Sumo2loom Documentation

David Flater
NIST
100 Bureau Drive, Stop 8264
Gaithersburg, MD 20899-8264

dflater@nist.gov

Abstract

In 2002, a project at the U.S. National Institute of Standards and Technology (NIST) called for the use of an upper ontology in conjunction with an inference engine. After a review of available ontologies and inference engines it was decided to translate SUMO into a form that could be used by version 4.0 of the LOOM inference engine. The translating program Sumo2loom was developed for this purpose. This document describes Baseline 8 of Sumo2loom with its input, output, and usage.

Keywords: KIF, logic, ontology, software, translation

1 Introduction

In 2002, a project at the U.S. National Institute of Standards and Technology (NIST) called for the use of an upper ontology in conjunction with an inference engine. At the time, the leading, publicly available candidates for upper ontology were the Suggested Upper Merged Ontology (SUMO) [1][2] and the OpenCyc ontology [3]. The authoritative version of the former was in a condensed version of the Knowledge Interchange Format (KIF) [4] called SUO-KIF[1] [5]; the authoritative version of the latter was embedded within the first public release of the OpenCyc inference engine, version 0.6.0b.

After a review of available ontologies and inference engines it was decided to translate SUMO into a form that could be used by version 4.0 of the LOOM inference engine [6]. The translating program Sumo2loom was developed for this purpose.

This document describes Baseline 8 of Sumo2loom with its input, output, and usage. The entire package can be obtained by sending an email request to dflater@nist.gov. If you have obtained the package some other way, please email dflater@nist.gov for tracking purposes. A large number of users or a collection of testimonials will increase the support for similar work in the future.

Please refer to Appendix 2 for important legal information about the distribution and use of the various pieces of Baseline 8.

2 Input

The file used as input for Baseline 8 is named sumo++.kif. This file was initially formed by concatenating the following four sources, all obtained from [1]:

- SUMO version 1.40
- Quality of service domain ontology rev. 2002-08-14, which describes computer systems and networks
- Financial domain ontology rev. 2002-08-08
- E-Commerce services domain ontology, no rev. date, as found on 2002-10-08

Commercial equipment and materials are identified in order to describe certain procedures. In no case does such identification imply recommendation or endorsement by the National Institute of Standards and Technology, nor does it imply that the materials or equipment identified are necessarily the best available for the purpose.

[1] The expansion of SUO-KIF, "Standard Upper Ontology – Knowledge Interchange Format," reflects the fact that SUMO was once submitted to the Standard Upper Ontology (SUO) Working Group of the Institute of Electrical and Electronics Engineers (IEEE) as a suggested starting point for an IEEE standard upper ontology.

Sumo++.kif was then edited to remove errors and to work around a few operational problems that arose when attempting to use the translated ontology. The modifications are summarized in general form below. The exact changes can be found by comparing sumo++.kif with the original sources, which are included in the distribution.

- As described in Section 6, the following items were deleted to work around fatal errors: relation Connected, the inheritance of Traverses by Penetrates, MaxResponseTime, and AverageResponseTime.

- Numerous trivial errors where an identifier was misspelled, an identifier was called by an incorrect name, the wrong identifier was referenced, an argument was missing, a variable name was converted to an identifier or vice-versa, were repaired.

- Some axioms that were found to be logically "broken," *e.g.*, by containing free variables in places that would lead to unintended assertions about every instance in the universe, were commented out.

- Statements containing references to undefined terms were corrected if the intent was clear, commented out if not.

- Several statements where functions and predicates were used incorrectly were fixed if the intent was clear, commented out if not.

- Several cases of unbalanced parenthesis or nested quotation inside of comments or character strings were dealt with so that Sumo2loom could continue using a very simple parsing algorithm.

- Eight functions in the services ontology were corrected to be instances of UnaryFunction instead of subclasses of it.

- Subrelations of AvailableBalance and Frequency whose definitions conflicted with that of the parent relation (specifically, BuyingPowerAmount, MarginBalanceAmount, ShortBalanceAmount, MarketValueAmount, CollectRate, SendRate, and HeartBeatRate) were made consistent with the parent.

- Subrelations of Precondition and Attribute whose definitions conflicted with that of the parent relation (specifically, ProcessPrecondition, ServiceInput, ServicePrecondition, and DegreeOfQuality) were divorced from the parent instead of harmonized with it in order to maintain consistency with complementary relations in the same domain ontologies.

It should be noted that these changes have been communicated to the maintainers of SUMO, and most or all of the errors should be fixed in the next releases of the ontologies.

