



WSML Deliverable
D28.3 v0.1
WSML ONTOLOGY SEMANTICS

WSML Final Draft – December 26, 2006

Authors:

Jos de Bruijn
Stijn Heymans

Editors:

Jos de Bruijn

Reviewers:

Stephan Grimm
Davy van Nieuwenborgh

Final version:

<http://www.wsmo.org/TR/d28/d28.3/v0.1/20061218/>

Latest version:

<http://www.wsmo.org/TR/d28/d28.3/v0.1/>

Previous version:

<http://www.wsmo.org/TR/d28/d28.3/v0.1/20061129/>



Abstract

The current version (at the time of writing: 0.21) of the Web Service Modeling Language (WSML) describes five language variants: WSML-Core, WSML-DL, WSML-Flight, WSML-Rule, and WSML-Full. While the semantics of the first four variants was specified, with the Core and DL variants based on classical First-Order Logic, and the Flight and Rule variants based on Logic Programming with nonmonotonic negation and F-Logic, the semantics of WSML-Full was not specified, but seen as an open research issue. Additionally, layering between, on the one hand, languages which use the DL style of modeling (WSML-Core and WSML-DL), and on the other hand, languages which use the F-Logic style of ontology modeling (WSML-Flight and WSML-Rule), was not demonstrated.

We specify a semantic framework for all of WSML, based on the nonmonotonic modal logic *first-order autoepistemic logic*, extended with Frames and concrete domains. Each WSML variant is specified as a syntactic subset of this language. We thus provide a semantics for WSML-Full, which establishes language layering by definition.

We demonstrate that each WSML variant semantically corresponds to its target formalism, i.e. WSML-DL corresponds to the Description Logic $\mathcal{SHIQ}(\mathbf{D})$, WSML-Rule corresponds to the Stable Model Semantics for Logic Programs, a generalization of the Well-Founded Semantics, WSML-Flight corresponds to the Perfect Model Semantics for locally stratified Logic programs, and WSML-Core corresponds to $\mathcal{DHL}(\mathbf{D})$ (without nominals), the Horn subset of $\mathcal{SHIQ}(\mathbf{D})$.

In order to show the correspondence for WSML-DL, we first establish that, for a novel class of formulas called the \mathcal{E} -safe formulas, the DL style and the F-Logic style of modeling can be used interchangeably by showing that there is an entailment preserving translation to F-Logic for \mathcal{E} -safe theories. We also show that every \mathcal{SHIQ} theory is an \mathcal{E} -safe theory, which we use to establish correspondence between WSML-DL and $\mathcal{SHIQ}(\mathbf{D})$.



Contents

1	Introduction	4
2	Preliminaries	6
2.1	First-Order Logic and Description Logics	6
2.2	Frame Logic	9
3	A Translation from Predicate-based to Frame-based Languages	11
3.1	Translating to Sorted F-Logic	12
3.2	Translating Cardinal Formulas	12
3.3	Related Work and Conclusions	18
4	WSML Semantic Framework	20
4.1	F-Logic Programs	20
4.2	Frame Logic with Concrete Domains	21
4.3	First-Order Autoepistemic Logic with Frames and Concrete Domains	23
4.4	WSML Logical Expressions	25
4.5	Correspondence with Target Formalisms	28
4.5.1	WSML-Full	28
4.5.2	WSML-DL	29
4.5.3	WSML-Rule	30
4.5.4	WSML-Flight	31
4.5.5	WSML-Core	31
4.6	Mapping to the WSML Surface Syntax	32
5	Conclusions and Future Work	33



1 Introduction

The Web Service Modeling Language WSML¹ [12] is a language for modeling Ontologies and Web Services, based on a number of logical formalisms, which underly the WSML language variants. Figure 1.1(a) shows the different variants and the relationships between them. These variants differ in logical expressiveness and in the underlying language paradigms.

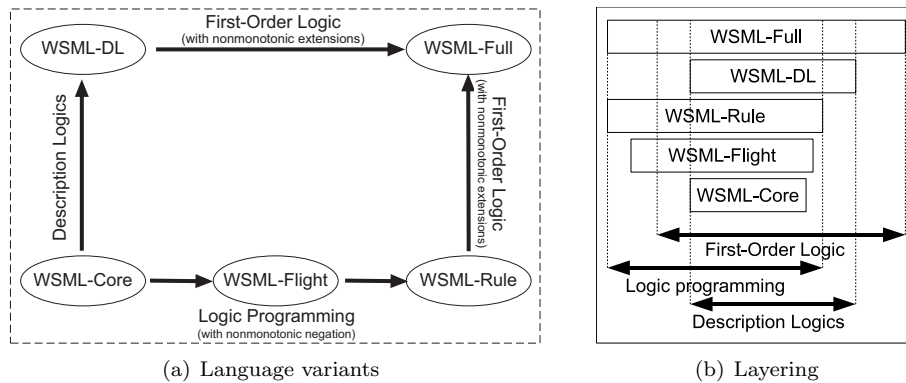


Figure 1.1: WSML Variants and Layering

WSML-Core is based on the intersection of the Description Logic $\mathcal{SHIQ}(\mathbf{D})$ and Horn Logic (without equality), called $\mathcal{DHL}(\mathbf{D})$, based on Description Logic Programs [19].

WSML-DL captures the Description Logic $\mathcal{SHIQ}(\mathbf{D})$.

WSML-Flight is based on the Datalog subset of F-Logic [24] programming, extended with inequality and (locally) stratified negation under the perfect model semantics [33].

WSML-Rule is based on F-Logic programming, extended with inequality and negation under the Well-Founded semantics [16].

WSML-Full unifies WSML-DL and WSML-Rule under a First-Order umbrella with nonmonotonic extensions. The semantics of WSML-Full was ongoing research at the time the WSML specification was published.

As shown in Figure 1.1(b), WSML has two alternative layerings, namely, $\text{WSML-Core} \Rightarrow \text{WSML-DL} \Rightarrow \text{WSML-Full}$ and $\text{WSML-Core} \Rightarrow \text{WSML-Flight} \Rightarrow \text{WSML-Rule} \Rightarrow \text{WSML-Full}$. For both layerings, WSML-Core and WSML-Full mark the least and most expressive layers, respectively. The two layerings are to a certain extent disjoint in the sense that inter-operation in WSML between the Description Logic variant (WSML-DL) on the one hand and the Logic Programming variants (WSML-Flight and WSML-Rule) on the other, is only

¹ <http://www.wsmo.org/wsml/wsml-syntax>



possible through a common core (WSML-Core) or through a very expressive, undecidable, superset (WSML-Full).

The original WSML specification [12] did not show any semantic properties of this layering, nor did it include the semantics for WSML-Full. In earlier work [11] we have shown that the WSML variants Core, Flight, Rule, and DL, as well as the first-order subset of Full, dubbed FOL, are properly layered, as suggested in the WSML specification. However, that work lacked a specification of the WSML-Full semantics, and lacked details about concrete domains as well as a few constructs, specific to WSML.

In this deliverable we take a slightly different approach. We specify a semantic framework for all WSML variants, based on first-order autoepistemic logic (FO-AEL) [25, 9], extended with Frames [24] and concrete domains [4]. We call this extended language FF-AEL. The individual WSML variants are defined as syntactical subsets of FF-AEL.

To prove correspondence between WSML-Rule and the Logic Programming semantics, we use the results on embedding non-ground Logic Programs in FO-AEL obtained in [9].

It turns out that the biggest challenge lies in demonstrating correspondence between WSML-Core and WSML-DL, on the one hand, and the Description Logics $\mathcal{DHL}(\mathbf{D})$ and $\mathcal{SHIQ}(\mathbf{D})$, on the other. WSML-Core and WSML-DL inherit the style of ontology modeling of F-Logic, in which classes and attributes are interpreted as elements of a domain, which are then associated to subsets of, and binary relations over, the domain via a second interpretation function, whereas in Description Logics such as $\mathcal{DHL}(\mathbf{D})$ and $\mathcal{SHIQ}(\mathbf{D})$, classes and attributes are interpreted directly as subsets of, and binary relations over, the domain of interpretation.

In Chapter 3 we establish that, following [11], these two styles of ontology modeling can be used interchangeably as long as the ontology falls in the novel class of \mathcal{E} -safe (equality-safe) formulas, which subsumes WSML-Core and WSML-DL. This establishes correspondence between WSML-Core and WSML-DL, on the one side, and the target formalisms, $\mathcal{DHL}(\mathbf{D})$ and $\mathcal{SHIQ}(\mathbf{D})$, on the other.

In Chapter 4 we specify the abstract syntax and semantics of WSML logical expressions, describe the restrictions on this syntax for each of the language variants, and show that these language variants correspond to the intended formalisms. Additionally, we describe a mapping between the abstract syntax and the surface syntax presented in the original specification [12].

Note that in this deliverable we only address the semantics of the ontology modeling elements in WSML, and not the semantics of the Web Service and Goal descriptions. The semantics of the functional and behavioral description of Web Services is a topic of ongoing investigation [21, 22, 35], and beyond the scope of this deliverable.



2 Preliminaries

2.1 First-Order Logic and Description Logics

The signature of a first-order language \mathcal{L} is of the form $\Sigma = \langle \mathcal{F}, \mathcal{P} \rangle$, where \mathcal{F} and \mathcal{P} are countable sets of function and predicate symbols and each function or predicate symbol has an associated arity n , which is a nonnegative integer. \mathcal{F} and \mathcal{P} are disjoint.

Given a signature Σ and a set of variable symbols \mathcal{V} , terms are either variables or constructed terms of the form $f(t_1, \dots, t_n)$ with $f \in \mathcal{F}$ an n -ary function symbol ($n \geq 0$) and t_1, \dots, t_n terms. Ground terms are also called *names*. Atomic formulas are expressions of the form $p(t_1, \dots, t_n)$ and $t_1 = t_2$, with $p \in \mathcal{P}$ an n -ary predicate symbol ($n \geq 0$) and t_1, \dots, t_n terms. Formulas of a first-order language \mathcal{L} are constructed as usual: every atomic formula is a formula in \mathcal{L} ; compound formulas are constructed using atomic formulas, the logical connectives $\neg, \wedge, \vee, \supset$, the quantifiers \exists, \forall , and the auxiliary symbols $(, ($.

An *interpretation* of a language \mathcal{L} is a tuple $w = \langle U, \cdot^I \rangle$, where U is a nonempty set (called *domain*) and \cdot^I is a mapping which assigns: a function $f^I : U^n \rightarrow U$ to every n -ary function symbol $f \in \mathcal{F}$, and a relation $p^I \subseteq U^n$, to every n -ary predicate symbol $p \in \mathcal{P}$. A variable assignment B is a mapping which assigns an element $x^B \in U$ to every variable symbol x . A variable assignment B' is an x -variant of B if $y^B = y^{B'}$ for every variable $y \in \mathcal{V}$ with $y \neq x$.

Given an interpretation $w = \langle U, \cdot^I \rangle$, a variable assignment B , and a term t , $t^{w,B}$ is defined as: $x^{w,B} = x^B$ for variable symbol x and $t^{w,B} = f^I(t_1^{w,B}, \dots, t_n^{w,B})$ for t of the form $f(t_1, \dots, t_n)$.

A *variable substitution* β is a set $\{x_1/t_1, \dots, x_k/t_k\}$, where x_1, \dots, x_k are distinct variables and t_1, \dots, t_k are names. β is *total* if it contains some x/n for every variable symbol $x \in \mathcal{V}$. Given a variable assignment B and a substitution β , if $\beta = \{x/t \mid x \in \mathcal{V}, t^w = x^B, \text{ for some name } t\}$, then β is *associated with* B . The *application* of a variable substitution β to some term, formula, or theory, denoted by appending β to it, is defined as syntactical replacement, as usual. Clearly, if the unique-names assumption applies, each variable assignment has a unique associated substitution; if the standard-names assumption applies, each associated substitution is total.

An interpretation w *satisfies* an atomic formula $p(t_1, \dots, t_n)$, given a variable assignment B , denoted $w, B \models p(t_1, \dots, t_n)$, if $(t_1^{w,B}, \dots, t_n^{w,B}) \in p^w$. $w, B \models t_1 = t_2$ iff $t_1^{w,B} = t_2^{w,B}$. This is extended to arbitrary formulas as usual: $w, B \models \phi_1 \wedge \phi_2$ (resp. $w, B \models \phi_1 \vee \phi_2$, $w, B \models \neg\phi_1$) iff $w, B \models \phi_1$ and $w, B \models \phi_2$ (resp. $w, B \models \phi_1$ or $w, B \models \phi_2$, $w, B \not\models \phi_1$); $w, B \models \forall x(\phi_1)$ (resp. $w, B \models \exists x(\phi_1)$) iff for every (resp. for some) B' which is an x -variant of B , $w, B' \models \phi_1$.

An interpretation w is a *model* of ϕ , denoted $w \models \phi$, if $w, B \models \phi$ for all variable assignments B ; ϕ is *satisfiable* if it has a model (unsatisfiable otherwise); ϕ is *valid* if every interpretation w is a model of ϕ . These definitions are straightforwardly extended to the case of first-order theories $\Phi \subseteq \mathcal{L}$.

A theory $\Phi \subseteq \mathcal{L}$ *entails* a formula $\phi \in \mathcal{L}$, denoted $\Phi \models \phi$, iff for all interpretations w in \mathcal{L} such that $w \models \Phi$, $w \models \phi$.



A predicate-based ontology language is a first-order language in which unary predicates represent classes of objects and binary predicates represent properties (relations between objects). Description Logics [3] are such predicate-based ontology languages.¹ Of special interest is $\mathcal{SHOIQ}(\mathbf{D})$, which is a slight generalization of the language underlying the Semantic Web ontology language OWL DL [32]. We present the syntax and semantics of $\mathcal{SHOIQ}(\mathbf{D})$ through a mapping to first-order logic with equality.

