Safe and Economic Re-Use of Ontologies: A Logic-Based Methodology and Tool Support

Technical Report, 5th November 2008

Ernesto Jiménez-Ruiz[1], Bernardo Cuenca Grau[2], Ulrike Sattler[3], Thomas Schneider[3], and Rafael Berlanga[1]

[1] Universitat Jaume I, Spain, {berlanga,ejimenez}@uji.es
[2] University of Oxford, UK, berg@comlab.ox.ac.uk
[3] University of Manchester, UK, {sattler,schneider}@cs.man.ac.uk

Abstract Driven by application requirements and using well-understood theoretical results, we describe a novel methodology and a tool for modular ontology design. We support the user in the *safe* use of imported symbols and in the *economic* import of the relevant part of the imported ontology. Both features are supported in a well-understood way: safety guarantees that the semantics of imported concepts is not changed, and economic import guarantees that no difference can be observed between importing the whole ontology and importing the relevant part.

1 Motivation

Ontology design and maintenance require an expertise in both the domain of application and the ontology language. Realistic ontologies typically model different aspects of an application domain at various levels of granularity; prominent examples are the National Cancer Institute Ontology (NCI)[4] [1], which describes diseases, drugs, proteins, etc., and GALEN[5], which represents knowledge mainly about the human anatomy, but also about other domains such as drugs.

Ontologies such as NCI and GALEN are used in bio-medical applications as *reference ontologies*, i.e., ontology developers reuse these ontologies and customise them for their specific needs. For example, ontology designers use concepts[6] from NCI or GALEN and refine them (e.g., add new sub-concepts), generalise them (e.g., add new super-concepts), or refer to them when expressing a property of some other concept (e.g., define the concept Polyarticular_JRA by referring to the concept Joint from GALEN).

[4] Online browser: `http://nciterms.nci.nih.gov/NCIBrowser/Dictionary.do`, latest version: `ftp://ftp1.nci.nih.gov/pub/cacore/EVS/NCI_Thesaurus`
[5] `http://www.co-ode.org/galen`
[6] We use the Description Logic terms "concept" and "role" instead of the OWL terms "class" and "property".

One of such use cases is the development within the Health-e-Child project of an ontology, called JRAO, to describe a kind of arthritis called JRA (Juvenile Rheumatoid Arthritis).[7] Following the ILAR[8], JRAO describes the kinds of JRA. Those are distinguished by several factors such as the joints affected or the occurrence of fever, and each type of JRA requires a different treatment. GALEN and NCI contain information that is relevant to JRA, such as detailed descriptions of the human joints as well as diseases and their symptoms. Figure 1 gives a fragment of NCI that defines JRA. It also shows our reuse scenario, where $C_1, \ldots, C_7$ refer to the kinds of JRA to be defined in JRAO.

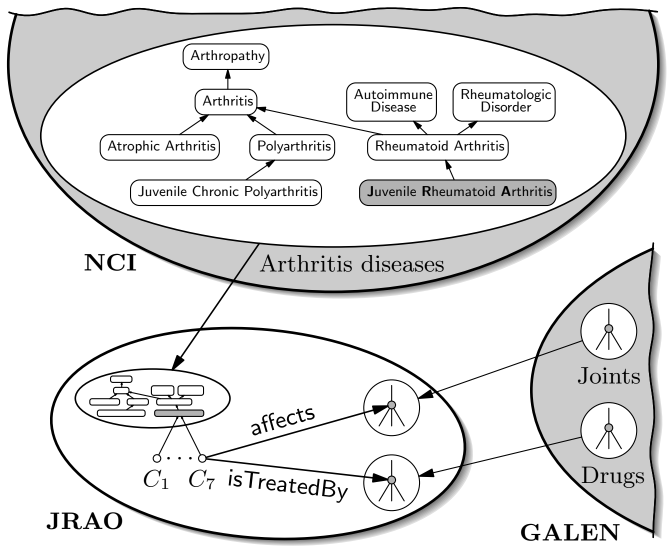

Figure 1. Constructing the ontology JRAO reusing fragments of GALEN and NCI

The JRAO developers want to reuse knowledge from NCI and GALEN for three reasons: (a) they want to save time through reusing existing ontologies rather than writing their own; (b) they value knowledge that is commonly accepted by the community and used in similar applications; (c) they are not experts in all areas covered by NCI and GALEN.

Currently, GALEN, NCI, and JRAO are written in OWL DL [2], and thus they come with a logic-based semantics, which allows for powerful reasoning services for classification and consistency checking. Thus, ontology reuse should take into account the semantics and, more precisely, should provide the following two guarantees. First, when reusing knowledge from NCI and GALEN, the developers of JRAO do not want to change the original meaning of the reused

[7] See `http://www.health-e-child.org`. This project aims at creating a repository of ontologies that can be used by clinicians in various applications.

[8] Int. League of Associations for Rheumatology `http://www.ilarportal.org/`

concepts. For example, due to (b) and (c) above, if it followed from the union of JRAO and NCI that JRA is a genetic disorder, then this should also follow from NCI alone. Second, only small parts of large ontologies like NCI and GALEN are relevant to the sub-types of JRA. For efficiency and succinctness, the JRAO developers want to import only those axioms from NCI and GALEN that are relevant for JRAO. By importing only fragments of NCI and GALEN, one should not lose important information; for example, if it follows from the union of JRAO and NCI that JRA is a rheumatologic disorder, then this also follows from the union of JRAO and the chosen fragment of NCI.

Our scenario has two main points in common with other ontology design scenarios: the ontology developer wants to reuse knowledge without changing it, and also to import only the relevant parts of an existing ontology. To support these scenarios whilst providing the two above guarantees, a logic-based approach to reuse is required. Current tools that support reuse, however, do not implement a logic-based solution and thus do not provide the above guarantees—and neither do existing guidelines and "best practices" for ontology design.

In this paper, we propose a methodology for ontology design in scenarios involving reuse which is based on a well-understood logic-based framework [3]. We describe a tool that implements this methodology and report on experiments.

2 Preliminaries on Modularity

Based on the application scenario in Section 1, we define the notions of a *conservative extension*, *safety*, and *module* [4, 3]. For simplicity of the presentation, we restrict ourselves to the description logic $\mathcal{SHIQ}$, which covers most of OWL DL [2]. Therefore the ontologies, entailments and signatures we consider are relative to $\mathcal{SHIQ}$. Some extra care needs to be taken to extend the results mentioned in this section to $\mathcal{SHOIQ}$ and therefore OWL DL [3].

2.1 The Notions of Conservative Extension and Safety

As mentioned in Section 1, when reusing knowledge from NCI and GALEN, the developer of JRAO should not change the original meaning of the reused concepts. This requirement can be formalised using the notion of a *conservative extension* [4, 5]. In the following, we use $\mathsf{Sig}()$ to denote the signature of an ontology or an axiom.[9]

Definition 1 (Conservative Extension). *Let $\mathcal{O}_1 \subseteq \mathcal{O}$ be ontologies, and $\mathbf{S}$ a signature. We say that $\mathcal{O}$ is an* $\mathbf{S}$-conservative extension *of $\mathcal{O}_1$ if, for every axiom α with $\mathsf{Sig}(\alpha) \subseteq \mathbf{S}$, we have $\mathcal{O} \models \alpha$ iff $\mathcal{O}_1 \models \alpha$; $\mathcal{O}$ is a* conservative extension *of $\mathcal{O}_1$ if $\mathcal{O}$ is an $\mathbf{S}$-conservative extension of $\mathcal{O}_1$ for $\mathbf{S} = \mathsf{Sig}(\mathcal{O}_1)$.*[10]

[9] $\mathcal{SHIQ}$-axioms are concept or role inclusions $C \sqsubseteq D$, $R \sqsubseteq S$, or transitivity statements $\mathsf{Trans}(R)$.