3 Translation

The translating program Sumo2loom has been tested with GNU [7] g++ version 3.2. A simple Makefile configured for g++ is provided in the distribution. To compile Sumo2loom, just type 'make.'

The usage of Sumo2loom is simple:

```
> sumo2loom sumo++.kif > sumo++.lisp
Warning:  ServiceAttribute asserted to inherit from Attribute more than once
Warning:  Server asserted to inherit from ComputationalSystem more than once
Warning:  UserRequest asserted to inherit from ComputerProcess more than once
```

However, the translation that it performs is not simple. The most complex feature of Sumo2loom is in fact its powerful logic for deciding which statements are translatable and which ones are not. Four separate lists of terms are used. Some terms are placed on these lists by static, hard-coded statements in the main program (under the comment reading "Initialize rejects"). Others are added automatically based on an analysis of the input as described in Section 3.2. There should be no need to alter these lists except as part of modifications that would be made by an expert user to work with significantly different input.

- Reject_at_mere_mention – by default, if a term that appears in this list appears anywhere within a SUO-KIF statement, that statement will be rejected. Exceptions are made according to the description of Accept_toplevel.

- Accept_toplevel – if a term appears in this list, then its use in position 0[2] in the lop level of a SUO-KIF statement will be ignored when evaluating Reject_at_mere_mention.

- Reject_invoked_anywhere – if a term that appears in this list is used in position 0 anywhere within a SUO-KIF statement, that statement will be rejected.

- Reject_invoked_toplevel – if a term that appears in this list is used in position 0 in the top level of a SUO-KIF statement, that statement will be rejected.

3.1 Initialization

The terms Equal, Exists, Not, Or, and InitialList are added to Reject_invoked_toplevel with the reason "wants to be an implication." SUMO contains a number of statements where these terms are used with implicitly universally quantified variables to make assertions that are not in the form that Sumo2loom can translate.

The term Precondition is added to Reject_invoked_anywhere with the reason "sometimes takes propositions as arguments." All other relations, functions, and predicates that take propositions as arguments are caught during the first pass (described below), but Precondition is declared in such a way that it escapes the automatic sweep.

The term Connected is added to Reject_at_mere_mention with the reason "is linked to enigmatic LOOM errors." Please see Section 6.3 for details.

The terms DisjointRelation, Partition, and DisjointDecomposition are added to Accept_toplevel because these relations are specially translated even though they meet the criteria for rejection that are used in the first pass.

3.2 First pass

The first pass through the input is only to fill out the list Reject_invoked_anywhere with terms that are declared to be VariableArityRelations or that take Formulas (propositions) as arguments. LOOM does not support variable-arity relations or passing propositions as arguments. Exceptions are made for And, Or, Not, =>, <=>, Equal, Instance, Subrelation, Subclass, Exists, and ListFn, which are later translated to LOOM built-ins. (*N.B.*, attempts to translate Forall with the analogous LOOM built-in resulted in uncomputable LOOM query expressions; this fact is noted in the reason string.)

3.3 Second pass

3.3.1 Blank lines and comments

Any unit of input that does not begin with an open parenthesis is passed through to the output.

3.3.2 Documentation

The Documentation relation is handled specially so that following processing steps are not complicated by the need to distinguish between passive documentation and active statements. Renamings are performed (see 3.3.5), then the statement is passed through to the output.

3.3.3 Rejects

Rejects are detected according to the logic previously described. When a statement is rejected, its original text and the reason for its rejection are appended to the relevant output buffer.

3.3.4 @ROW variables

Sumo2loom includes logic to unroll statements containing @ROW variables (a SUO-KIF extension) into several statements that do not contain row variables. However, it is not exercised in Baseline 8 since every statement

[2] Position 0 is where the name of the function or operator that is being invoked or the predicate or relation that is being asserted at appears. For example, in the statement (Undo Troy Cat), Undo appears in position 0 and Troy and Cat are arguments 1 and 2 respectively. This obscure terminology is being used to avoid the confusion that would result from using the LISP terminology "function" in this context where the term may not refer to a function at all.

containing a @ROW variable also includes a reject term. @ROW variables are primarily of use in the axioms defining variable-arity relations.

3.3.5 Renamings

Note: While SUMO class and relation names have been capitalized for readability in the remainder of this document, in this section they must be presented exactly as they appear in the ontology to avoid changing the meaning.

The following strings are prefixed with "Sumo" to eliminate clashes with LOOM built-ins: Character, Class, Collection, domain, Function, Integer, List, Number, Proposition, property, range, Relation, Set, Sequence, subset, disjoint, documentation, inverse.

The following strings are suffixed with "-rel" to eliminate clashes that result when case-sensitive SUMO is rendered into case-insensitive LOOM: agent, attribute.

