesitan el uso del razonador y sólo requieren realizar consultas sintácticas sobre la ontología . Sin embargo, las técnicas semiautomáticas de reparación de errores, debidos a cambios concurrentes sobre la misma ontología o a la integración de ontologías independientes, requieren el uso del razonador para extraer las justificaciones o conjunto de axiomas causantes de las consecuencias lógicas no deseadas. Consecuentemente, el tiempo necesario para extraer todas las justificaciones para un conjunto muy elevado de consecuencias puede ser prohibitivo. El uso de razonamiento incremental y módulos lógicos, junto con una reparación parcial e iterativa de errores, se presentan como técnicas clave para mejorar la escalabilidad de los métodos propuestos.

Finalmente, un aspecto no considerado en las tres contribuciones de la tesis es la especificación formal de los requerimientos. Esta especificación formal nos permitiría realizar una evaluación más precisa de los métodos propuestos, de la entrada requerida y de los resultados que se deben obtener.

Palabras clave: representación de conocimiento, ontología, tesauro, lógica de descripciones, OWL, integración de ontologías, modularización de ontologías, desarrollo concurrente de ontologías

Agradecimientos

Afortunadamente el trabajo realizado en esta tesis no ha sido únicamente una tarea individual y hay una larga lista de personas que han contribuido de una manera o de otra en ella; por ello me gustaría agradecer su apoyo durante estos años.

En 2004 empezó mi colaboración con el grupo de Bases de Conocimiento Temporal (TKBG) a través del proyecto final de carrera y gracias a Ismael Sanz que me convenció acerca del trabajo que se realizaba en el grupo. Mi doctorado empezó al año siguiente, bajo la supervisión de Rafael Berlanga, quien no sólo ha cumplido con su figura de tutor sino que se ha involucrado totalmente en la evolución de la tesis y sin duda ha sido una pieza clave en el trabajo desarrollado.

El ambiente en el grupo también fue muy importante, y me sentí integrado desde el primer instante. Por ello quiero agradecer a todos los compañeros de trabajo y a los integrantes (pasados y presentes) del TKBG (María José, Lola, Rafa, Ismael, Juanma, Jordi, Roxana, Victoria, María, Henry, Lisette, Aurora) por haber hecho más ameno el día a día en el trabajo. Especialmente me gustaría destacar a Juanma, una de las mejores personas que he conocido tanto en el ámbito profesional como personal, y del que siempre mantendré vivo un gran recuerdo.

Durante el primer año de tesis la investigación sufrió algún altibajo dada la dificultad de definir un rumbo en el que la aportación a la comunidad fuese significativa. Sin embargo varios sucesos ayudaron a ir superando las metas que se iban marcando. A finales de 2005 realicé una estancia de educación/investigación de 3 meses en Concepción (Chile) donde maduré bastante y conocí un magnífico país. En 2006 comenzó mi colaboración más directa con Maat GKnowledge, empresa que ya había colaborado en otras ocasiones con el grupo TKBG, gracias a su director de tecnología Alfonso Ríos y a Claudio Cosín. Nuestra colaboración con Maat GKnowledge se tradujo en la participación en el proyecto europeo Health-e-Child que nos abrió las puertas a un dominio de aplicación perfecto para la tesis. Además nos brindó la oportunidad de ganar experiencia asistiendo a reuniones y presentando nuestro trabajo en las revisiones anuales. A mediados de 2006 obtuve la financiación para realizar el doctorado gracias a una de las becas predoctorales FPI financiadas por la Generalitat Valenciana. Esta beca me proporcionó una estabilidad y tranquilidad necesarias para poder dedicarme exclusivamente a la investigación. También merecen una mención las financiaciones temporales obtenidas a través de la Fundació UJI-Bancaixa y la participación en proyectos CICYT del ministerio.

La obtención de una beca predoctoral también me permitió realizar estancias de investigación en centros extranjeros. La primera estancia se realizó en 2007 en el *European Bioinformatics Institute* (EBI) en Cambridge (Reino Unido), donde tuve la oportunidad de trabajar con Antonio Jimeno y Dietrich Rebholz-Schuhmann durante 3 meses. Antonio Jimeno no solo supervisó y me ayudó en mi trabajo sino que también se convirtió en un bueno amigo. La segunda estancia fue realizada, también durante tres meses y a finales de 2007, en el *Information Management Group* de la Universidad de Manchester bajo la supervisión de Ulrike Sattler y Bernardo Cuenca. Esta estancia fue crucial para el desarrollo de la tesis, y gracias a Uli, Bernardo y a la colaboración de Thomas Schneider conseguí la base teórica necesaria para enriquecer la investigación a realizar. Bernardo mostró además un gran interés en mi tesis y se convirtió en mi co-supervisor. Los dos años sieguientes, 2008 y 2009, volví a realizar sendas estancias con Bernardo pero esta vez en el *Knowledge Representation and Reasoning Group* de la Universidad de Oxford y bajo la valiosa supervisión de Ian Horrocks. Bernardo y Rafa han sido unos supervisores modelo, de los que destaco su ayuda, su cercanía y la relación de amistad que hemos tenido durante estos años y que espero que podamos mantener durante mucho tiempo.

Mi familia no puede quedar fuera de mis agradecimientos, ya que de forma indirecta e incluso en algunos casos de forma directa me ha ayudado significativamente. Mis padres me han apoyado durante toda mi carrera académica y aunque no podían entender gran parte de las cosas que hacía siempre mostraban interés en escucharme. Mis hermanas Mayte y Pili, y mis nuevos hermanos Nando y Toni, también han sido de gran ayuda y siempre me han valorado incluso más de lo que me merecía. Además, no saben cuánto les agradezco sus visitas durante mis estancias en el extranjero.

Respecto a mis amigos, quizá muchos piensen no tienen cabida en estos agradecimientos y que incluso han perjudicado más que beneficiado en esta tesis; pero han sido en gran medida también los culpables de que me sintiese valorado y arropado. Creo que siguen confiando más en mi de lo que lo hago yo.

Finalmente, también me gustaría agradecer la disponibilidad de los miembros del tribunal (Sven Casteleyn, Oscar Pastor, Eduardo Mena, Ismael Sanz y María José Aramburu) y de los revisores de la tesis (Asunción Gómez, Ian Horrocks y Dietrich Rebholz-Schuhmann). Sus comentarios han sido muy importantes para la correcta finalización de la tesis.

Financiación

El trabajo llevado a cabo ha sido financiado principalmente por la beca FPI de la Generalitat Valenciana (BFPI06/372) y parcialmente por el acuerdo de colaboración entre el grupo TKBG y la empresa Maat G Knowledge S.L. a través de los proyectos “Investigación y desarrollo de nuevos servicios para la plataforma MAAT-G” y Health-e-Child (IST 2004-027749).

Adicionalmente, también se ha recibido financiación para estancias y asistencia a congresos proveniente de proyectos del Ministerio (TIC2002-04586-C04-03, TIN2005-09098-C05-04, TIN2008-01825/TIN) y Bancaja (P1B2004-30) del grupo TKBG.

Index

List of Figures

List of Tables

List of Algorithms

CHAPTER 1

Introduction

Ontologies —formal specifications of a shared domain conceptualization [Gru93a, BAT97]— are playing a key role in the development of the Semantic Web [LHL01], and are already being used in domains as diverse as biomedicine [SAR$^+$07], agriculture [LS06], defence [LAF$^+$05], robotics [JYM$^+$08, Wil08, LJRCM09] and astronomy [LD$^+$07].

This chapter briefly describes current ontology representation formalisms and recapitulates our main research challenges and how they have been addressed.

Our research has mainly focused on investigating general methods and techniques to support ontology development in its different lifecycle phases. Biomedicine has been selected as the motivating application domain given the large amount of ontological resources currently available. Our techniques, however, can be applied to any other domain.

This chapter is organized as follows. Section 1.1 gives an overview of existing ontology representation paradigms. An introduction to the Ontology Web Language OWL is given in Section 1.2. In Section 1.3 some issues regarding the ontology-based representations in biomedicine are discussed. Section 1.4 presents an application scenario and introduces our main research challenges. Finally, Section 1.5 summarizes our main contributions.

1.1. Ontology modelling paradigms

Historically, the three main paradigms [NB03, SCM03, SJR09] for the modelling of ontologies have been the network-based paradigm (i.e., semantic networks and other graph-based formalisms), the frame-based paradigm, and the logic-based paradigm.

Semantic networks are a family of formalisms for graphically representing concepts an their relationships. Concepts are represented by nodes, whereas relationships are links connecting nodes. The representation paradigm based on semantic networks has been widely used in artificial intelligence to represent knowledge for expert systems. Originally, as stated by Woods [Woo75] and Brachman [Bra77], semantic networks lacked precise semantics to describe concepts and their relationships, which is crucial to enable automated reasoning in an unambiguous manner.

KL-ONE and *frames* systems were among the first efforts to formalize semantic networks. The language in KL-ONE [BS85] provided formal semantics to many of the basic constructs used in semantic networks. Concepts were represented and structured based on the idea of *structured inheritance networks* [Bra79] by using structure-forming operations such as specialization, restriction or differentiation. The KL-ONE language, in contrast to semantic networks, introduced a clear separation between intensional and extensional knowledge. Furthermore, KL-ONE introduced the notion of *automatic classification*, i.e., the computation of the implicit subsumption relations between concepts.

The concept of *frame* was originally introduced by Minsky [MW75] as an alternative to logic-oriented approaches in order to provide a more natural way to represent knowledge. The frames paradigm was interpreted by many as simply an alternative syntax to first-order logic [Hay79]. There is no doubt, however, that both frames and the KL-ONE language laid the foundations of modern knowledge representation systems.

1.1.1. Logic-based representations

The obvious way to provide an unambiguous semantics to semantic networks and frames is to map their constructs to an underlying logic, such as propositional or first order predicate logic (FOL).

As stated by Hayes [Hay79], the intended meaning of the basic constructs available in semantic networks and frame-based systems could be formalized using FOL. However, the validity problem as well as other central reasoning problems are undecidable in FOL — that is, there is no algorithm that can decide the validity of an arbitrary FOL formula. Levesque and Brachman [BL84, LB87] emphasized the need for knowledge representation languages that provide a *trade-off* between expressive power and nice computational properties, and they identified a fragment of FOL that could express the basic constructs available in semantic networks and frame systems. Originally this subset was called *terminological language* or *concept language*, and eventually evolved into a family of knowledge representation languages called *Description Logics* (DL) [BCM$^+$03].

KL-ONE can be seen as the predecessor of current DL-based systems. As in KL-ONE, DL systems make an explicit distinction between the terminological or intensional knowledge (a.k.a. Terminological Box or TBox), which refers to the general knowledge about the domain, and the assertional or extensional knowledge (a.k.a. Assertional Box or ABox), which represents facts about specific individuals. An overview of early DL systems can be found in [Doy91]. FaCT [TH06], HermiT [SMH08, MSH09], CEL [BLS06], Pellet [SPG$^+$07], SHER [DFK$^+$09] and RacerPro [HM01] are currently used DL systems.

Finally, it is also worth mentioning the Ontolingua system [Gru93b, FFR97], a frame-based system with logic-based formal semantics. Ontolingua was built on top of KIF (Knowledge Interchange Format) [RFB$^+$92] — a language based on FOL; however, ontologies could be defined in Ontolingua using exclusively frame-like constructs (thus restricting FOL's expressive power).

1.2. Formal ontology representation using the Ontology Web Language

The definition of a standard language for the formal definition of ontologies has been for many years a central research topic for the Semantic Web and Knowledge Representation communities [CGP00, GPC02].

The Ontology Web Language (OWL) is a World Wide Web Consortium (W3C) standard [PSHH04, MPSCG09], and it is being widely used in ontology modelling. The formal underpinning of OWL ontologies is based on formal logic [BCM$^+$03], with prominent dialects of OWL such as OWL DL and OWL Lite having a direct correspondence with description logics. There is currently an extensive range of logic-based algorithmic techniques and infrastructure available for OWL. OWL 2 [CHM$^+$08], a revision of the former OWL 1 [HPSvH03], is the language of choice in this research. OWL 2 presents, among others, the following main improvements with respect to OWL 1: (1) it provides new DL constructs which were not available in OWL 1 [MPSCG09], some of which provide additional expressive power to the language; (2) it clarifies the structure and specification of the language [MPSP09]; and (3) it provides more flexibility in the use of annotations [MPSP09] — extra-logical constructs that are widely used in ontology modelling.

We next recapitulate the description logic $\mathcal{SROIQ}$ (refer to [HKS06] for a more extensive description), the underlying logic of OWL 2, and introduce the main reasoning services typically implemented by DL systems.

1.2.1. $\mathcal{SROIQ}$ in a nutshell

A $\mathcal{SROIQ}$-signature $\mathbf{S}$ is the union of the disjoint sets $\mathbf{R}$ of *atomic roles* (denoted by $R, S, \ldots$), $\mathbf{C}$ of *atomic concepts* (denoted by $A, B, \ldots$) and $\mathbf{I}$ of *individuals* (denoted by $a, b, c, \ldots$).

A *$\mathcal{SROIQ}$-role* is either $R \in \mathbf{R}$ or an *inverse role* R^- with $R \in \mathbf{R}$. We denote by $\mathbf{Rol}$ the set of $\mathcal{SROIQ}$-roles for the signature $\mathbf{S}$. The function $\mathsf{Inv}(\cdot)$ is defined on $\mathbf{Rol}$ as follows: $\mathsf{Inv}(R) = R^-$ and $\mathsf{Inv}(R^-) = R$. A role characteristic axiom is one of the following forms: $\mathsf{Trans}(R)$ (Transitive), $\mathsf{Sym}(R)$ (Symetric), $\mathsf{Asy}(S)$ (Asymetric), $\mathsf{Ref}(R)$ (Reflexive), $\mathsf{Irr}(S)$ (Irreflexive), $\mathsf{Func}(S)$ (Functional), $\mathsf{InvFunc}(S)$ (Inverse-

Functional), and $\mathsf{Dis}(S_1, S_2,)$, where R is a role and S, S_1, S_2 are *simple* roles.[1] A role inclusion axiom (RIA) is of the form $w \sqsubseteq S$ where w is a finite string of roles (i.e. role chain) and R an atomic role. An RBox $\mathcal{R}$ is a finite set of RIAs and role characteristic axioms. To ensure decidability, $\mathcal{R}$ should satisfy certain *regularity conditions*, which we omit here for simplicity. The following grammar defines the $\mathcal{SROIQ}$*-concepts* for $\mathbf{S}$:

$$\mathbf{Con} := \top \mid \{a\} \mid A \mid \neg C \mid C_1 \sqcap C_2 \mid \exists R.C \mid \exists S.\mathsf{Self} \mid \geqslant n\, S.C \qquad (1.1)$$

where $a \in \mathbf{I}$, $A \in \mathbf{C}$, $C_{(i)} \in \mathbf{Con}$, $R, S \in \mathbf{Rol}$, with S a simple role, and n a positive integer. We use the following abbreviations: $C \sqcup D$ stands for $\neg(\neg C \sqcap \neg D)$; $\bot$ stands for $\neg\top$; $\forall R.C$ stands for $\neg(\exists R.\neg C)$; and $\leqslant n\, S.C$ stands for $\neg(\geqslant n{+}1\, S.\neg C)$.

A GCI is of the form $C_1 \sqsubseteq C_2$ with $C_i \in \mathbf{Con}$. A TBox $\mathcal{T}$ is a finite set of GCIs. Assertions have the following forms: $C(a)$, $R(a, b)$, $\neg R(a, b)$, $a \approx b$, or $a \not\approx b$, with $C \in \mathbf{Con}$, $a, b \in \mathbf{I}$, and $R \in \mathbf{Rol}$. An ABox $\mathcal{A}$ is a finite set of assertions. A $\mathcal{SROIQ}$ axiom α is either a GCI, an RIA, a role characteristic or an ABox assertion. A $\mathcal{SROIQ}$-ontology $\mathcal{O}$ is a tuple $\mathcal{O} = \langle \mathcal{T}, \mathcal{R}, \mathcal{A} \rangle$, where $\mathcal{T}$ is a TBox, $\mathcal{R}$ is an RBox and $\mathcal{A}$ is an ABox. Table 1.1 gives examples of a subset of $\mathcal{SROIQ}$ axioms and concept constructors.

Following the model-theoretic semantics of DL ontologies, an $\mathbf{S}$-*interpretation* $\mathcal{I}$ is a pair $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, with $\Delta^{\mathcal{I}}$ a non-empty set, called the *domain* of the interpretation, and $\cdot^{\mathcal{I}}$ a function that assigns to each $R \in \mathbf{R}$ a relation $R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$, to each $A \in \mathbf{C}$ a set $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$, and to each $a \in \mathbf{I}$ an object $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$. The function $\cdot^{\mathcal{I}}$ is extended to complex roles, strings of roles and concepts in the standard way (see [HKS06]).

The *satisfaction* relation $\mathcal{I} \models \alpha$ between $\mathcal{I}$ and an axiom α is also standard [HKS06]. An interpretation $\mathcal{I}$ is a *model* of $\mathcal{O}$ if $\mathcal{I}$ satisfies all axioms in $\mathcal{O}$. We say that $\mathcal{O}$ is *consistent* if a model of $\mathcal{O}$ exists. An ontology $\mathcal{O}$ *entails* α, written $\mathcal{O} \models \alpha$, if $\mathcal{I} \models \alpha$ for every model $\mathcal{I}$ of $\mathcal{O}$. An ontology $\mathcal{O}$ entails $\mathcal{O}'$, written $\mathcal{O} \models \mathcal{O}'$ if $\mathcal{O} \models \alpha$ for each $\alpha \in \mathcal{O}'$. The ontologies $\mathcal{O}, \mathcal{O}'$ are semantically equivalent, written $\mathcal{O} \equiv \mathcal{O}'$, if $\mathcal{O} \models \mathcal{O}'$ and $\mathcal{O}' \models \mathcal{O}$. Finally, we say that a concept C is satisfiable w.r.t. $\mathcal{O}$ if a model of $\mathcal{O}$ exists in which $C^I \neq \emptyset$.

From now on, we differentiate between *inconsistent ontologies* (i.e. without a model) and *incoherent ontologies* [SC03] (i.e. with unsatisfiable concepts).

Definition 1.1 (Coherent Ontology)**.** *An ontology $\mathcal{O}$ is coherent if it is consistent and does not contains unsatisfiable concepts.*

1.3. Ontology representations in biomedicine

The differences between *ontologies* and other terminological knowledge resources such as *lexicons*, *terminologies* and *thesauri* are often blurred in the literature

[1] Refer to [HKS06] for the definition of a simple role

Table 1.1: Some $\mathcal{SROIQ}$ axioms and concept expressions

OWL Axioms		
GCI	$C \sqsubseteq D$	$Patient \sqsubseteq Person$
RIA	$R \sqsubseteq S$	$complicates \sqsubseteq affects$
Class Assertion	$C(a)$	$Patient(john)$
Property Assertion	$R(a, b)$	$hasAge(jonh, 28)$
Different Individuals	$a \not\approx b$	$negative\_factor \not\approx positive\_factor$
Concept Expressions		
Atomic Concept	A	$Patient$
Complement	$\neg C$	$\neg SteroidalDrug$
Intersection	$C \sqcap D$	$Disease \sqcap \forall affects.WholeBody$
Union	$C \sqcup D$	$Steroid \sqcup NSAID$
Universal Restriction	$\forall R.C$	$\forall hasTreatment.NSAID$
Value Restriction	$\exists R\{a\}$	$\exists hasGender\{female\}$
Existential	$\exists R.C$	$\exists hasSymptom.Fever$
Min Cardinality	$\geqslant n\, R.C$	$\geqslant 3\, affects.Joint$
Nominal	$\{a, b, c\}$	$\{female, male\}$

[JYJRBRS09b]. Ontologies and terminological resources mainly differ in their expressive power. While terminological resources mostly describe linguistic properties (e.g., collecting synonyms) [Hir04, Bod06] and giving a basic organization of terms within a taxonomy (e.g., hypernymy); ontologies intend to provide formal descriptions of concepts, which could be rather complex. Thus, ontologies and terminological resources often serve to different purposes and therefore a distinction should be made.

In Figure 1.1 we have arranged the existing formalisms (depicted by boxes) according to their semantic expressiveness. Existing biomedical resources are placed to their closer formalism. Genuine lexical resources are placed closer to the left part of the diagram, like the Biolexicon [PJYLRS08] or the UMLS Specialist lexicon [BMS00]. More complex resources lie in between the definition of ontology and lexicon like MeSH [MNA$^+$99], ICD classification [icd07], the UMLS[2] Metathesaurus [Bod04] and the set of OBO ontologies [SAR$^+$07] (e.g., the Gene Ontology [ABB$^+$00]) that account for representations similar to semantic networks. Finally, at the end of the spectrum we find expressive ontologies such as FMA[3] [JR04], BioPAX[4] [LS07], GALEN[5] [RR06], NCI[6] [HdCD$^+$05] or SNOMED CT[7] [Spa00], which express stronger semantics.

[2] Unified Medical Language System
[3] Foundational Model of Anatomy
[4] Biological PAthways eXchange
[5] Generalized Architecture for Languages, Encyclopaedias and Nomenclatures in medicine
[6] National Cancer Institute
[7] Systematized NOmenclature of MEDicine Clinical Terms

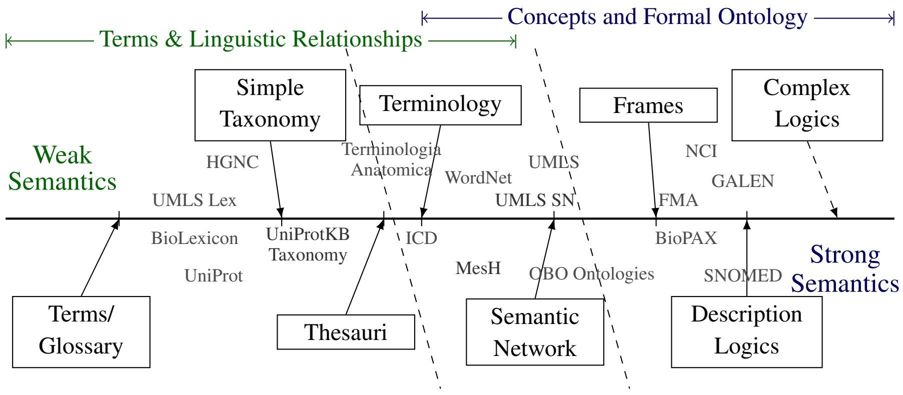


Figure 1.1: Adapted Ontology Spectrum based on [LM01, Bod06, JYJRBRS09b]

1.3.1. OBO and OWL ontologies

As already mentioned, OWL is the W3C standard to represent DL ontologies. Nevertheless, within the biomedical domain, OBO [SAR$^+$07, DR06] is the most widely used ontology language. OWL provides an expressive and formal language, however domain experts are not necessarily experts in knowledge representation using DLs. Furthermore, they have been so far interested in gathering together agreed knowledge instead of giving a more formal and functional representation for it [Lam06].

There have been several efforts [GHH$^+$07, MM07, TAM$^+$09] to provide formal DL semantics to OBO ontologies and to convert them into OWL. These approaches have focused on the definition of syntactical and semantic mappings between the OBO format and OWL. Additionally [TAM$^+$09] also identified the OBO language as a strict subset of OWL DL, which they call *OWL-Bio*.

The conversion of OBO ontologies into OWL allows for the interoperability between current biomedical resources and the Semantic Web infrastructure (editors, reasoners, and so on). However, although OBO ontologies represent an important community effort, they are somewhere in-between what is expected from an ontology and from a thesaurus [JYJRBRS09b] (see Figure 1.1) and they should be adapted in order to benefit from Semantic Web techniques.

Moreover, OBO ontologies are not exploiting all the expressivity of the OBO language, and in most of the cases they are limited to taxonomies, like the Human Disease Ontology [KS$^+$08]. Other OBO ontologies like the Gene Ontology [ABB$^+$00] also contain *part-whole* relationships (i.e., meronymy and holonymy) which also forms a taxonomy. Thus, one of the most important limitations of OBO ontologies is the lack of formal descriptions for complex concepts. Instead, they use *descriptive names* to label such concepts. These labels are closer to a text definition than to a concept name. For example, the Gene Ontology contains the *(biological_process)* concept *GO*:0002498 with label *"proteolysis within endoplasmic reticulum associated with antigen processing and presentation"*. Such complex concept should be described in a formal ontology by combining somehow smaller units of meaning, e.g., the concept GO:0002498 could be formally described as in axiom 1.2 where the semantics for each of its elements are also defined in such a formal ontology.

$$\begin{aligned}\mathsf{GO{:}0002498} \quad \equiv \quad & \mathsf{Proteolysis} \sqcap \exists \mathsf{has\_location.Endoplasmic\_Reticulum} \sqcap \\ & \exists \mathsf{associated\_with.(Antigen\_Processing} \sqcap \mathsf{Antigen\_Presentation)}\end{aligned} \tag{1.2}$$

1.3.2. Thesauri-ontology linkage

Thesauri and ontologies can be integrated in different ways. In the simplest approach, the terminology (synonyms, definitions, links to other resources, etc.) is incorporated directly into the ontology using annotation properties. The advantage of this approach is that all the necessary terminological information is included (at least syntactically); however, there are two main limitations: (1) the ontology should be kept up-to-date with changes in the set of synonyms and links to external resources; (2) the ontology could be overloaded with excessive metadata making it hard to manage.

OBO ontologies are an example of ontologies being integrated with terminological data. For example, the Human Disease Ontology contains 14040 classes, 7 properties, 15139 subsumption relationships and 445032 annotations (synonyms, references to entries of other thesauri such as UMLS, ICD, SNOMED and MESH). Therefore it associates an average of more than 30 annotations per class. The case of the Gene Ontology is similar, containing more than 190000 annotations for less than 32000 classes.

A loose coupling between the domain ontologies and thesauri is, however, desirable [Hir04]. The development of the FMA ontology is based on this idea and concept labels are reused from a thesaurus named *Terminologia Anatomica* [Ros01]. Figure 1.2 shows an example of this desirable setup [JYJRBRS09b] where the thesaurus is in charge of maintaining the set of synonyms (i.e., labels) for the concept, the definitions, links to external resources, and so on. Whereas the ontology concepts only have to keep a link to the thesaurus through a single annotation (see OWL code in Table 1.2). This setup not only provides a clearer organization of the knowledge (semantics and logics within the ontology; and lexicon and structure within the thesaurus) but also eases the alignment and integration of ontologies avoiding the application of lexical matching (e.g., concepts O_1:C and O_2:D from Figure 1.2 are linked to same thesaurus entry and thus they can be potentially aligned).

1.4. Application scenario and challenges

The examples and use cases selected for this dissertations are based on the *arthritis domain* of the Health-e-Child (HeC) project [FCI+06, JRBS+06] (specifically, on the creation of an ontology for the Juvenile Idiopathic Arthritis (**JIAO**)).

HeC aimed to develop an integrated health care platform for European paediatrics and decision support tools to access personalized health information. HeC project

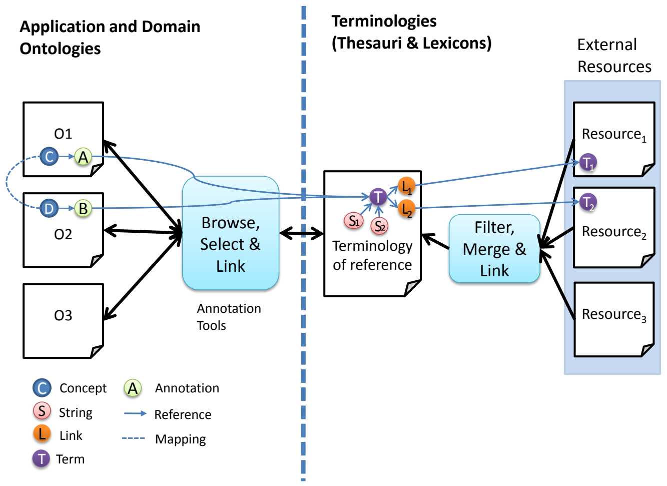


Figure 1.2: Ontology thesaurus link

```
<owl:Class rdf:about="ESR_Westergren">
  <rdfs:label>Sedimentation rate, Westergren</rdfs:label>
  <rdfs:subClassOf rdf:resource="ESR"/>
  <dc:identifier>http://krono.act.uji.es/thesaurus#HeCTh1000430</dc:identifier>
</owl:Class>
```

Table 1.2: Ontology-thesaurus link through an OWL annotation

focused on three paediatric diseases: (1) heart disorders, (2) inflammatory disorders (e.g. *Juvenile Idiopathic Arthritis (JIA)*) and (3) brain tumours. One of the objectives of HeC project was to create several ontologies for representing the relevant domain knowledge at different levels of granularity: molecular (e.g. genomic and proteomic data), cellular (e.g. results of blood tests), tissue (e.g. synovial fluid tests), organ (e.g. affected joints, heart description), body (e.g. examinations, treatments), and population (e.g. epidemiological studies). The purpose of this multilevel representation was to give a complete characterization of the different HeC diseases in order to provide a *rich ontological layer* to the HeC System. The goal of this semantic layer was to support a number of tasks such as *data integration* of heterogeneous sources, *linkage* to external knowledge, *enhancement of queries* over the patient data, and *decision support* for diagnosis, prognosis and follow-up [JRBS+06, TZH07, ABB+07].

Juvenile Idiopathic Arthritis (JIA) is a rare kind of arthritis and there is not yet a consensus about its classification nor even its name [D+05]. So far, three classification schemes have been proposed, namely: ACR (American College of Rheumatology), which uses the term *Juvenile Rheumatoid Arthritis (JRA)* as preferred name and proposes three disease subtypes, EULAR (European League Against Rheumatism), which opts for *Juvenile Chronic Arthritis (JCA)* and proposes six disease subtypes, and finally ILAR (International League of Associations for Rheumatology) which prefers JIA and proposes eight subtypes.

As already mentioned, the knowledge represented by the HeC ontologies involves different subdomains (e.g. genetics, drug classification, clinical tests) and granularities (from molecular to population). Thus, the development of **JIAO** is likely to be conducted *collaboratively* (changes over the ontology should be discussed and reconciled) and *concurrently* (i.e., several developers performing changes at the same time).

Each subtype of JIA is characterized by affecting different set and number of joints, the occurrence of some symptoms like fever or rash, the laboratory tests that are analyzed, the different treatments that are applied, etc. The development of **JIAO** from scratch would imply the conceptualization of the different joints of the body, the classification of the drugs for the treatments, the characterization of the different laboratory tests, etc. Nevertheless this knowledge is already well known by the community (unlike JIA). Thus, it may be assumed that available biomedical thesauri and ontologies can provide this knowledge in order to be *reused*.

Reused knowledge from biomedical ontologies must be *integrated*, however, available biomedical ontologies usually belong to independent projects and they do not use a common vocabulary. Moreover, the conceptualization process may have been guided by different points of view. Therefore, correspondences (i.e., alignments or mappings) between entities of the different ontologies have to be discovered, and the logic compatibility of the ontology have to be evaluated (diagnosis of alignments).

Reuse, *integration* and *concurrency&collaboration* arise as key challenges to develop an ontology such as **JIAO**. In this dissertation we have designed and implemented logic-based methods and techniques to support the development of ontologies such as **JIAO**. However, it is worth mentioning that the creation of such ontologies is outside the scope of this dissertation, and only some prototype ontologies were created for evaluation purposes.

1.5. Organization

The research conducted in this dissertation has been motivated and focused on the challenges presented in previous sections. We have designed methods and techniques to provide developers with logic-based support to address the three challenges: reuse, integration and concurrency. We have also implemented a suite of tools that implement our proposed techniques. A brief summary of the following chapters is presented next.

1.5.1. Chapter 2: Reuse in ontology development

Developers prefer to reuse knowledge that is commonly accepted by the community, either because they are not experts in all the subdomains to be modelled, or because they just want to save time and commit themselves to well-established knowledge. The very large size of thesaurus and ontologies as well their variety, however, makes it difficult to deploy them in particular applications mainly due to *scalability* (the management of such resources consume huge amount of computational resources) and *visualization* (editors can barely load and display these resources) issues. Moreover, applications usually does not require comprehensive descriptions of the domains but rather a handful *subset* of entities and axioms from them.

In Chapter 2, we present two methods to reuse knowledge from thesauri and ontologies, respectively. The first method focuses on the refinement of current thesauri in order to obtain a reference vocabulary for ontology development tasks. The second method is based on formal semantics to reuse ontologies keeping certain logic-based guarantees.

1.5.2. Chapter 3: Supporting concurrent ontology evolution

Large domain ontologies are being developed concurrently. The ontology developers can be geographically distributed and may contribute in different ways and to different extents. For example, the NCI ontology is being developed by 20 full time editors and one curator. Each editor works on a separate copy of the ontology and, at the end of a two week editing cycle, the curator uses a workflow management tool to review and approve the changes made by each editor [dCWF$^+$09]. Therefore, designing and maintaining such large ontologies is a highly complex process, which involves tracking and managing the frequent changes to the ontology, reconciling conflicting views of the domain from different developers, minimising the introduction of errors (e.g., ensuring that the ontology does not have unintended logical consequences), and so on.

Chapter 3 gives an overview of existing approaches and presents our novel techniques and tool support.

1.5.3. Chapter 4: Logic-based assessment of ontology integration

The vocabularies of independent developed ontologies are likely to diverge since they use different names or naming conventions to refer to their entities. There has been a growing interest in the development of techniques for identifying correspondences (i.e., mappings) between the ontologies to be integrated. However, the integration of the ontologies and the mappings usually leads to errors. These errors may be caused due to incorrect mappings or to inherent incompatibilities between the ontologies. Errors usually manifest themselves as unintended logical consequences (e.g., unsatisfiable concepts or unintended subsumptions), and they can be difficult to detect, understand and repair.

Current tools provide little or no support for the user in trying to assess the logical consequences when integrating ontologies using mappings. In Chapter 4, we present novel semi-automatic and automatic methods to provide such support: to detect and repair the unintended consequences of the integration.

1.5.4. Chapter 5: Conclusions and open lines

Chapter 5 recapitulates our main contributions and results, discusses the limitations of our proposed techniques, and suggests possible extensions or open research lines. Moreover, a list of the publications related to the dissertation is presented.

CHAPTER 2

Reuse in ontology development

The development of an ontology usually requires the reuse of knowledge from several sources and in different ways [Bor09, GPdCSF09]. Reusing non-logical symbols can help to provide a meaningful label to a concept. Importing a taxonomy from thesauri such as WordNet [Mil95, Fel98] or UMLS [Bod04] may provide the basis to organize concepts within the ontology. Borrowing logic descriptions from other ontologies may be also useful in order to take advantage of available rich descriptions of the domain. In summary, there are three main motivations to reuse knowledge:

- Developers, in general, save time[1] through reusing existing knowledge sources rather than creating their own from scratch.
- The reused knowledge is commonly accepted by the community.
- Developers are not always experts in all the subdomains modelled in an ontology.

Thus, external resources will play an important role in the development of ontologies (e.g., **JIAO**). In [GPdCSF09], several scenarios requiring knowledge reuse were proposed in the context of the NeOn methodology framework [DSFB$^+$09]. The contributions of this chapter fit within two of these scenarios: the *reuse and adaptation of non-ontological resources* (e.g., thesauri), which is described in Section 2.1, and the *reuse and import of ontologies*, which is described in Section 2.2.

[1] Reuse in ontology engineering, as in software engineering, will have benefits but it may also have costs due to re-engineering tasks, as analyzed in [BMT05]. The use of the proper technology (e.g., methods and tools) and resources (e.g., ontologies and thesauri) will be key factors.

2.1. Reusing knowledge from domain thesauri

In [JYJRBRS09b] we analyzed the limitations of current thesauri in order to be reused for ontology engineering tasks and we proposed a method to extract usable portions from them. UMLS Metathesaurus (UMLS-Meta) [Bod04] was used as the core provider of terms and structure, whereas SwissProt[2] [BBA+03] and DrugBank[3] [WKG+08] served as complementary thesauri.

UMLS-Meta represents the main effort for the creation of a multipurpose *reference thesaurus*. UMLS-Meta contains concepts from more than one hundred terminologies, classifications, and thesauri; e.g. FMA, MeSH, SNOMED CT or ICD. UMLS-Meta 2008AB includes almost two million terms and more than three million term names, hypernymy classification with more than one million relationships, and around forty millions of other kinds of relationships.

SwissProt is a manually curated biological database of protein sequences which aims at providing reliable protein sequences associated with descriptive annotations such as the protein function, domain structure, variants, etc. DrugBank database combines detailed drug (i.e. chemical, pharmacological and pharmaceutical) data with comprehensive drug target (i.e. sequence, structure, pathway) information. Note that, unlike UMLS-Meta, both SwissProt and Drugbank represents specialized lexicons.

Both UMLS-Meta, SwissProt and DrugBank represent very important efforts and they are considered a reference within the bioinformatics community; however, they should be refined and adapted in order to get a more useful *resource for ontology developers*. Section 2.1.1 discusses the main drawbacks of these resources, whereas Section 2.1.2 presents our method to reuse and adapt them in order to build a customized thesaurus to serve as the basis for **JIAO**.

2.1.1. Lexical and structural problems of current thesauri

A number of efforts [MA09, SAM09, JYJRL+08, JYJRBRS09b, HvMS+10] have focused on the normalization (e.g., rewriting synonyms, filtering redundancy, solving ambiguity) of terminological resources in order to make them more suitable for a particular application. Next, we present the main lexical and structural drawbacks we detected in UMLS-Meta, SwissProt and Drugbank [JYJRL+08, JYJRBRS09b]:

- *Complex Ambiguity Cases.* Some ambiguity cases are rather hard to solve. For example, the term *Prostate Cancer* has two associated UMLS-Meta entries: $C0600139_{UMLS}$ and $C0376358_{UMLS}$. Both concepts refer to the semantic type *Neoplastic Process*, and they have as preferred labels *Carcinoma of prostate* and *Malignant tumor of prostate*, respectively. These neoplastic processes have a close relationship, indeed the former is represented as a child of the later within the NCI and UMLS-Meta taxonomies.

[2] SwissProt: `http://www.expasy.ch/sprot/`
[3] DrugBank: `http://www.drugbank.ca/`

- *Descriptive names.* As in OBO ontologies (see Section 1.3.1), some UMLS-Meta synonyms are closer to a text definition than to a term name. For example, UMLS-Meta *Therapeutic or Preventive* term $C0580168_{UMLS}$: *"Amputation of finger through distal interphalangeal joint"*. Swissprot and DrugBank contain less cases but still some of the entries do not represent a desired concept label (e.g., Drug $APRD00506_{DRUGBANK}$, *"calcium carbonate with vitamin d, magnesium, zinc, copper and manganese"*). Let us emphasize that not all concepts can be described with a few words, indeed, such complex concepts should be described in formal ontologies by combining smaller units of meaning of the thesaurus, e.g. term $C0580168_{UMLS}$ can be formally described as in axiom $Amputation \sqcap \exists involve.Finger \sqcap \exists through.InterphalangealJoint$, where the semantics for each of its elements should be defined in an ontology.
- *Parametrization in the label.* The *Clinical Drug* $C1614077_{UMLS}$ has the preferred name *"Etanercept 50 mg/mL"*. This term indicates not only the drug name but also the dosage for this pharmaceutical product. The thesaurus should contain only the generic name, and then the ontology should provide a formal representation of $C1614077_{UMLS}$ as either a subclass of *"Etanercept"* (e.g., $Etanercept\_50 \sqsubseteq Etanercept \sqcap \exists hasDosage.``50mg/mL"$) or an instance.
- *Structural Problems.* The UMLS-Meta taxonomy contains undesired relationships and cycles since it integrates several taxonomies and vocabularies where terms are not always described and classified following the same criteria [4]. Within this evolution and integration process, new terms are matched to existing ones or a new entry is created, in both cases the resulting classification is hard to determine. For example, $Chronic\ Childhood\ Arthritis$ ($C0553662_{UMLS}$) has itself as a parent (i.e. broader term) and as a child (i.e. more specific term) according to $SNOMED$ and $ICD\text{-}10$ classifications. Contrary to UMLS-Meta, Swissprot and DrugBank, although they are quite comprehensive vocabularies, lack of a rich classification scheme, being their organization limited to a set of families or categories.

2.1.2. Thesauri reuse and adaptation method

JIATh, a light-weight thesaurus[5] [JYJRBRS09b, JYJRBRS09a] for the Arthritis domain of HeC project, was created to serve as a basis of the **JIAO** development. **JIATh** aims at containing a clear and not overloaded organization of terms, with lexical information (e.g., synonyms) and scope information (e.g., semantic group). We adopted SKOS [MMB$^+$05], an RDF-like language, to represent **JIATh**. SKOS has a rich support for labelling and reporting term metadata (e.g. Preferred label, Alternate labels, definitions, examples) as well as for defining term relationships (e.g. Has Broader, Has Narrower, Exact Match). Additionally information about the Scope and Origin Scheme can be added.

[4] In Chapter 4 a discussion about the logic-based compatibility of UMLS sources is given

[5] We regard a light-weight thesaurus as a thesaurus covering a domain partially

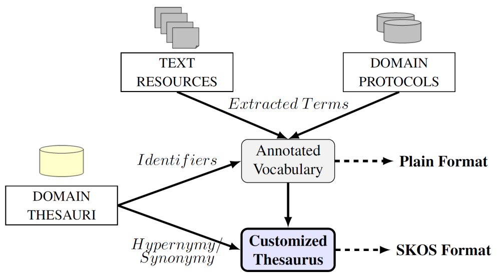


Figure 2.1: From requirements to a customized thesaurus

UMLS-Meta terms and relationships were reused and filtered to build our customized term organization. The applied method is depicted in Figure 2.1. Note that we have followed an informal procedure which could be enriched with formal patterns such as the presented in [GSGPdCSFVT08]. Our method is split into the following phases:

- *Vocabulary extraction.* Required terms for the domain were extracted by mining a set of medical protocols [BJRR+08, BJRNS10] and text resources from the literature [JYJRBRS07, JYJRL+08, LASPPJR08]. UMLS-Meta , Swissprot and DrugBank were used to annotate terms within protocols and text. Besides the automatic annotation techniques, manual intervention was also necessary. As a result a flat vocabulary [JYJRBRS09a] linked to the source domain thesauri was obtained.

- *Fragment extraction.* Given a set of terms satisfying the requirements (i.e. medical protocols), we applied the fragment extraction method described in [NL09]. This method encodes the whole UMLS-Meta with an interval index scheme that allows fast reconstructions of the hierarchies where the selected terms participate. Additionally, a selection of representative ancestors of the required terms is performed in order to organize these terms within the fragment.

- *Fragment Repair.* The obtained fragment can still contain undesired relationships. For example *Erythrocyte* has *Disease* as broader concept, among others. This is mainly due to wrong correspondences between the thesauri integrated within UMLS-Meta, which can imply cycles or undesired subsumptions. Fortunately, UMLS-Meta associates a *semantic type* [McC89] to each term, such that terms can be grouped within *semantic groups* [BM03a]. As in current UMLS-Meta auditing approaches [Cim98, CMP03, MBB09, MGHP09], these groups were used to establish compatibility criteria between terms and semantic types, that is, two terms can keep a broader-narrower relationship provided that both terms belong to the same semantic group (i.e. they have compatible semantic types). In the given example, *Erythrocyte* has *Cell* as a semantic type which belongs to the *Anatomy* semantic group, whereas *Disease or Syndrome*, the semantic type of *Disease*, belongs to the *Disorders* semantic group. Thus, *Erythrocyte* and *Disease* cannot have a broader-narrower relationship between them.

- *Term Grouping*. The terms that have not been classified under another term can be considered as *roots* since they do not have a broader term; however, in some cases they are not supposed to be root terms. This lack of classification may be due to failures in the UMLS-Meta term organization or an incomplete domain vocabulary selection. In order to alleviate these potential problems we have defined a set of preferred *top terms* and we have associated to each UMLS-Meta semantic type one of these top terms. Thus, if a term has no broader terms and it is not included within the set of top terms, then that term is organized under one of the top terms, according to its semantic type. For example, *Clinical_Finding* is defined as preferred root concept, and *Finding*, *Pathologic_Function*, *Cell_or_Molecular_Dysfunction* and *Temporal_Concept* semantic types are associated to this root.

- *Completion*. Swissprot and Drugbank specialised vocabularies were used to complete, with new entries and additional synonyms, the fragment extracted from UMLS-Meta. These vocabularies lack of a rich classification scheme therefore new terms (not included in UMLS-Meta) were associated a semantic type (e.g. Protein, Drug) and were organized within **JIATh** using the *term grouping method* commented above.

As a result **JIATh** thesaurus [JYJRBRS09b, JYJRBRS09a] contains 816 terms, organized in a nine-roots forest with 1135 broader relationships, and with a maximum depth of 11 levels. Moreover **JIATh** comprises 6097 term labels, at least one semantic scope label (i.e. semantic type) per term, and links to UMLS-Meta, SwissProt and DrugBank identifiers. Figure 2.2 shows and example of a SKOS-like **JIATh** entry.

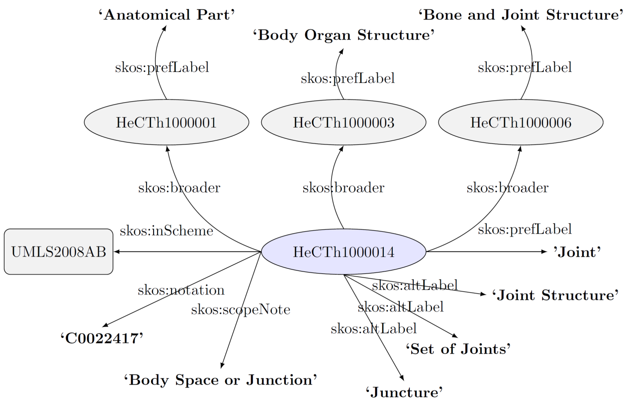


Figure 2.2: Example of a SKOS-like **JIATh** entry

2.2. Supporting safe and economic ontology reuse

Thesauri provide the lexical information about the domain and a basic structure of the knowledge; however, as already discussed in Sections 1.3.1 and 2.1.1, richer semantic descriptions requires a logic-based representation. Such logic-based representations can be defined from scratch or borrowed from domain ontologies. For the purposes of this section, NCI [HdCD$^+$05, GFH$^+$03] and GALEN [RR06] have been selected as *reference ontologies*. They contain information that is relevant to the domain of **JIAO**, such as detailed descriptions of the human joints as well as diseases and their symptoms. As an example, Figure 2.3 shows the global picture of the reusing scenario, where a set of concepts are reused from NCI and GALEN.

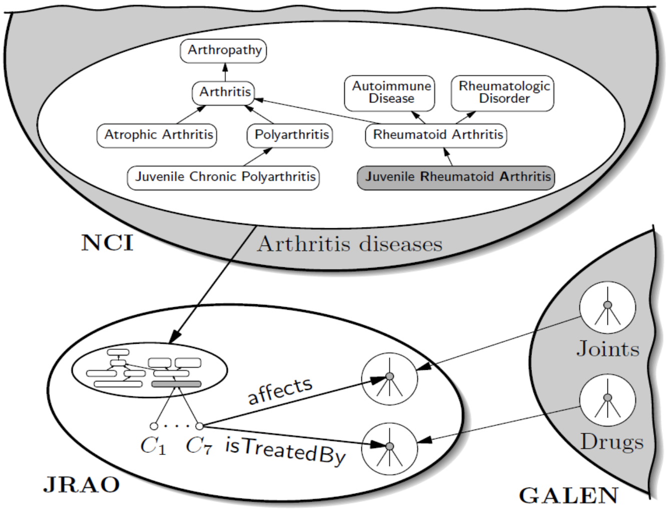


Figure 2.3: Constructing **JIAO** reusing fragments of GALEN and NCI

The developers of **JIAO** should be concerned about the following two issues when reusing knowledge from ontologies: (1) the consequences of importing/reusing knowledge, and (2) the necessary size of knowledge to import. In general, modellers do not want to *damage* the original meaning of the reused concepts from NCI and GALEN, and they are only interested in the (relatively small) parts of NCI and GALEN which are relevant to the **JIAO** domain.

In this section, we describe a novel method, based on a well-founded logic-based framework [GHKS08], and ProSÉ [6] [JRGS$^+$08b, JRGS$^+$08c, JRGS$^+$08a] a tool for modular ontology reuse. We support the user in the *safe* use of imported symbols and in the *economic* import of the relevant part of the reused ontology. Both features are

[6] ProSÉ: a Protégé plugin for Reusing Ontologies: Safe and Économique. This Protégé 4 plugin is freely available for download: `http://krono.act.uji.es/people/Ernesto/safety-ontology-reuse`

supported in a well-understood way: safety guarantees that the semantics of imported concepts is not changed, and economy guarantees that no difference can be observed between importing the whole ontology and importing only the relevant part.

Next sections are organized as follows. Section 2.2.1 introduces the main underlying formalisms used within our method. The ontology reuse method and logic-based guarantees followed by ProSÉ are presented in Sections 2.2.2 - 2.2.4. Finally, Section 2.2.5 discusses the related work regarding reuse and underlying techniques.