Definition 1 applies to our example as follows: $\mathcal{O}_1 = \text{NCI}$ is the ontology to be reused, $\mathcal{O}$ is the union of JRAO and NCI, $\mathbf{S}$ represents the symbols reused from NCI, such as JRA and $\mathsf{Rheumatologic\_Disorder}$, and α stands for any axiom over the reused symbols only, e.g., $\mathsf{JRA} \sqsubseteq \mathsf{Rheumatologic\_Disorder}$.

Proposition 2 (Transitivity of Conservative Extensions). *Let $\mathcal{O}$ be an $\mathbf{S}$-conservative extension of $\mathcal{O}_1$ and $\mathcal{O}_1$ an $\mathbf{S}$-conservative extension of $\mathcal{O}_2$, then $\mathcal{O}$ is an $\mathbf{S}$-conservative extension of $\mathcal{O}_2$.*

Proof. Let $\mathcal{O}$ be an $\mathbf{S}$-conservative extension of $\mathcal{O}_1$; therefore for every axiom α with $\mathsf{Sig}(\alpha) \subseteq \mathbf{S}$, we have $\mathcal{O} \models \alpha$ iff $\mathcal{O}_1 \models \alpha$. $\mathcal{O}_1$ an $\mathbf{S}$-conservative extension of $\mathcal{O}_2$; therefore, $\mathcal{O}_1 \models \alpha$ iff $\mathcal{O}_2 \models \alpha$. Thus, $\mathcal{O} \models \alpha$ iff $\mathcal{O}_2 \models \alpha$ and $\mathcal{O}$ is an $\mathbf{S}$-conservative extension of $\mathcal{O}_2$.

Definition 1 assumes that the ontology to be reused (e.g. NCI) is static. In practice, however, ontologies such as NCI are under development and may evolve beyond the control of the JRAO developers. Thus, it is convenient to keep NCI separate from the JRAO and make its axioms available on demand via a reference such that the developers of the JRAO need not commit to a particular version of NCI. The notion of *safety* [3] 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 3 (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 $\mathcal{O}'$.*

2.2 The Notion of Module

As mentioned in Section 1, by importing only a fragment of NCI and GALEN, one should not lose important information. This idea can be formalised using the notion of a *module* [3]. Intuitively, when checking an arbitrary entailment over the signature of the JRAO, importing a module of NCI should give exactly the same answers as if the whole NCI had been imported.

Definition 4 (Module for a Signature). *Let $\mathcal{O}'_1 \subseteq \mathcal{O}'$ be ontologies and $\mathbf{S}$ a signature. We say that $\mathcal{O}'_1$* is a module for $\mathbf{S}$ in $\mathcal{O}'$ *(or an* $\mathbf{S}$-module in $\mathcal{O}'$*) 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 $\mathcal{O} \cup \mathcal{O}'_1$ for $\mathsf{Sig}(\mathcal{O})$.*

Proposition 5 (Properties of Modules).

1. ***Compactness:*** *Let $\mathcal{O}'_1$ be an $\mathbf{S}$-module in $\mathcal{O}'$ and let $\mathbf{S}_0 \subseteq \mathbf{S}$, then $\mathcal{O}'_1$ is a $\mathbf{S}_0$-module in $\mathcal{O}'$.*
2. ***Transitivity:*** *Let $\mathcal{O}'_1$ be an $\mathbf{S}$-module in $\mathcal{O}'$ and $\mathcal{O}'_2$ a $\mathbf{S}$-module in $\mathcal{O}'_1$, then $\mathcal{O}'_2$ is a $\mathbf{S}$-module in $\mathcal{O}'$.*

[10] $\mathcal{SHIQ}$ is a monotonic logic; hence the "only if" in "$\mathcal{O} \models \alpha$ iff $\mathcal{O}_1 \models \alpha$" is trivial.

Proof. **Compactness:** $\mathcal{O}'_1$ is a $\mathbf{S}$-module in $\mathcal{O}'$; therefore, by Definition 4, we have that for every $\mathcal{L}$-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 conservative extension of $\mathcal{O} \cup \mathcal{O}'_1$ for $\mathsf{Sig}(\mathcal{O})$ w.r.t. $\mathcal{L}$. Since $\mathbf{S}_0 \subseteq \mathbf{S}$, every $\mathcal{O}$ such that $\mathsf{Sig}(\mathcal{O}) \cap \mathsf{Sig}(\mathcal{O}') \subseteq \mathbf{S}_0$ is such that $\mathsf{Sig}(\mathcal{O}) \cap \mathsf{Sig}(\mathcal{O}') \subseteq \mathbf{S}$. Therefore, $\mathcal{O}'_1$ is a $\mathbf{S}_0$-module in $\mathcal{O}'$.
Transitivity: Since $\mathcal{O}'_1$ is a $\mathbf{S}$-module in $\mathcal{O}'$ we have that, by Definition 4, for every $\mathcal{L}$-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 $\mathsf{Sig}(\mathcal{O})$-conservative extension of $\mathcal{O} \cup \mathcal{O}'_1$ w.r.t. $\mathcal{L}$. Since $\mathcal{O}'_2$ a $\mathbf{S}$-module in $\mathcal{O}'_1$, then for every $\mathcal{L}$-ontology $\mathcal{O}$ with $\mathsf{Sig}(\mathcal{O}) \cap \mathsf{Sig}(\mathcal{O}'_1) \subseteq \mathbf{S}$ it holds that $\mathcal{O} \cup \mathcal{O}'_1$ is a $\mathsf{Sig}(\mathcal{O})$-conservative extension of $\mathcal{O} \cup \mathcal{O}'_2$ w.r.t. $\mathcal{L}$. Since $\mathcal{O} \cup \mathcal{O}'$ is a $\mathsf{Sig}(\mathcal{O})$-conservative extension of $\mathcal{O} \cup \mathcal{O}'_1$ and $\mathcal{O} \cup \mathcal{O}'_1$ is a $\mathsf{Sig}(\mathcal{O})$-conservative extension of $\mathcal{O} \cup \mathcal{O}'_2$, by Proposition 2, $\mathcal{O} \cup \mathcal{O}'$ is a $\mathsf{Sig}(\mathcal{O})$-conservative extension of $\mathcal{O} \cup \mathcal{O}'_2$ and therefore $\mathcal{O}'_2$ is a $\mathbf{S}$-module in $\mathcal{O}'$.

The notions of safety and module are related as follows:

Proposition 6 ([3], Safety vs. Modules). *If $\mathcal{O}' \setminus \mathcal{O}'_1$ is safe for $\mathbf{S} \cup \mathsf{Sig}(\mathcal{O}'_1)$, then $\mathcal{O}'_1$ is an $\mathbf{S}$-module in $\mathcal{O}'$.*

2.3 Locality Conditions

The decision problems associated with conservative extensions, safety and modules—i.e., whether $\mathcal{O}$ is an $\mathbf{S}$-conservative extension of $\mathcal{O}_1$, whether $\mathcal{O}$ is safe for $\mathbf{S}$, or whether $\mathcal{O}'_1$ is an $\mathbf{S}$-module in $\mathcal{O}$—are undecidable for $\mathcal{SHOIQ}$ [6, 3]. Sufficient conditions for safety have been proposed: if an ontology satisfies such conditions, then we can guarantee that it is safe, but the converse does not necessarily hold [3]. By means of Proposition 6, such conditions could be used for extracting modules. A particularly useful condition is *locality* [3]: it is widely applicable in practice and it can be checked syntactically.

As mentioned in Section 1, when using a symbol from NCI or GALEN, the JRAO developers may refine or extend its meaning, or refer to it for expressing a property of another symbol. The simultaneous refinement and generalisation of a given "external" symbol, however, may compromise safety. For example, JRAO cannot simultaneously contain the following axioms:

$$\mathsf{Polyarticular\_JRA} \sqsubseteq \underline{\mathsf{JRA}} \quad (\bot\text{-local}) \tag{1}$$
$$\underline{\mathsf{Juvenile\_Chronic\_Polyarthritis}} \sqsubseteq \mathsf{Polyarticular\_JRA} \quad (\top\text{-local}) \tag{2}$$

where the underlined concepts are reused from NCI, see Figure 1. These axioms imply $\mathsf{Juvenile\_Chronic\_Polyarthritis} \sqsubseteq \mathsf{JRA}$, and therefore an ontology containing axioms (1) and (2) is not safe w.r.t. $\mathbf{S} = \{\mathsf{JRA}, \mathsf{Juvenile\_Chronic\_Polyarthritis}\}$. Thus, when designing sufficient conditions for safety, we are faced with a fundamental choice depending on whether the ontology designer wants to reuse or generalise the reused concepts. Each choice leads to a different locality condition.

