A Realistic Architecture for the Semantic Web

Michael Kifer[1], Jos de Bruijn[2], Harold Boley[3], and Dieter Fensel[2]

[1]Department of Computer Science
State University of New York at Stony Brook
Stony Brook, NY 11794
kifer AT cs.sunysb.edu

[2]Digital Enterprise Research Institute (DERI),
University of Innsbruck, Austria
National University of Ireland, Galway
{jos.debruijn, dieter.fensel} AT deri.org
http://www.deri.org/

[3]Institute for Information Technology - e-Business,
National Research Council of Canada,
Fredericton, NB, E3B 9W4, Canada
Harold.Boley AT nrc-cnrc.gc.ca

Abstract. In this paper we argue that a realistic architecture for the Semantic Web must be based on multiple independent, but interoperable, stacks of languages. In particular, we argue that there is a very important class of rule-based languages, with over thirty years of history and experience, which cannot be layered on top of OWL and must be included in the Semantic Web architecture alongside with the stack of OWL-based languages. The class of languages we are after includes rules in the Logic Programming style, which support default negation. We briefly survey the logical foundations of these languages and then discuss an interoperability framework in which such languages can co-exist with OWL and its extensions.

1 Introduction

An alternative architecture for the Semantic Web was recently proposed by several groups at the W3C Workshop on Rule Languages for Interoperability[1] and presented in the talk "Web for real people" by Tim Berners-Lee.[2] An older architecture, depicted in Figure 1, assumed that the main languages that comprise the Semantic Web should form a single stack and every new development in that area should build on top of the previous linguistic layers.[3] The older layers at the lower part of the stack are supposed to be upward compatible with the new developments, and in this way any investment made in the old technology will be preserved as the Semantic Web technology matures and expands.

[1] http://www.w3.org/2004/12/rules-ws/

[2] http://www.w3.org/2005/Talks/0511-keynote-tbl/

[3] However, SparQL (http://www.w3.org/TR/rdf-sparql-query/) has recently joined as a language sitting outside of the stack.

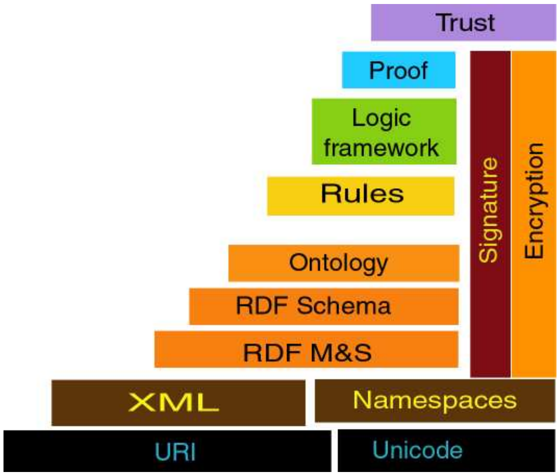

Fig. 1. Old Semantic Web Stack

While a single-stack architecture would hold aesthetic appeal and simplify interoperability, many workers in the field believe that such architecture is *unrealistic* and *unsustainable*. For one thing, it is presumptuous to assume that any technology will preserve its advantages forever and to require that any new development must be compatible with the old. If this were a law in the music industry (for example) then MP3 players would not be replacing compact disks, compact discs would have never displaced tapes, and we might still be using gramophones.

The Semantic Web has had a shorter history than music players, but it already saw its share of technological difficulties. RDF(S), the first layer in the Web architecture that was dubbed "semantic" [LS99], was proposed and standardized without ... a semantics. Years later a group of logicians was seen scrambling to define a formal semantics that would be reasonably compatible with the informal prose that was accompanying the official W3C's RDF(S) recommendation. The resulting semantics for RDF(S) [Hay04] was certainly formal, but one might question whether the decision to follow the original *informal* RDF semantics (and even syntax) *to the letter* was justified. The difficulties that this created for OWL [DS04]—the subsequent and more expressive layer of the Semantic Web—are well-documented in [HPSvH03]. It is well-known, but rarely mentioned, that OWL-DL is not properly layered on top of RDF(S),[4] and that the single-stack architecture for the budding Semantic Web technology already has a small crack.

