Semantic Characterization of Ontologies

Michael Grüninger
Department of Industrial Engineering
University of Toronto
gruninger@eil.utoronto.ca

Abstract

Using a semantic definition of ontologies, we provide a characterization of ontologies with respect to reusability, merging, and integration. This definition also provides a model-theoretic approach to distinguishing ontologies from the domain theories which use them, and the relationships among ontologies within an ontology library.

Introduction

Ontological engineering was born with the promise of reusability, integration, and interoperability. Of increasing importance are the problems merging ontologies from different domains and translating among multiple ontologies from the same domain. However, to a large degree, we have not yet delivered these promised benefits. What we lack is a framework within which people can develop and share reusable ontologies.

We need standards for specifying ontologies, in such a way that reusability, merging, and integration can be formally and unambiguously evaluated. Such a standard would involve the specification of the language of the ontology, along with an axiomatization (model theory and proof theory). The model theory provides a rigorous mathematical characterization of the terminology of the ontology. The proof theory provides axioms for the interpretation of terms in the ontology.

The problem is that the meaning of the terminology for many ontologies are in people's heads. Any ideas that are implicit are a possible source of ambiguity and confusion, and are barriers to reusability, merging, and integration. By specifying the model theory and proof theory of the ontology, we can make this explicit. Different interpretations for the terminology can be given, but one of these will be the intended interpretation that guides the development of the axioms. The axiomatization allows a characterization of these interpretations. We can reason about the meaning of the terminology of the ontologies using the models of the axioms.

With this approach, the focus of the ontology is not only on the terms themselves, but also on their definitions. We can include an infinite set of terms in our ontology, but they can only be shared if we agree on their definitions. It is the definitions which are being shared, not simply the terms themselves.

Once the model theory and proof theory of the ontology are in place, we can identify the semantic criteria that an ontology must satisfy to enable the following:

- reusability
- merging (multiple ontologies from the same domain)
- structuring libraries of ontologies
- integration (ontologies from different domains)

A formal characterization of these notions would involve a specification of the necessary conditions for an ontology to satisfy the semantic criteria.

The Structure of Ontologies

An ontology is specified by a set of axioms in some formal language. However, this is not an amorphous set; some axioms are conservative definitions, while some are sentences specifying constraints over objects in the domain. This section presents an architecture which attempts to make this structure explicit.

Foundational Theories and Definitions

We first consider the distinction between axioms for defined terminology and axioms for terminology which is intuitively primitive or undefined. [1]

Definition 1 *A sentence Φ is a definition with respect to a theory T (equivalently, Φ is a definition in $\mathcal{L}(T)$) iff for any sentence $\Phi' \in \mathcal{L}(T)$,*

$$T \cup \{\Phi\} \models \Phi' \ iff \ T \models \Phi'$$

In other words, a sentence is a definition with respect to a theory T if any sentence that can be proven using the definition can also be proven using T alone; intuitively, the sentence does not add any new entailments to our theory. All models of the definition which are

[1] Throughout this paper, we will restrict ourselves to ontologies with first-order axiomatizations. We will be using the ontologies in the Appendix as examples; full axiomatizations for these ontologies can be found at *http://www.ie.utoronto.ca/EIL/tove/toveont.html*

restricted to $\mathcal{L}(T)$ (the language of T) are models of the theory T.

For example, the sentences in the Appendix are definitions for the relations `(achieved ?f ?s)`, `(falsified ?f ?s)` with respect to the situation calculus as axiomatized by theory $T_{sitcalc}$ in the Appendix.

Definition 2 *A set of sentences T_{def} is a definitional extension with respect to a theory T iff every sentence in T_{def} is a definition with respect to T or a sentence entailed by T.*

The sentences in the TOVE Fluent Module T^{def}_{fluent} in the Appendix form a definitional extension with respect to the foundational theory $T_{sitcalc}$.