The following definition introduces these conditions and refers to Figure 2, where C and R denote arbitrary concepts and roles.

Definition 7 (Syntactic $\bot$-Locality and $\top$-Locality). *Let* $\mathbf{S}$ *be a signature. An axiom* α *is* $\bot$-local w.r.t. $\mathbf{S}$ *($\top$-local w.r.t* $\mathbf{S}$*) if* $\alpha \in \mathsf{Ax}(S)$*, as defined in Figure 2 (a) ((b)), where* C *and* R *denote arbitrary concepts and roles. An ontology* $\mathcal{O}$ *is* $\bot$*-local ($\top$-local) w.r.t.* $\mathbf{S}$ *if* α *is* $\bot$*-local ($\top$-local) w.r.t.* $\mathbf{S}$ *for all* $\alpha \in \mathcal{O}$*.*

(a) $\bot$*-Locality*	Let $A^\bot, R^\bot \notin \mathbf{S}$, $C^\bot \in \mathbf{Bot}(\mathbf{S})$, $C_{(i)}^\top \in \mathbf{Top}(\mathbf{S})$, $\bar{n} \in \mathbb{N} \setminus \{0\}$
$\mathbf{Bot}(\mathbf{S})$ ::=	$A^\bot \mid \bot \mid \neg C^\top \mid C \sqcap C^\bot \mid C^\bot \sqcap C \mid \exists R.C^\bot \mid \geqslant \bar{n}\, R.C^\bot \mid \exists R^\bot.C \mid \geqslant \bar{n}\, R^\bot.C$
$\mathbf{Top}(\mathbf{S})$ ::=	$\top \mid \neg C^\bot \mid C_1^\top \sqcap C_2^\top \mid \geqslant 0\, R.C$
$\mathsf{Ax}(\mathbf{S})$ ::=	$C^\bot \sqsubseteq C \mid C \sqsubseteq C^\top \mid R^\bot \sqsubseteq R \mid \mathsf{Trans}(R^\bot)$

(b) $\top$*-Locality*	Let $A^\top, R^\top \notin \mathbf{S}$, $C^\bot \in \mathbf{Bot}(\mathbf{S})$, $C_{(i)}^\top \in \mathbf{Top}(\mathbf{S})$, $\bar{n} \in \mathbb{N} \setminus \{0\}$
$\mathbf{Bot}(\mathbf{S})$ ::=	$\bot \mid \neg C^\top \mid C \sqcap C^\bot \mid C^\bot \sqcap C \mid \exists R.C^\bot \mid \geqslant \bar{n}\, R.C^\bot$
$\mathbf{Top}(\mathbf{S})$ ::=	$A^\top \mid \top \mid \neg C^\bot \mid C_1^\top \sqcap C_2^\top \mid \exists R^\top.C^\top \mid \geqslant \bar{n}\, R^\top.C^\top \mid \geqslant 0\, R.C$
$\mathsf{Ax}(\mathbf{S})$ ::=	$C^\bot \sqsubseteq C \mid C \sqsubseteq C^\top \mid R \sqsubseteq R^\top \mid \mathsf{Trans}(R^\top)$

Figure 2. Syntactic locality conditions

Axiom (2) is $\top$-local w.r.t. $\mathbf{S} = \{\mathsf{Juvenile\_Chronic\_Polyarthritis}\}$, and Axiom (1) is $\bot$-local w.r.t. $\mathbf{S} = \{\mathsf{JRA}\}$. Note that the locality conditions allow us to refer to a reused concept for expressing a property of some other concept; for example, the axiom $\mathsf{Polyarticular\_JRA} \sqsubseteq\, \geqslant 5\, \mathsf{affects}.\underline{\mathsf{Joint}}$ is $\bot$-local w.r.t. $\mathbf{S} = \{\mathsf{Joint}\}$.

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

Proposition 8 ([3], Locality Implies Safety). *If an ontology* $\mathcal{O}$ *is* $\bot$*-local or* $\top$*-local w.r.t.* $\mathbf{S}$*, then* $\mathcal{O}$ *is safe for* $\mathbf{S}$*.*

Propositions 6 and 8 suggest the following definition of modules in terms of locality.

Definition 9. *Let* $\mathcal{O}_1 \subseteq \mathcal{O}$ *be ontologies, and* $\mathbf{S}$ *a signature. We say that* $\mathcal{O}_1$ *is a* $\bot$-module ($\top$-module) for $\mathbf{S}$ in $\mathcal{O}$ *if* $\mathcal{O} \setminus \mathcal{O}_1$ *is* $\bot$*-local ($\top$-local) w.r.t.* $\mathbf{S} \cup \mathsf{Sig}(\mathcal{O}_1)$*.*

We illustrate these notions by an example. Figure 3 (a) shows an example ontology (TBox). The set of external symbols is $\mathbf{S}_0 = \{A, t_1, t_2\}$. In order to extract the $\bot$-module, we extend $\mathbf{S}_0$ stepwise as in Figure 3 (b). The $\top$-module is obtained analogously and consists of $\mathbf{S}'_5 = \{A, A_1, C_1, D_1, F_1, r_1, t_1, t_2\}$ and all axioms $\alpha \in \mathcal{O}$ with $\mathsf{Sig}(\alpha) \subseteq \mathbf{S}'_5$.
It is clear that $\bot$-modules and $\top$-modules are modules as in Definition 4:

Proposition 10 ([3], Locality-based Modules are Modules). *Let* $\mathcal{O}_1$ *be either a* $\bot$*-module or a* $\top$*-module for* $\mathbf{S}$ *in* $\mathcal{O}$ *and let* $\mathbf{S}' = \mathbf{S} \cup \mathsf{Sig}(\mathcal{O}_1)$*. Then* $\mathcal{O}_1$ *is an* $\mathbf{S}'$*-module in* $\mathcal{O}$*.*

These modules enjoy an important property which determines their scope: suppose that $\mathcal{O}_1$ ($\mathcal{O}_2$) is a $\bot$-module ($\top$-module) for $\mathbf{S}$ in $\mathcal{O}$, then $\mathcal{O}_1$ ($\mathcal{O}_2$) will contain all super-concepts (sub-concepts) in $\mathcal{O}$ of all concepts in $\mathbf{S}$:

$$\mathcal{O} = \{ \begin{array}{llll} A \sqsubseteq A_2, & A_2 \sqsubseteq \forall s_2.E_2, & A_1 \sqsubseteq A, & \forall s_1.E_1 \sqsubseteq A_1, \\ A_2 \sqsubseteq \exists r_2.C_2, & A_2 \sqsubseteq \forall t_2.F_2, & \exists r_1.C_1 \sqsubseteq A_1, & \forall t_1.F_1 \sqsubseteq A_1 \\ A_2 \sqsubseteq \forall r_2.D_2, & & \forall r_1.D_1 \sqsubseteq A_1 & \end{array} \}$$