The signature of a $\mathcal{SHOIQ}(\mathbf{D})$ language consists of disjoint sets of concept (\mathcal{C}), datatype (\mathcal{D}), abstract role (\mathcal{R}_a), concrete role (\mathcal{R}_c), individual (\mathcal{F}_a), and data value identifiers (\mathcal{F}_c). $\mathcal{SHOIQ}(\mathbf{D})$ descriptions are of the following form, with A a concept identifier, D a datatype identifier, C, C' descriptions, R, R' role identifiers, S, S' abstract role identifiers, U, U' concrete role identifiers, a, b individual identifiers, o_1, \dots, o_n individual or data value identifiers, and n a non-negative integer.

$$C, C' \quad \longrightarrow \quad \top \mid \perp \mid A \mid C \sqcap C' \mid C \sqcup C' \mid \neg C \mid \{o_1, \dots, o_n\} \mid \exists S.C \mid \exists U.D \mid \forall S.C \mid \forall U.D \mid \geq nS.C \mid \geq nS.D \mid \leq nS.C \mid \leq nU.D$$

A $\mathcal{SHOIQ}(\mathbf{D})$ ontology consists of axioms of the following forms.

$$C \sqsubseteq C' \mid C \equiv C' \mid S \sqsubseteq S' \mid U \sqsubseteq U' \mid R \equiv R' \mid S \equiv S'^- \mid \text{Trans}(S) \mid a \in A \mid \langle a, o_1 \rangle \in R \mid a = b \mid a \neq b$$

Additionally, we have that in the *number restrictions* $\geq nS.C$ and $\leq nR.S$, R has to be *simple*, i.e., S and its sub-roles may not be transitive (with transitivity indicated by $\text{Trans}(S)$).

We present the semantics of $\mathcal{SHOIQ}(\mathbf{D})$ through a translation to first-order logic. Table 2.1 defines the mapping function π_y from $\mathcal{SHOIQ}(\mathbf{D})$ descriptions to first-order logic formulas with one free variable, X . Table 2.2 defines the translation from $\mathcal{SHOIQ}(\mathbf{D})$ axioms to closed first-order logic formulas, using the mapping function π_y .

Notice that in our presentation of FOL we did not distinguish between the abstract and concrete domains. We will come back to that later on in this deliverable.

The Description Logic $\mathcal{SHIQ}(\mathbf{D})$ corresponds to $\mathcal{SHOIQ}(\mathbf{D})$ without enumeration ($\{o_1, \dots, o_n\}$). \mathcal{SHOIQ} and \mathcal{SHIQ} refer to $\mathcal{SHOIQ}(\mathbf{D})$ and $\mathcal{SHIQ}(\mathbf{D})$ without concrete domains, respectively. In the remainder of the deliverable, when referring to $\mathcal{SHOIQ}(\mathbf{D})$ (resp. $\mathcal{SHIQ}(\mathbf{D})$) axioms, we refer to the first-order logic (FOL) version of these axioms.

Description Logic Programs (DLP) [19] can be seen as the expressive intersection of Description Logics and logic programming. The Description Logic \mathcal{DHL} is the Horn logic subset of the expressive Description Logic \mathcal{SHIQ} . With $\mathcal{DHL}\mathcal{O}$ we denote the extension of \mathcal{DHL} with nominals, while staying in the Horn fragment. This extension was described in [14]. A Description Logic Program (DLP) Π_O is obtained from a \mathcal{DHL} ontology O by rewriting the axioms in the ontology to Horn formulas and interpreting the formulas using the standard minimal Herbrand semantics (see e.g. [27]). By the standard results in Logic Programming, we know that O and Π_O agree on ground entailment.

¹We recognize that there are several nonmonotonic extensions of Description Logics. However, we restrict ourselves here to classical Description Logics.



DL syntax	FOL syntax
$\pi_y(A, X)$	$A(X)$
$\pi_y(\top, X)$	$X = X$
$\pi_y(\perp, X)$	$\neg(X = X)$
$\pi_y(C_1 \sqcap \dots \sqcap C_n, X)$	$\bigwedge \pi_y(C_i, X)$
$\pi_y(C_1 \sqcup \dots \sqcup C_n, X)$	$\bigvee \pi_y(C_i, X)$
$\pi_y(\neg C, X)$	$\neg \pi_y(C, X)$
$\pi_y(\{o_1 \dots o_n\}, X)$	$\bigvee X = o_i$
$\pi_y(\exists S.C, X)$	$\exists y(S(X, y) \wedge \pi_x(C, y))$
$\pi_y(\exists U.D, X)$	$\exists y(U(X, y) \wedge \pi_x(D, y))$
$\pi_y(\forall S.C, X)$	$\forall y(S(X, y) \supset \pi_x(C, y))$
$\pi_y(\forall U.D, X)$	$\forall y(U(X, y) \supset \pi_x(D, y))$
$\pi_y(\geq nS.C, X)$	$\exists y_1, \dots, y_n (\bigwedge S(X, y_i) \wedge \bigwedge \pi_x(C, y_i) \wedge \bigwedge \neg y_i = y_j)$
$\pi_y(\geq nU.D, X)$	$\exists y_1, \dots, y_n (\bigwedge U(X, y_i) \wedge \bigwedge \pi_x(D, y_i) \wedge \bigwedge \neg y_i = y_j)$
$\pi_y(\leq nS.C, X)$	$\forall y_1, \dots, y_{n+1} ((\bigwedge S(X, y_i) \wedge \bigwedge \pi_x(C, y_i)) \supset \bigvee y_i = y_j)$
$\pi_y(\leq nU.D, X)$	$\forall y_1, \dots, y_{n+1} ((\bigwedge U(X, y_i) \wedge \bigwedge \pi_x(D, y_i)) \supset \bigvee y_i = y_j)$
π_x is defined as π_y by substituting x and x_i for y and y_i , respectively	

Table 2.1: $\mathcal{SHOIQ}(\mathbf{D})$ Descriptions and their FOL correspondence

DL syntax	FOL syntax
<i>Class Axioms</i>	
$C \sqsubseteq C'$	$\forall x(\pi_y(C, x) \supset \pi_y(C', x))$
$C \equiv C'$	$\forall x(\pi_y(C, x) \supset \pi_y(C', x)) \wedge \forall x(\pi_y(C', x) \supset \pi_y(C, x))$
<i>Property Axioms</i>	
$R \sqsubseteq R'$	$\forall x, y(R(x, y) \supset R'(x, y))$
$S \equiv S'^{-}$	$\forall x, y(S(x, y) \supset S'(y, x)) \wedge \forall x, y(S'(y, x) \supset S(x, y))$
Trans (S)	$\forall x, y, z(S(x, y) \wedge S(y, z) \supset S(x, z))$
<i>Individual Axioms</i>	
$a \in C$	$\pi_y(C, a)$
$\langle a, o \rangle \in R$	$R(a, o)$
$a = b$	$a = b$
$a \neq b$	$\neg(a = b)$

Table 2.2: \mathcal{SHOIQ} Axioms and their FOL correspondence



We now describe the syntax of $\mathcal{DHLO}(\mathbf{D})$, which is the extension of \mathcal{DHLCO} with datatypes. The semantics is as in $\mathcal{SHOIQ}(\mathbf{D})$.

$\mathcal{DHLO}(\mathbf{D})$ descriptions are of the following form. C_L, C'_L (resp. C_R, C'_R) are descriptions which are allowed only on the left-hand (resp. right-hand) side of the inclusion symbol \sqsubseteq .

$$\begin{aligned} C, C' &\longrightarrow A \mid C \sqcap C' \mid \exists R. \{o_1\} \\ C_L, C'_L &\longrightarrow C \mid C_L \sqcup C'_L \mid \exists S. C_L \mid \exists U. D \geq 1R \mid \\ &\quad \{o_1, \dots, o_n\} \\ C_R, C'_R &\longrightarrow C \mid \forall S. C_R \mid \forall U. D \end{aligned}$$

A $\mathcal{DHLO}(\mathbf{D})$ ontology consists of axioms of the following forms.

$$\begin{aligned} C_L \sqsubseteq C'_R \mid C \equiv C' \mid S \sqsubseteq S' \mid U \sqsubseteq U' \mid R \equiv R' \mid S \equiv S'^- \mid \\ \text{Trans}(S) \mid \top \sqsubseteq \forall S^-. C_R \mid \top \sqsubseteq \forall S. C_R \mid \top \sqsubseteq \forall U. D \mid a \in A \mid \\ \langle a, o_1 \rangle \in R \end{aligned}$$

$\mathcal{DHL}(\mathbf{D})$ is $\mathcal{DHLO}(\mathbf{D})$ without nominals, \mathcal{DHLCO} is $\mathcal{DHLO}(\mathbf{D})$ without concrete domains, and \mathcal{DHL} is $\mathcal{DHL}(\mathbf{D})$ without concrete domains.

2.2 Frame Logic

Frame Logic [23, 24] (F-Logic) is an extension of first-order logic which adds explicit support for object-oriented modeling. It is possible to explicitly specify methods, as well as generalization/specialization and instantiation relationships. The syntax of F-Logic has some seemingly higher-order features, namely, the same identifier can be used for a class, an instance, and a method. However, the semantics of F-Logic is strictly first-order². To simplify matters, we do not consider parameterized methods, functional (single-valued) methods, inheritable methods, and compound molecules.

The signature of an F-language \mathcal{L}^F is of the form $\Sigma = \langle \mathcal{F}, \mathcal{P} \rangle$ with \mathcal{F} a set of function symbols and \mathcal{P} a set of predicate symbols, each with an associated arity $n \geq 0$. Let \mathcal{V} be a set of variable symbols. Terms and atomic formulas are constructed as in first-order logic: $x \in \mathcal{V}$ is a term and $f(t_1, \dots, t_n)$ is a term, with $f \in \mathcal{F}$ an n -ary function symbol and t_1, \dots, t_n terms.

A molecule in F-Logic is one of the following statements: (i) an *is-a* assertion of the form $C : D$, (ii) a *subclass-of* assertion of the form $C :: D$, or (iii) a data molecule of the form $C[D \rightarrow E]$, with C, D, E terms. An F-Logic molecule is *ground* if it does not contain variables.

Formulas of an F-language \mathcal{L}^F are either atomic formulas, molecules, or compound formulas which are constructed in the usual way from atomic formulas, molecules, and the logical connectives $\neg, \wedge, \vee, \supset$, the quantifiers \exists, \forall and the auxiliary symbols $), ($. We denote universal closure with (\forall) .

F-Logic Horn formulas are of the form $(\forall) B_1 \wedge \dots \wedge B_n \supset H$, with B_1, \dots, B_n, H atomic formulas or molecules. F-Logic Datalog formulas are F-Logic Horn for-

²Note that F-Logic is also often used as an extension of nonmonotonic logic programming. We treat such nonmonotonic F-Logic programs later on in this deliverable, in Section 4.1.



mulas without function symbols such that every variable in H occurs in some equality-free B_1, \dots, B_n .

An F -structure is a tuple $\mathbf{I} = \langle U, \prec_U, \in_U, \mathbf{I}_F, \mathbf{I}_P, \mathbf{I}_{\rightarrow} \rangle$. Here, \prec_U is an irreflexive partial order on the domain U and \in_U is a binary relation over U . We write $a \preceq_U b$ when $a \prec_U b$ or $a = b$, for $a, b \in U$. For each F-structure holds that if $a \in_U b$ and $b \preceq_U c$ then $a \in_U c$. Thus, if $b \preceq_U c$, then $\{k \mid k \in_U b, k \in U\} \subseteq \{k \mid k \in_U c, k \in U\}$, but not the other way around.

An n -ary function symbol $f \in F$ is interpreted as a function over the domain U : $\mathbf{I}_F(f) : U^n \rightarrow U$. An n -ary predicate symbol $p \in P$ is interpreted as a relation over the domain U : $\mathbf{I}_P(p) \subseteq U^n$. \mathbf{I}_{\rightarrow} associates a partial function³ $U \rightarrow \mathcal{P}(U)$ with each element of U : $\mathbf{I}_{\rightarrow} : U \rightarrow U \rightarrow \mathcal{P}(U)$. Variable assignments are as in first-order logic.

Given an interpretation \mathbf{I} , a variable assignment B , and a term t of \mathcal{L}^F , $t^{\mathbf{I}, B}$ is defined as: $x^{\mathbf{I}, B} = x^B$ for variable symbol x and $t^{\mathbf{I}, B} = \mathbf{I}_F(f)(t_1^{\mathbf{I}, B}, \dots, t_n^{\mathbf{I}, B})$ for t of the form $f(t_1, \dots, t_n)$.

F -satisfaction of ϕ in \mathbf{I} , given the variable assignment B , denoted $\mathbf{I}, B \models_f \phi$, is defined as:

- $\mathbf{I}, B \models_f p(t_1, \dots, t_n)$ iff $(t_1^{\mathbf{I}, B}, \dots, t_n^{\mathbf{I}, B}) \in \mathbf{I}_P(p)$,
- $\mathbf{I}, B \models_f t_1 : t_2$ iff $t_1^{\mathbf{I}, B} \in_U t_2^{\mathbf{I}, B}$,
- $\mathbf{I}, B \models_f t_1 :: t_2$ iff $t_1^{\mathbf{I}, B} \preceq_U t_2^{\mathbf{I}, B}$,
- $\mathbf{I}, B \models_f t_1 [t_2 \rightarrow t_3]$ iff $\mathbf{I}_{\rightarrow}(t_2^{\mathbf{I}, B})(t_1^{\mathbf{I}, B})$ is defined and $t_3^{\mathbf{I}, B} \in \mathbf{I}_{\rightarrow}(t_2^{\mathbf{I}, B})(t_1^{\mathbf{I}, B})$, and
- $\mathbf{I}, B \models_f t_1 = t_2$ iff $t_1^{\mathbf{I}, B} = t_2^{\mathbf{I}, B}$.

Extension to satisfaction of compound formulas is as in first-order logic.