Definition 3 *$\mathcal{L}^{def}(T_{def})$ is the set of relations and functions with definitions in the definitional extension T_{def}. We will refer to this as the lexicon of T_{def}.*

For example, the lexicon of T^{fluent}_{def} is:
`(achieved ?f ?s), (falsified ?f ?s)`

Note that the theory T which is used to define terminology in the definitional extension may itself contain definitions. However, we will want to concentrate on those theories with which we can provide definitions for all terminology.

Definition 4 *A foundational theory is a theory which is not a definitional extension of any other theory.*

Thus, a theory is a foundational theory iff it contains no definitions. Since none of the relations, functions, or constants in a foundational theory have definitions, their intended interpretations are defined using only the axioms in the foundational theories.

Examples of foundational theories are $T_{sitcalc}$ and T_{Do} in the Appendix, which are axiomatizations of situation calculus and complex actions.

Domain Theories

To this point, the notions of foundational theory or definitional extension are not in themselves new, and there is nothing to distinguish them from an arbitrary first-order theory. However, ontologies are different from arbitrary theories in that we intuitively think of ontologies as being the reusable portion of domain theories. Of course, this begs the question of defining domain theories, and it raises the perennial debate of the difference between ontologies and knowledge bases.

The reusability of an ontology arises because it is invariant across the set of domain theories which use the ontology. Intuitively, the ontology must satisfy two criteria:

- Each domain theory has its own language which is an expansion of the language of the ontology.
- Each model of a domain theory is a model of the ontology, such that the language of the model is an expansion of the language of the ontology.

First of all, there is a distinction between the language of the foundational theories and definitional extensions on the one hand, and the language of the domain theories on the other. For example, in a process ontology, there will be axiomatizations of such concepts as *activity*, *occurrence*, and *state*; any domain theory using this ontology will also introduce domain-specific terminology such as *pickup*, $fabricate\_part$ which denote activities. Thus, the language of the domain theory is the language of the ontology expanded to include the domain-specific lexicon of the particular domain theory.

Definition 5 *Given a foundational theory $T_{foundation}$ and a set of definitional extensions T_{def}, a domain-specific lexicon (denoted $\mathcal{L}^{dsl}_i$), is a set of terms disjoint from $\mathcal{L}(T_{foundation} \cup T_{def})$ which denote elements in some domain.*

The second criterion of reusability is the relationship between the interpretation of the domain-specific lexicon and the models of the foundational theories of the ontology. For example, the domain theories are not arbitrary sets of sentences in the language of the domain-specific lexicon, but are in some way constrained by the axiomatization of the foundational theories.

Types and Models In order to provide a such semantic characterization of the relationship between ontologies and domain theories, we resort to the notion of types from model theory ([Hodges 93], [Goldstern and Judah 95]). Types describe a model of a theory from the point of view of a single element or a finite set of elements.

Definition 6 *Let $\mathcal{M}$ be a model for a language $\mathcal{L}$. The type of an element $a \in M$ is defined as*
$type_{\mathcal{M}}(a) := \{\phi : \phi$ is a formula of $\mathcal{L}, \mathcal{M} \models \phi \}$

An n-type for a theory T is a set $\Phi(x_1, ..., x_n)$ of formulae, such that for some model $\mathcal{M}$ of T, and some n-tuple $\overline{a}$ of elements of $\mathcal{M}$, we have $\mathcal{M} \models \phi(\overline{a})$ for all ϕ in Φ.

If t is an n-type, then a model $\mathcal{M}$ realizes t iff there are $a_1, ..., a_n \in M$ such that

$$\mathcal{M} \models t(a_1, ..., a_n)$$

Informally, the type for an element in a model is a set of formula which are satisfied by some set of elements in the model. An n-type for a theory is a consistent set of formulae (each of which has n free variables) which is satisfied by a model of the theory.

Models of Domain Theories and Ontologies The intuition is that the models of domain theories can be specified using the models of the underlying foundational theories and definitional extensions. To formalize this intuition, we first define domain theories using the types for the foundational theories, and then show that this gives a complete characterization of the models of domain theories as models of the foundational theories. Thus, the intuition of the reusability of an ontology is captured in its set of models.