(a) The TBox

	Consideration	Consequence
(1)	$A \in \mathbf{S}_0,\ A_2 \notin \mathbf{S}_0 \ \Rightarrow\ A \sqsubseteq A_2 \notin \mathsf{Ax}(\mathbf{S}_0)$	$\mathbf{S}_1 = \mathbf{S}_0 \cup \{A_2\}$
(2)	$A_2 \in \mathbf{S}_1,\ C_2 \notin \mathbf{S}_1 \ \Rightarrow\ \exists r_2.C_2 \in \mathbf{Con}(\bar{\mathbf{S}}_1)$ $\Rightarrow\ A_2 \sqsubseteq \exists r_2.C_2 \notin \mathsf{Ax}(\mathbf{S}_1)$	$\mathbf{S}_2 = \mathbf{S}_1 \cup \{r_2, C_2\}$
(3)	$A_2 \in \mathbf{S}_2,\ r_2 \in \mathbf{S}_2,\ D_2 \notin \mathbf{S}_2 \ \Rightarrow\ \exists r_2.\neg D_2 \notin \mathbf{Con}(\bar{\mathbf{S}}_2)$ $\Rightarrow\ \neg\exists r_2.\neg D_2 \notin \mathbf{Con}(\mathbf{S}_2) \ \Rightarrow\ A_2 \sqsubseteq \forall r_2.D_2 \notin \mathsf{Ax}(\mathbf{S}_2)$	$\mathbf{S}_3 = \mathbf{S}_2 \cup \{D_2\}$
(4)	$A_2 \in \mathbf{S}_3,\ s_2 \notin \mathbf{S}_3,\ E_2 \notin \mathbf{S}_3 \ \Rightarrow\ \exists s_2.\neg E_2 \in \mathbf{Con}(\bar{\mathbf{S}}_3)$ $\Rightarrow\ \neg\exists s_2.\neg E_2 \in \mathbf{Con}(\mathbf{S}_3) \ \Rightarrow\ A_2 \sqsubseteq \forall s_2.E_2 \in \mathsf{Ax}(\mathbf{S}_3)$	$\mathbf{S}_4 = \mathbf{S}_3$
(5)	analogous to (3)	$\mathbf{S}_5 = \mathbf{S}_4 \cup \{F_2\}$
	$\mathbf{S}_5 = \{A, A_2, C_2, D_2, F_2, r_2, t_1, t_2\}$ The $\bot$-module consists of $\mathbf{S}_5$ and all axioms $\alpha \in \mathcal{O}$ with $\mathsf{Sig}(\alpha) \subseteq \mathbf{S}_5$.	

(b) Extracting the $\bot$-module.

Figure 3. An example illustrating $\bot$- and $\top$-modules.

Proposition 11 ([3], Module Scope). *Let $\mathcal{O}$ be an ontology, X, Y be concept names in $\mathcal{O} \cup \{\top\} \cup \{\bot\}$, $\alpha := (X \sqsubseteq Y)$, $\beta := (Y \sqsubseteq X)$, and $\mathcal{O}_X \subseteq \mathcal{O}$ with $X \in \mathbf{S}$. Then the following statements hold.*

(i) *If $\mathcal{O}_X$ is a $\bot$-module in $\mathcal{O}$ for $\mathbf{S}$, then $\mathcal{O}_X \models \alpha$ iff $\mathcal{O} \models \alpha$.*
(ii) *If $\mathcal{O}_X$ is a $\top$-module in $\mathcal{O}$ for $\mathbf{S}$, then $\mathcal{O}_X \models \beta$ iff $\mathcal{O} \models \beta$.*

For example, if we were to reuse the concept JRA from NCI as shown in Figure 1 by extracting a $\bot$-module for a signature that contains JRA, such a module would contain all the super-concepts of JRA in NCI, namely Rheumatoid_Arthritis, Autoimmune_Disease, Rheumatologic_Disorder, Arthritis, and Arthropathy. Since such a fragment is a module, it will contain the axioms necessary for entailing those subsumption relations between the listed concepts that hold in NCI.

Finally, given $\mathcal{O}$ and $\mathbf{S}$, there is a unique minimal $\bot$-module and a unique minimal $\top$-module for $\mathbf{S}$ in $\mathcal{O}$ [3]. We denote these modules by $\mathsf{UpMod}(\mathcal{O}, \mathbf{S})$ and $\mathsf{LoMod}(\mathcal{O}, \mathbf{S})$. This is motivated by the alternative terms "upper/lower module" that refer to the property from Proposition 11. Following a similar approach as exemplified in Figure 3 (b), the modules can be computed efficiently [3].

3 A Novel Methodology for Ontology Reuse

Based on our scenario in Section 1 and the theory of modularity summarised in Section 2, we propose a novel methodology for designing an ontology when knowledge is to be borrowed from several external ontologies. This methodology provides precise guidelines for ontology developers to follow, and ensures that a set of logical guarantees will hold at certain stages of the design process.

3.1 The Methodology

We propose the working cycle given in Figure 4. 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 transferred into the current ontology. Note that the separation between offline and online 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.

The Offline Phase starts with the ontology $\mathcal{O}$ being developed, e.g., JRAO. The ontology engineer specifies the set $\mathbf{S}$ of symbols to be reused from external ontologies, and associates to each symbol the external ontology from which it will be borrowed. In Figure 4 this signature selection is represented in the *Repeat* loop: each $\mathbf{S}_i \subseteq \mathbf{S}$ represents the external symbols to be borrowed from a particular ontology $\mathcal{O}'_i$; in our example, we have $\mathbf{S} = \mathbf{S}_1 \uplus \mathbf{S}_2$, where $\mathbf{S}_1$ is associated with NCI and contains JRA, and $\mathbf{S}_2$ is associated with GALEN and contains symbols related to joints and drugs. This part of the offline phase may involve an "online" component since the developer may browse through the external ontologies to choose the symbols she wants to import.

Next, the ontology developer decides, for each $\mathbf{S}_i$, whether she wants to refine or generalise the symbols from this set. For instance, in the reuse example shown in Figure 1, the concept JRA from NCI is refined by the sub-concepts abbreviated $C_1, \ldots, C_7$. In both cases, the user may also reference the external symbols via roles; in our example, certain types of JRA are defined by referencing concepts in GALEN (e.g., joints) via the roles $\mathsf{affects}$ and $\mathsf{isTreatedBy}$. As argued in Section 1, refinement and generalisation, combined with reference, constitute the main possible intentions when reusing external knowledge. Therefore it is reasonable for the user, both from the modelling and tool design perspectives, to declare her intention. This step is represented by the *For* loop in Figure 4.

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:

Definition 12 (Safety Guarantee). *The ontology $\mathcal{O}$* guarantees safety w.r.t. the signatures $\mathbf{S}_1, \ldots, \mathbf{S}_n$ *if $\mathcal{O}$ is safe for $\mathbf{S}_i$ for all $1 \leq i \leq n$.*

In the next subsection, we will show how to guarantee safety.

In the Online Phase, the ontology engineer imports the relevant knowledge from each of the external ontologies. As argued in Section 1 and 2, we aim at extracting only those fragments from the external ontologies that are relevant to the reused symbols, and therefore the extracted fragments should be modules in the sense of Definition 4.

