Composing and decomposing ontologies: a logic-based approach — A Case for Support

PRINCIPAL INVESTIGATORS: Ulrike Sattler[1] and Frank Wolter[2].
CO-INVESTIGATORS: Ian Horrocks[1], Boris Konev[2], and Bijan Parsia[1].

1. School of Computer Science, University of Manchester, Manchester M13 9PL
{sattler,horrocks,parsia}@cs.man.ac.uk
2. Department of Computer Science, University of Liverpool, Liverpool L69 3BX
{wolter,konev}@csc.liv.ac.uk

PART 1: Previous Research

The project will be jointly undertaken by the University of Manchester and the University of Liverpool. The team members from both institutions are world leaders in their own fields and have, between them, the range of complementary expertise that will be necessary for the successful completion of the project. The project will build on a number of collaborations already established by members of the team. For example, **Sattler** and **Wolter** have been jointly working on the relation between first-order and description logics (DLs) and DLs for describing actions and services [21, 2]. **Horrocks**, **Sattler**, and **Wolter** are working together on the DynamO project (GR/S63168/01) investigating logics for the description of services and related reasoning problems, and they have jointly organised international events such as the International Description Logic Workshop 2005. **Konev** and **Wolter** are working on the algorithmic properties of expressive temporal logics and logics for dynamical systems [17, 19]. **Horrocks** and **Sattler** have designed expressive DLs and practicable reasoning algorithms, and have worked on their usage as logical underpinning for ontology languages [12, 13, 11]. Of particular interest for this project is their joint recent work on modularity of DLs [3, 7]. **Parsia**, **Horrocks** and **Sattler** are working together on further developments of OWL, namely OWL 1.1, and on related reasoning problems and tools [8].

Ulrike Sattler is a Reader in the Information Management Group. She has worked in the field of knowledge representation and reasoning since 1994 [14, 13]. Most of her work on practical inference algorithms was carried out in a productive co-operation with Ian Horrocks. An outcome of this line of research was the design of the $\mathcal{SHIQ}$ family of DLs: these are DLs that form the logical basis of ontology languages such as OIL, DAML+OIL, and OWL [1]. Besides this "applied" research, she is interested in the computational complexity of inference problems underlying the system services of knowledge representation systems. She is/has been involved in several national and international research projects, including DynamO, Extending Expressive Description Logics (GR/S87171/01), DWQ (Esprit LTR-22469), and TONES (IST-007603-2), and was a DAAD fellow. She was a member of the editorial board of JAIR, is a corner editor of JLC, a member of the DL Steering Committee, and a member of numerous programme committees, e.g., ISWC 2003-2006, ECAI 2006, WWW 2003, AAAI 2006, and AiML 2004.

Frank Wolter is a Professor of Logic and Computation. He has been conducting advanced research in computational logic and its applications to knowledge representation and reasoning for more than ten years. He is co-author of the research monograph "Many-dimensional modal logics: theory and applications", Elsevier 2003 and co-editor of the Handbook of Modal Logic, Elsevier, 2006. He was an invited speaker at CADE 2005, has served on many programme committees, including KR 2002-2006, AiML 2002-2006, IJCAR 2005, ECAI 2006, and is member of the steering committees of DL, FroCoS and AiML. Wolter has been a principal investigator of two DFG (Germany) projects on combining modal and description logics (WO 583/3-(1-3)) and is principal investigator of the EPSRC projects "Knowledge Representation and Reasoning about Distances" (GR/S63182/01) and DynamO. Of particular importance for this project are his recent publications on composing logics using E-connections [20] and on conservative extensions [5, 6, 23]. He is also an invited speaker at the first International Workshop on Modular Ontologies, Athens, USA, 2006.

Ian Horrocks is a Professor in the School of Computer Science. His FaCT system revolutionised the design of DL systems, redefining the notion of tractability for DLs and establishing a new standard for DL implementations. He won the *best paper award* at KR'98 [10], and has given invited talks about his work at research labs, universities and conferences around the world. He is/has been a member of the programme committees of numerous international conferences and journals including AAAI, CADE, IJCAI, IJCAR, ISWC, KR, WWW, JAIR, ETAI and the Journal on Web Semantics. He was the program chair of the 2002 International Semantic Web Conference, the 2002 Description Logic Workshop, and the Semantic Web track of the 2003 World Wide Web Conference. He is/has been involved in numerous national and international research projects including DynamO, REOL (EP/C537211/1), LOGO (EP/C543319/1), CO-ODE (JISC), HyOntUse (GR/S44686/01), myGrid (GR/R67743/01), Reasoning About Conjunctive Query Containment Under Constraints (GR/R00340/01), Camelot (GR/L54516/01), TONES (IST-007603-2), Knowledge Web (IST-2004-507482), OntoWeb (IST-2000-25056), and DWQ (Esprit LTR-22469), was a consultant to the DARPA DAML program, and was the coordinator of the EU IST WonderWeb project (IST-2001-33052). He is/was a member of the Joint EU/US Committee on Agent Markup Languages, the W3C Web Ontology Language working group and the W3C Rule Interchange Format working group, and was a prime mover in and editor of the OIL, DAML+OIL, and OWL ontology language standards. In 2005 he received the BCS Roger Needham award for his contribution to the "formal foundations for ontology languages and the semantic web", and was also awarded an EPSRC Senior Research Fellowship.