The alternative architecture proposed at the workshop recognizes the difficulties (even at the philosophical level) with the single-stack architecture (henceforth called *SSA*). The key idea is that a more realistic architecture must allow

[4] For instance, there are statements that are valid RDF, but not OWL-DL.

multiple technological stacks to exist side-by-side, as in Figure 2. Ideally, adjacent stacks should be interoperable to a high degree, but when this is not possible a loosely coupled integration will be acceptable in practice.

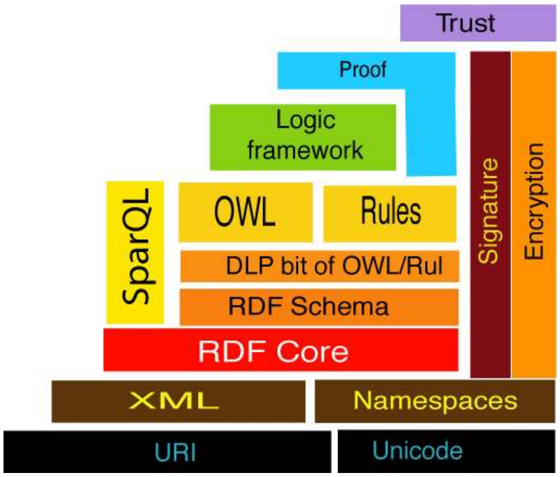

Fig. 2. New Semantic Web Stack

The new architecture is more realistic not only because, in the long run, a single stack architecture could saddle us with the Semantic Web equivalent of a gramophone, but also because it would make us use a gramophone to toast bread and mow grass. That is, it is a vain hope that a single upward-compatible language, developed in the Semantic Web's infancy, will suffice for all the future semantic chores to be done on the Web. This expectation is certainly not borne out of the fifty years of experience with programming languages.

To avoid any misinterpretation, it is *not* our intention to claim that the existing parts in the current stack of Semantic Web technologies are obsolete. However, every technology, including language design, eventually becomes obsolete, and no technology can address all problems.

We are therefore convinced that the multi-stack architecture (henceforth called *MSA*) is timely now because limitations of the currently standardized technology are already felt on the Semantic Web—especially in the subarea of rules. The need for rule-based processing on the Semantic Web was envisioned at the very early stage of the development [Bol96,FDES98,MS98,DBSA98], even before the very term "Semantic Web" was coined in, and the rule/ontology combination SHOE was already implemented and used in a web context to reason with annotations of web resources before the rise of XML [HHL03]. Now that OWL has standardized the base level of ontology specification and RuleML [BTW01] has provided a standard serialization layer, rules for the Web have become the focus of intense activity. SWRL [HPSB$^+$04], SWSL-Rules [BBB$^+$05], and WRL [ABdB+05] are some of the languages in this domain that have been proposed recently.

SWRL is a new technology, which extends OWL-DL and permits the use of Description Logic with certain kinds of rules. However, Description Logic is not a technology that comes to mind when one hears about "rules." The use of rules for knowledge representation and intelligent information systems dates back over thirty years. By now it is a mature technology with decades of theoretical development and practical and commercial use. The accumulated experience in this area exceeds the experience gathered with the use of Description Logics and the field is arguably more mature when it comes to rules.[5]

What does the SSA vs. MSA discussion have to do with rules? The problem is that the mature technology for rule-based applications mentioned in the previous paragraph is based on *logic programming* [Llo87] and *nonmonotonic reasoning* (*LPNMR*), which is not fully compatible with classical first-order logic (*FOL*) on which OWL and SWRL are built. The aforesaid Web rules proposals, WRL and SWSL-Rules, are based on LPNMR. Few people realize that SQL, arguably the most important rule-based language, has LPNMR as its foundation. Thus, while the OWL-based stack is undoubtedly important and SWRL will find its uses, the vast majority of the rule-based applications cannot be done *in principle* in SWRL or cannot be done *conveniently and efficiently.* This problem gives rise to the second main stack in the MSA diagram in Figure 2.

In the rest of this paper we briefly sketch the ideas underlying LPNMR and their motivation. We then describe interoperability frameworks for the rule-based stack and the OWL-based stack and also address the recent critique of MSA that appeared in [HPPSH05].

2 The Underpinnings of the Rules Stack

In a recent paper [HPPSH05], a number of arguments were laid out to suggest that the layers of the rules stack in Figure 2 are not properly extending each other. In particular, the paper claimed that the DLP layer is not properly extended by the Datalog layer. Unfortunately, it seems that [HPPSH05] mostly argues against a strawman that it itself constructed. To address this criticism, we need to clarify the relationship between the different layers of the rules stack in Figure 2.

