MODEL-THEORETIC APPROACHES TO SEMANTIC INTEGRATION (EXTENDED ABSTRACT)

MICHAEL GRÜNINGER

1. Introduction

Many tasks require correct and meaningful communication and integration among intelligent agents and information resources. A major barrier to such interoperability is semantic heterogeneity: different applications, databases, and agents may ascribe disparate meanings to the same terms or use distinct terms to convey the same meaning. The development of ontologies has been proposed as a key technology to support semantic integration—two software systems can be semantically integrated through a shared understanding of the terminology in their respective ontologies.

A semantics-preserving exchange of information between two software applications requires mappings between logically equivalent concepts in the ontology of each application. The challenge of semantic integration is therefore equivalent to the problem of generating such mappings, determining that they are correct, and providing a vehicle for executing the mappings, thus translating terms from one ontology into another.

Current approaches to semantic integration do not fully exploit the model-theoretic structures underlying ontologies. These approaches are typically based on the taxonomic structure of the terminology ([7], [8]) or heuristics-based comparisons of the symbols of the terminology ([1, 5]). Such techniques are well-suited to working with many ontologies currently under development, most of which define a terminology with minimal formal grounding and a set of possible models that does not contain a rich set of features and properties.

However, automated and correct approaches to semantic integration will require ontologies with a deeper formal grounding so that decisions may be made by autonomous software when comparing ontologies for integration. This article presents an approach toward this goal using techniques based on the development of strong ontologies with terminologies grounded in properties of the underlying possible models. With these as inputs, semi-automated and automated components may be used to create mappings between ontologies and perform translations.

The Process Specification Language (PSL) ([2], [3]) is used in this article to demonstrate this approach to ontology construction and integration. PSL consists of a core ontology which outlines basic objects that exist in the domain, and a multitude of definitional extensions that provide a rich terminology for describing process knowledge. These extensions are based on invariants, properties preserved by isomorphism, which partition the first-order models of the core ontology. Using these invariants, semantic mappings between application ontologies and PSL may be semi-automatically generated. In addition, the direct relationship between the

Dagstuhl Seminar Proceedings 04391
Semantic Interoperability and Integration
http://drops.dagstuhl.de/opus/volltexte/2005/39

PSL terminology and the invariants improves the ability to verify the generated results. These semantic mappings may then be used to perform integration between applications or ontologies. They may also be used to analyze the application as well as to bootstrap an ontology to those applications which do not have an associated, explicit, formal ontology.

2. Supporting Interoperability

The organization of the PSL Ontology and the properties of its extensions have been shaped by several design principles. In presenting these principles we make a distinction between hypotheses (that constrain uses of the PSL Ontology) and criteria (that specify properties of the PSL Ontology itself).

2.1. **Interoperability and Completeness.** Intuitively, two applications will be interoperable if they share the semantics of the terminology in their corresponding theories. Sharing semantics between applications is equivalent to sharing models of their theories, that is, the theories have isomorphic sets of models. Of course, the theories (and their models) will have different languages, so we need to make this intuition about isomorphism more precise. In this section, we introduce some of the basic notions required to characterize interoperability; we will demonstrate the relationship to model isomorphism at later in the paper.

We begin with the following definition taken from [4]:

Definition 1. *Let $\mathcal{M}$ be a structure in a language $\mathcal{K}$ and let $\mathcal{N}$ be a structure in a language $\mathcal{L}$. An interpretation Γ of $\mathcal{N}$ in $\mathcal{M}$ consists of*

1. *a formula $\partial_\Gamma(x_0, ..., x_{n-1}) \in \mathcal{K}$,*
2. *for each atomic formula $\phi(y_0, ..., y_{m-1}) \in \mathcal{L}$, a formula $\phi_\Gamma(x_0, ..., x_{m-1}) \in \mathcal{K}$,*
3. *a surjective map $f_\Gamma : \partial_\Gamma(M^n) \rightarrow N$, such that for all formulae $\phi \in \mathcal{L}$ and all $a_i \in \partial_\Gamma(M^n)$,*

$$\mathcal{M} \models \phi_\Gamma(a_0, ..., a_{m-1}) \Leftrightarrow \mathcal{N} \models \phi(f_\Gamma(a_0), ..., f_\Gamma(a_{m-1})$$