Boris Konev is a Lecturer in the Department of Computer Science. He has been working on complexity of proofs and decision procedures, the development of resolution procedures for temporal logics [4], the development of implementation techniques for non-classical resolution [18], and the application of the developed tools to verification problems. Konev served as a PC member for a number of international events including TIME'07, JELIA'06, IWIL (since 2003) and as a special issue editor of JANCL.

Bijan Parsia is a Lecturer in the Information Management Group. In his prior position at the University of Maryland, College Park, he led the development of a new generation of ontology engineering tools including the OWL ontology editor Swoop. He has developed practical debugging services for expressive ontology languages by extending traditional "axiom pinpointing" services to the $\mathcal{SHIQ}$ family of DLs, developing new services [16], integrating these services with Swoop [15], and formally evaluating their usability. He was ontology editor for the Journal of Web Semantics (2005-2006), and a member of many programme committees, e.g., ISWC 2004-2006, AAAI 2006, and IJCAI 2007. Of particular interest for this project is his recent work on modular ontologies [9].

Dirk Walther is the named Research Assistant to be funded by the project at Liverpool. He is currently a PhD student supervised by Professors Frank Wolter and Michael Wooldridge at Liverpool University and will submit his thesis in March 2007. He has published a number of high quality papers on the complexity of temporal logics [24, 22]. Of particular interest for this project is his joint work with **Wolter** on conservative extensions in expressive DLs [23].

The Team. It can thus be seen that the Investigators (and named RA) form a very strong and balanced team. They have done groundbreaking theoretical and practical work in knowledge representation, automated reasoning, computational logic, and ontologies. This highly productive association will be continued and developed during the proposed ambitious research.

References

[1] F. Baader, I. Horrocks, and U. Sattler. Description logics as ontology languages for the semantic web. In *Mechanizing Mathematical Reasoning*, LNAI. Springer Verlag, 2005.

[2] F. Baader, C. Lutz, M. Milicic, U. Sattler, and F. Wolter. Integrating description logics and action formalisms. In *Proc. of AAAI*. AAAI Press, 2005.

[3] B. Cuenca Grau, I. Horrocks, O. Kutz, and U. Sattler. Will my ontologies fit together? In *Proc. of DL*. CEUR, 2006.

[4] A. Degtyarev, M. Fisher, and B. Konev. Monodic temporal resolution. *ACM Transactions on Computational Logic*, 7(1), 2006.

[5] S. Ghilardi, C. Lutz, and F. Wolter. Did I damage my ontology? A case for conservative extensions in description logic. In *Proc. of KR*. AAAI Press, 2006.

[6] S. Ghilardi, C. Lutz, F. Wolter, and M. Zakharyaschev. Conservative extensions in modal logics. In *Proc. of AiML 6*. College Publications, 2006.

[7] B. Cuenca Grau, I. Horrocks, Y. Kazakov, and U. Sattler. A logic-based framework for ontology modularity. In *Proc. of IJCAI*. AAAI Press, 2007. To appear.

[8] B. Cuenca Grau, I. Horrocks, B. Parsia, P. Patel-Schneider, and U. Sattler. Next steps for OWL. In *Proc. of OWL: Experiences and Directions*, 2006.

[9] B. Cuenca Grau, B. Parsia, E. Sirin, and A. Kalyanpur. Modularity and web ontologies. In *Proc. of KR*. AAAI Press, 2006.

[10] I. Horrocks. Using an expressive description logic: FaCT or fistion? In *Proc. of KR*. Morgan Kaufmann, 1998.

[11] I. Horrocks, O. Kutz, and U. Sattler. The even more irresistible $\mathcal{SROIQ}$. In *Proc. of KR*. AAAI Press, 2006.

[12] I. Horrocks and U. Sattler. Decidability of $\mathcal{SHIQ}$ with complex role inclusion axioms. *Artificial Intelligence Journal*, 120, 2004.

[13] I. Horrocks and U. Sattler. A tableau decision procedure for $\mathcal{SHOIQ}$. *J. of Automated Reasoning*, 2006. To appear.

[14] I. Horrocks, U. Sattler, and S. Tobies. Practical reasoning for expressive description logics. In *Proc. of LPAR*, LNAI. Springer Verlag, 1999.

[15] A. Kalyanpur, B. Parsia, E. Sirin, and B. Cuenca Grau. Repairing unsatisfiable concepts in OWL ontologies. In *Proc. of ESWC*, LNCS. Springer Verlag, 2006.

[16] A. Kalyanpur, B. Parsia, E. Sirin, and J. A. Hendler. Debugging unsatisfiable classes in OWL ontologies. *J. of Web Semantics*, 3, 2005.

[17] B. Konev, F. Wolter, and M. Zakharyaschev. Temporal logics over transitive states. In *Proc. of CADE*, LNAI. Springer Verlag, 2005.

[18] B. Konev, A. Degtyarev, C. Dixon, M. Fisher, and U. Hustadt. Mechanising first-order temporal resolution. *Information and Computation*, 199, 2005.

[19] B. Konev, R. Kontchakov, F. Wolter, and M. Zakharyaschev. On dynamic topological and metric logics. *Studia Logica*, 2006. To appear.

[20] O. Kutz, C. Lutz, F. Wolter, and M. Zakharyaschev. E-connections of abstract description systems. *Artificial Intelligence*, 156, 2004.

[21] C. Lutz, U. Sattler, and F. Wolter. Modal logics and the two-variable fragment. In *Proc. of CSL*, LNCS. Springer Verlag, 2001.