A common feature of all the layers in the rules stack is that logical specifications are divided in two categories: *rule sets* and *queries.* A rule set is a set of statements—called *rules*—of the form[6]

$$\texttt{head} \ :- \ \texttt{body} \qquad (1)$$

[5] Some say that there are many more OWL descriptions on the Web than there are rule-based programs, but this argument compares apples and oranges. In which column do we place online databases and SQL applications?

[6] The actual syntax varies. For instance, sometimes rules are written as `body => head`.

The head is an atomic formula[7] and the body is a conjunction of literals. A *literal* is either an atomic formula or a negation of such a formula. The most common form of negation used in the rule bodies is *default* negation (more on this later). However, extensions that permit weakened forms of classical negation (both in the rule heads and bodies) have been studied [GL91,Gro99]. Finally, we should note that all variables in a rule are assumed to be *universally quantified* outside of the rule.

Various other syntactic extensions of rules exist, which allow disjunctions and even explicit quantifiers in the rule body, and conjunctions in the rule head. However, we will not deal with these extensions here.

A *fact* is a special kind of a rule where the body part is an empty (hence, *true*) conjunction. Often, facts are also considered ground (variable-free), although sometimes universally quantified variables are allowed as well.

A *query* is a statement of the form

$$\exists \overline{\mathtt{X}}\,(\mathtt{atomicFormula}) \tag{2}$$

where $\overline{\mathtt{X}}$ is a list of all variables in `atomicFormula`. In general, queries can be much more general. For instance, instead of `atomicFormula`, conjunctions of atomic formulas and of their negation can be allowed. However, such queries can be reduced to the form (2) by introducing rules with `atomicFormula` in the head.

An *answer* to such a query with respect to a rule set **R** is a list of values $\bar{v}$ for the variables in $\overline{\mathtt{X}}$ such that **R** entails (according to the chosen semantics) $\mathtt{atomicFormula}'$, denoted[8]

$$\mathbf{R} \;\mid\!\approx\; \mathtt{atomicFormula}' \tag{3}$$

where $\mathtt{atomicFormula}'$ is obtained from `atomicFormula` by consistently replacing each variable in $\overline{\mathtt{X}}$ with the corresponding value in $\bar{v}$.

Thus, in rule-based systems, entailment is limited to inferencing of sets of *facts* only (or their negation, if the language includes negation). This is quite different from first-order logic systems, such as Description Logic and OWL, where more general formulas can be inferred.

We now examine the layers of the rules stack in more detail. We start with *Description Logic Programs* (DLP) [GHVD03] and then clarify their relationship with the RDF layer below and the more expressive layers above.

Description Logic Programs layer. The *rule-set part* of the DLP layer is a set of all statements in Description Logic that are translatable into Horn rules [GHVD03]. A *Horn rule* is a rule of the form (1) where the head and the body consist of only atomic formulas (no negation of any kind). For Horn rules, the entailment used in (3) is classical first-order.

[7] Typically of the form $\mathtt{predicate}(\mathtt{arg}_1, ..., \mathtt{arg}_N)$, but can also have other forms, if extensions of predicate logic, such as HiLog [CKW93] or F-logic [KLW95] are used.

[8] We use $\mid\!\approx$ instead of $\models$ to emphasize that the entailment relation used in rule languages is typically nonmonotonic and, therefore, non-classical.

The *query part* of DLP is of the form (2) above. Thus, even though DLP is a subset of Description Logic, the only entailments that are considered in the DLP layer are inferences of atomic formulas. (Note that [GHVD03] also defined DHL—*Description Horn Logic*—which is like DLP, but arbitrary entailments are allowed from the rules.)

Since DLP is translated into Horn rules, the entailment in (3) is the *classical entailment* in first-order logic and, therefore, the *semantics of DLP in the OWL stack and in the rules stack are the same.*

RDF layer. In the architecture diagrams, DLP (and the rest of OWL) is depicted as sitting on top of the RDF layer. This statement requires clarifications. From the logical point of view, RDF graphs are largely just sets of facts. However, RDF also includes two additional interesting features. The first feature, *reification*, cannot be modeled in DLP or even in more general description logics. In that sense, neither DLP nor OWL-DL truly reside on top of RDF. Second, RDF has so-called *blank nodes.* RDF graphs that contain blank nodes logically correspond to sets of atomic formulas that include existentially quantified variables. This is not a problem for description logics in general, but (at a first glance) seems to be a problem for DLP, since the latter does not allow existential variables in rule heads (and thus neither in facts).