2.2.1. Underlying formalisms

The goal of this section is to give an brief overview of the used underlying formalisms: *conservative extension*, *safety*, *module* and *locality*. We refer the interested readers to [GLW06, LWW07, CHKS07, GHKS07, GHKS08] for a more comprehensive description of these formalisms.

2.2.1.1. Conservative extensions

As already mentioned, the developers of **JIAO** should not change the original meaning of the reused concepts. This requirement can be formalised using the notion of a *conservative extension* [GLW06, LWW07, GHKS08]. In the following, we use $\mathsf{Sig}()$ to denote the signature of an ontology or an axiom (i.e. $\mathsf{Sig}(\mathcal{O})$ and $\mathsf{Sig}(\alpha)$).

Definition 2.1 (Conservative Extension)**.** *Let $\mathcal{O}' \subseteq \mathcal{O}$ be ontologies, and $\mathbf{S}$ a signature. We say that $\mathcal{O}$ is an* $\mathbf{S}$-conservative extension *of $\mathcal{O}'$ if, for every axiom α with* $\mathsf{Sig}(\alpha) \subseteq \mathbf{S}$*, we have $\mathcal{O} \models \alpha$ iff $\mathcal{O}' \models \alpha$; $\mathcal{O}$ is a* conservative extension *of $\mathcal{O}'$ if $\mathcal{O}$ is an $\mathbf{S}$-conservative extension of $\mathcal{O}'$ for $\mathbf{S} = \mathsf{Sig}(\mathcal{O}')$.*

In our example, Definition 2.1 can be applied as follows: $\mathcal{O}' =$ NCI is the ontology to be reused, $\mathcal{O}$ is the union of **JIAO** and NCI, $\mathbf{S}$ represents the symbols reused from NCI, such as JRA and Rheumatologic_Disorder, and α stands for any axiom over the reused symbols only, e.g., $\mathsf{JRA} \sqsubseteq \mathsf{Rheumatologic\_Disorder}$.

2.2.1.2. Notion of safety

Definition 2.1 assumes that the ontology to be reused is static. In practice, however, ontologies such as NCI are under development and may evolve beyond the control of the **JIAO** developers. Thus, it is convenient to keep NCI separate from **JIAO** and make its axioms available on demand via a reference such that the developers of **JIAO** need not commit to a particular version of NCI. The notion of *safety* [GHKS08] can be seen as a stronger version of conservative extension that abstracts from the particular ontology to be reused and focuses only on the reused *symbols*.

Definition 2.2 (Safety for a Signature)**.** *Let $\mathcal{O}$ be an ontology and $\mathbf{S}$ a signature. We say that $\mathcal{O}$* is safe for $\mathbf{S}$ *if, for every ontology $\mathcal{O}'$ with* $\mathsf{Sig}(\mathcal{O}) \cap \mathsf{Sig}(\mathcal{O}') \subseteq \mathbf{S}$*, we have that $\mathcal{O} \cup \mathcal{O}'$ is a conservative extension of $\mathcal{O}'$. That is, $\mathcal{O} \cup \mathcal{O}'$ has the same consequences over $\mathbf{S}$ as $\mathcal{O}'$ alone, therefore $\mathcal{O} \cup \mathcal{O}' \models \alpha$ iff $\mathcal{O}' \models \alpha$ for every axiom with* $\mathsf{Sig}(\alpha) \subseteq \mathbf{S}$

Definition 2.2 allows us to consider a set of symbols $\mathbf{S}$ of our ontology $\mathcal{O}$ as *external*, with independence of the ontology $\mathcal{O}'$ to be finally imported.

From now on, we distinguish between *external* and *local* symbols in our local ontologies [GHKS07]. Intuitively, the external symbols of an ontology are those that are assumed to be defined externally in other ontologies (in the local ontology are only named), whereas local symbols are defined within the ontology itself by possibly reusing the external symbols in their definitions.

2.2.1.3. Notion of module

As already mentioned, **JIAO** should import only the relevant parts of NCI and GALEN, and without losing important information. This idea can be formalised using the notion of *module* [GHKS08].

Definition 2.3 (Module for a Signature)**.** *Let $\mathcal{O}'_{\mathbf{S}} \subseteq \mathcal{O}'$ be ontologies and $\mathbf{S}$ a signature. We say that $\mathcal{O}'_{\mathbf{S}}$* is a module for $\mathbf{S}$ in $\mathcal{O}'$ *(or an* $\mathbf{S}$-module in $\mathcal{O}'$*) if, for every importing ontology $\mathcal{O}$ with* $\mathsf{Sig}(\mathcal{O}) \cap \mathsf{Sig}(\mathcal{O}') \subseteq \mathbf{S}$*, we have that $\mathcal{O} \cup \mathcal{O}'$ is a conservative extension of $\mathcal{O} \cup \mathcal{O}'_{\mathbf{S}}$ for* $\mathbf{S} \subseteq \mathsf{Sig}(\mathcal{O})$.

Definition 2.3 states that importing a module for $\mathbf{S}$ in $\mathcal{O}'$ should give exactly the same answers as if the whole $\mathcal{O}'$ had been imported.

Safety and module can also be related as follows:

Proposition 2.1 ([GHKS08], Safety and Modules)**.** *If $\mathcal{O}' \setminus \mathcal{O}'_{\mathbf{S}}$ is safe for* $\mathbf{S} \cup \mathsf{Sig}(\mathcal{O}'_{\mathbf{S}})$*, then $\mathcal{O}'_{\mathbf{S}}$ is an $\mathbf{S}$-module in $\mathcal{O}'$.*

2.2.1.4. Transitivity property

Propositions 2.2 and 2.3 presents the transitivity property for conservative extensions and modules which will be used later on in this Section.

Proposition 2.2 (Transitivity of Conservative Extensions)**.** *Let $\mathcal{O}'$ be an $\mathbf{S}$-conservative extension of $\mathcal{O}'_{\mathbf{S}}$, and $\mathcal{O}'_{\mathbf{S}}$ an $\mathbf{S}$-conservative extension of $\mathcal{O}''_{\mathbf{S}}$, then $\mathcal{O}'$ is an $\mathbf{S}$-conservative extension of $\mathcal{O}''_{\mathbf{S}}$.*

Proof. Let $\mathcal{O}'$ be an $\mathbf{S}$-conservative extension of $\mathcal{O}'_{\mathbf{S}}$; therefore for every axiom α with $\mathsf{Sig}(\alpha) \subseteq \mathbf{S}$, we have $\mathcal{O} \models \alpha$ iff $\mathcal{O}'_{\mathbf{S}} \models \alpha$. $\mathcal{O}'_{\mathbf{S}}$ an $\mathbf{S}$-conservative extension of $\mathcal{O}''_{\mathbf{S}}$; therefore, $\mathcal{O}'_{\mathbf{S}} \models \alpha$ iff $\mathcal{O}''_{\mathbf{S}} \models \alpha$. Thus, $\mathcal{O} \models \alpha$ iff $\mathcal{O}''_{\mathbf{S}} \models \alpha$ and $\mathcal{O}'$ is an $\mathbf{S}$-conservative extension of $\mathcal{O}''_{\mathbf{S}}$.

□

Proposition 2.3 (Transitivity of Modules)**.** *Let $\mathcal{O}'_{\mathbf{S}}$ be an $\mathbf{S}$-module in $\mathcal{O}'$, and $\mathcal{O}''_{\mathbf{S}}$ a $\mathbf{S}$-module in $\mathcal{O}'_{\mathbf{S}}$, then $\mathcal{O}''_{\mathbf{S}}$ is a $\mathbf{S}$-module in $\mathcal{O}'$.*

Proof. Since $\mathcal{O}'_{\mathbf{S}}$ is a $\mathbf{S}$-module in $\mathcal{O}'$ we have that, by Definition 2.3, for every ontology $\mathcal{O}$ with $\mathsf{Sig}(\mathcal{O}) \cap \mathsf{Sig}(\mathcal{O}') \subseteq \mathbf{S}$ it holds that $\mathcal{O} \cup \mathcal{O}'$ is a $\mathbf{S}$-conservative extension of $\mathcal{O} \cup \mathcal{O}'_{\mathbf{S}}$. Since $\mathcal{O}''_{\mathbf{S}}$ a $\mathbf{S}$-module in $\mathcal{O}'_{\mathbf{S}}$, then for every ontology $\mathcal{O}$ with

$\mathsf{Sig}(\mathcal{O}) \cap \mathsf{Sig}(\mathcal{O}'_{\mathbf{S}}) \subseteq \mathbf{S}$ it holds that $\mathcal{O} \cup \mathcal{O}'_{\mathbf{S}}$ is a $\mathbf{S}$-conservative extension of $\mathcal{O} \cup \mathcal{O}''_{\mathbf{S}}$. Since $\mathcal{O} \cup \mathcal{O}'$ is a $\mathbf{S}$-conservative extension of $\mathcal{O} \cup \mathcal{O}'_{\mathbf{S}}$ and $\mathcal{O} \cup \mathcal{O}'_{\mathbf{S}}$ is a $\mathbf{S}$-conservative extension of $\mathcal{O} \cup \mathcal{O}''_{\mathbf{S}}$, by Proposition 2.2, $\mathcal{O} \cup \mathcal{O}'$ is a $\mathbf{S}$-conservative extension of $\mathcal{O} \cup \mathcal{O}''_{\mathbf{S}}$ and therefore $\mathcal{O}''_{\mathbf{S}}$ is a $\mathbf{S}$-module in $\mathcal{O}'$. □

2.2.1.5. Locality condition

Checking whether $\mathcal{O}$ is an $\mathbf{S}$-conservative extension of $\mathcal{O}_{\mathbf{S}}$, whether $\mathcal{O}$ is safe for $\mathbf{S}$, or whether $\mathcal{O}_{\mathbf{S}}$ is an $\mathbf{S}$-module in $\mathcal{O}$ is, unfortunately, undecidable for expressive DLs such as $\mathcal{SHOIQ}$ or $\mathcal{SROIQ}$.

Therefore, the undecidability problem is addressed in [GHKS08] by means of a *locality condition*, a sufficient condition for safety. Thus if an ontology (i.e. all its axioms) satisfy the locality condition for a given $\mathbf{S}$, then we can guarantee that it is safe (i.e. local) for $\mathbf{S}$; however the reciprocal does not necessarily hold [GHKS08].

A set of *classes of interpretations* were characterized in [GHKS08] depending on the interpretation ($\perp$ or $\top$) given to the entities outside the selected signature $\mathbf{S}$. Two classes of interpretations are used in ProSÉ :

- $\perp$-Interpretation ($\mathcal{I}_{\perp}$): This interpretation is used when we are interested in *refine and/or reference* the symbols from the signature $\mathbf{S}$. Entities not belonging to $\mathbf{S}$ are interpreted as $\perp$.
- $\top$-Interpretation ($\mathcal{I}_{\top}$): This interpretation is used when we are interested in *generalize and/or reference* the symbols from the signature $\mathbf{S}$. Entities not belonging to $\mathbf{S}$ are interpreted as $\top$.

An ontology $\mathcal{O}$ will be safe for a Signature $\mathbf{S}$ if all its axioms α are satisfied by one of the proposed interpretations. An axiom α is $\perp$-local ($\top$-local) w.r.t. $\mathbf{S}$ if $\mathcal{I}_{\perp} \models \alpha$ ($\mathcal{I}_{\perp} \models \alpha$).

The locality condition is widely applicable in practice and it can be checked syntactically. [GHKS08] proposed a set rules to characterize $\perp$-local or $\top$-local axioms (see Figure 2.4) depending on the required interpretation ($\mathcal{I}_{\top}$ or $\mathcal{I}_{\perp}$)

Definition 2.4 (Syntactic $\perp$-Locality and $\top$-Locality)**.** *Let* $\mathbf{S}$ *be a signature. An axiom* α *is* $\perp$-local w.r.t. $\mathbf{S}$ *($\top$-local w.r.t* $\mathbf{S}$*) if* $\alpha \in \mathsf{Ax}(S)$*, as defined in Figure 2.4. An ontology* $\mathcal{O}$ *is $\perp$-local ($\top$-local) w.r.t.* $\mathbf{S}$ *if* α *is $\perp$-local ($\top$-local) w.r.t.* $\mathbf{S}$ *for all* $\alpha \in \mathcal{O}$.

Both $\top$-locality and $\perp$-locality are sufficient for safety:

Proposition 2.4 ([GHKS08], Locality Implies Safety)**.** *If an ontology* $\mathcal{O}$ *is $\perp$-local or $\top$-local w.r.t.* $\mathbf{S}$*, then* $\mathcal{O}$ *is safe for* $\mathbf{S}$.

Modules can be defined in terms of locality by using Propositions 2.1 and 2.4.

Definition 2.5 (Locality Modules)**.** *Let* $\mathcal{O}_{\mathbf{S}} \subseteq \mathcal{O}$ *be ontologies, and* $\mathbf{S}$ *a signature. We say that* $\mathcal{O}_{\mathbf{S}}$ *is a* $\perp$-module (respectively $\top$-module) for $\mathbf{S}$ in $\mathcal{O}$ *if* $\mathcal{O} \setminus \mathcal{O}_{\mathbf{S}}$ *is $\perp$-local (respectively $\top$-local) w.r.t.* $\mathbf{S} \cup \mathsf{Sig}(\mathcal{O}_{\mathbf{S}})$.

$\mathsf{Ax}(\mathbf{S})$	$::=$	$C^\perp \sqsubseteq C \mid C \sqsubseteq C^\top \mid R^\perp \sqsubseteq R \mid \mathsf{Trans}(R^\perp) \mid \mathsf{Funct}(R^\perp) \mid a : C^\top$
$\mathbf{Con}^\perp(\mathbf{S})$	$::=$	$A^\perp \mid \neg C^\top \mid C \sqcap C^\perp \mid C^\perp \sqcap C \mid C_1^\perp \sqcup C_2^\perp \mid \exists R.C^\perp \mid \geqslant n\, R.C^\perp \mid \exists R^\perp.C \mid \geqslant n\, R^\perp.C$
$\mathbf{Con}^\top(\mathbf{S})$	$::=$	$\neg C^\perp \mid C_1^\top \sqcap C_2^\top \mid C \sqcup C^\top \mid C^\top \sqcup C \mid \forall R.C^\top \mid <n\, RC^\top \mid \forall R^\perp.C \mid <n\, R^\perp C$

(b) $\top$*-Locality*

$\mathsf{Ax}(\mathbf{S})$	$::=$	$C^\perp \sqsubseteq C \mid C \sqsubseteq C^\top \mid R \sqsubseteq R^\top \mid \mathsf{Trans}(R^\top) \mid a : C^\top \mid r^\top(a,b)$
$\mathbf{Con}^\perp(\mathbf{S})$	$::=$	$\neg C^\top \mid C \sqcap C^\perp \mid C^\perp \sqcap C \mid C_1^\perp \sqcup C_2^\perp \mid \exists R.C^\perp \mid \geqslant n\, R.C^\perp \mid \forall R^\top.C^\perp \mid <n\, R^\top C^\perp$
$\mathbf{Con}^\top(\mathbf{S})$	$::=$	$A^\top \mid \neg C^\perp \mid C_1^\top \sqcap C_2^\top \mid C \sqcup C^\top \mid C^\top \sqcup C \mid \exists R^\top.C^\top \mid \geqslant n\, R^\top.C^\top \mid \forall R.C^\top \mid$
		$<n\, RC^\top \mid \forall R^\perp.C \mid <n\, R^\perp C$

Figure 2.4: Syntactic locality conditions. Where, $C^\perp \in \mathbf{Con}^\perp(\mathbf{S})$, $C_i^\top \in \mathbf{Con}^\top(\mathbf{S})$, $A^\perp, A^\top, r^\top, r^\perp \notin \mathbf{S}$, $\mathsf{Sig}(R^\perp), \mathsf{Sig}(R^\top) \nsubseteq \mathbf{S}$, and C is any concept and R is any role.

Locality modules $\perp$-modules and $\top$-modules are modules [GHKS08]:

Proposition 2.5 ([GHKS08], Locality-based Modules are Modules)**.** *Let $\mathcal{O}_\mathbf{S}$ be either a $\perp$-module or a $\top$-module for* $\mathbf{S}$ *in $\mathcal{O}$ and let* $\mathbf{S}' = \mathbf{S} \cup \mathsf{Sig}(\mathcal{O}_\mathbf{S})$*. Then $\mathcal{O}_\mathbf{S}$ is a* $\mathbf{S}'$*-module in $\mathcal{O}$.*

In [JRGS$^+$08b] we combined the notions of $\top$-module and $\perp$-module in order to extract a module as small as possible. From Propositions 2.3 and 2.5 we can affirm that a $\top$-module of a $\perp$-module is also a module.

Proposition 2.6 (A $\top$-module of a $\perp$-module is a Module)**.** *Let $\mathcal{O}_{\mathbf{S}_\perp}$ be a $\perp$-module for* $\mathbf{S}$ *in $\mathcal{O}$ and let $\mathcal{O}_{\mathbf{S}_\top}$ a $\top$-module for* $\mathbf{S}$ *in $\mathcal{O}_{\mathbf{S}_\perp}$. Then $\mathcal{O}_{\mathbf{S}_\top}$ is also a* $\mathbf{S}$*-module in $\mathcal{O}$.*

Proof. Let $\mathbf{S}_\perp = \mathsf{Sig}(\mathcal{O}_{\mathbf{S}_\perp}) \cup \mathbf{S}$ and $\mathbf{S}_\top = \mathsf{Sig}(\mathcal{O}_{\mathbf{S}_\top}) \cup \mathbf{S}$. By Proposition 2.5, $\mathcal{O}_{\mathbf{S}_\perp}$ is a $\mathbf{S}_\perp$-module in $\mathcal{O}$ and $\mathcal{O}_{\mathbf{S}_\top}$ is a $\mathbf{S}_\top$-module in $\mathcal{O}_{\mathbf{S}_\perp}$. By Proposition 2.3, $\mathcal{O}_{\mathbf{S}_\top}$ is a $\mathbf{S}$-module in $\mathcal{O}$. $\square$

2.2.1.6. Locality in a nutshell

Table 2.1 summarizes the grammar in Figure 2.4 with the $\perp$-local (respectively $\top$-local) requirements for some simple axioms. Thus this table should be solely used as a guide. The gramar in Figure 2.4, which contains all syntactic locality conditions, should be referred when analysing more compex axioms.

Reusing entities in **JIAO**

Consider the example where **JIAO** defines the axioms 2.1-2.3. Underlined concepts are reused from NCI or GALEN (see scenario in Figure 2.3) and represents the

Axiom type	Locality type	Requirements to be local
(1) $A \sqsubseteq B$	$\bot$-Locality	$A = \bot$ or $B = \top$ or $A \notin \mathbf{S}$
	$\top$-Locality	$A = \bot$ or $B = \top$ or $B \notin \mathbf{S}$
(2) $A \equiv B$	$\bot$-Locality	$(A = \bot$ and $B = \bot)$ or $(A \notin \mathbf{S}$ and $B \notin \mathbf{S})$ or $(A = \top$ and $B = \top)$
	$\top$-Locality	
(3) $A \sqsubseteq \exists R.B$	$\bot$-Locality	$A = \bot$ or $A \notin \mathbf{S}$
	$\top$Locality	$A = \bot$ or $((B = \top$ or $B \notin \mathbf{S})$ and $R \notin \mathbf{S})$
(4) $A \sqsubseteq \exists R.\Phi$	$\bot$-Locality	$A = \bot$ or $A \notin \mathbf{S}$
	$\top$Locality	$A = \bot$ or $R \notin \mathbf{S}$
(5) $A \sqsubseteq \forall R.B$	$\bot$-Locality	$A = \bot$ or $A \notin \mathbf{S}$ or $B = \top$ or $R \notin \mathbf{S}$
	$\top$Locality	$A = \bot$ or $B = \top$ or $B \notin \mathbf{S}$
(6) $A \sqsubseteq \forall R.\Phi$	$\bot$-Locality	$A = \bot$ or $A \notin \mathbf{S}$ or $R \notin \mathbf{S}$
	$\top$Locality	Always since Φ is interpreted as $\top$
(7) $A \sqsubseteq \geqslant n\, R.B$	Same operation as $A \sqsubseteq \exists R.B$	
(8) $A \sqsubseteq < n\, RB$	Same operation as $A \sqsubseteq \forall R.B$	
(9) $A \sqsubseteq \ni R.\mathit{indiv}$	$\bot$-Locality	$A = \bot$ or $A \notin \mathbf{S}$
	$\top$-Locality	$A = \bot$ or $(R \notin \mathbf{S}$ and $\mathit{indiv} \notin \mathbf{S})$
(10) $A \sqsubseteq B_1 \sqcap B_2$	$\bot$-Locality	$A = \bot$ or $A \notin \mathbf{S}$ or $\forall i, B_i = \top$
	$\top$-Locality	$A = \bot$ or $(\forall i, B_i \notin \mathbf{S}$ or $B_i = \top)$
(11) $A \sqsubseteq B_1 \sqcup B_2$	$\bot$-Locality	$A = \bot$ or $A \notin \mathbf{S}$ or $\exists i, B_i = \top$
	$\top$-Locality	$A = \bot$ or $(\exists i, B_i \notin \mathbf{S}$ or $B_i = \top)$
(12) $A_1 \sqcap A_2 \sqsubseteq \bot$	$\bot$-Locality	$(\exists i, A_i \notin \mathbf{S}$ or $A_i = \top)$
	$\top$-Locality	$(\exists i, A_i = \top)$
(13) $\exists R.A \sqsubseteq B$	$\bot$-Locality	$A = \bot$ or $A \notin \mathbf{S}$ or $R \notin \mathbf{S}$ or $B = \top$
	$\top$-Locality	$A = \bot$ or $B = \top$ or $B \notin \mathbf{S}$
(14) $\exists R.\Phi \sqsubseteq B$	$\bot$-Locality	Always
	$\top$-Locality	$B = \top$ or $B \notin \mathbf{S}$
(15) $\forall R.A \sqsubseteq B$	$\bot$-Locality	$B = \top$
	$\top$-Locality	$B = \top$ or $B \notin \mathbf{S}$ or $(A = \bot$ and $R \notin \mathbf{S})$
(16) $\forall R.\Phi \sqsubseteq B$	$\bot$-Locality	$B = \top$
	$\top$-Locality	$B = \top$ or $B \notin \mathbf{S}$
(17) $\geqslant n\, R.A \sqsubseteq B$	Same operation as $\exists R.A \sqsubseteq B$	
(18) $< n\, RA \sqsubseteq B$	Same operation as $\forall R.A \sqsubseteq B$	

Table 2.1: Basic axioms and requirements to be local. Where A, B, B_1 and B_2 represents atomic concepts, R an atomic role, and Φ a *datatype*. Note that $A = \bot$ (respectively $B = \top$) involves a syntactic test, thus A (respectively B) is the *bottom* (respectively *top*) concept itself.

external symbols $\mathbf{S} = \mathbf{S}_{NCI} \uplus \mathbf{S}_{GALEN}$, with $\mathbf{S}_{GALEN} = \{Joint\}$ and $\mathbf{S}_{NCI} = \{JRA, Juvenile\_Chronic\_Polyarthritis\}$.

$$\begin{aligned} \mathsf{Polyarticular\_JRA} &\sqsubseteq \underline{\mathsf{JRA}} \quad (\bot\text{-local}) & (2.1)\\ \underline{\mathsf{Juvenile\_Chronic\_Polyarthritis}} &\sqsubseteq \mathsf{Polyarticular\_JRA} \quad (\top\text{-local}) & (2.2)\\ \mathsf{Polyarticular\_JRA} &\sqsubseteq \geqslant 5\, \mathsf{affects}.\underline{\mathsf{Joint}} \quad (\bot\text{-local}) & (2.3) \end{aligned}$$

Axiom 2.2 is $\top$-local w.r.t. $\mathbf{S}_{NCI}$ since defines a more general concept (see row 1 in Table 2.1), whereas axioms 2.1 and 2.3 are $\bot$-local w.r.t. $\mathbf{S}_{NCI}$ and $\mathbf{S}_{GALEN}$ respectively (see rows 1 and 7 from Table 2.1), since they define more specialized concepts from reused symbols. Note that, the simultaneous refinement and generalisation for a given Sg and ontology (e.g. symbols from NCI $\mathbf{S}_{NCI}$) may compromise safety. For example, **JIAO** should not simultaneously contain axioms 2.1 and 2.2, since they imply $\underline{\mathsf{Juvenile\_Chronic\_Polyarthritis}} \sqsubseteq \underline{\mathsf{JRA}}$, and therefore **JIAO** is not safe w.r.t. $\mathbf{S}_{NCI}$.

Extracting local modules

Local modules are extracted iteratively by checking each axiom with respect to the current required signature. If an axiom is evaluated as non local then it is added to the module and the required signature is extended with the signature of the axiom. The iterative process finishes if after checking all axioms the signature has the same size as in the beginning of the iteration.

Table 2.2 shows an example ontology within the **JIAO** domain. Tables 2.3 and 2.4 show how a $\bot$-module and a $\top$-module are extracted, respectively. The initial signature $\mathbf{S}_0 = \{Poly\_JIA, suffered\_by, Systemic\_Disease\}$ (underlined concepts in Table 2.2) and the initial empty module $\mathcal{M}_0 = \{\}$ are extended stepwise. For simplification reasons, we only show completely the first iteration for each example, for example, in Table 2.4 two iterations (separated with a dashed line) are needed to include all axioms. Note that rows from Table 2.1 are referenced as "(i)" where *i* is the row identifier.

$$\begin{array}{|lrcl|} \mathcal{O} = \{\ (\alpha_1) & \underline{Poly\_JIA} & \sqsubseteq & JIA, \\ (\alpha_2) & \underline{Poly\_JIA} & \sqsubseteq & \forall hasRF.Positive\_RF, \\ (\alpha_3) & \underline{Poly\_JIA} & \sqsubseteq & \geqslant 5\, affects.Joint, \\ (\alpha_4) & Systemic\_JIA & \sqsubseteq & JIA, \\ (\alpha_5) & Systemic\_JIA & \sqsubseteq & \underline{Systemic\_Disease}, \\ (\alpha_6) & JIA & \sqsubseteq & \forall \underline{suffered\_by}.\mathsf{Child}, \\ (\alpha_7) & \exists affects.WholeBody & \sqsubseteq & \underline{Systemic\_Disease}\ \} \end{array}$$

Table 2.2: An example ontology with external entities

	Consideration	Consequence
(α_1)	$Poly\_JIA \in \mathbf{S}_0 \ \Rightarrow\ \alpha_1$ is not $\bot$-local (1)	$\mathbf{S}_1 = \mathbf{S}_0 \cup \{JIA\}$ $\mathcal{M}_1 = \mathcal{M}_0 \cup \{\alpha_1\}$
(α_2)	$hasRF \notin \mathbf{S}_1 \ \Rightarrow\ \alpha_2$ is $\bot$-local (5)	$\mathbf{S}_2 = \mathbf{S}_1$ $\mathcal{M}_2 = \mathcal{M}_1$
(α_3)	$Poly\_JIA \in \mathbf{S}_2 \ \Rightarrow\ \alpha_3$ is not $\bot$-local (7)	$\mathbf{S}_3 = \mathbf{S}_2 \cup \{affects, Joint\}$ $\mathcal{M}_3 = \mathcal{M}_2 \cup \{\alpha_3\}$
(α_4)	$Systemic\_JIA \notin \mathbf{S}_3 \ \Rightarrow\ \alpha_4$ is $\bot$-local (1)	$\mathbf{S}_4 = \mathbf{S}_3$ $\mathcal{M}_4 = \mathcal{M}_3$
(α_5)	analogous to α_4	$\mathbf{S}_5 = \mathbf{S}_4$ $\mathcal{M}_5 = \mathcal{M}_4$
(α_6)	$JIA, suffered\_by \in \mathbf{S}_5 \ \Rightarrow\ \alpha_6$ is not $\bot$-local (5)	$\mathbf{S}_6 = \mathbf{S}_5 \cup \{Child\}$ $\mathcal{M}_6 = \mathcal{M}_5 \cup \{\alpha_6\}$
(α_7)	$WholeBody \notin \mathbf{S}_6 \ \Rightarrow\ \alpha_7$ is $\bot$-local (13)	$\mathbf{S}_7 = \mathbf{S}_6$ $\mathcal{M}_7 = \mathcal{M}_6$

Table 2.3: An example illustrating the construction of a $\bot$-module

	Consideration	Consequence
(α_1)	$JIA \notin \mathbf{S}_0 \ \Rightarrow\ \alpha_1$ is $\top$-local (1)	$\mathbf{S}_1 = \mathbf{S}_0$ $\mathcal{M}_1 = \mathcal{M}_0$
(α_2)	$Positive\_RF \notin \mathbf{S}_1 \ \Rightarrow\ \alpha_2$ is $\top$-local (5)	$\mathbf{S}_2 = \mathbf{S}_1$ $\mathcal{M}_2 = \mathcal{M}_1$
(α_3)	$affects, Joint \notin \mathbf{S}_2 \ \Rightarrow\ \alpha_3$ is $\top$-local (7)	$\mathbf{S}_3 = \mathbf{S}_2$ $\mathcal{M}_3 = \mathcal{M}_2$
(α_4)	$JIA \notin \mathbf{S}_3 \ \Rightarrow\ \alpha_4$ is $\top$-local (1)	$\mathbf{S}_4 = \mathbf{S}_3$ $\mathcal{M}_4 = \mathcal{M}_3$
(α_5)	$Systemic\_Disease \in \mathbf{S}_4 \ \Rightarrow\ \alpha_5$ is not $\top$-local (1)	$\mathbf{S}_5 = \mathbf{S}_4 \cup \{Systemic\_JIA\}$ $\mathcal{M}_5 = \mathcal{M}_4 \cup \{\alpha_5\}$
(α_6)	$Child \notin \mathbf{S}_5 \ \Rightarrow\ \alpha_6$ is $\top$-local (5)	$\mathbf{S}_6 = \mathbf{S}_5$ $\mathcal{M}_6 = \mathcal{M}_5$
(α_7)	$Systemic\_Disease \in \mathbf{S}_6 \ \Rightarrow\ \alpha_7$ is not $\top$-local (13)	$\mathbf{S}_7 = \mathbf{S}_6 \cup$ $\{affects, WholeBody\}$ $\mathcal{M}_7 = \mathcal{M}_6 \cup \{\alpha_7\}$
(α_3)	$affects \in \mathbf{S}_7 \ \Rightarrow\ \alpha_3$ is not $\top$-local (7)	$\mathbf{S}_8 = \mathbf{S}_7 \cup \{Joint\}$ $\mathcal{M}_8 = \mathcal{M}_7 \cup \{\alpha_3\}$

Table 2.4: An example illustrating the construction of a $\top$-module

2.2.2. Reuse method overview

Based on the theory of modularity summarised in Section 2.2.1, we present a novel method for designing an ontology when knowledge is borrowed from several external

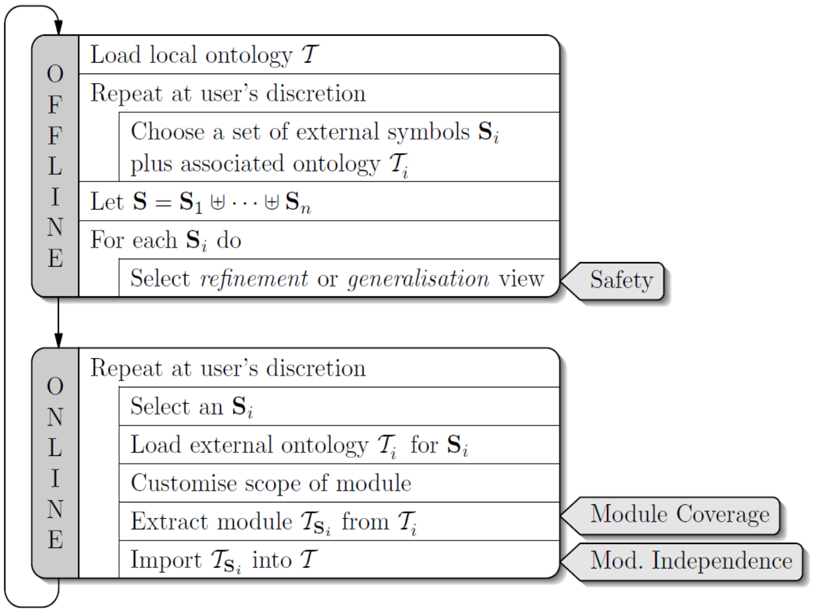

OFFLINE
Load local ontology $\mathcal{T}$
Repeat at user's discretion
Choose a set of external symbols $\mathbf{S}_i$ plus associated ontology $\mathcal{T}_i$
Let $\mathbf{S} = \mathbf{S}_1 \uplus \cdots \uplus \mathbf{S}_n$
For each $\mathbf{S}_i$ do
Select *refinement* or *generalisation* view
Safety
ONLINE
Repeat at user's discretion
Select an $\mathbf{S}_i$
Load external ontology $\mathcal{T}_i$ for $\mathbf{S}_i$
Customise scope of module
Extract module $\mathcal{T}_{\mathbf{S}_i}$ from $\mathcal{T}_i$
Import $\mathcal{T}_{\mathbf{S}_i}$ into $\mathcal{T}$
Module Coverage
Mod. Independence

Figure 2.5: The two phases of import with the required guarantees

ontologies. This method provides precise guidelines for ontology developers to follow, and ensures that a set of logical guarantees (*safety*, *module coverage* and *module independence*) will hold at certain stages of the design process. It is worth mentioning that this method is complementary to current ontology engineering methodologies. Moreover it assumes a single-developer scenario, thus collaboration issues are not considered. Chapter 3 presents several techniques, included our proposal, to support some of the necessary tasks in collaborative ontology development.

The cycle of the proposed method is given in Figure 2.5. This cycle consists of an *offline phase*—which is performed independently from the current contents of the external ontologies—and an *online phase*—where knowledge from the external ontologies is extracted and imported into the current ontology. Note that this separation is not strict: the first phase is called “offline” simply because it does not need to be performed online, however, the user may still choose to do so.

2.2.3. Tool support in the offline phase

The offline phase starts with an ontology $\mathcal{O}$ being developed (e.g. **JIAO**). The ontology developer specifies the set $\mathbf{S}$ of symbols[7] to be reused from external ontologies. These symbols can be split into different groups $\mathbf{S} = \mathbf{S}_1 \uplus \ldots \uplus \mathbf{S}_n$ where each $\mathbf{S}_i \subseteq \mathbf{S}$ represents the external symbols to be borrowed from a particular ontology

[7] Note that the labels for these symbols may be borrowed from domain thesaurus as in Section 2.1

$\mathcal{O}_i$ and where "$\uplus$" denotes that the $\mathbf{S}_i$ are pairwise disjoint. Note that at this stage it is only required to indentify groups of entities (i.e. $\mathbf{S}_i$) belonging to a same external source (i.e. $\mathcal{O}_i$). The specification of the concrete external ontologies (e.g. GALEN, NCI) is optional at this stage, and, even if indicated, the ontology identifier or URI may not refer to an existing ontology, but it may simply act as a temporary name.

Tool Support. ProSÉ provides functionality for declaring entities as external as well as for defining the external signature subgroups by means of an external ontology identifier or URI. As commented above, the ontology identifier could be provisional. This information is stored in the ontology using OWL annotations [MPSP09] as follows: we use an ontology annotation axiom per external ontology, an entity annotation axiom to declare an entity as external, and an entity annotation axiom per external entity to indicate its external ontology. The set of external entities with the same external ontology identifier can be viewed as one of the $\mathbf{S}_i$ (see top part of Figure 2.6).

Additionally, ProSÉ also provides an "online" operation in this stage in order to allow developers to browse through the external ontologies, if already specified, to choose the symbols they want to reuse.

2.2.3.1. Safety guarantee

At this stage, we want to ensure that the designer of $\mathcal{O}$ does not change the original meaning of the reused concepts, independently of what their particular meaning is in the external ontologies. This requirement can be formalised using the notion of safety introduced in Section 2.2.1.2:

Definition 2.6 (Safety Guarantee)**.** *Given an ontology $\mathcal{O}$ and signatures $\mathbf{S}_1, \ldots, \mathbf{S}_n$, $\mathcal{O}$* guarantees safety *if $\mathcal{O}$ is safe for $\mathbf{S}_i$ for all $1 \leq i \leq n$.*

In Sections 2.2.1.5 and 2.2.1.6 we argued that the simultaneous refinement and generalisation of the same signature group $\mathbf{S}_i$ may violate safety. To preserve safety, as already introduced in Section 2.2.1.5, we use $\bot$-locality and $\top$-locality conditions. $\bot$-locality for those group of symbols $\mathbf{S}_i$ to be refined and $\top$-locality for generalisation.

Proposition 2.7. *Let $\mathcal{O}$ be an ontology and $\mathbf{S} = \mathbf{S}_1 \uplus \ldots \uplus \mathbf{S}_n$ be the union of disjoint signatures. If, for each $\mathbf{S}_i$, either $\mathcal{O}$ is $\bot$-local or $\top$-local w.r.t. $\mathbf{S}_i$, then $\mathcal{O}$ guarantees safety w.r.t. $\mathbf{S}_1, \ldots, \mathbf{S}_n$.*

In order to achieve the safety guarantee at the end of the offline phase, we propose to follow the procedure sketched in Table 2.5.

Tool Support. ProSÉ allows for the specification, for each external signature group $\mathbf{S}_i$, whether it will be refined or generalised. Once the external entities have been declared and divided into these groups, the tool allows for safety checking of the ontology under development w.r.t. each group of external symbols separately. The safety check[8] uses $\bot$-locality ($\top$-locality) for signature groups that adopt the refinement (generalisation) view. Figure 2.6 shows the non-local axioms for the signature group $\mathbf{S}_{NCI} = \{Juvenile\_Rheumatoid\_Arthritis, Juvenile\_Chronic\_Polyarthritis\}$.

[8] A syntactic locality checker was implemented according to [GHKS08].

```
Input   Ontology $\mathcal{O}$,   disjoint signatures $\mathbf{S}_1, \ldots, \mathbf{S}_n$
        a choice among refinement and generalisation for each $\mathbf{S}_i$
Output  an ontology $\mathcal{O}_{\mathcal{L}}$ that guarantees safety

1: $\mathcal{O}_{\mathcal{L}} := \mathcal{O}$
2: while exists $\mathbf{S}_i$ such that $\mathcal{O}$ not local according to the selection for $\mathbf{S}_i$ do
3:     check $\begin{cases} \bot\text{-locality of } \mathcal{O}_{\mathcal{L}} \text{ w.r.t. } \mathbf{S}_i \text{ if } \mathbf{S}_i \text{ is to be refined} \\ \top\text{-locality of } \mathcal{O}_{\mathcal{L}} \text{ w.r.t. } \mathbf{S}_i \text{ if } \mathbf{S}_i \text{ is to be generalised} \end{cases}$
4:     if non-local then
5:         $\mathcal{O}_{\mathcal{L}}$ := repair $\mathcal{O}_{\mathcal{L}}$ until it is local for $\mathbf{S}_i$ according to the choice for $\mathbf{S}_i$
6:     end if
7: end while
8: return $\mathcal{O}_{\mathcal{L}}$
```

Table 2.5: A procedure for checking safety

2.2.4. Tool support in the online phase

In this phase the ontology designer imports the relevant knowledge from each of the external ontologies. As already argued, we aim at extracting only those modules from the external ontologies that are relevant to the reused symbols.

As shown in Figure 2.5, the import for each external ontology $\mathcal{O}_i$ is performed in four steps: (1) First, the $\mathcal{O}_i$ for $\mathbf{S}_i$ is loaded, and therefore the ontology designer commits to a particular version of it. (2) Then, $\mathbf{S}_i$ can be exteded with super-concepts and sub-concepts in order to customize the scope of the module to be extracted from $\mathcal{O}_i$. (3) A module $\mathcal{O}_{\mathbf{S}_i}$ ($\bot$-module or a $\top$-module) of $\mathcal{O}_i$ is extracted for the customized $\mathbf{S}_i$ according to Definition 2.3. (4) Finally, $\mathcal{O}_{\mathbf{S}_i}$ is imported into $\mathcal{O}$.

2.2.4.1. Module coverage guarantee

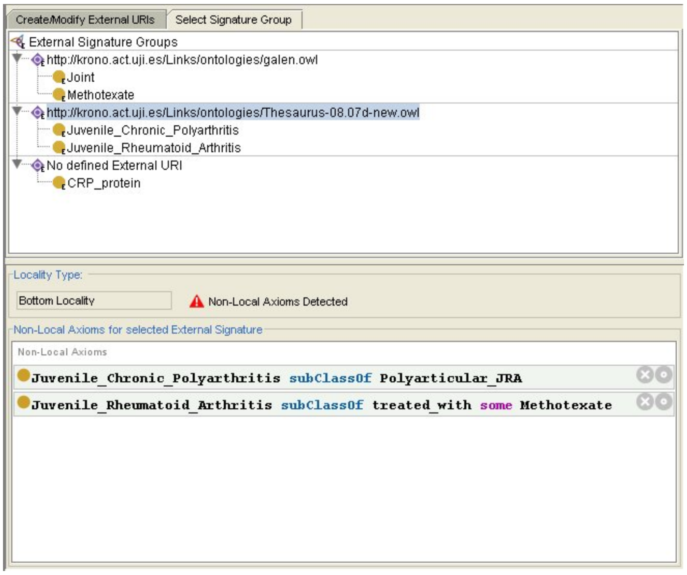
The fragment extracted for each customised signature in the online phase must satisfy the module coverage guarantee. As seen in Section 2.2.1.5 Proposition 2.5, $\bot$-locality and $\top$-locality can be used for extracting modules.

Definition 2.7 (Module Coverage Guarantee). *Let* $\mathbf{S}_i$ *be a signature and* $\mathcal{O}_{\mathbf{S}_i} \subseteq \mathcal{O}_i$ *ontologies.* $\mathcal{O}_{\mathbf{S}_i}$ guarantees coverage of $\mathbf{S}_i$ in $\mathcal{O}_i$ *if* $\mathcal{O}_{\mathbf{S}_i}$ *is a module for* $\mathbf{S}_i$ *in* $\mathcal{O}_i$.

As shown in Tables 2.3 and 2.4, the extraction of $\bot$-modules or $\top$-modules may introduce symbols not in $\mathbf{S}_i$, and potentially unnecessary. To make the module as small as possible, we can extract the $\top$-module of the $\bot$-module for $\mathbf{S}_i$ as in Proposition 2.6.

Proposition 2.8. *Let* $\mathcal{O}_{\mathbf{S}_\bot} = \bot\text{–}\mathsf{Module}(\mathcal{O}_i, \mathbf{S}_i)$, *and* $\mathcal{O}_{\mathbf{S}_\top} = \top\text{–}\mathsf{Module}(\mathcal{O}_{\mathbf{S}_\bot}, \mathbf{S}_i)$. *Then* $\mathcal{O}_{\mathbf{S}_\top}$ *guarantees coverage of* $\mathbf{S}_i$ *in* $\mathcal{O}_i$.

Proof. Let $\mathbf{S}_\top = \mathsf{Sig}(\mathcal{O}_{\mathbf{S}_\top}) \cup \mathbf{S}$. By Proposition 2.6 $\mathcal{O}_{\mathbf{S}_\top}$ is a module for $\mathbf{S}$ in $\mathcal{O}$. By Definition 2.7, $\mathcal{O}_{\mathbf{S}_\top}$ guarantees coverage of $\mathbf{S}$ in $\mathcal{O}$. □



External Signature Groups
http://krono.act.uji.es/Links/ontologies/galen.owl
Joint
Methotexate
http://krono.act.uji.es/Links/ontologies/Thesaurus-08.07d-new.owl
Juvenile_Chronic_Polyarthritis
Juvenile_Rheumatoid_Arthritis
No defined External URI
CRP_protein

Locality Type:
Bottom Locality
Non-Local Axioms Detected

Non-Local Axioms for selected External Signature
Non-Local Axioms
Juvenile_Chronic_Polyarthritis subClassOf Polyarticular_JRA
Juvenile_Rheumatoid_Arthritis subClassOf treated_with some Methotexate

Figure 2.6: Signature groups and safety test

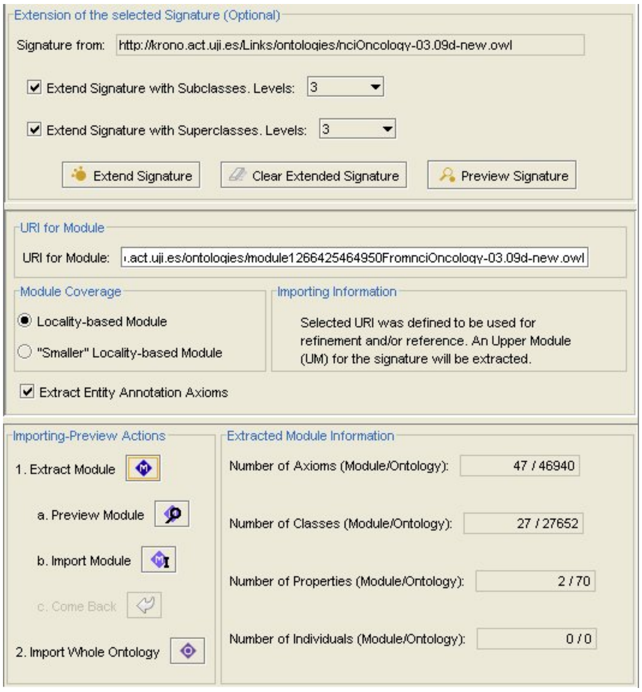
Tool Support. As already commented, the selected external signature group $\mathbf{S}_i$ can be customised by adding super-concepts and sub-concepts of their symbols. ProSÉ provides the functionality for previewing the concept hierarchy of the corresponding external ontology for this purpose. Once the specific signature group under consideration has been customised, a module for it can be extracted[9]. As it is shown in Figure 2.7, the user can compute the module, preview it in a separate frame, and either import it or cancel the process and come back to the signature customisation stage. Additionally, the user has also the option to import the whole external ontology instead of importing a module.

2.2.4.2. Module independence guarantee

The import of the module $\mathcal{O}_{\mathbf{S}_i}$ for $\mathbf{S}_i$ into $\mathcal{O}$ is the last step in the offline phase. The effect of this import is that the ontology $\mathcal{O}$ being developed evolves to $\mathcal{O} \cup \mathcal{O}_{\mathbf{S}_i}$. This new ontology might violate safety over the remaining signature groups $\mathbf{S}_j$. Such an effect is obviously undesirable. Hence the following guarantee should be provided:

Definition 2.8 (Module Independence Guarantee). *Let $\mathcal{O}$ be the importing ontology, $\mathbf{S}_i, \mathbf{S}_j$ be signatures.* Module independence is guaranteed *if, for all $\mathcal{O}_i$ with $\mathsf{Sig}(\mathcal{O}) \cap \mathsf{Sig}(\mathcal{O}_i) \subseteq \mathbf{S}_i$ and for all $\mathcal{O}_j$ with $\mathsf{Sig}(\mathcal{O}) \cap \mathsf{Sig}(\mathcal{O}_j) \subseteq \mathbf{S}_j$, it holds that $\mathcal{O} \cup \mathcal{O}_i \cup \mathcal{O}_j$ is a conservative extension of both $\mathcal{O} \cup \mathcal{O}_i$ and $\mathcal{O} \cup \mathcal{O}_j$.*

[9] A module extractor was implemented, on top of the locality checker, following [CHKS07, GHKS08]


Extension of the selected Signature (Optional)

Signature from: http://krono.act.uji.es/Links/ontologies/nciOncology-03.09d-new.owl

☑ Extend Signature with Subclasses. Levels: 3

☑ Extend Signature with Superclasses. Levels: 3

URI for Module

URI for Module: ι.act.uji.es/ontologies/module1266425464950FromnciOncology-03.09d-new.owl

Module Coverage

● Locality-based Module

○ "Smaller" Locality-based Module

Importing Information

Selected URI was defined to be used for refinement and/or reference. An Upper Module (UM) for the signature will be extracted.

☑ Extract Entity Annotation Axioms

Extracted Module Information

Number of Axioms (Module/Ontology):	47 / 46940
Number of Classes (Module/Ontology):	27 / 27652
Number of Properties (Module/Ontology):	2 / 70
Number of Individuals (Module/Ontology):	0 / 0

Figure 2.7: Signature extension and module import

According to Definition 2.8, importing a module for a signature $\mathbf{S}_i$ from an external ontology $\mathcal{O}_i$ into $\mathcal{O}$ should not affect the safety of $\mathcal{O}$ w.r.t. to the other external ontologies $\mathcal{O}_j$.

In the conference paper [JRGS$^+$08b] was established that the module independence guarantee always holds provided the signature of external ontologies are disjoint and the importing ontology is safe for them. However, in the technical report [JRGS$^+$08d] was shown that external ontologies may contain unsafe or dangeorus axioms which can cause the violation of the module independece guarantee.[10] Thus, external ontologies must also be local w.r.t. the *empty signature*, that is, to be consistent, coherent (as in Definition 1.1) and not including unsafe GCIs [GPSK06] such as $\top \sqsubseteq \mathsf{Joint}$ (redefinition of the universal concept) or $\top \sqsubseteq \{\mathsf{ernesto}\}$ (limiting the size of the domain of interpretation).