Definition 2. *Let $\mathcal{M}^A$ be the set of models of a theory T_A, and let $\mathcal{M}^B$ be the set of models of a theory T_B. The theory T_A is sharable with the theory T_B iff there is an injection $\mu : \mathcal{M}^A \rightarrow \mathcal{M}^B$ such that $\mathcal{M}$ is interpretable in $\mu(\mathcal{M})$.*

However, applications do not explicitly share the models of their theories. Instead, they exchange sentences in such a way that the semantics of the terminology of these sentences is preserved.

Definition 3. *An exchange of a sentence Φ from a theory T_A to a theory T_B is a KIF-conformant syntactic mapping $\sigma : \mathcal{L}(T_A) \rightarrow \mathcal{L}(T_B)$.*

In general, there is no automatable way of specifying the complete set of models of a theory, or even a way of specifying the models themselves. Applications perform inferences in such a way that provable sentences are those sentences that are satisfied by all models of the theory.

Definition 4. *A theory T_A is interoperable with a theory T_B iff for any sentence Φ in $\mathcal{L}(T_A)$ there exists an exchange σ from T_A to T_B such that*

$$T_A \vdash \Phi \Leftrightarrow T_B \vdash \sigma(\Phi)$$

The definitions of sharability and interoperability are a restriction and simplification of previous work in ontology translation discussed in [Ciociou and Nau 2000], where the exchange mapping is called a logical rendering, and the concept of sharable theories is equivalent to the existence of an ontology-based translation between the theories.

For first-order theories and complete first-order inference engines, sharability and interoperability are equivalent, so we adopt the following hypthesis:

Interoperability Hypothesis: :

We are considering interoperability among complete first-order inference engines that exchange first-order sentences.

The soundness and completeness of first-order logic guarantees that the inferences made by a deductive inference engine are exactly those sentences that are satisfied by all models, and that any truth assignment given by a consistency checker is isomorphic to a model.

If we move beyond the expressiveness of first-order logic, we lose completeness, so that there will be sentences that are entailed by a set of models but which are provable by any deductive inference engine. We could therefore have two theories that are sharable but not interoperable.

Note that this is also the reason why we require that the inference engines themselves are complete. An incomplete inference engine would not be able to deduce all sentences that are satisfied by the set of models for its ontology. In this case, the theories would be interoperable, but the applications would not be able to preserve this interoperability.

Also note that we are not imposing the requirement that the ontologies themselves be categorical or even complete. The two applications must simply share the same set of models (up to isomorphism). Ambiguity does not arise from the existence of multiple models for an ontology – it arises because the two applications have nonisomorphic models, that is, the ontology for application A has a model $\mathcal{M}$ that is not isomorphic to any model for the ontology of application B.

2.2. **Quasi-Elementary Classes of Structures.** By the Interoperability Hypothesis, we do not need to restrict ourselves to elementary classes of structures when we are axiomatizing an ontology.

Since the the applications are equivalent to first-order inference engines, they cannot distinguish between structures that are elementarily equivalent.

We therefore introduce the notion of quasi-elementary classes of structures.

Definition 5. *A class of structures $\mathcal{M}_i$ is elementary iff there exists a first-order theory T such that $\mathcal{M}_i = Mod(T)$.*

Definition 6. *$\mathcal{M}_j$ quasi-elementary class of structures iff there exists a first-order theory T such that $\mathcal{M}_j \subseteq Mod(T)$ and any structure $\mathcal{N} \in Mod(T) \setminus \mathcal{M}_j$ is elementarily equivalent to a structure in $\mathcal{M}_j$.*

Thus, the quasi-elementary class forms a subset of the elementary class. Intuitively, the structures in $\mathcal{M}_j$ are the "intended" structures. If $\mathcal{M}_j$ is in fact elementary, then it is axiomatizable by some first-order theory T, and there will only be intended structures in the set of models of T. However, classes of structures that are not axiomatizable will also contain unintended or nonstandard structures.