The notions of a model and of validity are defined analogous to first-order logic. A theory $\Phi \subseteq \mathcal{L}^F$ F -entails a formula $\phi \in \mathcal{L}^F$, denoted $\Phi \models_f \phi$, iff for all F-structures \mathbf{I} such that $\mathbf{I} \models_f \Phi$, $\mathbf{I} \models_f \phi$.

Sorted F-Logic In predicate-based ontology languages, the sets of symbols used for concepts, roles and individuals are disjoint. This is not the case in F-Logic. This disjointness can be regained by using a *sorted* F-Logic language.

We consider a sorted F-Logic language with three sorts: individuals, concepts and roles. A sorted F-Logic language has a sorted signature $\Sigma = \langle \mathcal{F}_a, \mathcal{F}_c, \mathcal{F}_r, \mathcal{P} \rangle$, where \mathcal{F}_a is a set of function symbols, as before, \mathcal{F}_c is a set of concept symbols, \mathcal{F}_r is a set of role symbols, and \mathcal{P} is a set of predicate symbols, as before. $\mathcal{F}_a, \mathcal{F}_c, \mathcal{F}_r$, and \mathcal{P} are pairwise disjoint. The usual restrictions on the use of symbols in formulas applies, namely only molecules of the form $a : c, c :: d, a[r \rightarrow b]$ are allowed, with a, b terms constructed from symbols in $\mathcal{F}_a \cup \mathcal{V}$, $c, d \in \mathcal{F}_c \cup \mathcal{V}$, and $r \in \mathcal{F}_r \cup \mathcal{V}$. Quantifiers need to be qualified with a, c, r to indicate over which domain (individual, concept, role) the variable quantifies.

A sorted F-structure has three pairwise disjoint domains: U_a, U_c, U_r for the individuals, concepts, and roles, respectively; \prec_U is an irreflexive partial order over U_c ; \in_U is a relation between U_a and U_c : $\in_U : U_a \times U_c$. \mathbf{I}_F interprets symbols in \mathcal{F}_a as functions over U_a , symbols in \mathcal{F}_c as elements in U_c , and symbols of \mathcal{F}_r as elements in U_r . \mathbf{I}_P interprets symbols in \mathcal{P} as n -ary relations over U_a^n . Finally, \mathbf{I}_{\rightarrow} associates a partial mapping $U_a \rightarrow \mathcal{P}(U_a)$ to each element of U_r .

³ $\mathcal{P}(U)$ denotes the power-set of U .



Entity	Predicate style	Frame style
Class	$\delta(A(X))$	$X : A$
Property	$\delta(R(X, Y))$	$X [R \rightarrow Y]$
Equality	$\delta(X = Y)$	$X = Y$
n -ary predicate	$\delta(P(\vec{X}))$	$P(\vec{X})$
Universal	$\delta(\forall \vec{x}. C)$	$\forall \vec{x}(\delta(C))$
Existential	$\delta(\exists \vec{x}. C)$	$\exists \vec{x}(\delta(C))$
Conjunction	$\delta(C \wedge D)$	$(\delta(C) \wedge \delta(D))$
Disjunction	$\delta(C \vee D)$	$(\delta(C) \vee \delta(D))$
Implication	$\delta(C \supset D)$	$(\delta(C) \supset \delta(D))$
Negation	$\delta(\neg C)$	$\neg(\delta(C))$

Table 3.1: Translating predicate-based to frame-based modeling

3 A Translation from Predicate-based to Frame-based Languages

In this chapter, we define a straightforward translation from predicate-based ontologies to F-Logic. We show that when considering sorted F-Logic, the translation preserves entailment for arbitrary first-order theories. We then show that this is not the case in general when translating the ontology to an unsorted F-Logic language. However, for certain classes of first-order formulas, namely those where entailment can be reduced to validity of a *cardinal* formula [8], the translation preserves entailment. Our translation preserves function-freeness, i.e., if no function symbol of arity > 0 was used in the original ontology, no function symbol of arity > 0 will occur in the translated ontology. This is not the case for the translation of Description Logics to F-Logic presented in [5].

Table 3.1 defines the mapping δ from the predicate style of ontology modeling to the frame style. In the table, A, B are unary predicate symbols, C, D are formulas, R is a binary predicate symbol, P is an n -ary predicate symbol, with $n = 0$ or $n \geq 3$, x is a variable symbol, and X, Y are terms. The mapping δ extends to sets of formulas in the natural way.

Definition 1. *Given a first-order language \mathcal{L} with the signature $\Sigma = \langle \mathcal{F}, \mathcal{P} \rangle$. Let \mathcal{L}^F be the F-Logic language with the same signature Σ , then \mathcal{L}^F corresponds to \mathcal{L} . Given a first-order theory $\Phi \subseteq \mathcal{L}$, then $\delta(\Phi) \subseteq \mathcal{L}^F$ is the corresponding F-Logic theory.*

In the remainder of this section, we will first show that the translation in Definition 1 is faithful (i.e. preserves entailment) when considering a sorted F-Logic language. We will then show that for a certain class of formulas, the class of *cardinal* formulas (see [8]), the translation is also faithful when considering an unsorted language. Besides the classes of cardinal formulas identified in [8], we identify the novel class of \mathcal{E} -safe formulas, show that reasoning in *SHIQ* can be reduced to checking validity of \mathcal{E} -safe formulas, and show that \mathcal{E} -safe formulas are cardinal. Note that the class of \mathcal{E} -safe formulas goes beyond *SHIQ*, and can capture a wide class of description languages.



We use the results of this chapter in the next chapter for defining the framework for the WSML semantics.

3.1 Translating to Sorted F-Logic

We first investigate a translation to sorted F-Logic. We augment the translation in Table 3.1 to ensure that variables only quantify over the domain of individuals U_i , by replacing each universal quantifier \forall in Table 3.1 with \forall_a and each existential quantifier \exists with \exists_a . We denote the thus obtained translation function with δ^s .

We now show equi-satisfiability of formulas in \mathcal{L} , and their F-Logic counterparts. If \mathcal{L} is a predicate-based ontology language with signature $\Sigma = \langle \mathcal{F}, \mathcal{P} \rangle$, then the corresponding sorted F-Logic language \mathcal{L}^F is the sorted F-Logic language obtained from the signature Σ .

Lemma 1. *Let ϕ be a formula in \mathcal{L} and let \mathcal{L}^F be the corresponding sorted F-Logic language, then ϕ is satisfied in some interpretation of \mathcal{L} if and only if $\delta^s(\phi)$ is satisfied in some sorted F-structure of \mathcal{L}^F .*

Proof. (Sketch) From any interpretation w of \mathcal{L} such that $w \models \phi$ one can easily construct a corresponding sorted F-structure \mathbf{I} such that $\mathbf{I} \models_f \delta^s(\phi)$, and vice versa. \square

Using the lemma we can now show correspondence with respect to entailment.

Theorem 1. *Let Φ be a set of formulas in \mathcal{L} , $\delta^s(\Phi) \subseteq \mathcal{L}^F$ be the corresponding F-Logic theory, and let $\phi \in \mathcal{L}$ be an arbitrary formula, then*

$$\Phi \models \phi \text{ iff } \delta(\Phi) \models_f \delta(\phi).$$

Proof. Follows immediately from Lemma 1 and the fact that, in FOL, checking the entailment $\Phi \models \phi$ can be reduced to checking unsatisfiability of the formula $(\bigwedge \Phi) \wedge \neg\phi$. \square

3.2 Translating Cardinal Formulas

We now consider the translation function δ of Table 3.1 in its original form and we consider unsorted languages, and F-structures of the form $\mathbf{I} = \langle U, \preceq_U, \in_U, \mathbf{I}_F, \mathbf{I}_P, \mathbf{I}_{\mapsto} \rangle$.

It turns out that we lose the correspondence of models in the general case with this augmented definition, i.e. Lemma 1 no longer applies. Consider, for example, the formula

$$\phi = (\forall x, y(x = y)) \supset (q(a) \leftrightarrow r(a)). \quad (3.1)$$

The formula ϕ is trivially satisfied in any interpretation with more than one element in the domain, since the antecedent will be trivially false in such an



interpretation. If we consider an interpretation w with only one element k , then the antecedent is true, but the consequent is not necessarily true, because q and r may be interpreted differently (e.g. $q^w = \emptyset, r = \{k\}$). Thus, ϕ is not valid in FOL. Now consider the corresponding F-Logic formula

$$\delta(\phi) = (\forall x, y(x = y)) \supset (a : q \leftrightarrow a : r).$$

As we have seen, the original formula ϕ is not valid in \mathcal{L} . However, $\delta(\phi)$ is valid in \mathcal{L}^F , since q and r must be interpreted by \mathbf{I}_F as the same class in every F-structure which has exactly one element.

From the example we can see that the translation δ is not faithful (with respect to entailment) for arbitrary predicate-based ontology languages. There is, however, a class of theories for which the correspondence does hold. This is the class of theories for which entailment can be reduced to checking validity of a so-called *cardinal* formula [8].

Definition 2. *Let ϕ be a formula in \mathcal{L} and let γ denote the number of symbols in \mathcal{L} . An interpretation $w = \langle U, \cdot^I \rangle$ is cardinal if $|U| \geq \gamma$. ϕ is cardinal if the following holds:*

If ϕ is true in every cardinal interpretation of \mathcal{L} , then ϕ is true in every interpretation of \mathcal{L} .

Definition 2 extends naturally to sets of formulas.

Note that this condition does not hold for the formula ϕ in (3.1), because ϕ is true in every interpretation with a domain of at least 3 elements, but it is not true in every interpretation of \mathcal{L} . The following definition of cardinality is equivalent to Definition 2.

Proposition 1. *Let ϕ be a formula in \mathcal{L} , then ϕ is cardinal if and only if*

ϕ is true in a cardinal interpretation of \mathcal{L} , whenever ϕ is true in an interpretation of \mathcal{L} .

Proof. Follows immediately from the contraposition of Definition 2. □

We can now strengthen Lemma 1 and Theorem 1 to the case of unsorted F-Logic:

Lemma 2. *Let ϕ be a formula in \mathcal{L} . Then*

- *if $\delta(\phi)$ is satisfied in some F-structure of \mathcal{L}^F , then ϕ is satisfied in some interpretation of \mathcal{L} , and*
- *if ϕ is cardinal and is satisfied in some interpretation of \mathcal{L} , then $\delta(\phi)$ is satisfied in some F-structure of \mathcal{L}^F .*

Proof. Given a cardinal interpretation $w = \langle U, \cdot^I \rangle$ of \mathcal{L} . Since $|U|$ is greater than or equal to the number of symbols in \mathcal{L} , we may assume that for each $q \in \mathcal{P}$ there is a unique individual $k_q \in U$. $\mathbf{I} = (w)^{FL} = \langle U, \preceq_U, \in_U, \mathbf{I}_F, \mathbf{I}_P, \mathbf{I}_{\rightarrow} \rangle$ is the corresponding F-Logic structure, which is defined as follows: (i) $\forall f \in \mathcal{F}$: $\mathbf{I}_F(f) = f^w$, (ii) \forall unary $c \in \mathcal{P}$: $\mathbf{I}_F(c) = k_c$, (iii) \forall binary $r \in \mathcal{P}$: $\mathbf{I}_F(r) = k_r$, (iv) \forall unary $c \in \mathcal{P}$ and every individual $k \in U$, if $k \in c^w$ then $k \in_U \mathbf{I}_F(c)$, (v) \forall unary $c_1, c_2 \in \mathcal{P}$: $\mathbf{I}_F(c_1) \preceq_U \mathbf{I}_F(c_2)$ if $c_1^w \subseteq c_2^w$, (vi) \forall binary $r \in \mathcal{P}$ and



$\forall k_1, k_2 \in U$, if $\langle k_1, k_2 \rangle \in r^w$ then $k_2 \in \mathbf{I}_{\rightarrow}(\mathbf{I}_F(r))(k_1)$, and (vii) \forall non-unary and non-binary $p \in \mathcal{P}$: $\mathbf{I}_P(p) = p^w$.

Given an F-structure $\mathbf{I} = \langle U, \preceq_U, \in_U, \mathbf{I}_F, \mathbf{I}_P, \mathbf{I}_{\rightarrow} \rangle$ for the language \mathcal{L}^F , the corresponding FOL interpretation $w = (\mathbf{I})^{FOL} = \langle U, \cdot^w \rangle$ for \mathcal{L} is defined as follows: (i) $\forall f \in \mathcal{F}$: $f^w = \mathbf{I}_F(f)$, (ii) \forall unary $c \in \mathcal{P}$: $c^w = \{k \mid k \in_U \mathbf{I}_F(c), k \in U\}$, (iii) \forall binary $r \in \mathcal{P}$: $r^w = \{\langle k_1, k_2 \rangle \mid k_2 \in \mathbf{I}_{\rightarrow}(\mathbf{I}_F(r))(k_1), \text{ for } k_1, k_2 \in U\}$, and (iv) \forall non-unary and non-binary $p \in \mathcal{P}$: $p^w = \mathbf{I}_P(p)$.

We now proceed to prove the lemma:

(1) Assume $\mathbf{I} \models_{\mathfrak{f}} \delta(\phi)$ for some F-structure \mathbf{I} , then it is easy to verify that $w = (\mathbf{I})^{FOL}$ satisfies ϕ .

(2) Assume $w \models \phi$ for some interpretation w and cardinal formula ϕ . By Proposition 1, there is a cardinal interpretation w' which is a model of ϕ .

Assume $w', B \models \phi$ for some variable assignment B . Since w' is cardinal, $\mathbf{I} = (w')^{FL}$ is defined. To prove the lemma, it is sufficient to show that $\mathbf{I}, B \models_{\mathfrak{f}} \delta(\phi)$. We proceed by induction over the structure of the formula ϕ .