Note that, in practice, the requirement $\mathsf{Sig}(\mathcal{O}_1) \cap \mathsf{Sig}(\mathcal{O}_2) = \emptyset$ is almost always met since different reference ontologies usually have different namespaces. However, although these ontologies are unrelated from a logical point of view, they may intuitively overlap, that is, they may partially refer to the same entities. A further discussion is given in Chapter 4 where we analyze the logical consequences of integrating independently developed ontologies through mappings (i.e. entity correspondences).

[10] Thanks to Thomas and Uli for noticing this error

Tool Support. ProSÉ has two types of import. The first type performs a classic import merging the module with the local knowledge, thus, module entities are considered as local and no further safety checks can be done over this signature. The second one, however, keeps separated the imported module entities and the local knowledge. Module entities remain as external entities and therefore guarantees can still be kept over this external signature.

2.2.5. Related work and discussion

The extraction and reuse of ontology modules has been treated in different ways. In the literature we can find several approaches (see [PS09] for an overview) which try to extract a fragment from an ontology according a set of requirements (e.g., a query or a set of entities). Two ontology modularization groups can be distinguished: traversal based and semantics preservation based. The former group is represented by those fragmentation techniques which rely on syntactic heuristics for traversing the ontology and detecting relevant axioms. Prominent examples are PROMPT [NM04, NM09], Seidenber's segmenter [SR06, Sei09], OntoPath [JRLS$^+$05, JRLNS07], Nebot's fragmenter [NL09] and Stuckenschmidt's partitioner [SK04, SS09]. These techniques do not attempt to formally specify the intended outputs and do not provide any guarantee about the coverage of the extracted fragments.

OntoPath [JRLS$^+$05, JRLNS07] represents our first effort in extracting ontology fragments according to the user requirements, which were given by means of a XPath-like [HM00] query. OntoPath system was also focused on the storage and management of ontologies within a graph-based database [Rio]. Nebot's fragmenter [NL09], used in Section 2.1.2 to extract fragments from UMLS-Meta, is an evolution of OntoPath were the storage of the ontologies were improved and the query language is signature-driven like in the locality module extractor.

Together with the locality-based approach [GHKS08], in which the presented reuse method has been based, there are other formal proposals to preserve the semantics of extracted modules and to "safely" combine them. Most of these proposals, such as $\mathcal{E}$-connections [GPS09], Distributed Description Logics [ST09] and Package-based Description Logics [BVSH09] propose a specialised semantics (i.e., extended DL semantics) for controlling the interaction between the importing and the imported modules to avoid side-effects; for an overview see [CK07, WHB07]. In contrast, our method and tool assume that reuse is performed by simply building the logical union of the axioms in the modules preserving the standard semantics. Moreover, we also provide the user with a collection of reasoning services, such as safety, to check for side-effects.

Finding the most suitable ontology to reuse has also been an important topic in ontology reuse. Our method, however, does not tackles this problem and considers that the ontologies to be reused are known. This premise does not always necessarily hold, thus, techniques to evaluate available ontologies according to a set of the requirements and competencies, such as [Ala06, DLE$^+$07, DTI07, dBG$^+$07, SNNB08, FOSM09, LED$^+$10], are required. Note that, the first step in ontology reuse requires the specification of the set of concepts to be reused (see Section 2.2.3). The evaluation of the target ontologies will be based on the similarity techniques applied over their concepts and the set of representative required concepts. These similarity techniques often rely on the discovering of mappings (i.e., alignments) between the ontology concepts and the requirements. The use of thesauri, as already discussed in Section 1.3.2, would ease the identification of these correspondences.

Finally, [TSK09] proposes an approach based on our reuse method which permits the simultaneous refinement and generalization of the same signature group. However, as stated in Section 2.2.1.6, this kind of reuse may compromise safety and should be avoided. Thus, at the requirements stage, it is necessary to face with a fundamental choice depending on whether the ontology designer wants to either refine or generalise a particular signature group to be reused from an external ontology.

CHAPTER 3

Supporting concurrent ontology evolution

The development of large domain ontologies usually requires expertise about different subdomains. For example the descriptions in the NCI ontology [HdCD$^+$05, GFH$^+$03] cover diseases, anatomy, genes, drugs and so on. Therefore, such large ontologies are likely to be developed concurrently and collaboratively by a team of knowledge engineers and domain experts. Furthermore, these ontologies are in continuous evolution [HKR08]. The developers of an ontology can be geographically distributed and may contribute in different ways and to different extents. GALEN, NCI and SNOMED are prominent examples of collaborative ontology development projects (see [SR07b] for a survey on multi-user ontology projects):

- The *NCI ontology* is being developed by 20 full time editors and one curator. Each editor works in a separate copy of the ontology and, at the end of a two week editing cycle, the curator uses a workflow management tool to integrate, review and approve the changes made by each editor [dCWF$^+$09].

- The development of *GALEN ontology* adopts a locking mechanism [SR07a, SR07b] to prevent editing conflicts. To this end, the ontology has been split in 2738 modules which can be extended concurrently, but only one user can modify a module at the same time. When a version of GALEN is released (commonly every 6 months), developers should return and unlock their extended modules. A single curator is in charge of merging modules, correcting errors, removing redundancy and classifying the resulting ontology. This process usually takes one or two weeks.

- The *SNOMED CT* development team is composed by a central team and four (distributed geographically) groups which extend different parts of the ontology. Every two weeks these parts are integrated and conflicts are resolved by the central team.

Designing and maintaining such large ontologies is, therefore, a highly complex process, which involves many complicated tasks:

1. to agree on a unified conceptual design and common modelling guidelines;
2. to assign different responsibilities and tasks to each group of developers;
3. to track and manage the frequent changes to the ontology made by different developers from distributed locations;
4. to compare different versions of the ontology lexically (e.g., names of the introduced ontology entities), structurally (e.g., shape of the axioms), and semantically (e.g., logical consequences);
5. to detect and reconcile conflicting views of the domain by merging different ontology versions; and
6. to minimise the introduction of errors (e.g., to ensure that the ontology does not have unintended logical consequences).

In this chapter, we propose ContentCVS [1] [JRGHL09a, JRGHL09b, JRGHL10b], a framework and tool support[2] that adapts the Concurrent Versioning paradigm. The ontology editing in ContentCVS is *asynchronous*, thus developers make changes concurrently over their local copies.

Currently, ContentCVS implements a *passive versioning method*. Therefore, versions must be compared to detect changes and potential conflicts. Change and conflict detection are based on novel techniques that take into account the structure and semantics of the ontology versions by reusing the notions of structural and *semantic differences* between ontologies. ContentCVS follows a method or workflow, which exploits logic-based techniques, to iteratively support the acceptance and rejection of changes in order to avoid the occurrence of errors (i.e. unintended consequences) within the semantic difference.

ContentCVS partially addresses tasks 3-6 previously mentioned. Thus, it should be seen as a complementary framework to the state-of-the-art approaches addressing ontology evolution and benefit from each other. These approaches are presented later on. A further description of ContentCVS is given in Section 3.2. Moreover, in Section 3.3 a preliminary performance evaluation and user study is provided.

[1] A **C**oncurrent **ONT**ology **EN**gineering **T**ool.

[2] A Protégé 4 plugin is freely available for download: `http://krono.act.uji.es/people/Ernesto/contentcvs`

3.1. State of the art

Ontology evolution (refer to [Sto04, FMK$^+$08, LM08, Pal09] for comprehensive surveys) is defined as the *ability to manage ontology changes and their effects* [NK04].

In the literature there exists several efforts analyzing the semantics of changes and their propagation to dependent entities (e.g., instances) and ontologies. Sections 3.1.1-3.1.3 discusses and differentiates three types of approaches depending on the main issue they address, namely: *change representation and management*, *logic-consistency preservation* and *concurrent and collaborative development*.

3.1.1. Change representation approaches

Ontology evolution and database-schema evolution can be considered similar problems. However, ontologies have important peculiarities [NK04, Sto04] with respect to database-schemas, such as the unclear separation between schema and data, or the implicit semantics of ontologies. Thus, ontology changes may have side effects which do not happen in traditional schema evolution.

Ontology changes have been characterized and classified in the literature according to its implications. Moreover, changes are split into atomic and composite. Prominent examples of change representation approaches can be found in [KN03, Sto04, Kle04, NK04, NCLM06, PHCGP09]. These approaches intend to keep record of every change providing them with formal meta-information in order to collect at least the *Five Ws* questions about the performed operation.

According to [NK04], previous approaches perform an active or traced versioning. However, in passive or untraced versioning ontology changes are not tracked. In such cases, if two ontology versions are required to be compared additional methods are necessary. For example, [KFKO02, NKKM04] use a number of heuristics to extract a *structural difference* between two ontology versions. Recently the literature has also paid attention to he notion of *semantic or logic* difference and several approaches using it can be found [KWW08, KWZ08, KAPS08, EFV10].

ContentCVS, as already mentioned, performs a passive versioning. Currently, change and conflict detection in ContentCVS is only based on the notions of structural and *semantic differences*, thus, a formal representation of changes could enrich this process.

3.1.2. Consistency preservation approaches

Modelling errors are rather common during the ontology life cycle and they manifest themselves as non desired logical consequences. Haase and Stojanovic [HS05] distinguished three types of consistency: *structural*, *logical* and *user defined*. For the purposes of this dissertation only the last two are considered.

Unsatisfiability is easy to detect but not always easy to solve since the source of an error is typically hard to understand. In the literature a number of approaches have proposed different solutions to represent and understand the source of an unsatisfiable concept. These approaches can be split into two sets[3]: *glass box techniques* [SC03, PT06, KPSH05, KPSG06, SHCvH07, BPS07] and *black box techniques* [HS05, WHR$^+$05, KPHS07, HPS08b, JQH09]. The former set is composed by those techniques that extend the operation of tableau-based reasoning algorithms in order to keep track of the followed path when getting a clash. The later techniques treat the reasoner as a black box and they only requires its output. Thus, glass box techniques are reasoner dependent whereas black box techniques do not require a particular reasoner nor description logic.

Both glass and black box techniques rely on the notion of *justification* or *MUPS* (Minimal Unsatisfiability Preserving Sub-TBoxes), originally introduced in [SC03]. Justifications are the set of axioms which are the cause of a logical consequence (e.g., concept subsumption, unsatisfiability).

Definition 3.1 (Justification)**.** *Let $\mathcal{O} \models \alpha$. A justification for α in $\mathcal{O}$ is an ontology $\mathcal{O}' \subseteq \mathcal{O}$ satisfying the following properties: (i) $\mathcal{O}' \models \alpha$, and (ii) there is no $\mathcal{O}'' \subset \mathcal{O}'$ s.t. $\mathcal{O}'' \models \alpha$.*

User-defined consistency involves user requirements and therefore it has to be manually or semi-automatically detected. Consequences can be logically correct, that is, all concepts are satisfied, but concepts can still be missclassified (e.g., $Arthritis \sqsubseteq Drug$) and/or individuals be placed in an incorrect class (e.g. $ernesto \in Professor$). Approaches in [KPHS07, HPS08b, JQH09] are not only focused on unsatisfiability but also in explaining general DL consequences. Moreover, [HPS08b, JQH09] also try to provide more precise explanations in order to make easier the reparation of such unintended consequences.

ContentCVS extends and adapts ontology debugging and repair techniques in order to fix identified logical errors (both unsatisfiability and general unintended consequences) in a concurrent setting.

3.1.3. Concurrent ontology development frameworks

Maintaining medium and large ontologies is usually done in a collaborative way. This involves a highly complex process which involves not only tracking and managing the changes to the ontology as in traditional ontology development, but also allowing communication and argumentation between experts, assigning roles and access policies to the knowledge, reconciling conflicting views of the domain from different developers, minimising the introduction of errors (e.g., ensuring that the ontology does not have unintended logical consequences), and so on.

In the literature we can find several frameworks which propose solutions to different necessities in the collaborative ontology lifecycle. In the earlier stages of our research, we presented a survey and a set of methodology requirements for the collaborative development of ontologies [JRL06]. The work carried out was rather prelimi-

[3] The referenced approaches has been distributed to each group according to its main contribution, however some of them present hybrid techniques and/or perform a comparison of both techniques

nary but some interesting ideas were discussed, such as the differentiation of public, private and agreed spaces for the developed knowledge, or the possibility of coexistence of local fragments of knowledge in conflict with the global knowledge or other local points of view.

Next we present an up-to-date list of current frameworks and approaches. The most of these frameworks have tried to control the set of concurrent changes over the ontology using mechanisms to restrict the access to the ontology and/or defining collaborative workflows to discuss and reconcile changes potentially conflictive.

- Collaborative Protégé [NTdCM08, TNTM08, SNTM08] is a prominent an quite complete example of a collaborative change management system. It allows ontology developers to hold discussions, chats, and annotate changes. It also assign roles and access policies to developers. Changes are formally annotated as instances of an ontology [KN03, NCLM06] and their commitment follows a collaborative workflow [SNTM08] to discuss and analyze potential errors. Moreover, ontology versions can also be compared using the PromptDiff algorithm [NKKM04], which calculates a *structural difference* between them using a number of heuristics.

- Palma [Pal09] presents in his PhD dissertation a framework to manage and propagate changes in collaborative ontology development scenarios. The development process is formalized by means of a collaborative editorial workflow [PHJ08]. Additionally, changes are characterized and represented using a meta-ontology [PHCGP09].

- DOGMA-MESS [dMLM06, LD08] also represents a prominent framework and tool support where the importance of developing common conceptual models is emphasised, especially when the process of collaborative ontology engineering crosses the boundaries of a single organisation. DOGMA-MESS contributes to the field with the following characteristics: (1) there is an explicit separation, as proposed in Section 1.3.2, of the lexical representation of concepts from its semantic description (i.e., logic-based axioms); thus, conflicts related to the use of different labels referring to the same entity are minimized. (2) Ontology developers extends a consensual upper ontology following some predefined restrictions on the extension of an upper knowledge (i.e., reuse policies [LdMM07]), then if a reuse policy is violated a conflict arises and must be revised.

- [RSDT08] have proposed a general framework to develop ontologies using a version control system. This framework does not provide a concrete mechanism or workflow to validate/reconcile changes, but a extensible conflict management operation by integrating state-of-the-art methods. Additionally, [RSDT08] also proposes the management of different versions and different development branches. These branches may be required to be merged and reconciled or they can coexist, even when they are incompatible, for application purposes as proposed in [JRL06, DBC06].

- The definition of formal or semi-formal argumentation models in order to achieve a consensus over changes has also been an interesting research topic within the literature. DILIGENT [TPSS05, TSL$^+$07, PTS09] and HCOME [KV06] are representative methodologies which follow a formal argumentation model based on the IBIS model [KR70]. For example, DILIGENT exploits an argumentation ontology to describe and store (as instances) the different discussion threads. Thus, the revision of past decisions and conclusions can be easily retrieved and reviewed, unlike traditional communications means like e-mail or chat. Cicero [DEB$^+$08] implements the DILIGENT methodology, and [Kot08] presents a system based on the HCOME argumentation model.

- Authors in [BCH06] presents an extension of the standard description logics called package-based DL (P-DL). The ontology is subdivided in packages or modules, which can be nested in other packages resulting in a hierarchy. The use of this package hierarchy allow ontology managers to define access policies and to restrict the visibility of entities and axioms, with different levels of granularity, in order control the concurrency of changes.

- The approach [SKKM03, KSKM07] presents a dependency management framework and tool support in which collaboratively developed ontologies are divided in several modules organized within a hierarchy of dependencies. The edition of modules is asynchronous, and only one developer can modify a module at the same time. That is, modules are *locked* before starting its extension. Changes over a module may influence dependent modules, thus, in order to detect and avoid undesired cases (e.g., inconsistency) changes are characterized and a number of countermeasures are taken depending on the change. In this way, dependent module owners can reject changes in case of conflict.

- Seidenberg and Rector [SR07a, SR07b] also proposes a locking mechanism to allow concurrent extension of ontologies. Developers selects the entity to be modified and then extract a fragment or segment [SR06, Sei09] for this entity, which will be locked. They also propose several types of locking allowing different degrees of freedom.

- WebODE [CFLGPV02], WebOnto [Dom98] and OntoEdit [SEA$^+$02] were pioneers systems giving support for collaborative and concurrent editing of the same ontology. WebODE supports the concurrent edition by means of synchronization mechanisms and role assignment. OntoEdit presents an entity-like locking mechanism, thus, OntoEdit modellers must lock the entity or set of entities before editing them. WebOnto forces the locking of the whole ontology to be modified but dependent sub-ontologies are kept unlocked. Tadzebao complements WebOnto with support to keep synchronous and asynchronous discussions between ontology modellers.

As stated in [BCH06, SR07a] the precise guarantees provided by methods based on access policies are not clear. Thus, ContentCVS techniques may serve to guarantee that the impact of merging independent set of changes is error free.

3.2. Assessment of concurrent ontology changes with ContentCVS

This section is devoted to the analysis of ContentCVS framework and tool. We start considering a motivating use case based on the collaborative and concurrent development of **JIAO**. This example is intended only for illustration purposes and hence it is rather simplistic. The types of conflicting ontology changes illustrated in this section, however, are indeed realistic as we will see later on in Section 3.3.1, where we analyse a sequence of versions of a medical ontology used in a real scenario.

	Ontology $\mathcal{O}^0$
α_1	$\mathsf{RA} \sqsubseteq \mathsf{Disease}$
α_2	$\mathsf{Systemic\_Disease} \sqsubseteq \mathsf{Disease} \sqcap \exists \mathsf{affects.WholeBody}$
α_3	$\mathsf{Disease} \sqcap \exists \mathsf{affects.WholeBody} \sqsubseteq \mathsf{Systemic\_Disease}$
α_4	$\mathsf{Disease} \sqcap \exists \mathsf{suffered\_By.Child} \sqsubseteq \mathsf{Juvenile\_Disease}$
α_5	$\mathsf{Negative\_RF} \sqcap \mathsf{Positive\_RF} \sqsubseteq \bot$
α_6	$\mathsf{AbnormalRA} \sqsubseteq \mathsf{RA} \sqcap \forall \mathsf{hasRF.Negative\_RF}$
α_7	$\mathsf{MultiJoint\_Disease} \sqsubseteq \mathsf{Disease} \sqcap \geqslant 5\, \mathsf{affects.Joint}$

Table 3.1: A fragment of an ontology about arthritis

Suppose that two developers, John and Anna, independently extend a version $\mathcal{O}^0$ of the ontology in Table 3.1 by describing types of systemic arthritis and juvenile arthritis respectively. To this end, both John and Anna define a kind of arthritis called JIA (Juvenile Idiopathic Arthritis). Hence, even if largely distinct, the domains described by John and Anna overlap, which may lead to conflicts. For simplicity, in what follows we only consider John and Anna's descriptions of JIA.

Suppose that John and Anna construct their respective versions $\mathcal{O}^1$ and $\mathcal{O}^2$ by adding to $\mathcal{O}^0$ the axioms $(\Delta\mathcal{O})^1$ and $(\Delta\mathcal{O})^2$ from Table 3.2 (i.e., $\mathcal{O}^1 = \mathcal{O}^0 \cup (\Delta\mathcal{O})^1$ and $\mathcal{O}^2 = \mathcal{O}^0 \cup (\Delta\mathcal{O})^2$). Some of the axioms added by John and Anna are the same (e.g., β_1), or present only minor (and semantically irrelevant) differences (e.g., β_2 and β_2'); however, other axioms are clearly different (e.g., γ_3 and δ_3).

To compare John and Anna's conceptualisations of the domain, the upper part of Table 3.3 presents some axioms and their entailment status w.r.t. $\mathcal{O}^1$ and $\mathcal{O}^2$. The table shows that John and Anna agree on some points; e.g., both think that Polyarticular JIA is a kind of disease (entailment σ_3) and neither claimed that every JIA is also a Systemic JIA (non-entailment σ_4). However, John's and Anna's views also present significant differences; e.g., John defined the notion of Olyarticular JIA , whereas Anna did not, and Anna's conceptualisation implies that JIA is a juvenile disease (entailment σ_2), whereas John's does not.

John and Anna's changes could be reconciled by building the union $\mathcal{O}^3 = \mathcal{O}^1 \cup \mathcal{O}^2$ of their ontologies. Due to complex interactions between $\mathcal{O}^1$ and $\mathcal{O}^2$, however, $\mathcal{O}^3$ entails new consequences which did not follow from either $\mathcal{O}^1$ or $\mathcal{O}^2$ alone; some of these are shown in the lower part of Table 3.3, together with an indication as to whether the consequence is desirable. Although some of these new consequences may

	Ontology $(\Delta\mathcal{O})^1$:		**Ontology** $(\Delta\mathcal{O})^2$:
β_1	$\mathsf{RA} \sqsubseteq \exists\mathsf{hasRF}.\top$		$\mathsf{RA} \sqsubseteq \exists\mathsf{hasRF}.\top$
β_2	$\mathsf{JIA} \sqsubseteq \exists\mathsf{treatment}.(\mathsf{Steroid} \sqcup \mathsf{DMAR})$	β_2'	$\mathsf{JIA} \sqsubseteq \exists\mathsf{treatment}.(\mathsf{DMAR} \sqcup \mathsf{Steroid})$
γ_1	$\mathsf{JIA} \sqsubseteq \mathsf{RA} \sqcap \mathsf{Systemic\_Disease}$	δ_1	$\mathsf{JIA} \sqsubseteq \mathsf{RA} \sqcap \exists\mathsf{suffered\_By}.\mathsf{Child}$
γ_2	$\mathsf{RA} \sqcap \mathsf{Systemic\_Disease} \sqsubseteq \mathsf{JIA}$	δ_2	$\mathsf{JIA} \sqcap \exists\mathsf{affects}.\mathsf{WholeBody} \sqsubseteq \mathsf{SystemicJIA}$
γ_3	$\mathsf{Poly\_JIA} \sqsubseteq \mathsf{JIA} \sqcap \mathsf{MultiJoint\_Disease}$	δ_3	$\mathsf{Poly\_JIA} \sqsubseteq \mathsf{JIA} \sqcap {=}\,3\,\mathsf{affects}.\mathsf{Joint}$
γ_4	$\mathsf{Poly\_JIA} \sqsubseteq \mathsf{AbnormalRA}$	δ_4	$\mathsf{Poly\_JIA} \sqsubseteq \forall\mathsf{hasRF}.\mathsf{Positive\_RF}$
γ_5	$\mathsf{Oly\_JIA} \sqsubseteq \mathsf{JIA} \sqcap \neg\mathsf{Poly\_JIA}$	δ_5	$\mathsf{SystemicJIA} \sqsubseteq \mathsf{JIA} \sqcap \exists\mathsf{hasSymptom}.\mathsf{Fever}$

Table 3.2: Versions $\mathcal{O}^1 = \mathcal{O}^0 \cup (\Delta\mathcal{O})^1$ and $\mathcal{O}^2 = \mathcal{O}^0 \cup (\Delta\mathcal{O})^2$ of an ontology $\mathcal{O}^0$

σ	**Axiom:**	$\mathcal{O}^1 \overset{?}{\models} \sigma$,	**follows from:**	$\mathcal{O}^2 \overset{?}{\models} \sigma$,	**follows from:**
σ_1	$\mathsf{Oly\_JIA} \sqsubseteq \mathsf{Systemic\_Disease}$	Yes	γ_1, γ_5	No	—
σ_2	$\mathsf{JIA} \sqsubseteq \mathsf{Juvenile\_Disease}$	No	—	Yes	$\alpha_1, \alpha_4, \delta_1$
σ_3	$\mathsf{Poly\_JIA} \sqsubseteq \mathsf{Disease}$	Yes	$\alpha_1, \gamma_1, \gamma_3$	Yes	$\alpha_1, \delta_1, \delta_3$
σ_4	$\mathsf{JIA} \sqsubseteq \mathsf{SystemicJIA}$	No	—	No	—

σ	**Axiom:**	$\mathcal{O}^3 \overset{?}{\models} \sigma$,	$\mathcal{O}^1 \overset{?}{\models} \sigma$,	$\mathcal{O}^2 \overset{?}{\models} \sigma$,	**follows from:**	**Desirable?**
σ_4	$\mathsf{JIA} \sqsubseteq \mathsf{SystemicJIA}$	Yes	No	No	$\gamma_1, \alpha_2, \delta_2$	No
σ_5	$\mathsf{Poly\_JIA} \sqsubseteq \bot$	Yes	No	No	$\gamma_4, \beta_1, \delta_4, \alpha_5, \alpha_6$ $\alpha_7, \gamma_3, \delta_3$	No
σ_6	$\mathsf{Oly\_JIA} \sqsubseteq \mathsf{Juvenile\_Disease}$	Yes	No	No	$\gamma_5, \alpha_1, \alpha_4, \delta_1$	Yes

Table 3.3: Example Subsumption Relations in $\mathcal{O}^1$, $\mathcal{O}^2$, and $\mathcal{O}^3 = \mathcal{O}^1 \cup \mathcal{O}^2$

be desirable (e.g., σ_6) others are clearly undesirable, and indicate modelling errors in the merged ontology (e.g., σ_4 and σ_5).

This example illustrates some of the challenges of collaborative ontology development. The development of an ontology may be the responsibility of several developers, each of whom typically makes small but relatively frequent modifications to the ontology. In this setting, developers need to regularly merge and reconcile their modifications to ensure that the ontology captures a consistent unified view of the domain. The changes performed by different users may, however, interact and conflict in complex ways. Tools supporting collaboration should therefore provide means for: *(i)* keeping track of ontology versions and changes and reverting, if necessary, to a previously agreed upon version, *(ii)* comparing potentially conflicting versions and identifying conflicting parts, *(iii)* identifying errors in the reconciled ontology constructed from the conflicting versions, and *(iv)* suggesting possible ways to repair the identified errors with a minimal impact on the ontology.

In order to address *(i)*, we propose to adapt the Concurrent Versioning paradigm to ontology development as described in Section 3.2.1. To address *(ii)* we propose a notion of *conflict* between ontology versions and provide means for identifying conflicting parts based on it, as described in Section 3.2.2. To address *(iii)* we propose in Section 3.2.3 a framework for comparing the entailments that hold in the compared versions and in the reconciled ontology, based on the notion of *deductive difference* [KWW08, KWZ08] and also describe techniques for helping users decide which of the reported entailments are intended. Finally, to address *(iv)*, we propose in Section 3.2.3 several improvements to existing techniques for ontology debugging and repair, and we adapt them to a collaborative setting.

In Sections 3.2.1, 3.2.2, and 3.2.3, we describe both our general approach and algorithmic techniques as well as their implementation in ContentCVS. Section 3.2.4 reuses ContentCVS techniques to guarantee an desired local evolution of the ontology.

3.2.1. CVS-based collaboration

In software engineering, a successful paradigm for collaboration in large projects has been the Concurrent Versioning Paradigm. A Concurrent Versioning System (CVS) uses a client-server architecture: a CVS server stores the current version of a project and its change history; CVS clients connect to the server to *check out* a copy of the project, allowing developers to work on their own *local copy*, and then later to *commit* their changes to the server. This allows several developers to make changes concurrently to a project. To keep the system in a consistent state, the server only accepts changes to the latest version of any given project file. Developers should hence use the CVS client to regularly commit their changes and *update* their local copy with changes made by others. Manual intervention is only needed when a *conflict* arises between a committed version in the server and a yet-uncommitted local version. Conflicts are reported whenever the two compared versions of a file are not *equivalent* according to a given notion of equivalence between versions of a file.

Our tool ContentCVS closely follows the CVS paradigm. The most recent version $\mathcal{O}^R$ of the ontology, which represents the developers' shared understanding of the domain, is kept in a server's shared repository. Each developer with access to the repository maintains a local copy $\mathcal{O}^L$ of the ontology, which can be modified at will. This local copy can be either in conflict with $\mathcal{O}^R$ ($\mathsf{conflict} = \mathsf{true}$) or not in conflict ($\mathsf{conflict} = \mathsf{false}$). Furthermore, the system maintains version numbers v_R and v_L for $\mathcal{O}^R$ and $\mathcal{O}^L$ respectively as well a local *backup* copy $\mathcal{O}^L_{\mathsf{bak}}$ of the latest local version that was "written" to the repository.

At any time, a developer can access the repository using one of the following basic operations: *export*, *check-out*, *update* and *commit*. These operations involve checking whether two ontology files $\mathcal{O}$ and $\mathcal{O}'$ are equivalent under a specific notion of equivalence between ontology files, which will be introduced in Section 3.2.2 (denoted $\mathcal{O} \sim \mathcal{O}'$).

The *checkout* operation (Figure 3.1(a)) allows a developer to acquire a local copy $\mathcal{O}^L$ of $\mathcal{O}^R$, provided that $\mathcal{O}^L$ does not already exist. The ontology resulting from a successful checkout is obviously in a non-conflicted state (i.e., $\mathsf{conflict} = \mathsf{false}$), and it inherits the version number v_R of $\mathcal{O}^R$.

The *export* operation (Figure 3.1(b)) allows a developer to create a new repository, provided that no such repository already exists. The local ontology is then *exported* to the repository and the version numbers of $\mathcal{O}^L$ and $\mathcal{O}^R$ are initialised.

The *update* operation (Figure 3.2) allows ontology developers to keep their local

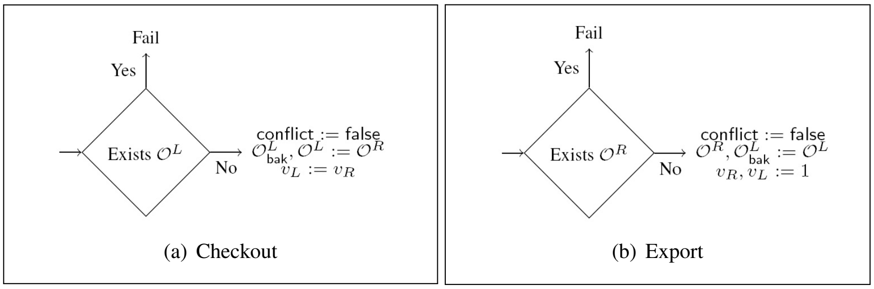


Figure 3.1: Semantics of the checkout and export operations in ContentCVS

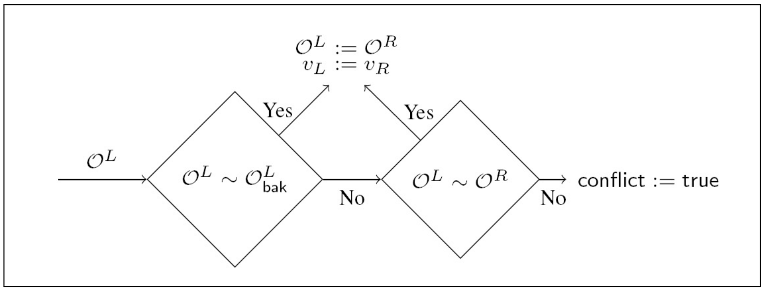

$\mathcal{O}^L := \mathcal{O}^R$
$v_L := v_R$
Yes
Yes
$\mathcal{O}^L$
$\mathcal{O}^L \sim \mathcal{O}^L_{\text{bak}}$
No
$\mathcal{O}^L \sim \mathcal{O}^R$
No
conflict := true

Figure 3.2: Semantics of the update operation in ContentCVS

copy $\mathcal{O}^L$ up-to-date by accessing the repository and incorporating the changes made by others. The update process starts by checking whether $\mathcal{O}^L$ has not changed since it was last updated (i.e., whether $\mathcal{O}^L_{\text{bak}} \backsim \mathcal{O}^L$); in case $\mathcal{O}^L$ has changed, it next checks whether the changes made by others are consistent with those made locally—that is whether $\mathcal{O}^L \backsim \mathcal{O}^R$. In either case (i.e., if either $\mathcal{O}^L_{\text{bak}} \backsim \mathcal{O}^L$ or $\mathcal{O}^L \backsim \mathcal{O}^R$) it is safe to replace $\mathcal{O}^L$ with the version $\mathcal{O}^R$ from the repository. Otherwise, a conflict is reported.

Finally, the *commit* operation (Figure 3.3), allows ontology developers to write (commit) their local changes to the repository. If $\mathcal{O}^L_{\text{bak}} \backsim \mathcal{O}^L$ then there are no meaningful local changes and hence no action is required. Otherwise, the commit process only succeeds if $\mathcal{O}^L$ is up-to-date ($v_L = v_R$) and not in conflict ($\text{conflict} = \text{false}$). In case of success, the commit operation involves replacing $\mathcal{O}^R$ with $\mathcal{O}^L$ and incrementing the version number.

Consider our running example and suppose that Anna has already committed her changes, so $\mathcal{O}^R = \mathcal{O}^2$; meanwhile, John modifies his local copy, so $\mathcal{O}^L = \mathcal{O}^1$. If John then tries to commit his changes, the operation will fail because $v_L \neq v_R$ (the local copy is not up-to-date); if he tries to update his local copy, the operation will fail because there have been local changes that are incompatible with those made by Anna, and $\mathcal{O}^L$ ends up in a conflicted state. Conflicts will need to be resolved before the commit operation can succeed.

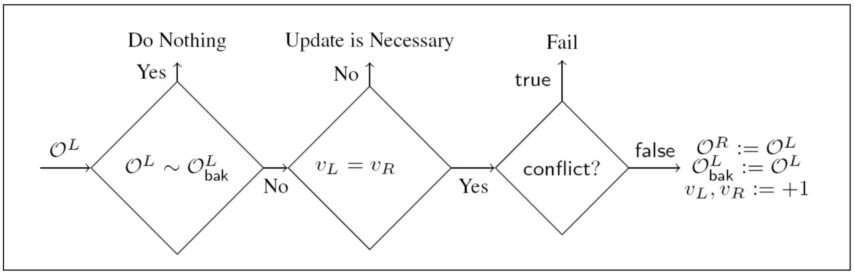

Do Nothing
Yes
Update is Necessary
No
Fail
true
$\mathcal{O}^L$
$\mathcal{O}^L \sim \mathcal{O}^L_{\mathsf{bak}}$
No
$v_L = v_R$
Yes
conflict?
false
$\mathcal{O}^R := \mathcal{O}^L$
$\mathcal{O}^L_{\mathsf{bak}} := \mathcal{O}^L$
$v_L, v_R := +1$

Figure 3.3: Semantics of the commit operation in ContentCVS

3.2.2. Change and conflict detection

As mentioned in Section 3.2.1, change and conflict detection amounts to checking whether two compared versions of a file are not "equivalent" according to a given notion of equivalence between versions of a file. A typical CVS treats the files in a software project as ordinary text files and hence checking equivalence amounts to determining whether the two versions are syntactically equal (i.e., they contain exactly the same characters in exactly the same order). This notion of equivalence is, however, too strict in the case of ontologies, since OWL files have very specific structure and semantics. For example, if two OWL files are identical except for the fact that two axioms appear in different order, the corresponding ontologies should be treated as "equivalent": an ontology contains a *set* of axioms and hence their order is irrelevant [CHM$^+$08]. Another possibility is to use the notion of logical equivalence. Logical equivalence is, however, too permissive: even if $\mathcal{O} \equiv \mathcal{O}'$—the strongest assumption from a semantic point of view—conflicts may still exist. This might result from the presence of incompatible annotations (statements that act as comments and do not carry logical meaning) [CHM$^+$08], or a mismatch in modelling styles; for example, $\mathcal{O}$ may be written in a simple language such as the OWL 2 EL profile [CHM$^+$08, MCH$^+$09] and contain $\alpha := (A \sqsubseteq B \sqcap C)$, while $\mathcal{O}'$ may contain $\beta := (\neg B \sqcup \neg C \sqsubseteq \neg A)$. Even though $\alpha \equiv \beta$, the explicit use of negation and disjunction means that $\mathcal{O}'$ is outside the EL profile.

Therefore, the notion of a conflict should be based on a notion of ontology equivalence "in-between" syntactical equality and logical equivalence. We propose to borrow the notion of *structural equivalence* from the OWL 2 specification [MPSP09]. Intuitively, this notion of equivalence is based solely on comparing structures by using the definition of the modelling constructs available in OWL 2; for example, several constructs are defined as sets of objects (e.g., ontologies are defined as sets of axioms, conjunction of concepts as a set of conjuncts, and so on); hence changes in the order in which these set elements appear in the ontology file should be seen as irrelevant (see Table 3.4).

In DL syntax, structural equivalence can be formalised as follows. For the sake of simplicity, our definition here comprises only the description logic $\mathcal{SROIQ}$ [HKS06], which provides the logical underpinning for OWL 2. This definition can be

Component Type	Syntax	Structural Characterization
Ontology	$\mathcal{O}$	*SET* of axioms
Concept Subsumption	$C \sqsubseteq D$	*LIST* of concept descriptions
Concept Equivalence	$C \equiv D$	*SET* of concept descriptions
Role Subsumption	$R \sqsubseteq S$	*LIST* of role descriptions
Role Equivalence	$R \equiv S$	*SET* of role descriptions
Concept Disjointness	$disjoint(C, D)$	*SET* of concept descriptions
Role Disjointness	$disjoint(R, S)$	*SET* of role descriptions
Inverse Roles	R^{-}	*SET* of role descriptions
Role Composition	$R \circ S$	*LIST* of role descriptions
Role Assertions	$a : C$	*LIST* of entity descriptions
Class Assertions	$R(a, b)$	*LIST* of entity descriptions
Universal Quantification	$\forall R.C$	*LIST* of entity descriptions
Existential Quantification	$\exists R.C$	*LIST* of entity descriptions
Number Restriction	$\geqslant n\, R.C$	*LIST* of entity descriptions
Union	$C \sqcup D$	*SET* of concept descriptions
Intersection	$C \sqcap D$	*SET* of concept descriptions
Nominals/Enumerations	$\{a1, \dots, a_n\}$	*SET* of individuals
Different Individuals	$Diff(a1, \dots, a_n)$	*SET* of individuals
Same Individuals	$Same(a1, \dots, a_n)$	*SET* of individuals
Unary Operators	i.e. $\neg C$, $Trans(R)$	Only one entity

Table 3.4: OWL 2 constructors and their structural characterization

easily extended to cover also datatypes and extra-logical statements, such as annotations. We refer the reader to [MPSP09] for a complete characterisation of structural equivalence.

Definition 3.2 (Structural Equivalence)**.** *The structural equivalence relation $\backsim$ over a set of concepts* $\mathbf{Con}$ *is defined by induction. First, $C \backsim C$ for each $C \in \mathbf{Con}$. For the induction step, we have:*

- *$C \backsim D$ implies $(\neg C) \backsim (\neg D)$;*
- *$C \backsim D$ implies $\lozenge R.C \backsim \lozenge R.D$ for $\lozenge \in \{\exists, \forall, \geq n, \leq n, = n\}$; and*
- *$C_1 \backsim C_2$ and $D_1 \backsim D_2$ implies $(C_1 \bowtie D_1) \backsim (C_2 \bowtie D_2) \backsim (D_2 \bowtie C_2)$, for $\bowtie \in \{\sqcap, \sqcup\}$.*

The relation $\backsim$ is extended to axioms over a set of concepts $\mathbf{Con}$ *and roles* $\mathbf{Rol}$ *as follows: $\alpha \backsim \alpha$ for each concept or role axiom α and, if $C_1 \backsim C_2$ and $D_1 \backsim D_2$, then $(C_1 \sqsubseteq D_1) \backsim (C_2 \sqsubseteq D_2)$, for $C_i, D_i \in \mathbf{Con}$. Finally $\backsim$ is extended to ontologies as follows: $\mathcal{O} \backsim \mathcal{O}'$ if, for every $\alpha \in \mathcal{O}$ (respectively $\alpha \in \mathcal{O}'$) there is a $\beta \in \mathcal{O}'$ (respectively $\beta \in \mathcal{O}$) such that $\alpha \backsim \beta$.*

For example, the axioms β_2 and β_2' in Table 3.2 are structurally equivalent because they only differ in the order of the elements in a disjunction. If $\mathcal{O} \backsim \mathcal{O}'$ we can safely assume that they are compatible and thus not in conflict.

The use of the notion of structural equivalence as a formal basis for detecting conflicts between ontology versions presents a number of compelling advantages:

- It is a notion "in-between" syntactical equality and logical equivalence: on the one hand irrelevant syntactic differences are ruled out as conflicts based solely on the structure of the OWL language; on the other hand, structurally equivalent ontologies are also guaranteed to be logically equivalent.
- It preserves species and profiles [MCH+09] of OWL and OWL 2 respectively; for example if $\mathcal{O}$ and $\mathcal{O}'$ are structurally equivalent and $\mathcal{O}$ is in OWL Lite (respectively in any of the profiles of OWL 2), then $\mathcal{O}'$ is also in OWL Lite (respectively in the same OWL 2 profile as $\mathcal{O}$).
- It takes into account extra-logical components of an ontology, such as annotations.
- It is an agreed-upon notion, obtained as a result of extensive discussions within the W3C OWL Working Group during the standardisation of OWL 2. Furthermore, it is not exclusive to OWL 2: it can be directly applied to OWL DL, and a similar notion could obviously be devised for most other ontology languages.
- It is supported by mainstream ontology development APIs, such as the OWL API.

Finally, the identification of the conflicting parts in $\mathcal{O}$ and $\mathcal{O}'$ using the notion of structural equivalence can be performed by computing their *structural difference*.

Definition 3.3 (Structural Difference)**.** *The structural difference between $\mathcal{O}_1$ and $\mathcal{O}_2$ is the set Λ_s of axioms $\alpha \in \mathcal{O}_i$ for which there is no $\beta \in \mathcal{O}_j$ s.t. $\alpha \backsim \beta$ with $i, j \in \{1, 2\}$ and $i \neq j$.*

ContentCVS reuses the functionality available in the OWL API for deciding structural equivalence and implements a straightforward algorithm for computing structural differences that follows directly from Definition 3.3.

3.2.2.1. Using a shared vocabulary

Some concepts are hard to describe and the selection of a proper label for them is not always a straightforward task. The problem of label (i.e. term) selection, to better describe the ontology concepts without ambiguity, is well known by the community and already a topic of discussion in ontology development projects like *Open-Galen* [ope10]. Furthermore, the occurrence of redundant entities (i.e. entities which are intended to have the same meaning but with different labels) is also an important

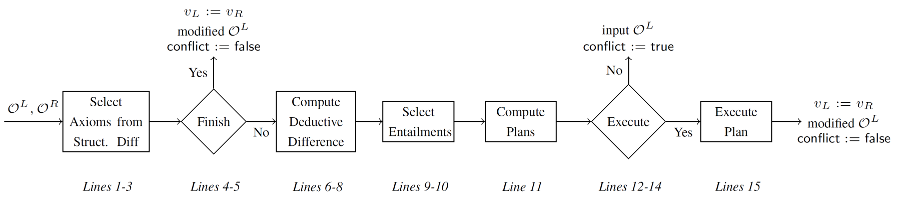


Figure 3.4: Conflict resolution method (informal).

problem that bothers the community when they develop an ontology collaboratively [SMS09].

ContentCVS assumes the existence of a thesaurus shared by all the ontology modellers. This shared thesaurus intends to serve as a bag of symbols (i.e., vocabulary provider) containing at least all concept labels which are being used within the ontology. Thus, when a modeller requires the creation of a new concept, she first checks if there exists an appropriated label within the thesaurus. Otherwise the modeller should create a new entry in the thesaurus.

The reuse of symbols from domain thesauri [JYJRBRS09b], as commented in Section 1.3.2, would reduce significantly the problem of merging two ontology versions since the alignment of the versions would not be necessary. In that way, ContentCVS can only focus on the analysis of the logic based consequences of the merging. In Chapter 4, we deal with the problem of integrating independently developed ontologies, which do not share a reference thesaurus, and thus a mapping is required in order to align their disparate vocabularies.

Future evolutions of ContentCVS will imply the integration with some initiatives for extending Protégé in order to link the ontology development process with available terminological resources such as UMLS-Meta[4], WordNet[5] or NCBO ontologies[6].

3.2.3. Conflict resolution method

Conflict resolution is the process of constructing a reconciled ontology from two ontology versions which are in conflict. Unlike text-based CVS systems the detection and resolution of errors can be performed semi-automatically by guiding the user with the candidate choices. Our approach allows users to resolve the identified conflicts at two different levels:

- *Structural*, where only the structure of the compared ontology versions is taken into account to build the reconciled ontology (see Section 3.2.3.1).

[4] UMLS Tab: `http://protegewiki.stanford.edu/index.php/UMLS_Tab`
[5] OntoLing: `http://protegewiki.stanford.edu/index.php/OntoLing`
[6] BioPortal Reference Plugin: `http://protegewiki.stanford.edu/index.php/BioPortal_Reference_Plugin`

```
Input: $\mathcal{O}^L, \mathcal{O}^R$: ontologies with $\mathcal{O}^L \not\approx \mathcal{O}^R$, $\mathsf{conflict} = \mathsf{true}$ and structural difference $\Lambda_s$
Output: $\mathcal{O}^L$: ontology; $\mathsf{conflict}$: boolean value;
 1: (✓) Select $\mathcal{S} \subseteq \Lambda_s$
 2: $\mathcal{O}^L_{\mathsf{temp}} := (\mathcal{O}^L \setminus \Lambda_s) \cup \mathcal{S}$
 3: (✓) if $\mathcal{O}^L_{\mathsf{temp}}$ is satisfactory return $\mathcal{O}^L := \mathcal{O}^L_{\mathsf{temp}}, \mathsf{conflict} := \mathsf{false}$
 4: (✓) Select approximation function $\mathsf{diff}_{\approx}$
 5: Compute $\mathsf{diff}_{\approx}(\mathcal{O}^L_{\mathsf{temp}}, \mathcal{O}^L)$, $\mathsf{diff}_{\approx}(\mathcal{O}^L_{\mathsf{temp}}, \mathcal{O}^R)$, $\mathsf{diff}_{\approx}(\mathcal{O}^L, \mathcal{O}^L_{\mathsf{temp}})$ and $\mathsf{diff}_{\approx}(\mathcal{O}^R, \mathcal{O}^L_{\mathsf{temp}})$
 6: (✓) Select $\Im^+ \subseteq \mathsf{diff}_{\approx}(\mathcal{O}^L_{\mathsf{temp}}, \mathcal{O}^L) \cup \mathsf{diff}_{\approx}(\mathcal{O}^L_{\mathsf{temp}}, \mathcal{O}^R)$
 7: (✓) Select $\Im^- \subseteq \mathsf{diff}_{\approx}(\mathcal{O}^L, \mathcal{O}^L_{\mathsf{temp}}) \cup \mathsf{diff}_{\approx}(\mathcal{O}^R, \mathcal{O}^L_{\mathsf{temp}})$
 8: Compute minimal plans $\mathbb{P}$ for $\mathcal{O}^L_{\mathsf{temp}}$ given $\Im^+, \Im^-, \mathcal{O}^+ := \Lambda_s \setminus \mathcal{S}$, and $\mathcal{O}^- := \mathcal{S}$
 9: (✓) if no satisfactory plan in $\mathbb{P}$, return $\mathcal{O}^L, \mathsf{conflict} := \mathsf{true}$
10: (✓) Select $\mathcal{P} = \langle \mathcal{P}^+, \mathcal{P}^- \rangle \in \mathbb{P}$
11: return $\mathcal{O}^L := (\mathcal{O}^L_{\mathsf{temp}} \cup \mathcal{P}^+) \setminus \mathcal{P}^-, \mathsf{conflict} := \mathsf{false}$
```

Table 3.5: Conflict resolution method.

- *Structural and semantic*, where both the structure and the logical consequences of the compared ontology versions as well as of the reconciled ontology are taken into consideration (see Sections 3.2.3.1—3.2.3.6).

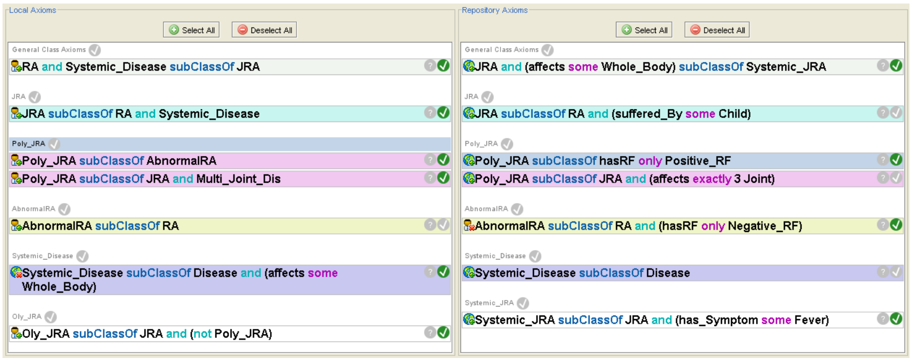
In the former case, the overhead involved in using a reasoner and examining its output is avoided; however, the reconciled ontology may contain errors (e.g., undesired logical consequences), which would remain undetected.

In the latter case, errors in the reconciliation process can be detected, with the assistance of a reasoner, by computing the logical consequences of the reconciled ontology and comparing them to those of the relevant ontology versions. Errors in the reconciled ontology could manifest themselves, however, not only as unsatisfiable concepts or unintended (or missing) subsumptions between atomic concepts, but also as unintended (or missing) entailments involving complex concepts. We propose to use the notion of *deductive difference* for error detection (see Section 3.2.3.2), which ensures that errors associated with complex entailments are also detected. However, considering complex entailments obviously comes at a price, both in terms of computational cost and of complication of the GUI. Thus, a CVS client should allow users to customise the types of relevant entailments for error detection and guide them in the selection process (see Section 3.2.3.2). Finally, error repair is a complicated process for which tool support should be provided. Our approach involves a number of techniques to achieve this goal (see Sections 3.2.3.3 and 3.2.3.6).