However, it turns out that extending Horn rules to accommodate existentially quantified facts is not difficult as long as the queries are still of the form (2) above. Indeed, if $\mathbf{R}$ is a set of Horn rules plus facts, where some facts are existentially quantified, then the entailment (3) holds (where $\models\!\approx$ should be understood as classical first-order logic entailment) if and only if $\mathbf{R}' \models \mathtt{atomicFormula}'$, where $\mathbf{R}'$ is a skolemization of $\mathbf{R}$ (i.e., is obtained from $\mathbf{R}$ by consistently replacing the occurrences of existential variables with new constants).

Thus, skolemization appears to be the right way to deal with blank nodes and with embedding RDF in DLP, and this is how various implementations of the N3 rules language for RDF [BL04] treat blank nodes, anyway.

Reification can also be added to DLP (and to all the layers above it in the rules stack) along the lines of [YK03]. Therefore, an extension of DLP can be said to reside on top of RDF.

Datalog layer. Datalog [MW88] is a subset of Horn logic that does not use function symbols. Since DLP is a subset of description logic, it does not use function symbols either and, therefore, the translation of DLP into rules yields a subset of Datalog.

Strictly speaking, Datalog cannot be said to reside on top of DLP, since the latter uses the syntax of description logics, which is different from the syntax of rules (1). However, Datalog certainly resides on top of the image of DLP under the translation described in [GHVD03]. Therefore, modulo such a translation, Datalog can be said to extend DLP. Later on, we will define this notion precisely.

Default negation. Default negation is an inference rule associated with a negation operator, `not`, that derives new information based on the inability to derive some other information. More precisely, `not q` may be derived because `q` cannot be. This type of negation has been a distinguishing feature of rule-based languages for more than thirty years. With such an inference rule, given a rule-base with the single rule `p :− not q`, we can derive `p` because `not q` can be derived by default (since `q` is not derivable).

One of the main reasons for the emergence of default negation is that it is impractical, and often impossible, to write down all the negative facts that might be needed in a knowledge base in order to take advantage of the classical negation. It is a common practice in knowledge representation to specify only positive true facts and leave derivation of the negative facts to the default negation rule. Default negation is also often associated with common sense reasoning used by humans who tend to conclude non-existence of something because existence is not positively known.

Default negation is sometimes also referred to as *negation as failure*. This terminology is unfortunate, since negation as failure is the traditional name for a specific form of default negation [Cla78]—one that is used in the Prolog language. Negation as failure (as used in Prolog) is known to be problematic [ABW88] and modern logic programming languages use either the *well-founded* default negation [GRS91] or the one based on *stable models* [GL88].

It is well-known that the default negation layer is a semantic and syntactic extension of the Datalog layer in the sense defined below.

Default negation is not the only nonmonotonic inference rule that we deem to be important on the rules stack of MSA. A related inference rule, called *default inheritance*, is used in object-oriented knowledge representation. F-logic [KLW95] offers a comprehensive logical framework, which supports default inheritance, and this inference rule is implemented in most F-logic based systems, such as FLORA-2 [YK02,YKZ03,Kif05] and Ontobroker [Ont].

Constraints. Support for database-style constraints is another important feature of knowledge representation on the rules stack.

Logic languages that are based on pure first-order logic, like OWL, do not support constraints and have no notion of *violation* of a constraint. Instead, they provide *restrictions*, which are statements about the desired state of the world. Unlike constraints, restrictions may produce new inferences. For instance, if a person is said to have at most one spouse and the knowledge base records that John has two, Mary and Ann, then OWL would conclude that Mary and Ann is the same person. In contrast, a rulebase with nonmonotonic semantics will view such a knowledge base as inconsistent.

The semantics of database constraints is closely related to nonmonotonic reasoning, since it relies on the notion of canonical models—a subset of models that are considered to be "correct"—and focusing on canonical models is a standard way of defining the semantics for default negation [Sho87]. In contrast, pure first-order logic based semantics considers *all* models of a theory. Therefore, database constraints belong on the rules stack of MSA.