Consider $\phi = A(t)$. $w', B \models \phi$ iff $t^{w', B} \in A^{w'}$ iff $t^{\mathbf{I}, B} \in_U \mathbf{I}_F(A)$. The ‘only if’ direction follows from (v) in the translation above. The ‘if’ direction follows from the fact that $\mathbf{I}_F(C) \neq k$ for any $k = \mathbf{I}_F(D)$, with $D \neq C$ a concept identifier. Similar for formulas of the form $R(t_1, t_2)$.

Consider $\phi = (t_1 = t_2)$. $w', B \models \phi$ iff $t_1^{w', B} = t_2^{w', B}$ iff $t_1^{\mathbf{I}, B} = t_2^{\mathbf{I}, B}$. The last ‘iff’ follows trivially from the construction of \mathbf{I} .

Consider $\phi = \forall x(\psi)$. $w', B \models \phi$ iff for every x -variant B' of B , $w', B' \models \psi$ iff $\mathbf{I}, B' \models_{\mathfrak{f}} \delta(\psi)$. The last ‘iff’ follows by induction and from the observation that the domains of w' and \mathbf{I} are the same. Similar for $\phi = \exists x(\psi)$. This can be trivially extended to formulas of the forms $\neg\psi$, $\psi_1 \wedge \psi_2$, and $\psi_1 \vee \psi_2$. \square

Theorem 2. *Let $\Phi \subseteq \mathcal{L}$ be a set of formulas and $\phi \in \mathcal{L}$ be a formula,*

$$\text{if } \Phi \models \phi \text{ then } \delta(\Phi) \models_{\mathfrak{f}} \delta(\phi).$$

If $\neg(\bigwedge \Phi) \vee \phi$ is cardinal, then also

$$\Phi \models \phi \text{ iff } \delta(\Phi) \models_{\mathfrak{f}} \delta(\phi).$$

Proof. Follows from Lemma 2 and the observation that checking the entailment $\Phi \models \phi$ can be reduced to checking validity of the formula $\neg(\bigwedge \Phi) \vee \phi$. \square

Results on cardinal formulas from [8] can be applied directly to our case. From [8] we know that equality-free sentences, as well as the negation of Horn clauses with no equality in the antecedent are cardinal. This is, however, not sufficient for many ontology languages. Description Logics such as *SHIQ* allow explicit assertion of equality between individuals and the introduction of equality statements through maximal number restrictions (see Table 2.1).

We define the class of \mathcal{E} -safe formulas (\mathcal{E} stands for ‘equality’) which allow only *safe* uses of equality. With ‘safe’ we mean that the use of the equality does not restrict the size of the domains of the models. The structure of \mathcal{E} -safe formulas is similar to the structure of *guarded* formulas [1]. The major distinctions are the restrictions on the use of the equality symbol in \mathcal{E} -safe formulas and the fact that the guard in an \mathcal{E} -safe formula may be a conjunction



of atoms, whereas in the guarded fragment, the guard always consists of a single atom.

We first define the class of *limited* \mathcal{E} -safe ($l\mathcal{E}$ -safe) formulas, denoted $l\mathcal{ESF}$,

$$l\mathcal{ESF} ::= A \mid \neg A \mid \phi_1 \wedge \phi_2 \mid \phi_1 \vee \phi_2 \mid \\ \forall \vec{x}(\chi \supset \phi) \mid \exists \vec{x}(\chi \wedge \phi)$$

where A is an atomic formula either of the form $p(\vec{t})$ or $t_1 = t_2$ with t_1, t_2 either both ground or both non-ground terms; ϕ, ϕ_1, ϕ_2 are $l\mathcal{E}$ -safe formulas, and χ is either an atom of the form $p(\vec{t})$ or a conjunction of atoms of the form $p(\vec{t})$ such that the variable graph of the atoms with free variables in χ is connected.¹ Finally, every free variable in ϕ must appear in χ . We now define the class of \mathcal{E} -safe formulas, denoted \mathcal{ESF} ,

$$\mathcal{ESF} ::= \varphi \mid \forall x(\phi) \mid \exists x(\phi) \mid \psi_1 \wedge \psi_2 \mid \psi_1 \vee \psi_2$$

with ψ_1, ψ_2 \mathcal{E} -safe formulas, ϕ, φ $l\mathcal{E}$ -safe formulas, and x the only free variable in ϕ . As usual, an \mathcal{E} -safe sentence is an \mathcal{E} -safe formula without free variables.

We consider formulas of the forms $\forall x(x = x \supset \phi)$ and $\exists x(x = x \wedge \phi)$, with ϕ an $l\mathcal{E}$ -safe formula with one free variable x , \mathcal{E} -safe, because they are equivalent to $\forall x(\phi)$ and $\exists x(\phi)$, respectively. As is usual in guarded logics, we thus assume that formulas $\forall x(\phi)$, $\exists x(\phi)$ are guarded by $x = x$.

Notice that the negation of an \mathcal{E} -safe formula is equivalent to an \mathcal{E} -safe formula as well. This can be easily verified by bringing the negated formula into negation normal form (NNF).

Example 1. *The following formulas are \mathcal{E} -safe:*

$$\begin{aligned} &\forall x(p(x) \supset q(x)) \\ &\forall x(s(x, y) \supset p(x)) \\ &\exists x, y(p(x) \wedge r(x, y) \wedge x = y) \\ &\forall x(r(x)) \end{aligned}$$

The following formulas are not \mathcal{E} -safe:

$$\begin{aligned} &\forall x, y(x = y) \\ &\forall x, y(a(x) \wedge a(y) \supset x = y) \\ &\forall x, y(x = y \supset p(x, y)) \end{aligned}$$

Many expressive Description Logic languages are \mathcal{E} -safe, including \mathcal{SHIQ} .

Proposition 2. *Any (negation of a) \mathcal{SHIQ} axiom ϕ can be rewritten to an \mathcal{E} -safe formula ϕ' such that ϕ and ϕ' are equivalent, i.e., share the same models.*

Proof. Assume ϕ is the first-order version of a \mathcal{SHIQ} axiom (translation of \mathcal{SHIQ} axioms to FOL formulas can be done according to Table 2.2). In case ϕ is a property or individual axiom, it is trivially \mathcal{E} -safe and $\phi' = \phi$.

Say, ϕ is a class axiom of the form $\forall x(\phi_0 \supset \phi_1)$. Given the form of ϕ and the translation in Table 2.2, one can transform $\phi_0 \supset \phi_1$ to a conjunction ψ of $l\mathcal{E}$ -safe formulas, e.g., removing disjunction from the antecedent induces a splitting of the original formula in a conjunction of formulas, such that $\phi' \equiv \forall x(\psi)$ is an \mathcal{E} -safe formula which is equivalent to ϕ .

As the negation of an \mathcal{E} -safe formula is equivalent to an \mathcal{E} -safe formula we have that the negation of a \mathcal{SHIQ} axiom is \mathcal{E} -safe as well. \square

¹The variable graph of a set of atoms is an undirected graph where nodes correspond to atoms and two nodes are connected through an edge if the corresponding atoms share a variable.



Note that *SHOIQ* formulas are not \mathcal{E} -safe in general, because of the possibility of using nominals. Consider, for example, the *SHOIQ* knowledge base $\{\top \sqsubseteq \{a\}\}$. This is equivalent to the first-order sentence $\forall x(x = a)$, which is not \mathcal{E} -safe. Every model of this knowledge base has exactly one element in its domain. This generalizes to most Description Logics with unrestricted use of nominals.

The class of \mathcal{E} -safe formulas is highly expressive. In fact, it is easy to see, with a slight modification of Proposition 2, that *SHIQ* knowledge bases extended with certain kinds of Horn formulas (those which can be translated to \mathcal{E} -safe formulas) can be equivalently translated to sets of \mathcal{E} -safe formulas. As entailment in this class of formulas is undecidable in general [26, 20]², entailment of \mathcal{E} -safe formulas is undecidable in general as well.

We now formulate our main result with respect to cardinal formulas.

Lemma 3. *The following classes of first-order formulas are cardinal.*

1. Sets of equality-free sentences,
2. formulas of the form $\neg S$, where S is a conjunction of Horn clauses without equality in the head, and
3. the class of \mathcal{E} -safe sentences.

Proof. Cardinality of the first and second class is shown in [8]. We proceed with the proof of cardinality of \mathcal{E} -safe formulas.

There are five types of \mathcal{E} -safe sentences: (1) *lESF* sentences, (2) universal and (3) existential \mathcal{E} -safe sentences, and (4) conjunctions and (5) disjunctions of \mathcal{E} -safe sentences. Any *lESF* sentence ϕ can be equivalently written as a universal sentence $\forall x(\phi)$. We now proceed to prove cardinality of sentences of the forms (2,3,4,5).

We need the following auxiliary notion. Given an interpretation $w = \langle U, \cdot^w \rangle$, $k \in U$ is *unused* in w if: (a) k does not occur in the domain or the range of a function $f^w : U^n \rightarrow U$ for $f \in \mathcal{F}_a$, and (b) k does not occur in a relation $p^w \subseteq U^n$ for $p \in \mathcal{P}$.

(2) We proceed by induction. Assume $w^{\geq \gamma} \models \forall x(\phi)$ for every cardinal interpretation $w^{\geq \gamma}$. We will show that if $w^{i+1} \models \forall x(\phi)$ for every interpretation w^{i+1} of cardinality $i+1$, then $w^i \models \phi$ for every interpretation w^i of cardinality i , with $i \geq 1$. By induction, this guarantees that every interpretation is a model of $\forall x(\phi)$, and thus the formula is cardinal. Let w^i be an interpretation of cardinality i , and let w^{i+1} be the interpretation obtained from w^i by adding one unused individual to the domain. By the induction hypothesis, $w^{i+1} \models \forall x(\phi)$. Thus, for every variable assignment B of w^{i+1} , $w^{i+1}, B \models \phi$. Since the domain of w^i is a subset of the domain of w^{i+1} , every variable assignment of w^i is a variable assignment of w^{i+1} . Thus, for every variable assignment B' of w^i , $w^{i+1}, B' \models \phi$. We now show by induction over the length of the formula ϕ that if $w^{i+1}, B' \models \phi$, then $w^i, B' \models \phi$.

If $w^{i+1}, B' \models (t_1 = t_2)$, then $t_1^{w^{i+1}, B'} = t_2^{w^{i+1}, B'}$; clearly, $t_1^{w^i, B'} = t_1^{w^{i+1}, B'}$ and $t_2^{w^i, B'} = t_2^{w^{i+1}, B'}$, and thus $t_1^{w^i, B'} = t_2^{w^i, B'}$ and $w^i, B' \models (t_1 = t_2)$.

²The proofs of undecidability of combinations of Description Logics (simpler than *SHIQ*) with Horn formulas in [26, 20] rely only on Horn formulas which are \mathcal{E} -safe.



If $w^{i+1}, B' \models p(t_1, \dots, t_n)$, then $(t_1^{w^{i+1}, B'}, \dots, t_n^{w^{i+1}, B'}) \in p^{w^{i+1}}$, and also in p^{w^i} , and thus $w^i, B' \models \phi$.

If $w^{i+1}, B' \models \neg(t_1 = t_2)$ then $t_1^{w^{i+1}, B'} \neq t_2^{w^{i+1}, B'}$, and by the same argument as above, $w^i, B' \models \neg(t_1 = t_2)$. Similar for $w^{i+1}, B' \models \neg p(t_1, \dots, t_n)$.

If $w^{i+1}, B' \models \psi_1 \wedge \psi_2$, $w^i, B' \models \psi_1$ and $w^i, B' \models \psi_2$, then, clearly, $w^i, B' \models \psi_1 \wedge \psi_2$. Similar for $\psi_1 \vee \psi_2$.

If $w^{i+1}, B' \models \exists \vec{x}(\chi \wedge \phi)$, then there is an \vec{x} -variant B'' of B' such that $w^{i+1}, B'' \models \chi \wedge \phi$. Assume B'' assigns a free variable in χ to an unused individual in w^{i+1} , then, clearly, $w^{i+1}, B'' \not\models \chi$. Therefore, we may assume that B'' is an \vec{x} -variant of B' which does not assign any variable to an unused individual, and $w^{i+1}, B'' \models \chi \wedge \phi$. By induction we have, $w^i, B'' \models \chi$ and $w^i, B'' \models \phi$, and thus $w^i, B' \models \exists \vec{x}(\chi \wedge \phi)$.

If $w^{i+1}, B' \models \forall \vec{x}(\chi \supset \phi)$, then $w^{i+1}, B'' \models \chi \supset \phi$ for every \vec{x} -variant B'' of B' of w^i (by the same argument as the outer induction). Clearly, if $w^{i+1}, B'' \not\models \chi$, then $w^i, B'' \not\models \chi$, since χ is a conjunction of atomic formulas. By induction we have that if $w^{i+1}, B'' \models \phi$, then $w^i, B'' \models \phi$, and thus $w^{i+1}, B' \models \forall \vec{x}(\chi \supset \phi)$.

(3) If $w \models \exists x(\phi)$, then there is a variable assignment B such that $w, B \models \phi$. Let \mathcal{I}^c be a cardinal interpretation obtained from w by adding a sufficient number of unused individuals to the domain. It is easy to verify using induction over the length of the formula, similar to the induction in (2), that if $w, B \models \phi$, then $\mathcal{I}^c, B \models \phi$ for ϕ an $l\mathcal{ESF}$ formula (note that B is a variable assignment of \mathcal{I}^c , because the domain of \mathcal{I}^c is a superset of that of w). Thus, by Proposition 1, $\exists x(\phi)$ is cardinal.

(4) Assume ψ_1, ψ_2 are cardinal. Now, if every cardinal interpretation w is a model of $\psi_1 \wedge \psi_2$, then every cardinal interpretation is a model of ψ_1 and ψ_2 , and, by cardinality of ψ_1, ψ_2 , every interpretation is a model of ψ_1 and ψ_2 . Therefore, every interpretation is a model of $\psi_1 \wedge \psi_2$ and thus $\psi_1 \wedge \psi_2$ is cardinal.