By the definition of elementary equivalence, any quasi-elementary class of structures will entail the same set of first-order sentences as some elementary class. In

particular, the quasi-elementary class of structures $\mathcal{M}_j$ will entail the first-order theory that axiomatizes the elementary class containing $\mathcal{M}_j$.

2.3. **Characterization of Models.** Employing the Interoperability Hypothesis and the notion of quasi-elementary classes of structures, we impose the following condition on the core theories of the PSL Ontology:

SatAx Criterion: :

Classes of structures for core theories within the PSL Ontology are axiomatized up to elementary equivalence – the core theory is satisfied by any model in the class, and any model of the core theory is elementarily equivalent to a model in the class. Further, each class of structures is characterized up to isomorphism.

If the PSL Ontology did not satisfy this criterion, then either there exist intended models that are not models of the ontology, or there exist unintended models of the ontology. In either case, we cannot guarantee that an exchanged theory is interoperable with the PSL Ontology.

The Definability Criterion can be applied as a methodology for evaluating the axiomatization of an ontology.

The first aspect of this approach is to identify the primary intuitions in some domain. Within PSL, for example, we have intuitions about concepts such as "activity", "activity occurrences", and "timepoints". These intuitions also restrict the scope of the axiomatic theories, and they serve as informal requirements which get formally specified in the classes of structures, and later axiomatized in the theory itself.

The second aspect of the methodology is the specification of some set of structures. These structures provides a rigorous mathematical characterization of the semantics of concepts in the domain. The class of structures corresponding to the intuitions of the ontology are defined either by specifying some class of algebraic or combinatorial structures, or by extending classes of structures defined for other theories within the ontology. The objective is to identify each concept with an element of some mathematical structure, such as a set or a set with additional structure; the underlying theory of the mathematical structure then becomes available as a basis for reasoning about the concepts and their relationships. Examples of structures include graphs, linear orderings, partial orderings, groups, fields, and vector spaces. In particular, given the nonlogical lexicon in some language, structures are isomorphic to the extensions of the relations, functions, and constants denoted by the predicate symbols, function symbols, and constant symbols of the lexicon.

If we wish to represent some domain, we want the necessary properties to be captured by the structures. Ideally, we want properties of the structures to be reflected by the properties of the corresponding concepts in the domain. These characteristics can be used to evaluate the adequacy of the intended structures. If some property is not captured, then we must make a decision about this property. If it is not necessary, then we can ignore it. If it is deemed necessary, then we must extend the characterization of the intended structures so that it includes some formalization of this property.

This relationship between the intuitions and the structures is, of course, informal, but we can consider the domain intuitions as providing a physical interpretation of the structures. In this sense, we can adopt an experimental or empirical approach to

the evaluation of the class of intended structures in which we attempt to falsify these structures. If we can find some objects or behaviour within the domain which do not correspond to an intended structure, then we have provided a counterexample to the class of structures. In response, we can either redefine the scope of the class of structures (i.e. we do not include the behaviour within the characterization of the structures) or we can modify the definition of the class of structures so that they capture the new behaviour.

For example, physicists use various classes of differential equations to model different phenomena. However, they do not use ordinary linear differential equations to model heat diffusion, and they do not use second-order partial differential equations to model the kinematics of springs. If we wish to model some phenomena using a class of differential equations, we can use the equations to predict behaviour of the physical system; if the predictions are falsified by observations, then we have an incorrect set of equations. Similarly, in our case, we can use some class of structures to predict behaviour or characterize states of affairs; if there is no physical scenario in the domain which corresponds to these behaviours or states of affairs, then we intuitively have an incorrect set of structures.

The final aspect of the methodology is the set of axiomatic theories, which are sets of sentences in some language using some logic. In this paper, we restrict ourselves to a first-order languages.

Once we have specified the class of structures, we can formally evaluate an axiomatic theory with respect to this specification. In particular, we want to prove two fundamental properties:

- **Satisfiability:** every structure in the class is a model of the axiomatic theory.
- **Axiomatizability:** every model of the axiomatic theory is isomorphic to some structure in the class.