Additional layers. The rules stack can be further extended with additional layers of which the more interesting ones include *classical negation*, *prioritized rules*, *object-orientation*, and *higher-order syntax*.

Extensions that permit classical negation alongside default negation have been proposed in [GL91,Gro99] and were implemented in a number of systems. Rule prioritization is part of Courteous Logic Programming [Gro99], and is supported by the Sweet Rules system.[9] Object-oriented extensions inspired by F-logic [KLW95] and HiLog higher-order extensions [CKW93] are part of the FLORA-2 system [YKZ03]. In fact, SWSL-Rules — a language that incorporates all of these layers have also been recently proposed [BBB^{+}05].

3 Scoped Inference

In (2), logical entailment happens with respect to an explicitly specified knowledge base, **R**. The assumption that the underlying knowledge base is known is a cornerstone of traditional knowledge representation. The Semantic Web challenges this assumption, since the boundaries of the Web cannot be clearly delineated. Therefore, the notion of inference on the Semantic Web needs to be revisited.

One idea that is beginning to take hold is the notion of *scoped inference*. The idea is that derivation of any literal, `q`, must be performed within the scope of an explicitly specified knowledge base. Different scopes can be used for different inferences, but the scope must always be declared. Scoped inference is an important feature of several knowledge representation systems for the Web. In FLORA-2 [YKZ03], the entire knowledge base is split into *modules* and inference is always made with respect to a particular module. In TRIPLE [SD02], the same idea goes under the name of a *context*.

Scoped inference can be realized using the notion of modules, as in FLORA-2 and TRIPLE, where the definition of a scope can be based on URIs, which dereference to concrete knowledge bases.

Related to the notion of scoped inference is an extension of the concept of default negation, called *scoped default negation*.[10] The idea is that the default negation inference rule must also be performed within the scope of an explicitly specified knowledge base. That is, `not q` is said to be true with respect to a knowledge base **K** if `q` is not derivable from **K**. A version of this rule is supported by some systems, such as FLORA-2, and is discussed in [Kif05].

While scoped inference is clearly useful even for deriving positive information, scope is imperative for deriving negative information from knowledge published on the Web. In fact, due to the open nature of the Web, it is not even meaningful to talk about the *inability* to derive something from a knowledge base whose bounds and the exact content are not known. On the other hand, with explicit

[9] http://sweetrules.projects.semwebcentral.org/

[10] This concept sometimes goes under the name *scoped negation as failure* or *SNAF*, which is unfortunate terminology for the reasons stated earlier.

scope, default negation becomes not only a meaningful derivation rule on the Web, but also as useful as in traditional knowledge bases.

4 The Relationship Between Layers

The layers of the rules stack are progressive syntactic and semantic extensions of each other (modulo the aforesaid caveats pertaining the RDF layer). Formally, this means that each layer is a syntactic and semantic extension of the previous layer, as defined next.

Language extensions. Let $L_1 \subseteq L_2$ be two logic languages and suppose their semantics are defined using the entailment relations $\models_1$ and $\models_2$. L_2 is said to be an *extension* of L_1 if for any pair of formulas $\phi, \psi \in L_1$, the entailment $\phi \models_1 \psi$ holds iff $\phi \models_2 \psi$ holds.

In case of a rules language, the set of formulas that can be used as premises is not the same as the formulas that can be used as consequents. Therefore, we should assume that $L_1 = Premises_1 \cup Consequents_1$ and $L_2 = Premises_2 \cup Consequents_2$. In addition, as in the case of DLP and Datalog, L_1 may not actually be a subset of L_2. Instead, it may be embedded in L_2 under a 1-1 transformation, ι. In our notation, this is expressed as $\iota(Premises_1) \subseteq Premises_2$ and $\iota(Consequents_1) \subseteq Consequents_2$.

We can now say that L_2 *extends* L_1 *under the embedding* ι if for every pair of formulas, $\phi \in Premises_1$ and $\psi \in Consequents_1$, the entailment $\phi \models_1 \psi$ holds iff $\iota(\phi) \models_2 \iota(\psi)$ holds.

With these definitions, we can now formally state (relying on the standard facts about Datalog and default negation) that Datalog extends DLP with respect to the DLP-to-Datalog embedding described in [GHVD03]. The default negation layer similarly extends Datalog with respect to the identity embedding.