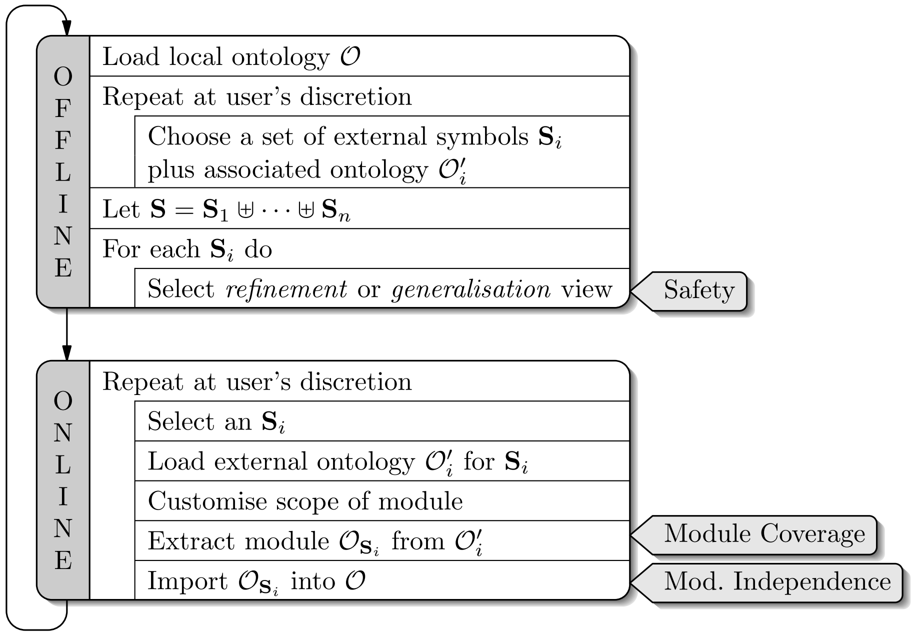

Figure 4. The two phases of import with the required guarantees

As shown in Figure 4, the import for each external ontology $\mathcal{O}'_i$ is performed in four steps. First, $\mathcal{O}'_i$ is loaded; by doing so, the ontology engineer commits to a particular version of it. Second, the scope of the module to be extracted from $\mathcal{O}'_i$ is customised; in practice, this means that the ontology engineer is given a view of $\mathcal{O}'_i$ and enabled to extend $\mathbf{S}_i$ by specifying requirements such as: "The module has to contain the concept Joint, all its direct super-concepts and two levels of its sub-concepts". In the third step, the actual fragment of $\mathcal{O}'_i$ is extracted. At this stage, we should ensure that the extracted fragment is a module for the customised signature according to Definition 4. Therefore, the following guarantee should be provided for each external ontology and customised signature:

Definition 13 (Module Coverage Guarantee). *Let* $\mathbf{S}$ *be a signature and* $\mathcal{O}'_{\mathbf{S}} \subseteq \mathcal{O}'$ *ontologies.* $\mathcal{O}'_{\mathbf{S}}$ guarantees coverage of $\mathbf{S}$ in $\mathcal{O}'$ *if* $\mathcal{O}'_{\mathbf{S}}$ *is a module for* $\mathbf{S}$ *in* $\mathcal{O}'$*.*

Finally, the actual module $\mathcal{O}_{\mathbf{S}_i}$ is imported. 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. Such an effect is obviously undesirable. Hence the following guarantee should be provided:

Definition 14 (Module Independence Guarantee). *Let* $\mathcal{O}$ *be an ontology and* $\mathbf{S}_1, \mathbf{S}_2$ *be signatures.* $\mathcal{O}$ guarantees module independence *if, for all* $\mathcal{O}'_1$ *with* $\mathsf{Sig}(\mathcal{O}) \cap \mathsf{Sig}(\mathcal{O}'_1) \subseteq \mathbf{S}_1$ *and for all* $\mathcal{O}'_2$ *with* $\mathsf{Sig}(\mathcal{O}) \cap \mathsf{Sig}(\mathcal{O}'_2) \subseteq \mathbf{S}_2$ *and* $\mathsf{Sig}(\mathcal{O}'_1) \cap \mathsf{Sig}(\mathcal{O}'_2) = \emptyset$*, it holds that* $\mathcal{O} \cup \mathcal{O}'_1 \cup \mathcal{O}'_2$ *is a conservative extension of* $\mathcal{O} \cup \mathcal{O}'_1$*.*

Note that, if we dropped the requirement $\mathsf{Sig}(\mathcal{O}'_1) \cap \mathsf{Sig}(\mathcal{O}'_2) = \emptyset$, then module independence would hold if, for all $\mathcal{O}'_1$ as above, $\mathcal{O} \cup \mathcal{O}'_1$ were safe for $\mathbf{S}_2$. However, this would be a meaningless definition because no $\mathcal{O}$ would guarantee module independence for any $\mathbf{S}_2$ with more than one concept name: if we chose $\mathbf{S}_2 = \{A, B\}$, for concept names A, B, and $\mathcal{O}'_1$ such that it entails $A \sqsubseteq B$, then $\mathcal{O} \cup T'_1$ would not be safe for S_2, for any $\mathcal{O}$.

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.

3.2 Achieving the Logical Guarantees

In order to provide the necessary guarantees of our methodology, we will now make use of the locality conditions introduced in Section 2.3 and some general properties of conservative extensions, safety and modules.

The Safety Guarantee. In Section 2.3, we argue that the simultaneous refinement and generalisation of an external concept may violate safety. To preserve safety, we propose to use two locality conditions: $\bot$-locality, suitable for refinement, and $\top$-locality, suitable for generalisation. These conditions can be checked syntactically using the grammars defined in Figure 2, and therefore they can be easily implemented. In order to achieve the safety guarantee at the end of the offline phase, we propose to follow the procedure sketched in Figure 5.

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 $\mathcal{O}_1$ that guarantees safety

$\mathcal{O}_1 := \mathcal{O}$
while exists $\mathbf{S}_i$ such that $\mathcal{O}$ **not** local according to the selection for $\mathbf{S}_i$ **do**
    check $\begin{cases} \bot\text{-locality of } \mathcal{O}_1 \text{ w.r.t. } \mathbf{S}_i \text{ if } \mathbf{S}_i \text{ is to be refined} \\ \top\text{-locality of } \mathcal{O}_1 \text{ w.r.t. } \mathbf{S}_i \text{ if } \mathbf{S}_i \text{ is to be generalised} \end{cases}$
    if non-local **then**
        $\mathcal{O}_1 :=$ repair $\mathcal{O}_1$ until it is local for $\mathbf{S}_i$ according to the choice for $\mathbf{S}_i$
    end if
end while
return $\mathcal{O}_1$

Figure 5. A procedure for checking safety

It is immediate to see that the following holds upon completion of the procedure: for each $\mathbf{S}_i$, if the user selected the refinement (generalisation) view, then $\mathcal{O}$ is $\bot$-local ($\top$-local) w.r.t. $\mathbf{S}_i$. This is sufficient to guarantee safety, due to Proposition 8:

Proposition 15. *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. $\mathbf{S}_1, \ldots, \mathbf{S}_n$.*

The Module Coverage Guarantee. The fragment extracted for each customised signature in the online phase must satisfy the module coverage guarantee. As seen in Section 2.3, $\bot$-locality and $\top$-locality can also be used for extracting modules in the sense of Definition 4. Given an external ontology $\mathcal{O}'$ and customised signature $\mathbf{S}_i$, the scope of the $\bot$-module and $\top$-module is determined by Proposition 11: as shown in Figure 3, the $\bot$-module will contain all the super-concepts in $\mathcal{O}'$ of the concepts in $\mathbf{S}_i$, whereas the $\top$-module will contain all the sub-concepts.

The construction in Figure 3 also shows that 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 proceed as follows, given $\mathcal{O}'_i$ and $\mathbf{S}_i$: first, extract the minimal $\bot$-module M for $\mathbf{S}$ in $\mathcal{O}'_i$; then, extract the minimal $\top$-module for $\mathbf{S}$ in M. The fragment obtained at the end of this process satisfies the module coverage guarantee as given in the following proposition:

Proposition 16. *Let* $\mathcal{O}'_1 = \mathsf{UpMod}(\mathcal{O}', \mathbf{S})$*, and* $\mathcal{O}'_2 = \mathsf{LoMod}(\mathcal{O}'_1, \mathbf{S})$*. Then* $\mathcal{O}'_2$ *guarantees coverage of* $\mathbf{S}$ *in* $\mathcal{O}'$*.*

Proof. Let $\mathbf{S}_1 = \mathsf{Sig}(\mathcal{O}'_1) \cup \mathbf{S}$ and $\mathbf{S}_2 = \mathsf{Sig}(\mathcal{O}'_2) \cup \mathbf{S}$. By Proposition 10, $\mathcal{O}'_1$ is a $\mathbf{S}_1$-module in $\mathcal{O}'$. By Proposition 1, $\mathcal{O}'_1$ is a $\mathbf{S}$-module in $\mathcal{O}'$. By Proposition 10, $\mathcal{O}'_2$ is a $\mathbf{S}_2$-module in $\mathcal{O}'_1$. By Proposition 1, $\mathcal{O}'_2$ is a $\mathbf{S}$-module in $\mathcal{O}'_1$. By Proposition 2, $\mathcal{O}'_2$ is a $\mathbf{S}$-module in $\mathcal{O}'$. By Definition 13, $\mathcal{O}'_2$ guarantees coverage of $\mathbf{S}$ in $\mathcal{O}'$.