(5) Assume ψ_1, ψ_2 are cardinal. If $w \models \psi_1 \vee \psi_2$ then $w \models \psi_1$ or $w \models \psi_2$. Say $w \models \psi_1$, then, by cardinality of ψ_1 and Proposition 1, there is a cardinal interpretation w' such that $w' \models \psi_1$; similar for ψ_2 . Thus, there is a cardinal interpretation w' such that $w' \models \psi_1 \vee \psi_2$ and thus $\psi_1 \vee \psi_2$ is cardinal. \square

The following corollary follows immediately from Theorem 2, Proposition 2 and Lemma 3:

Corollary 1. *Let Φ be a set of (FOL) SHIQ axioms and ϕ a (FOL) SHIQ axiom, then*

$$\Phi \models \phi \quad \text{iff} \quad \delta(\Phi) \models_{\mathfrak{r}} \delta(\phi).$$

We conclude this section with the observation that the results of Lemma 3 immediately apply to HiLog, since our definition of cardinality coincides with the definition of cardinality in [8]. The following Corollary follows from Lemma 3 and the results in [8].

Corollary 2. *Let ϕ be an \mathcal{E} -safe sentence, then ϕ is valid in HiLog if and only if ϕ is valid in first-order logic. And, if Φ is a set of \mathcal{E} -safe sentences, then Φ HiLog-entails ϕ if and only if $\Phi \models \phi$.*



There are several proposals for layering F-Logic programming on top of \mathcal{DHL} (e.g. [23, 13, 2, 7]). The following proposition shows that this layering is justified, in the sense that it preserves entailment.

Proposition 3. *Let O be a \mathcal{DHL} ontology and let $\pi(O)$ be the FOL equivalent, with π as defined in Table 2.2, then, for the F-Logic theory $\delta(\pi(O))$, with δ as in Table 3.1,*

$$O \models \alpha \quad \text{iff} \quad \delta(\pi(O)) \models_{\mathcal{F}} \delta(\alpha),$$

with α an equality-free ground atomic formula.

Proof. Equivalence (with respect to entailment, modulo the transformation δ) between $\pi(O)$ and $\delta(\pi(O))$ follows from Theorem 2, Lemma 3 and the fact that $\pi(O)$ is equivalent to a set of Horn formulas without equality in the head. \square

3.3 Related Work and Conclusions

Balaban [5] proposes to use F-Logic as an underlying framework for Description Logics and uses the flexibility of F-Logic to extend Description Logics. DFL [6] uses F-Logic to reason about ontologies and rules. The major differences between the approach of Balaban and our approach are: (a) we do not need function symbols if the original language does not use function symbols; (b) we allow arbitrary predicate-based ontology languages, whereas Balaban's translation is restricted to Description Logics; and (c) Balaban uses a sorted F-Logic, whereas we do not need sorts for a large class of formulas.

F-OWL [37] uses FLORA [36], an F-Logic programming implementation, to reason over OWL. The authors capture the semantics of OWL using entailment rules over RDF triples. It is not clear exactly which part of the semantics of OWL is captured in F-OWL.

Two proposals for extending OWL DL with meta-modeling support are presented in [28]. The proposals are based on the contextual predicate calculus and HiLog [8]. It was not discussed in [28] whether HiLog- \mathcal{SHOIQ} is a proper extension of \mathcal{SHOIQ} in the sense that a \mathcal{SHOIQ} knowledge base Φ entails an axiom ϕ if and only if Φ HiLog-entails ϕ . We conjecture that by Corollary 2 and the fact that the semantics of HiLog- \mathcal{SHOIQ} is very close to HiLog, HiLog- \mathcal{SHIQ} is a proper extension of \mathcal{SHIQ} , but HiLog- \mathcal{SHOIQ} is not a proper extension of \mathcal{SHOIQ} : it might be the case that Φ HiLog-entails ϕ , but $\Phi \not\models \phi$.

In predicate-based ontology representation languages (e.g. Description Logics), classes are modeled as unary predicates and properties as binary predicates, which are interpreted as sets and as binary relations, respectively. In F-Logic, classes and properties are both first interpreted as objects and then related to sets and relations, respectively.

In this chapter we have introduced a translation from predicate-based ontologies to ontologies in F-Logic. We have shown that this translation preserves entailment for large classes of predicate-based ontology languages, including the class of *cardinal* formulas. Intuitively, cardinal formulas do not restrict the size of the domains of the models. We have defined the class of \mathcal{E} -safe formulas



and shown that \mathcal{E} -safe formulas are cardinal. Finally, we have shown that the class of \mathcal{E} -safe formulas is a very expressive class of formulas which includes the Description Logic *SHIQ*.

We use the results obtained in this chapter in the next chapter to show correspondence between WSML variants and target formalisms. Additionally, the results can be used for, for example, F-Logic based reasoning with, and extension of, classes of predicate-based ontology languages. Another application of the results is the use of F-Logic as a vehicle for the extension of RDF, similar to the first-order extensions of RDF described in [10]. This encoding of RDF(S) in F-Logic is future work.



4 WSML Semantic Framework

In this chapter, we specify the WSML language variants and the WSML abstract syntax for logical expressions, together with its semantics. The language variant prescribes the allowed part of the overall logical expression syntax.

In the remainder of this chapter we first define F-Logic Programs, F-Logic with concrete domains and first-order autoepistemic logic with frames and concrete domains, dubbed FF-AEL, in the Sections 4.1, 4.2 and 4.3. We then proceed to define the WSML variants 4.4, and prove their correspondences with the intended target formalisms, in Section 4.5. Finally, we present a mapping between the abstract syntax used in this chapter and the surface syntax specified in the WSML language reference [12], in Section 4.6.

4.1 F-Logic Programs

A *normal F-Logic program* P consists of rules of the form

$$h \leftarrow b_1, \dots, b_m, \text{not } c_1, \dots, \text{not } c_n, \quad (4.1)$$

where $h, b_1, \dots, b_m, c_1, \dots, c_n$ are (equality-free) atoms or molecules. h is the *head atom* of r , $B^+(r) = \{b_1, \dots, b_m\}$ is the set of *positive body atoms* of r , and $B^-(r) = \{c_1, \dots, c_n\}$ is the set of *negative body atoms* of r . If $B^-(r) = \emptyset$, then r is *positive*. If every variable in r occurs in $B^+(r)$, then r is *safe*. If every rule $r \in P$ is positive (safe, respectively), then P is positive (safe, respectively). Additionally, every F-Logic program contains the following rules, which axiomatize the semantics of molecules:

$$x :: z \leftarrow x :: y, y :: z \quad (4.2)$$

$$x : z \leftarrow x : y, y :: z \quad (4.3)$$

$$x :: x \quad (4.4)$$

The first rule (4.2) axiomatizes transitivity of the subclass relation; the second rule (4.3) axiomatizes inheritance of class membership; the third rule (4.4) axiomatizes the fact that every class is a subclass of itself.¹

The F-Logic signature Σ_P is a superset of the function and predicate symbols which occur in P . Let \mathcal{L}_P^F denote the F-Logic language based on Σ_P . We assume that Σ_P contains at least one 0-ary function symbol or only 0-ary predicate

¹Note that this third rule is not safe, and thus may not be suitable for efficient implementation. However, the identity subclass relation is trivial and therefore often not required for query answering; alternatively, one could approximate the axiomatization as follows:

$$x :: x \leftarrow x :: y$$

$$x :: x \leftarrow y :: x$$

$$x :: x \leftarrow y : x$$

so the identity subclass relation is only returned in queries in case the term x is used as a class.



symbols. The *Herbrand base*, B_H , of \mathcal{L}_P^F is the set of ground atomic formulas and molecules of \mathcal{L}_P^F . Subsets of B_H are called *Herbrand interpretations*.

The *grounding* of a logic program P , denoted $gr(P)$, is the union of all possible ground instantiations of P , obtained by replacing each variable in a rule r with a ground term in Σ_P , for each rule $r \in P$.

Let P be a positive program. A Herbrand interpretation M of P is a *model* of P if, for every rule $r \in gr(P)$, $B^+(r) \subseteq M$ implies $H(r) \cap M \neq \emptyset$. A Herbrand model M is *minimal* iff for every model M' such that $M' \subseteq M$, $M' = M$.

Following [18], the *reduct* of a logic program P with respect to an interpretation M , denoted P^M , is obtained from $gr(P)$ by deleting (i) each rule r with $B^-(r) \cap M \neq \emptyset$, and (ii) *not c* from the body of every remaining rule r with $c \in B^-(r)$. If M is a minimal Herbrand model of P^M , then M is a *stable model* of P .

If P is a positive logic program, then the corresponding Horn F-Logic theory Φ is obtained by, for every rule, replacing the arrow \leftarrow with material implication \supset , and replacing every comma $(,)$ in the body of the rule with \wedge .

Theorem 3. *Let P be a positive logic program and Φ be the corresponding Horn F-Logic theory, then P has one stable model M and for every ground atom or molecule α , $\alpha \in M$ iff $\Phi \models \alpha$.*

Proof. Let Φ be an F-Logic theory and let Φ' be the FOL theory obtained from Φ by replacing every molecule of the form $t_1 : t_2$ with an atomic formula of the form $_isa(t_1, t_2)$, molecules of the form $t_1 :: t_2$ with atomic formulas of the form $_subclass(t_1, t_2)$, and molecules of the form $t_1[t_2 \rightarrow t_3]$ with atomic formulas of the form $_att(t_1, t_2, t_3)$, with t_1, t_2, t_3 terms and $_isa, _subclass, _att$ predicate symbols which do not occur in Φ , and adding the following formulas:

$$\begin{aligned} \forall x, y, z \quad & (_subclass(x, y) \wedge _subclass(y, z) \supset _subclass(x, z)), \\ \forall x, y, z \quad & (_isa(x, y) \wedge _subclass(y, z) \supset _isa(x, z)), \text{ and} \\ \forall x \quad & (_subclass(x, x)). \end{aligned}$$

It is easy to verify that the F-Logic models of Φ and the FOL models of Φ' are isomorphic. The Theorem follows immediately from the correspondence between minimal Herbrand models and first-order entailment, and the correspondence between minimal Herbrand models and stable models for positive programs. \square

4.2 Frame Logic with Concrete Domains

In this section, we define F-Logic with concrete domains. A signature is of the form $\Sigma = \langle \mathcal{F}, \mathcal{P}, \mathcal{F}^D, \mathcal{P}^D \rangle$, with \mathcal{F} and \mathcal{P} as before, \mathcal{F}^D is a set of concrete function symbols, and \mathcal{P}^D is a set of concrete predicate symbols. The difference between abstract and concrete symbols, is that the interpretation of the latter is fixed, and the same in every model, whereas the interpretation of the former varies. In the remainder we assume that the signature of each language contains a set of concrete predicate symbols which are used for identifying particular datatypes, such as $_int, _string$, which are used to identify the domains of integers and strings, respectively. Abstract terms are constructed using symbols



from \mathcal{F} and \mathcal{V} as usual. Concrete terms are constructed using symbols from \mathcal{F}^D and \mathcal{V} . Terms are either abstract terms, concrete terms, or variables. Atomic formulas and molecules are constructed as usual. Concrete atomic formulas are constructed from concrete terms and symbols in \mathcal{P}^D . Formulas are constructed as usual, with the difference that abstracts quantifiers are indexed with a and concrete quantifiers are indexed with c ($\exists_a, \forall_a, \exists_c, \forall_c$). Finally, variables quantified using an abstract quantifier (\exists_a, \forall_a) may not occur in a concrete term or an atom which has a concrete predicate symbol.

An *F-structure with concrete domains* is a tuple $\mathbf{I} = \langle U, U^D, \prec_U, \in_U, \mathbf{I}_F, \mathbf{I}_P, \mathbf{I}_{\rightarrow} \rangle$. Here, \prec_U is an irreflexive partial order on the domain U and \in_U is a binary relation over U . We write $a \preceq_U b$ when $a \prec_U b$ or $a = b$, for $a, b \in U$. For each F-structure holds that if $a \in_U b$ and $b \preceq_U c$ then $a \in_U c$. Thus, if $b \preceq_U c$, then $\{k \mid k \in_U b, k \in U\} \subseteq \{k \mid k \in_U c, k \in U\}$. U^D is the concrete domain.

An n -ary function symbol $f \in \mathcal{F}$ is interpreted as a function over the domain U : $\mathbf{I}_F(f) : U^n \rightarrow U$. An n -ary concrete function symbol $f \in \mathcal{F}^D$ is interpreted as a function over the domain U^D : $\mathbf{I}_F(f) : (U^D)^n \rightarrow U^D$. An n -ary predicate symbol $p \in \mathcal{P}$ is interpreted as a relation over the domain $U \cup U^D$: $\mathbf{I}_P(p) \subseteq (U \cup U^D)^n$. An n -ary concrete predicate symbol $p \in \mathcal{P}^D$ is interpreted as a relation over the domain U^D : $\mathbf{I}_P(p) \subseteq (U^D)^n$. \mathbf{I}_{\rightarrow} associates a partial function $U \cup U^D \rightarrow \mathcal{P}(U \cup U^D)$ with each element of U : $\mathbf{I}_{\rightarrow} : U \rightarrow U \cup U^D \rightarrow \mathcal{P}(U \cup U^D)$.

A variable assignment B assigns each variable x to an individual $x^B \in U \cup U^D$. A variable assignment B_a is an abstract x -variant of B if B_a is an x -variant of B and $x^{B_a} \in U$. A variable assignment B_c is a concrete x -variant of B if B_c is an x -variant of B and $x^{B_c} \in U^D$. The interpretation of a term t in some \mathbf{I} with respect to some variable assignment B , written $t^{\mathbf{I}, B}$, is defined as usual.