[22] C. Lutz, D. Walther, and F. Wolter. Quantitative temporal logics: PSPACE and below. *J. of Information and Computation*, 2006. To appear.

[23] C. Lutz, D. Walther, and F. Wolter. Conservative extensions in expressive description logics. In *Proc. of IJCAI*. AAAI Press, 2007. To appear.

[24] D. Walther, C. Lutz, F. Wolter, and M. Wooldridge. ATL satisfiability is indeed Exptime-complete. *J. of Logic and Computation*, 2006. To appear.

PART 2: The Proposal

A. Background

Ontologies and their Applications In computer science, ontologies are used to provide a common vocabulary for a domain of interest together with *descriptions* of the *meaning* of terms built from the vocabulary and relationships between them. Ontologies in this sense are increasingly used in knowledge management systems, medical and bioinformatics [62, 61, 45], and are set to play a key role in the semantic web and grid [33, 65]. In order to be computer-accessible, modern ontologies are formulated in an ontology language based on *description logics* such as OWL [32]. Current applications are leading to the development of large and complex ontologies (sometimes more than 300,000 different terms). Engineering and maintaining such ontologies is a complex task, and it has to be carried out with care for the ontology to be of use. It may involve a group of ontology engineers and domain experts co-operating in order to design the ontology, update it to reflect changes/developments in the domain, and integrate it with other ontologies so as to cover larger domains. For example, the National Cancer Institute (NCI) [44] has approximately twelve people working on its oncology ontology at any given time. These people are geographically distributed and range from dedicated ontologists to managers who contribute an occasional change. In the last eighteen months, the number of classes in the ontology has grown from approximate 40,000 to over 57,000 (including many changes to existing terms).

Current support The two key advantages of using a description logic [28] based ontology language such as OWL over alternative representation mechanism (such as semantic nets or frames) is that they have an unambiguous semantics, and that we can make use of *reasoning* services of description logic (DL) reasoners [60, 66, 46, 30] for ontology engineering. These services typically include

- computing the subsumption hierarchy between classes (classification),
- answering queries,
- testing the consistency of class description, and
- finding explanations for inconsistencies.

The availability of these services, especially in editors such as SWOOP and Protégé OWL [50, 48], change how ontology engineers work. Classification, for example, facilitates bottom up construction of taxonomies. So, the ontology engineer can focus on the *definition* of terms, rather than the *relations* between them.

However, these services are not sufficient for engineering and maintaining large ontologies, especially in the collaborative case. Local changes to an ontology, and interactions between such changes, can have highly non-local effects that are currently unpredictable. The only time to examine these effects is *after* the changes have been made, in the light of *all* the proposed changes. And even then there are changes whose impact is not detectable using the current suite of reasoning services. For example, they might only affect subsumptions between certain complex terms, or interact unexpectedly with future additions—which make them dangerous “time bombs”. This has lead the NCI modeling team to put stringent, though ad-hoc, laborious and still error-prone, restrictions on their work process: there is a procedure for “checking out” a class; overnight, the ontology is classified; the effects of changes are studied the next day; finally the changes must be approved by an editor before they are incorporated into the ontology.

A further problem is that ontologies are published and used as they are developed: as monolithic entities. For example, the NCI ontology is focused on cancers and genes, yet it also contains an impoverished ontology of cooking. This area is obviously of only tangential interest to the NCI: they just need to talk about the risks for certain cancers associated with char-grilled fish. However, this fragment could be used and developed by other groups, but only if the fragments can be *correctly* separated from the rest and extensions *safely* merged back in.

To sum up, the state of ontology engineering is very similar to the state of software engineering before the advent of structured programming techniques: ontologies cannot be decomposed into semantically distinct components, we cannot predict the scope of a (local) change, and how to re-use parts of ontologies or safely compose them are open problems. In software engineering, human documentation and rigorous process restrictions were put into place, as well as preliminary mechanisms for structuring programs (e.g., type and module systems). As these mechanisms have grown more sophisticated, they have led to new automated techniques for transforming programs for performance (e.g., separate compilation), understanding (e.g., refactoring), and re-use (e.g., modules).

Composing and decomposing ontologies It has been convincingly argued [58, 56, 31, 63, 9, 7] that methodologies and algorithmic support for composing and decomposing ontologies *in a controlled way* will be the key to supporting collaborative ontology engineering and re-use.

More precisely, it will be crucial to develop methodologies and algorithmic support for

T1 developing ontologies with interfaces (and acceptable restrictions on their usage) which guarantee that, if such an ontology is composed with other ontologies, it neither corrupts nor is corrupted by the ontologies they are composed with;

T2 evaluating the consequences of the composition of a set of given ontologies which may have been built in a completely unrestricted way;

T3 decomposing a large ontology into modules that can be edited in a controlled way.

The first item is *prescriptive* in the sense that the developer will be “forced” to follow a certain design method which automatically leads to well-behaved modular ontologies. The second item is *analytical* in that it supports the evaluation of the result of composing arbitrary ontologies. The last item comprises both approaches: after automatically segmenting a given ontology *analytically*, the designer might want to follow a certain methodology to ensure that editing the segment does not have damaging effects or, alternatively, might prefer to edit the segment “arbitrarily” but be supported in evaluating the effects of merging it back.