Definition 7 *Let $T_{foundation}$ be a foundational theory and let T_{def} be a definitional extension of $T_{foundation}$.*

A domain theory for $T_{foundation} \cup T_{def}$ is a boolean combination of n-types for $T_{foundation} \cup T_{def}$ which are realized in some model of $T_{foundation} \cup T_{def}$ with the language $\mathcal{L}(T_{foundation} \cup T_{def}) \cup \mathcal{L}_{dsl}$, for some domain-specific lexicon $\mathcal{L}_{dsl}$.

Note that the models of the domain theories satisfy the axioms of the ontology in $T_{foundation} \cup T_{def}$, but that the language of these models includes the domain-specific lexicon. However, the theory does not include arbitrary sentences in the domain-specific lexicon; it is restricted to sentences which are types for the ontology and which are realized in some model of the ontology. Thus, domain theories are formulae satisfied by elements in different models of $T_{foundation} \cup T_{def}$. We will use the notation $\tau^n(T_{foundation} \cup T_{def})$ to refer to the set of n-types for the theory $T_{foundation} \cup T_{def}$ in $\mathcal{L}(T_{foundation} \cup T_{def}) \cup \mathcal{L}_{dsl}$, for some domain-specific lexicon $\mathcal{L}_{dsl}$.

For the foundational theory $T_{sitcalc}$, types are precondition and effect axioms, and domain theories are equivalent to basic action theories [Reiter 91]. For the foundational theory T_{Do}, types are complex action definitions, which specify the constraints under which subactions occur.

We will now show how the notion of types for the ontology formalizes the intuitions behind the criteria for reusability.

Lemma 1 *Let $T_{foundation}$ be a foundational theory and let T_{def} be a definitional extension of $T_{foundation}$.*

The models of a domain theory for $T_{foundation} \cup T_{def}$ form a subset of the models of $T_{foundation} \cup T_{def}$ with the language $\mathcal{L}(T_{foundation} \cup T_{def}) \cup \mathcal{L}_{dsl}$, for some domain-specific lexicon $\mathcal{L}_{dsl}$.

Lemma 2 *Let $T_{foundation}$ be a foundational theory, let T_{def} be a definitional extension of $T_{foundation}$, and let $\mathcal{L}_{dsl}$ be a domain-specific lexicon.*

Every consistent subset of models of $T_{foundation} \cup T_{def}$ with the language $\mathcal{L}(T_{foundation} \cup T_{def}) \cup \mathcal{L}_{dsl}$ is equivalent to a set of models for some domain theory for $T_{foundation} \cup T_{def}$.

The next theorem follows easily from these two lemmas:

Theorem 1 *Let $T_{foundation}$ be a foundational theory, let T_{def} be a definitional extension of $T_{foundation}$, and let $\mathcal{L}_{dsl}$ be a domain-specific lexicon.*

Any sentence in a domain theory for $T_{foundation} \cup T_{def}$ is consistent iff it is satisfied by some set of models of $T_{foundation} \cup T_{def}$ with the language $\mathcal{L}(T_{foundation} \cup T_{def}) \cup \mathcal{L}_{dsl}$.

Corollary 1 *Let $T_{foundation}$ be a foundational theory and let T_{def} be a definitional extension of $T_{foundation}$.*

Any sentence consistent with $T_{foundation} \cup T_{def}$ is entailed by some domain theory for $T_{foundation} \cup T_{def}$.

Since domain theories are formulae satisfied by elements in different models of $T_{foundation} \cup T_{def}$, the types of the ontology characterize the domain theories for the foundational theories and definitional extensions of the ontology.

Using Ontologies With this characterization of ontologies, we can define what it means for a theory to use an ontology.

A theory (which can be a foundational theory, definitional extension, or a domain theory) uses a definitional extension iff the language of the theory uses relations and functions with definitions in T_{def}.

Definition 8 *A theory T uses a definitional extension T_{def} iff $\mathcal{L}^{def}(T_{def}) \subseteq \mathcal{L}(T)$.*

A theory uses an ontology iff it uses the definitional extensions of the theory and the sentences in the theory satisfy the types for the ontology (i.e. the theory is a domain theory for the ontology).