Strictly speaking, we only need to show that a model exists in order to demonstrate that a theory is satisfiable. However, in the axiomatization of domain theories, we need a complete characterization of the possible models. For example, since we are considering the domain of computer vision, to show that a theory is satisfiable, we need only specify an image and scene which together with the axioms are satisfied by some structure. The problem with this approach is that we run the risk of having demonstrated satisfiability only for some restricted class of images, scenes, or surfaces. For example, a theory of activities that supports scheduling may be shown to be consistent by constructing a satisfying interpretation, but the interpretation may require that resources cannot be shared by multiple activities; although such a model may be adequate for such scenes, it would in no way be general enough for our purposes. We want to propose a comprehensive theory of activities, so we need to explicitly characterize the classes of activities, timepoints, objects, and other assumptions which are guaranteed to be satisfied by the specified structures.

The purpose of the Axiomatizability Theorem is to demonstrate that there do not exist any unintended models of the theory, that is, any models which are not specified in the class of structures.

2.4. **The Ontological Stance.** When building translators, we are faced with the additional challenge that almost no application has an explicitly axiomatized ontology. However, we can characterize a software application as if it were an inference system with an axiomatized ontology, and use this ontology to predict the set of sentences that the inference system decides to be satisfiable. This is the Ontological Stance, and is analogous to the intentional stance (Dennet 87), which is the strategy of interpreting the behavior of an entity by treating it as if it were a rational agent who performs activities in accordance with some set of intentional constraints.

The Ontological Stance is an operational characterization of the set of intended models for the application's terminology. In this sense, it should be treated as a semantic constraint on the application. It does not postulate a specific set of axioms, but rather a set of intended models.

Ontological Stance: :

Given an application A, there exists a class of models $\mathcal{M}^A$ such that any sentence Φ is decided by A to be satisfiable iff there exists $\mathcal{M} \in \mathcal{M}^A$ such that

$$\mathcal{M} \models \Phi$$

The ontology of an application can be evaluated with respect to this set of intended models, just as the methodology proposed in the preceding section can be used to evaluate the PSL Ontology. Let us say that an application ontology is weak if it does not axiomatize the class of intended models for the application as postulated by the Ontological Stance. If an application A has a weak ontology T_A, then there exists a sentence Φ such that either

- A decides Φ to be satisfiable but there does not exist a model $\mathcal{M}_1$ of T_A such that $\mathcal{M}_1 \models \Phi$,
- or there exists a model $\mathcal{M}_2$ of T_A such that $\mathcal{M}_2 \models \Phi$, but A decides Φ to be unsatisfiable.

In either case, the application will not be interoperable with other applications.

2.5. **Weak Translation Theorem.** In this section, we demonstrate how the notion of sharability is equivalent to model isomorphism.

Conformance Hypothesis: :

Every structure that is a model of the application ontology is interpretable in a model of a core theory that is an extension of PSL-Core.

This is a rather strong hypothesis, since it entails that all application ontologies are sharable with PSL-Core. However, it plays a role that is analogous to the SatAx Criterion for PSL; the Conformance Hypothesis is one way of ensuring that the set of intended models for the application (as postulated by the Ontological Stance) have been axiomatized.

We will adopt the following approach from [6]. Let $\mathcal{N}$ be a structure in $\mathcal{L}_0$ and let $\mathcal{M}$ be a structure in $\mathcal{L}$. We say that $\mathcal{N}$ is definable in $\mathcal{M}$ iff we can find a definable subset X of M^n and we can interpret the symbols of $\mathcal{L}_0$ as definable subsets and functions on X so that the resulting structure in $\mathcal{L}_0$ is isomorphic to $\mathcal{N}$.

Definition 7. *Let $\mathcal{M}$ be a structure in a language $\mathcal{L}$. For each definable equivalence relation E_i on M^n, let $S_{E_i} = M^n/E_i$ and let $\pi_{E_i} : M^n \rightarrow M^n/E_i$ be the quotient map. The language $\mathcal{L}^{eq}$ is the expansion of $\mathcal{L}$ in which we add a new relation symbol to $\mathcal{L}$ for every equivalence relation.*

$\mathcal{M}^{eq}$ is the structure whose underlying set is the disjoint union of M and all of the S_{E_i}.