Satisfactory solutions to (T1)–(T3) are indispensable to support ontology engineers in collaborative editing and reuse of ontologies. For example, they would allow for simultaneous editing of an ontology by segmenting the ontology into modules in such a way that reasoning can be carried out locally and independently on each module, and that the effects of local changes would not corrupt the whole ontology nor other modules. No understanding of the whole ontology would be required while editing a module. As a valuable side-effect, segmenting large ontologies is seen as a key new optimization technique for reasoning, allowing both for larger ontologies to be processed, but also allowing for nearer to real time processing.

In the above enumeration, terms such as “in a controlled way”, “corrupted ontology”, and “consequence” are deliberately left rather vague. Indeed, one goal of this project is to provide rigorous but practical and useful *formal specifications* of items (T1)–(T3) above. It is only very recently that research in this direction has been carried out [25, 6, 5, 23, 9, 3, 7] and the proposers have already made some pioneering contributions towards their actual development. The approaches taken are *logic-based* and founded on the notion of *conservative extensions*.

Conservative extensions The notion of a conservative extension plays a fundamental role in mathematical logic, philosophy of science, and software specification. The *deductive* version is formulated as follows:

- a logical theory T_2 is a *conservative extension of* another theory T_1 if any consequence of T_2, which only uses symbols from T_1, is a consequence of T_1 as well.

Weaker variants of this notion are obtained by considering only consequences which are in a certain sublanguage.The stronger, *model theoretic* version,

- every model for T_1 can be expanded to a model for T_2,

is equally important in those fields. In software specification, conservative extensions have been proposed and used to define what it means that one specification *refines* another specification [67, 52, 36] and to enable *specifications in the large* by supporting structured specification, modularisation, and decomposition [55, 53].

Conservative extensions can play a fundamental role for composing and decomposing ontologies because ontologies are logical theories, and thus their “corruption” needs to be formulated in terms of changes to their logical consequences/models. Now T_2 being a conservative extension of T_1 captures exactly this: T_2 extends T_1 *without affecting* its consequences/models. Therefore, to control the consequences of a composition of two ontologies, one should start with the requirement that this composition is a suitable form of conservative extension of its components. Stronger dependencies between components should then be accommodated for by considering carefully designed exceptions (i.e., interfaces) and choosing weak notions of conservative extensions.

Although the above mentioned research on conservative extensions in software specification is highly relevant for tasks (T1)–(T3), no results can be applied directly because of the very different nature of the logics and their purpose.

B. The Research Programme

The goal of this project is to develop meaningful notions and related reasoning services for composing and decomposing ontologies to serve as the basis for the collaborative development and re-use of ontologies. As argued above, a principled, logic-based approach to support ontology developers in composing and decomposing ontologies will be based on *variants of* conservative extensions. As a first approximation, those notions will be used to specify items (T1)–(T3) above as follows:

S1 The composition of ontologies using appropriate interfaces should ensure that the composed ontology is (an appropriate form of) a conservative extension of its components. In this approach, the component ontologies are “black boxes” whose interfaces consist of subsets of their vocabulary which might be shared, connected, or imported. The appropriate notion of conservativity of the composed ontology relative to its components ensures that it has only the desired consequences and that the component ontologies are not corrupted.

S2 When composing arbitrary ontologies (without using specified interfaces) the designer should be supported by reasoning services which detect whether the composed ontology is (an appropriate form of) a conservative extension of its parts.

S3 A *module* of an ontology should be defined as a part of the ontology of which the whole ontology is an (appropriate form of) conservative extension. Moreover, it should be specified which type of editing of such a module preserves this relation.

In (S1)–(S3), conservativity plays different roles. In (S1), because the ontologies are regarded as “black boxes”, one has to ensure conservativity *for any* ontology obeying certain syntactical/semantical constraints. In contrast, in (S2), one has to decide a posteriori whether a given ontology is a conservative extension of another. The appropriate variants of the notion of a conservative extension have to be determined based on the following criteria (and might well not be same for all tasks):

- What do the users want? What kind of consequences should be preserved? E.g., should all consequences be preserved, or only the subsumption hierarchy, or something in between? We are going to develop the notions required to compare ontologies through liaisons with designers of ontologies from bio-informatics and medical informatics.
- What is the computational complexity of detecting whether one ontology is a conservative extension of another? Can we design “practical” algorithms which detect this for real ontologies?
- Which notions of conservativity have important meta-properties? For example, does it support modularity in the sense of the Robinson consistency property (see below for details)?
- For which notions do there exist transparent, syntactical, acceptable approximations? That is, restrictions that (i) can be explained to a designer and are acceptable, (ii) are syntactic so that they can be controlled by ontology

editors, and (iii) ensure that compositions with other ontologies are conservative extensions.

Research Aims. Our research aims are:

1. To specify notions of conservative extensions between DL ontologies and investigate their computational and meta-properties.
2. To provide rigorous specifications for (T1)–(T3) by applying those notions to the problems of composing and decomposing DL ontologies for both the *prescriptive* and the *analytical* approach.
3. To design algorithms which support those applications. This includes algorithms for deciding certain notions of conservativity and algorithms that use these notions for decomposing ontologies.
4. To implement and evaluate these algorithms in cooperation with ontology editor developers and ontology designers.

Programme of work The research programme below is broken down into three *workpackages* (WP), each of which is further divided into *tasks*. For each WP, we present a *description* and the *methodology* to be employed, and we name two *leading investigators* for each task. It is to be emphasized that the role of the named investigators is merely to guide and manage the research tasks since we assume that all team members cooperate in all tasks; and that the tasks of a WP are strongly connected to each other and inform each other.