Our approach is summarised in Figure 3.4 and Table 3.5. The steps marked with a tickmark (✓) are those that require human intervention. We next describe in detail each of the steps in Table 3.5.

3.2.3.1. Selection of axioms using structural difference

Conflict resolution in text files is usually performed by first identifying and displaying the conflicting sections in the two files (e.g., a line, or a paragraph) and then manually selecting the desired content. Analogously, our proposal for structural con-



Local Axioms

General Class Axioms
RA and Systemic_Disease subClassOf JRA

JRA
JRA subClassOf RA and Systemic_Disease

Poly_JRA
Poly_JRA subClassOf AbnormalRA
Poly_JRA subClassOf JRA and Multi_Joint_Dis

AbnormalRA
AbnormalRA subClassOf RA

Systemic_Disease
Systemic_Disease subClassOf Disease and (affects some Whole_Body)

Oly_JRA
Oly_JRA subClassOf JRA and (not Poly_JRA)

Repository Axioms

General Class Axioms
JRA and (affects some Whole_Body) subClassOf Systemic_JRA

JRA
JRA subClassOf RA and (suffered_By some Child)

Poly_JRA
Poly_JRA subClassOf hasRF only Positive_RF
Poly_JRA subClassOf JRA and (affects exactly 3 Joint)

AbnormalRA
AbnormalRA subClassOf RA and (hasRF only Negative_RF)

Systemic_Disease
Systemic_Disease subClassOf Disease

Systemic_JRA
Systemic_JRA subClassOf JRA and (has_Symptom some Fever)

Figure 3.5: GUI for structural differences in ContentCVS

flict resolution involves computing and displaying the structural difference Λ_s (i.e., those axioms for which a structurally equivalent axiom does not occur in both ontologies) and then manually selecting which of these axioms should be included in a (provisional) version $\mathcal{O}^L_{\mathsf{temp}}$ of $\mathcal{O}^L$ (Step 1). The ontology $\mathcal{O}^L_{\mathsf{temp}}$ is obtained from the non-conflicting part of $\mathcal{O}^L$ plus the selected axioms $\mathcal{S}$ from the conflicting part (Step 2).

After constructing $\mathcal{O}^L_{\mathsf{temp}}$, the user may declare the conflict resolved (Step 3), in which case conflict resolution remains a purely syntactic process, as in the case of text files. Otherwise, semantic consequences of their choices and make sure that $\mathcal{O}^L_{\mathsf{temp}}$ meets their requirements (typically, includes as much information as possible without leading to inconsistencies or other undesired entailments).

ContentCVS implements a simple GUI to facilitate the selection of axioms from the structural difference, which is shown in Figure 3.5 for our running example. The left-hand-side (respectively the right-hand-side) of the figure shows the axioms in $\Lambda_s \cap \mathcal{O}^L$ (respectively in $\Lambda_s \cap \mathcal{O}^R$). Unlike PromptDiff [NKKM04], ContentCVS exploits the definition of structural difference now provided in the OWL 2 specification [CHM+08].

To facilitate the comparison, axioms are sorted and aligned according to the entities they define. Axioms not aligned with others are shown last in a distinguished position. The selected axioms are indicated in the GUI using a highlighted tickmark (✓). Furthermore, ContentCVS provides additional functionality for determining the origin of each axiom in the structural difference. In particular, ContentCVS, indicates whether an axiom appears in the difference as a result of an addition or a deletion by comparing $\mathcal{O}^L$ and $\mathcal{O}^R$ to the local backup copy $\mathcal{O}^L_{\mathsf{bak}}$ of the latest local version that was "written" to the repository. For example, the axiom ($\mathsf{Poly\_JRA} \sqsubseteq \mathsf{AbnormalRA}$) on the left-hand-side of Figure 3.5 was added to $\mathcal{O}^L_{\mathsf{bak}}$ in the local ontology (indicated by an icon representing a user with a '+'), whereas the axiom ($\mathsf{Systemic\_Disease} \sqsubseteq \mathsf{Disease} \sqcap \exists \mathsf{affects}.\mathsf{WholeBody}$) was deleted from $\mathcal{O}^L_{\mathsf{bak}}$ in the repository (indicated by an icon representing the Globe with a cross '$\times$').

3.2.3.2. Deductive differences

In contrast to text files, the selected parts from $\mathcal{O}^L$ and $\mathcal{O}^R$ can interact in unexpected ways, which may lead to errors that should be repaired. To help users detect such errors, we propose to compare the entailments that hold in $\mathcal{O}^L_{\mathsf{temp}}$ with those that hold in $\mathcal{O}^L$ and $\mathcal{O}^R$ by using the notion of *deductive difference* [KWW08, KWZ08].

Definition 3.4 (Deductive Difference)**.** *The deductive difference* $\mathsf{diff}(\mathcal{O}, \mathcal{O}')$ *between* $\mathcal{O}$ *and* $\mathcal{O}'$ *expressed in a DL* $\mathcal{DL}$ *is the set of* $\mathcal{DL}$*-axioms* α *s.t.* $\mathcal{O} \not\models \alpha$ *and* $\mathcal{O}' \models \alpha$*.*

Intuitively, this difference is the set of *all* (possibly complex) entailments that hold in one ontology but not in the other.

Example 3.1. *In our running example, for the selection in Figure 3.5, there are entailments that* (i) *hold in* $\mathcal{O}^L_{\mathsf{temp}}$ *and not in* $\mathcal{O}^L$*, such as* $\tau_1 := (\mathsf{SystemicJRA} \sqsubseteq \exists \mathsf{has\_Symptom.Fever})$*;* (ii) *hold in* $\mathcal{O}^L_{\mathsf{temp}}$ *but not in either* $\mathcal{O}^L$ *or* $\mathcal{O}^R$*, such as* σ_5 *from Table 3.3;* (iii) *hold in* $\mathcal{O}^L$ *but not in* $\mathcal{O}^L_{\mathsf{temp}}$*, such as* $\tau_2 := (\mathsf{RA} \sqcap \exists \mathsf{affects.WholeBody} \sqsubseteq \mathsf{JRA})$*; and finally* (iv) *hold in* $\mathcal{O}^R$ *but not in* $\mathcal{O}^L_{\mathsf{temp}}$*, such as* σ_2 *in Table 3.3. Cases (i) and (ii) represents new entailments in* $\mathcal{O}^L_{\mathsf{temp}}$*, wheras (iii) and (iv) lost entailments w.r.t.* $\mathcal{O}^L$ *or* $\mathcal{O}^R$*.*

Therefore, we argue that the relevant deductive differences between $\mathcal{O}^L_{\mathsf{temp}}$, $\mathcal{O}^L$ and $\mathcal{O}^R$ capture all potential errors that may have been introduced in the reconciliation process. However, the notion of deductive difference has several drawbacks in practice. First, checking whether $\mathsf{diff}(\mathcal{O}, \mathcal{O}') = \emptyset$ is undecidable in expressive DLs, such as $\mathcal{SROIQ}$ (OWL 2) and $\mathcal{SHOIQ}$ (OWL DL) [KWW08]. Second, the number of entailments in the difference can be huge (even infinite), and so likely to overwhelm users. These practical drawbacks motivate the need for *approximations*— subsets of the deductive difference (see Step 4 in Table 3.5).

Definition 3.5 (Approximation of Deductive Difference)**.** *A function* $\mathsf{diff}_{\approx}(\mathcal{O}, \mathcal{O}')$ *is an approximation for* $\mathsf{diff}(\mathcal{O}, \mathcal{O}')$ *if for each pair* $\mathcal{O}, \mathcal{O}'$ *of ontologies,* $\mathsf{diff}_{\approx}(\mathcal{O}, \mathcal{O}') \subseteq \mathsf{diff}(\mathcal{O}, \mathcal{O}')$.

A useful approximation should be easy to compute, yet still provide meaningful information to the user. One possibility is to define an approximation by considering only entailments of a certain form. Our tool ContentCVS allows users to customise approximations by selecting among the following kinds of entailments, where A, B are atomic concepts (including $\top$, $\bot$) and R, S atomic roles or inverses of atomic roles: (i) $A \sqsubseteq B$, (ii) $A \sqsubseteq \neg B$, (iii) $A \sqsubseteq \exists R.B$, (iv) $A \sqsubseteq \forall R.B$, and (v) $R \sqsubseteq S$. The smallest implemented approximation considers only axioms of the form (i), which amounts to comparing the classification hierarchy of both ontologies, while the largest considers all types (i)—(v). Clearly, the larger the class of entailments presented to the user, the more errors could be detected. The corresponding differences, however, are harder to compute, harder to present to the user, and may be harder for the user to understand them.

Example 3.2. *In our running example,* ContentCVS *would cover the entailments τ_1 in Example 3.1 (using an approximation of the form* (iii)*), and σ_2 and σ_5 from Table 3.3 (using entailments of the form* (i)*). On the other hand, to cover entailments such as τ_2 (Example 3.1) an approximation based on a more expressive language would be needed.*

Although these approximations can all be algorithmically computed, only the entailments of the form (i) and (v) are typically provided by reasoners as standard outputs of classification. Computing deductive differences based on entailments (ii)-(iv) is potentially very expensive since it may involve performing a large number of additional entailment tests. To reduce the number of tests, ContentCVS uses the notion of a *locality-based module* [CHKS07, GHKS08] (see Section 2.2).

In order to check for all entailments of the form (ii)-(iv), ContentCVS first extracts the locality-based module for A and looks for potential entailments only within the module. For example, in the case of (iii) ContentCVS would only test entailments where both R and B are in the vocabulary of the module $\mathcal{O}_A$, which significantly reduces the search space. Furthermore, the actual relevant entailments can be checked w.r.t. the (small) module, and not with respect to the (potentially large) original ontology. Our experiments in Section 3.3 suggest that the use of locality-based modules makes the computation of approximations to the deductive difference based on all types (i)-(v) of entailments computationally feasible.

The OWLDiff tool[7] implements the deductive difference between OWL 2 EL ontologies [KWW08]. The differences are shown in a GUI as a set of highlighted entities, which are the ones whose meaning differs between the two ontologies. However, unlike ContentCVS , the tool does not provide means for defining tractable approximations or for explaining these differences to the user.

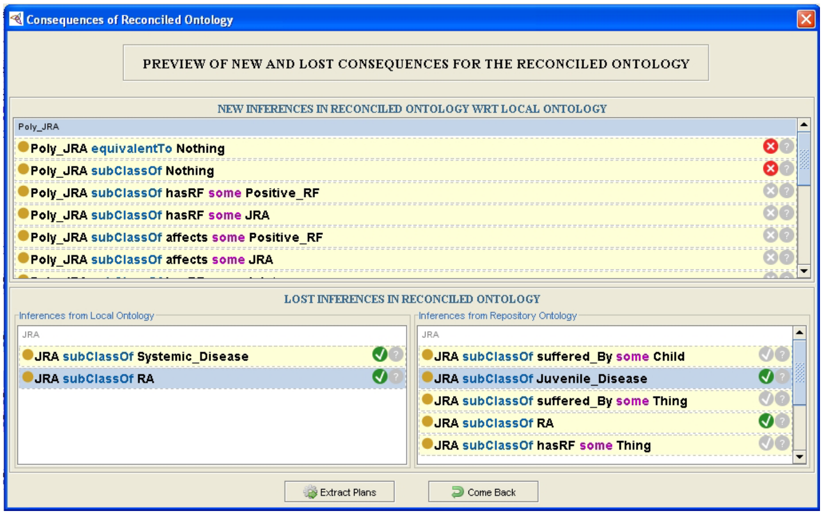
3.2.3.3. Selection of entailments

While some entailments in the computed differences are intended, others reveal errors in $\mathcal{O}^L_{\mathsf{temp}}$, as illustrated by the following example.

Example 3.3. *In our example (Table 3.3), the entailment* $\mathsf{JRA} \sqsubseteq \mathsf{Juvenile\_Disease}$ *(σ_2) is intended. In contrast,* $\mathsf{Poly\_JRA} \sqsubseteq \bot$ *(σ_5) reveals an inconsistency in $\mathcal{O}^L_{\mathsf{temp}}$, and hence an obvious error.*

Steps 6 and 7, from Table 3.5, thus involve selecting entailments that: *(i)* are intended and should follow from $\mathcal{O}^L_{\mathsf{temp}}$ (written $\Im^+$ in Table 3.5), and *(ii)* are unintended and should not follow from $\mathcal{O}^L_{\mathsf{temp}}$ (written $\Im^-$). Figure 3.6 shows the ContentCVS GUI for selecting $\Im^-$ (top part) and $\Im^+$ (bottom part) entailments.

[7] OWLDiff: `http://krizik.felk.cvut.cz/km/owldiff/index.html`, `http://sourceforge.net/projects/owldiff`



PREVIEW OF NEW AND LOST CONSEQUENCES FOR THE RECONCILED ONTOLOGY

NEW INFERENCES IN RECONCILED ONTOLOGY WRT LOCAL ONTOLOGY

Poly_JRA

- Poly_JRA equivalentTo Nothing
- Poly_JRA subClassOf Nothing
- Poly_JRA subClassOf hasRF some Positive_RF
- Poly_JRA subClassOf hasRF some JRA
- Poly_JRA subClassOf affects some Positive_RF
- Poly_JRA subClassOf affects some JRA

LOST INFERENCES IN RECONCILED ONTOLOGY

Inferences from Local Ontology

JRA

- JRA subClassOf Systemic_Disease
- JRA subClassOf RA

Inferences from Repository Ontology

JRA

- JRA subClassOf suffered_By some Child
- JRA subClassOf Juvenile_Disease
- JRA subClassOf suffered_By some Thing
- JRA subClassOf RA
- JRA subClassOf hasRF some Thing

Figure 3.6: Displaying new and lost entailments for $\mathcal{O}^L_{\text{temp}}$ in ContentCVS

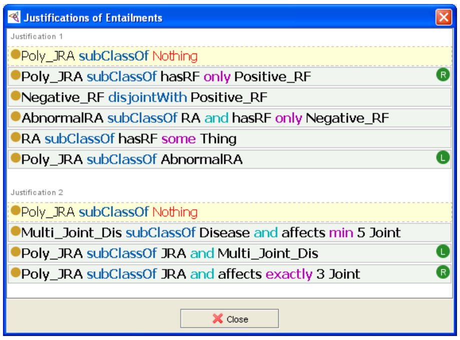
3.2.3.4. Explaining the entailments

The development of techniques to help users understand the relevant deductive differences and subsequently select the sets of intended and unintended entailments is especially challenging. Thus, a tool should explain, on the one hand, why the new entailments that hold in $\mathcal{O}^L_{\text{temp}}$ do not hold in $\mathcal{O}^L$ and $\mathcal{O}^R$ alone and, on the other hand, why the lost entailments that hold in $\mathcal{O}^L$ and $\mathcal{O}^R$ do not hold in $\mathcal{O}^L_{\text{temp}}$.

ContentCVS has reused and adapted the notion of justification from Definition 3.1. Note that, we denote by $\text{Just}(\alpha, \mathcal{O})$ the set of all justifications for α in $\mathcal{O}$.

Figure 3.7 shows the ContentCVS GUI to represent all justifications for the entailment Poly_JRA $\sqsubseteq \bot$. We have extended the operation provided by Protégé 4 [HPS08a], which gives explanations for arbitrary entailments, in order to graphically indicate which axioms in the justifications were selected in Step 2 of Table 3.5 from $\mathcal{O}^L$ and $\mathcal{O}^R$, marking them with 'L' and 'R' respectively. The unmarked axioms occur in both ontologies. Additionally, axioms shared by all justifications are marked with 'J'.

Computing all justifications is expensive, so ContentCVS uses the optimisation from [SQJH08], which have proved effective in practice. Our algorithm for extracting all the justifications for an entailment of the form $A \sqsubseteq C$, for A an atomic concept, is based on extracting first the locality-based module for A in the ontology and then compute the justifications w.r.t. the module instead of w.r.t. the whole ontology.



Justifications of Entailments

Justification 1

Poly_JRA subClassOf Nothing
Poly_JRA subClassOf hasRF only Positive_RF R
Negative_RF disjointWith Positive_RF
AbnormalRA subClassOf RA and hasRF only Negative_RF
RA subClassOf hasRF some Thing
Poly_JRA subClassOf AbnormalRA L

Justification 2

Poly_JRA subClassOf Nothing
Multi_Joint_Dis subClassOf Disease and affects min 5 Joint
Poly_JRA subClassOf JRA and Multi_Joint_Dis L
Poly_JRA subClassOf JRA and affects exactly 3 Joint R

Figure 3.7: Justifications for entailment $\mathsf{Poly\_JRA} \sqsubseteq \bot$

3.2.3.5. Ordering the entailments

Even with the explanations provided, the potentially large number of relevant entailments may overwhelm users. These entailments should therefore be presented in a way that makes them easier to understand and manage. To this end, ContentCVS extends known ontology debugging techniques by identifying dependencies between entailments. As an illustration, consider $\sigma_2 := (\mathsf{JRA} \sqsubseteq \mathsf{Juvenile\_Disease})$ from Table 3.3 and $\tau_4 := (\mathsf{SystemicJRA} \sqsubseteq \mathsf{Juvenile\_Disease})$ which hold in $\mathcal{O}^L_{\mathsf{temp}}$ but not in $\mathcal{O}^L$. The entailment τ_4 *depends* on σ_2 since whenever σ_2 is invalidated by removing axioms from $\mathcal{O}^L_{\mathsf{temp}}$, then τ_4 is also invalidated. Similarly, the entailment $\tau_5 := (\mathsf{Oly\_JRA} \sqsubseteq \exists \mathsf{affects.WholeBody})$ depends on the entailment $\tau_6 := (\mathsf{JRA} \sqsubseteq \exists \mathsf{affects.WholeBody})$: adding any set of axioms from $\mathcal{O}^L$ or $\mathcal{O}^R$ that causes τ_6 to hold in $\mathcal{O}^L_{\mathsf{temp}}$ would also cause τ_5 to hold. We formalise these intuitions in our setting as follows:

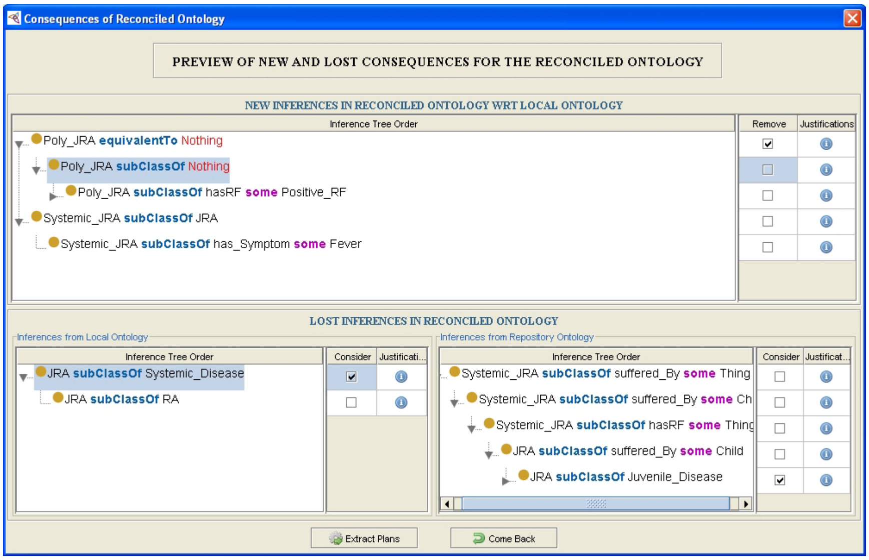
Definition 3.6 (Orderings)**.** *Let $\mathcal{O} \models \alpha, \beta$. The orderings $\rhd^+$ and $\rhd^-$ over $\mathcal{O}$ are defined as follows:*

1. *$\alpha \rhd^+ \beta$ iff for each $\mathcal{J}_\alpha \in \mathsf{Just}(\alpha, \mathcal{O})$, there is a $\mathcal{J}_\beta \in \mathsf{Just}(\beta, \mathcal{O})$ s.t. $\mathcal{J}_\beta \subseteq \mathcal{J}_\alpha$.*
2. *$\alpha \rhd^- \beta$ iff for each $\mathcal{J}_\beta \in \mathsf{Just}(\beta, \mathcal{O})$ there is a $\mathcal{J}_\alpha \in \mathsf{Just}(\alpha, \mathcal{O})$ s.t. $\mathcal{J}_\alpha \subseteq \mathcal{J}_\beta$.*

The orderings $\rhd^+$ and $\rhd^-$ are consistent with our intuitions as shown in the following proposition, which follows directly from Definitions 3.1 (Justification) and 3.6 (Orderings):

Proposition 3.1. *Let $\mathcal{O} \models \alpha, \beta$, $\mathcal{O}' \subset \mathcal{O}$. 1) If $\alpha \rhd^- \beta$ then $\mathcal{O}' \not\models \alpha$ implies $\mathcal{O}' \not\models \beta$, and 2) if $\alpha \rhd^+ \beta$ $\mathcal{O}' \models \alpha$ implies $\mathcal{O}' \models \beta$.*

Proof. First we prove Condition 1. Let $\alpha \rhd^- \beta$ and $\mathcal{O}' \not\models \alpha$. Since $\mathcal{O} \models \alpha$, for each $\mathcal{J}_\alpha \in \mathsf{Just}(\alpha, \mathcal{O})$ there is an axiom $\gamma \in \mathcal{J}_\alpha$ s.t. $\gamma \notin \mathcal{O}'$. Otherwise, $\mathcal{O}'$ would imply α. Let $\mathcal{J}_\beta \in \mathsf{Just}(\beta, \mathcal{O})$. By definition of $\rhd^-$, there must be a $\mathcal{J}_\alpha \in \mathsf{Just}(\alpha, \mathcal{O})$ s.t. $\mathcal{J}_\alpha \subseteq \mathcal{J}_\beta$. Since $\mathcal{O}'$ does not include one axiom from each $\mathcal{J}_\alpha$, then it also does not


Consequences of Reconciled Ontology

PREVIEW OF NEW AND LOST CONSEQUENCES FOR THE RECONCILED ONTOLOGY

NEW INFERENCES IN RECONCILED ONTOLOGY WRT LOCAL ONTOLOGY

Inference Tree Order	Remove	Justifications
Poly_JRA **equivalentTo** Nothing	✓	●
Poly_JRA **subClassOf** Nothing	☐	●
Poly_JRA **subClassOf** hasRF **some** Positive_RF	☐	●
Systemic_JRA **subClassOf** JRA	☐	●
Systemic_JRA **subClassOf** has_Symptom **some** Fever	☐	●

LOST INFERENCES IN RECONCILED ONTOLOGY

Inferences from Local Ontology

Inference Tree Order	Consider	Justificati...
JRA **subClassOf** Systemic_Disease	✓	●
JRA **subClassOf** RA	☐	●

Inferences from Repository Ontology

Inference Tree Order	Consider	Justificat...
Systemic_JRA **subClassOf** suffered_By **some** Thing	☐	●
Systemic_JRA **subClassOf** suffered_By **some** Ch	☐	●
Systemic_JRA **subClassOf** hasRF **some** Thing	☐	●
JRA **subClassOf** suffered_By **some** Child	☐	●
JRA **subClassOf** Juvenile_Disease	✓	●

Extract Plans   Come Back

Figure 3.8: Orderings $\rhd^+$ and $\rhd^-$ in ContentCVS

include one axiom from each $\mathcal{J}_\beta$ and therefore $\mathcal{O}' \not\models \beta$, as required. We now prove Condition 2. Let $\alpha \rhd^+ \beta$ and $\mathcal{O}' \models \alpha$. Since $\mathcal{O} \models \alpha$, there exists a justification $\mathcal{J}_\alpha \in \mathsf{Just}(\alpha, \mathcal{O})$ s.t. $\mathcal{J}_\alpha \subseteq \mathcal{O}'$. By definition of $\rhd^+$ there must exist a $\mathcal{J}_\beta \in \mathsf{Just}(\beta, \mathcal{O})$ s.t. $\mathcal{J}_\beta \subseteq \mathcal{J}_\alpha$. Therefore $\mathcal{J}_\alpha \subseteq \mathcal{O}'$ and $\mathcal{O}' \models \alpha$, as required. $\square$

Figure 3.8 shows the ContentCVS GUI with the new and lost entailments, as in Figure 3.6, but represented within a dependency tree according to $\rhd^-$ (top part) and $\rhd^+$ (bottom part) orderings. Entailments can also be added to $\Im^+$ or $\Im^-$, or their justification can be shown.

3.2.3.6. Repair plans generation

Changing the set of entailments can only be achieved by modifying the ontology itself. In general, there may be zero or more possible choices of sets of axioms to add and/or remove in order to satisfy a given set of requirements. We call each of these possible choices a *repair plan* (or *plan*, for short).

Definition 3.7 (Repair Plan)**.** *Let $\mathcal{O}$, $\Im^+$, $\Im^-$, $\mathcal{O}^+$ and $\mathcal{O}^-$ be finite sets of axioms s.t. $\mathcal{O}^- \subseteq \mathcal{O}$, $\mathcal{O}^+ \cap \mathcal{O} = \emptyset$, $\mathcal{O} \models \Im^-$, $\mathcal{O} \cup \mathcal{O}^+ \models \Im^+$, and $\mathcal{O} \not\models \alpha$ for each $\alpha \in \Im^+$. A repair plan for $\mathcal{O}$ given $\mathcal{O}^+$, $\mathcal{O}^-$ $\Im^+$ and $\Im^-$ is a pair $\mathcal{P} = \langle \mathcal{P}^+, \mathcal{P}^- \rangle$ such that $\mathcal{P}^+ \subseteq \mathcal{O}^+$, $\mathcal{P}^- \subseteq \mathcal{O}^-$ and the following conditions hold:*

1. $(\mathcal{O} \cup \mathcal{P}^+) \setminus \mathcal{P}^- \models \alpha$ *for each* $\alpha \in \Im^+$*, and*

2. $(\mathcal{O} \cup \mathcal{P}^+) \setminus \mathcal{P}^- \not\models \beta$ *for each* $\beta \in \Im^-$.

$\mathcal{P}$ is minimal if there is no $\mathcal{P}_1$ s.t. $\mathcal{P}_1^+ \subset \mathcal{P}^+$ and $\mathcal{P}_1^- \subset \mathcal{P}^-$.

Definition 3.7 extends the notion of a plan proposed in the ontology repair literature (e.g., see [KPSG06]). In particular, the goal of a plan in [KPSG06] is always to remove a set of axioms so that certain entailments do not hold anymore; hence, a plan is always guaranteed to exist. In contrast, a plan as in Definition 3.7 also involves adding axioms so that certain entailments hold; therefore, possibly conflicting sets of constraints need to be satisfied. Furthermore, existing repair techniques (e.g. [KPSG06]) are restricted to obvious inconsistencies (i.e., unsatisfiable concepts), whereas our techniques apply to errors caused by arbitrary entailments.

Step 8 from Table 3.5 involves the computation of all minimal plans (denoted $\mathbb{P}$). In our case, the ontology $\mathcal{O}$ to be repaired is $\mathcal{O}^L_{\mathsf{temp}}$ from Step 3. The intended and unintended entailments ($\Im^+$ and $\Im^-$) are those selected in Steps 6 and 7. We assume that a plan can add any subset of the axioms in Λ_s, which were not originally selected in Step 2 (i.e., $\mathcal{O}^+ = \Lambda_s \setminus \mathcal{S}$), and it can remove any subset of the selected axioms (i.e., $\mathcal{O}^- = \mathcal{S}$). Hence, we assume that a plan should not remove axioms outside Λ_s, since they are common to both versions of the ontology.

Example 3.4. *Given $\Im^+ = \{\sigma_2\}$ and $\Im^- = \{\sigma_5\}$ from Example 3.3, four minimal plans can be identified: $\mathcal{P}_1 = \langle\{\delta_1\}, \{\delta_3, \delta_4\}\rangle$; $\mathcal{P}_2 = \langle\{\delta_1\}, \{\delta_3, \gamma_4\}\rangle$; $\mathcal{P}_3 = \langle\{\delta_1\}, \{\gamma_3, \gamma_4\}\rangle$; $\mathcal{P}_4 = \langle\{\delta_1\}, \{\gamma_3, \delta_4\}\rangle$.*

In Step 9 users can select from $\mathbb{P}$ a plan to be applied. If no plan matches their intentions, the conflict resolution process ends as it started; that is, by returning the old version of $\mathcal{O}^L$ in a conflicting state (Step 9). In contrast, if a plan $\mathcal{P}$ is selected, then $\mathcal{P}$ is applied by returning the ontology $(\mathcal{O}^L_{\mathsf{temp}} \cup \mathcal{P}^+) \setminus \mathcal{P}^-$ in a non-conflicting state (Steps 10–11), which is then ready to be committed.

Definition 3.7 suggests a simple procedure for computing all plans: for each possible $\mathcal{P}^+ \subseteq \mathcal{O}^+$ and $\mathcal{P}^- \subseteq \mathcal{O}^-$, use a reasoner to check if $\langle\mathcal{P}^+, \mathcal{P}^-\rangle$ satisfies Conditions 1 and 2 from Definition 3.7. ContentCVS , however, implements an optimised version of Algorithm 1, which uses inverted indexes, reduces the number of combinations and avoids potentially expensive entailment checks by reusing the justifications already computed when obtaining the dependency relations ($\rhd^+$ and $\rhd^-$) from Definition 3.6. The correctness of the algorithm is a direct consequence of the following: *(i)* in order for an entailment $\alpha \in \Im^+$ to hold after the execution of a plan $\langle\mathcal{P}^+, \mathcal{P}^-\rangle$, $(\mathcal{O} \cup \mathcal{P}^+) \setminus \mathcal{P}^-$ must contain at least one justification for α in $\mathcal{O} \cup \mathcal{O}^+$ (Lines 6–10); *(ii)* in order for an entailment $\beta \in \Im^-$ not to hold after the execution of a plan $\langle\mathcal{P}^+, \mathcal{P}^-\rangle$, it is sufficient to show that no justification for β in $\mathcal{O} \cup \mathcal{O}^+$ is contained in $(\mathcal{O} \cup \mathcal{P}^+) \setminus \mathcal{P}^-$ (Lines 11–15). The set of all minimal plans can be straightforwardly computed once all the plans have been obtained.

Proposition 3.2. *Algorithm 1 returns all plans for the given input.*

Proof. For any $\mathcal{P} \in \mathbf{P}$, with $\mathcal{P} = \langle\mathcal{P}^+, \mathcal{P}^-\rangle$, we show that $\mathcal{P}$ is a plan for $\mathcal{O}$ given $\mathcal{O}^+$, $\mathcal{O}^-$ $\Im^+$ and $\Im^-$, as per Definition 3.7. The facts that $\mathcal{P}^+ \subseteq \mathcal{O}^+$ and $\mathcal{P}^- \subseteq \mathcal{O}^-$ are clear from Line 2. For each $\alpha \in \Im^+$, Line 7 implies that there is a

Algorithm 1 Computing all plans using justifications

```
Procedure all_Plans($\mathcal{O}, \mathcal{O}^+, \mathcal{O}^-, \Im^+, \Im^-$)
Input: $\mathcal{O}$, $\mathcal{O}^+$, $\mathcal{O}^-$, $\Im^+$, $\Im^-$ as in Definition 3.7, $\mathsf{Just}(\gamma, \mathcal{O} \cup \mathcal{O}^+)$ for each $\gamma \in \Im^+ \cup \Im^-$
Output: $\mathbf{P}$: set of all plans
 1: $\mathbf{P} := \emptyset$
 2: for each $\mathcal{P}^+ \subseteq \mathcal{O}^+$ and each $\mathcal{P}^- \subseteq \mathcal{O}^-$ do
 3:     validPlan := true
 4:     for each $\alpha \in \Im^+$ do
 5:         foundJust := false
 6:         for each $\mathcal{J}_\alpha \in \mathsf{Just}(\alpha, \mathcal{O} \cup \mathcal{O}^+)$ do
 7:             if $\mathcal{J}_\alpha \subseteq (\mathcal{O} \cup \mathcal{P}^+) \setminus \mathcal{P}^-$ foundJust := true
 8:         end for
 9:         if foundJust = false then validPlan := false
10:     end for
11:     for each $\beta \in \Im^-$ do
12:         for each $\mathcal{J}_\beta \in \mathsf{Just}(\beta, \mathcal{O} \cup \mathcal{O}^+)$ do
13:             if $\mathcal{J}_\beta \subseteq (\mathcal{O} \cup \mathcal{P}^+) \setminus \mathcal{P}^-$ then validPlan := false
14:         end for
15:     end for
16:     if validPlan = true then $\mathbf{P} := \mathbf{P} \cup \{\langle \mathcal{P}^+, \mathcal{P}^- \rangle\}$
17: end for
18: return $\mathbf{P}$
```

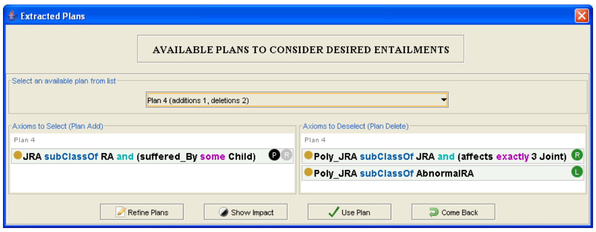
$\mathcal{J}_\alpha \in \mathsf{Just}(\alpha, \mathcal{O} \cup \mathcal{O}^+)$ s.t. $\mathcal{J}_\alpha \in (\mathcal{O} \cup \mathcal{P}^+) \setminus \mathcal{P}^-$, and the definition of justification immediately implies that $\mathcal{P}$ satisfies Condition 1 from Definition 3.7. For each $\beta \in \Im^-$, Lines 14 and 15 imply that $\mathcal{P}^-$ removes at least one axiom from each justification for β in $\mathcal{O} \cup \mathcal{P}^+$; hence $\mathcal{P}$ also satisfies Condition 2.

Let us assume that there exists a plan $\langle \mathcal{P}^+, \mathcal{P}^- \rangle \notin \mathbf{P}$ for $\mathcal{O}$ given $\mathcal{O}^+$, $\mathcal{O}^-$ $\Im^+$ and $\Im^-$. This means that either there is an $\alpha \in \Im^+$ s.t. no justification for α in $\mathcal{O} \cup \mathcal{O}^+$ is contained in $(\mathcal{O} \cup \mathcal{P}^+) \setminus \mathcal{P}^-$, or there is a $\beta \in \Im^-$ and a justification $\mathcal{J}_\beta$ for β in $\mathcal{O} \cup \mathcal{O}^+$ s.t. $\mathcal{J}_\beta \subseteq (\mathcal{O} \cup \mathcal{P}^+) \setminus \mathcal{P}^-$. In the first case, there can be no justification for α in $(\mathcal{O} \cup \mathcal{P}^+) \setminus \mathcal{P}^-$, because $(\mathcal{O} \cup \mathcal{P}^+) \setminus \mathcal{P}^- \subseteq \mathcal{O} \cup \mathcal{O}^+$, so $(\mathcal{O} \cup \mathcal{P}^+) \setminus \mathcal{P}^- \not\models \alpha$, contradicting our assumption that $\langle \mathcal{P}^+, \mathcal{P}^- \rangle$ is a plan. In the second case, $\mathcal{J}_\beta \in (\mathcal{O} \cup \mathcal{P}^+) \setminus \mathcal{P}^-$, which implies $(\mathcal{O} \cup \mathcal{P}^+) \setminus \mathcal{P}^- \models \beta$, also contradicting our assumption. $\square$

3.2.3.7. Plan selection in ContentCVS

Figure 3.9 illustrates the ContentCVS GUI for visualising plans. It shows one of the minimal plans for our running example. ContentCVS identifies the axioms in the structural difference of $\mathcal{O}^L$ (marked with a 'L') or $\mathcal{O}^R$ (marked with a 'R'), and also detects axioms that are shared by all minimal plans (marked with a 'P').

The user can execute any of the obtained plans using the option "Use Plan" in the bottom part of Figure 3.9. However, selecting the most suitable plan is not always an easy task and may involve hundred or even thousand of possibilities. For this reason, ContentCVS provided users with techniques to support the plan selection. Although the number of plans can be rather huge the number of involved axioms in them may be relatively small, thus ContentCVS shows all the axioms in plans in a separate frame,

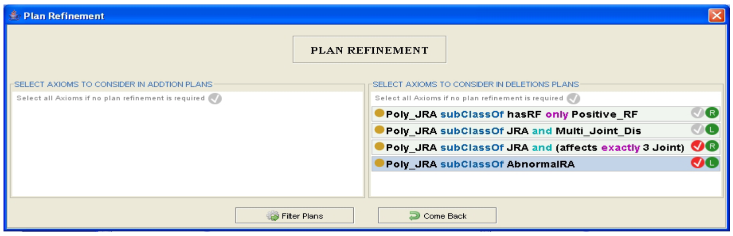

Extracted Plans
AVAILABLE PLANS TO CONSIDER DESIRED ENTAILMENTS
Select an available plan from list
Plan 4 (additions 1, deletions 2)
Axioms to Select (Plan Add)
Axioms to Deselect (Plan Delete)
Plan 4
Plan 4
Refine Plans
Show Impact
Use Plan
Come Back

JRA subClassOf RA and (suffered_By some Child)

Poly_JRA subClassOf JRA and (affects exactly 3 Joint)

Poly_JRA subClassOf AbnormalRA

Figure 3.9: GUI for plan selection in ContentCVS


Plan Refinement
PLAN REFINEMENT
SELECT AXIOMS TO CONSIDER IN ADDTION PLANS
SELECT AXIOMS TO CONSIDER IN DELETIONS PLANS
Select all Axioms if no plan refinement is required
Select all Axioms if no plan refinement is required
Filter Plans
Come Back

Poly_JRA subClassOf hasRF only Positive_RF

Poly_JRA subClassOf JRA and Multi_Joint_Dis

Poly_JRA subClassOf JRA and (affects exactly 3 Joint)

Poly_JRA subClassOf AbnormalRA

Figure 3.10: Refinement of plans in ContentCVS

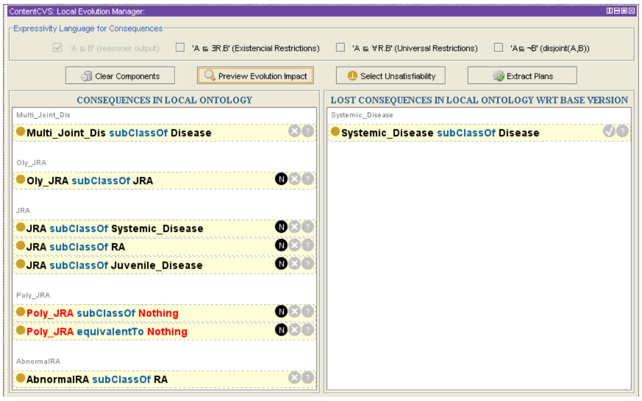
enabling the user to select which ones among them a plan must either add or delete (see Figure 3.10). Note that not all the refinements guarantee the existence of a plan.

Additionally, ContentCVS also allows the previsualization of the impact of a repair plan by computing the logical difference, as in Section 3.2.3.2, between the repaired ontology $(\mathcal{O}^L_{\mathsf{temp}} \cup \mathcal{P}^+) \setminus \mathcal{P}^-)$ and $\mathcal{O}^L_{\mathsf{temp}}$. This impact will show if the plan corrected the errors, recovered the intended lost entailments and/or other entailments where added or lost.

3.2.4. Local evolution in ContentCVS

ContentCVS also integrates a local evolution functionality in which developers can compare the current local version $\mathcal{O}^L$ w.r.t. the base revision $\mathcal{O}^L_{\mathsf{bak}}$ (i.e. the version imported in the las update) in order to analyse the logical impact of local changes. Figure 3.11 shows the GUI to represent the logical difference between $\mathcal{O}^L$ and $\mathcal{O}^L_{\mathsf{bak}}$. Axioms in the right hand side represents all the consequences in $\mathcal{O}^L$ (new consequences not entailed in $\mathcal{O}^L_{\mathsf{bak}}$ are marked with a 'N'). The left hand side shows those entailments in $\mathcal{O}^L_{\mathsf{bak}}$ but not in $\mathcal{O}^L$, that is, lost entailments in $\mathcal{O}^L$ that may be required to be recovered.

The GUI allows the selection of intended and unintended entailments, respectively, and the generation of repair plans as explained in Sections 3.2.3.3–3.2.3.7.


ContentCVS: Local Evolution Manager:
Expressivity Language for Consequences
'A ⊑ B' (reasoner output)
'A ⊑ ∃R.B' (Existencial Restrictions)
'A ⊑ ∀R.B' (Universal Restrictions)
'A⊑ ¬B' (disjoint(A,B))
Clear Components
Preview Evolution Impact
Select Unsatisfiability
Extract Plans
CONSEQUENCES IN LOCAL ONTOLOGY
LOST CONSEQUENCES IN LOCAL ONTOLOGY WRT BASE VERSION

Multi_Joint_Dis
Multi_Joint_Dis subClassOf Disease

Oly_JRA
Oly_JRA subClassOf JRA

JRA
JRA subClassOf Systemic_Disease
JRA subClassOf RA
JRA subClassOf Juvenile_Disease

Poly_JRA
Poly_JRA subClassOf Nothing
Poly_JRA equivalentTo Nothing

AbnormalRA
AbnormalRA subClassOf RA

Systemic_Disease
Systemic_Disease subClassOf Disease

Figure 3.11: Local consistency preservation in ContentCVS

3.3. Evaluation of ContentCVS

Our evaluation tries to answer the following important questions:

1. Do conflicts of the kind described in Section 3.2 occur in practice?
2. Are the implemented algorithms computationally feasible in practice?
3. Do our proposed techniques provide useful assistance to ontology engineers?
4. Is ContentCVS easy and intuitive to use?

To address the first question we analyse in Section 3.3.1 a sequence of versions of a realistic ontology and the respective change logs of each version. To address the second question, we describe in Section 3.3.2 a number of synthetic experiments using the same sequence of ontology versions. Finally, to address the last two questions, we have conducted a pilot user study, which we describe in Section 3.3.3.

3.3.1. Analysis of real changes

In this section, we study a sequence of 10 versions of a medical ontology developed at the University of Manchester and used in the context of the Clinergy project[8]. The sizes of the different versions vary from 71 concepts, 13 roles and 195 axioms in the first version to 207 concepts, 38 roles and 620 axioms in the last version; all the versions are expressed in the description logic $\mathcal{SHIQ}(D)$. The ontology was developed during a short period of time: from July 21st 2008 until July 28th 2008. On average, the developers generated one or two versions of the ontology each day. This situation, in which versions are generated very frequently, is consistent with the scenario described in Section 3.2.

[8] Clinergy project: `http://www.opengalen.org/sources/software.html`

This medical ontology is divided into two main disjoint concepts: Document_entity and Domain_entity. Most document related concepts are placed under Document_content_spec which is split into Section_content_spec and Clin_Holder. The latter is aimed at containing relations to clinical variables. Within the set of domain entities we can fin anatomical entities and entities related to the patient, namely: Condition, Act, Feature, Indicant, and Value. It also has a set of properties relating document and domain entities such as references and has_topic.

Table 3.6 summarises the content of the change logs of each version[9]. For example, the second column represents the changes performed from version $\mathcal{O}_1$ to version $\mathcal{O}_2$. The table clearly shows how the ontology grows as it is being developed: most of the changes involve addition of entities, axioms and annotations. Most of the added axioms are concept axioms (mostly inclusions, equivalence and disjoint axioms), which is a typical situation when modelling using OWL. Interestingly there are also a significant number of deletions, which reflects the fact that ontology developers are revising their modelling choices and fixing errors. Extra-logical changes such as modifications in the annotations are also fairly common, which suggests that they should be taken into account when identifying potential conflicts.

Change	$\mathcal{O}_2$	$\mathcal{O}_3$	$\mathcal{O}_4$	$\mathcal{O}_5$	$\mathcal{O}_6$	$\mathcal{O}_7$	$\mathcal{O}_8$	$\mathcal{O}_9$	$\mathcal{O}_{10}$
Concepts added	8	3	10	38	24	1	27	11	23
Concepts deleted	2	0	1	1	0	0	1	3	1
Roles added	0	0	3	4	3	0	6	1	0
Roles deleted	0	0	1	0	0	0	0	1	0
Concept axioms added	31	6	18	53	51	5	47	26	52
Concept axioms deleted	8	0	10	9	0	3	5	14	17
Role axioms added	0	0	3	6	6	0	7	2	0
Role axioms deleted	0	0	3	0	0	0	2	0	0
Annotations added	10	5	32	19	30	2	43	26	38
Annotations deleted	2	2	9	2	0	0	2	6	6

Table 3.6: Summary of change logs.

	Excerpt from verion $\mathcal{O}_4$
α_1	Document_entity $\sqcap$ Domain_entity $\sqsubseteq \bot$
α_2	Document_content_spec $\sqsubseteq$ Document_entity
α_3	Indicant $\sqsubseteq$ Domain_entity
α_4	Present_absent_indicant $\sqsubseteq$ Indicant
α_5	Section_content_spec $\sqsubseteq$ Document_content_spec
α_6	$\exists$includes_sub_doc.$\top \sqsubseteq$ Document_content_spec
α_7	has_sub_doc $\sqsubseteq$ includes_sub_doc

Table 3.7: A relevant fragment of version $\mathcal{O}_4$

[9] A document with an overview of the changes can be downloaded from: `http://krono.act.uji.es/people/Ernesto/contentcvs/synthetic-study`

	Changes over Domain Entities $(\Delta\mathcal{O}_4)^1$
γ_1	$\mathsf{Bruise\_to\_surface\_structure} \sqsubseteq \mathsf{Trauma\_to\_surface\_structure}$
γ_2	$\mathsf{Trauma\_to\_surface\_structure} \sqsubseteq \mathsf{Present\_absent\_indicant}$
γ_3	$\mathsf{Trauma\_to\_surface\_structure} \sqsubseteq \forall \mathsf{has\_locus}.\mathsf{Surface\_Anatomical\_structure}$
γ_4	$\mathsf{Surface\_Anatomical\_structure} \sqsubseteq \mathsf{Anatomic\_structure}$
γ_5	$\mathsf{Anatomic\_structure} \sqsubseteq \mathsf{Domain\_entity}$

	Changes over Document Entities $(\Delta\mathcal{O}_4)^2$
δ_1	$\mathsf{Bruise\_to\_surface\_structure} \sqsubseteq \mathsf{Section\_content\_spec}$
δ_2	$\mathsf{Surface\_trauma\_subsection\_spec} \sqsubseteq \mathsf{Section\_content\_spec}$
δ_3	$\mathsf{Bruise\_to\_surface\_structure} \sqsubseteq \exists \mathsf{has\_sub\_doc}.\mathsf{First\_heart\_sound\_clin\_holder}$
δ_4	$\mathsf{Bruise\_to\_surface\_structure} \sqsubseteq \exists \mathsf{has\_sub\_doc}.\mathsf{Second\_heart\_sound\_clin\_holder}$
δ_5	$\mathsf{Bruise\_to\_surface\_structure} \sqsubseteq \exists \mathsf{has\_sub\_doc}.\mathsf{Heart\_murmur\_clin\_holder}$

Table 3.8: Excerpt of changes $(\Delta\mathcal{O}_4)^1$ and $(\Delta\mathcal{O}_4)^2$ performed over version $\mathcal{O}_4$

In order to verify that conflicts of the kind described in Section 3.2 are likely to occur in practice, we have also performed a detailed analysis of the change logs. Our findings suggest that changes leading to an error, such as the unsatisfiability of a concept, may involve the simultaneous modification of different aspects of the domain. As an example, we have analyzed the evolution from version $\mathcal{O}_4$ to version $\mathcal{O}_5$. Two groups of changes, $(\Delta\mathcal{O}_4)^1$ and $(\Delta\mathcal{O}_4)^2$ from Table 3.8, were detected using the structural difference between $\mathcal{O}_4$ and $\mathcal{O}_5$.

Merging $(\Delta\mathcal{O}_4)^1$ and $(\Delta\mathcal{O}_4)^2$ together with the fragment of $\mathcal{O}_4$ from Table 3.7 leads to the unsatisfiability of the concept Bruise_to_surface_structure. This unsatisfiability is caused by the simultaneous use of Bruise_to_surface_structure both as a Domain_entity and as Document_entity. The changes in $(\Delta\mathcal{O}_4)^1$ describe Bruise_to_surface_structure as an anatomical structure (and hence as a *domain entity*), whereas the changes in $(\Delta\mathcal{O}_4)^2$ describe it as a section of content (and hence as a *document entity*).

Under the assumption that changes in $(\Delta\mathcal{O}_4)^1$ and $(\Delta\mathcal{O}_4)^2$ have been performed concurrently by different ontology engineers, the presence of incompatible changes leads precisely to the issues pointed out in Section 3.2. This assumption is reasonable, as different aspects of the domain (e.g., domain entities and document entities) are likely to be developed by different experts.