By Theorem 5.3.1 of [4], a structure $\mathcal{N}$ is interpretable in a structure $\mathcal{M}$ iff $\mathcal{N}$ is isomorphic to a relativized reduct of a definitional expansion of $\mathcal{M}^{eq}$.

Definition 8. *Let λ_A be the expansion of the nonlogical lexicon of PSL by the introduction of the nonlogical lexicon of the application ontology T_A.*

A weak translation axiom for a relation $p \in \lambda_A$ is a sentence of the form

$$(\forall \overline{x})\, p(\overline{x}) \equiv \Phi(\overline{x})$$

where $\Phi(\overline{x}) \in \mathcal{L}(T_{psl})$.

Conjecture 1. Weak Translation Theorem: *If the application ontology satisfies the Conformance Hypothesis, and the PSL Ontology satisfies the SatAx Criterion, then there exists a set of weak translation axioms T_{weak} for the application ontology such that $T_{psl} \cup T_{weak}$ is interoperable with T_A.*

Conjecture 2. *If the application ontology satisfies the Conformance Hypothesis, and the PSL Ontology satisfies the SatAx Criterion, then there exists a set of weak translation axioms for the application ontology that axiomatize the models of the ontology up to elementary equivalence.*

Without the Conformance Hypothesis, there may exist models of T_A that are not isomorphic to models of PSL, in which case T_A is not sharable. Without the SatAx Criterion, we cannot guarantee that weak translation axiomatizes the models of T_A. In this sense, the Conformance Hypothesis also imposes conditions on the PSL Ontology, which must be rich enough to axiomatize the application ontology.

Note that the Conformance Hypothesis supports interoperability even for applications that have weak ontologies. The Weak Translation Theorem does not require that the models of the application ontology are axiomatized, but simply that they are isomorphic to models of the PSL Ontology. In this sense, the weak translation can also serve as a means of axiomatizing the intended models of weak ontologies up to elementary equivalence.

3. Supporting Self-Integration

3.1. **Classification and Invariants.** The terminology within the definitional extensions intuitively corresponds to classes of activities and objects. Within the PSL Ontology, the terminology arises from the classification of the models of the core theories with respect to sets of invariants.

Invariants are properties of models that are preserved by isomorphism. A set of invariants is complete for a class of structures if and only if it can be used to classify the structures up to isomorphism. For example, a finite abelian group can be classified up to isomorphism by the subgroups whose orders are factors of the group's order. In general, it is not possible to formulate a complete set of invariants; for example, there is no known set of invariants that can be used to classify graphs up to isomorphism. However, even without a complete set, invariants can still be used to provide a classification of the models of a core theory in PSL.

Classification Criterion: :

The set of models for the core theories of PSL are partitioned into equivalence classes defined with respect to the set of invariants of the models.

Invariants can also be used as the basis for decidable model isomorphism theorems. For example, although the theory of groups is in general undecidable, the theory of abelian groups is decidable, and this result utilizes the invariants in the classification of abelian groups.

3.2. **The Role of Definitional Extensions.** Although we may not be able to find a complete set of invariants for models of the PSL Ontology, we can specify the equivalence classes associated with any given set of invariants.

Definitional Extension Criterion: :

Each equivalence class in the classification of PSL models can be axiomatized within a definitional extension of PSL:

for each equivalence class $\mathcal{C}$, there exists a unary relation p in a definitional extension, such that

$$\langle \mathbf{x} \rangle \in \mathbf{p} \Leftrightarrow \mathbf{x} \in \mathcal{C}$$

Thus the definition of a relation is satisfied by every element in the equivalence class, and every element in the equivalence class satisfies the definition of the relation.

In particular, each definitional extension in the PSL Ontology is associated with a unique invariant; the different classes of activities or objects that are defined in an extension correspond to different properties of the invariant.