F-satisfaction of ϕ in \mathbf{I} , given the variable assignment B , denoted $\mathbf{I}, B \models_f \phi$, is defined as:

- $\mathbf{I}, B \models_f p(t_1, \dots, t_n)$ iff $(t_1^{\mathbf{I}, B}, \dots, t_n^{\mathbf{I}, B}) \in \mathbf{I}_P(p)$,
- $\mathbf{I}, B \models_f t_1 : t_2$ iff $t_1^{\mathbf{I}, B} \in_U t_2^{\mathbf{I}, B}$,
- $\mathbf{I}, B \models_f t_1 :: t_2$ iff $t_1^{\mathbf{I}, B} \preceq_U t_2^{\mathbf{I}, B}$,
- $\mathbf{I}, B \models_f t_1 [t_2 \rightarrow t_3]$ iff $\mathbf{I}_{\rightarrow}(t_2^{\mathbf{I}, B})(t_1^{\mathbf{I}, B})$ is defined and $t_3^{\mathbf{I}, B} \in \mathbf{I}_{\rightarrow}(t_2^{\mathbf{I}, B})(t_1^{\mathbf{I}, B})$, and
- $\mathbf{I}, B \models_f t_1 = t_2$ iff $t_1^{\mathbf{I}, B} = t_2^{\mathbf{I}, B}$.

This is extended to arbitrary formulas as follows: $w, B \models \phi_1 \wedge \phi_2$ (resp. $w, B \models \phi_1 \vee \phi_2$, $w, B \models \neg \phi_1$) iff $w, B \models \phi_1$ and $w, B \models \phi_2$ (resp. $w, B \models \phi_1$ or $w, B \models \phi_2$, $w, B \not\models \phi_1$); $w, B \models \forall_a x(\phi_1)$ (resp. $w, B \models \exists_a x(\phi_1)$) iff for every (resp. for some) B'_a which is an abstract x -variant of B , $w, B'_a \models \phi_1$; $w, B \models \forall_c x(\phi_1)$ (resp. $w, B \models \exists_c x(\phi_1)$) iff for every (resp. for some) B'_c which is a concrete x -variant of B , $w, B'_c \models \phi_1$.



4.3 First-Order Autoepistemic Logic with Frames and Concrete Domains

First-Order Autoepistemic Logic (FO-AEL) [25, 9] is an extension of first-order logic with a modal operator. This modal operator is sometimes called the knowledge or belief operator, and is interpreted nonmonotonically. We specify a version of FO-AEL with extensions based on F-Logic, as well as concrete domains, which are required for treating datatypes in a special way. In our treatment of FO-AEL we follow [9]. We refer to the F-Logic extension of FO-AEL, with concrete domains, as FF-AEL.

An FF-AEL language \mathcal{L}_L^F is defined relative to an F-Logic language with concrete domains \mathcal{L}^F :

- any atomic formula in \mathcal{L}^F is a formula in \mathcal{L}_L^F ,
- if ϕ is a formula in \mathcal{L}_L^F , then $L\phi$, called a *modal atom*, is a formula in \mathcal{L}_L^F , and
- *complex formulas* are constructed as in F-Logic with concrete domains.

A formula without modal atoms is an *objective* formula.

An *autoepistemic interpretation* is a pair $\langle \mathbf{I}, \Gamma \rangle$, where $\mathbf{I} = \langle U, U^D, \prec_U, \in_U, \mathbf{I}_F, \mathbf{I}_P, \mathbf{I}_{\mapsto} \rangle$ is an F-structure and $\Gamma \subseteq \mathcal{L}_L^F$ is a set of sentences, called the *belief set*. Satisfaction of objective atomic formulas in $\langle \mathbf{I}, \Gamma \rangle$ corresponds to satisfaction in \mathbf{I} .

Satisfaction of a formula $L\phi$ ($\phi \in \mathcal{L}_L^F$) in an interpretation $\langle \mathbf{I}, \Gamma \rangle$ with respect to a variable assignment B under the *any-name semantics*² is defined as follows:

$\mathbf{I}, B \models_{\Gamma} L\phi$ iff, for some variable substitution(s) β , associated with B , $\phi\beta$ is closed and $\phi\beta \in \Gamma$.

Satisfaction of complex formulas is defined as follows:

- $\mathbf{I}, B \models_{\Gamma} \neg\phi$ iff $\mathbf{I}, B \not\models_{\Gamma} \phi$
- $\mathbf{I}, B \models_{\Gamma} \phi \wedge \psi$ iff $\mathbf{I}, B \models_{\Gamma} \phi$ and $\mathbf{I}, B \models_{\Gamma} \psi$,
- $\mathbf{I}, B \models_{\Gamma} \phi \vee \psi$ iff $\mathbf{I}, B \models_{\Gamma} \phi$ or $\mathbf{I}, B \models_{\Gamma} \psi$,
- $\mathbf{I}, B \models_{\Gamma} \phi \supset \psi$ iff $\mathbf{I}, B \models_{\Gamma} \neg\phi$ or $\mathbf{I}, B \models_{\Gamma} \psi$,
- $\mathbf{I}, B \models_{\Gamma} \forall_a x.\phi$ iff for every abstract x -variant B'_a of B , $\mathbf{I}, B'_a \models_{\Gamma} \phi$,
- $\mathbf{I}, B \models_{\Gamma} \exists_a x.\phi$ iff for some abstract x -variant B'_a of B , $\mathbf{I}, B'_a \models_{\Gamma} \phi$,
- $\mathbf{I}, B \models_{\Gamma} \forall_c x.\phi$ iff for every concrete x -variant B'_c of B , $\mathbf{I}, B'_c \models_{\Gamma} \phi$, and
- $\mathbf{I}, B \models_{\Gamma} \exists_c x.\phi$ iff for some concrete x -variant B'_c of B , $\mathbf{I}, B'_c \models_{\Gamma} \phi$.

$\langle \mathbf{I}, \Gamma \rangle$ is a *model* of ϕ , denoted $\mathbf{I} \models_{\Gamma} \phi$, if $\mathbf{I}, B \models_{\Gamma} \phi$ for every variable assignment B . This extends to sets of formulas in the usual way. A set of formulas $A \subseteq \mathcal{L}_L^F$ *entails* a sentence ϕ with respect to a belief set Γ , denoted $A \models_{\Gamma} \phi$, if for every interpretation \mathbf{I} such that $\mathbf{I} \models_{\Gamma} A$, $\mathbf{I} \models_{\Gamma} \phi$.

A *stable expansion* is the set of beliefs of an ideally introspective agent, given some base set. A belief set $T \subseteq \mathcal{L}_L^F$ is a *stable expansion* of a base set $A \subseteq \mathcal{L}_L^F$ iff $T = \{\phi \mid A \models_T \phi\}$.

²[9] presents also the all-names semantics, but we follow [25, 9] in their choice for the any-name semantics.



A formula ϕ is an *autoepistemic consequence* of A if ϕ is included in every stable expansion of A . $Cons(A)$ denotes the set of all autoepistemic consequences of A . $Cons_o(A)$ denotes the restriction of $Cons(A)$ to objective formulas: $Cons_o(A) = Cons(A) \cap \mathcal{L}^F$.

Every stable expansion T is a stable set, which means that it fulfills the following properties: (a) T is closed under first-order entailment, (b) if $\phi \in T$ then $\mathsf{L}\phi \in T$, and (c) if $\phi \notin T$ then $\neg\mathsf{L}\phi \in T$. If T is consistent, the converses of (b) and (c) also hold.

Embedding Logic Programs We define an embedding as a function which takes a logic program P as its argument and returns a set of sentences in the FF-AEL language obtained from Σ_P .

Since the unique-names assumption does not hold in FF-AEL in general, it is necessary to axiomatize default uniqueness of names (as introduced by [25]). With UNA_Σ we denote the set of axioms

$$\neg\mathsf{L}(t_1 = t_2) \supset t_1 \neq t_2, \quad \text{for all distinct ground terms } t_1, t_2.$$

The authors of [9] have chosen to axiomatize default uniqueness ($\neg\mathsf{L}(t_1 = t_2) \supset t_1 \neq t_2$), instead of rigid uniqueness ($t_1 \neq t_2$), because they wanted to allow a first-order theory which is added to the embedding to override this inequality, rather than introduce an inconsistency.

Let r be a normal rule of form (4.1). Then:

$$\tau_{HP}(r) = (\forall) \bigwedge b_i \wedge \bigwedge \neg\mathsf{L}c_j \supset h,$$

such that each variable which occurs in a concrete body atom is quantified using \forall_c , and all other variables are quantified using \forall_a . For a normal logic program P , we define:

$$\tau_{HP}(P) = \{\tau_{HP}(r) \mid r \in P\} \cup UNA_{\Sigma_P}.$$

Recall the three rules (4.2), (4.3), and (4.4), which are part of every F-Logic program. These rules translate to FF-AEL as follows:

$$\forall_a \ x, y, z \quad (x :: y \wedge y :: z \supset x :: z), \quad (4.5)$$

$$\forall_a \ x, y, z \quad (x : y \wedge y :: z \supset x : z), \text{ and} \quad (4.6)$$

$$\forall_a \ x \quad (x :: x). \quad (4.7)$$

It can be easily verified using the definition of interpretations and satisfaction in F-Logic that these formulas are all tautologies of F-Logic and thus of FF-AEL (they are included in every stable expansion). Therefore, one may disregard these rules in the translation.

Faithfulness of the embedding is established in the following theorem.

Theorem 4. *A Herbrand interpretation M of a normal F-Logic program P is a stable model of P iff there is a consistent stable expansion T of $\tau_{HP}(P)$ such that M coincides with the set of objective ground atoms and molecules in T .*

Proof. Follows immediately from [9, Theorem 1] and Theorem 3. \square

We define the combination of a program P and an F-Logic theory $\Phi \in \mathcal{L}^F$ as: $\iota(\Phi, P) = \Phi \cup \tau_{HP}(P) \subseteq \mathcal{L}_\perp^F$, where $\Sigma_{\mathcal{L}^F}$ is the union of the signatures Σ_Φ and Σ_P .



4.4 WSML Logical Expressions

The signature of a WSML language \mathcal{L}^F is $\Sigma = \langle \mathcal{F}, \mathcal{P}, \mathcal{F}^D, \mathcal{P}^D \rangle$, as in F-Logic with concrete domains, with the distinction that function and predicate symbols do not have an associated arity, and $_ot, _it \notin \mathcal{P} \cup \mathcal{P}^D$. The symbols $_ot, _it$ may not be used in the language, because they are used to axiomatize the semantics of the *ot* (*ofType*) and *it* (*impliesType*) molecules.

Terms and atoms are defined as in F-Logic with concrete domains. If t_1, t_2 are terms, then $t_1 = t_2$ is an atom. Molecules are defined analogously to F-Logic: if t_1, t_2, t_3 are terms, then $t_1 : t_2$, $t_1 :: t_2$, and $t_1[t_2 \times t_3]$, with $\times \in \{ot, it, hv\}$, are molecules.

WSML formulas are inductively defined as follows, with $\phi, \psi \in \mathcal{L}^F$:

- atoms and molecules are formulas;
- $\neg\phi$ is a formula;
- *not* ϕ is a formula;
- $\phi \star \psi$, with $\star \in \{\wedge, \vee, \supset, \equiv\}$, is a formula;
- $\forall_a x(\phi)$, with $x \in \mathcal{V}$, is a formula;
- $\exists_a x(\phi)$, with $x \in \mathcal{V}$, is a formula;
- $\forall_c x(\phi)$, with $x \in \mathcal{V}$, is a formula; and
- $\exists_c x(\phi)$, with $x \in \mathcal{V}$, is a formula.

The semantics of WSML formulas is defined through a translation to FF-AEL: let Φ be a set of WSML formulas, then $tr(\Phi)$ is an FF-AEL theory, obtained as follows: for each $\phi \in \Phi$, $tr(\phi) = \forall(\phi')$, where ϕ' is obtained from ϕ by replacing each occurrence of *not* with \neg ; $tr(\Phi) = \{tr(\phi) \mid \phi \in \Phi\}$, *hv* is replaced with \rightarrow , and molecules of the forms $t_1[t_2 \text{ ot } t_3]$, $t_1[t_2 \text{ it } t_3]$, with t_1, t_2, t_3 terms, are replaced with atoms of the forms $_ot(t_1, t_2, t_3)$, $_it(t_1, t_2, t_3)$, respectively. Finally, the following formulas are added in order to axiomatize the semantics of the *ot* and *it* molecules:

$$\forall_a x, y, z, v, w \quad (_ot(x, y, z) \wedge v[y \rightarrow w] \wedge v : x \wedge \neg Lw : z \supset \perp), \quad (4.8)$$

$$\forall_a x, y, z, v, w \quad (_it(x, y, z) \wedge v[y \rightarrow w] \wedge v : x \supset w : z), \text{ and} \quad (4.9)$$

$$\forall_a x, y, z \quad (\forall_a v, w (v[y \rightarrow w] \wedge v : x \supset w : z) \supset _it(x, y, z)). \quad (4.10)$$

Any WSML formula is a WSML-Full formula. A *WSML-Full theory* is a set of WSML-Full sentences, plus the sentences (4.8), (4.9), and (4.10).

A WSML-FOL formula is a WSML-Full formula which does not contain *ot* molecules, and the default negation operator *not*. A *WSML-FOL theory* is a set of WSML-FOL sentences, plus the sentences (4.9) and (4.10).



WSML-Rule WSML-Rule formulas are of the form

$$(\forall)b_1 \wedge \dots \wedge b_l \wedge \text{not } c_1 \wedge \dots \wedge \text{not } c_m \supset h \quad (4.11)$$

where $b_1, \dots, b_l, c_1, \dots, c_m$ are atoms or *it-free*³ molecules, with l, m nonnegative integers, and h is an abstract equality-free atom or molecule. Additionally, each quantifier is either abstract (\forall_a) or concrete (\forall_c). Finally, variables quantified using an abstract quantifier (\forall_a) may not occur in a concrete term or a concrete atom.

A *WSML-Rule theory* is a set of WSML-Rule sentences, plus the sentences (4.8) and (4.9).