Definition 9 *Let $T_{foundation}$ be a foundational theory, let T_{def} be a set of definitions with respect to $T_{foundation}$, and let $\tau^n(T_{foundation} \cup T_{def})$ be a set of n-types for $T_{foundation} \cup T_{def}$.*

Let T_{onto} be the set of sentences in $T_{foundation} \cup T_{def} \cup \tau^n(T_{foundation} \cup T_{def})$.

A theory T is a T_{onto}-theory iff T is a domain theory for $T_{foundation} \cup T_{def}$ and T uses T_{def}.

Merging Ontologies

We can also use this characterization of ontologies to define various relationships among definitional extensions necessary for characterizing the semantic criteria for merging ontologies. Two ontologies can be merged if they can be embedded into a third ontology; that is, every model of each ontology can be embedded into a model of the merged ontology.

In this paper we are restricting ourselves to characterizing merging ontologys using only the lexicons of the definitional extensions rather than directly using the terminology of the foundational theories. In this case, merging two ontologies is equivalent to showing that there exists a third ontology which is able to provide definitions for the lexicon of the two embedded ontologies.

A sentence can be expressed using the definitional extension T_{def} iff there is a sentence which uses only the terminology of T_{def}, and which we can prove is equivalent to a sentence which uses terminology from the foundational theory. In addition, the axioms in the foundational theory alone are used to prove this equivalence.

Definition 10 *Let $T_{foundation}$ be a foundational theory and let T_{def} be a set of definitions with respect to $T_{foundation}$.*

Let Φ be a definition in $\mathcal{L}(T)$.

Φ can be expressed using T_{def} iff there exists a sentence $\Phi^{def} \in \mathcal{L}^{def}(T_{def})$ such that

$$T_{foundation} \cup T_{def} \models \Phi \equiv \Phi^{def}$$

The following example is taken from PSL translation definitions between the PSL Ontology [Schlenof et al. 99] and an ontology for Ilog Schedule:

```
(forall (?occ1 ?occ2 ?d)
(<=> (endsAfterEnd ?occ1 ?occ2 ?d)
     (exists (?occ3)
       (and (subaction_occurrence ?occ1 ?occ3)
            (subaction_occurrence ?occ2 ?occ3)
            (after_end_delay ?occ1 ?occ2 ?occ3 ?d]
```

In this case, the relation *endsAfterEnd* is part of the Ilog Schedule Ontology, while the relations *subaction_occurrence* and $after\_end\_delay$ are part of a definitional extension within the PSL Ontology. The sentence demonstrates that *endsAfterEnd* can be expressed using the PSL Ontology.

It is sometimes not possible to show that a definition from one ontology can be expressed in another ontology; that is, we cannot write a sentence using the first ontology which is logically equivalent to the definition in another ontology. A weaker approach is to show that any element in a class defined by a sentence written using terminology in the definitional extension T_{def} realizes types for another ontology.

Definition 11 *Let $T_{foundation}$ be a foundational theory, let T_{def} be a set of definitions with respect to $T_{foundation}$, and let $\tau^n(T_{foundation} \cup T_{def})$ be the types for $T_{foundation} \cup T_{def}$.*

Let $\Phi(\overline{x})$ be a definition in $\mathcal{L}(T)$.

$\Phi(\overline{x})$ can be weakly expressed using $T_{def} \cup \tau^n(T_{foundation} \cup T_{def})$ iff for any model $\mathcal{M}$ of $T_{foundation} \cup T_{def}$ and any tuple of elements $\overline{a} \in M$, if

$$\mathcal{M} \models \Phi(\overline{a})$$

then the type of $\overline{a}$ in $\mathcal{M}$ is a subset of $\tau^n(T_{foundation} \cup T_{def})$.