The Definitional Extension Criterion is analogous to the SatAx Criterion insofar as both are used to evaluate the adequacy of the axiomatization of some extension within the PSL Ontology. Whereas the SatAx Criterion evaluates the adequacy of a core theory with respect to some class of structures, the Definitional Extension Criterion evaluates the adequacy of a definitional extension to the classification of the models of some set of core theories.

3.3. **Strong Translation Theorem.**

Coverage Hypothesis: :

The set of models for an application ontology can be partitioned into equivalence classes defined with respect to the set of invariants of the models.

By the Conformance Hypothesis, such a set of invariants can be used to partition the set of models of PSL core theories into equivalence classes.

Definition 9. *Let λ_A be the expansion of the nonlogical lexicon of PSL by the introduction of the nonlogical lexicon of the application ontology T_A.*

A translation definition for a relation $p \in \lambda_A$ is a sentence of the form

$$(\forall \overline{x})\, p(\overline{x}) \equiv \Phi(\overline{x})$$

where $\Phi(\overline{x})$ contains only relation symbols from definitional extensions of PSL.

Conjecture 3. Strong Translation Theorem: *If an application ontology satisfies the Conformance and Coverage Hypotheses and the PSL Ontology satisfies the SatAx, Classification, and Definitional Extension Criteria, then there exists a set of translation definitions T_{strong} for the application ontology such that $T_{psl} \cup T_{strong}$ is interoperable with T_A.*

We can use the translation definitions to assign a profile to every class of elements in the application ontology. Such a profile consists of the set of equivalence classes for each invariant in the classification theorem.

Conjecture 4. *If an application ontology satisfies the Conformance and Coverage Hypotheses and the PSL Ontology satisfies the SatAx, Classification, and Definitional Extension Criteria, then the translation definitions for the application ontology axiomatize the models of the ontology up to elementary equivalence.*

The Coverage Hypothesis for definitional extensions is analogous to the Conformance Hypothesis for core theories. If the set of invariants for classifying models of the application ontology is different from the set of invariants used in the classification of models of PSL, then we cannot use translation definitions.

4. Summary

This paper has described how model-theoretic invariants of an ontology can be used to specify semantic mappings translation definitions between application ontologies and an interlingua. In particular, examples have been presented using the Process Specification Language (PSL) ontology as the neutral medium in integration.

The sets of models for the core theories of PSL are partitioned into equivalence classes defined with respect to the invariants of the models. Each equivalence class in the classification of PSL models is axiomatized using a definitional extension of PSL. Software tools based on these invariants and definitional extensions support semi-automatic generation of semantic mappings between an application ontology and the PSL Ontology. This approach can be generalized to other ontologies by specifying the invariants for the models of the axiomatizations.

References

[1] Bouquet, P., Serafini, L., Zanobini, S., and Benerecetti, M. (2003) An Algorithm for Semantic Coordination. *Semantic Integration Workshop, International Semantic Web Conference 2003*, 21–26.

[2] Gruninger, M. (2003) A Guide to the Ontology of the Process Specification Language, in *Handbook on Ontologies in Information Systems*, R. Studer and S. Staab (eds.). Springer-Verlag.

[3] Gruninger, M. and Menzel, C. (2003) Process Specification Language: Principles and Applications, *AI Magazine*, 24:63-74.

[4] Hodges, W. (1993) *Model Theory*. Cambridge University Press.

[5] Mahdavan, J., Bernstein, P., and Rahm, E. (2001) Generic Schema Matching with Cupid, *Proc. 27th VLDB Conference*, 49–58.

[6] Marker, D. (2000) *Model Theory: An Introduction*. Springer-Verlag.

[7] Noy, N. and Musen, M. (2000) PROMPT: Algorithm and tool for automated ontology merging and alignment, Proceedings of AAAI-2000.

[8] Stuckenschmidt, H. and Visser, U. (2000) Semantic Translation Based on Approximate Reclassification. In Proceedings of the Seventh International Conference on Knowledge Representation and Reasoning, Breckenridge, Colorado.

National Institute of Standards and Technology, 100 Bureau Drive, Gaithersburg, MD 20899-8260, gruning@cme.nist.gov