Concrete atoms in WSML-Rule correspond to the common built-in atoms in Logic Programming. Since the difference between concrete and abstract quantifiers is usually not made explicit in Logic Programming, one could induce the type of quantifier as follows: if a variable occurs in a concrete atom or term, then it should be quantified with the \forall_c , and otherwise with \forall_a .

WSML-Flight A *WSML-Flight theory* is a WSML-Rule theory for which holds that, for every formula of the form (4.11), every variable occurs in a positive abstract body atom⁴ b_i , no function symbol in (4.11) is used with an arity higher than 0, and the theory is *locally stratified*⁵.

These conditions correspond to the usual safety condition which must hold for Datalog programs, and the usual local stratification for Logic Programs.

WSML-DL We describe the WSML-DL syntax in a way similar to the usual syntax specification of a Description Logic, with the difference that we use a classical logic-style syntax, with predicate symbols and variables.

\top, \perp are descriptions; if $p \in \mathcal{P}^D$, $f \in \mathcal{F}$ and $x \in \mathcal{V}$, then $x:f, p(x)$ are descriptions. Let ϕ, ψ be descriptions, then:

- $\phi \wedge \psi$ is a description;
- $\phi \vee \psi$ is a description;
- $\neg\phi$ is a description;
- $\forall_a y(x[r \text{ hv } y] \supset \phi_y)$, with $y \in \mathcal{V}$ and $r \in \mathcal{F}$, is a description;

³This requirement is necessary, because *it* molecules have an if and only if semantics, which can only be partly axiomatized in a rules language.

⁴Note that extensions might relax this condition and define certain binding patterns for concrete predicates. This is possible if there is a functional dependency between the arguments of a predicate.

⁵The ground instantiation of a WSML-Flight theory Φ , denoted $gr(\Phi)$, is the union of all possible substitutions of the variables in the formulas in Φ with ground terms, where variables with an abstract quantifier are substituted with abstract terms, and variables with a concrete quantifier are substituted with concrete terms.

Each atom or molecule in $gr(\Phi)$ is assigned a stratum, which is an integer. We say that $gr(\Phi)$ is stratified if there is an assignment of strata such that: if an atom or molecule p occurs positively in a rule with q as its head, then p has the same or a lower stratum, and if p occurs negatively in a rule with q as its head, then the stratum of p must be lower than the stratum of q .

If $gr(\Phi)$ is stratified, then Φ is locally stratified.



- $\exists_a y(x[r \text{ hv } y] \wedge \phi_y)$, with $y \in \mathcal{V}$ and $r \in \mathcal{F}$, is a description;
- $\exists_a y_1, \dots, y_n(\wedge x[r \text{ hv } y_i] \wedge \wedge \phi_{y_i} \wedge \wedge \neg y_i = y_j)$, with $y_1, \dots, y_n \in \mathcal{V}$ and $r \in \mathcal{F}$, is a description;
- $\forall_a y_1, \dots, y_{n+1}(\wedge x[r \text{ hv } y_i] \wedge \wedge \phi_{y_i} \supset \vee y_i = y_j)$, with $y_1, \dots, y_{n+1} \in \mathcal{V}$ and $r \in \mathcal{F}$, is a description;
- $\forall_c y(x[r \text{ hv } y] \supset p(y))$, with $y \in \mathcal{V}$, $r \in \mathcal{F}$, and $p \in \mathcal{P}^D$, is a description;
- $\exists_c y(x[r \text{ hv } y] \wedge p(y))$, with $y \in \mathcal{V}$, $r \in \mathcal{F}$, and $p \in \mathcal{P}^D$, is a description;
- $\exists_c y_1, \dots, y_n(\wedge x[r \text{ hv } y_i] \wedge \wedge p(y) \wedge \wedge \neg y_i = y_j)$, with $y_1, \dots, y_n \in \mathcal{V}$, $r \in \mathcal{F}$, and $p \in \mathcal{P}^D$, is a description; and
- $\forall_c y_1, \dots, y_{n+1}(\wedge x[r \text{ hv } y_i] \wedge \wedge p(y) \supset \vee y_i = y_j)$, with $y_1, \dots, y_{n+1} \in \mathcal{V}$, $r \in \mathcal{F}$, and $p \in \mathcal{P}^D$, is a description,

where ϕ_y is obtained from ϕ by renaming each occurrence of x (y) to y (x).

If $p \in \mathcal{F}$, $p' \in \mathcal{P}^D$ and $a, b, c_1, c_2 \in \mathcal{F}$, $a' \in \mathcal{F}^D$, then $a:p$, $p'(a')$, $a[p \text{ hv } b]$, $a[p \text{ hv } a']$, $a = b$, and $\neg a = b$ are formulas. Let ϕ, ψ be descriptions, then:

- $c_1 :: c_2$ is a formula;
- $c_1[p \text{ it } c_2]$ is a formula;
- $\forall_a x(\phi \supset \psi)$ is a formula;
- $\forall_a x(\phi \equiv \psi)$ is a formula;
- $\forall_a x, y(x[p \text{ hv } y] \supset x[q \text{ hv } y])$, with $p, q \in \mathcal{F}$ and $x, y \in \mathcal{V}$, is a formula;
- $\forall_a x, y(x[p \text{ hv } y] \supset y[q \text{ hv } x])$, with $p, q \in \mathcal{F}$ and $x, y \in \mathcal{V}$, is a formula;
- $\forall_a x, y(x[p \text{ hv } y] \equiv x[q \text{ hv } y])$, with $p, q \in \mathcal{F}$ and $x, y \in \mathcal{V}$, is a formula;
- $\forall_a x, y(x[p \text{ hv } y] \equiv y[q \text{ hv } x])$, with $p, q \in \mathcal{P}$ and $x, y \in \mathcal{V}$, is a formula; and
- $\forall_a x, y, z(x[p \text{ hv } y] \wedge y[p \text{ hv } z] \supset x[p \text{ hv } z])$, with $p \in \mathcal{F}$ and $x, y, z \in \mathcal{V}$, is a formula.

Given the molecules $a:c_1$, $c_1 :: c_2$, $c_1[p \text{ it } c_2]$ and $a[p \text{ hv } b]$, with $a, b, c, p \in \mathcal{F}$, then we say that c_1, c_2 are used as class identifiers, p is used as an attribute identifier, and a, b are used as instance identifiers.

A *WSML-DL theory* is a set of WSML-DL formulas plus the sentences (4.9) and (4.10), such that the function symbols in \mathcal{F} used as class identifiers, attribute identifiers, and instance identifiers, are disjoint. Additionally, no variable x in a concrete (abstract) atom $p(x)$ or molecule $x:p$ occurs in the scope of an abstract (concrete) quantifier, and for every molecule of the form $x[r \text{ hv } y]$, the variable x is in the scope of an abstract quantifier and if the variable y occurs in the scope of an abstract (concrete) quantifier, then for every hv molecule with r as its attribute symbol, it is the case that the second argument is either an abstract (concrete) term or a variable occurring in the scope of an abstract (concrete) quantifier. The latter condition essentially ensures a separation of abstract and concrete attribute identifiers.



In Description Logic terminology, the concrete predicate symbols correspond with datatypes; each such predicate symbol, which may only be used with the arity 1, corresponds with a datatype. The symbols in \mathcal{F} correspond with the class, abstract role, concrete role, and individual identifiers. Although there is not such a strict separation in the signature as is usual in DLs, the conditions formulated above ensure such a separation in each theory.

WSML-Core A WSML-DL theory which is also a WSML-Flight theory, with the addition of the sentence (4.10), is a WSML-Core theory.

4.5 Correspondence with Target Formalisms

In this section we show the correspondences between the WSML language variants and the logical language formalisms which have originally motivated these variants. We call these formalisms the *target formalisms*.

The target formalisms of WSML-Core, WSML-DL, WSML-Flight and WSML-Rule are, $DHL(\mathbf{D})$, $SHIQ(\mathbf{D})$, the Perfect Model Semantics for logic programs, and the Well-Founded Semantics for logic programs, respectively. The reasoning tasks for these formalisms are slightly different from the usual reasoning tasks for autoepistemic logic.

In the remainder of this section, we treat each WSML variant separately; we first briefly recall the target formalism and the associated reasoning tasks and then formally prove the correspondence between the WSML variant and the target formalism.

4.5.1 WSML-Full

The usual reasoning tasks for autoepistemic logic are existence of stable expansions, inclusion of a formula in one stable expansion, and inclusion of a formula in all stable expansions (autoepistemic consequence) (cf. [31]). It is expected that these reasoning tasks are relevant for WSML-Full as well.

A subset of WSML-Full which is often considered useful is WSML-FOL. From the definition, we can see that WSML-FOL does not make use of the nonmonotonic modal operator L and thus corresponds with classical first-order logic with equality, extended with Frames. It was shown in [25] that in case an FO-AEL theory is objective, it has exactly one stable expansion T and the objective subset of T , T_o , is the first-order closure of the theory. Therefore, entailment in first-order logic is equivalent to checking inclusion of a formula in a stable expansion in WSML-FOL.

Theorem 5. *Given a WSML language \mathcal{L}^F , a WSML-FOL theory $\Phi \in \mathcal{L}^F$ and a WSML-FOL formula ϕ , then*

- $\Phi \models_{\mathfrak{f}} \phi$ iff
- ϕ is included in any stable expansion of Φ iff
- ϕ is an autoepistemic consequence of Φ ,



and

- Φ is unsatisfiable iff
- T is a stable expansion of Φ and $\perp \in T$ iff
- T is a stable expansion of Φ and $T \cap \mathcal{L}^F = \mathcal{L}^F$.

Proof. Follows from the definition of WSML-FOL, the definition of \models_f , and the results in [25]. \square

4.5.2 WSML-DL

The usual reasoning tasks for the Description Logic $\mathcal{SHIQ}(\mathbf{D})$ are concept satisfiability, knowledge base satisfiability and logical entailment (usually restricted to formulas of a specific shape, such as ground atoms and subsumption axioms).

A WSML-DL theory is completely first-order. It is well known, as mentioned above, that an objective FO-AEL theory Φ has exactly one stable expansion T and its kernel, $T \cap \mathcal{L}^F$, is the first-order closure of Φ , and thus contains all objective formulas first-order entailed by Φ .

Theorem 6. *If Φ is a WSML-DL theory and ϕ is a WSML-DL axiom, then there are a corresponding $\mathcal{SHIQ}(\mathbf{D})$ theory Φ' and $\mathcal{SHIQ}(\mathbf{D})$ axiom ϕ' (and vice versa) such that*

- $\Phi' \models \phi'$ iff
- ϕ is included in any stable expansion of Φ iff
- ϕ is an autoepistemic consequence of Φ ,
- Φ' is unsatisfiable iff
- T is a stable expansion of Φ and $\perp \in T$ iff
- T is a stable expansion of Φ and $T \cap \mathcal{L}^F = \mathcal{L}^F$,

and

- a concept A is unsatisfiable with respect to Φ' iff
- T is a stable expansion of $\Phi \cup A(i)$ and $\perp \in T$ iff
- T is a stable expansion of $\Phi \cup A(i)$ and $T \cap \mathcal{L}^F = \mathcal{L}^F$,

where i is a constant symbol not used in Φ .

Proof. It is easy to verify that for each $\mathcal{SHIQ}(\mathbf{D})$ theory Ψ , there is a WSML-DL theory Φ such that $\Phi' = \Psi$.

Let Φ be a WSML-DL theory and ϕ be a WSML-DL axiom and let ϕ' be the $\mathcal{SHIQ}(\mathbf{D})$ axiom obtained from ϕ by replacing each molecule of the form $t:f$ with an atom of the form $f(t)$, each molecule of the form $t_1[r\text{hvt}_2]$ with an atom of the form $r(t_1, t_2)$, each formula of the form $t_1 :: t_2$ with a formula of



the form $\forall x(t_1(x) \supset t_2(x))$, each formula of the form $t_1[t_2 \text{ it } t_3]$ with a formula of the form $\forall x, y(t_1(x) \wedge t_2(x, y) \supset t_3(y))$ and let Φ' be obtained from Φ in the same way, leaving out the formula (4.9). Φ and ϕ are \mathcal{E} -safe, and thus cardinal. It can be shown that this translation preserves satisfiability and entailment, in a similar way as Lemma 2 and Theorem 2. The only-if direction then follows immediately from Theorem 5. \square

4.5.3 WSML-Rule

The usual reasoning task for the Well-Founded Semantics is ground entailment, i.e. inclusion in the well-founded model. Additionally, as WSML-Rule has integrity constraints, consistency checking is also an important reasoning task.

Reasoning in the Well-Founded Semantics can be seen as an approximation to reasoning in the Stable Model Semantics. In fact, given a logic program P , if a ground atom α is true in the well-founded model of P , then α is included in every stable model of P , and thus is entailed using cautious reasoning. Since there is a natural correspondence between the stable model semantics and autoepistemic logic, we use the stable model semantics in the remainder.

The three main reasoning tasks for the stable model semantics are: existence of a stable model (program consistency), inclusion of a ground atom in a stable model (brave reasoning), and inclusion of a ground atom in all stable models (cautious reasoning). It is conjectured that the main reasoning task for the Semantic Web is querying, which corresponds to cautious reasoning.

Note that in case P is negation-free, then P has one stable model and thus (by Theorem 4) $\tau_{HP}(P)$ has one consistent stable expansion. In this case, brave and cautious reasoning coincide. Furthermore, the stable model corresponds to the minimal Herbrand model of P . Therefore, instead of brave or cautious reasoning, we also speak about ground entailment in this case.

Theorem 7. *If Φ is a WSML-Rule theory and α is a ground atom or it-free molecule, then there are a corresponding normal F-Logic Program P and ground atom or it-free molecule α' (and vice versa) such that*

- α' is a consequence of P under cautious reasoning iff
- α is an autoepistemic consequence of Φ ,
- α' is a consequence of P under brave reasoning iff
- α is included in any stable expansion of Φ ,

and

- P is inconsistent (i.e. has no stable model) iff
- Φ has no consistent stable expansion.