It is worth mentioning that the unsatisfiability was not repaired until version $\mathcal{O}_{10}$. Therefore, developers were not able to detect it until that version. Frameworks such as ContentCVS would have eased such detection and would have proposed a set of candidate repair plans. Moreover an early detection of an error usually eases its understanding. For example, in version $\mathcal{O}_5$ the explanation of the unsatisfiability involves 4 justifications with 9 axiom each, whereas in version $\mathcal{O}_9$ involves 12 justifications with 8-10 axioms each.

```
Input: $\mathcal{O}, \mathcal{O}'$: ontologies; approximation function $\mathsf{diff}_{\approx}$
  Compute $\Lambda_s$ and store its size (I) and computation time (II)
  repeat
      Randomly select $\mathcal{S} \subseteq \Lambda_s$, and compute $\mathcal{O}_{\mathsf{aux}} := \mathcal{O} \cup \mathcal{S}$
      Compute $\mathsf{diff}_{\approx}(\mathcal{O}_{\mathsf{aux}}, \mathcal{O}) \cup \mathsf{diff}_{\approx}(\mathcal{O}_{\mathsf{aux}}, \mathcal{O}')$ and store its size (III)
      Compute $\mathsf{diff}_{\approx}(\mathcal{O}, \mathcal{O}_{\mathsf{aux}})$ and store its size (IV)
      Get all justifications for entailments in $\mathsf{diff}_{\approx}$; store avg. time per justification (V)
      Compute $\rhd^{+}$ and $\rhd^{-}$, and store the number of roots (VI)
      Randomly select $\Im^{-}$ from roots of $\rhd^{-}$ and $\Im^{+}$ from roots of $\rhd^{-}$
      Compute $\mathbb{P}$ (minimal plans) and store number of plans (VII) and computation time (VIII)
  until 200 iterations have been performed
```

Table 3.9: Synthetic experiments

3.3.2. Performance evaluation

In our experiments we have simulated the evolution of an ontology by using the sequence of versions from Section 3.3.1. The experiments were performed on a laptop computer with a 1.82 GHz processor and 3Gb of RAM. The average classification time of an ontology in the sequence is approximately one second when using the Pellet reasoner.

For each pair $\mathcal{O}_i, \mathcal{O}_{i+1}$, $i \in \{1, \ldots, 9\}$ of consecutive versions, and both the smallest and largest approximations of the deductive difference implemented in ContentCVS, we have performed the experiment in Table 3.9. The Roman numbers in Table 3.9 refer to measurements that are stored during the experiment and presented in Table 3.10. These experiments follow our approach for conflict resolution in Table 3.5, with the assumption that version $\mathcal{O}_i$ is the local ontology, version $\mathcal{O}_{i+1}$ is the ontology in the repository, and the steps in Table 3.5 requiring manual intervention are performed randomly.

Table 3.10 summarises our results[10]. Most of the values in the table are either average or maximum values for the 200 iterations in the loop from Table 3.9. Average values are indicated with the tag ‘avg’ in the header, and maximum values with the tag ‘max’.

First, from a computational point of view, the main bottleneck is the computation of all the justifications for the entailments of interest. Once the justifications have been computed, the time needed for computing the plans is relatively low. In Table 3.10, we can see that the average time needed per justification can reach 5,9 seconds (see **V**, $\mathcal{O}_4\&\mathcal{O}_5$); if 300 justifications have to be computed in total, then the total time may reach 30 minutes. Hence, it is important to investigate optimisation techniques for computing all justifications; first steps in this direction have been taken in [KPHS07, HPS08b, SQJH08].

Second, the amount of information presented to the user largely depends on the selected approximation for the deductive difference (see Section 3.2.3.2). In the case of the smallest approximation, the average number of axioms in the relevant differences

[10] Synthetic study: `http://krono.act.uji.es/people/Ernesto/contentcvs/synthetic-study`

			Smallest diff$_\approx$ approximation						Largest diff$_\approx$ approximation					
	I	**II**	**III**	**IV**	**V**	**VI**	**VII**	**VIII**	**III**	**IV**	**V**	**VI**	**VII**	**VIII**
$\mathcal{O}$ **&** $\mathcal{O}'$			avg	avg	avg	avg	avg/max	avg	avg	avg	avg	avg	avg/max	avg
$\mathcal{O}_1 \& \mathcal{O}_2$	50	0.03	15	6	0.1	15	1 / 1	1.5	111	17	2.0	33	495 / 5508	10.3
$\mathcal{O}_3 \& \mathcal{O}_4$	82	0.02	13	4	0.26	13	3 / 18	1.7	128	90	0.9	30	46 / 896	3.6
$\mathcal{O}_4 \& \mathcal{O}_5$	93	0.02	31	14	0.1	29	3 / 32	1.2	267	48	5.9	49	2.7 / 6	30
$\mathcal{O}_7 \& \mathcal{O}_8$	110	0.03	19	15	0.02	18	1 / 4	0.07	216	78	1.2	47	488 / 3888	4
$\mathcal{O}_8 \& \mathcal{O}_9$	79	0.02	15	6	0.06	14	1 / 2	0.3	251	14	3.7	46	101 / 720	21.5
$\mathcal{O}_9 \& \mathcal{O}_{10}$	117	0.01	24	8	1.5	24	7 / 50	15.6	208	154	5.3	31	35 / 225	22.7

Table 3.10: Summary of results. Roman numbers refer to Table 3.9. Time measures are given in seconds

(see **III** and **IV**) is in the range 4–31, and the average number of minimal plans (see **VII**) is in the range 1–50. In contrast, in the case of the largest approximation, these average numbers are in the ranges 14–267, and 6–5508 respectively. The amount of information the user would need to consider is thus much larger. Table 3.10 also shows that the use of the dependency relations $\rhd^+$ and $\rhd^-$ can lead to a significant reduction in the amount of information that is initially presented to the user (**VI**). Note that for the largest approximation the number of roots for $\rhd^+$ and $\rhd^-$ is comparable to the size of the relevant deductive differences for the smallest approximation.

Overall, we believe that this experiment demonstrates that the algorithms implemented in ContentCVS exhibit reasonable performance, and that our approach is computationally feasible. The use of larger approximations of deductive difference may, however, require improved techniques for computing justifications. The use of larger approximations may also risk overwhelming the user with information, although presentation techniques such as the dependency one implemented in ContentCVS can help to ameliorate this problem.

3.3.3. User study

We have conducted a pilot user study to evaluate the usability of the GUI implemented in ContentCVS, as well as to provide empirical evidence of the adequacy of our approach in practice. The details of the conducted study, including the questionnaire and the test ontologies, are available online[11].

3.3.3.1. Design of the study

The user study consists of three main parts, each of which involves the completion of a number of tasks, as we describe next.

Part 1: Local evolution of an ontology

The first part of the study simulates a conventional ontology repair scenario where a (single) developer performs a number of changes to his/her ontology $\mathcal{O}_0$ and, as a result, creates a new version $\mathcal{O}_1$ of the ontology in which errors may have been

[11] User study: `http://krono.act.uji.es/people/Ernesto/contentcvs/user-study`

introduced. The main goal is to evaluate the repair techniques implemented in ContentCVS, in particular the identification of errors using deductive differences and the repair of these errors via the generation and selection of suitable plans.

The test ontology used in this part of the study describes the domain of academic publications and bibliographic references, which the participants in the study are expected to be relatively familiar with. The changes to the original version $\mathcal{O}_0$ involve the definition of three new concepts and the deletion of a property domain restriction. Users were first asked to use a reasoner to classify $\mathcal{O}_1$, examine the resulting entailments and try to understand the given justifications. Next, users were asked to identify and repair two kinds of errors, namely the occurrence of unintended entailments in $\mathcal{O}_1$ that did not occur in $\mathcal{O}_0$, and the lost of intended entailments that held in $\mathcal{O}_0$, but not in $\mathcal{O}_1$. Finally, users were asked to repeat this process by taking into account not only simple subsumptions, but also entailments of the form $A \sqsubseteq \neg B$, $A \sqsubseteq \exists R.B$ and $A \sqsubseteq \forall R.B$.

Part 2: Reconciliation of two independently-developed ontology versions

This part of the study simulates the scenario where a (single) developer working with a local copy $\mathcal{O}^L$ of an ontology performs a CVS-update and needs to reconcile the local version $\mathcal{O}^L$ with the version $\mathcal{O}^R$ in the repository. The main goal is twofold; first, to evaluate the functionality in ContentCVS for directly comparing ontology versions, both from a syntactic and from a semantic point of view; second, to evaluate the means provided by ContentCVS for building an error-free reconciled version ready to be committed to the CVS repository.

Users were asked to reconcile two versions of an ontology describing types of Juvenile Idiopathic Arthritis. To this end, they first examined the structural difference between both versions and selected the axioms to be included in a temporary version $\mathcal{O}^L_{\mathsf{temp}}$ of the reconciled ontology. Next, users classified $\mathcal{O}^L_{\mathsf{temp}}$ and identified errors in the form of missing intended entailments or new unintended ones. As in Part 1, users were then asked to repeat this latter step by considering additional types of entailments and to use the proposed dependency relation between entailments to group them. Finally users were asked to repair the identified errors by selecting a suitable plan. Note that the test ontology versions used in this part of the study closely reproduces our running example, and the tasks involved follow our methodology in Table 3.5 from Section 3.2.3.

Part 3: Collaborative development of an ontology

The final part of the study simulates the scenario where a number of users are developing an ontology concurrently using ContentCVS. As in Part 1, we used the familiar domain of publications and bibliographic references.

Each test involved three or four participants in the study. To produce a controlled experiment, each participant was asked to extend an initial version of the ontology by performing a number of changes specified a-priori. The first participant was in charge of performing changes concerning different types of academic staff members; the second one made changes concerning events such as conferences; the third one made changes concerning academic organisations; finally, the fourth one was asked to describe different kinds of resources and publications.

Each participant was asked to perform a CVS-commit either upon completion of all the changes, or when explicitly indicated in their task sheets. In order to provide a more realistic environment, the exact point in time in which users attempt to commit their changes was not a-priori fixed. If the commit failed, the participant was asked to perform an update and reconcile the changes using their ContentCVS client. Once the participants had agreed upon a reconciled version of the ontology, they were asked to discuss it among themselves and with the coordinator of the study.

3.3.3.2. Results and discussion

In total, eight people participated in Parts 1 and 2 of the study. In the case of Part 3, we conducted three collaborative tests each of which involved either three or four participants.

The participants of the study are academic researchers, most of them working in fields other than the Semantic Web. For example, some of the participants work in a bio-genomics group, others in a robotics and cognitive sciences group, and so on. Most users evaluated their experience in knowledge representation as "intermediate", in first order logic as either "intermediate" or "low" and in description logics and OWL also as either "intermediate" or "low". All participants except for one had tried Protégé before and half of them had used a reasoner before when developing an ontology. However, none of the participants was familiar with justification-based explanations. The results can be summarised as follows:

- *Part 1*: Most users were able to understand the justifications provided by Protégé, although most of them found it "hard" or "very hard" to resolve potential errors manually. Furthermore, all the participants were able to identify both new unintended entailments and lost intended entailments when using ContentCVSand described the functionality provided by our tool for identifying these entailments as either "good" or "very good". Most participants were satisfied with the smallest approximation of the deductive difference implemented in ContentCVS, and complained about excessive amounts of displayed information when using the largest implemented approximation instead. None of them considered that ContentCVS should aim at implementing richer approximations. Concerning the generation of plans, most users declared this functionality as either "useful" or "very useful" and found the capabilities of ContentCVS to recommend plans also useful.

- *Part 2*: Most users considered either "useful" or "very useful" the computation of structural differences between ontology versions. However, many users found it difficult to detect potential errors simply by examining the structural difference. As in Part 1, users liked the functionality in ContentCVS for detecting potential errors using approximations of the deductive difference. Interestingly, by using the largest approximation implemented in ContentCVS, users were able to detect errors other than unsatisfiable concepts and atomic subsumptions, which they found useful. Furthermore, all users considered that the use of a large approximation leads to an excessive amount of displayed information; however, all of them also found the presentation technique based on the dependency relation ($\triangleright$) very useful in alleviating this problem, but complained about the response time. Finally, most users were either "very satisfied" or "satisfied" with the reconciled ontology obtained after the execution of the selected repair plan.

- *Part 3*: Most participants had used a CVS system before for managing text files and described the CVS functionality implemented in ContentCVS as either "very useful" or "useful". Many participants emphasised the importance of some previous training for taking full advantage of the CVS functionality in ContentCVS. As in parts 1 and 2, the use of a combination of structural and deductive differences for detecting errors plus the computation of plans for repairing them was evaluated very positively. Concerning the ontology obtained as a result of the collaboration, the participants were able to obtain an error-free ontology and were satisfied with the result. Only in one case the final discussion revealed an error in the final ontology; however the participants acknowledged that this error was not due to a deficiency of the tool.

Finally, all users evaluated the tool very positively; in particular, most of them evaluated the GUI as "good" and the ontology development workflow implemented in the tool as either "very good" or "good". Therefore, we consider the feedback very positive in general. The main points of criticism were the following:

- Excessive amounts of information displayed when using "large" approximations of the deductive difference. Even if the identification of dependencies between entailments helped in alleviating this problem, we consider it important to investigate new ways of organising a potentially overwhelming number of entailments.

- Slow response of the tool when computing all justifications of certain entailments and/or computing large approximations of the deductive difference. For large-scale ontology development, the further optimisation of our algorithms will be necessary. To this end, we consider especially promising the use of *incremental reasoning* techniques (see for example first results in [CHWK07]), which aim at avoiding unnecessary re-computations after performing a (small) number of changes to the ontology.

3.4. Discussion

We have proposed a novel approach for facilitating concurrent ontology development, described a tool that implements it and presented an evaluation of the tool. The main contributions of our research can be summarised as follows:

- We have adapted the Concurrent Versioning paradigm to ontology engineering, which allows developers to make changes concurrently and remotely to the same ontology, track changes, and manage versions.
- We have proposed notions of equivalence and difference between ontology versions.
- We have proposed a collection of techniques for resolving conflicts between ontology versions both at the structural and at the semantic level.
- We have adapted and enhanced state-of-the art ontology debugging and repair techniques to our collaborative setting.
- We have implemented and evaluated a prototypical tool and obtained promising empirical results and encouraging feedback from users.

In future work, we plan to improve our tool in a number of ways. First, we are working on improving the system's performance and in particular the computation of justifications. As pointed out in Section 3.3, the use of incremental reasoning techniques is particularly interesting in this regard. Second, we are enhancing the tool with new features. In particular, we plan to support richer approximations and the use of *unit tests*—files containing a set of unintended entailments that can be used to detect modelling errors. Furthermore, we aim to integrate in our tool some of the functionality provided by state-of-the-art frameworks, such as Collaborative Protégé, for holding discussions and annotating changes. Another interesting direction for future research would be to provide means for assigning responsibilities and duties to different ontology developers and support for automatically checking whether changes made by developers are consistent with their duties. Finally, the integration of a reference thesaurus [JYJRBRS09b] within the proposed framework would minimise the lexical conflicts related to the use of different labels to refer the same entity

CHAPTER 4

Logic-based assessment of ontology integration

Effective ontology integration techniques are often needed both during ontology development (e.g., when an ontology being developed reuses one or more external ontologies), and when ontologies are used in conjunction with data (e.g. when integrating and querying data sources annotated using different ontologies).

When the ontologies to be integrated have been independently developed, their vocabularies will most likely diverge, either because they use different namespaces, or because they use different names or naming conventions to refer to their entities. As a consequence, these ontologies will most likely be unrelated from a logical point of view, even if they intuitively overlap.

To exchange or migrate data between ontology-based applications, it is crucial to establish correspondences (or *mappings*) between their ontologies. Creating such mappings manually is often unfeasible due to the size and complexity of modern ontologies. For example, SNOMED CT in its version from January 2009 contains more than 300,000 entities, while NCI (version 08.05d) contain around 79,000 entities. Since the number of potential mappings grows (at least) quadratically with the number of entities in the relevant ontologies, it would be necessary to consider (at least) 10 billion candidate mappings between SNOMED CT and NCI.

The problem of automatically generating mappings between independently developed ontologies (aka the ontology alignment/matching problem) has been a prominent topic of research and it has been extensively investigated (e.g., see [KS03, CSH06, ES07, AG09, SE08, SE10] for comprehensive surveys and available tools).

Ontology integration has also benefited from the relationship between ontologies and database schemas. The alignment and integration of schemas has been a longstanding problem for the database community, and it is still an active research area (e.g., see [RB01, SE05, DH05] for excellent surveys).

When reasoning with the ontologies to be integrated together with the automatically generated mappings, however, errors are likely to occur. There are two main causes for these errors. On the one hand, mappings suggested by automated tools are error-prone. On the other hand, even if the correct mappings have been found, the ontologies may contain conflicting descriptions of the overlapping entities. These errors manifest themselves as unintended logical consequences (e.g., unsatisfiable concepts or unintended subsumptions), and they can be difficult to detect, understand and repair. However, existing tools provide little or no support for the user in trying to obtain the right logical consequences when integrating ontologies using mappings.

The contributions of this chapter are aimed to provide such support by means of semi-automatic and automatic techniques to detect and repair unintended consequences of the integration. The formal semantics of the mappings are defined in Section 4.1. Section 4.2 presents a method and a tool to guide the user in the analysis of the impact of integrating two ontologies using mappings. In Section 4.3 a set of heuristics have been implemented according to logic-based principles in order to automatically assess and repair the integration of large ontologies.

4.1. Formal representation of ontology mappings

Mappings are often represented as a tuple $\langle id, e_1, e_2, n, \rho \rangle$ [Euz07], where id is a unique identifier for the mapping, e_1, e_2 are entity names in the vocabulary of $\mathcal{O}_1$ and $\mathcal{O}_2$ respectively, n is a numeric confidence measure between 0 and 1, and ρ is a relation between e_1 and e_2 (typically subsumption ($\sqsubseteq$), equivalence ($\equiv$), or disjointness ($\bot$)).

The use of logic-based techniques requires the adoption of a *formal representation* of the mappings. This is crucial to reason unambiguously with the source ontologies and the corresponding mappings. A number of specialised semantics for ontology mappings have been proposed so far in the literature (e.g. [Euz07, ST09, GPS09]), but there is no consensus on which ones are more suitable for practical applications. In this research, however, we have chosen to represent ontology mappings as OWL 2 axioms [CHM+08]. Such a representation seems semantically coherent, and allows us to reuse the extensive range of OWL algorithmic techniques and infrastructure currently available. In this setting, we can therefore restrict ourselves to consider the situation where OWL 2 ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$ are integrated via a third OWL 2 ontology $\mathcal{M}$, which contains the relevant mappings (e.g., $\mathcal{U} := \mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}$).

We assume from now on that a set of mappings is represented as an OWL 2 ontology $\mathcal{M}$, where mappings are given as OWL axioms of the form $\mathsf{SubClassOf}(e_1\ e_2)$, $\mathsf{EquivalentClasses}(e_1\ e_2)$, or $\mathsf{DisjointClasses}(e_1\ e_2)$, with id (the mapping id) and n (the confidence value) added as OWL axiom annotations [CHM+08]. We denote with $\mathsf{conf}(\alpha)$ the confidence value with which an axiom α is annotated. Representing mappings in this way gives them a standard "crisp" semantics, with ids and confidence values being represented as annotations and thus having no effect on the entailments.

4.2. Supporting the right ontology integration using ContentMap

In this section we describe a novel method and a tool[1] called ContentMap [2] [JRCHB09, JRGHL09c] to guide the user in the assessment of the integration of two independently developed ontologies using mappings.

The rest of the sections are organized as follows. Section 4.2.1 introduces the main challenges that should be addressed for an error-free ontology integration. The method, algorithms and tool support are presented in Section 4.2.2. Finally, Section 4.2.3 gives an evaluation of ContentMap , which suggests that our approach is useful in practice.

Table 4.1: Example ontologies

	Ontology $\mathcal{O}_1$
α_1	$\mathsf{Juvenile\_Arthritis} \sqsubseteq \mathsf{Systemic\_Disease} \sqcap \mathsf{Rheumatoid\_Arthritis}$
α_2	$\mathsf{Multi\_Joint\_Disease} \sqsubseteq \mathsf{Disease} \sqcap \geqslant 5\, \mathsf{affects.Joint}$
α_3	$\mathsf{Rheumatoid\_Arthritis} \sqsubseteq \mathsf{Disease} \sqcap \exists \mathsf{has\_Factor}.\top$
α_4	$\mathsf{Poly\_Juvenile\_Arthritis} \sqsubseteq \mathsf{Juvenile\_Arthritis} \sqcap \mathsf{Multi\_Joint\_Disease}$
α_5	$\mathsf{Oly\_Juvenile\_Arthritis} \sqsubseteq \mathsf{Juvenile\_Arthritis} \sqcap \neg \mathsf{Poly\_Juvenile\_Arthritis}$
α_6	$\mathsf{Oly\_Juvenile\_Arthritis} \sqsubseteq \forall \mathsf{has\_Factor.Negative\_Factor}$
α_7	$\mathsf{Negative\_Factor} \sqcap \mathsf{Positive\_Factor} \sqsubseteq \bot$
	Ontology $\mathcal{O}_2$
β_1	$\mathsf{Juv\_Rheum\_Arthritis} \sqsubseteq \mathsf{Rheum\_Arthritis} \sqcap \mathsf{Juv\_Disease}$
β_2	$\mathsf{Rheum\_Arthritis} \sqsubseteq \mathsf{Disease} \sqcap \exists \mathsf{has\_Rheum\_Factor}.\top$
β_3	$\mathsf{Poly\_Juv\_Rheum\_Arthritis} \sqsubseteq \mathsf{Juv\_Rheum\_Arthritis} \sqcap = 3\, \mathsf{affects.Joint}$
β_4	$\mathsf{Poly\_Juv\_Rheum\_Arthritis} \sqsubseteq \forall \mathsf{has\_Rheum\_Factor.Positive\_Rheum\_Factor}$
β_5	$\mathsf{Negative\_Rheum\_Factor} \sqcap \mathsf{Positive\_Rheum\_Factor} \sqsubseteq \bot$

4.2.1. Challenges in ontology integration

In this section, we discuss some of the problems that arise when integrating ontologies using mappings, and we analyze the requirements for a suitable tool. As use case, we have considered the domain of **JIAO** (see Section 1.4). For example, suppose that Peter wants to develop an ontology about juvenile forms of arthritis. He finds

[1] A Protégé 4 plugin is freely available for download: `http://krono.act.uji.es/people/Ernesto/contentmap`

[2] A logiC-based ONtology inTEgratioN Tool using MAPpings

Table 4.2: Example mappings

Intended Mappings
$\langle \mu_1$, 1:Joint, 2:Joint, 1,0, $(\equiv)\rangle$;
$\langle \mu_2$, 1:Disease, 2:Disease, 1, $(\equiv)\rangle$;
$\langle \mu_3$, 1:Disease, 2:Disease, 1,0, $(\equiv)\rangle$;
$\langle \mu_4$, 1:affects, 2:affects, 0,87, $(\equiv)\rangle$;
$\langle \mu_5$, 1:Rheumatoid_Arthritis, 2:Rheum_Arthritis, 1,0, $(\equiv)\rangle$;
$\langle \mu_6$, 2:Juv_Disease, 1:Disease, 0,63, $(\sqsubseteq)\rangle$;
$\langle \mu_7$, 1:Poly_Juvenive_Arthritis, 2:Poly_Juv_Rheum_Arthritis, 0,7, $(\equiv)\rangle$;
$\langle \mu_8$, 1:has_Factor, 2:has_Rheum_Factor, 0,7, $(\equiv)\rangle$;
$\langle \mu_9$, 1:Positive_Factor, 2:Positive_Rheum_Factor, 0,75, $(\equiv)\rangle$;
$\langle \mu_{10}$, 1:Negative_Factor, 2:Negative_Rheum_Factor, 0,75, $(\equiv)\rangle$;
$\langle \mu_{11}$, 2:Juv_Rheum_Arthritis, 1:Rheumatoid_Arthritis, 0,5, $(\sqsubseteq)\rangle$;
Erroneous Mappings
$\langle \mu_{12}$, 1:Rheumatoid_Arthritis, 2:Juv_Rheum_Arthritis, 0,5, $(\sqsubseteq)\rangle$;
$\langle \mu_{13}$, 1:Disease, 2:Juv_Disease, 0,63, $(\sqsubseteq)\rangle$;
$\langle \mu_{14}$, 1:Positive_Factor, 2:Negative_Rheum_Factor, 0,63, $(\equiv)\rangle$;
$\langle \mu_{15}$, 2:Positive_Factor, 1:Negative_Rheum_Factor, 0,63, $(\equiv)\rangle$;

on the Web two independently developed ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$ with different vocabularies, which describe different types of arthritis and juvenile diseases respectively, and decides that an integration of these two ontologies would make a good starting point. Although largely independent, the two ontologies do overlap; for example, both describe a particular form of juvenile arthritis known as JRA (Juvenile Rheumatoid Arthritis), although they use different vocabulary in their descriptions. The overlapping parts of $\mathcal{O}_1$ and $\mathcal{O}_2$ are shown in Table 4.1.

Suppose that Peter—the ontology developer—uses an automatic mapping tool to find correspondences between entities in $\mathcal{O}_1$ and $\mathcal{O}_2$. Table 4.2 contains the mappings obtained by Peter using that ontology mapping tool [3]. The prefixes ‘1:’ and ‘2:’ for the entity names refer to the namespaces (omitted in Table 3.2) of $\mathcal{O}_1$ and $\mathcal{O}_2$. Note that the mappings and confidence values suggested by different tools can vary enormously, so experience and/or experimentation may be needed in order to select the most suitable tool (or combination of tools) for the problem at hand.

As shown in Table 4.2, some of the mappings are clearly erroneous and, under any reasonable semantics for the mappings, will lead to unintended logical consequences. Indeed, the mappings μ_{14} and μ_{15} entail the logical equivalence of two concepts that are disjoint in both $\mathcal{O}_1$ and $\mathcal{O}_2$ (see also intended mappings μ_9 and μ_{10}); also, the mapping μ_{12} will lead to unintended subsumptions since not all forms of rheumatoid arthritis affect only children. Furthermore, even if only intended mappings had been generated, errors would still occur due to conflicting descriptions of some of the enti-

[3] The tool OLA (OWL lite alignment tool) [EV04] was used for this example.

ties *shared* in both ontologies. For example, the descriptions of Polyarticular Juvenile Rheumatoid Arthritis (axioms α_4, α_2 in $\mathcal{O}_1$ and β_3 in $\mathcal{O}_2$) are contradictory: in $\mathcal{O}_1$ it is described as a disease that affects at least 5 different joints, whereas in $\mathcal{O}_2$ it is said to affect exactly 3 joints. Thus, the intended mapping μ_7 reveals this incompatibility between $\mathcal{O}_1$ and $\mathcal{O}_2$.

This example suggests some requirements for a tool supporting ontology integration using mappings: *(i)* generating sets of mappings, either manually or by selecting one or more mapping algorithms and setting their parameters; *(ii)* editing sets of mappings and filtering them according to different criteria; *(iii)* reasoning with the ontologies to be integrated together with the relevant mappings; *(iv)* comparing the entailments holding before and after the integration and detecting possible unintended entailments; *(v)* suggesting possible ways to repair the identified errors.

4.2.2. Proposed method, algorithms and tool support

Our novel approach to address the requirements from Section 4.2.1 is given at a high level in Figure 4.1 and formally in Table 4.3. It is an interactive process where some steps involve computations that tools should perform automatically, while others (marked with ($\star$)) may require manual intervention. It consists of four main parts: *i)* Computation of mappings (Steps 1–3); *ii)* Computation of new entailments (Steps 4–6); *iii)* Detection of errors (Step 7); *iv*) Repair of the identified errors (Steps 8–12).

Table 4.3: Ontology integration method

```
Input: $\mathcal{O}_1, \mathcal{O}_2$: ontologies with $\mathsf{Sig}(\mathcal{O}_1) = \mathbf{S}_1, \mathsf{Sig}(\mathcal{O}_2) = \mathbf{S}_2$ and $\mathbf{S}_1 \cap \mathbf{S}_2 = \{\top, \bot\}$
Output: $\mathcal{O}'_1, \mathcal{O}'_2$: modified ontologies
$\mathcal{M}'$: Mappings between $\mathbf{S}_1$ and $\mathbf{S}_2$
 1: ($\star$) Select mapping algorithm $\mathsf{map}(\mathcal{O}_1, \mathcal{O}_2)$
 2: ($\star$) Select $\mathcal{M} \subseteq \mathsf{map}(\mathcal{O}_1, \mathcal{O}_2)$
 3: ($\star$) if $\mathcal{O}_1, \mathcal{O}_2, \mathcal{M}$ satisfactory, then return $\mathcal{O}'_1 := \mathcal{O}_1, \mathcal{O}'_2 := \mathcal{O}_2, \mathcal{M}' := \mathcal{M}$
 4: $\mathcal{U} := \mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}$
 5: ($\star$) Select approximation functions $\mathsf{diff}^{\approx}$ and $\mathsf{mdiff}^{\approx}$
 6: Compute $\Lambda = \mathsf{diff}^{\approx}_{\mathbf{S}_1}(\mathcal{O}_1, \mathcal{U}) \cup \mathsf{diff}^{\approx}_{\mathbf{S}_2}(\mathcal{O}_2, \mathcal{U}) \cup \mathsf{mdiff}^{\approx}_{\mathbf{S}_1, \mathbf{S}_2}(\mathcal{M}, \mathcal{U})$
 7: ($\star$) Select $\Im^+, \Im^- \subseteq \Lambda$
 8: ($\star$) Select $\mathcal{O}^- \subseteq \mathcal{U}$
 9: $\mathbb{P}$ := all minimal plans for $\mathcal{U}$ given $\Im^+$, $\Im^-$, and $\mathcal{O}^-$
10: ($\star$) if no satisfactory plan in $\mathbb{P}$, then come back to either Step 2 or Step 7
11: ($\star$) Select $\mathcal{P} \in \mathbb{P}$
12: return $\mathcal{O}'_1 := \mathcal{O}_1 \setminus \mathcal{P}, \mathcal{O}'_2 := \mathcal{O}_2 \setminus \mathcal{P}, \mathcal{M}' := \mathcal{M} \setminus \mathcal{P}$
```

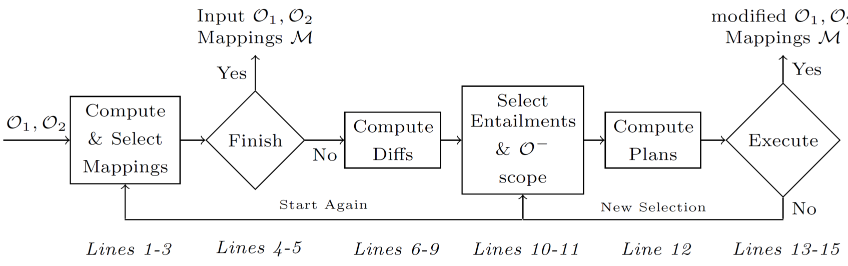


Figure 4.1: Merging method (informal). Numbers refer to Table 4.3

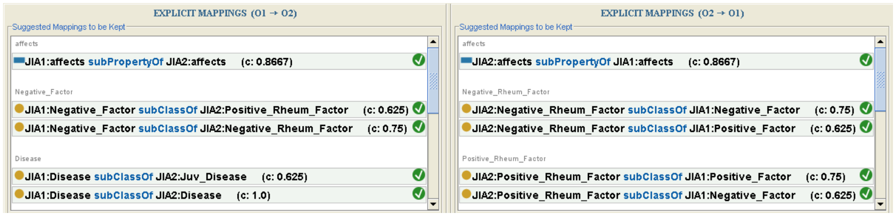
4.2.2.1. Computation of the mappings

Ontology mappings can be computed using one or more mapping tools (see Step 1), and can subsequently be refined (Step 2) in various ways (e.g., via manual selection or filtering according to a given threshold). After obtaining and possibly refining the mappings, a user may decide that the integration process is complete (Step 3), in which case it remains a purely syntactic process. They may, however, want to analyse the semantic consequences of the integration, in which case a reasoner is required.

ContentMap provides for the input ontologies to be loaded, and for one or more mapping tools to be selected. Different weights can also be assigned to each of the mapping tools. It is also possible to load pre-computed mappings in the form of an OWL ontology. The GUI for visualising the mappings in ContentMap is shown in Figure 4.2. Mappings marked with ✓ are to be accepted, whereas mappings marked with ⊗ are to be rejected. Users can automatically filter the mappings by setting a confidence threshold τ: mappings with a confidence value lower than τ are automatically marked for rejection while those with a confidence value greater than τ are marked for acceptance. These selections can then be refined at will.

4.2.2.2. Computation of new entailments

To help users understand the semantic consequences of the integration, they should be informed about new entailments that hold in the merged ontology $\mathcal{U}$, but not in $\mathcal{O}_1$,


EXPLICIT MAPPINGS (O1 → O2)

Suggested Mappings to be Kept

affects
JIA1:affects subPropertyOf JIA2:affects (c: 0.8667) ✓

Negative_Factor
JIA1:Negative_Factor subClassOf JIA2:Positive_Rheum_Factor (c: 0.625) ✓
JIA1:Negative_Factor subClassOf JIA2:Negative_Rheum_Factor (c: 0.75) ✓

Disease
JIA1:Disease subClassOf JIA2:Juv_Disease (c: 0.625) ✓
JIA1:Disease subClassOf JIA2:Disease (c: 1.0) ✓

EXPLICIT MAPPINGS (O2 → O1)

Suggested Mappings to be Kept

affects
JIA2:affects subPropertyOf JIA1:affects (c: 0.8667) ✓

Negative_Rheum_Factor
JIA2:Negative_Rheum_Factor subClassOf JIA1:Negative_Factor (c: 0.75) ✓
JIA2:Negative_Rheum_Factor subClassOf JIA1:Positive_Factor (c: 0.625) ✓

Positive_Rheum_Factor
JIA2:Positive_Rheum_Factor subClassOf JIA1:Positive_Factor (c: 0.75) ✓
JIA2:Positive_Rheum_Factor subClassOf JIA1:Negative_Factor (c: 0.625) ✓

Figure 4.2: GUI for visualising explicit mappings in ContentMap

$\mathcal{O}_2$ and $\mathcal{M}$ alone. To this end, as in Definition 3.4, we reuse the notion of *deductive difference* [KWW08] $(\mathsf{diff}_{\mathbf{S}}(\mathcal{O},\mathcal{O}'))$.

In our integration scenario, we want to inform the user about new entailments that hold over the signature of $\mathcal{O}_1$ or $\mathcal{O}_2$ (i.e., $\mathbf{S}_1$ or $\mathbf{S}_2$) as a result of the integration, i.e., $\mathsf{diff}_{\mathbf{S}_1}(\mathcal{O}_1,\mathcal{U})$ and $\mathsf{diff}_{\mathbf{S}_2}(\mathcal{O}_2,\mathcal{U})$. In our example from Tables 4.1 and 4.2, where the mappings are given as an OWL 2 ontology as described above, the entailments $(\mathsf{Positive\_Factor} \sqsubseteq \bot)$ and $(\mathsf{Rheum\_Arthritis} \sqsubseteq \mathsf{Juv\_Disease})$ belong to $\mathsf{diff}_{\mathbf{S}_1}(\mathcal{O}_1,\mathcal{U})$ and $\mathsf{diff}_{\mathbf{S}_2}(\mathcal{O}_2,\mathcal{U})$ respectively.

In addition, we want to inform the user about new entailments that contain symbols from both $\mathcal{O}_1$ and $\mathcal{O}_2$. These entailments can be seen as inferred mappings provided that the notion of a mapping is generalised to be an arbitrary DL axiom using terms from two signatures:

Definition 4.1 ($\mathcal{DL}$-mapping). *Let $\mathcal{DL}$ be a DL and let $\mathbf{S}_1, \mathbf{S}_2$ be two $\mathcal{DL}$-signatures s.t. $\mathbf{S}_1 \cap \mathbf{S}_2 = \{\top,\bot\}$. Let $\mathbf{S} = \mathbf{S}_1 \cup \mathbf{S}_2$. A $\mathcal{DL}$-mapping between $\mathbf{S}_1$ and $\mathbf{S}_2$ is a $\mathcal{DL}$-axiom α over $\mathbf{S}$ such that $(\mathsf{Sig}(\alpha) \cap \mathbf{S}_i) \setminus \{\top,\bot\} \neq \emptyset$ for each $i \in \{1,2\}$. The axiom α is annotated with an identifier and with a confidence value $0 < \mathsf{conf}(\alpha) \leq 1$.*

Definition 3.4 (*deductive difference*) can be extended to take into account the new mappings, i.e.:

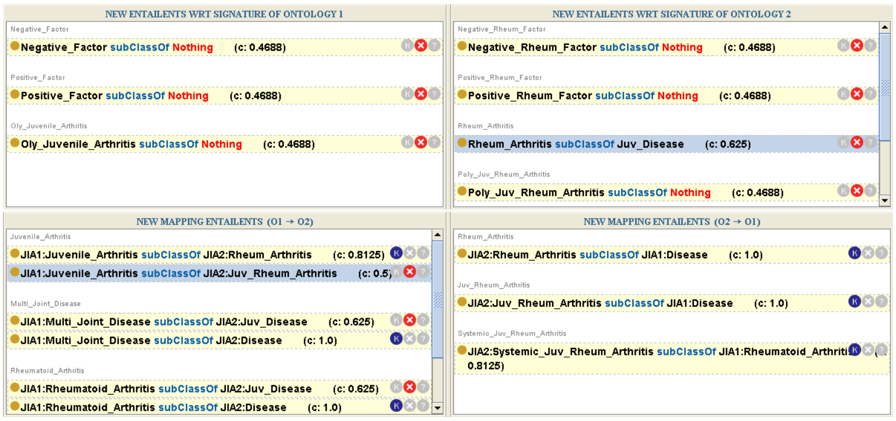
$$\mathsf{mdiff}_{\mathbf{S}_1,\mathbf{S}_2}(\mathcal{O},\mathcal{O}') = \{\alpha \in \mathsf{diff}_{\mathbf{S}}(\mathcal{O},\mathcal{O}') \mid \alpha\ \mathcal{DL}\text{-mapping between } \mathbf{S}_1, \mathbf{S}_2\} \quad (4.1)$$

In our scenario, inferred mappings are captured by $\mathsf{mdiff}_{\mathbf{S}_1,\mathbf{S}_2}(\mathcal{M},\mathcal{U})$. For example, the mapping $(\mathsf{1{:}Juvenile\_Arthritis} \sqsubseteq \mathsf{2{:}Rheum\_Arthritis})$ would belong to this difference, because it is entailed by $\mathcal{U}$ but not by $\mathcal{M}$ alone.

The deductive difference, as commented previously in Section 3.2.3.2, is undecidable for complex logics. Currently, algorithms only exist for (fragments of) the OWL 2 EL and QL profiles [MCH+09, KWW08, KWZ08], however, their output could be infinite. These drawbacks motivate the need for *approximations*— subsets of the difference—that the user can select in Step 5 from Table 4.3. The approximation function $\mathsf{diff}^{\approx}_{\mathbf{S}}(\mathcal{O},\mathcal{O}')$ for $\mathsf{diff}_{\mathbf{S}}(\mathcal{O},\mathcal{O}')$ is defined as in Definition 3.5. Analogously, the set of new mappings can also be approximated as follows:

Definition 4.2 (Approximation of the Deductive Difference for $\mathcal{DL}$-mappings). *A function $\mathsf{mdiff}^{\approx}_{\mathbf{S}_1,\mathbf{S}_2}(\mathcal{O},\mathcal{O}')$ is an approximation for $\mathsf{mdiff}_{\mathbf{S}_1,\mathbf{S}_2}(\mathcal{O},\mathcal{O}')$ if for any two $\mathcal{DL}$-ontologies $\mathcal{O},\mathcal{O}'$, $\mathsf{mdiff}^{\approx}_{\mathbf{S}_1,\mathbf{S}_2}(\mathcal{O},\mathcal{O}') \subseteq \mathsf{mdiff}_{\mathbf{S}_1,\mathbf{S}_2}(\mathcal{O},\mathcal{O}')$.*

ContentMap, as ContentCVS in Section 3.2.3.2, provides an approximation of the deductive difference by only allowing entailments of the following kind: *(i)* $A \sqsubseteq B$, *(ii)* $A \sqsubseteq \neg B$, *(iii)* $A \sqsubseteq \exists R.B$, *(iv)* $A \sqsubseteq \forall R.B$, and *v)* $R \sqsubseteq S$. Where A, B are atomic concepts (including $\top,\bot$) and R, S atomic roles. Note that, the smallest implemented approximation considers only axioms of the form *(i)*, which amounts to comparing the classification hierarchy of both ontologies. Figure 4.3 shows the ContentMap GUI with the deductive differences, where new ontology entailments are displayed above and new entailed mappings below.



NEW ENTAILENTS WRT SIGNATURE OF ONTOLOGY 1

Negative_Factor
Negative_Factor subClassOf Nothing (c: 0.4688)

Positive_Factor
Positive_Factor subClassOf Nothing (c: 0.4688)

Oly_Juvenile_Arthritis
Oly_Juvenile_Arthritis subClassOf Nothing (c: 0.4688)

NEW ENTAILENTS WRT SIGNATURE OF ONTOLOGY 2

Negative_Rheum_Factor
Negative_Rheum_Factor subClassOf Nothing (c: 0.4688)

Positive_Rheum_Factor
Positive_Rheum_Factor subClassOf Nothing (c: 0.4688)

Rheum_Arthritis
Rheum_Arthritis subClassOf Juv_Disease (c: 0.625)

Poly_Juv_Rheum_Arthritis
Poly_Juv_Rheum_Arthritis subClassOf Nothing (c: 0.4688)

NEW MAPPING ENTAILENTS (O1 → O2)

Juvenile_Arthritis
JIA1:Juvenile_Arthritis subClassOf JIA2:Rheum_Arthritis (c: 0.8125)
JIA1:Juvenile_Arthritis subClassOf JIA2:Juv_Rheum_Arthritis (c: 0.5)

Multi_Joint_Disease
JIA1:Multi_Joint_Disease subClassOf JIA2:Juv_Disease (c: 0.625)
JIA1:Multi_Joint_Disease subClassOf JIA2:Disease (c: 1.0)

Rheumatoid_Arthritis
JIA1:Rheumatoid_Arthritis subClassOf JIA2:Juv_Disease (c: 0.625)
JIA1:Rheumatoid_Arthritis subClassOf JIA2:Disease (c: 1.0)

NEW MAPPING ENTAILENTS (O2 → O1)

Rheum_Arthritis
JIA2:Rheum_Arthritis subClassOf JIA1:Disease (c: 1.0)

Juv_Rheum_Arthritis
JIA2:Juv_Rheum_Arthritis subClassOf JIA1:Disease (c: 1.0)

Systemic_Juv_Rheum_Arthritis
JIA2:Systemic_Juv_Rheum_Arthritis subClassOf JIA1:Rheumatoid_Arthritis (c: 0.8125)

Figure 4.3: GUI for visualising new entailments in ContentMap

4.2.2.3. Giving explanation to errors

The development of techniques to help users understand the entailments form the deductive difference and subsequently select the sets of intended and unintended entailments is especially challenging. Thus, support is required to explain why the new entailments that hold in $\mathcal{U}$ do not hold in $\mathcal{O}_1$, $\mathcal{O}_2$ and $\mathcal{M}$ alone.

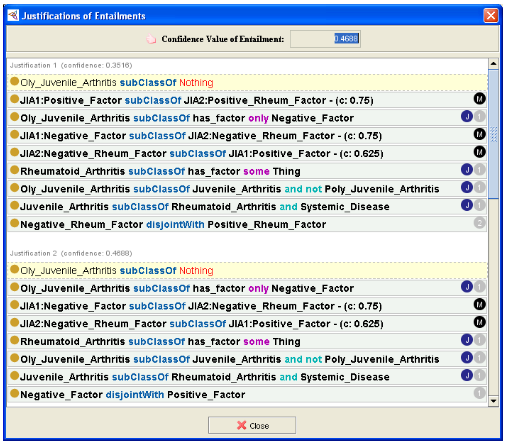
Some entailments in the relevant (approximate) differences may be intended, while others may reveal potential errors in the merged ontology $\mathcal{U}$. Step 7 therefore involves selecting entailments that: *(i)* are intended and should be entailed in $\mathcal{U}$ (written $\Im^+$ in Table 4.3), and *(ii)* are unintended and should not be entailed by $\mathcal{U}$ (written $\Im^-$).

In our example, the entailments $(\mathsf{Positive\_Factor} \sqsubseteq \bot)$ and $(\mathsf{Rheum\_Arthritis} \sqsubseteq \mathsf{Juv\_Disease})$ in $\mathsf{diff}_{\mathbf{S}_1}(\mathcal{O}_1, \mathcal{U})$ and $\mathsf{diff}_{\mathbf{S}_2}(\mathcal{O}_2, \mathcal{U})$ respectively, are unintented and should belong to $\Im^-$. In contrast, the entailed mapping $(1{:}\mathsf{Rheum\_Arthritis} \sqsubseteq 2{:}\mathsf{Disease})$ is intended and should be included in $\Im^+$.

ContentMap has also reused the notion of justification (see Definition 3.1). Note that, $\mathsf{Just}(\alpha, \mathcal{O})$ represents the set of all justifications for α in $\mathcal{O}$. Figure 4.4 shows the GUI to represent all justifications for the entailment $\mathsf{Oly\_Juvenile\_Arthritis} \sqsubseteq \bot$. Like in ContentCVS , the operation provided by Protégé 4 [HPS08a] is extended. On the one hand, ContentMap uses the optimisation from [SQJH08]. On the other hand, new graphic characteristics are added, so that the GUI indicates whether the axioms in the justifications come from $\mathcal{O}_1$, $\mathcal{O}_2$, or the mappings $\mathcal{M}$ (marked with '1', '2' and 'M' respectively). Additionally, axioms shared by all justifications are marked with 'J'.

4.2.2.4. Ordering and suggesting entailments

The potentially large number of relevant entailments may overwhelm the user. Thus, ContentMap organizes the entailments within a hierarchy, and perform suggestions about which entailments to include in $\Im^+$ and $\Im^-$.



Confidence Value of Entailment: 0.4688

Justification 1 (confidence: 0.3516)

- Oly_Juvenile_Arthritis subClassOf Nothing
- JIA1:Positive_Factor subClassOf JIA2:Positive_Rheum_Factor - (c: 0.75) M
- Oly_Juvenile_Arthritis subClassOf has_factor only Negative_Factor J 1
- JIA1:Negative_Factor subClassOf JIA2:Negative_Rheum_Factor - (c: 0.75) M
- JIA2:Negative_Rheum_Factor subClassOf JIA1:Positive_Factor - (c: 0.625) M
- Rheumatoid_Arthritis subClassOf has_factor some Thing J 1
- Oly_Juvenile_Arthritis subClassOf Juvenile_Arthritis and not Poly_Juvenile_Arthritis J 1
- Juvenile_Arthritis subClassOf Rheumatoid_Arthritis and Systemic_Disease J 1
- Negative_Rheum_Factor disjointWith Positive_Rheum_Factor 2

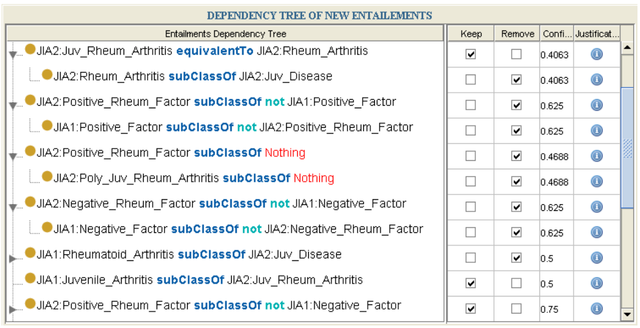
Justification 2 (confidence: 0.4688)

- Oly_Juvenile_Arthritis subClassOf Nothing
- Oly_Juvenile_Arthritis subClassOf has_factor only Negative_Factor J 1
- JIA1:Negative_Factor subClassOf JIA2:Negative_Rheum_Factor - (c: 0.75) M
- JIA2:Negative_Rheum_Factor subClassOf JIA1:Positive_Factor - (c: 0.625) M
- Rheumatoid_Arthritis subClassOf has_factor some Thing J 1
- Oly_Juvenile_Arthritis subClassOf Juvenile_Arthritis and not Poly_Juvenile_Arthritis J 1
- Juvenile_Arthritis subClassOf Rheumatoid_Arthritis and Systemic_Disease J 1
- Negative_Factor disjointWith Positive_Factor 1

Figure 4.4: Justifications for entailment Oly_Juvenile_Arthritis $\sqsubseteq \bot$

Like in ContentCVS (Section 4.2.2.4), ContentMap exploits the dependencies between entailments to organise the way in which they are presented (see Figure 4.5). Intuitively an entailment β depends on α in $\mathcal{O}$ if, whenever α is invalidated by removing a set of axioms from $\mathcal{O}$, then β is also invalidated. This implies that if $\alpha \in \Im^-$, then β should also be included in $\Im^-$ (and never in $\Im^+$). Thus, we have reused the dependency order $\rhd^-$ from Definition 3.6.