Interoperability through language extension. With a proper definition of language extensions, we can now address a recent criticism of the layered structure of the rules stack. It is claimed in [HPPSH05] that it is incorrect to say that Datalog is an extension of the DLP layer because, given a single fact, such as `knows(pat,jo)`, DLP and Datalog give different answers to the question of whether `pat` knows exactly one person.

The answer to this apparent paradox (relatively to our earlier discussion) is simple: the above question cannot be formulated in either DLP or Datalog! In the OWL stack, this query requires a more expressive description logic and on the rules side it requires default negation. Therefore, *as stated*, the above argument falls flat on its face. However, a restatement of this argument is worth debating:

> *Given a set of RDF facts and two "similar" queries—one expressed in the rules stack and the other in the OWL stack—does it matter that the two queries might return different answers?*

The word *similar* is in quotes because it is unclear whether—outside of Datalog—an acceptable systematic mapping exists to map OWL queries into rules, or vice versa. For instance, the aforesaid question about `pat` knowing exactly one person requires radically different expressions in OWL and in the default negation layer. Nevertheless, *intuitively* these two queries can be viewed as similar. Under the OWL semantics the answer will be "unknown" since it is not possible to either prove or disprove that `pat` knows exactly one person; under the rules semantics the answer will be a "yes." We argue, however, that both answers are right! A user who chooses to write an application using the rules stack does so because of a desire to use the language and semantics of that stack. Otherwise, a competent user should choose OWL and SWRL.

5 Interoperability Between Rules and OWL

It has often been observed that DLP, the *intersection* of Description Logic and Logic Programming, is rather minimalistic—a good reference point perhaps, but too small for realistic knowledge representation in the Semantic Web. On the other hand, the *union* of various classes of Description Logic and Logic Programming leads to supersets of first-order logic with default negation, which is not easily formalized model-theoretically and computationally. To achieve a usable level of interoperability between the two paradigms of knowledge representation, we need a "logical framework" that will be sitting *above* the OWL and rules stack and will enable inferences performed by OWL to be used by the rules stack, and vice versa.

As discussed in previous sections, OWL-based ontologies and rules are best viewed as complementary stacks in a *hybrid* Semantic Web architecture. Our interoperability framework derives from these observations. When we say "rules" here, we mean rule bases with nonmonotonic semantics. Pure first-order rules, as in SWRL, belong to the OWL stack, and we will include them under the rubric of "OWL-based ontologies."

The basic idea is that rules and OWL will view each other as "black boxes" with well-defined interfaces defined through *exported predicates*. OWL-based ontologies will export some of their classes and properties, while rule-based knowledge bases will export some of the predicates that they define. Each type of the knowledge base will be able to refer to the predicates defined in the other knowledge bases and treat them *extensionally*, as collections of facts.

One of the earliest integration frameworks in this spirit was AL-log [DLNS98]. AL-Log is a uni-directional approach where rules can refer to description logic based ontologies, but not vice versa. This approach is appropriate when OWL-based ontologies are used to classify objects into classes (analogously to database schema), and rules supply additional inferences.

Bi-directional integration is more powerful, but the semantics of an integrated knowledge base may not be clear if rules refer to ontologies and ontologies refer back to rules within the same knowledge base in a recursive manner. One example when such a semantics can be defined under certain restrictions was given in

[ELST04]. However, we believe that recursive references across the rules/OWL boundary are unlikely, and this semantic complication will not arise (and can probably be disallowed in a practical language).

In sum, the hybrid architecture offers a way to combine expressive classes of nonmonotonic rulebases with OWL-based ontologies. The two kinds of knowledge bases can use inferences produced by each other or they can be used in a standalone mode. It is not hard to see that the interoperability framework discussed in this section can be implemented on top of the infrastructure provided by modules (or contexts) used in systems like FLORA-2 and TRIPLE, which was introduced in Section 3—the same infrastructure that can be used to solve the problem of scoped inference.

6 Conclusion

In this paper we provided an in-depth discussion of the multi-stack architecture (MSA) for the Semantic Web and argued that a stack of rule-based languages, complete with default negation, should exist side-by-side with the ontology stack based on OWL. We surveyed the theoretical underpinning of the rules stack and proposed a framework for interoperability between rules and ontologies. We also discussed the idea of scoped inference and highlighted its importance in the Web environment. We observed that both scoped inference and the interoperability framework can be implemented using the idea of modules.