The Module Independence Guarantee. When a module is imported in the online phase (see Figure 4), module independence should be guaranteed—that is, after importing a module for a signature $\mathbf{S}_i$ from an external ontology $\mathcal{O}'_i$ into $\mathcal{O}$, the extended ontology $\mathcal{O}$ should still satisfy safety. It would be convenient if $\bot$-locality or $\top$-locality of $\mathcal{O}$ with respect to each of the signatures $\mathbf{S}_1, \mathbf{S}_2$ guaranteed module independence.[11] Unfortunately, this is not the case, as the following example shows.

Let $\mathcal{O} = \{A \sqsubseteq B\}$, $\mathcal{O}'_1 = \{A \equiv \top\}$, $\mathcal{O}'_2 = \{B \sqsubseteq \bot\}$ with $\mathbf{S}_1 = \{A\}$ and $\mathbf{S}_2 = \{B\}$. Then $\mathcal{O}$ is $\top$-local w.r.t. $\mathbf{S}_1$ and $\bot$-local w.r.t. $\mathbf{S}_2$, but $\mathcal{O} \cup \mathcal{O}'_1$ inconsistent and therefore not safe w.r.t. $\mathbf{S}_2$.

This example shows that we cannot deduce independence only from safety of $\mathcal{O}$ with respect to both signatures, the reasons being that safety is guaranteed through different locality types and that the external ontologies entail (here, even explicitly state) that concept names from the corresponding signatures are equal to $\bot$ or $\top$. The latter problem might be circumvented by checking for such entailments, but this is not feasible because it involves reasoning, which is computationally complex for $\mathcal{SHIQ}/\mathcal{SHOIQ}$ and would destroy the efficiency of our methodology. However, we can at least guarantee safety for $\mathbf{S}_2$ after an

[11] In the conference and workshop versions [7, 8, 9] of this report, we have erroneously claimed that $\bot$-locality or $\top$-locality of $\mathcal{O}$ with respect to each of the signatures $\mathbf{S}_1, \mathbf{S}_2$ suffices to guarantee module independence. We apologise for this mistake and correct it here.

import of $\mathcal{O}'_1$ if we approximate the check for the critical entailments using locality.

Proposition 17. *Let $\mathcal{O}$ be an ontology and $\mathbf{S}_1, \mathbf{S}_2$ disjoint signatures. If the following conditions are satisfied, where x denotes either $\bot$ or $\top$, then $\mathcal{O} \cup \mathcal{O}'_1$ is x-local, and therefore safe, with respect to $\mathbf{S}_2$.*

1. *$\mathcal{O}$ is x-local with respect to $\mathbf{S}_2$.*
2. *$\mathsf{Sig}(\mathcal{O}'_1) \cap \mathbf{S}_2 = \emptyset$.*
3. *$\mathcal{O}'_1$ is x-local with respect to the empty signature.*

Proof. Conditions 2 and 3 imply that $\mathcal{O}'_1$ is x-local with respect to $\mathbf{S}_2$. This, together with 1, yields that $\mathcal{O} \cup \mathcal{O}'_1$ is x-local with respect to $\mathbf{S}_2$. The last step is due to the fact that both locality types are closed under the union of ontologies, which immediately follows from their definition.

Two remarks seem appropriate. First, we could as well replace Conditions 2 and 3 with requiring x-locality of $\mathcal{O}'_1$ w.r.t. $\mathbf{S}_2$. However, this is generally a stronger restriction than 3 alone, and would need to be re-checked everytime $\mathbf{S}_2$ is changed. Since we can safely assume the signatures of the external ontologies to be disjoint (see remark at the end of Section 3.1), we automatically have 2, which implies that 3 and x-locality of $\mathcal{O}'_1$ w.r.t. $\mathbf{S}_2$ are equivalent. Second, safety of $\mathcal{O}$ w.r.t. $\mathbf{S}_1$ is not needed to guarantee safety after the first import. Consequently, the type x of locality used in 3 is only determined by the type of locality used to guarantee safety of $\mathcal{O}$ w.r.t. $\mathbf{S}_2$ in 1.

It remains to describe how to fit the provision of *safety after import* into our methodology. Since conventional safety is checked and—if necessary—repaired during the offline phase, *safety after import* can only be lost if Condition 3 from Proposition 17 is violated. Checking this condition involves looking at each external ontology; hence it is only reasonable to perform this check in the online phase. More precisely, each external ontology T'_i usually needs to be checked for $\bot$-locality *and* $\top$-locality w.r.t. the empty signature. "Usually" means that we assume that safety w.r.t the other external ontologies is achieved via *different* locality types ($\bot$ and $\top$). In the unlikely event of achieving safety for all other external ontologies via *the same* locality type x, the external ontology T'_i needs to be checked for x-locality only.

If this test fails for any external ontology T'_i, we cannot expect the user to "repair" *safety after import* by modifying $\mathcal{O}'_i$. This would contradict one of the basic assumptions underlying our import scenario: one reason for importing $\mathcal{O}'_i$ is that the developer of $\mathcal{O}$ does not need to be an expert in all areas covered by $\mathcal{O}'_i$. Hence, she cannot be expected to devise the necessary modifications and to be aware of their consequences. Rather, we propose to still import $\mathcal{O}'_i$ with the risk of sacrificing safety of $\mathcal{O}$ w.r.t. any of the remaining signatures. The latter then needs to be re-checked signature by signature and can be repaired as described earlier. After this has been done, the online phase continues until the next violation of the above Condition 3 is encountered.

4 The Ontology Reuse Tool

We have developed a Protégé 4[12] plugin that supports the methodology presented in Section 3. The plugin and user manual can be downloaded from `http://krono.act.uji.es/people/Ernesto/safety-ontology-reuse`.

The Offline Phase. The first step of the offline phase involves the selection of the external entities. Our plugin provides functionality for declaring entities as external as well as for defining the external ontology URI (or signature subgroup) for the selected entities; this information is stored in the ontology using OWL 1.1 annotations [10] as follows: we use an ontology annotation axiom per external ontology, an entity annotation axiom to declare an entity external, and an entity annotation axiom per external entity to indicate its external ontology. The set of external entities with the same external ontology URI can be viewed as one of the $\mathbf{S}_i$. Finally, the UI of the plugin also allows for the specification, for each external ontology, whether it will be refined or generalised. Once the external entities have been declared and divided into groups, the tool allows for safety checking of the ontology under development w.r.t. each group of external symbols separately. The safety check uses $\bot$-locality ($\top$-locality) for signature groups that adopt the refinement (generalisation) view. The non-local axioms to be repaired are appropriately displayed.

Figure 6 shows the ProSÉ tab with the set of signature subgroups in the top left corner, and the non-local axioms in the bottom left corner. Note that, in this phase, our tool does allow the user to work completely offline, without the need of extracting and importing external knowledge, and even without knowing exactly from which ontology the reused entities will come from. Indeed, the specification of the URI of the external ontologies is optional at this stage, and, even if indicated, such URI may not refer to a real ontology, but it may simply act as a temporary name.

The Online Phase. In the online phase, the user chooses external ontologies and imports axioms from them. At this stage, the groups of external symbols to be imported should refer to the location of a "real" external ontology. Once an external signature group $\mathbf{S}_i$ has been selected for import, the selected signature can be customised by adding super-concepts and sub-concepts of the selected symbols. The tool 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. The module is computed using the procedure in Proposition 16; 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. The user is also given the option to import the whole external ontology instead of importing a module. Note that currently the import of a module is done "by value", in the sense that the module becomes independent from the original ontology: if the external ontology on the Web evolves, the previously extracted module will not change.