WP 1. Requirements Analysis and Evaluation

OVERVIEW: The aim of WP 1 is to inform us about the requirements of ontology designers and to evaluate our approaches in realistic use cases. To this end, we will collaborate and liaise with ontology editor developers and ontology designers. This workpackage will inform work on the other packages throughout the whole project.

TASKS:

1.1. Collaboration & liaison with ontology editor developers.

Leading investigators: Parsia and Horrocks

1.2. Continuous requirements analysis.

Leading investigators: Parsia and Sattler

1.3. Evaluation.

Leading investigators: Parsia and Horrocks

DESCRIPTION AND METHODOLOGY: *Task 1.1.* In order to ensure that the problems investigated in the other workpackages can indeed provide the required support for the composition and decomposition of ontologies, and thereby supporting ontology engineering tasks as described in (T1)–(T3) above, we will continuously make use of our on-going collaborations and liaisons with ontology editor developers and with ontology engineers. This is facilitated through our direct involvement in the development of the SWOOP ontology editor [48], our on-going liaison with the developers of the Protégé OWL editor [50], and our on-going liaison with ontology engineers using these tools.[1] More precisely, we plan to extend both editors to support the ontology engineer in following the syntactic restrictions imposed by the prescriptive approach, and with interfaces to the prototypical implementations of the various reasoning services developed within WP 3.

Task 1.2. In order to ensure the relevance of the research carried out in other workpackages, we need to analyse the relevance of various parameters of this research. We will examine existing ontology engineering processes to help identify what kinds of services related to composing and decomposing of ontologies are useful. Our preliminary work has identified the areas described in (T1)–(T3), which involve the following services:

- to help the ontology designer to develop ontologies that can be composed in a controlled way, i.e., guide the designer in following the prescriptive approach;
- to show, to the ontology designer, the consequences of a composition of ontologies;
- to decompose an ontology into modules; and
- to extract, from an ontology, a component that is "relevant" for a given set of terms.

Within this task, we plan to extend this list and describe these services in detail, e.g., by analyzing what notion of conservativity is useful for which service, how acceptable certain syntactic restrictions are to the engineer, or what acceptable performance is.

Task 1.3. We plan to test the usability and performance of the services identified in Task 1.1 in two different ways. (1) Similar to our field study described in [7], we will make use of existing ontologies [42] to evaluate the performance of our services and analyze their outcome. This will provide some insight into their applicability and scalability, but will only partially answer questions concerning their usefulness for ontology engineering in practice. (2) Therefore, we will make use of the extensions to ontology editors undertaken in Task 1.2 and our ongoing liaison with ontology engineers to evaluate our services in practice, i.e., during ontology engineering. This will provide insight into the acceptability of the syntactic restrictions imposed in the prescriptive approach and the acceptability of the performance of our prototypical implementation for the analytical approach.

WP 2. Notions of conservative extensions, their meta-properties, and their decision problems

OVERVIEW: The aim of this workpackage is to develop a hierarchy of notions of conservative extensions for DLs, characterize them model-theoretically, investigate their meta-properties, and investigate the complexity of the corresponding decision problems.

TASKS:

2.1. Develop a hierarchy of notions of conservativity for different DLs and characterize them model-theoretically.

Leading investigators: Sattler and Wolter

2.2. Investigate important meta-properties (e.g., Robinson-consistency, existence of uniform interpolants, and distribution over projections) of notions of conservativity for different DLs.

---

[1] See letters of support from Matthew Horridge, Mark Musen, NCI, Robert Stevens, FreshWaterLife, OrdnanceSurvey, Alan Ruttenberg, Aditya.

Leading investigators: Konev and Wolter

2.3. Investigate the computational complexity of deciding conservativity for different DLs and notions of conservativity.

Leading investigators: Konev and Wolter

DESCRIPTION AND METHODOLOGY: *Task 2.1* will isolate, in a continuous feedback loop with WP 1, the *parameters* of conservative extensions. Which consequences are to be preserved and what is the model-theoretic meaning of the resulting variants of conservative extensions? Answers to these questions will provide formal specifications of the terms "in a controlled way", "corrupted ontology", and "consequence" used in the description of our items (T1)–(T3). Different DLs will give rise to different model-theoretic characterizations and, for each DL, we will obtain a *hierarchy* of notions of conservative extensions. The DLs we take into account will range from sub-Boolean DLs such as $\mathcal{EL}$ [27, 30] and DL-Lite [37] to very expressive ones such that $\mathcal{SHIQ}$ and $\mathcal{SHOIQ}$ (the DLs underlying OWL) and expressive variants of the modal μ-calculus [34]. The notions considered will range from the standard model-theoretic notion of conservativity to conservativity for conjunctive queries to underlying databases [26] and the standard deductive version of conservativity to much weaker versions such as preservation of the subsumption hierarchy. First results [5, 23] as well as expressivity results for modal, temporal and description logics show that various forms of *bisimulations* [51, 68, 40] will play an important role in characterizing those notions model-theoretically.