In order to suggest to the user which entailments to include in $\Im^+$ and $\Im^-$, ContentMap exploits both the dependencies between entailments and the confidence values for each explicit mapping. We use the confidence values in the mappings to compute confidence values in the entailments from each of the obtained differences: we consider our confidence in an entailment to be equal to the maximum confidence we have in one of its justifications, and our confidence in a justification to be the product of our confidences in each of the axioms it contains. We formalise this in the following



Entailments Dependency Tree	Keep	Remove	Confi...	Justificat...
JIA2:Juv_Rheum_Arthritis equivalentTo JIA2:Rheum_Arthritis	☑	☐	0.4063	ⓘ
JIA2:Rheum_Arthritis subClassOf JIA2:Juv_Disease	☐	☑	0.4063	ⓘ
JIA2:Positive_Rheum_Factor subClassOf not JIA1:Positive_Factor	☐	☑	0.625	ⓘ
JIA1:Positive_Factor subClassOf not JIA2:Positive_Rheum_Factor	☐	☑	0.625	ⓘ
JIA2:Positive_Rheum_Factor subClassOf Nothing	☐	☑	0.4688	ⓘ
JIA2:Poly_Juv_Rheum_Arthritis subClassOf Nothing	☐	☑	0.4688	ⓘ
JIA2:Negative_Rheum_Factor subClassOf not JIA1:Negative_Factor	☐	☑	0.625	ⓘ
JIA1:Negative_Factor subClassOf not JIA2:Negative_Rheum_Factor	☐	☑	0.625	ⓘ
JIA1:Rheumatoid_Arthritis subClassOf JIA2:Juv_Disease	☐	☑	0.5	ⓘ
JIA1:Juvenile_Arthritis subClassOf JIA2:Juv_Rheum_Arthritis	☑	☐	0.5	ⓘ
JIA2:Positive_Rheum_Factor subClassOf not JIA1:Negative_Factor	☑	☐	0.75	ⓘ

Figure 4.5: GUI for visualising the dependency relationship

Algorithm 2 Heuristic suggestions for $\Im^+$, $\Im^-$

```
Input: $\Lambda = \mathsf{diff}^{\approx}_{\mathbf{S}_1}(\mathcal{O}_1,\mathcal{U}) \cup \mathsf{diff}^{\approx}_{\mathbf{S}_2}(\mathcal{O}_2,\mathcal{U}) \cup \mathsf{mdiff}^{\approx}_{\mathbf{S}_1,\mathbf{S}_2}(\mathcal{M},\mathcal{U})$ as in Table 4.3
$\tau_{del}, \tau_{add}$: real values with $0 < \tau_{del} \leq \tau_{add} \leq 1$
$\rhd^-$: dependency relation between axioms in $\Lambda$ w.r.t. $\mathcal{U}$
Output: $\Im^+, \Im^-$: entailments suggested to be hold or not
 1: $\Im^+, \Im^- := \emptyset$
 2: for each $\alpha \in \Lambda$ do
 3:     if $\alpha$ of the form $A \sqsubseteq \bot$ for $A$ an atomic concept then
 4:         $\Im^- := \Im^- \cup \{\alpha\}$
 5:         for each $\beta \in \Lambda$ such that $\alpha \rhd^- \beta$ do $\Im^- := \Im^- \cup \{\beta\}$
 6:     end if
 7:     $\mathsf{conf}(\alpha) :=$ confidence of $\alpha$ as in Definition 4.3
 8:     if $\mathsf{conf}(\alpha) \leq \tau_{del}$ then $\Im^- := \Im^- \cup \{\alpha\}$
 9:     if $\mathsf{conf}(\alpha) \geq \tau_{add}$ and $\alpha \notin \Im^-$ then $\Im^+ := \Im^+ \cup \{\alpha\}$
10: end for
11: return $\Im^+, \Im^-$
```

definition, in which we extend $\mathsf{conf}()$ to arbitrary (explicit) axioms and justifications. We assume that we have complete confidence in an axiom (i.e., $\mathsf{conf}(\alpha) = 1$) if it is not a mapping.

Definition 4.3 (Confidence in an Entailment). *Let $\mathcal{O}$ be an ontology, α an axiom in $\mathcal{O}$ that is not annotated with a confidence value, $\mathcal{J}$ a justification, and β an entailment s.t. $\mathcal{O} \models \beta$. We define* $\mathsf{conf}(\alpha) = 1$*, and* $\mathsf{conf}(\mathcal{J})$ *and* $\mathsf{conf}(\beta)$ *as follows:*

$$\mathsf{conf}(\mathcal{J}) = \prod_{\gamma \in \mathcal{J}} \mathsf{conf}(\gamma) \quad \textit{and} \quad \mathsf{conf}(\beta) = \max\Big(\bigcup_{\mathcal{J} \in \mathsf{Just}(\beta,\mathcal{O})} \mathsf{conf}(\mathcal{J})\Big) \tag{4.2}$$

The following proposition shows that the confidence values in Definition 4.3 are well-behaved w.r.t. Definition 3.6: if β depends on α then the confidence in β is smaller than the confidence in α.

Proposition 4.1. *If* $\alpha \rhd^- \beta$*, then* $\mathsf{conf}(\beta) \leq \mathsf{conf}(\alpha)$

Proof. if $\alpha \rhd^- \beta$, then for each $\mathcal{J}_\beta \in \mathsf{Just}(\beta,\mathcal{U})$ there must exist a $\mathcal{J}_\alpha \in \mathsf{Just}(\alpha,\mathcal{U})$ s.t. $\mathcal{J}_\alpha \subseteq \mathcal{J}_\beta$. By definition of $\mathsf{conf}(\mathcal{J})$ (see Definition 4.3), it is immediate to see that $\mathsf{conf}(\mathcal{J}_\beta) \leq \mathsf{conf}(\mathcal{J}_\alpha)$. Thus, for each $\mathcal{J}_\beta \in \mathsf{Just}(\beta,\mathcal{U})$ there must exist a $\mathcal{J}_\alpha \in \mathsf{Just}(\alpha,\mathcal{U})$ s.t. $\mathsf{conf}(\mathcal{J}_\beta) \leq \mathsf{conf}(\mathcal{J}_\alpha)$. By Definition 4.3 it is immediate to see that $\mathsf{conf}(\beta) \leq \mathsf{conf}(\alpha)$. □

In order to compute initial suggestions for $\Im^-$ and $\Im^+$, ContentMap implements the heuristics of Algorithm 2, which are based on the ordering between entailments and their relative confidence. The algorithm accepts as input the entailments in the relevant differences, their dependencies, and two confidence thresholds τ_{del} and τ_{add}. For each input entailment α, Algorithm 2 uses $\rhd^-$, τ_{del} and τ_{add} to either include it in $\Im^+$ (i.e. α is intended), in $\Im^-$ (i.e. α is unintended) or in neither of them. An entailment α is included in $\Im^-$ if it reveals a contradiction (Line 3), depends on a contradiction (Line 5), or if its confidence according to Definition 4.3 is lower than τ_{del} (Line 8). In contrast, α is included in $\Im^+$ if it is not contained in $\Im^-$ and its confidence is higher than the threshold τ_{add} (Line 9).

4.2.2.5. Repair plans generation

If the user has selected one or more unintended entailments (i.e., $\Im^- \neq \emptyset$), then $\mathcal{U}$ clearly contains errors. These errors can always be repaired by removing axioms from $\mathcal{U}$—in the limit, removing *all* the axioms from $\mathcal{U}$ will eliminate all entailments. However, any removal of axioms should also respect $\Im^+$, the user's selection of intended entailments, i.e., the entailments in $\Im^+$ should still hold after any removal of axioms.

As previously mentioned, errors could be due to erroneous mappings, to inherently conflicting knowledge in the two ontologies, or to some combination of both. Repairing such errors might, therefore, require the removal of axioms from $\mathcal{M}$, $\mathcal{O}_1$ and $\mathcal{O}_2$. The user may, however, have a (principled or pragmatic) preference regarding the source of axioms to be removed; e.g., $\mathcal{M}$ might be considered the most likely source of errors, or it may be impossible to change (one or both of) $\mathcal{O}_1$ and $\mathcal{O}_2$. We adapt, from Defintion 3.7, the notion of a *repair plan* (or *plan*) used for ContentCVS. A plan is defined for an ontology $\mathcal{O}$ w.r.t. the (un-) intended entailments $\Im^-$ and $\Im^+$, and a subset $\mathcal{O}^-$ of $\mathcal{O}$, where the axioms in $\mathcal{O}^-$ are those that are allowed to be removed from $\mathcal{O}$. Note that current plans, unlike ContentCVS plans, only involve axioms to be deleted (there is not information loss). Thus, such plans are simply a subset of the axioms in $\mathcal{O}^-$ whose removal from $\mathcal{O}$ both eliminates the entailments in $\Im^-$ and preserves those in $\Im^+$. In general, there may be zero or more of such plans.

Definition 4.4. *Let $\mathcal{O}$, $\Im^+$, $\Im^-$, and $\mathcal{O}^-$ be finite sets of axioms such that $\mathcal{O}^- \subseteq \mathcal{O}$, $\mathcal{O} \models \Im^-$, $\mathcal{O} \models \Im^+$, and $\Im^+ \cap \Im^- = \emptyset$.*

A repair plan for $\mathcal{O}$ given $\mathcal{O}^-$, $\Im^+$ and $\Im^-$ is a set $\mathcal{P} \subseteq \mathcal{O}^-$ such that: 1) $(\mathcal{O} \setminus \mathcal{P}) \models \alpha$ *for each* $\alpha \in \Im^+$, *and* 2) $(\mathcal{O} \setminus \mathcal{P}) \not\models \beta$ *for each* $\beta \in \Im^-$. *A plan $\mathcal{P}$ is minimal if there is no $\mathcal{P}_1$ such that $\mathcal{P}_1 \subset \mathcal{P}$.*

According to Definition 4.4, plans are computed according to three kinds of requirements: the errors to fix ($\Im^-$), the entailments to preserve ($\Im^+$), and the set of axioms that a plan could potentially remove ($\mathcal{O}^-$). In our setting, these are selected by the user in Steps 7, 8 from Table 4.3. ContentMap implements an optimized algorithm following Definition 4.4.

In our example, if Peter imports $\mathcal{O}_1$ and $\mathcal{O}_2$ from the Web, and if he is not willing to copy and modify them (e.g., if he wants to always import the latest versions), then plans should only remove axioms from the mappings (i.e., $\mathcal{O}^- = \mathcal{M}$). However, the descriptions of Polyarticular Juvenile Rheumatoid Arthritis in $\mathcal{O}_1$ and $\mathcal{O}_2$ are inherently in contradiction and hence, if $\mathcal{O}^- = \mathcal{M}$, the corresponding error cannot be fixed without deleting reasonable mappings. In this case a more sensible choice would be to repair one or both of $\mathcal{O}_1$ and $\mathcal{O}_2$.

Note that conflicting choices in $\Im^+$ and $\Im^-$ may make it impossible to find any plan. Some of these conflicts can be detected using the dependency relation $\rhd^-$, as shown in the following proposition:

Proposition 4.2. *Let $\mathcal{O}$, $\Im^+$, $\Im^-$ and $\mathcal{O}^-$ be as in Definition 4.4 and let $\mathcal{O} \models \alpha, \beta$. If $\alpha \rhd^- \beta$ w.r.t. $\mathcal{O}$, $\alpha \in \Im^-$ and $\beta \in \Im^+$, then no plan exists.*

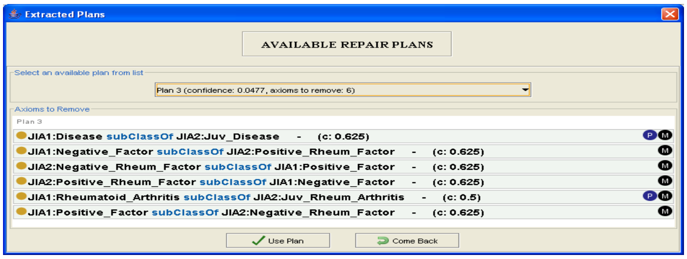
Proof. Assume that $\alpha \in \Im^-$, $\beta \in \Im^+$, and that there is a plan $\mathcal{P}$ for $\mathcal{O}$ given $\mathcal{O}^-$, $\Im^+$ and $\Im^-$. From the definition of a plan, $\mathcal{O}' = \mathcal{O} \backslash \mathcal{P}$ is s.t. $\mathcal{O}' \not\models \alpha$, and from Proposition 3.1, $\mathcal{O} \not\models \beta$. However, since $\beta \in \Im^+$, we also have that $\mathcal{O}' \models \beta$, contradicting our assumption that $\mathcal{P}$ is a plan. □

4.2.2.6. Plan selection support

There may also be particular selections of $\mathcal{O}^-$, $\Im^+$ and $\Im^-$ for which the number of minimal plans is very large. In this case, selecting the most suitable minimal plan becomes difficult for users.

In order to assist users in the selection of a minimal plan (Steps 10-11), ContentMap implements a number of heuristics based on the confidence of the mappings in the plans and the total number of involved axioms in the plan. The definition of confidence is extended to a plan $\mathcal{P}$ as in Equation 4.3. The selection of low confidence plans is desirable since it involves the removal of low confidence mappings.

$$\mathsf{conf}(\mathcal{P}) = \prod_{\alpha \in \mathcal{P}} \mathsf{conf}(\alpha) \tag{4.3}$$


Extracted Plans

AVAILABLE REPAIR PLANS

Select an available plan from list

Plan 3 (confidence: 0.0477, axioms to remove: 6)

Axioms to Remove

Plan 3

```
JIA1:Disease subClassOf JIA2:Juv_Disease   -   (c: 0.625)                                    P M
JIA1:Negative_Factor subClassOf JIA2:Positive_Rheum_Factor   -   (c: 0.625)                    M
JIA2:Negative_Rheum_Factor subClassOf JIA1:Positive_Factor   -   (c: 0.625)                    M
JIA2:Positive_Rheum_Factor subClassOf JIA1:Negative_Factor   -   (c: 0.625)                    M
JIA1:Rheumatoid_Arthritis subClassOf JIA2:Juv_Rheum_Arthritis   -   (c: 0.5)                 P M
JIA1:Positive_Factor subClassOf JIA2:Negative_Rheum_Factor   -   (c: 0.625)                    M
```

Figure 4.6: Selection of plans in ContentMap

The GUI in ContentMap for displaying and selecting minimal plans is shown in Figure 4.6. The confidence in the plan is given in the upper part of the figure. The axioms are marked according to their provenance (in the figure, they are all mappings and hence are marked with ‘M’). Axioms that occur in all minimal plans are additionally marked with a ‘P’.

4.2.3. Evaluation

The evaluation concerns three important aspects of our approach: *(1) Performance:* do the algorithms implemented in ContentMap run in a reasonable time?. *(2) Usability:* is a suitable amount of information presented to the user during each step of the integration process? *(3) Quality of the ontologies and mappings after the integration*: does our approach help to both resolve inherent disagreements between the ontologies and improve the quality of the mappings automatically generated by existing tools?.

In our experiments, we used a suite of four ontologies adapted from the 2004 EON Ontology Alignment Contest[4] which describe the domain of bibliographic references. These ontologies have been developed independently by INRIA ($\mathcal{O}_{\mathsf{INR}}$), MIT ($\mathcal{O}_{\mathsf{MIT}}$), UMBC ($\mathcal{O}_{\mathsf{UMBC}}$) and AIFB Karlsruhe ($\mathcal{O}_{\mathsf{AIFB}}$) respectively. Their sizes vary from 58 classes, 46 object properties, 26 data properties and 235 axioms in $\mathcal{O}_{\mathsf{AIFB}}$ to 18 classes, 12 object properties, 19 data properties and 96 axioms in $\mathcal{O}_{\mathsf{UMBC}}$. Even if small, all these ontologies are fairly expressive ($\mathcal{O}_{\mathsf{INR}}$ and $\mathcal{O}_{\mathsf{MIT}}$ are expressible in the DL $\mathcal{ALCHQ}(\mathcal{D})$, $\mathcal{O}_{\mathsf{UMBC}}$ in $\mathcal{ALCIN}(\mathcal{D})$ and $\mathcal{O}_{\mathsf{AIFB}}$ in $\mathcal{ALCI}(\mathcal{D})$ respectively). Their average classification time is less than a second when using Pellet 1,5.

In the 2004 EON contest, $\mathcal{O}_{\mathsf{INR}}$ was used as reference ontology, and the other ontologies were independently aligned with it using each of the competing tools. For evaluation, a manually produced gold standard containing the agreed-upon, correct mappings was provided for each pair of aligned ontologies. The gold standards for $\mathcal{O}_{\mathsf{MIT}}$, $\mathcal{O}_{\mathsf{UMBC}}$ and $\mathcal{O}_{\mathsf{AIFB}}$ contain 119, 83 and 98 mappings respectively. The experiments are organised in two parts, which we specify next, and were performed on a laptop with a 1,82 GHz processor and 3GB of RAM.

4.2.3.1. Repair of ontologies using gold standard

First, we have evaluated the semantic consequences of integrating each of $\mathcal{O}_{\mathsf{MIT}}$, $\mathcal{O}_{\mathsf{UMBC}}$ and $\mathcal{O}_{\mathsf{AIFB}}$ with $\mathcal{O}_{\mathsf{INR}}$ using the corresponding gold standard to detect inherent disagreements between them. In each case, we have applied our method from Table 4.3, with the mappings in Step 2 being the corresponding gold standard. We have used both the smallest and the largest approximations of the deductive differences implemented in ContentMap (see Section 4.2.2.2), and obtained the following results.

1. Alignment $\mathcal{O}_{\mathsf{MIT}}$, $\mathcal{O}_{\mathsf{INR}}$: for the smallest (resp. the largest) approximation, there were 3 (resp. 33) new entailments in $\mathcal{O}_{\mathsf{MIT}}$, 13 (resp. 85) in $\mathcal{O}_{\mathsf{INR}}$ and 35 (resp. 189) new mappings.
2. Alignment $\mathcal{O}_{\mathsf{UMBC}}$, $\mathcal{O}_{\mathsf{INR}}$: for the smallest (largest) approximation, there were 2 (7) new entailments in $\mathcal{O}_{\mathsf{UMBC}}$, 10 (300) in $\mathcal{O}_{\mathsf{INR}}$ and 28 (176) new mappings.
3. Alignment $\mathcal{O}_{\mathsf{AIFB}}$, $\mathcal{O}_{\mathsf{INR}}$:for the smallest (largest) approximation, there were 4 (37) new entailments in $\mathcal{O}_{\mathsf{AIFB}}$, 2 (19) in $\mathcal{O}_{\mathsf{INR}}$ and 46 (140) new mappings.

[4] `http://oaei.ontologymatching.org/2004/Contest/`

Note that the number of new entailments (and hence the amount of information that ContentMap shows to users) largely depends on the selected approximation.

In all cases we found a significant number of obviously unintended inferences due to inherent disagreements between the ontologies being integrated. For example, when integrating $\mathcal{O}_{\mathsf{AIFB}}$ and $\mathcal{O}_{\mathsf{INR}}$, our ContentMap detected a total of 34 newly unsatisfiable concepts in the two ontologies. The large number of (rather complex) justifications for some of these entailments makes manual repair almost unfeasible. When computing the corresponding plans, we found that the problem was originated by the ranges of two datatype properties, which were incompatible; hence, the modification of two axioms sufficed to fix all the errors at once. In most cases, a significant number of obviously unintended new subsumptions were also detected. For example, merging $\mathcal{O}_{\mathsf{MIT}}$ and $\mathcal{O}_{\mathsf{INR}}$ resulted in the new subsumptions $\mathsf{TechnicalReport} \sqsubseteq \mathsf{Date}$ and $\mathsf{TechnicalReport} \sqsubseteq \exists \mathsf{date.Reference}$. Again, the obtained plans revealed the origin of the problem, see axioms 4.4-4.7.

$$\mathcal{O}_{\mathsf{INR}}{:}\mathsf{year} \sqsubseteq \mathcal{O}_{\mathsf{MIT}}{:}\mathsf{hasYear} \tag{4.4}$$

$$\mathcal{O}_{\mathsf{MIT}}{:}\mathsf{TechnicalReport} \sqsubseteq \geqslant \mathcal{O}_{\mathsf{MIT}}{:}hasYear\ 1.Literal \tag{4.5}$$

$$\mathcal{O}_{\mathsf{INR}}{:}\mathsf{TechnicalReport} \sqsubseteq \mathcal{O}_{\mathsf{MIT}}{:}\mathsf{Technicalreport} \tag{4.6}$$

$$\mathcal{O}_{\mathsf{INR}}{:}\mathsf{year}\ \ \mathsf{hasDomain}\ \ \mathcal{O}_{\mathsf{INR}}{:}\mathsf{Date} \tag{4.7}$$

4.2.3.2. Synthetic repair of automatically generated mappings

Having repaired the test ontologies using the gold standard mappings, we used ContentMap to automatically detect and repair errors resulting from the generation of new mappings using the mapping tools OLA [EV04, KEV07] , AROMA [Dav08, Dav09] and CIDER [GM08, GdM09]. Our goal was twofold: first, to show that the algorithms in ContentMap are practical; second, to show that ContentMap can be used to automatically detect and repair errors, as well as to improve the quality of automatically generated mappings.

For each pair of test ontologies, and for various sets of automatically generated mappings, we have performed the synthetic experiments in Table 4.4, which closely follow our proposed method from Table 4.3. In contrast to Table 4.3, however, the repair of errors is performed in two stages: first, the obvious errors are repaired (i.e. unsatisfiable concepts); then, those entailments that ContentMap found unintended given a threshold τ_{del}. For these experiments we have used the smallest approximation of the deductive difference available in ContentMap . The roman numbers in Table 4.4 refer to measurements that are stored during the experiment.

Obtained results are given in Tables 4.5 and 4.6. Next, we summarise the main conclusions drawn from these experiments. First, from a computational point of view, the main bottleneck is the computation of all the justifications for the entailments of interest. Once the justifications have been computed, the time needed for computing the plans is relatively low. Hence, it is important to investigate optimisations for computing all justifications as in [SQJH08].

Table 4.4: Synthetic experiments

```
Input: $\mathcal{O}_1, \mathcal{O}_2$: ontologies with $\mathsf{Sig}(\mathcal{O}_1) = \mathbf{S}_1, \mathsf{Sig}(\mathcal{O}_2) = \mathbf{S}_2$ and $\mathbf{S}_1 \cap \mathbf{S}_2 = \emptyset$
$\mathcal{M}$: automatically-generated mappings between $\mathbf{S}_1$ and $\mathbf{S}_2$, $\mathcal{M}_{\mathcal{G}}$: gold standard
1: Filter $\mathcal{M}$ given a confidence threshold $\tau$ (I, II)
2: Compute precision (III) and recall (IV) of $\mathcal{M}$ with respect to $\mathcal{M}_{\mathcal{G}}$
3: Compute $\mathsf{diff}^{\approx}_{\mathbf{S}_1}(\mathcal{O}_1,\mathcal{U})$, $\mathsf{diff}^{\approx}_{\mathbf{S}_2}(\mathcal{O}_2,\mathcal{U})$ and $\mathsf{mdiff}^{\approx}_{\mathbf{S}_1,\mathbf{S}_2}(\mathcal{M},\mathcal{U})$ (V, VI) for $\mathcal{U} := \mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}$
4: Stage 1: Deletion of unsatisfiability (IX)
5: Compute all minimal plans for $\mathcal{U}$ given $\mathcal{O}^- = \mathcal{M}$, $\Im^+ = \emptyset$, and $\Im^- :=$ new concept unsatisfiability entailments (XII).
6: Store number of plans (XIII) and extraction time (XIV).
7: Compute $\mathcal{O}'_1 := \mathcal{O}_1 \setminus \mathcal{P}, \mathcal{O}'_2 := \mathcal{O}_2 \setminus \mathcal{P}, \mathcal{M}' := \mathcal{M} \setminus \mathcal{P}$(XV), for $\mathcal{P}$ the best plan.
8: Compute precision (XVIII) and recall (XIX) of $\mathcal{M}'$ with respect to $\mathcal{M}_{\mathcal{G}}$
9: Stage 2: Deletion of suggested entailments (IX)
10: Compute $\Lambda = \mathsf{diff}^{\approx}_{\mathbf{S}_1}(\mathcal{O}'_1,\mathcal{U}') \cup \mathsf{diff}^{\approx}_{\mathbf{S}_2}(\mathcal{O}'_2,\mathcal{U}') \cup \mathsf{mdiff}^{\approx}_{\mathbf{S}_1,\mathbf{S}_2}(\mathcal{M}',\mathcal{U}')$ (V, VI)(XVI, XVII) for $\mathcal{U}' := \mathcal{O}'_1 \cup \mathcal{O}'_2 \cup \mathcal{M}'$
11: Compute dependency relation $\rhd^-$ over $\Lambda$
12: Store time of computing $\rhd^-$ (VIII) and number of roots for $\rhd^-$ (VII)
13: Compute all minimal plans for $\mathcal{U}'$ given $\mathcal{O}^- = \mathcal{M}'$, $\Im^+ = \emptyset$ and $\Im^-$ (XI):= as obtained in Algorithm 2 for a pre-fixed confidence threshold $\tau_del$ (X).
14: Store number of plans (XIII) and extraction time (XIV).
15: Compute $\mathcal{O}''_1 := \mathcal{O}'_1 \setminus \mathcal{P}, \mathcal{O}''_2 := \mathcal{O}'_2 \setminus \mathcal{P}, \mathcal{M}'' := \mathcal{M}' \setminus \mathcal{P}$ (XIV), for $\mathcal{P}$ the best plan.
16: Compute $\Lambda = \mathsf{diff}^{\approx}_{\mathbf{S}_1}(\mathcal{O}''_1,\mathcal{U}'') \cup \mathsf{diff}^{\approx}_{\mathbf{S}_2}(\mathcal{O}''_2,\mathcal{U}'') \cup \mathsf{mdiff}^{\approx}_{\mathbf{S}_1,\mathbf{S}_2}(\mathcal{M}'',\mathcal{U}'')$ (XVI, XVII) for $\mathcal{U}'' := \mathcal{O}''_1 \cup \mathcal{O}''_2 \cup \mathcal{M}''$
17: Compute precision (XVIII) and recall (XIX) of $\mathcal{M}''$ with respect to $\mathcal{M}_{\mathcal{G}}$
```

Second, the use of the dependencies relationship $\rhd^-$ significantly reduces the amount of information that the user would need to examine. Furthermore, errors can be grouped under the same root or entailment.

Third, the use of automatically generated mappings did result in the occurrence of a significant number of unintended entailments. For example, when aligning $\mathcal{O}_{\mathsf{AIFB}}$ and $\mathcal{O}_{\mathsf{INR}}$ using CIDER with confidence threshold $\tau = 0{,}1$ (notice that CIDER usually works with low confidence values), we found 55 new unsatisfiable concepts. After fixing these errors, ContentMap found 34 unintended subsumption relationships using Algorithm 2 for a threshold $\tau_1 = 0{,}3$. Moreover, we checked manually selected examples and found out that the heuristically detected unintended subsumptions were indeed errors; furthermore, these errors were mostly caused by incorrect mappings.

Fourth, the application of the plans resulted in the automatic correction of the identified errors, which resulted in an improvement in the precision of the mappings. The improvement in precision occurred in all cases and varied from 1 %-5 %, achieving a good balance with respect to the recall, which remained unchanged in most cases, although a decrease from 1 %-3 % was observed in a few isolated cases.

(a) Results for OLA

$\mathcal{O}_1 \sim \mathcal{O}_2$	Gold Standard			Tool Results								Repair Information						After Applying Plan				
				I	II	III	IV	V	VI	VII	VIII	IX	X	XI	XII	XIII	XIV	XV	XVI	XVII	XVIII	XIX
	$\mathcal{M}_\mathcal{G}$	diff$^\approx$	mdiff$^\approx$	τ	$\mathcal{M}$	Precision	Recall	diff$^\approx$	mdiff$^\approx$	$\triangleright$	$\mathcal{J}$t(s)	Stage	τ_{del}	$\Im^-$	$\bot$	$\mathbb{P}$	$\mathbb{P}$t(s)	$\mathcal{M}$	diff$^\approx$	mdiff$^\approx$	Precision.	Recall
$\mathcal{O}_{\mathsf{INR}} \sim \mathcal{O}_{\mathsf{AIFB}}$	98	6	46	0.7	90	0.86	0.79	44	5	-	-	$\bot$	-	-	44	4	496.5	89	16	57	0.87	0.79
					89	0.87	0.79	16	57	52	41.1	$\Im^-$	0.5	1	0	4	0.00	88	4	44	0.88	0.79
$\mathcal{O}_{\mathsf{INR}} \sim \mathcal{O}_{\mathsf{UMBC}}$	83	12	28	0.7	88	0.58	0.61	22	38	-	-	$\bot$	-	-	12	86	363.8	87	14	51	0.58	0.6
					87	0.58	0.6	14	51	60	869.5	$\Im^-$	0.55	16	0	30	0.42	80	2	26	0.63	0.60
$\mathcal{O}_{\mathsf{INR}} \sim \mathcal{O}_{\mathsf{MIT}}$	119	16	35	0.7	60	0.83	0.42	26	46	-	-	$\bot$	-	-	0	0	-	-	-	-	-	-
					60	0.83	0.42	26	46	29	26.4	$\Im^-$	0.55	20	0	8	0.04	59	2	16	0.85	0.42

(b) Results for CIDER

$\mathcal{O}_1 \sim \mathcal{O}_2$	Gold Standard			Tool Results								Repair Information						After Applying Plan				
				I	II	III	IV	V	VI	VII	VIII	IX	X	XI	XII	XIII	XIV	XV	XVI	XVII	XVIII	XIX
	$\mathcal{M}_\mathcal{G}$	diff$^\approx$	mdiff$^\approx$	τ	$\mathcal{M}$	Prec.	Recall	diff$^\approx$	mdiff$^\approx$	$\triangleright$	$\mathcal{J}$t(s)	Stage	τ_{del}	$\Im^-$	$\bot$	$\mathbb{P}$	$\mathbb{P}$t(s)	$\mathcal{M}$	diff$^\approx$	mdiff$^\approx$	Precision	Recall
$\mathcal{O}_{\mathsf{INR}} \sim \mathcal{O}_{\mathsf{AIFB}}$	98	6	46	0.1	136	0.6	0.84	57	16	-	-	$\bot$	-	-	55	47	746	135	14	68	0.6	0.83
					135	0.6	0.8	14	68	68	17.8	$\Im^-$	0.3	34	0	10	0.04	124	2	34	0.65	0.83
				0.2	118	0.7	0.84	39	33	-	-	$\bot$	-	-	32	42	58.9	115	12	58	0.7	0.83
					115	0.7	0.83	12	58	56	18.5	$\Im^-$	0.3	14	0	6	0.01	110	4	40	0.73	0.83
				0.3	98	0.83	0.83	25	33	-	-	$\bot$	-	-	19	16	27.1	96	10	52	0.84	0.83
					96	0.84	0.83	10	52	50	18.1	$\Im^-$	0.25	8	0	5	0.0	94	2	40	0.85	0.82
				0.4	96	0.84	0.83	20	27	-	-	$\bot$	-	-	19	16	26.4	94	3	46	0.86	0.83
					94	0.86	0.83	3	46	45	14.2	$\Im^-$	0.25	1	0	3	0.0	93	2	45	0.86	0.81
$\mathcal{O}_{\mathsf{INR}} \sim \mathcal{O}_{\mathsf{UMBC}}$	83	12	28	0.2	78	0.74	0.7	36	55	-	-	$\bot$	-	-	5	48	6.2	75	33	60	0.77	0.7
					75	0.77	0.7	33	60	39	162.4	$\Im^-$	0.15	2	0	12	0.0	74	11	37	0.78	0.7
				0.3	70	0.83	0.7	33	60	-	-	$\bot$	-	-	0	0	-	-	-	-	-	-
					70	0.83	0.7	33	60	39	167	$\Im^-$	0.15	2	0	12	0.01	69	11	37	0.84	0.7
$\mathcal{O}_{\mathsf{INR}} \sim \mathcal{O}_{\mathsf{MIT}}$	119	16	35	0.1	106	0.87	0.77	20	30	-	-	$\bot$	-	-	9	31	138.9	105	15	37	0.88	0.77
					105	0.88	0.77	15	37	35	15.9	$\Im^-$	0.1	2	0	11	0.0	104	13	37	0.88	0.77
				0.2	104	0.88	0.77	15	37	-	-	$\bot$	-	-	0	0	-	-	-	-	-	-
					104	0.88	0.77	15	37	35	15.6	$\Im^-$	0.1	2	0	11	0.0	103	13	37	0.89	0.77

Table 4.5: Obtained results for OLA and CIDER

$\mathcal{O}_1 \sim \mathcal{O}_2$	Gold Standard			Tool Results								Repair Information						After Applying Plan				
	$\mathcal{M}_{\mathcal{G}}$	diff$^{\approx}$	mdiff$^{\approx}$	I	II	III	IV	V	VI	VII	VIII	IX	X	XI	XII	XIII	XIV	XV	XVI	XVII	XVIII	XIX
				τ	$\mathcal{M}$	Precision	Recall	diff$^{\approx}$	mdiff$^{\approx}$	$\triangleright$	$\mathcal{J}$t(s)	Stage	τ_{del}	$\Im^-$	$\perp$	$\mathbb{P}$	$\mathbb{P}$t(s)	$\mathcal{M}$	diff$^{\approx}$	mdiff$^{\approx}$	Precision	Recall
$\mathcal{O}_{\mathsf{INR}} \sim \mathcal{O}_{\mathsf{AIFB}}$	98	6	46	0.0	124	0.65	0.82	53	19	-	-	$\perp$	-	-	51	5	329	122	13	66	0.65	0.8
					122	0.64	0.8	13	66	61	22.4	$\Im^-$	0.5	14	0	10	0.01	118	2	50	0.66	0.8
				0.5	94	0.82	0.79	44	19	-	-	$\perp$	-	-	43	47	202	93	9	57	0.83	0.79
					93	0.83	0.79	9	57	54	20.4	$\Im^-$	0.45	3	0	15	0.00	91	2	47	0.85	0.79
$\mathcal{O}_{\mathsf{INR}} \sim \mathcal{O}_{\mathsf{UMBC}}$	83	12	28	0.0	70	0.79	0.66	29	2	-	-	$\perp$	-	-	39	20	437	68	11	33	0.79	0.65
					68	0.79	0.65	11	33	32	10.2	$\Im^-$	0.0	0	0	0	-	-	-	-	-	-
$\mathcal{O}_{\mathsf{INR}} \sim \mathcal{O}_{\mathsf{MIT}}$	119	16	35	0.0	98	0.9	0.74	21	15	-	-	$\perp$	-	-	19	35	261.2	95	12	36	0.93	0.74
					95	0.93	0.74	12	36	29	38.8	$\Im^-$	0.3	11	0	10	0.01	94	2	25	0.94	0.74
				0.2	96	0.9	0.72	21	15	-	-	$\perp$	-	-	19	35	252.2	93	12	36	0.93	0.72
					93	0.93	0.72	12	36	29	35	$\Im^-$	0.3	11	0	10	0.01	92	2	25	0.94	0.72
				0.3	94	0.92	0.72	17	13	-	-	$\perp$	-	-	17	46	247.7	92	12	37	0.94	0.72
					92	0.94	0.72	12	36	30	36.6	$\Im^-$	0.3	11	0	10	0.01	91	2	26	0.95	0.72
				0.5	92	0.94	0.72	13	18	-	-	$\perp$	-	-	13	46	159	90	4	31	0.96	0.72
					90	0.96	0.72	4	31	30	52.4	$\Im^-$	0.0	0	0	0	-	-	-	-	-	-
				0.7	80	0.95	0.64	8	22	-	-	$\perp$	-	-	7	13	4.7	78	4	29	0.97	0.64
					78	0.97	0.64	4	29	28	19.3	$\Im^-$	0.0	0	0	0	-	-	-	-	-	-

Table 4.6: Obtained results for AROMA

4.2.4. Related work and conclusion

In the last few years, the problem of automatically generating mappings between ontologies has been extensively investigated. A comprehensive and up-to-date source of information about the topic (including papers, tools, ontologies for evaluation, etc.) can be found in [SE10]. The debugging and revision of mappings [MFRW00, MST07, MST09, JMSK09, MS09, MSvZ09] have also been treated in the literature. Chimaera [MFRW00] was one of the pioneering systems for ontology merging and diagnosis. The diagnosis process was based on a test suit to evaluate the completeness and the correcteness of the merged ontology. The research carried out in [MST07, MST09] has so far been focused on mappings represented using Distributed Description Logics (DDL) [ST09], and therefore they provide extra semantics for the management of mappings. In our approach, due to application purposes, we have reused the semantics of OWL 2 [CHM$^+$08] and adopted a much simpler representation of mappings. Approaches in [JMSK09, MS09] present heuristics to repair errors, however they do not consider other unintended entailments. In [MSvZ09] a web-based tool was proposed to support the user in the mapping evaluation. As ContentMap , this tool tries to involve the user within the evaluation process. However, it does not consider so far the general impact of the mappings (i.e., unintended entailments) but only provides suggestions to repair/avoid obvious unsatisfiability errors.

Our work is also related to the existing approaches for debugging and repairing inconsistencies in OWL ontologies (e.g. [SC03, KPSG06, HPS08b, JQH09]). From them we have borrowed the notion of *justification* and the implementation integrated within Protégé 4 [HPS08a]. However, in both of these lines of research, the detected and repaired errors are limited to unsatisfiable concepts and inconsistent ontologies.

We believe that our approach, when compared to existing work, presents a number of improvements. First, the entailments to be repaired are not restricted to obvious inconsistencies, but can include any unintended entailment. Second, users can customise the kinds of entailments to be taken into account when comparing the integrated ontology to the individual ones in order to detect errors (i.e., select the approximation of the deductive difference). Third, users can select not only which entailments should be invalidated, but also which ones should necessarily hold upon completion of the repair process. To this end, we provide a number of novel techniques for helping the user to select which entailments are (un)intended, such as the computation of the dependencies between entailments (the relation $\rhd^-$), confidence in entailments according to the confidence in the mappings, automatic suggestions, etc. Fourth, we provide efficient algorithms for computing *all* the repair plans and to help the user select the most suitable plan from amongst those computed. Fifth, when compared to existing work on debugging and repair in OWL, we provide a clear distinction between the ontologies being integrated and the mappings, and users can customise the ontologies from which the plans are allowed to delete axioms. Finally, we provide a fully-fledged editor and reasoning infrastructure integrated with Protégé 4.

Our approach, however, has a weak point when dealing with big ontologies and a high number of errors. As already discussed in Section 4.2.3 the main bottleneck is the computation of all the justifications for the entailments of interest. For example, the integration of FMA and NCI given a set of candidate mappings produces more than 40,000 unsatisfiable concepts. The extraction of all justifications for those errors in order to obtain the repair plans would be computationally prohibitive, even when only an small set of axioms are the cause of the errors.

Recent approaches have tried to design heuristics and provide efficient method to automatically discard conflicting mappings and solve most of the errors. Prominent examples are [JMSK09, MS09], which have been already introduced above, and [JRGHL09d, JRGHL10a], which belong to our research contribution and are described in Section 4.3. The approach in Section 4.3 applies a set of principles to assess the integration of large ontologies. These principles are not intended to produce an error-free output but a reduction in the number of errors. Then, a tool such as ContentMap can be applied in order to repair the remaining errors for which manual intervention is usually necessary.

4.3. Auditing the integration of large ontologies

UMLS Metathesaurus (UMLS-Meta) [Bod04], as already introduced in Section 2.1, represents the most comprehensive effort for integrating biomedical thesauri and ontologies through mappings. Currently, the integration of new sources in UMLS-Meta combines lexical algorithms [Aro01, MBB06, HGHC07, HGH$^+$09], expert assessment [Bod04] and auditing protocols [GPHC09]. In its 2009AA version, UMLS-Meta integrates more than one hundred thesauri and ontologies, including SNOMED CT, FMA, and NCI, and contains more than 6 million entities. UMLS-Meta provides a list with more than two million unique identifiers (CUIs). Each CUI can be associated to entities belonging to different sources. Pairs of entities from different sources with the same CUI are synonyms and hence can be represented as an equivalence mapping.

It has been noticed that UMLS-Meta, despite being carefully curated by domain experts, may contain errors [Cim98, CMP03, MBB09, MGHP09, JYJRBRS09b]. Current auditing techniques aimed at detecting potential errors mostly rely on the UMLS *semantic network* [McC89] — a “top level” semantic model grouping the entities from the UMLS-Meta sources into suitable *semantic categories*. Semantic categories are then organised into so-called *semantic groups* [BM03b]. For example “Heart” is associated with the semantic category “Body Part, Organ, or Organ Component” and the semantic group “Anatomy”. Errors can then be detected by identifying incompatibilities in the assignment of such semantic categories to entities in the sources. For example the UMLS-Meta “Globular Actin” has, among others, “Amino Acid, Peptide, or Protein” and “Cell Component” as semantic categories, which belong to two different semantic groups “Chemicals & Drugs” and “Anatomy” respectively.

In this Section we present two main contributions [JRGHL09d, JRGHL10a]. First, we provide empirical evidence suggesting that UMLS-Meta in its 2009AA version contains a significant number of logic-based errors, and we prove that the logic-based semantics of the ontology sources may be used to enhance current UMLS-Meta design and auditing methods in order to avoid such errors. Note that, we have used FMA, NCI and SNOMED CT as the UMLS-Meta ontology sources to conduct our experiments. Second, we propose general principles and specific logic-based techniques to effectively detect and repair such errors. Such principles and techniques are intended to support the assessment of the integration of large ontologies (e.g., FMA, SNOMED CT or NCI). As commented in Section 4.2.4, tools such as ContentMap must be complemented with automatic techniques in order to reduce the number of errors when integrating large ontologies.

Our empirical results are very encouraging and show the effectiveness of our techniques in practice. Furthermore, we believe that our novel techniques [JRCHB09, JRGHL09d, JRGHL10a] are complementary to current UMLS-Meta auditing methods [Cim98, CMP03, MBB09, MGHP09, GPHC09] and studying how they can be effectively combined constitutes an interesting direction for the auditing of UMLS-Meta and for the general assessment of ontology integration.

4.3.1. Logical representation of UMLS-Meta mappings

A *formal representation* of the mappings, as introduced in Section 4.1, is required in order to reason with the source ontologies and the corresponding mappings (e.g., UMLS-Meta). Therefore, the first step is to provide formal semantics to the mappings in UMLS-Meta.

We have processed the *MRCONSO* file from the UMLS-Meta distribution [oM10]. This file contains every entity in UMLS-Meta together with its *concept unique identifier* (CUI), its source vocabulary, its language, and other attributes not relevant for this work. Table 4.7 shows an excerpt from the rows in the *MRSCONSO* file associated to the CUI C0022417 (which represents the notion of "Joint") with source vocabulary FMA, SNOMED CT or NCI.

CUI	Language	Source	Entity
C0022417	ENG	FMA	Joint
			Set_of_joints
		SNOMED CT	Joint_structure
		NCI	Joint
			Articulation

Table 4.7: An excerpt from the MRCONSO file for "Joint"

It follows from Table 4.7 that the notion of "Joint" is shared by FMA, SNOMED CT and NCI. In particular, FMA contains the entities $Joint$ and Set_of_joints, NCI the entities $Articulation$ and $Joint$, and SNOMED only the entity $Joint\_structure$. All these entities have been annotated with the CUI C0022417 and therefore, according to UMLS-Meta's intended meaning, they are synonyms. Then, for each pair of entities e and e' from *different* sources and annotated with the same CUI, we have generated the OWL 2 mapping axiom EquivalentClasses$(e\ e')$. The axioms obtained

Mapped Ontologies	Generated Mappings

FMA $\sim$ NCI	EquivalentClasses(FMA:$Joint$ NCI:$Joint$)
	EquivalentClasses(FMA:$Joint$ NCI:$Articulation$)
	EquivalentClasses(FMA:Set_of_joints NCI:$Joint$)
	EquivalentClasses(FMA:Set_of_joints NCI:$Articulation$)

FMA $\sim$ SNOMED CT	EquivalentClasses(FMA:$Joint$ $SNOMED$:$Joint\_structure$)
	EquivalentClasses(FMA:Set_of_joints $SNOMED$:$Joint\_structure$)

SNOMED CT $\sim$ NCI	EquivalentClasses($SNOMED$:$Joint\_structure$ NCI:$Joint$)
	EquivalentClasses($SNOMED$:$Joint\_structure$ NCI:$Articulation$)

Table 4.8: Mappings between FMA, NCI and SNOMED CT. The prefixes "FMA:", "NCI:" and "$SNOMED$:" are shown explicitly to emphasise that each ontology source uses a different namespace to refer to its entities. However, for simplicity, we will often obviate these prefixes in the text.

for our example CUI are given in Table 4.8. We do not explicitly generate axioms involving entities from the same source because we are interpreting UMLS-Meta as a mapping theory, whose purpose is to integrate independently developed sources, rather than to model the domain (i.e., to add explicit content to each of the sources independently). Note, however, that the mappings from Table 4.8 do modify the contents of the ontology sources *implicitly* (for example, they imply that the entities $Articulation$ and $Joint$ from NCI are equivalent, even if they are not in the original source). In the following section, we argue that such implicit modifications of a source due only to the mappings are one of the main causes of logical errors.

Once UMLS-Meta mappings has a logic-based representation, we can enable logical reasoning over the union of the source ontologies and their respective UMLS-Meta mappings and obtain logical consequences that were not derivable from any of them in isolation. Our main hypothesis is that such logical consequences can be used to identify errors in the mappings (e.g., UMLS-Meta mappings) as well as to detect inherent incompatibilities between the source ontologies (e.g. FMA, NCI, SNOMED CT). To verify this hypothesis, we have identified three general principles, which are described in Section 4.3.2. and designed a number of logic-based techniques that follow those general principles (See Section 4.3.3).

4.3.2. Proposed principles

We have identified three general principles, which describe how logic-based techniques can be applied to the integration of two ontology sources $\mathcal{O}_1$ and $\mathcal{O}_2$ using a third mapping ontology $\mathcal{M}$.

1. The Conservativity Principle: Given an ontology source (say, $\mathcal{O}_1$) and the mappings $\mathcal{M}$, the union $\mathcal{O}_1 \cup \mathcal{M}$ should not introduce new semantic relationships between entities from $\mathcal{O}_1$.

The conservativity principle is based on the purpose of $\mathcal{M}$, which is to enable the interaction between $\mathcal{O}_1$ and $\mathcal{O}_2$, rather than to provide a new description of the do-

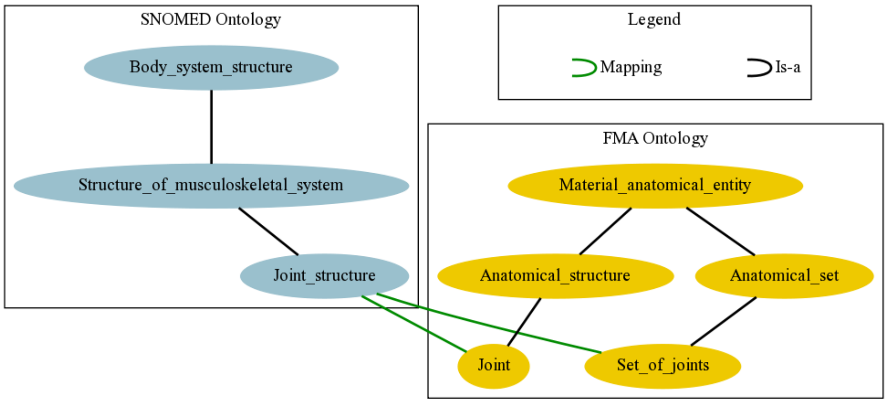


Figure 4.7: Conservativity principle violation

main. In the case of our previous example about "Joints", UMLS-Meta contains two mappings establishing the equivalence between the entity *Joint_structure* from SNOMED CT and the FMA entities *Joint* and *Set_of_joints* respectively. As a consequence, UMLS-Meta implies that *Joint* is also equivalent to *Set_of_joints*. However, in FMA *Joint* neither subsumes, nor it is subsumed by *Set_of_joints* (see Figure 4.7). The conservativity principle suggests that the obtained mappings are in conflict and (at least) one of them is likely to be incorrect.

2. The Consistency Principle: The ontology $\mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}$ should be consistent and all the entities in its vocabulary should be satisfiable [BCM+03].

According to the consistency principle, the integration of well-established ontologies should not introduce logical inconsistencies, which are clear manifestations of a design error. These may be due to either erroneous mappings or to inherent incompatibilities between the source ontologies. In any case, in order for the integrated ontology to be successfully used in an application, these errors should be repaired by modifying either the source ontologies or the mappings.

For example, as shown in Figure 4.8, UMLS-Meta maps the FMA concept *Protein* to the NCI concept *Protein*, and the FMA concept *Lymphokine* to the NCI concept *Therapeutic_Lymphokine*. In FMA, *Lymphokine* is a *Protein*, whereas in NCI *Therapeutic_Limphokine* is a *Pharmacologic_Substance*. Furthermore, *Pharmacologic_Substance* and *Protein* are disjoint in NCI and hence the union of NCI, FMA and UMLS-Meta would imply that *Lymphokine* and *Therapeutic_Limphokine* are unsatisfiable (i.e., there can be no instances of either entity).