[12] Ontology Editor Protégé 4: `http://www.co-ode.org/downloads/protege-x/`

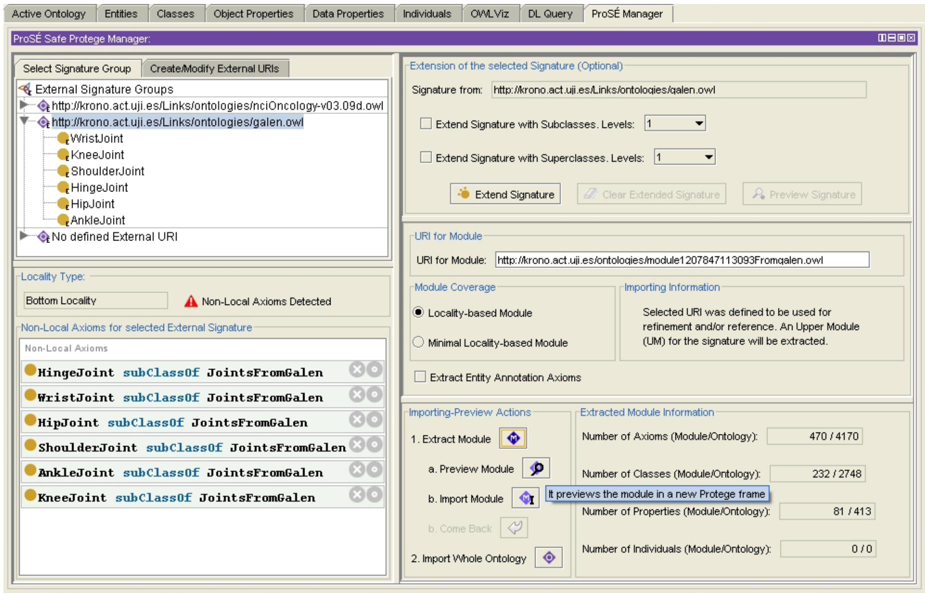

Figure 6. ProSÉ—a Protégé-4 *P*lugin for *R*eusing *O*ntologies: *S*afe and *É*conomique

The right hand side of the ProSÉ tab (see Figure 6) includes the set of necessary components to support the proposed steps for the online phase.

5 Evaluation

So far, we have demonstrated our tool to various ontology developers[13] who have expressed great interest, and we are currently working on a proper user study. In the following, we describe various experiments we have performed to prove that locality-based modules are reasonably sized compared to the whole ontology.

5.1 Systematic experiments using randomly generated signatures

For each concept name A in NCI, Galen[14], or SNOMED[15], we have proceeded as follows.

– For each pair (u, ℓ) between $(0, 0)$ and $(3, 3)$, we have constructed the signature $\mathbf{S}(A, u, \ell)$ by taking A, its super-concepts in the next u levels and its sub-concepts in the next ℓ levels.

[13] Thanks to Elena Beißwanger, Sebastian Brandt, Alan Rector, and Holger Stenzhorn for valuable comments and feedback.

[14] We have used a fragment of GALEN expressible in OWL: `http://krono.act.uji.es/Links/ontologies/galen.owl/view`

[15] SNOMED describes health care terminology. [16]

– We have extracted $\mathcal{O}'_1 = \mathsf{UpMod}(\mathcal{O}', \mathbf{S}_A)$ and $\mathcal{O}'_2 = \mathsf{LoMod}(\mathcal{O}'_1, \mathbf{S}_A)$, see Prop. 16.

In the remainder of this section, we refer to the modules $\mathcal{O}'_1$ and $\mathcal{O}'_2$ mentioned above as **UM** (Upper Module) and **LUM** (Lower of Upper Module). These names are motivated by the fact that the $\perp$-module (the $\top$-module) for a signature $\mathbf{S}$ contains the super-concepts (sub-concepts) of all concepts in $\mathbf{S}$.

We have grouped the extracted modules according to the size of the input signature in order to evaluate its impact on the size of the modules. Figure 7 shows the obtained results for a small version of Galen[17] that consists of 4170 axioms. The size of a module is the number of its axioms. The following conclusions can be drawn from the empirical results, where the initial signatures $\mathbf{S}(A, u, \ell)$ contained between 1 and 330 entities.

1. The modules obtained are small on average—99% of UMs have at most 487 axioms ($\sim$ 12% of the size of Galen) and 99% of the LUMs contain at most and 386 axioms ($\sim$ 9% of the size of Galen).
2. The growth in the size of the modules with respect to the size of the initial signature is smooth and linear up to initial signatures containing 100 entities.

We have similar findings for NCI (395,124 axioms) and SNOMED (389,541 ax.).

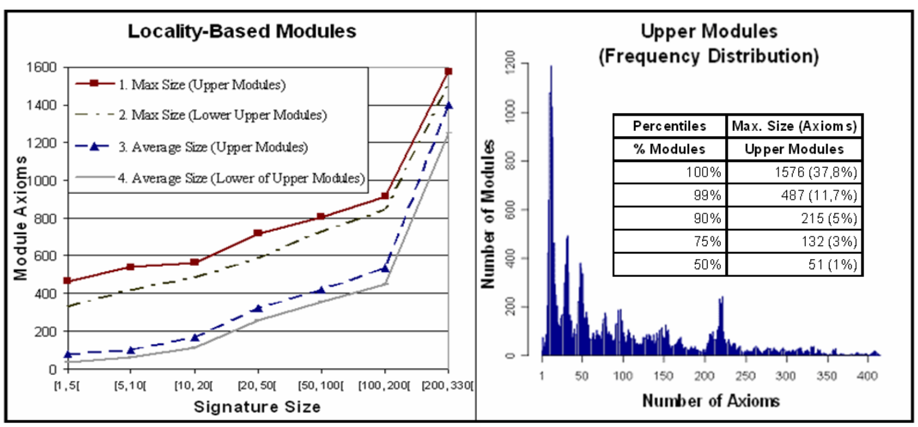

Figure 7. Module information obtained from Galen. (left) Module average size against max size. (right) Frequency distributions for module sizes

5.2 Individual experiments based on "real-life" signatures

In addition to the "synthetic" experiments, we have undertaken some real experiments in the context of the Health-e-Child project's user scenario, see Section 1.

[17] `http://krono.act.uji.es/Links/ontologies/galen.owl/view`

The experiments focus on JRA and Cardiomyopathies (CMP)—a group of diseases that are central to the project. Using our tool, members of the project have manually selected signatures that are relevant to JRA and Cardiomyopathies from both Galen and NCI, expanded these signatures as in the case of the synthetic tests, and extracted the corresponding modules. For example, in the case of JRA in Galen, the initial signature and expanded signature consisted of 40 and 131 entities. The following tables show the sizes of all signatures and modules extracted from (a) Galen and (b) NCI.

(a)

Disease	JRA	JRA	CMP
Signature size	40	131	77
# axioms	490	1151	614
# concepts	296	663	334
# roles	69	116	56

(b)

Disease	JRA	JRA	CMP
Signature size	48	356	124
# axioms	300	1258	537
# concepts	193	613	283
# roles	17	21	10

5.3 Comparisons with other segmentation algorithms.

We are currently carrying out experiments where we compare locality-based modules with fragments obtained via other segmentation algorithms, for instance [11, 12, 13]. The aim of this ongoing work is to compare modules and segments for the same signature with respect to their size, content, and computation time. Most of these other segmentation algorithms do not provide logical guarantees, and it is worthwhile to find out how heavily these guarantees are broken.

Based on real-life signatures consisting of 40 and 48 entities taken from Galen and NCI similarly as in the previous subsection, we have computed UMs, LUMs and segments according to the approach in [12]. We call the latter segments SR segments, referring to the authors of that paper. By now, we have obtained the following findings, see also `http://krono.act.uji.es/people/Ernesto/safety-ontology-reuse/module-experiments` for more details.