Task 2.2 is devoted to investigating important meta-properties of the notions developed in Task 1. An example is the following variant of the Robinson-consistency property (RC) [38]: suppose T_1 and T_2 are both conservative extensions of T_0 and T_0 contains all shared symbols of T_1 and T_2. Then $T_1 \cup T_2$ is a conservative extension to T_0 as well. For cooperative ontology design, this property is crucial: if it holds, then ontologies developers can safely join their independently developed extensions. Whether RC holds depends on the DL and the chosen notion of conservativity. For decomposing ontologies, the following problem is important: given an ontology T and a subset V of its vocabulary, does there exist an ontology T' over V of which T is a conservative extension. This problem is closely related to a uniform interpolation property [43, 68], which depends again on the language and notion of conservative extension chosen. Informed by WP 1, we will investigate these and related properties. The investigation will be based on the model-theoretic characterizations from Task 2.1 and extend and modify techniques such as amalgamation of models, saturated models, and representation of terms by tree automata [64, 38, 41].

Task 2.3 consists in choosing notions of conservative extensions from Task 2.1 together with well-suited DLs and investigate the computational complexity of the following problems for their combination:

- "Is ontology T_2 a conservative extension of ontology T_1?" and "If this is not the case, compute a counterexample", where a counterexample is, depending on the variant of conservative extension investigated, a consequence that is *not* preserved or a model which cannot be expanded.
- "Is T_2 a conservative extension *of every* T_1 (of a certain syntactic form)?" and "If this is not the case, compute a counterexample?"

These problems are both fundamental for the analytical approach and provide the theoretical background for the *prescriptive* approach. First results on the first problem for standard model-theoretic and deductive versions of conservativity have been obtained in [5, 23]. They indicate that the model-theoretic variant will often be more complex (even highly undecidable) than the deductive versions (which is often 2ExpTime-complete). Its complexity depends in a subtle way on the expressivity of the DL under consideration. No results have yet been obtained for the second, quantified, version, but we hope that in some cases the decision problem becomes simpler. When investigating those problems, we will use the model-theoretic characterisations from Task 2.1 and try to extend the methods developed in [5, 23]. However, we will also use automata theoretic approaches [47, 41] and make use of techniques introduced for investigating uniform interpolants and bisimulation quantifiers [68, 39, 41].

WP 3. "Practical" algorithms for deciding conservativity and decomposing ontologies, and syntactic conditions for conservativity

OVERVIEW: The aim of this workpackage is to develop, for the *analytical approach*, "practical" reasoning algorithms for deciding conservativity and decomposing ontologies, and evaluate them using a prototype implementation. For the *prescriptive approach*, sufficient syntactic conditions for conservativity will be developed and corresponding decomposition algorithms will be developed and implemented.

TASKS:

3.1. Development and implementation of reasoning algorithms for checking conservativity for appropriate DLs and notions of conservativity.

Leading investigators: Konev and Sattler

3.2. Development of sufficient syntactic conditions for conservativity.

Leading investigators: Horrocks and Sattler

3.3. Development and implementation of reasoning algorithms for decomposing ontologies.

Leading investigators: Wolter and Sattler

DESCRIPTION AND METHODOLOGY: Work in this workpackage will be based on the requirements identified in WP 1 and on the insights gained through the analysis of different notions of conservative extensions and their computational complexity in WP 2. The results will be evaluated in WP 1.

Task 3.1. We will choose relevant notions of conservative extensions and develop "practical" reasoning algorithms for them for important DLs, which will provide solutions to (T2) above. This will involve the exploration of new terrain in automated reasoning for DLs, and the reasoning techniques employed will depend on the kind of conservativity and the DL. Using the model-theoretic characterizations from WP 2, it may turn out that model-generating algorithms such as tableau algorithms (which are underlying most DL-reasoners) can be expanded to decide conservativity. Automata- and resolution-based approaches are also of interest. For example, for some DLs, it appears promising to encode the conservativity problems into the satisfiability problem for monadic second-order logic over trees and thus use techniques similar to those underlying the MONA system [49]. The investigators have profound expertise in both areas; see, e.g., [57, 29, 12, 13]. Moreover, it will depend on the requirements determined in WP 1 and the computational complexity results from WP 2 whether we will work towards decision procedures or towards semi-decision procedures: our preliminary work shows that there could be interesting notions of conservativity for widely used DLs that are undecidable, and thus cannot be solved by decision procedures. We expect a prototypical implementation of some of the reasoning algorithms to be built on existing DL reasoners [60, 66, 54] since, in principle, any algorithm for deciding conservativity and computing witness concepts will require satisfiability procedures as sub-routines. Testing and evaluation of the algorithms will be carried out on the ontologies collected within WP 1.

Task 3.2. We will use the insight gained in WP 2 to formulate sufficient (but mostly not necessary) syntactic conditions for suitable notions of conservativity and different DLs, which will provide solutions to (T1) above. These syntactic conditions will be used to define notions of a module with an interface together with restrictions of its usage. Preliminary results in this direction [7, 9] have been obtained. They suggest that sufficient, yet reasonably general, syntactic conditions for conservativity in the model-theoretic sense can be obtained by considering so-called “local” ontology axioms. We will try to extend this approach, but also consider the suitability of syntactic restrictions derived from specialized languages for modular DLs and combining DLs such as distributed DLs [59, 35], E-connections [20], and package-based DLs [31].

Task 3.3. We will develop algorithms for the decomposition of one ontology into well-behaved modules, i.e., into parts structured by dependencies, which will provide solutions to (T3) above. So far, we have identified two versions of this service: (1) to decompose an ontology into modules that can be reasoned over independently and on which ontology engineers can work either completely independently or with dependencies that are known in advance, and (2) to extract, from an ontology, a module that “captures” a certain set of terms, so that this module can be re-used in another ontology without having to use the whole ontology. Here, for a module T_m of T to “capture” a set of terms means that the composition of an ontology with T_M leads to the same consequences as the composition with T, i.e., that T_M is a uniform interpolant of T as described in Task 2.2.