3. The Locality Principle: If two entities e_1 and e_2 from ontologies $\mathcal{O}_1$ and $\mathcal{O}_2$ are correctly mapped, then the entities semantically related to e_1 in $\mathcal{O}_1$ are likely to be mapped to those semantically related to e_2 in $\mathcal{O}_2$.

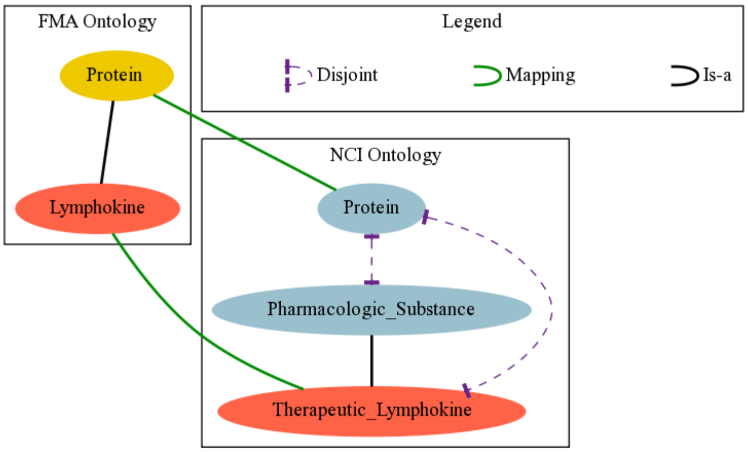


Figure 4.8: Consistency principle violation

If the locality principle does not hold, then the following situations can be identified: (1) $\mathcal{M}$ may be incomplete and new mappings should be discovered, (2) the definitions of both concepts in their respective ontologies may be different or incompatible, or (3) the mapping between e_1 and e_2 may be erroneous.

4.3.3. Implemented techniques

We next propose a collection of logic-based techniques based on each of these general principles. Our techniques exploit the following observations about UMLS-Meta ontology sources and mappings:

- **Ob1**: The OWL 2 ontology $\mathcal{M}$ that encodes the contents of UMLS-Meta only contains axioms of the form EquivalentClasses$(e_1\ e_2)$ where e_1 is only mentioned in $\mathcal{O}_1$ and e_2 is only mentioned in $\mathcal{O}_2$ (note that, as illustrated in Table 4.8, different ontology sources use different namespaces to refer to their entities). This observation is crucial to the design of techniques based on the conservativity principle.

- **Ob2**: UMLS-Meta ontology sources such as SNOMED CT, NCI and FMA contain both positive and negative information (e.g., if something is a "Protein", then it is *not* a "Drug"). Logical inconsistencies can arise due to the simultaneous presence (either explicit or implicit) of two conflicting statements containing positive and negative information, e.g., the statement in FMA that lymphokine is a kind of protein and the (implicit) statement that lymphokine is not a kind of protein. This observation is important to design techniques based on the consistency principle.

- **Ob3**: The entities described in UMLS-Meta ontology sources such as SNOMED CT, NCI and FMA are "loosely interconnected". Roughly speaking, this implies that the "meaning" of an entity in each of these ontologies only depends on a very small set of entities in the ontology that are "semantically related" to it. To formalise the notion of an entity being "semantically related" to another entity in an ontology, we use the logic-based ontology modularisation framework [GHKS08] already introduced in Section 2.2. This observation is important to design techniques based on either the consistency or the locality principles.

4.3.3.1. Implementing the conservativity principle

The conservativity principle can be directly expressed as the following *reasoning problem*. We say that an ontology source (say, $\mathcal{O}_1$) violates conservativity if there exists an OWL 2 axiom α such that $\mathcal{O}_1 \cup \mathcal{M}$ implies α, but $\mathcal{O}_1$ does not imply α. This problem is strongly related to the notion of *conservative extension* [GLW06, LWW07, GHKS08]. It is well-known that its computational complexity is very high even for lightweight ontology languages, and no practical algorithms currently exists. As we describe next, however, in the case of UMLS-Meta we can exploit Observation **Ob1** to significantly simplify the problem.

Let $\mathcal{M}$ contain only axioms of the form $\mathsf{EquivalentClasses}(e_1\ e_2)$ where e_1 is only mentioned in $\mathcal{O}_1$ and e_2 is only mentioned in $\mathcal{O}_2$. Then, $\mathcal{O}_1$ violates conservativity if and only if there exist axioms $\mathsf{EquivalentClasses}(e_1\ e_2)$ and $\mathsf{EquivalentClasses}(e'_1\ e_2)$ in $\mathcal{M}$, with e_1 and e'_1 different entities in $\mathcal{O}_1$, such that $\mathcal{O}_1$ alone does not imply the axiom $\mathsf{EquivalentClasses}(e_1\ e'_1)$. In such case the mappings $\mathsf{EquivalentClasses}(e_1\ e_2)$ and $\mathsf{EquivalentClasses}(e'_1\ e_2)$ are in conflict and one of them may be incorrect.

In our previous example (recall Figure 4.7), the mappings $\mathsf{EquivalentClasses}(Joint\ Joint\_structure)$ and $\mathsf{EquivalentClasses}(Set\_of\_joints\ \ Joint\_structure)$ between FMA and SNOMED CT are likely to be in conflict. In order to identify such conflicting mappings, it suffices to (syntactically) check in $\mathcal{M}$ whether two entities from one of the sources (e.g., $Joint$ and Set_of_joints from FMA) are mapped to the same entity in the other source (e.g., $Joint\_structure$ from SNOMED CT) and then check (semantically) whether these two entities were already equivalent with respect (only) to the former source. These checks can be performed efficiently in practice: the former is syntactic, and the latter involves a single semantic test using an ontology reasoner (e.g., Does FMA imply that $Joint$ and Set_of_joints are equivalent?).

4.3.3.2. Implementing the consistency principle

Similarly to the conservativity principle, the consistency principle can also be easily formulated as a reasoning problem. Let us denote with $\mathsf{sig}(\mathcal{O})$ the vocabulary of an ontology $\mathcal{O}$. We say that $\mathcal{O}_1$, $\mathcal{O}_2$ and $\mathcal{M}$ violate consistency if there is an entity $e \in \mathsf{sig}(\mathcal{O}_1) \cup \mathsf{sig}(\mathcal{O}_2)$ that is unsatisfiable with respect to $\mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}$ (i.e., $\mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}$ implies the axiom $\mathsf{EquivalentClasses}(e\ \ owl{:}Nothing)$).

The obvious way to check consistency violation and identify sets of conflicting mappings is to use an ontology reasoner to check the satisfiability of each entity in the vocabulary of $\mathcal{O}_1 \cup \mathcal{O}_2 \cup \mathcal{M}$ and then apply state-of-the-art ontology debugging techniques (see Section 3.1.2) or frameworks such as ContentMap (see Section 4.2) to identify and disambiguate conflicts. In our example from Figure 4.8, we could use a reasoner to identify that the concepts $Lymphokine$ and $Therapeutic\_Limphokine$ are unsatisfiable with respect to the integration of FMA and NCI via UMLS-Meta. Then, debugging techniques could be used to identify the mappings and axioms from the source ontologies responsible for these errors. This approach, however, as already commented in Section 4.2.4, could be computationally prohibitive when the ontologies to be integrated are large and the number of errors are likely to be high.

In our particular setting, unsatisfiable concepts can be caused by either erroneous mappings, or by inherent incompatibilities between the source ontologies. Our method relies on first identifying and disambiguating conflicting pairs of mappings (i.e., mappings that when occurring together make a concept unsatisfiable) and only then detecting and resolving incompatibilities between the sources.

Detecting conflicting mappings

We next exploit Observations **Ob2** and **Ob3** to define a simple heuristic technique, which is along the lines of those presented in [MST08, JMSK09, MS09]. The *disjointness-based inconsistency heuristic* is defined as follows. If e and e' from $\mathcal{O}_1$ are mapped to f and f' from $\mathcal{O}_2$ and $\mathcal{O}_1$ implies that e is subsumed by e', but $\mathcal{O}_2$ implies that f and f' are disjoint, then the consistency principle is violated (recall Figure 4.8). Note that the converse does not necessarily hold. This heuristic requires semantic tests to check whether e is subsumed by e' in $\mathcal{O}_1$ and whether f and f' are disjoint in $\mathcal{O}_2$. However, this heuristic requires only the classification of each of the source ontologies *independently*. Note that many reasoners can produce also the implicit disjointness relationships between their entities as an additional output of classification with only a relatively short additional delay. Furthermore, by Observation **Ob3** we can optimise even further and perform only the classification of the logic-based modules for the mapped entities in each of the source ontologies. Consider, for example, the mappings between FMA and SNOMED CT. The module in FMA for the entities that are mapped to SNOMED CT contains only 10,204 entities (out of 67,000), and the corresponding module in SNOMED CT contains only 15,428 entities (out of 300,000). That is, extracting and classifying the modules instead of classifying the source ontologies as a whole results in a considerable simplification. Finally, the set of conflicting mappings is obtained and therefore there is no need to apply expensive debugging techniques to retrieve the sets of axioms responsible for the inconsistency.

4.3.3.3. Implementing the locality principle

The conservativity and consistency principles allow us to identify pairs of mappings in UMLS-Meta that are in mutual conflict. However, since UMLS-Meta does not assign a confidence value to each mapping, it is not clear how to disambiguate these conflicts (e.g., how to decide whether to map $Joint\_structure$ to $Joint$ or to Set_of_joints). If there are too many conflicts, manual disambiguation becomes unfeasible. We propose to apply the locality principle in order to compute a confidence value for each conflicting mapping, which we can then exploit for (partially) automating the disambiguation process.

Recent works [TvOS09, TvOSW10] have also implemented disambiguation techniques exploiting the scope of the mapped entities. However, these techniques have been only based on the alignment of thesauri, thus the scope has been limited to the analysis of broader and narrower entities.

Computing confidence values

Assume that e from $\mathcal{O}_1$ is mapped via a mapping μ to f from $\mathcal{O}_2$. As already mentioned in Observation **Ob3**, the application of the locality principle relies on a well-known ontology modularisation framework (refer to [GHKS08] or Section 2.2). Therefore, if most of the entities in the module M_e^1 for e in $\mathcal{O}_1$ (i.e., those entities that are “semantically related” to e in $\mathcal{O}_1$) are also mapped to those in the module M_f^2 for f in $\mathcal{O}_2$, then we can assign a high confidence value to μ. Intuitively, such confidence values $\mathsf{conf}(\mu)$ can be obtained by computing the ratio between the number of entities in the modules which are mapped via UMLS-Meta, and the total number of entities in the modules:

$$\mathsf{conf}(\mu) = \frac{\mid \text{Mapped entities in } \mathsf{sig}(M_e^1) \mid + \mid \text{Mapped entities in } \mathsf{sig}(M_f^2) \mid}{\mid \mathsf{sig}(M_e^1) \mid + \mid \mathsf{sig}(M_f^2) \mid} \tag{4.8}$$

However, since the modules are of relatively small size, UMLS-Meta often does not contain enough mappings between the modules to obtain an accurate value. For example, UMLS-Meta contains the mapping between $Upper\_Extremity$ from NCI and Arm from FMA, but none of the entities, apart from the mentioned ones, in the module for $Upper\_Extremity$ in NCI is mapped to an entity in the module for Arm in FMA. To address this issue, we used the ISUB lexical matching algorithm [SSK05] to obtain additional lexical correspondences between entities in the modules. Algorithm 3 calculates a refined confidence value $\mathsf{conf_r}(\mu)$. Note that this new confidence value does not necessarily ranges from 0 to 1.

Disambiguating conservativity conflicts automatically.

Figure 4.9 depicts sets of conflicting mappings obtained using the conservativity principle together with the confidence values we have obtained for each of them. Consider the mappings between NCI and FMA on the left-hand-side of the figure. Let μ_1, μ_2 represent the mappings respectively connecting $Upper\_Extremity$ in NCI to $Upper\_limb$ and Arm in FMA, and let μ_3, μ_4 represent those relating Arm in NCI to $Upper\_Limb$ and Arm from FMA, respectively. We can identify the following four conflicts:

$$\kappa_1 = \{\mu_1, \mu_2\} \quad \kappa_2 = \{\mu_3, \mu_4\} \quad \kappa_3 = \{\mu_1, \mu_3\} \quad \kappa_4 = \{\mu_2, \mu_4\} \tag{4.9}$$

In order to disambiguate all the conflicts between two source ontologies, we need to remove one mapping per conflict in such a way that the result of adding their confidence values is minimised. This is a standard *diagnosis* problem, for which practical

Algorithm 3 Obtaining additional lexical correspondences

Input: M_e^1, M_f^2, $\mu := \mathsf{EquivalentClasses}(e\ f)$, $\mathcal{M}_{UMLS}$
Output: Refined confidence value $\mathsf{conf_r}(\mu)$

```
1:  $\xi_{lex} := 1$                                      ▷ We count the $\mu$ mapping
2:  for each $e_i \in M_e^1$ do
3:      for each $f_i \in M_f^2$ do
4:          $\mu_i := \mathsf{EquivalentClasses}(e_i\ f_i)$
5:          if $\mu_i \in \mathcal{M}_{UMLS}$ then
6:              $\xi_{lex}++$                              ▷ UMLS mappings are worth 1
7:          else
8:              if $\mathsf{conf_{isub}}(\mu_i) \geq 0{,}75$ then
9:                                  ▷ We only consider reliable lexical correspondences
10:                 $\xi_{lex} := \xi_{lex} + \mathsf{conf_{isub}}(\mu_i)$
11:             end if
12:         end if
13:     end for
14: end for
15: $\mathsf{conf_r}(\mu) = \frac{2 \times \xi_{lex}}{|\mathsf{sig}(M_e^1)| + |\mathsf{sig}(M_f^2)|}$
16: return $\mathsf{conf_r}(\mu)$
```

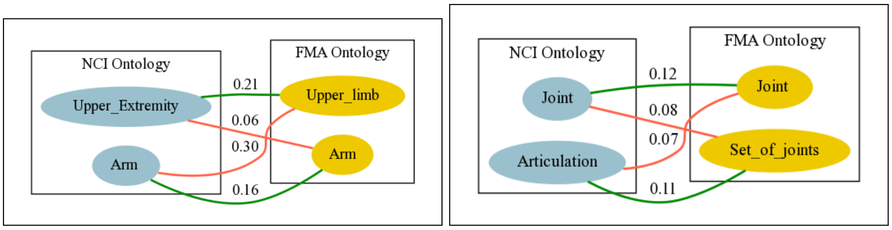


Figure 4.9: Combined cases of ambiguity

algorithms are well-known [Rei87]. In our example, the solution involves removing the mappings μ_2 (with confidence 0,06) and μ_3 (with confidence 0,30).

Disambiguating consistency conflicts automatically.

Consistency conflicts are detected, as described in Section 4.3.3.2, by checking the logic compatibility between pair of mappings. The disambiguation process can be as simple as discarding the mappings with less confidence value. In the example cases from Figure 4.10, the mappings with confidence 0,09 and 0,29 would be discarded. However we detected that a considerable number of mappings were involved in several conflicts. Thus, in order to avoid innecesary deletions of mappings we applied a voting mechanisms. Thus, the desambiguation of a conflict, using the confidence values, leads to the increment of the number of (negative) votes of a mapping, instead of its direct elimination. After the voting process, mappings are reviewed and those with more than 2 votes are discarded. Mappings with 1 or 2 votes are saved if and only if all the mappings which voted them have a higher number of votes. Obviously, mappings with no votes are kept.

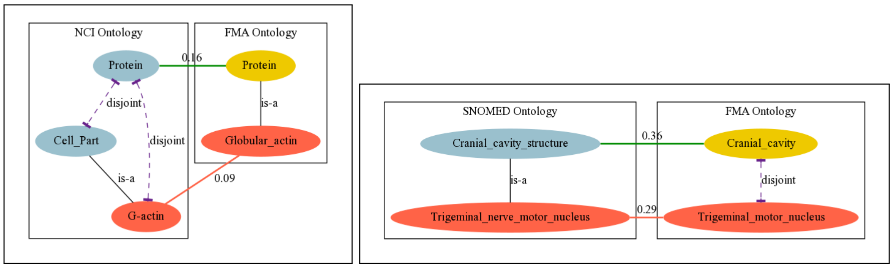


Figure 4.10: Disambiguating consistency conflicts

4.3.3.4. Resolving incompatibilities between ontologies

Even if the mappings between two ontology sources are the intended ones, the sources may describe a particular aspect of the domain in incompatible ways. For example, consider Figure 4.11 describing the notion of “Visceral Pleura” in FMA and NCI. The three mappings between the entities “Visceral Pleura”, “Lung” and “Thoracic Cavity” in both ontologies are clearly the intended ones. However, their integration results in $Visceral\_Pleura$ becoming unsatisfiable. According to NCI, the visceral pleura is located in a lung; furthermore, it is a pleural tissue, which can only be located in the thoracic cavity. However, according to FMA the thoracic cavity is an immaterial anatomical entity, whereas the lung is a material anatomical entity. Finally, material and immaterial entities are disjoint, as implied by FMA. Therefore, the visceral pleura is located in some anatomical entity that is both material and immaterial, which leads to a contradiction.

These source incompatibilities are not always apparent; for example Figure 4.12 shows the justification for $Visceral\_Pleura \sqsubseteq \bot$ where twenty axioms (three of them are mappings) are involved. We believe that, in such cases, the ontology engineer must participate in the repair process and need to be supported by suitable tools like ContentMap [JRCHB09].

As presented in Section 4.2, ContentMap proposes different repair plans for those entailments that the user indicates are unintended and ranks them accoding to the impact of their application. These plans may involve both the deletion or modification of source ontology axioms, and thus expert intervention is required to select the most appropriate repair.

4.3.4. Empirical results

We have evaluated our techniques using UMLS-Meta version 2009AA and the corresponding versions of FMA, NCI and SNOMED CT, which contain 66,724, 78,989

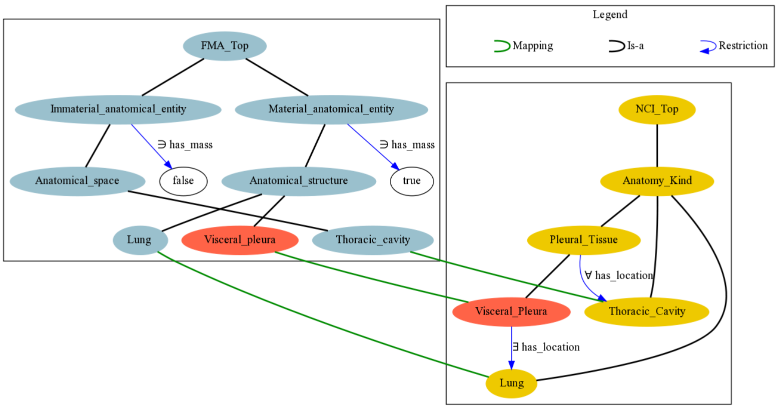
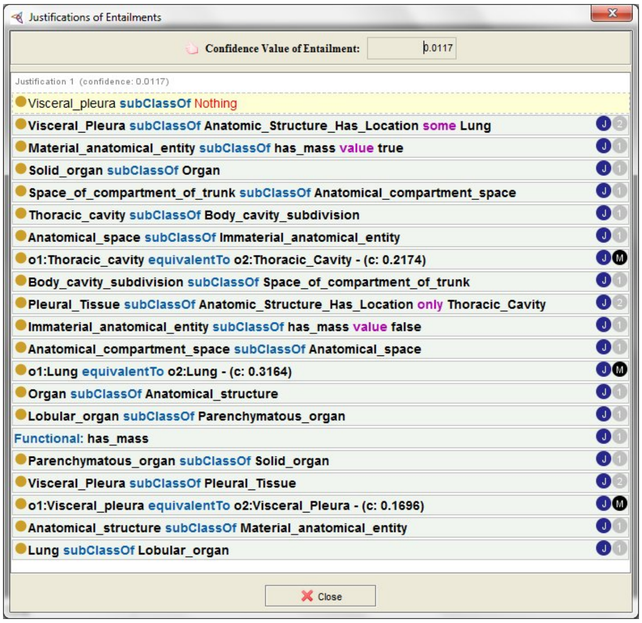


Figure 4.11: Unsatisfiability due to incompatibilities between FMA and NCI


Justifications of Entailments

Confidence Value of Entailment: 0.0117

Justification 1 (confidence: 0.0117)

Visceral_pleura subClassOf Nothing
Visceral_Pleura subClassOf Anatomic_Structure_Has_Location some Lung
Material_anatomical_entity subClassOf has_mass value true
Solid_organ subClassOf Organ
Space_of_compartment_of_trunk subClassOf Anatomical_compartment_space
Thoracic_cavity subClassOf Body_cavity_subdivision
Anatomical_space subClassOf Immaterial_anatomical_entity
o1:Thoracic_cavity equivalentTo o2:Thoracic_Cavity - (c: 0.2174)
Body_cavity_subdivision subClassOf Space_of_compartment_of_trunk
Pleural_Tissue subClassOf Anatomic_Structure_Has_Location only Thoracic_Cavity
Immaterial_anatomical_entity subClassOf has_mass value false
Anatomical_compartment_space subClassOf Anatomical_space
o1:Lung equivalentTo o2:Lung - (c: 0.3164)
Organ subClassOf Anatomical_structure
Lobular_organ subClassOf Parenchymatous_organ
Functional: has_mass
Parenchymatous_organ subClassOf Solid_organ
Visceral_Pleura subClassOf Pleural_Tissue
o1:Visceral_pleura equivalentTo o2:Visceral_Pleura - (c: 0.1696)
Anatomical_structure subClassOf Material_anatomical_entity
Lung subClassOf Lobular_organ

Close

Figure 4.12: Justification for $Visceral\_Pleura \sqsubseteq \bot$

and 304,802 entities, respectively. After translating the UMLS-Meta mappings into OWL 2, we obtained 3,024 mapping axioms between FMA and NCI, 9,072 between FMA and SNOMED CT and 19,622 between SNOMED CT and NCI.

When reasoning over each of the source ontologies independently, all their entities were found satisfiable. However, after the respective integrations via UMLS-

Meta mappings, we obtained a huge number of unsatisfiable entities, namely 5,015 when integrating FMA and NCI, 16,764 with FMA and SNOMED CT, and 76,025 with SNOMED CT and NCI. Thus, from a semantic point of view, the integration of these ontologies via UMLS-Meta is far from error-free.

In order to identify conflicts between the obtained UMLS-Meta mappings, we have then applied the conservativity and consistency principles.

- Using the principle of conservativity, we found 991 conflicting mapping between FMA and NCI, 2,426 between FMA and SNOMED CT and 9,080 between SNOMED CT and NCI.

- Using the disjointness-based inconsistency heuristic, we found 300 conflicting mapping pairs between FMA and NCI, 14,959 between FMA and SNOMED CT and 34,628 between SNOMED CT and NCI. Note that each of these conflicts will certainly lead to the unsatisfiability of an entity in the union of the respective source ontologies and UMLS-Meta mappings.

As discussed in Section 4.3.3.3, the locality principle allows us to assign a confidence value to each UMLS-Meta mapping and then exploit this value to automatically disambiguate the conflicts detected by the conservativity and consistency principles. The automatic disambiguation process removed 570 (19 %) of the mappings between FMA and NCI, 4,077 (45 %) of those between FMA and SNOMED CT and 13,358 (63 %) of those between SNOMED CT and NCI.

After automatic disambiguation, we found only 2 unsatisfiable entities when integrating FMA and NCI (recall example from Figure 4.11), 44 for FMA and SNOMED CT, and none for SNOMED CT and NCI. As already discussed, these errors are most likely due to inherent incompatibilities between the ontology sources; thus, expert assessment is required. ContentMap was then used to understand and repair these remaining inconsistencies. ContentMap proposed 19 repair plans for FMA and NCI and 372 FMA and SNOMED CT. We have inspected the top-ranked repair plans and found them intuitive and reasonable from a modelling perspective.

Despite the large number of deprecated mappings during automatic disambiguation, the integration of our source ontologies via UMLS-Meta still results in a considerable number of (possibly intended) new logical consequences. For example, after repairing all inconsistencies using ContentMap, the integration between FMA and SNOMED CT still results in 973 new subsumption relationships between FMA entities and 586 new subsumption relationships between SNOMED CT entities. Further manual revision (e.g., using again our tool ContentMap) would be necessary to determine which ones among these new consequences are unintended.

4.3.5. Discussion

When integrating ontology sources via UMLS-Meta, the contents of the ontology sources should be taken into account. Several authors have proposed different kinds of *structural analysis* of SNOMED CT, FMA and NCI to evaluate their compatibility. For example, Bodenreider and Zhang [BZ06] compared the representation and coverage of anatomy in SNOMED CT and FMA; to this end, they applied matching techniques to obtain lexical correspondences and performed a structural validation of them. Bodenreider [Bod08] also performed a comparison of SNOMED CT and NCI mapped via UMLS-Meta. The comparison relied on the analysys of the "neighbourhood" (e.g. shared superclasses, shared subclasses, related classes) of the SNOMED CT and NCI entities considered as equivalent according to UMLS-Meta.

We have argued that the rich *logic-based semantics* of ontology sources should also be considered together with their lexical and structural information. We have provided empirical evidence suggesting that, by taking into account the semantics of the sources, one can detect a significant number of additional errors in UMLS-Meta as well as many inherent incompatibilities between the sources' description of particular domains. We therefore consider that our results naturally complement those in [BZ06, Bod08], as well as current auditing methodologies in UMLS-Meta [Cim98, CMP03, MBB09, MGHP09, GPHC09].

Furthermore, we have identified general principles and specific repair techniques, which can be applied not only to assess resources such as UMLS-Meta but also the integration of ontologies in general.

Although the obtained results show the feasibility of our techniques in practice, we consider, however, that they can be imporved in several ways. In particular, our automatic disambiguation is rather aggressive, in the sense that a significant number of mappings are discarded in order to prevent logical errors. We plan to explore how the disambiguation process can be relaxed to include as many of the original mappings as possible. Finally, we aim at seeking feedback from domain experts concerning both the automatic and the tool-assisted disambiguation processes.

CHAPTER 5

Conclusions and open lines

In this research, we have proposed logic-based techniques and implemented prototype systems for knowledge reuse, concurrent evolution, and integration of independent resources.

Our techniques have been designed for the Ontology Web Language (OWL) and its revision OWL 2 [PSHH04, MPSCG09], but can also be applied to any other ontology formalisms (e.g., OBO language), provided that such formalisms can be expressed in or translated into OWL. Our software prototypes are compatible with current Semantic Web infrastructure, such as the OWL API [1] [BVL03, HB09], and are currently available as Protégé plugins[2]; however, they could also be adapted and integrated into other similar frameworks, such as the NeOn toolkit[3] [EBHW08].

The well-founded modularization framework from Cuenca-Grau et al. [GHKS08] deserves an special mention since it has provided our contributions with a transversal technology, which has been applied in a number of situations and for different purposes, e.g., reuse, visualization, optimized processing, extraction of justifications, reasoning with modules, and so on.

Proposed methods and systems have been discussed in the context of state-of-the-art technologies; however further comparison with current available techniques and tools would be necessary in order to provide a more precise evaluation of our proposed techniques. Next, we recapitulate our main contributions and discuss possible lines for future research.

[1] OWL API: `http://owlapi.sourceforge.net/`

[2] Implemented Tools: ProSÉ , ContentCVS and ContentMap , `http://krono.act.uji.es/people/Ernesto/index_html#tools`

[3] NeOn toolkit: `http://www.neon-toolkit.org`

5.1. Contributions to knowledge reuse

Our contribution to knowledge reuse has been twofold. First, we have presented a method for reusing knowledge from non-ontological resources (e.g., thesauri). Second, we have designed a framework to reuse knowledge from ontologies in an economic and safe way, based on the modularization formalism from [GHKS08]. It is worth mentioning that both reuse methods do not require the use of a DL reasoner, as they are solely based on the structure of the input thesauri/ontologies.

The approach to safely reuse from external ontologies restricts the shape of the axioms involving external entities, according to the *locality* conditions from [GHKS08]. These locality restrictions, however, could be too strict for certain applications. Therefore, an interesting direction for future research would be to analyze other reuse strategies which can be complemented with locality conditions.

Another interesting research direction could be the evaluation of the impact of the reused knowledge over the importing ontology (and not only over the reused knowledge). The techniques described in Chapters 3 and 4 could be extended and adapted for this purpose. In this way, the evaluation of the impact would be useful to analyze the suitability of the reused resources.

Finally, a common limitation of both contributions is that they neglect a formal specification of requirements such as the proposed in [dCSFGPVT09]. For future work, we are planning to consider formal approaches to specify user requirements in order to determine which knowledge resources should be reused .

5.1.1. Publications

Most of the results described here have been documented in the following publications:

[JYJRBRS09b] Antonio Jimeno-Yepes, *Ernesto Jiménez-Ruiz*, Rafael Berlanga, and Dietrich Rebholz-Schuhmann. **Reuse of terminological resources for efficient ontological engineering in life sciences**. BMC Bioinformatics, 10(Suppl 10):S4, 2009.

[JRGS$^+$08b] *Ernesto Jiménez-Ruiz*, Bernardo Cuenca Grau, Ulrike Sattler, Thomas Schneider, and Rafael Berlanga Llavori. **Safe and Economic Re-Use of Ontologies: A Logic-Based Methodology and Tool Support**. In The Semantic Web: Research and Applications, 5th European Semantic Web Conference, ESWC, volume 5021 of Lecture Notes in Computer Science, 2008.

[JRGS$^+$08c] *Ernesto Jiménez-Ruiz*, Bernardo Cuenca Grau, Ulrike Sattler, Thomas Schneider, and Rafael Berlanga Llavori. **Safe and Economic Re-Use of Ontologies: A Logic-Based Methodology and Tool Support**. Proceedings of the 21st International Workshop on Description Logics (DL), Volume 353 of CEUR WS Proceedings, 2008. Shortened version of conference paper [JRGS$^+$08b].

[JRGS$^+$08a] *Ernesto Jiménez-Ruiz*, Bernardo Cuenca Grau, Ulrike Sattler, Thomas Schneider, and Rafael Berlanga Llavori. **ProSÉ: a Protégé plugin for Reusing**

Ontologies, Safe and Économique (DEMO). In XIII Jornadas de Ingeniería del Software y Bases de Datos (JISBD), 2008.

Former research on ontology fragmentation was also published in the following papers:

[JRLNS07] *Ernesto Jiménez-Ruiz*, Rafael Berlanga Llavori, Victoria Nebot, and Ismael Sanz. **OntoPath: A language for retrieving ontology fragments**. In The 6th International Conference on Ontologies, DataBases, and Applications of Semantics (ODBASE), OTM Conferences, volume 4803 of Lecture Notes in Computer Science, 2007.

[JRNL$^+$07] *Ernesto Jiménez-Ruiz*, Victoria Nebot, Rafael Berlanga, Ismael Sanz and Alfonso Rios. **A Protégé Plug-in-based System to Manage and Query large Domain Ontologies**. 10th International Protégé Conference, July 15-18, 2007, Budapest, Hungary.

[JRLS$^+$05] *E. Jiménez-Ruiz*, R. Berlanga, I. Sanz, M. J. Aramburu, R. Danger. **OntoPathView: A Simple View Definition Language for the Collaborative Development of Ontologies**. In B. López et al. (Eds.): Artificial Intelligence Research and Development, pages 429-436. IOS Press, 2005.

It is worth mentioning that text mining techniques have also been applied to both the reuse of terms from the literature or medical protocols and the detection of interesting fragments from available thesauri, such as UMLS-Meta. The following publications summarize our contributions to this particular area:

[JYJRL$^+$08] Antonio Jimeno-Yepes, *Ernesto Jiménez-Ruiz*, Vivian Lee, Sylvain Gaudan, Rafael Berlanga, and Dietrich Rebholz-Schuhmann. **Assessment of disease named entity recognition on a corpus of annotated sentences**. BMC Bioinformatics, 9(Suppl 3):S3, 2008.

[BJRNS10] Rafael Berlanga, *Ernesto Jiménez-Ruiz*, Victoria Nebot, and Ismael Sanz. **FAETON: Form analysis and extraction tool for ontology construction**. International Journal of Computer Applications in Technology (IJCAT). In press 2010.

[BJRR$^+$08] Rafael Berlanga, *Ernesto Jiménez-Ruiz*, Dmitry Rogulin, Victoria Nebot, David Manset, Andrew Branson, Tamas Hauer, Richard Mc- Clatchey, Dmitry Rogulin, Jetendr Shamdasani, and et al. **Medical data integration and the semantic annotation of medical protocols**. In The 21th IEEE International Symposium on Computer-Based Medical Systems (CBMS), 2008.

[LASPPJR08] Rafael Berlanga Llavori, Henry Anaya-Sánchez, Aurora Pons- Porrata, and *Ernesto Jiménez-Ruiz*. **Conceptual subtopic identification in the medical domain**. In Advances in Artificial Intelligence - IBERAMIA 2008, 11th Ibero-American Conference on AI, volume 5290 of Lecture Notes in Computer Science, 2008.

5.2. Contributions to concurrent ontology evolution

User studies have proved that ontology developers are mainly interested in simple but functional tools [SMS09]. Mechanisms to allow communication among developers and techniques to describe changes are the most demanded functionalities. Nevertheless they also required mechanisms to prevent conflicts (e.g., undesired logical consequences) between concurrent changes.

We have designed and implemented a framework to assess concurrent changes in collaborative ontology development. This framework follows an *asynchronous* editing paradigm, performs an *untraced versioning*, and evaluates *the logic impact* of merging concurrent changes. Moreover, suggestions are given in order to *curate* errors (i.e., unintended consequences). We have also conducted a preliminary user study and performance evaluation from which we obtained encouraging feedback from users and promising empirical results.

Our framework could be improved in a number of ways. Currently, our framework does not consider access policies nor predefined restriction on the changes (e.g., unit test or extension policies). An interesting future work would involve the design of extension policies to better control the concurrent evolution of the ontology. Moreover, current repair suggestions only involve the deletion or addition of axioms, which in some cases may be too strong. Suggestions involving the modification of axioms should be a future feature for the framework.

Another interesting line would be the integration of our framework within current collaborative ontology evolution methodologies and workflows (e.g., [Pal09]) in order to consider both technical (e.g., assignment of duties, change representation and characterization) and social aspects (e.g., discussion and argumentation).

Regarding scalability, the computation of justifications for a huge number of consequences could be computationally prohibitive. Therefore, the generation of plans or suggestion to repair a large number of errors could also be computationally unfeasible. The use of incremental reasoning techniques and the use of logic-based modules is particularly interesting in this regard. Furthermore, it should be noticed that the longer modellers have been working independently, the larger the number of possible errors and conflicts. For example, NCI and SNOMED CT require a complex curation process after each development cycle which may take sevaral weeks [SR07b].

Finally, the assessment of the compatibility of independent sets of changes should only focus on the semantic level, avoiding lexical problems about the normalization of labels referring to the same entity. Thus, the reuse of terminological resources would avoid many of such lexical discrepancies [JYJRBRS09b].

5.2.1. Publications

The following papers summarize our contributions to the community:

[JRGHL10b] *E. Jiménez-Ruiz*, B. Cuenca Grau, I. Horrocks, R. Berlanga. **Supporting Concurrent Development of Ontologies: Framework, Algorithms and Tool**. Submitted to Data & Knowledge Engeneering Journal, 2010.

[JRGHL09a] *E. Jiménez-Ruiz*, B. Cuenca Grau, I. Horrocks, R. Berlanga. **Building Ontologies Collaboratively using ContentCVS**. Proceedings of the 22nd International Workshop on Description Logics (DL2009). Volume 477 of CEUR WS Proceedings, 2009.

[JRGHL09b] *E. Jiménez-Ruiz*, B. Cuenca Grau, I. Horrocks, R. Berlanga. **ContentCVS: A CVS-based Collaborative ONTology ENgineering Tool (DEMO)**. Proceedings of the 2nd International Workshop on Semantic Web Applications and Tools for Life Sciences (SWAT4LS2009). Volume 559 of CEUR WS Proceedings, 2009

[JRL06] *E. Jiménez-Ruiz*, R. Berlanga. **A View-based Methodology for Collaborative Ontology Engineering: an Approach for Complex Applications (VIMethCOE)**. In 1st International Workshop (WETICE workshop) on Semantic Technologies in Collaborative Applications (STICA), 2006. Selected "best workshop paper"based on reviews.

5.3. Contributions to the integration of independent ontological resources

Ontology integration should be an audited process since ontologies may contain conflicting descriptions of the overlapping entities; thus, even if the appropriate correspondences have been established, the integrated ontology may contain errors (e.g., unintended consequences). We have designed and implemented semi-automatic and automatic techniques to assess the integration of independently created ontologies.

Our semi-automatic techniques support the user in the analysis and understanding of the consequences derived from the integration. Moreover, they provide suggestions to repair logic errors. These techniques are similar to the ones we have applied to the assessment of concurrent changes; therefore, they have similar benefits and limitations. Scalability is even more critical in the setting of ontology integration since, in contrast to the case of concurrent development where "commit" operations are relatively frequent, the number of mappings between medium or large ontologies, and the consequences of the integration are usually huge. Thus, manual assessment of mapping and their respective consequences can be rather costly.

Implemented automatic techniques are based on logic-based principles and aimed at reducing the number of erroneous or conflictive mappings and the number of obvious unintended consequences. Obtained results showed the feasibility of these techniques in practice; however a significant number of mappings were discarded. We plan to explore how these techniques can be relaxed in order to be less aggressive and discard smaller sets of mappings.

5.3.1. Publications

We have contributed to the ontology integration community with the following publications:

[JRCHB09] *Ernesto Jiménez-Ruiz*, Bernardo Cuenca Grau, Ian Horrocks, and Rafael Berlanga. **Ontology integration using mappings: Towards getting the right logical consequences**. In Proc. of European Semantic Web Conference (ESWC), volume 5554 of LNCS, pages 173-187. Springer-Verlag, 2009.

[JRGHL10a] *Ernesto Jiménez-Ruiz*, Bernardo Cuenca Grau, Ian Horrocks, and Rafael Berlanga Llavori. Logic-based assessment of the compatibility of UMLS ontology sources. Accepted for publication in Journal of Biomedical Semantics, 2010.

[JRGHL09c] *Ernesto Jiménez-Ruiz*, Bernardo Cuenca Grau, Ian Horrocks, and Rafael Berlanga Llavori. Logic-based ontology integration using ContentMap. In XIV Jornadas de Ingeniería del Software y Bases de Datos (JISBD), pages 316-319, 2009.

[JRGHL09d] *Ernesto Jiménez-Ruiz*, Bernardo Cuenca Grau, Ian Horrocks, and Rafael Berlanga Llavori. **Towards a logic-based assessment of the compatibility of UMLS sources**. In Proceedings of the 2nd International Workshop on Semantic Web Applications and Tools for Life Sciences (SWAT4LS 2009), Amsterdam, The Netherlands, volume 559 of CEUR Workshop Proceedings. CEUR-WS.org, 2009.

Bibliography

[ABB$^+$00] Michael Ashburner, Catherine A. Ball, Judith A. Blake, David Botstein, Heather Butler, J. Michael Cherry, Allan P. Davis, Kara Dolinski, et al. Gene Ontology: tool for the unification of biology. *Nature Genetics*, 25(1):25–29, May 2000.

[ABB$^+$07] Ashiq Anjum, Peter Bloodsworth, Andrew Branson, Tamas Hauer, Richard McClatchey, Kamran Munir, Dmitry Rogulin, and Jetendr Shamdasani. The requirements for ontologies in medical data integration: A case study. In *IDEAS '07: Proceedings of the 11th International Database Engineering and Applications Symposium*, pages 308–314, Washington, DC, USA, 2007. IEEE Computer Society.

[AG09] Mark A. Musen Amir Ghazvinian, Natalya Fridman Noy. Creating mappings for ontologies in biomedicine: Simple methods work. In *AMIA Annual Symposium*, pages 198–202, 2009.

[Ala06] Harith Alani. Position paper: ontology construction from online ontologies. In *Proceedings of the 15th international conference on World Wide Web, WWW*, pages 491–495. ACM, 2006.

[Aro01] Alan R. Aronson. Effective mapping of biomedical text to the UMLS Metathesaurus: the MetaMap program. *Proc AMIA Symp*, pages 17–21, 2001.

[BAT97] Pim Borst, Hans Akkermans, and Jan Top. Engineering ontologies. *Int. J. Hum.-Comput. Stud.*, 46(2-3):365–406, 1997.

[BBA$^+$03] Brigitte Boeckmann, Amos Bairoch, Rolf Apweiler, Marie-Claude Blatter, Anne Estreicher, Elisabeth Gasteiger, Maria Jesus Martin, Karine Michoud, Claire O'Donovan, Isabelle Phan, Sandrine Pilbout, and Michel Schneider. The swiss-prot protein knowledgebase and its supplement trembl in 2003. *Nucleic Acids Research*, 31(1):365–370, 2003.

[BCH06] Jie Bao, Doina Caragea, and Vasant Honavar. Towards collaborative environments for ontology construction and sharing. In *CTS '06: Proceedings of the International Symposium on Collaborative Technologies and Systems*, pages 99–108, Washington, DC, USA, 2006. IEEE Computer Society.

[BCM$^+$03] Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003.

[BJRNS10] Rafael Berlanga, Ernesto Jimenez-Ruiz, Victoria Nebot, and Ismael Sanz. FAETON: Form analysis and extraction tool for ontology construction. *International Journal of Computer Applications in Technology (IJCAT)*, 2010.

[BJRR$^+$08] Rafael Berlanga, Ernesto Jiménez-Ruiz, Dmitry Rogulin, Victoria Nebot, David Manset, Andrew Branson, Tamas Hauer, Richard McClatchey, Dmitry Rogulin, Jetendr Shamdasani, and et al. Medical data integration and the semantic annotation of medical protocols. In *The 21th IEEE International Symposium on Computer-Based Medical Systems (CBMS)*, pages 644–649, 2008.

[BL84] Ronald J. Brachman and Hector J. Levesque. The tractability of subsumption in frame-based description languages. In *AAAI*, pages 34–37, 1984.

[BLS06] Franz Baader, Carsten Lutz, and Boontawee Suntisrivaraporn. CEL - a polynomial-time reasoner for life science ontologies. In Furbach and Shankar [FS06], pages 287–291.

[BM03a] Olivier Bodenreider and Alexa T. McCray. Exploring semantic groups through visual approaches. *Journal of Biomedical Informatics*, 36(6):414 – 432, 2003. Unified Medical Language System.

[BM03b] Olivier Bodenreider and Alexa T. McCray. Exploring semantic groups through visual approaches. *Journal of Biomedical Informatics*, 36(6):414–432, 2003.

[BMS00] Allen C. Browne, Alexa T. McCray, and Suresh Srinivasan. The Specialist Lexicon. National Library of Medicine. Technical Report. http://lexsrv3.nlm.nih.gov/SPECIALIST/, 2000.

[BMT05] Elena Paslaru Bontas, Malgorzata Mochol, and Robert Tolksdorf. Case studies on ontology reuse. In *5th International Conference on Knowledge Management (IKnow*, 2005.

[Bod04] Olivier Bodenreider. The Unified Medical Language System (UMLS): integrating biomedical terminology. *Nucleic acids research*, 32(Database issue), January 2004.

[Bod06] Olivier Bodenreider. Lexical, terminological and ontological resources for biological text mining. In *Text mining for biology and biomedicine. Artech House*, 2006.

[Bod08] Olivier Bodenreider. Comparing snomed ct and the nci thesaurus through semantic web technologies. In *Proceedings of the Third International Conference on Knowledge Representation in Medicine, KR-MED*, volume 410 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2008.

[Bor09] Alexander Borgida. On importing knowledge from ontologies. In Stuckenschmidt et al. [SPS09], pages 91–112.

[BPS07] Franz Baader, Rafael Peñaloza, and Boontawee Suntisrivaraporn. Pinpointing in the description logic $\mathcal{EL}^+$. In Joachim Hertzberg, Michael Beetz, and Roman Englert, editors, *30th Annual German Conference on AI, KI*, volume 4667 of *Lecture Notes in Computer Science*, pages 52–67. Springer, 2007.

[Bra77] Ronald J. Brachman. What's in a concept: structural foundations for semantic networks. *International Journal of Man-Machine Studies*, 9(2):127 – 152, 1977.

[Bra79] Ronald J. Brachman. On the epistemological status of semantic networks. In N. V. Findler, editor, *Associative networks: Representation and use of knowledge by computers*. New York: Academic, 1979.

[BS85] Ronald J. Brachman and James G. Schmolze. An overview of the KL-ONE knowledge representation system. *Cognitive Science*, 9(2):171 – 216, 1985.

[BVL03] Sean Bechhofer, Raphael Volz, and Phillip W. Lord. Cooking the Semantic Web with the OWL API. In *International Semantic Web Conference*, volume 2870 of *Lecture Notes in Computer Science*, pages 659–675. Springer, 2003.

[BVSH09] Jie Bao, George Voutsadakis, Giora Slutzki, and Vasant Honavar. Package-based description logics. In Stuckenschmidt et al. [SPS09], pages 349–371.

[BZ06] Olivier Bodenreider and Songmao Zhang. Comparing the representation of anatomy in the fma and snomed ct. In *AMIA Annual Symposium*, pages 46–50, 2006.

[CFLGPV02] Óscar Corcho, Mariano Fernández-López, Asunción Gómez-Pérez, and Óscar Vicente. Webode: An integrated workbench for ontology representation, reasoning, and exchange. In *13th International Conference on Knowledge Engineering and Knowledge Management, EKAW*, volume 2473 of *Lecture Notes in Computer Science*, pages 138–153. Springer, 2002.

[CGP00] Óscar Corcho and Asunción Gómez-Pérez. A roadmap to ontology specification languages. In *EKAW '00: Proceedings of the 12th European Workshop on Knowledge Acquisition, Modeling and Management*, pages 80–96, London, UK, 2000. Springer-Verlag.

[CHKS07] Bernardo Cuenca Grau, Ian Horrocks, Yevgeny Kazakov, and Ulrike Sattler. Just the right amount: extracting modules from ontologies. In *proc. of the 16th International Conference on World Wide Web (WWW 2007)*, pages 717–726, 2007.

[CHM$^+$08] B. Cuenca Grau, I. Horrocks, B. Motik, B. Parsia, P. Patel-Schneider, and U. Sattler. OWL 2: The next step for OWL. *J. Web Semantics*, 6(4):309–322, 2008.

[CHWK07] Bernardo Cuenca Grau, Christian Halaschek-Wiener, and Yevgeny Kazakov. History matters: Incremental ontology reasoning using modules. In *Proc. of the 6th International Semantic Web Conference (ISWC)*, volume 4825 of *LNCS*, pages 183–196. Springer, 2007.

[Cim98] James J. Cimino. Auditing the unified medical language system with semantic methods. *Journal of the American Medical Informatics Association (JAMIA)*, 5(1):41–51, 1998.

[CK07] Bernardo Cuenca Grau and Oliver Kutz. Modular ontology languages revisited. In *SWeCKa 2007: Proc. of the IJCAI-2007 Workshop on Semantic Web for Collaborative Knowledge Acquisition , Hyderabad, India*, 2007.

[CMP03] James J. Cimino, Hua Min, and Yehoshua Perl. Consistency across the hierarchies of the umls semantic network and metathesaurus. *Journal of Biomedical Informatics*, 36(6):450–461, 2003.

[CSH06] Namyoun Choi, Il-Yeol Song, and Hyoil Han. A survey on ontology mapping. *SIGMOD Rec.*, 35(3):34–41, 2006.

[D$^+$05] Ciaran M. Duffy et al. Nomenclature and classification in chronic childhood arthritis: Time for a change? *Arthritis and Rheumatism*, 52(2):382–385, 2005.

[Dav08] Jérôme David. Aroma results for OAEI 2008. In *Proceedings of the 3rd International Workshop on Ontology Matching (OM)*, volume 431 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2008.

[Dav09] Jérôme David. Aroma results for OAEI 2009. In *Proceedings of the 4th International Workshop on Ontology Matching, OM*, volume 551 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2009.

[DBC06] Alicia Diaz, Guillermo Baldo, and Gerome Canals. Co-Protege: Collaborative ontology building with divergences. *Database and Expert Systems Applications*, pages 156–160, 2006.

[dBG+07] Mathieu d'Aquin, Claudio Baldassarre, Laurian Gridinoc, Sofia Angeletou, Marta Sabou, and Enrico Motta. Characterizing knowledge on the semantic web with watson. In *Proceedings of the 5th International Workshop on Evaluation of Ontologies and Ontology-based Tools, EON, Co-located with the ISWC*, volume 329 of *CEUR Workshop Proceedings*, pages 1–10. CEUR-WS.org, 2007.

[dCSFGPVT09] María del Carmen Suárez-Figueroa, Asunción Gómez-Pérez, and Boris Villazón-Terrazas. How to write and use the ontology requirements specification document. In *On the Move to Meaningful Internet Systems: OTM 2009, Confederated International Conferences*, volume 5871 of *Lecture Notes in Computer Science*, pages 966–982, 2009.