We would like to further remark that the proposed multi-stack architecture is extensible and additional stacks can be added to it as long as they can interoperate according to the guidelines of Section 5. One candidate for such an additional stack is the SparQL language.[11]

Acknowledgement

Michael Kifer was supported in part by NSF grant CCR-0311512 and by U.S. Army Medical Research Institute under a subcontract through Brookhaven National Lab. Jos de Bruijn was supported by the European Commission under the projects SEKT and ASG. Harold Boley was supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC). Dieter Fensel was supported in part by the Science Foundation Ireland (SFI).

References

[ABdB$^+$05] Jürgen Angele, Harold Boley, Jos de Bruijn, Dieter Fensel, Pascal Hitzler, Michael Kifer, Reto Krummenacher, Holger Lausen, Axel Polleres, and Rudi Studer. Web rule language (wrl), Juny 2005. Technical Report.

[11] http://www.w3.org/TR/rdf-sparql-query/

[ABW88] K. R. Apt, H. A. Blair, and A. Walker. *Foundations of Deductive Databases and Logic Programming*, chapter Towards a theory of declarative knowledge, pages 89–148. Morgan Kaufmann Publishers, Los Altos, CA, 1988.

[BBB$^+$05] S. Battle, A. Bernstein, H. Boley, B. Grosof, M. Gruninger, R. Hull, M. Kifer, D. Martin, S. McIlraith, D. McGuinness, J. Su, and S. Tabet. Swsl-rules: A rule language for the semantic web, July 2005. Technical Report.

[BL04] T. Berners-Lee. Primer: Getting into RDF & Semantic Web using N3, 2004. http://www.w3.org/2000/10/swap/Primer.html.

[Bol96] Harold Boley. Knowledge Bases in the World Wide Web: A Challenge for Logic Programming. In Paul Tarau, Andrew Davison, Koen De Bosschere, and Manuel Hermenegildo, editors, *Proc. JICSLP'96 Post-Conference Workshop on Logic Programming Tools for INTERNET Applications*, pages 139–147. COMPULOG-NET, Bonn, Sept. 1996. Revised versions in: International Workshop "Intelligent Information Integration", KI-97, Freiburg, Sept. 1997; DFKI Technical Memo TM-96-02, Oct. 1997.

[BTW01] Harold Boley, Said Tabet, and Gerd Wagner. Design Rationale of RuleML: A Markup Language for Semantic Web Rules. In *Proc. Semantic Web Working Symposium (SWWS'01)*, pages 381–401. Stanford University, July/August 2001.

[CKW93] W. Chen, M. Kifer, and D.S. Warren. HiLog: A foundation for higher-order logic programming. 15(3):187–230, February 1993.

[Cla78] K. L. Clark. *Logic and Data Bases*, chapter Negation as Failure, pages 293–322. Plenum Press, NY, USA, 1978.

[DBSA98] Stefan Decker, Dan Brickley, Janne Saarela, and Jürgen Angele. A query and inference service for RDF. In *QL'98 - The Query Languages Workshop, W3C Workshop*, 1998.

[DLNS98] F.M. Donini, M. Lenzerini, D. Nardi, and A. Schaerf. AL-log: Integrating datalog and description logics. *Journal of Intelligent Information Systems*, 10(3):227–252, 1998.

[DS04] Mike Dean and Guus Schreiber, editors. *OWL Web Ontology Language Reference*. 2004. W3C Recommendation 10 February 2004.

[ELST04] Thomas Eiter, Thomas Lukasiewicz, Roman Schindlauer, and Hans Tompits. Combining answer set programming with description logics for the semantic web. In *Proc. of the International Conference of Knowledge Representation and Reasoning (KR04)*, 2004.

[FDES98] Dieter Fensel, Stefan Decker, Michael Erdmann, and Rudi Studer. Ontobroker: The very high idea. In *Proceedings of the 11th International FLAIRS Conference (FLAIRS-98)*, pages 131–135, Sanibel Island, Florida, USA, 1998.

[GHVD03] B.N. Grosof, I. Horrocks, R. Volz, and S. Decker. Description logic programs: Combining logic programs with description logic. In *12th International Conference on the World Wide Web (WWW-2003)*, May 2003.