For example, one ontology may have a definition for the class of deterministic activities. The other ontology, although it may not be able to express the definition of *deterministic*, can guarantee that for any deterministic activity, there exists a type for the ontology which is equivalent to the activity definition. Thus, any domain theory using one ontology can be translated to a logically equivalent domain theory using the other ontology.

Relationships among Ontologies

We can now apply these ideas to define various relationships among the foundational theories and definitional extensions of an ontology.

For foundational theories, we can simply use standard notions from mathematical logic ([Goldstern and Judah 93], [Hodges 93]). For example, two theories may be mutually inconsistent or independent of one another. A particularly useful relationship is the following:

Definition 12 *Let $\mathcal{L}, \mathcal{L}^+$ be first-order languages with $\mathcal{L} \subseteq \mathcal{L}^+$, and let T, T^+ be theories in $\mathcal{L}, \mathcal{L}^+$. Then T^+ is a conservative extension of T if for any sentence ϕ of $\mathcal{L}$,*

$$T \models \phi$$

iff

$$T^+ \models \phi$$

An example of this is the foundational theory $T_{sitcalc} \cup T_{Do}$ in the Appendix, which is a conservative extension of $T_{sitcalc}$ in which complex actions are axiomatized. Notice that $\mathcal{L}(T_{Do}) = \mathcal{L}(T_{sitcalc}) \cup \{Do, atomic, subaction\}$, so that new relations are added to the language.

This relationship is particularly useful in defining a partial ordering over the set of foundational theories within an ontology library. If $T_1 \subset_f T_2$ denotes that T_2 is a conservative extension of T_1, then we can show that any linear ordering of theories in $\subset_f$ is a consistent theory iff each theory in the ordering is consistent. We will use this fact below as a means of structuring the ontologies in a library.

Since definitional extensions are already sets of conservative extensions, we cannot order them using logical entailment; rather, we can define an ordering relation over a set of definitional extensions based on their relative expressiveness:

Definition 13 *A definitional extension T^1_{def} is as expressive as the definitional extension T^2_{def} (written $T^1_{def} \preceq T^2_{def}$) iff every sentence in $\mathcal{L}^{def}(T^1_{def})$ can be expressed using T^2_{def}.*

A weaker relation simply considers an extension of the language of the definitional extension:

Definition 14 *A definitional extension T^1_{def} is a library extension of the definitional extension T^2_{def} iff $\mathcal{L}(T^2_{def}) \subseteq \mathcal{L}(T^1_{def})$.*

Note that for any definitional extension T_{def} with respect to a theory T, we have $\mathcal{L}(T) \subseteq T_{def}$, since new terminology is introduced in the extension.

Different definitional extensions may have equivalent expressiveness; thus we may have different domain theories which nevertheless are satisfied by the same models of the foundational theories.

Definition 15 *A definitional extension T^1_{def} is a conservative library extension of the definitional extension T^2_{def} iff $\mathcal{L}(T^2_{def}) \subseteq \mathcal{L}(T^1_{def})$ and $T^1_{def} \preceq T^2_{def}$, $T^2_{def} \preceq T^1_{def}$.*

Conservative library extensions provide a means for adopting one definitional extension as a standard terminology for a domain, but allow applications to use their own terminology without sacrificing expressiveness or preventing sharability.

Building an Ontology Library

If the set of axioms within an ontology are partitioned into a set of modules within an ontology library, what does it mean to say that a module in the library requires

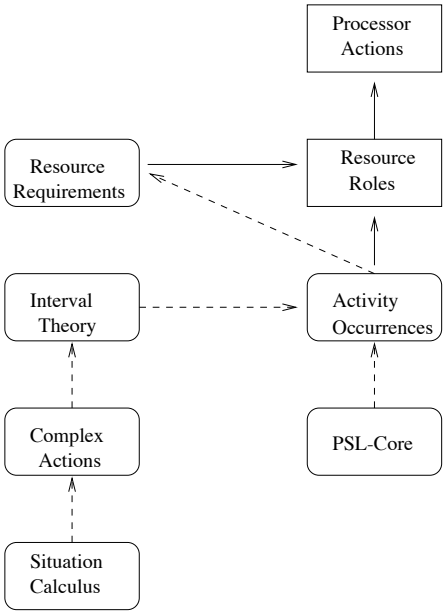

Figure 1: Part of the PSL Ontology Library

a foundational theory or a definitional extension? For example, consider the set of modules in Figure 1, in which foundational theories are depicted by oval boxes and definitional extensions are depicted by rectangular boxes. The dashed and solid lines indicate which modules are required by other modules within the library. A dashed line indicates that a foundational theory requires some other module, while a solid line indicates that a definitional extension requires another module in the library.