[dCWF+09] Sherri de Coronado, Lawrence W. Wright, Gilberto Fragoso, Margaret W. Haber, Elizabeth A. Hahn-Dantona, Francis W. Hartel, Sharon L. Quan, Tracy Safran, Nicole Thomas, and Lori Whiteman. The NCI thesaurus quality assurance life cycle. *Journal of Biomedical Informatics*, 42(3):530 – 539, 2009.

[DEB+08] Klaas Dellschaft, Hendrik Engelbrecht, José Monte Barreto, Sascha Rutenbeck, and Steffen Staab. Cicero: Tracking design rationale in collaborative ontology engineering. In *The Semantic Web: Research and Applications, 5th European Semantic Web Conference, ESWC*, pages 782–786, 2008.

[DFK+09] Julian Dolby, Achille Fokoue, Aditya Kalyanpur, Edith Schonberg, and Kavitha Srinivas. Scalable highly expressive reasoner (SHER). *J. Web Sem.*, 7(4):357–361, 2009.

[DH05] AnHai Doan and Alon Y. Halevy. Semantic integration research in the database community: A brief survey. *AI Magazine*, 26(1):83–94, 2005.

[DLE$^+$07] Yihong Ding, Deryle W. Lonsdale, David W. Embley, Martin Hepp, and Li Xu. Generating ontologies via language components and ontology reuse. In *12th International Conference on Applications of Natural Language to Information Systems, NLDB*, volume 4592 of *Lecture Notes in Computer Science*, pages 131–142. Springer, 2007.

[dMLM06] Aldo de Moor, Pieter De Leenheer, and Robert Meersman. DOGMA-MESS: A meaning evolution support system for interorganizational ontology engineering. In *14th International Conference on Conceptual Structures, ICCS*, volume 4068 of *Lecture Notes in Computer Science*, pages 189–202. Springer, 2006.

[Dom98] Jonh Domingue. Tadzebao and WebOnto: Discussing, browsing, and editing ontologies on the Web. In *11th Knowledge Acquisition for Knowledge-Based Systems Workshop, Banff, Canada*, 1998.

[Doy91] Jon Doyle, editor. *Special issue on implemented knowledge representation and reasoning systems*, volume 2, New York, NY, USA, 1991. ACM.

[DR06] John Day-Richter. The OBO flat file format specification, version 1.2, 2006. Gene Ontology Project.

[DSFB$^+$09] Martin Dzbor, Mari Carmen Suárez-Figueroa, Eva Blomqvist, Holger Lewen, Mauricio Espinoza, Asunción Gómez-Pérez, and Raul Palma. D5.6.2 Experimentation and Evaluation of the NeOn Methodology. NeOn Deliverable available at: http://www.neon-project.org/, February, 2009.

[DTI07] Paul Doran, Valentina A. M. Tamma, and Luigi Iannone. Ontology module extraction for ontology reuse: an ontology engineering perspective. In *Proceedings of the Sixteenth ACM Conference on Information and Knowledge Management, CIKM*, pages 61–70. ACM, 2007.

[EBHW08] Michael Erdmann, M. Becker, W. Hihn, and R. Wroblenski. D6.6.2 NeOn Toolkit Documentation. NeOn Deliverable available at: http://www.neon-project.org/, October, 2008.

[EFV10] Thomas Meyer Enrico Franconi and Ivan Varzinczak. Semantic Diff as the basis for knowledge base versioning. In *13th International Workshop on Non-Monotonic Reasoning (NMR)*, 2010.

[ES07] Jérôme Euzenat and Pavel Shvaiko. *Ontology matching*. Springer-Verlag, Heidelberg (DE), 2007.

[Euz07] Jérôme Euzenat. Semantic precision and recall for ontology alignment evaluation. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence, IJCAI*, pages 348–353, 2007.

[EV04] Jérôme Euzenat and Petko Valtchev. Similarity-based ontology alignment in owl-lite. In *Proceedings of the 16th Eureopean Conference on Artificial Intelligence, ECAI*, pages 333–337, 2004. Tool available at: `http://ola.gforge.inria.fr/`.

[FCI$^+$06] J. Freund, D. Comaniciu, Y. Ioannis, P. Liu, R. McClatchey, E. Moley-Fletcher, X. Pennec, G. Pongiglione, and X.S. Zhou. Health-e-Child: An integrated biomedical platform for grid-based pediatrics. In *Proceedings of Health-Grid 2006*, volume 120 of *Studies in Health Technology and Informatics*, pages 259–270, Valencia, Spain, 2006.

[Fel98] Christiane Fellbaum. *WordNet: An Electronic Lexical Database (Language, Speech, and Communication).* The MIT Press, 1998.

[FFR97] Adam Farquhar, Richard Fikes, and James Rice. The Ontolingua server: a tool for collaborative ontology construction. *International Journal of Human-Computer Studies*, pages 707–727, 1997.

[FMK$^+$08] Giorgos Flouris, Dimitris Manakanatas, Haridimos Kondylakis, Dimitris Plexousakis, and Grigoris Antoniou. Ontology change: classification and survey. *Knowledge Eng. Review*, 23(2):117–152, 2008.

[FOSM09] Miriam Fernández, Chwhynny Overbeeke, Marta Sabou, and Enrico Motta. What makes a good ontology? a case-study in fine-grained knowledge reuse. In *The Semantic Web, Fourth Asian Conference, ASWC*, volume 5926 of *Lecture Notes in Computer Science*. Springer, 2009.

[FS06] Ulrich Furbach and Natarajan Shankar, editors. *Automated Reasoning, Third International Joint Conference, IJCAR 2006, Seattle, WA, USA, August 17-20, 2006, Proceedings*, volume 4130 of *Lecture Notes in Computer Science*. Springer, 2006.

[GdM09] Jorge Gracia, Mathieu d'Aquin, and Eduardo Mena. Large scale integration of senses for the semantic web. In *Proceedings of the 18th International Conference on World Wide Web (WWW)*, pages 611–620, 2009.

[GFH$^+$03] Jennifer Golbeck, Gilberto Fragoso, Frank W. Hartel, James A. Hendler, Jim Oberthaler, and Bijan Parsia. The National Cancer Institute's Thésaurus and Ontology. *JWS*, 1(1):75–80, 2003.

[GHH$^+$07] Christine Golbreich, Matthew Horridge, Ian Horrocks, Boris Motik, and Rob Shearer. OBO and OWL: Leveraging semantic web technologies for the life sciences. In *The sixth International Semantic Web Conference (ISWC 2007)*, pages 169–182, 2007.

[GHKS07] Bernardo Cuenca Grau, Ian Horrocks, Yevgeny Kazakov, and Ulrike Sattler. A logical framework for modularity of ontologies. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence, IJCAI*, pages 298–303, 2007.

[GHKS08] Bernardo Cuenca Grau, Ian Horrocks, Yevgeny Kazakov, and Ulrike Sattler. Modular reuse of ontologies: Theory and practice. *J. of Artificial Intelligence Research (JAIR)*, 31:273–318, 2008.

[GLW06] Silvio Ghilardi, Carsten Lutz, and Frank Wolter. Did I damage my ontology? a case for conservative extensions in description logics. In *Proceedings, Tenth International Conference on Principles of Knowledge Representation and Reasoning, KR*, pages 187–197. AAAI Press, 2006.

[GM08] Jorge Gracia and Eduardo Mena. Ontology matching with CIDER: Evaluation report for the OAEI 2008. In *Proceedings of the 3rd International Workshop on Ontology Matching, OM*, volume 431 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2008.

[GPC02] Asunción Gómez-Pérez and Oscar Corcho. Ontology specification languages for the semantic web. *IEEE Intelligent Systems*, 17(1):54–60, 2002.

[GPdCSF09] Asunción Gómez-Pérez and María del Carmen Suárez-Figueroa. Scenarios for building ontology networks within the NeOn methodology. In Yolanda Gil and Natasha Fridman Noy, editors, *Proceedings of the 5th International Conference on Knowledge Capture, K-CAP*, pages 183–184. ACM, 2009.

[GPHC09] James Geller, Yehoshua Perl, Michael Halper, and Ronald Cornet. Special issue on auditing of terminologies. *Journal of Biomedical Informatics*, 42(3):407–411, 2009.

[GPS09] Bernardo Cuenca Grau, Bijan Parsia, and Evren Sirin. Ontology integration using $\mathcal{E}$-Connections. In Stuckenschmidt et al. [SPS09], pages 293–320.

[GPSK06] Bernardo Cuenca Grau, Bijan Parsia, Evren Sirin, and Aditya Kalyanpur. Modularity and web ontologies. In *Tenth International Conference on Principles of Knowledge Representation and Reasoning, KR*, pages 198–209. AAAI Press, 2006.

[Gru93a] Thomas R. Gruber. Towards Principles for the Design of Ontologies Used for Knowledge Sharing. In N. Guarino and R. Poli, editors, *Formal Ontology in Conceptual Analysis and Knowledge Representation*, 1993. Updated Definition: http://tomgruber.org/writing/ontology-definition-2007.htm.

[Gru93b] Thomas R. Gruber. A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5(2):199–220, 1993.

[GSGPdCSFVT08] Andrés García-Silva, Asunción Gómez-Pérez, María del Carmen Suárez-Figueroa, and Boris Villazón-Terrazas. A pattern based approach for re-engineering non-ontological resources into ontologies. In *The Semantic Web, 3rd Asian Semantic Web Conference, ASWC*, volume 5367 of *Lecture Notes in Computer Science*, pages 167–181. Springer, 2008.

[Hay79] Patrick J. Hayes. The logic of frames. In D. Metzing, editor, *Frame Conceptions and Text Understanding*, pages 46–61. Walter de Gruyter and Co., Berlin, 1979.

[HB09] Matthew Horridge and Sean Bechhofer. The OWL API: A Java API for working with OWL 2 ontologies. In *Proceedings of the 5th International Workshop on OWL: Experiences and Directions, OWLED*, volume 529 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2009.

[HdCD$^+$05] Frank W. Hartel, Sherri de Coronado, Robert Dionne, Gilberto Fragoso, and Jennifer Golbeck. Modeling a description logic vocabulary for cancer research. *Journal of Biomedical Informatics*, 38(2):114–129, 2005.

[HGH$^+$09] Kuo-Chuan Huang, James Geller, Michael Halper, Yehoshua Perl, and Junchuan Xu. Using wordnet synonym substitution to enhance UMLS source integration. *Artif. Intell. Med.*, 46(2):97–109, 2009.

[HGHC07] Kuo-Chuan Huang, James Geller, Michael Halper, and James J. Cimino. Piecewise synonyms for enhanced UMLS source terminology integration. In *Proceedings of the 2007 American Medical Informatics Association (AMIA) Annual Symposium*, pages 339–343, November 2007.

[Hir04] Graeme Hirst. Ontology and the lexicon. In *Handbook on Ontologies in Information Systems*, pages 209–230. Springer, 2004.

[HKR08] Michael Hartung, Toralf Kirsten, and Erhard Rahm. Analyzing the evolution of life science ontologies and mappings. In *DILS*

'08: proceedings of the 5th international workshop on Data Integration in the Life Sciences, volume 5109 of *Lecture Notes in Computer Science*, pages 11–27. Springer, 2008.

[HKS06] Ian Horrocks, Oliver Kutz, and Ulrike Sattler. The even more irresistible $\mathcal{SROIQ}$. In *Proceedings, Tenth International Conference on Principles of Knowledge Representation and Reasoning*, pages 57–67, 2006.

[HM00] Elliotte Rusty Harold and W. Scott Means. Chapter 9: Xpath. In Laurie Petrycki, editor, *XML in a Nutshell: A Desktop Quick Reference*, Sebastopol, CA, USA, 2000. O'Reilly & Associates, Inc. Chapter 9 available at: http://www.oreilly.com/catalog/xmlnut/chapter/ch09.html.

[HM01] Volker Haarslev and Ralf Moller. RACER system description. In *First International Joint Conference on Automated Reasoning, IJCAR*, volume 2083 of *Lecture Notes in Computer Science*, pages 701–706. Springer, 2001.

[HPS08a] Matthew Horridge, Bijan Parsia, and Ulrike Sattler. Explanation of OWL entailments in Protege 4. In Christian Bizer and Anupam Joshi, editors, *International Semantic Web Conference (Posters & Demos)*, volume 401 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2008.

[HPS08b] Matthew Horridge, Bijan Parsia, and Ulrike Sattler. Laconic and precise justifications in OWL. In Amit P. Sheth, Steffen Staab, Mike Dean, Massimo Paolucci, Diana Maynard, Timothy W. Finin, and Krishnaprasad Thirunarayan, editors, *International Semantic Web Conference*, volume 5318 of *Lecture Notes in Computer Science*, pages 323–338. Springer, 2008.

[HPSvH03] Ian Horrocks, Peter F. Patel-Schneider, and Frank van Harmelen. From $\mathcal{SHIQ}$ and RDF to OWL: the making of a web ontology language. *Journal Web Semantics*, 1(1):7–26, 2003.

[HS05] Peter Haase and Ljiljana Stojanovic. Consistent evolution of OWL ontologies. In *ESWC*, pages 182–197. Springer, 2005.

[HvMS$^+$10] Kristina Hettne, Erik van Mulligen, Martijn Schuemie, Bob Schijvenaars, and Jan Kors. Rewriting and suppressing umls terms for improved biomedical term identification. *Journal of Biomedical Semantics*, 1(1):5, 2010.

[icd07] International Classification of Diseases (ICD), 2007. World Health Organization: `http://www.who.int/classifications/icd/en/`.

[JMSK09] Yves R. Jean-Mary, E. Patrick Shironoshita, and Mansur R. Kabuka. Ontology matching with semantic verification. *Journal of Web Semantics*, 2009.

[JQH09] Qiu Ji, Guilin Qi, and Peter Haase. A relevance-directed algorithm for finding justifications of DL entailments. In Asunción Gómez-Pérez, Yong Yu, and Ying Ding, editors, *ASWC*, volume 5926 of *Lecture Notes in Computer Science*, pages 306–320. Springer, 2009.

[JR04] José L. V. Mejino Jr. and Cornelius Rosse. Symbolic modeling of structural relationships in the foundational model of anatomy. In *proc. of First International Workshop on Formal Biomedical Knowledge Representation (KR-MED 2004)*, pages 48–62, 2004.

[JRBS$^+$06] Ernesto Jiménez-Ruiz, Rafael Berlanga, Ismael Sanz, Richard McClatchey, Roxana Danger, David Manset, Jordi Paraire, and Alfonso Ríos. The management and integration of Biomedical knowledge: Application in the Health-e-Child project. In *OnTo-Content'06, 1st International Workshop on Ontology content and evaluation in Enterprise*, volume 4278 of *LNCS*, pages 1062–1067, 2006.

[JRCHB09] Ernesto Jimenez-Ruiz, Bernardo Cuenca Grau, Ian Horrocks, and Rafael Berlanga. Ontology integration using mappings: Towards getting the right logical consequences. In *Proc. of European Semantic Web Conference (ESWC)*, volume 5554 of *LNCS*, pages 173–187. Springer-Verlag, 2009. Technical Report and Tool also available at: `http://krono.act.uji.es/people/Ernesto/contentmap`.

[JRGHL09a] Ernesto Jiménez-Ruiz, Bernardo Cuenca Grau, Ian Horrocks, and Rafael Berlanga Llavori. Building ontologies collaboratively using ContentCVS. In *Proceedings of the 22nd International Workshop on Description Logics (DL 2009), Oxford, UK*, volume 477 of *CEUR Workshop Proceedings*, 2009.

[JRGHL09b] Ernesto Jiménez-Ruiz, Bernardo Cuenca Grau, Ian Horrocks, and Rafael Berlanga Llavori. ContentCVS: A CVS-based Collaborative ONTology ENgineering Tool (demo). In *Proceedings of the 2nd International Workshop on Semantic Web Applications and Tools for Life Sciences (SWAT4LS 2009), Amsterdam, The Netherlands*, volume 559 of *CEUR Workshop Proceedings*, 2009.

[JRGHL09c] Ernesto Jiménez-Ruiz, Bernardo Cuenca Grau, Ian Horrocks, and Rafael Berlanga Llavori. Logic-based ontology integration using

ContentMap. In *XIV Jornadas de Ingeniería del Software y Bases de Datos (JISBD)*, pages 316–319, 2009.

[JRGHL09d] Ernesto Jiménez-Ruiz, Bernardo Cuenca Grau, Ian Horrocks, and Rafael Berlanga Llavori. Towards a logic-based assessment of the compatibility of UMLS sources. In *Proceedings of the 2nd International Workshop on Semantic Web Applications and Tools for Life Sciences (SWAT4LS 2009), Amsterdam, The Netherlands*, volume 559 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2009.

[JRGHL10a] Ernesto Jiménez-Ruiz, Bernardo Cuenca Grau, Ian Horrocks, and Rafael Berlanga Llavori. Logic-based assessment of the compatibility of UMLS ontology sources. *Submitted to Journal of Biomedical Semantics*, 2010.

[JRGHL10b] Ernesto Jiménez-Ruiz, Bernardo Cuenca Grau, Ian Horrocks, and Rafael Berlanga Llavori. Supporting concurrent development of ontologies: Framework, algorithms and tool. *Data & Knowledge Engeneering (Being reviewed)*, 2010.

[JRGS$^+$08a] Ernesto Jiménez-Ruiz, Bernardo Cuenca Grau, Ulrike Sattler, Thomas Schneider, and Rafael Berlanga Llavori. ProSÉ: a Protégé plugin for Reusing Ontologies, Safe and Économique. In *XIII Jornadas de Ingeniería del Software y Bases de Datos (JISBD)*, pages 449–452, 2008.

[JRGS$^+$08b] Ernesto Jiménez-Ruiz, Bernardo Cuenca Grau, Ulrike Sattler, Thomas Schneider, and Rafael Berlanga Llavori. Safe and Economic Re-Use of Ontologies: A Logic-Based Methodology and Tool Support. In *The Semantic Web: Research and Applications, 5th European Semantic Web Conference, ESWC, Tenerife, Canary Islands, Spain*, volume 5021 of *Lecture Notes in Computer Science*, pages 185–199. Springer, 2008.

[JRGS$^+$08c] Ernesto Jiménez-Ruiz, Bernardo Cuenca Grau, Ulrike Sattler, Thomas Schneider, and Rafael Berlanga Llavori. Safe and Economic Re-Use of Ontologies: A Logic-Based Methodology and Tool Support. In *Proceedings of the 21st International Workshop on Description Logics*, volume 353 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2008.

[JRGS$^+$08d] Ernesto Jiménez-Ruiz, Bernardo Cuenca Grau, Ulrike Sattler, Thomas Schneider, and Rafael Berlanga Llavori. Safe and Economic Re-Use of Ontologies: A Logic-Based Methodology and Tool Support. Technical Report. http://www.cs.man.ac.uk/s̃chneidt/publ/safe-eco-reuse-report.pdf, November, 2008.

[JRL06] Ernesto Jiménez-Ruiz and Rafael Berlanga Llavori. A View-Based Methodology for Collaborative Ontology Engineering: An Approach for Complex Applications (VIMethCOE). In *15th IEEE International Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises (WETICE)*, pages 376–381. IEEE Computer Society, 2006.

[JRLNS07] Ernesto Jiménez-Ruiz, Rafael Berlanga Llavori, Victoria Nebot, and Ismael Sanz. OntoPath: A language for retrieving ontology fragments. In *The 6th International Conference on Ontologies, DataBases, and Applications of Semantics (ODBASE), OTM Conferences*, volume 4803 of *Lecture Notes in Computer Science*, pages 897–914. Springer, 2007.

[JRLS+05] Ernesto Jiménez-Ruiz, Rafael Berlanga Llavori, Ismael Sanz, María José Aramburu Cabo, and Roxana Dánger. Ontopathview: A simple view definition language for the collaborative development of ontologies. In *Proceedings of the 8th International Conference of the Catalan Association of Artificial Intelligence, CCIA*, volume 131 of *Frontiers in Artificial Intelligence and Applications*, pages 429–436. IOS Press, 2005.

[JRNL+07] Ernesto Jiménez-Ruiz, Victoria Nebot, Rafael Berlanga Llavori, Ismael Sanz, and Alfonso Rios. A Protégé plug-in-based system to manage and query large domain ontologies. In *10th International Protégé Conference*, 2007.

[JYJRBRS07] Antonio Jimeno-Yepes, Ernesto Jimenez-Ruiz, Rafael Berlanga, and Dietrich Rebholz-Schuhmann. Towards enrichement of a biomedical ontology based on text mining. Technical report: http://krono.act.uji.es/publications/techrep/tkbg-ebi-report, 2007.

[JYJRBRS09a] A. Jimeno-Yepes, E. Jimenez-Ruiz, R. Berlanga, and D. Rebholz-Schuhmann. Health-e-child terminological resources: Vocabulary, Thesaurus (HeCth) and Ontology. `http://krono.act.uji.es/people/Ernesto/hec-thesaurus`, 2009.

[JYJRBRS09b] Antonio Jimeno-Yepes, Ernesto Jimenez-Ruiz, Rafael Berlanga, and Dietrich Rebholz-Schuhmann. Reuse of terminological resources for efficient ontological engineering in life sciences. *BMC Bioinformatics*, 10(Suppl 10):S4, 2009.

[JYJRL+08] Antonio Jimeno-Yepes, Ernesto Jimenez-Ruiz, Vivian Lee, Sylvain Gaudan, Rafael Berlanga, and Dietrich Rebholz-Schuhmann. Assessment of disease named entity recognition on a corpus of annotated sentences. *BMC Bioinformatics*, 9(Suppl 3):S3, 2008.

[JYM$^+$08] Benjamin Johnston, Fangkai Yang, Rogan Mendoza, Xiaoping Chen, and Mary-Anne Williams. Ontology based object categorization for robots. In *PAKM '08: Proceedings of the 7th International Conference on Practical Aspects of Knowledge Management*, volume 5345 of *LNCS*, pages 219–231, Berlin, Heidelberg, 2008. Springer-Verlag.

[KAPS08] Petr Kremen, Jan Abrahamcik, Jaromir Pufler, and Marek Smid. Owldiff system, 2008. Available at: `http://krizik.felk.cvut.cz/km/owldiff/index.html` and `http://sourceforge.net/projects/owldiff`.

[KEV07] Jean François Djoufak Kengue, Jérôme Euzenat, and Petko Valtchev. Ola in the OAEI 2007 evaluation contest. In *Proceedings of the 2nd International Workshop on Ontology Matching, OM*, volume 304 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2007.

[KFKO02] Michel C. A. Klein, Dieter Fensel, Atanas Kiryakov, and Damyan Ognyanov. Ontology versioning and change detection on the web. In *13th International Conference on Knowledge Engineering and Knowledge Management, EKAW*, volume 2473 of *Lecture Notes in Computer Science*, pages 197–212. Springer, 2002.

[Kle04] Michel C. A. Klein. Change management for distributed ontologies, 2004. PhD thesis, Vrije Universiteit, Amsterdam.

[KN03] Michel C. A. Klein and Natalya Fridman Noy. A component-based framework for ontology evolution. In *Proceedings of the IJCAI Workshop: Ontologies and Distributed Systems*, 2003.

[Kot08] Konstantinos Kotis. On supporting HCOME-3O ontology argumentation using semantic wiki technology. In *On the Move to Meaningful Internet Systems: OTM 2008 Workshops*, pages 193–199, 2008.

[KPHS07] Aditya Kalyanpur, Bijan Parsia, Matthew Horridge, and Evren Sirin. Finding all justifications of OWL DL entailments. In *ISWC 2007*, pages 267–280, 2007.

[KPSG06] Aditya Kalyanpur, Bijan Parsia, Evren Sirin, and Bernardo Cuenca Grau. Repairing unsatisfiable concepts in OWL ontologies. In *ESWC 2006*, volume 4011 of *LNCS*, pages 170–184. Springer, 2006.

[KPSH05] Aditya Kalyanpur, Bijan Parsia, Evren Sirin, and James A. Hendler. Debugging unsatisfiable classes in OWL ontologies. *J. Web Sem.*, 3(4):268–293, 2005.

[KR70] Werner Kunz and Horst W. J. Rittel. *Issues as Elements of Information Systems*. Center for Planning and Development Research, University of California at Berkeley, 1970.

[KS03] Yannis Kalfoglou and Marco Schorlemmer. Ontology mapping: the state of the art. *Knowl. Eng. Rev.*, 18(1):1–31, 2003.

[KS$^+$08] Warren Kibbe, Lynn Schriml, et al. Disease Ontology (DO), 2008. Disease Ontology Wiki : `http://diseaseontology.sourceforge.net/`.

[KSKM07] Kouji Kozaki, Eiichi Sunagawa, Yoshinobu Kitamura, and Riichiro Mizoguchi. A framework for cooperative ontology construction based on dependency management of modules. In *Proceedings of the First International Workshop on Emergent Semantics and Ontology Evolution, ESOE 2007, co-located with ISWC 2007 + ASWC 2007*, pages 33–44, 2007.

[KV06] Konstantinos Kotis and George A. Vouros. Human-centered ontology engineering: The HCOME methodology. *Knowl. Inf. Syst.*, 10(1):109–131, 2006.

[KWW08] Boris Konev, Dirk Walther, and Frank Wolter. The logical difference problem for description logic terminologies. In *IJCAR-08*, volume 5195 of *LNCS*, pages 259–274. Springer, 2008.

[KWZ08] Roman Kontchakov, Frank Wolter, and Michael Zakharyaschev. Can you tell the difference between DL-Lite ontologies? In *Proceedings of the Eleventh International Conference on Principles of Knowledge Representation and Reasoning, KR*, pages 285–295. AAAI Press, 2008.

[LAF$^+$05] Lee Lacy, Gabriel Aviles, Karen Fraser, William Gerber, Alice M. Mulvehill, and Robert Gaskill. Experiences using OWL in military applications. In *Proceedings of the Workshop on OWL: Experiences and Directions, OWLED*, volume 188 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2005.

[Lam06] Patrick Lambrix. Ontologies in bioinformatics and systems biology. In *Artificial Intelligence Methods and Tools for Systems Biology*, Amsterdam, The Netherlands, The Netherlands, 2006. Springer Netherlands.

[LASPPJR08] Rafael Berlanga Llavori, Henry Anaya-Sánchez, Aurora Pons-Porrata, and Ernesto Jiménez-Ruiz. Conceptual subtopic identification in the medical domain. In *Advances in Artificial Intelligence - IBERAMIA 2008, 11th Ibero-American Conference on AI*, volume 5290 of *Lecture Notes in Computer Science*, pages 312–321. Springer, 2008.

[LB87] Hector J. Levesque and Ronald J. Brachman. Expressiveness and tractability in knowledge representation and reasoning. *Computational Intelligence*, 3:78–93, 1987.

[LD$^+$07] S. Lesteven, S. Derriere, et al. Ontologies for astronomy. In *Library and Information Services in Astronomy V: Common Challenges, Uncommon Solutions*. Astronomical Society of the Pacific Conference Series, 2007.

[LD08] Pieter De Leenheer and Christophe Debruyne. DOGMA-MESS: A tool for fact-oriented collaborative ontology evolution. In *On the Move to Meaningful Internet Systems: OTM 2008 Workshops*, volume 5333 of *Lecture Notes in Computer Science*, pages 797–806. Springer, 2008.

[LdMM07] Pieter De Leenheer, Aldo de Moor, and Robert Meersman. Context dependency management in ontology engineering: A formal approach. *J. Data Semantics*, 8:26–56, 2007.

[LED$^+$10] Deryle W. Lonsdale, David W. Embley, Yihong Ding, Li Xu, and Martin Hepp. Reusing ontologies and language components for ontology generation. *Data Knowl. Eng.*, 69(4):318–330, 2010.

[LHL01] Berners T. Lee, J. Hendler, and O. Lassila. The semantic web. *Scientific American*, May 2001.

[LJRCM09] Zoe Falomir Llansola, Ernesto Jiménez-Ruiz, Lledó Museros Cabedo, and María Teresa Escrig Monferrer. An ontology for qualitative description fo images. In *COSIT 2009 Workshop proceedings, Spatial and Temporal Reasoning for Ambient Intelligence Systems*, 2009.

[LM01] Ora Lassila and Deborah McGuinness. The Role of Frame-Based Representation on the Semantic Web. *Electronic Transactions on Artificial Intelligence*, 6(5), 2001.

[LM08] Pieter De Leenheer and Tom Mens. Ontology evolution. In *Ontology Management, Semantic Web, Semantic Web Services, and Business Applications*, volume 7 of *Semantic Web And Beyond Computing for Human Experience*, pages 131–176. Springer, 2008.

[LS06] Boris Lauser and Margherita Sini. From AGROVOC to the agricultural ontology service/concept server: an OWL model for creating ontologies in the agricultural domain. In *DCMI '06: Proceedings of the 2006 international conference on Dublin Core and Metadata Applications*, pages 76–88. Dublin Core Metadata Initiative, 2006.

[LS07] Joanne Luciano and Robert Stevens. e-Science and biological pathway semantics. *BMC Bioinformatics*, 8(Suppl 3):S3, 2007.

[LWW07] Carsten Lutz, Dirk Walther, and Frank Wolter. Conservative extensions in expressive description logics. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence, IJCAI*, pages 453–458, 2007.

[MA09] James G. Mork and Alan R. Aronson. Filtering the UMLS metathesaurus for MetaMap. Technical report: http://skr.nlm.nih.gov/papers/references/filtering09.pdf, 2009.

[MBB06] Fleur Mougin, Anita Burgun, and Olivier Bodenreider. Using WordNet to improve the mapping of data elements to UMLS for data sources integration. *AMIA Symposium*, pages 574–578, 2006.

[MBB09] Fleur Mougin, Olivier Bodenreider, and Anita Burgun. Analyzing polysemous concepts from a clinical perspective: Application to auditing concept categorization in the umls. *Journal of Biomedical Informatics*, 42(3):440–451, 2009.

[McC89] Alexa T. McCray. The UMLS Semantic Network. In Kingsland LC, editor, *Proc 13th Annu Symp Comput App Med Care*, pages 503–507. IEEE Computer Society Press, 1989.

[MCH$^+$09] Boris Motik, Bernardo Cuenca Grau, Ian Horrocks, Zhe Wu, Achille Fokoue, and Carsten Lutz. OWL 2 Web Ontology Language: Profiles. *W3C Recommendation*, 2009. `http://www.w3.org/TR/owl2-profiles/`.

[MFRW00] Deborah L. McGuinness, Richard Fikes, James Rice, and Steve Wilder. An environment for merging and testing large ontologies. In *Proc. 17th Intl. Conf. on Principles of Knowledge Representation and Reasoning, KR*, pages 483–493, 2000.

[MGHP09] C. Paul Morrey, James Geller, Michael Halper, and Yehoshua Perl. The Neighborhood Auditing Tool: A hybrid interface for auditing the UMLS. *Journal of Biomedical Informatics*, 42(3):468–489, 2009.

[Mil95] George A. Miller. WordNet: a lexical database for English. *Commun. ACM*, 38(11):39–41, 1995.

[MM07] Dilvan A. Moreira and Mark A. Musen. OBO to OWL: a Protégé owl tab to read/save obo ontologies. *Bioinformatics*, 23(14):1868–1870, 2007.

[MMB$^+$05] Alistair Miles, Brian Matthews, Dave Beckett, Dan Brickley, Michael Wilson, and Nikki Rogers. SKOS: A language to describe simple knowledge structures for the web. In *Proc of the XTech Conference: XML, the Web and beyond.*, 2005.

[MNA$^+$99] Stuart Nelson Md, Stuart J. Nelson, Alan R. Aronson, Tamas E. Doszkocs, and H. Florence Chang Ms. Automated assignment of medical subject headings. In *Proceedings of the American Medical Informatics Association (AMIA) Annual Symposium*, 1999.

[MPSCG09] Boris Motik, Peter F. Patel-Schneider, and Bernardo Cuenca-Grau. OWL 2 Web Ontology Language Direct Semantics. *W3C Recommendation*, 2009. `http://www.w3.org/TR/owl2-semantics/`.

[MPSP09] Boris Motik, Peter F. Patel-Schneider, and Bijan Parsia. OWL 2 web ontology language structural specification and functional-style syntax. *W3C Recommendation*, 2009. `http://www.w3.org/TR/owl2-syntax/`.

[MS09] Christian Meilicke and Heiner Stuckenschmidt. An efficient method for computing alignment diagnoses. In *Third International Conference on Web Reasoning and Rule Systems, RR*, pages 182–196, 2009.

[MSH09] Boris Motik, Rob Shearer, and Ian Horrocks. Hypertableau Reasoning for Description Logics. *Journal of Artificial Intelligence Research*, 36:165–228, 2009.

[MST07] Christian Meilicke, Heiner Stuckenschmidt, and Andrei Tamilin. Repairing ontology mappings. In *Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence*, pages 1408–1413. AAAI Press, 2007.

[MST08] Christian Meilicke, Heiner Stuckenschmidt, and Andrei Tamilin. Supporting manual mapping revision using logical reasoning. In *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence, AAAI*, pages 1213–1218, 2008.

[MST09] Christian Meilicke, Heiner Stuckenschmidt, and Andrei Tamilin. Reasoning Support for Mapping Revision. *J Logic Computation*, 19(5):807–829, 2009.

[MSvZ09] Christian Meilicke, Heiner Stuckenschmidt, and Ondřej Šváb Zamazal. A reasoning-based support tool for ontology mapping evaluation. In *ESWC 2009 Heraklion: Proceedings of the 6th European Semantic Web Conference on The Semantic Web*, pages 878–882, Berlin, Heidelberg, 2009. Springer-Verlag.

[MW75] M. Minsky and P. H. Winston. A framework for representing knowledge. In *The Psychology of Computer Vision*, New York, 1975. McGraw Hill.

[NB03] Daniele Nardi and Ronald J. Brachman. An introduction to Description Logics. In Baader et al. [BCM$^+$03], pages 5–44.

[NCLM06] Natalya Fridman Noy, Abhita Chugh, William Liu, and Mark A. Musen. A framework for ontology evolution in collaborative environments. In *ISWC-06*, pages 544–558, 2006.

[NK04] Natalya F. Noy and Michel C. A. Klein. Ontology evolution: Not the same as schema evolution. *Knowledge and Information Systems*, 6:428–440, 2004.

[NKKM04] Natalya Fridman Noy, Sandhya Kunnatur, Michel C. A. Klein, and Mark A. Musen. Tracking changes during ontology evolution. In *The Semantic Web - ISWC 2004: Third International Semantic Web Conference*, pages 259–273, 2004.

[NL09] Victoria Nebot and Rafael Berlanga Llavori. Efficient retrieval of ontology fragments using an interval labeling scheme. *Inf. Sci.*, 179(24):4151–4173, 2009.

[NM04] Natalya Fridman Noy and Mark A. Musen. Specifying ontology views by traversal. In *Third International Semantic Web Conference (ISWC), Hiroshima, Japan*, volume 3298 of *Lecture Notes in Computer Science*, pages 713–725. Springer, 2004.

[NM09] Natalya Fridman Noy and Mark A. Musen. Traversing ontologies to extract views. In Stuckenschmidt et al. [SPS09], pages 245–260.

[NTdCM08] Natalya Fridman Noy, Tania Tudorache, Sherri de Coronado, and Mark A. Musen. Developing biomedical ontologies collaboratively. In *AMIA 2008*, 2008.

[oM10] National Library of Medicine. UMLS reference manual. http://web.ncbi.nlm.nih.gov/bookshelf/br.fcgi?book=nlmumls, 2010.

[ope10] Label selection problem., 2010. Open Galen project: `http://www.opengalen.org/themodel/labels.html`.

[Pal09] Raul Palma. Ontology metadata management in distributed environments, 2009. PhD thesis, Universidad Politécnica de Madrid.

[PHCGP09] Raul Palma, Peter Haase, Oscar Corcho, and Asunción Gómez-Pérez. Change representation for OWL 2 ontologies. In *Proceedings of the Sixth OWLED Workshop on OWL: Experiences and Directions, collocated with the 8th International Semantic Web Conference (ISWC)*, 2009.

[PHJ08] Raul Palma, Peter Haase, and Qiu Ji. D1.3.2. Change management to support collaborative workflows. NeOn Deliverable available at: http://www.neon-project.org/, December, 2008.

[PJYLRS08] Piotr Pezik, Antonio Jimeno-Yepes, Vivian Lee, and Dietrich Rebholz-Schuhmann. Static dictionary features for term polysemy identification. In *Language Resources and Evaluation Conference (LREC), workshop on Building and evaluating resources for biomedical text mining*, 2008.

[PS09] Christine Parent and Stefano Spaccapietra. An overview of modularity. In Stuckenschmidt et al. [SPS09], pages 5–23.

[PSHH04] Peter F. Patel-Schneider, Patrick Hayes, and Ian Horrocks. Web Ontology Language OWL Abstract Syntax and Semantics. *W3C Recommendation*, 2004. `http://www.w3.org/TR/owl-semantics/`.

[PT06] Peter Plessers and Olga De Troyer. Resolving inconsistencies in evolving ontologies. In *Proc. 3rd European Semantic Web Conference (ESWC 2006)*, pages 200–214. Springer, 2006.

[PTS09] Helena Sofia Pinto, Christoph Tempich, and Steffen Staab. Ontology engineering and evolution in a distributed world using DILIGENT. In Steffen Staab and Rudi Studer, editors, *Handbook on Ontologies*, International Handbooks on Information Systems, pages 153–176. Springer, 2009.

[RB01] Erhard Rahm and Philip A. Bernstein. A survey of approaches to automatic schema matching. *VLDB J.*, 10(4):334–350, 2001.

[Rei87] Raymond Reiter. A theory of diagnosis from first principles. *Artif. Intell.*, 32(1):57–95, 1987.

[RFB$^+$92] Michael Genesereth Richard, Richard E. Fikes, Ronald Brachman, Thomas Gruber, Patrick Hayes, Reed Letsinger, Vladimir Lifschitz, Robert Macgregor, John Mccarthy, Peter Norvig, and Ramesh Patil. Knowledge Interchange Format version 3.0 reference manual, 1992.

[Rio] Alfonso Rios. G Platform (MaatG). Technical Reports available at: http://www.maatg.com.

[Ros01] Cornelius Rosse. Terminologia anatomica: Considered from the perspective of next-generation knowledge sources. *Clinical Anatomy*, 14(2):120–133, 2001.

[RR06] Alan L. Rector and Jeremy Rogers. Ontological and practical issues in using a description logic to represent medical concept systems: Experience from GALEN. In *proc. of Reasoning Web*, pages 197–231, 2006.

[RSDT08] Timothy Redmond, Michael Smith, Nick Drummond, and Tania Tudorache. Managing change: An ontology version control system. In *Proceedings of the Fifth OWLED Workshop on OWL: Experiences and Directions, collocated with the 7th International Semantic Web Conference (ISWC)*, 2008.

[SAM09] Sonya E. Shooshan, Alan R. Aronson, and James G. Mork. Ambiguity in the UMLS Metathesaurus. Technical report: http://skr.nlm.nih.gov/papers/references/ambiguity09.pdf, 2009.

[SAR$^+$07] Barry Smith, Michael Ashburner, Cornelius Rosse, Jonathan Bard, William Bug, Werner Ceusters, Louis J. Goldberg, et al. The OBO foundry: coordinated evolution of ontologies to support biomedical data integration. *Nature biotechnology*, 25(11):1251–1255, November 2007.

[SC03] Stefan Schlobach and Ronald Cornet. Non-standard reasoning services for the debugging of description logic terminologies. In Georg Gottlob and Toby Walsh, editors, *IJCAI*, pages 355–362. Morgan Kaufmann, 2003.

[SCM03] Ulrike Sattler, Diego Calvanese, and Ralf Molitor. Relationships with other formalisms. In Baader et al. [BCM$^+$03], pages 142–183.

[SE05] Pavel Shvaiko and Jérôme Euzenat. A survey of schema-based matching approaches. *Journal on Data Semantics IV*, pages 146–171, 2005.

[SE08] Pavel Shvaiko and Jérôme Euzenat. Ten challenges for ontology matching. In *On the Move to Meaningful Internet Systems (OTM Conferences)*, volume 5332 of *Lecture Notes in Computer Science*, pages 1164–1182. Springer, 2008.

[SE10] Pavel Shvaiko and Jérôme Euzenat. Ontology Matching Initiative. http://www.ontologymatching.org/, 2010.

[SEA$^+$02] York Sure, Michael Erdmann, Jurgen Angele, Steffen Staab, Rudi Studer, and Dirk Wenke. OntoEdit: Collaborative ontology development for the semantic web. In *International Semantic Web Conference (ISWC)*, pages 221–235, 2002.

[Sei09] Julian Seidenberg. Web ontology segmentation: Extraction, transformation, evaluation. In Stuckenschmidt et al. [SPS09], pages 211–243.

[SHCvH07] Stefan Schlobach, Zhisheng Huang, Ronald Cornet, and Frank van Harmelen. Debugging incoherent terminologies. *J. Autom. Reasoning*, 39(3):317–349, 2007.

[SJR09] Ismael Sanz and Ernesto Jiménez-Ruiz. Ontologías en informática. In A. Alcina, E. Valero, and E. Rambla, editors, *Terminología y Sociedad del conocimiento*, pages 255–286. Peter Lang, Berna, 2009.

[SK04] Heiner Stuckenschmidt and Michel C. A. Klein. Structure-based partitioning of large concept hierarchies. In *Third International Semantic Web Conference (ISWC), Hiroshima, Japan*, volume 3298 of *Lecture Notes in Computer Science*, pages 289–303. Springer, 2004.

[SKKM03] Eiichi Sunagawa, Kouji Kozaki, Yoshinobu Kitamura, and Riichiro Mizoguchi. An environment for distributed ontology development based on dependency management. In *The Semantic Web - ISWC 2003, Second International Semantic Web Conference*, pages 453–468, 2003.

[SMH08] Rob Shearer, Boris Motik, and Ian Horrocks. HermiT: A highly-efficient OWL reasoner. In *Proceedings of the Fifth OWLED Workshop on OWL: Experiences and Directions, OWLED*, volume 432 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2008.

[SMS09] Daniel Schober, James Malone, and Robert Stevens. Observations in collaborative ontology editing using Collaborative Protégé. In *Proceedings of the Workshop on Collaborative Construction, Management and Linking of Structured Knowledge (CK2009), collocated with the 8th International Semantic Web Conference (ISWC)*, 2009.

[SNNB08] Rodolfo Stecher, Claudia Niederée, Wolfgang Nejdl, and Paolo Bouquet. Adaptive ontology re-use: finding and re-using sub-ontologies. *IJWIS*, 4(2):198–214, 2008.

[SNTM08] Abraham Sebastian, Natalya Fridman Noy, Tania Tudorache, and Mark A. Musen. A generic ontology for collaborative ontology-development workflows. In *16th International Conference on Knowledge Engineering, EKAW*, volume 5268 of *Lecture Notes in Computer Science*, pages 318–328, 2008.

[Spa00] K. Spackman. SNOMED RT and SNOMED CT. promise of an international clinical ontology. *M.D. Computing*, 17, 2000.

[SPG$^+$07] Evren Sirin, Bijan Parsia, Bernardo Cuenca Grau, Aditya Kalyanpur, and Yarden Katz. Pellet: A practical OWL-DL reasoner. *J. Web Sem.*, 5(2):51–53, 2007.

[SPS09] Heiner Stuckenschmidt, Christine Parent, and Stefano Spaccapietra, editors. *Modular Ontologies: Concepts, Theories and Techniques for Knowledge Modularization*, volume 5445 of *Lecture Notes in Computer Science*. Springer, 2009.

[SQJH08] Boontawee Suntisrivaraporn, Guilin Qi, Qiu Ji, and Peter Haase. A modularization-based approach to finding all justifications for OWL DL entailments. In John Domingue and Chutiporn Anutariya, editors, *ASWC*, volume 5367 of *Lecture Notes in Computer Science*, pages 1–15. Springer, 2008.

[SR06] Julian Seidenberg and Alan L. Rector. Web ontology segmentation: analysis, classification and use. In *Proceedings of the 15th international conference on World Wide Web, WWW*, pages 13–22, 2006.

[SR07a] Julian Seidenberg and Alan L. Rector. A methodology for asynchronous multi-user editing of semantic web ontologies. In Derek H. Sleeman and Ken Barker, editors, *K-CAP*, pages 127–134. ACM, 2007.

[SR07b] Julian Seidenberg and Alan L. Rector. The state of multi-user ontology engineering. In Bernardo Cuenca Grau, Vasant Honavar, Anne Schlicht, and Frank Wolter, editors, *2nd International Workshop on Modular Ontologies, WoMO*, volume 315 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2007.

[SS09] Heiner Stuckenschmidt and Anne Schlicht. Structure-based partitioning of large ontologies. In Stuckenschmidt et al. [SPS09], pages 187–210.

[SSK05] Giorgos Stoilos, Giorgos B. Stamou, and Stefanos D. Kollias. A string metric for ontology alignment. In *Proc. of the International Semantic Web Conference (ISWC)*, pages 624–637, 2005.

[ST09] Luciano Serafini and Andrei Tamilin. Composing modular ontologies with distributed description logics. In Stuckenschmidt et al. [SPS09], pages 321–347.

[Sto04] Ljiljana Stojanovic. Methods and tools for ontology evolution, 2004. PhD thesis, University of Karlsruhe.

[TAM+09] Syed Hamid Tirmizi, Stuart Aitken, Dilvan A. Moreira, Chris Mungall, Juan Sequeda, Nigam H. Shah, and Daniel P. Miranker. OBO & OWL: Roundtrip ontology transformations. In *Proceedings of the 2nd International Workshop on Semantic Web Applications and Tools for Life Sciences (SWAT4LS 2009), Amsterdam, The Netherlands*, 2009.

[TH06] Dmitry Tsarkov and Ian Horrocks. FaCT++ description logic reasoner: System description. In Furbach and Shankar [FS06], pages 292–297.

[TNTM08] Tania Tudorache, Natalya Fridman Noy, Samson W. Tu, and Mark A. Musen. Supporting collaborative ontology development in Protégé. In *International Semantic Web Conference (ISWC)*, pages 17–32, 2008.

[TPSS05] Christoph Tempich, Helena Sofia Pinto, York Sure, and Steffen Staab. An Argumentation Ontology for DIstributed, Loosely-controlled and evolvInG Engineering processes of oNTologies (DILIGENT). In *The Semantic Web: Research and Applications, Second European Semantic Web Conference, ESWC*, pages 241–256, 2005.

[TSK09] Eleni Tsalapati, Giorgos B. Stamou, and Giorgos Koletsos. A method for approximation to ontology reuse problem. In *Proceedings of the International Conference on Knowledge Engineering and Ontology Development (KEOD)*, pages 416–419. INSTICC Press, 2009.

[TSL+07] Christoph Tempich, Elena Simperl, Markus Luczak, Rudi Studer, and H. Sofia Pinto. Argumentation-based ontology engineering. *IEEE Intelligent Systems*, 22(6):52–59, 2007.

[TvOS09] Anna Tordai, Jacco van Ossenbruggen, and Guus Schreiber. Combining vocabulary alignment techniques. In *Proceedings of the 5th International Conference on Knowledge Capture, K-CAP*, pages 25–32. ACM, 2009.

[TvOSW10] Anna Tordai, Jacco van Ossenbruggen, Guus Schreiber, and Bob Wielinga. Aligning large SKOS-like vocabularies: Two case studies. In *Proceedings of the 7th Extended Semantic Web Conference, ESWC*, Lecture Notes in Computer Science. Springer, 2010.

[TZH07] A. Tsymbal, S. Zillner, and M. Huber. Ontology-supported machine learning and decision support in biomedicine. *Lecture Notes in Computer Science*, 4544:156, 2007.

[WHB07] Yimin Wang, Peter Haase, and Jie Bao. A survey of formalisms for modular ontologies. In *SWeCKa 2007: Proc. of the IJCAI-2007 Workshop on Semantic Web for Collaborative Knowledge Acquisition , Hyderabad, India*, 2007.

[WHR$^+$05] Hai Wang, Matthew Horridge, Alan L. Rector, Nick Drummond, and Julian Seidenberg. Debugging OWL-DL ontologies: A heuristic approach. In Yolanda Gil, Enrico Motta, V. Richard Benjamins, and Mark A. Musen, editors, *International Semantic Web Conference*, volume 3729 of *Lecture Notes in Computer Science*, pages 745–757. Springer, 2005.

[Wil08] Mary-Anne Williams. Representation = grounded information. In *PRICAI '08: Proceedings of the 10th Pacific Rim International Conference on Artificial Intelligence*, volume 5351 of *LNCS*, pages 473–484, Berlin, Heidelberg, 2008. Springer-Verlag.

[WKG$^+$08] David S. Wishart, Craig Knox, Anchi Guo, Dean Cheng, Savita Shrivastava, Dan Tzur, Bijaya Gautam, and Murtaza Hassanali. Drugbank: a knowledgebase for drugs, drug actions and drug targets. *Nucleic Acids Research*, 36(Database-Issue):901–906, 2008.

[Woo75] William A. Woods. What's in a link: Foundations for semantic networks. In D. Bobrow and A. Collins, editors, *Representation and Understanding: Studies in Cognitive Science*. Academic Press, New York, NY, 1975.