Proof. It is easy to verify that for each WSML-Rule theory Φ , there is a logic program P such that $\tau_{HP}(P) = \Phi$.

Let P be a normal F-Logic Program and α' be a ground atom or it-free molecule, then $\tau_{HP}(P)$ is obviously a WSML-Rule theory, and α is obtained



from α' by replacing an atom of the form $\textit{it}(t_1, t_2, t_3)$ with a molecule $t_1[t_2 \textit{it} t_3]$, and an atom of the form $\textit{ot}(t_1, t_2, t_3)$ with a molecule $t_1[t_2 \textit{ot} t_3]$. The only-if direction of the equivalences follow immediately from Theorem 4. \square

4.5.4 WSML-Flight

The usual reasoning task for logic programs under the perfect model semantics is ground entailment, which we call here *consequence*. Additionally, as WSML-Flight has integrity constraints, consistency checking is also an important reasoning task. The correspondence for WSML-Flight follows immediately from Theorem 7, since every locally stratified logic program has exactly one stable model, which corresponds with the perfect model [17]:

Corollary 3. *If Φ is a WSML-Flight theory and α is a ground atom or it-free molecule, then there are a corresponding normal F-Logic Program P and ground atom or it-free molecule α' (and vice versa) such that*

- α' is a consequence of P under the perfect model semantics iff
- α is an autoepistemic consequence of Φ ,

and

- P is inconsistent (i.e. has no stable model) iff
- Φ has no consistent stable expansion.

4.5.5 WSML-Core

The reasoning tasks for WSML-Core are subsumed by those of WSML-DL. The correspondence for WSML-Core follows immediately from Theorem 6:

Corollary 4. *If Φ is a WSML-Core theory and ϕ is a WSML-Core axiom, then there are a corresponding $\mathcal{DHL}(\mathbf{D})$ theory Φ' and $\mathcal{DHL}(\mathbf{D})$ axiom ϕ' (and vice versa) such that*

- $\Phi' \models \phi'$ iff
- ϕ is included in any stable expansion of Φ iff
- ϕ is an autoepistemic consequence of Φ ,

and

- Φ' is unsatisfiable iff
- T is a stable expansion of Φ and $\perp \in T$ iff
- T is a stable expansion of Φ and $T \cap \mathcal{L}^F = \mathcal{L}^F$.



Abstract Syntax	Surface Syntax
$tr(p(t_1, \dots, t_n))$	$p(t_1, \dots, t_n)$
$tr(\top)$	true
$tr(\perp)$	false
$tr(t_1 = t_2)$	$t_1 ::= t_2$
$tr(t_1 : t_2)$	t_1 memberOf t_2
$tr(t_1 :: t_2)$	t_1 subConceptOf t_2
$tr(t_1[t_2 \text{ hv } t_3])$	$t_1[t_2$ hasValue $t_3]$
$tr(t_1[t_2 \text{ ot } t_3])$	$t_1[t_2$ ofType $t_3]$
$tr(t_1[t_2 \text{ it } t_3])$	$t_1[t_2$ impliesType $t_3]$
$tr(\neg\phi)$	neg $tr(\phi)$
$tr(\text{not } \phi)$	naf $tr(\phi)$
$tr((\forall)b_1 \wedge \dots \wedge b_l \wedge$ $\text{not } c_1 \wedge \dots \wedge \text{not } c_m \supset h),$ with $m \geq 1$	$tr(h)$:- $tr(b_1)$ and \dots and $tr(b_l)$ and naf $tr(c_1)$ and \dots and naf $tr(c_m)$
$tr(\phi \wedge \psi)$	$tr(\phi)$ and $tr(\psi)$
$tr(\phi \vee \psi)$	$tr(\phi)$ or $tr(\psi)$
$tr(\phi \supset \psi)$	$tr(\phi)$ implies $tr(\psi)$
$tr(\phi \equiv \psi)$	$tr(\phi)$ equivalent $tr(\psi)$
$tr(\forall x(\phi))$	forall $?x(tr(\phi))$
$tr(\exists x(\phi))$	exists $?x(tr(\phi))$

Table 4.1: Mapping from WSML abstract to surface syntax

4.6 Mapping to the WSML Surface Syntax

Table 4.1 shows the mapping from the WSML abstract syntax to the surface syntax specified in the language reference [12]. In the table, t_1, \dots, t_n are terms, p is a predicate symbol, x is a variable, $b_1, \dots, b_l, c_1, \dots, c_m, h$ are atoms or molecules, and ϕ, ψ are formulas.

Notice that the original WSML specification allows complex formulas in both the bodies and the heads of the rules of WSML-Flight and WSML-Rule Ontologies. These can be eliminated through the usual Lloyd-Topor transformations described in the specification. Notice also that the WSML surface syntax allows anonymous identifiers; these simply correspond to ground terms in the abstract syntax in the usual way.

The WSML language reference [12] also specifies a conceptual syntax. The reference documents describes the translation from the conceptual syntax to the logical expression syntax, which is used to obtain the semantics. Future versions of the referenced document might consider the syntax specified in this document.



5 Conclusions and Future Work

In this deliverable we have presented a comprehensive semantic framework for WSML, based on first-order autoepistemic logic (FO-AEL), extended with Frames and concrete domains. We have provided semantics for WSML-Full, we have demonstrated language layering, and we have shown that each of the defined WSML variants semantically corresponds to the target formalism. This allows the use of standard reasoning techniques and efficient implementations for the DL and Rule variants.

An interesting topic for future work is the application of this semantic framework to other Semantic Web languages, such as RDF, OWL and RIF, as well as other approaches to combining rules and Ontologies, such as dl-programs [15], $\mathcal{DL}+log$ [34], and SWRL [20]. In case only positive rules are considered, SWRL is very close to our semantics.

As a starting point for the incorporation of RDF into our framework, we will use the logical reconstruction in [10].

An alternative formalism which has been used for combining rules and Ontologies is MKNF [30]. This approach is very similar to ours (except that Frames are not considered in MKNF), although it remains to be seen what the actual relationship is between FO-AEL and MKNF. The embedding of logic programs used in [30] is slightly different from the embedding τ_{HP} which we considered in this deliverable, but it is very close to the embedding τ_{EH} considered in [9], which is obtained from τ_{HP} by including modal atoms in the bodies and heads of the rules for each positive atom in the logic program. However, an embedding similar to τ_{HP} is also possible in MKNF. It is illustrated in [29] how, in this setting, nonmonotonic rules can be used to extend the expressiveness of OWL DL Ontologies.

Acknowledgements

The work is funded by the European Commission under the projects DIP, Knowledge Web, InfraWebs, SEKT, and ASG; by the Science Foundation Ireland under Grant No. SFI/02/CE1/I131; by the FIT-IT (Forschung, Innovation, Technologie - Informationstechnologie) under the projects RW² and TSC.

The authors would like to thank to all the members of the WSML working group for their advice and input into this document.



Bibliography

- [1] H. Andréka, J. van Benthem, and I. Németi. Modal languages and bounded fragments of predicate logic. *Journal of Philosophical Logic*, 27:217–274, 1998.
- [2] 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). W3C Member Submission 09 September 2005, 2005.
- [3] Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors. *The Description Logic Handbook*. Cambridge University Press, 2003.
- [4] Franz Baader and Philipp Hanschke. A scheme for integrating concrete domains into concept languages. Technical Report RR-91-10, Deutsches Forschungszentrum für Künstliche Intelligenz GmbH, 1991.
- [5] Mira Balaban. The F-logic approach for description languages. *Annals of Mathematics and Artificial Intelligence*, 15(1):19–60, 1995.
- [6] Mira Balaban and Adi Eyal. DFL - a dialog based integration of concept and rule reasoners. *Data & Knowledge Engineering*, 38(3):301–334, 2001.
- [7] Steve Battle, Abraham Bernstein, Harold Boley, Benjamin Grosf, Michael Gruninger, Richard Hull, Michael Kifer, David Martin, Sheila McIlraith, Deborah McGuinness, Jianwen Su, and Said Tabet. Semantic web services language (SWSL). W3C Member Submission 09 September 2005, 2005.
- [8] Weidong Chen, Michael Kifer, and David Scott Warren. HILOG: A foundation for higher-order logic programming. *Journal of Logic Programming*, 15(3):187–230, 1993.
- [9] Jos de Bruijn, Thomas Eiter, Axel Polleres, and Hans Tompits. Embedding non-ground logic programs into autoepistemic logic for knowledge-base combination. In *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI-07)*, Hyderabad, India, January 6–12 2007.
- [10] Jos de Bruijn, Enrico Franconi, and Sergio Tessaris. Logical reconstruction of normative RDF. In *OWL: Experiences and Directions Workshop (OWLED-2005)*, Galway, Ireland, November 2005.
- [11] Jos de Bruijn and Stijn Heymans. Translating ontologies from predicate-based to frame-based languages. In *Proceedings of the Second International Conference on Rules and Rule Markup Languages for the Semantic Web (RuleML-2006)*, Athens, Georgia, USA, November 10-11 2006. IEEE.
- [12] Jos de Bruijn, Holger Lausen, Reto Krummenacher, Axel Polleres, Livia Predoiu, Michael Kifer, and Dieter Fensel. The web service modeling language WSML. WSML Final Draft D16.1v0.21, WSML, 2005.



- [13] Jos de Bruijn, Holger Lausen, Axel Polleres, and Dieter Fensel. The web service modeling language: An overview. In *Proceedings of the 3rd European Semantic Web Conference (ESWC2006)*, number 4011 in Lecture Notes in Computer Science, pages 590–604, Budva, Montenegro, June 2006. Springer-Verlag.
- [14] Jos de Bruijn, Axel Polleres, Rubén Lara, and Dieter Fensel. OWL⁻. Final draft d20.1v0.2, WSML, 2004.
- [15] 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.
- [16] 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.
- [17] 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.
- [18] Michael Gelfond and Vladimir Lifschitz. Classical negation in logic programs and disjunctive databases. *New Generation Computing*, 9(3/4):365–386, 1991.
- [19] Benjamin N. Grosz, Ian Horrocks, Raphael Volz, and Stefan Decker. Description logic programs: Combining logic programs with description logic. In *Proc. Intl. Conf. on the World Wide Web (WWW-2003)*, Budapest, Hungary, 2003.
- [20] Ian Horrocks and Peter F. Patel-Schneider. A proposal for an OWL rules language. In *Proc. of the Thirteenth International World Wide Web Conference (WWW 2004)*, pages 723–731. ACM, 2004.
- [21] Uwe Keller, Rubén Lara, Holger Lausen, Axel Polleres, and Dieter Fensel. Automatic location of services. In *Proceedings of the 2nd European Semantic Web Conference (ESWC2005)*, pages 1–16. Springer-Verlag, 2005.
- [22] Uwe Keller, Holger Lausen, and Michael Stollberg. On the semantics of functional descriptions of web services. In *Proceedings of the 3rd European Semantic Web Conference (ESWC2006)*, Budva, Montenegro, June 2006. Springer.
- [23] Michael Kifer. Rules and ontologies in f-logic. In *Reasoning Web, First International Summer School, Tutorial Lectures*, pages 22–34, Msida, Malta, July 2005.
- [24] Michael Kifer, Georg Lausen, and James Wu. Logical foundations of object-oriented and frame-based languages. *JACM*, 42(4):741–843, 1995.
- [25] Kurt Konolige. Quantification in autoepistemic logic. *Fundamenta Informaticae*, 15(3–4):275–300, November-December 1991.



- [26] Alon Y. Levy and Marie-Christine Rousset. Combining horn rules and description logics in CARIN. *Artificial Intelligence*, 104:165 – 209, 1998.
- [27] John W. Lloyd. *Foundations of Logic Programming (2nd edition)*. Springer-Verlag, 1987.
- [28] Boris Motik. On the properties of metamodeling in OWL. In *4th International Semantic Web Conference (ISWC2005)*, pages 548–562, 2005.
- [29] Boris Motik, Ian Horrocks, and Ulrike Sattler. Can owl and logic programming live together happily ever after? In *Proc. of the 5th Int. Semantic Web Conf. (ISWC 2006)*, Athens, GA, USA, November 5 – 9 2006.
- [30] Boris Motik and Riccardo Rosati. A faithful integration of description logics with logic programming. In *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI-07)*, Hyderabad, India, January 6–12 2007.
- [31] Ilkka Niemelä. On the decidability and complexity of autoepistemic reasoning. *Fundamenta Informaticae*, 17(1,2):117–155, 1992.
- [32] Peter F. Patel-Schneider, Patrick Hayes, and Ian Horrocks. OWL web ontology language semantics and abstract syntax. Recommendation 10 February 2004, W3C, 2004.
- [33] T. C. Przymusiński. On the semantics of stratified deductive databases. In *Proceedings of the Workshop on the Foundations of Deductive Databases and Logic Programming*, pages 433–443, Washington, DC, USA, 1986.
- [34] Riccardo Rosati. $\mathcal{DL} + \text{log}$: Tight integration of description logics and disjunctive datalog. In *KR2006*, 2006.
- [35] James Scicluna, Dumitru Roman, Dieter Fensel, Axel Polleres, and Jos de Bruijn. Ontology-based choreography of WSMO services. Final Draft D14v0.3, WSMO, 19 May 2006.
- [36] Guizhen Yang, Michael Kifer, and Chang Zhao. FLORA-2: A rule-based knowledge representation and inference infrastructure for the semantic web. In *Proceedings of the Second International Conference on Ontologies, Databases and Applications of Semantics (ODBASE)*, Catania, Sicily, Italy, 2003.
- [37] Youyoung Zou, Tim Finin, and Harry Chen. F-OWL: an inference engine for the semantic web. In *Formal Approaches to Agent-Based Systems, Third International Workshop, FAABS 2004, Revised Selected Papers*, volume 3228 of *LNCS*, Greenbelt, USA, 2004. Springer.