The complete set of axioms for a module which is a foundational theory is the union of the set of foundational theories required by the module. In this case, the module is a conservative extension of each of the foundational theories which it requires. For example, consider the Interval Theory module. The dashed line indicates that this theory requires the foundational theory Complex Actions. The complete set of axioms for Interval Theory is the union of the set of axioms in the module itself in addition to the axioms in the modules Complex Actions and Situation Calculus, so that it is a conservative extension of both Situation Calculus and Complex Actions.

If a foundational theory requires definitional extensions, then the axioms of the module use terminology defined in the definitional extensions. Note that this means that the foundational theories required by the definitional extension are also required by the module.

Suppose the module is a definitional extension. The set of foundational theories required by the module is characterized by the following: the lexicon of the module has conservative definitions with respect to the union of the theories required by the module. For example, the Resource Roles module contains definitions which are conservative with respect to the union of all foundational theories in Figure 1.

The set of definitional extensions required by another definitional extension is characterized by the following: the definitions for the lexicon of the given module includes a lexicon whose definitions are found in the listed extensions. Thus, a definitional extension requires another definitional extension if it is a library extension. For example, the Processor Actions module is a library extension of the Resource Roles module.

Integrating Domain-Specific Ontologies

One major problem deals with the integration of generic and domain-specific ontologies. With generic ontologies, there may be relations and functions which are part of the domain-specific lexicon of a domain theory, while in another ontology, these same relations and functions will be axiomatized as part of the foundational theories. This requires a characterization of the relationship of the set of types for the two ontologies for which $\mathcal{L}_{dsl}$, the language of the domain theories for both ontologies, is not equivalent.

An important case is where the foundational theories of one ontology are an extension of the foundational theories of the other. In particular, this considers ontologies where the foundational theories are extended by axiomatizing relations, functions, and constants in $\mathcal{L}_{dsl}$, and the set of types for the ontologies is unchanged.

For example, consider the relationship between a generic Product Ontology and its domain-specific extensions. In the generic ontology, each product has a product definition theory, which specifies properties of the product and how the product is assembled from its components. Such a product definition is a type for the Product Ontology and the lexicon of these theories is $\mathcal{L}_{dsl}$. The extended ontology would axiomatize the relations and functions in this lexicon, such as *connected* or *assembly*. Note that the set of types is unchanged – each product still has a product definition theory; what has been added is the axiomatization of new foundational theories. Thus, the set of domain theories for the ontology is unchanged, but the set of models of the domain theories is now determined by the new extended foundational theories. We therefore need a formal characterization of this relationship between foundational theories and their extension for the axiomatization of $\mathcal{L}_{dsl}$.

References

Goldstern, M. and Judah, H. (1995) *The Incompleteness Phenomenon: A New Course in Mathematical Logic.* A.K. Peters

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

Reiter, R. (1991). The frame problem in the situation calculus: a simple solution (sometimes) and a

completeness result for goal regression. In Vladimir Lifschitz, editor, *Artificial Intelligence and Mathematical Theory of Computation: Papers in Honor of John McCarthy*, pages 418-440. Academic Press, San Diego.