- The size of the Galen-LUMs is 40–70% of the size of the corresponding SR segments. In each case, the overlap is large, but both kinds of fragments contain entities and axioms that the other does not contain.
- The Galen-UMs are slightly larger than the SR segments. Except for very few entities and axioms, the latter are subsets of the UMs.
- The NCI-LUMs are very small subsets (2–45%) of the SR segments.
- The NCI-UMs and the SR segments coincide. This might be caused by the pure taxonomic character of NCI.

Furthermore, we are currently extracting larger modules from SNOMED, again in order to compare them with other segments based on the same initial signature. From three signatures containing some 4,000, 16,000, and 24,000 class names that have proven to be relevant in practice over the last few years, we have extracted modules comprising approximately 4%, 10%, and 15% of the whole ontology.

6 Related Work

Ontology Engineering Methodologies. Several ontology engineering methodologies can be found in the literature; prominent examples are Methontology [14], On-To-Knowledge (OTK) [15], and ONTOCLEAN [16]. These methodologies, however, do not address ontology development scenarios involving reuse. Our proposed methodology is complementary and can be used in combination with them.

Ontology Segmentation and Ontology Integration Techniques. In the last few years, a growing body of work has been developed addressing Ontology Modularisation, Ontology Mapping and Alignment, Ontology Merging, Ontology Integration and Ontology Segmentation, see [17, 18, 19] for surveys. This field is diverse and has originated from different communities.

In particular, numerous techniques for extracting fragments of ontologies are known. Most of them, such as [11, 12, 13], rely on syntactic heuristics for detecting relevant axioms. These techniques do not attempt to formally specify the intended outputs and do not provide any guarantees.

Ontology Reuse techniques. There are various proposals for "safely" combining modules; most of these proposals, such as $\mathcal{E}$-connections, Distributed Description Logics and Package-based Description Logics propose a specialised semantics for controlling the interaction between the importing and the imported modules to avoid side-effects, for an overview see [20]. In contrast, in our paper we assume that reuse is performed by simply building the logical union of the axioms in the modules under the standard semantics. We provide the user with a collection of reasoning services, such as safety testing, to check for side-effects. Our paper is based on other work on modular reuse of ontologies [21, 22, 6, 5] which enables us to provide the necessary guarantees. We extend this work with a methodology and tool support.

7 Lessons Learned and Future Work

We have described a logic-based approach to the reuse of ontologies that is both *safe* (i.e., we guarantee that the meaning of the imported symbols is not changed) and *economic* (i.e., we import only the module relevant for a given set of symbol and we guarantee that we do not lose any entailments compared to the import of the whole ontology). We have described a methodology that makes use of this approach, have implemented tool support for it in Protégé, and report on experiments that indicate that our modules are indeed of acceptable size.

In the future, we will extend the tool support so that the user can "shop" for symbols to reuse: it will allow to browse an ontology for symbols to reuse and provide a simple mechanism to pick them and, on "check-out", will compute the relevant module. Moreover, we are working on more efficient ways of module extraction that make use of already existing computations. Next, we plan to carry out a user study to learn more about the usefulness of the interface and how to further improve it. Finally, our current tool support implements a "by value" mechanism: modules are extracted and added at the user's request. In addition, we would like to support import "by reference": a feature that checks whether the imported ontology has changed and thus a new import is necessary.

Acknowledgements

This work was partially supported by the PhD Fellowship Program of the *Generalitat Valenciana*, by the *Fundació Caixa Castelló-Bancaixa*, and by the UK EPSRC grant no. EP/E065155/1.

References

[1] Golbeck, J., Fragoso, G., Hartel, F.W., Hendler, J.A., Oberthaler, J., Parsia, B.: The National Cancer Institute's Thésaurus and Ontology. J. of Web Semantics **1**(1) (2003) 75–80

[2] Horrocks, I., Patel-Schneider, P.F., van Harmelen, F.: From $\mathcal{SHIQ}$ and RDF to OWL: The making of a web ontology language. J. of Web Semantics **1**(1) (2003) 7–26

[3] Cuenca Grau, B., Horrocks, I., Kazakov, Y., Sattler, U.: Modular reuse of ontologies: Theory and practice. J. of Artificial Intelligence Research **31** (2008) 273–318

[4] Ghilardi, S., Lutz, C., Wolter, F.: Did I damage my ontology? A case for conservative extensions in description logics. In Doherty, P., Mylopoulos, J., Welty, C., eds.: Proc. of KR-06, AAAI Press (2006) 187–197

[5] Lutz, C., Walther, D., Wolter, F.: Conservative extensions in expressive description logics. In: Proc. of IJCAI-07, AAAI (2007) 453–459

[6] Cuenca Grau, B., Horrocks, I., Kazakov, Y., Sattler, U.: A logical framework for modularity of ontologies. In: Proc. of IJCAI-07, AAAI (2007) 298–304

[7] Jiménez-Ruiz, E., Cuenca Grau, B., Sattler, U., Schneider, T., Berlanga, R.: Safe and economic re-use of ontologies: A logic-based methodology and tool support. In Bechhofer, S., Hauswirth, M., Hoffmann, J., Koubarakis, M., eds.: ESWC. Volume 5021 of LNCS., Springer (2008) 185–199

[8] Jiménez-Ruiz, E., Cuenca Grau, B., Sattler, U., Schneider, T., Berlanga, R.: Safe and economic re-use of ontologies: A logic-based methodology and tool support. Volume 353., CEUR (`http://ceur-ws.org/`) (2008)

[9] Jiménez-Ruiz, E., Cuenca Grau, B., Sattler, U., Schneider, T., Berlanga, R.: Safe and economic re-use of ontologies: A logic-based methodology and tool support. In: Proc. of OWLED. (2008)

[10] Motik, B., Patel-Schneider, P.F., Horrocks, I.: OWL 1.1 Web Ontology Language Structural Specification and Functional-Style Syntax. W3C Member Submission (2007)

[11] Noy, N., Musen, M.: The PROMPT suite: Interactive tools for ontology mapping and merging. Int. J. of Human-Computer Studies **6**(59) (2003)

[12] Seidenberg, J., Rector, A.L.: Web ontology segmentation: analysis, classification and use. In: Proc. of WWW 2006, ACM (2006) 13–22

[13] Jiménez-Ruiz, E., Berlanga, R., Nebot, V., Sanz, I.: Ontopath: A language for retrieving ontology fragments. In: Proc. of ODBASE, LNCS. (2007) 897–914

[14] M. Fernandez, A. Gomez-Perez, e.a.: Methontology: From ontological art towards ontological engineering. In: AAAI, Stanford, USA. (1997)

[15] Sure, Y., Staab, S., Studer, R.: On-to-knowledge methodology. In: In Handbook on Ontologies. Edited by S. Staab and R. Studer (eds.). Springer. (2003)

[16] Guarino, N., Welty, C.: Evaluating ontological decisions with ontoclean. Commun. ACM **45**(2) (2002) 61–65

[17] Kalfoglou, Y., Schorlemmer, M.: Ontology mapping: The state of the art. The Knowledge Engineering Review **18** (2003) 1–31

[18] Noy, N.F.: Semantic integration: A survey of ontology-based approaches. SIGMOD Record **33**(4) (2004) 65–70

[19] Noy, N.F.: Tools for mapping and merging ontologies. In Staab, S., Studer, R., eds.: Handbook on Ontologies. International Handbooks on Information Systems. Springer (2004) 365–384

[20] Cuenca Grau, B., Kutz, O.: Modular ontology languages revisited. In: Proc. of the Workshop on Semantic Web for Collaborative Knowledge Acquisition. (2007)

[21] Cuenca Grau, B., Horrocks, I., Kazakov, Y., Sattler, U.: Just the right amount: extracting modules from ontologies. In Williamson, C.L., Zurko, M.E., Patel-Schneider, P.F., Shenoy, P.J., eds.: WWW, ACM (2007) 717–726

[22] Cuenca Grau, B., Horrocks, I., Kazakov, Y., Sattler, U.: Ontology reuse: Better safe than sorry. In: Proc. of DL 2007. Volume 250 of CEUR WS Proc. (2007)