Whether solutions to these services will be based on the algorithms developed in Task 3.1 or the sufficient syntactic conditions developed in Task 3.2, will depend on the performance of the algorithms of Task 3.1 and the suitability of syntactic conditions in Task 3.2. Preliminary results based on sufficient syntactic conditions for conservativity are promising [9, 7]. Moreover, it should be possible to use non-logic-based approaches to modularization [56, 58, 63] to generate candidate decompositions. To apply them, it has to be checked which logical consequences they preserve.

References

[25] G. Antoniou and K. Kehagias. A note on the refinement of ontologies. *Int. J. of Intelligent Systems*, 15, 2000.

[26] C. Lutz B. Glimm, I. Horrocks and U. Sattler. Conjunctive query answering for the description logic SHIQ. In *Proc. of IJCAI*. AAAI Press, 2007. To appear.

[27] F. Baader, S. Brandt, and C. Lutz. Pushing the $\mathcal{EL}$ envelope. In *Proc. of IJCAI*. Morgan Kaufmann Publishers, 2005.

[28] F. Baader, D. Calvanes, D. McGuiness, D. Nardi, and P. Patel-Schneider. *The Description Logic Handbook: Theory, implementation and applications*. Cambridge Univ. Press, 2003.

[29] F. Baader, J. Hladik, C. Lutz, and F. Wolter. From tableaux to automata for description logics. *Fundamenta Informaticae*, 57, 2003.

[30] F. Baader, C. Lutz, and B. Suntisrivaraporn. CEL—a polynomial-time reasoner for life science ontologies. In *Proc. of IJCAR*, LNAI. Springer Verlag, 2006.

[31] J. Bao, D. Caragea, and V. Honavar. On the semantics of linking and importing in modular ontologies. In *Proc. of ISWC*, LNCS. Springer Verlag, 2006.

[32] S. Bechhofer, F. van Harmelen, J. Hendler, I. Horrocks, D. L. McGuinness, P. Patel-Schneider, and L. Stein. OWL Web Ontology Language reference. W3C Recommendation, 2004.

[33] T. Berners-Lee, J. Hendler, and O. Lassila. The semantic Web. *Scientific American*, 284(5), 2001.

[34] P. Bonatti, C. Lutz, A. Murano, and M. Vardi. The complexity of enriched μ-calculi. In *Proc. of ICALP*, LNCS. Springer Verlag, 2006.

[35] P. Bouquet, F. Giunchiglia, F. van Harmelen, L. Serafini, and H. Stuckenschmidt. Contextualizing ontologies. *Journal of Web Semantics*, 1, 2004.

[36] P. Byers and D. Pitt. Conservative extensions: A cautionary note. *EATCS-Bulletin*, 1990.

[37] D. Calvanese, G. De Giacomo, D. Lembo, M.Lenzerini, and R. Rosati. Dl-lite: Tractable description logics for ontologies. In *Proc. of AAAI*. AAAI Press, 2005.

[38] C. Chang and H. Keisler. *Model Theory*. Elsevier, 1990.

[39] G. D’Agostino and M. Hollenberg. Logical questions concerning the μ-calculus: interpolation, Lyndon and Łoś-Tarski. *J. Symbolic Logic*, 65, 2000.

[40] G. D’Agostino and G. Lenzi. An axiomatization of bisimulation quantifiers via the μ-calculus. *Theoret. Comput. Sci.*, 338, 2005.

[41] T. French. *Bisimulation quantifiers for modal logics*. PhD thesis, University of Western Australia, 2006.

[42] T. Gardiner, I. Horrocks, and D. Tsarkov. Automated benchmarking of description logic reasoners. In *Proc. of DL*. CEUR, 2006.

[43] S. Ghilardi and M. Zawadowski. Undefinability of propositional quantifiers in the modal system S4. *Studia Logica*, 55, 1995.

[44] J. Golbeck, G. Fragaso, F. Hartel, J. Hendler, B. Parsia, and J. Oberhaler. The national cancer institute’s thesaurus and ontology. *J. of Web Semantics*, 2003.

[45] C. Golbreich, S. Zhang, and O. Bodenreider. The foundational model of anatomy in OWL: Experience and perspectives. *J. of Web Semantics*, 4(3), 2006.

[46] V. Haarslev and R. Möller. RACER system description. In *Proc. of IJCAR*, LNAI. Springer Verlag, 2001.

[47] D. Janin and I. Walukiewicz. Automata for the modal μ-calculus and related results. In *Proc. of MFCS*, LNCS. Springer Verlag, 1995.

[48] A. Kalyanpur, B. Parsia, E. Sirin, B. Cuenca- Grau, and J. Hendler. SWOOP – a web ontology editing browser. *J. of Web Semantics*, 4(1), 2005.

[49] E. Karabaev and C. Lutz. Mona as a DL reasoner. In *Proc. of DL*, CEUR, 2004.

[50] H. Knublauch, M. Musen, and A. Rector. Editing description logic ontologies with the Protégé OWL plugin. In *Proc. of DL*. CEUR, 2004.

[51] N. Kurtonina and M. de Rijke. Expressiveness of concept expressions in first-order description logics. *Artificial Intelligence*, 107, 1999.

[52] T. Maibaum. Conservative extensions, interpretations between theories and all that! In *Proc. of CAAP/FASE*, LNCS. Springer Verlag, 1997.