Schlenof, C., Grüninger, M., Ciocoiu, M. (1999), The essence of process specification, submitted to *Special Issue on Modeling and Simulation of Manufacturing Systems (International Society for Computer Simulation)*

Appendix: Examples of Ontologies

Situation Calculus

```
(define-theory Basic_Situation_Calculus
:axioms
```

The initial situation S_0 is not the successor of any situation.

```
(forall (?a ?s)
        (not (= S_0 (do ?a ?s))))
```

Situations form a tree.

```
(forall (?a1 ?a2 ?s1 ?s2)
        (=>     (= (do ?a1 ?s1) (do ?a2 ?s2))
                (and (= ?a1 ?a2)
(= ?s1 ?s2))))
```

No situation is earlier than the initial situation S_0.

```
(forall (?s)
        (not (< ?s S_0))
```

A situation ?s1 is earlier than the successor situation of ?s2 iff the action is poss in ?s2 and situation ?s2 is later than ?s1.

```
(forall (?a ?s1 ?s2)
        (<=>    (< ?s1 (do ?a ?s2))
                (and    (poss ?a ?s2)
                        (leq ?s1 ?s2))))
```

If two situations agree on state (which fluents hold in the situation), then they agree on the extension of poss.

```
(forall (?s1 ?s2)
        (=>     (state_equiv ?s1 ?s2)
                (poss_equiv ?s1 ?s2)))
```

If two situations agree on state (which fluents hold in the situation), then they agree on state for all successor situations in which the same action occurs.

```
(forall (?s1 ?s2)
        (=>     (state_equiv ?s1 ?s2)
                (effects_equiv ?s1 ?s2))))
```

TOVE Fluent Module

A fluent ?f is achieved in a situation ?s iff it does not hold in ?s but it does hold in the successor situation.

```
(defrelation achieved (?f ?s) :=
(exists (?a)
  (and (holds ?f (do ?a ?s))
       (not (holds ?f ?s)))))
```

A fluent ?f is falsified in a situation ?s iff it holds in ?s but it does not hold in the successor situation.

```
(defrelation falsified (?f ?s) :=
(exists (?a)
   (and (not (holds ?f (do ?a ?s)))
        (holds ?f ?s))))
```

This relation defines the interval over which a fluent holds – the fluent ?f is achieved in ?s1, falsified in ?s2, and it holds for all situations between ?s1 and ?s2.

```
(defrelation fluent_interval (?f ?s1 ?s2) :=
(and (achieved ?f ?s1)
     (falsified ?f ?s2)
     (< ?s1 ?s2)
     (forall (?s)
        (=> (and  (?s1 ?s)
                  (?s ?s2))
            (holds ?f ?s)))))
```

Complex Actions

```
(define-theory Complex-Activity
```

If an activity occurs between two situations ?s1 and ?s2, then there exists an atomic subactivity which occurs in ?s1, and ?s2 is equal to the occurrence of an atomic subactivity.

```
(forall (?a ?s1 ?s2)
(=> (Do ?a ?s1 ?s2)
    (and (exists (?a1 ?s3)
           (and (atomic ?a1)
                (subaction ?a1 ?a)
                (= ?s2 (do ?a1 ?s3))))
         (exists (?a2 ?s4)
           (and (atomic ?a2)
                (subaction ?a2 ?a)
                (= ?s4 (do ?a2 ?s1))))))
```

If an activity occurs between two situations ?s1 and ?s2, then there exist atomic subactiviti es which occur on legal branches between ?s1 and ?s2.

```
(forall (?a ?s1 ?s2)
(=> (Do ?a ?s1 ?s2)
    (exists (?a1 ?s3)
       (and (atomic ?a1)
            (subaction ?a1 ?a)
            (leq S_0 ?s1)
            (leq ?s1 (do ?a1 ?s3))
            (leq (do ?a1 ?s3) ?s2)))))
```

For any atomic activity which occurs along a branch, there exists an activity containing the atomic activity and which also occurs along the branch (that is, the atomic activity occurs during the activity that contains it.)

```
(forall (?a1 ?s1 ?s2 ?s3)
(=>  (and    (< ?s1 (do ?a1 ?s3))
             (< (do ?a1 ?s3) ?s2))
     (exists (?a)
        (and  (subaction ?a1 ?a)
              (Do ?a ?s1 ?s2)))))
```