[GL88] Michael Gelfond and Vladimir Lifschitz. The stable model semantics for logic programming. In Robert A. Kowalski and Kenneth Bowen, editors, *Proceedings of the Fifth International Conference on Logic Programming*, pages 1070–1080, Cambridge, Massachusetts, 1988. The MIT Press.

[GL91] M. Gelfond and V. Lifschitz. Classical negation in logic programs and disjunctive databases. *New Generation Computing*, 9(3/4):365–386, 1991.

[Gro99] B.N. Grosof. A courteous compiler from generalized courteous logic programs to ordinary logic programs. Technical Report RC 21472, IBM, July 1999.

[GRS91] Allen Van Gelder, Kenneth Ross, and John S. Schlipf. The well-founded semantics for general logic programs. *Journal of the ACM*, 38(3):620–650, 1991.

[Hay04] Patrick Hayes. RDF semantics. Technical report, W3C, 2004. W3C Recommendation 10 February 2004. http://www.w3.org/TR/rdf-mt/.

[HHL03] J. Heflin, J. Hendler, and S Luke. *SHOE: A Blueprint for the Semantic Web.* MIT Press, Cambridge, MA, 2003.

[HPPSH05] Ian Horrocks, Bijan Parsia, Peter Patel-Schneider, and James Hendler. Semantic web architecture: Stack or two towers? In *Third Workshop on Principles and Practice of Semantic Web Reasoning*, Dagstuhl, Germany, September 2005.

[HPSB$^+$04] Ian Horrocks, Peter F. Patel-Schneider, Harold Boley, Said Tabet, Benjamin Grosof, and Mike Dean. SWRL: A semantic web rule language combining OWL and RuleML. Member submission 21 may 2004, W3C, 2004. Available from http://www.w3.org/Submission/SWRL//.

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

[Kif05] M. Kifer. Nonmonotonic reasoning in FLORA-2. In *2005 Intl. Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'05)*, Diamante, Italy, September 2005.

[KLW95] M. Kifer, G. Lausen, and J. Wu. Logical foundations of object-oriented and frame-based languages. *Journal of the ACM*, 42:741–843, July 1995.

[Llo87] John W. Lloyd. *Foundations of Logic Programming.* Springer-Verlag, Berlin, Heidelberg, New York, 1987.

[LS99] Ora Lassila and Ralph R. Swick. Resource Description Framework (RDF) Model and Syntax Specification. Recommendation REC-rdf-syntax-19990222, W3C, February 1999.

[MS98] Massimo Marchiori and Janne Saarela. Query + metadata + logic = metalog. In *QL'98 - The Query Languages Workshop, W3C Workshop*, 1998.

[MW88] D. Maier and D.S. Warren. *Computing with Logic: Logic Programming with Prolog.* Benjamin-Cummings, Menlo Park, CA, 1988.

[Ont] Ontoprise, GmbH. Ontobroker. http://www.ontoprise.com/.

[SD02] Michael Sintek and Stefan Decker. TRIPLE – A Query, Inference, and Transformation Language for the Semantic Web. In *1st International Semantic Web Conference (ISWC2002).* Sardinia, Italy, June 2002.

[Sho87] Y. Shoham. Nonmonotonic logics: meaning and utility. In *Proc. 10th International Joint Conference on Artificial Intelligence*, pages 388–393. Morgan Kaufmann, 1987.

[YK02] G. Yang and M. Kifer. Well-founded optimism: Inheritance in frame-based knowledge bases. In *Intl. Conference on Ontologies, DataBases, and Applications of Semantics for Large Scale Information Systems (ODBASE)*, October 2002.

[YK03] Guizhen Yang and Michael Kifer. Reasoning about Anonymous Resources and Meta Statements on the Semantic Web. In Stefano Spaccapietra, Salvatore T. March, and Karl Aberer, editors, *J. Data Semantics I*, volume 2800 of *Lecture Notes in Computer Science*, pages 69–97. Springer, 2003.

[YKZ03] G. Yang, M. Kifer, and C. Zhao. FLORA-2: A rule-based knowledge representation and inference infrastructure for the Semantic Web. In *International Conference on Ontologies, Databases and Applications of Semantics (ODBASE-2003)*, November 2003. The system is available at http://flora.sourceforge.net.