[53] P. Mosses, editor. *CASL Reference Manual*. LNCS. Springer Verlag, 2004.

[54] B. Motik and U. Sattler. A comparison of reasoning techniques for querying large description logic ABoxes. In *Proc. of LPAR*, LNCS. Springer Verlag, 2006.

[55] J. Goguen R. Diaconescu and P. Stefaneas. Logical support for modularisation. In G. Huet and G. Plotkin, editors, *Logical Environments*, 1993.

[56] A. Rector. Modularisation of domain ontologies implemented in description logics and related formalisms including OWL. In *Proc. of K-CAP*. ACM Press, 2003.

[57] U. Sattler and M. Y. Vardi. The hybrid mu-calculus. In *Proc. of IJCAR*, LNAI. Springer Verlag, 2001.

[58] J. Seidenberg and A. Rector. Techniques for segmenting large description logic ontologies. In *K-CAP 2005 Workshop on Integrating Ontologies*. CEUR, 2005.

[59] L. Serafini, A. Borgida, and A. Tamilin. Aspects of distributed and modular ontology reasoning. In *Proc. of IJCAI*. Morgan Kaufmann Publishers, 2005.

[60] E. Sirin, B. Parsia, B. Cuenca Grau, A. Kalyanpur, and Y. Katz. Pellet: A practical OWL-DL reasoner. Submitted to the Journal of Web Semantics.

[61] K.A. Spackman. Managing clinical terminology hierarchies using algorithmic calculation of subsumption: Experience with SNOMED-RT. *J. of the Amer. Med. Informatics Ass.*, 2000.

[62] R. Stevens, C. Goble, I. Horrocks, and S. Bechhofer. Building a bioinformatics ontology using OIL. *IEEE Transactions on Information Technology in Biomedicine*, 6, 2002.

[63] H. Stuckenschmidt and M. Klein. Structure-based partitioning of large class hierarchies. In *Proc. of ISWC*, LNCS. Springer Verlag, 2004.

[64] B. ten Cate, W. Conradie, M. Marx, and Y. Venema. Definitorially complete description logics. In *Proc. of KR*. AAAI Press, 2006.

[65] The DAML Services Coalition. DAML-S: Web service description for the semantic web. In *Proc. of ISWC*, LNCS. Springer Verlag, 2003.

[66] D. Tsarkov and I. Horrocks. Optimised classification for taxonomic knowledge bases. In *Proc. of DL*. CEUR, 2005.

[67] W.M. Turski and T. Maibaum. *The Specification of Computer Programs*. Addison-Wesley, 1987.

[68] A. Visser. Uniform interpolation and layered bisimulation. In *Gödel ’96 (Brno, 1996)*, volume 6 of *Lecture Notes Logic*. Springer Verlag, 1996.

Relevance to beneficiaries

A wide range of UK and international research efforts in areas such as the Semantic Web/Grid, knowledge management, medical-informatics, bio-informatics and e-Science are now dependent on ontologies to capture domain semantics and to promote interoperability—this is evidenced by the widespread use of other ontology design tools such as Protégé and SWOOP [48, 50]. As argued above, it is crucial to support the design and maintenance of these ontologies with services to enable collaborative editing and re-use of ontologies, both of which are based on controlled composition and decomposition of ontologies.

In addition to its direct benefit to those who aim to develop and/or use ontologies, this work will also benefit the wider UK research community: in the first place, it will deepen our understanding of modularity for logical theories in general (not only DLs but also temporal logics/modal logics), and in the second place, it will help to consolidate the UK’s already established world leadership in research on knowledge representation and reasoning in general and ontology languages and reasoning in particular.

In the longer term, this work will also be of benefit to the increasing number of companies who are developing ontology design/deployment tools and ontology based applications. This includes technology companies such as Siemens, and IBM, and ontology users such as NASA, Freshwaterlife, and Ordnance Survey.

compare with letters of support!

Dissemination and Exploitation

Dissemination will be via three channels: collaboration with ontology tool developers and ontology engineers (see WP 1), presentations at relevant national and international meetings, and publications in leading conferences and journals.

As described in WP 1, the developers of SWOOP are directly involved in this project, and we liaise with the developers of Protégé in order to make the reasoning services we develop available to ontology engineers. We will also work with the ontology engineers themselves in order to encourage adoption of the new services.

Dissemination will also be pursued via presentations, courses and tutorials at relevant meetings such as Manchester’s successful series of OWL tutorials, GO Users Meetings, Protégé workshops, and various summer schools.

Finally, we will continue our established pattern of conference and journal publications. These will target the wide range of communities where the proposed research will be of relevance, including logic and automated reasoning, knowledge representation, the Semantic Web/Grid, medical- and bio-informatics.

Management

The project will be managed by Sattler and Wolter, with Sattler being in overall charge; both have considerable experience managing EPSRC and EU projects, and have worked together, e.g., on the EPSRC DynamO project (GR/S63168/01). They will monitor and discuss the progress of the project via email and monthly meetings, the latter being possible due to the close proximity between Liverpool and Manchester. The proposed research and time schedule are ambitious, but is achievable given the strength of the team and the fact that the project builds on existing work and expertise. Moreover, the structure of the project helps to mitigate risk: although the tasks complement and inform each other, many of them can also succeed independently. For example, even if one of the two approaches, the analytic or the prescriptive, proves to be unfeasible in some applications, the other one could still be extremely successful in these applications.