ListFn is replaced with the LOOM built-in :the-list. (This is moot since no occurrences of ListFn survived.)

3.3.6 Partitions

The information contained in SUMO's Partition and DisjointDecomposition relations is attached to the relevant concepts so that analogous LOOM partitions can be generated in the output. The disjointness constraint of disjoint decompositions is not preserved because two different ways of expressing this in LOOM both caused fatal operational problems.

3.3.7 Disjoint relations

SUMO statements declaring relations to be disjoint are translated into LOOM implications of the following form:

```
(implies Rel1 (:satisfies (?x ?y) (:not (Rel2 ?x ?y))))
```

3.3.8 Implications (=>)

Only implications having a "simple" consequent are translated into LOOM implications. An expression is not "simple" if it contains an invocation of And, Or, Not, =>, <=>, or Exists.

SUMO implications are translated into LOOM implications of the following form:

```
(implies (:satisfies (?FOO0 … ?FOON) (:for-some ([antecedent vars]) (:and
  ([translated antecedent expression])
  (= ?FOO0 [translated expression from argument 1 of consequent])
  …
  (= ?FOON [translated expression from argument N of consequent])
))) consequent-relation)
```

Expressions appearing within the implication are translated by replacing And, Or, Not, =>, Equal, Instance, Subrelation, and Subclass with the corresponding LOOM built-ins, by replacing <=> with the :and of two :implies, and by promoting any Exists into the outermost :for-some. In conjunction with the global rejection of Forall, this last step prevents :for-some or :for-all from appearing within the scope of the outer :for-some. (Attempted uses of nested quantifiers caused LOOM to declare most or all of these query expressions uncomputable.)

3.3.9 Bi-implications (<=>)

Bi-implications (<=>) are broken down into two => implications that are then handled as described in Section 3.3.8. Only the commentary that is generated when one or both halves is rejected reflects the fact that it began as a single statement.

3.3.10 Other definitions

It may take numerous SUMO statements to express the information that goes into a single LOOM concept or relation definition. Sumo2loom accumulates definitions for concepts and relations one piece at a time. A "find-or-create" strategy is used to access the relevant records so that it does not matter which particular combination or ordering of SUMO statements appears.

- When an Instance proposition appears, an analogous LOOM declaration is generated. However, there may be additional actions depending on the class that is referenced.
    - When something is declared to be an instance of BinaryRelation, BinaryPredicate, UnaryFunction, TernaryRelation, TernaryPredicate, BinaryFunction, *etc.*, Sumo2loom finds-or-creates a relation and sets its arity accordingly. SpatialRelation is handled as a special case since the arity is ambiguous, and CaseRole is recognized as a subclass of BinaryPredicate.
    - When something is declared to be an instance of SymmetricRelation or CommutativeFunction, the symmetric or commutative property of the found-or-created relation is set to true.
    - When something is declared to be an instance of Class, InheritableRelation, or DeonticAttribute, Sumo2loom finds-or-creates a concept.
- When a Subrelation proposition appears, Sumo2loom finds-or-creates the subrelation and adds the parent relation to its set of parents.
- When an Inverse proposition appears, Sumo2loom finds-or-creates the two relations and sets the inverse attribute of each one to the name of the other.
- When a Range or RangeSubclass proposition appears, Sumo2loom finds-or-creates the relation and sets its range attribute appropriately. For now, RangeSubclass is generalized to Range; in the future, an appropriate union of subclasses could be generated instead. For ranges that are the union or intersection of concepts, UnionFn and IntersectionFn are replaced by the LOOM built-ins :or and :and respectively.
- When a Domain or DomainSubclass proposition appears, Sumo2loom finds-or-creates the relation and sets the selected element of the domain vector appropriately. For now, DomainSubclass is generalized to Domain; in the future, an appropriate union of subclasses could be generated instead. For domains that are the union or intersection of concepts, UnionFn and IntersectionFn are replaced by the LOOM built-ins :or and :and respectively.
- When a Subclass proposition appears, Sumo2loom finds-or-creates the subconcept and adds the parent concept to its set of parents.
- When a Disjoint proposition appears, Sumo2loom generates complementary implications of the form `(implies X (not Y))`.

3.3.11 Trivial propositions

Any propositions not otherwise handled by now are passed through to the output.

3.4 Post-processing

3.4.1 Flagging nonprimitive concepts

In LOOM, the definition of a nonprimitive concept gives necessary and sufficient conditions for membership while the definition of a primitive concept gives only necessary conditions. This distinction is absent from SUMO but critical in LOOM. Careless choices lead either to the merging of concepts that should not be merged or the creation of incoherent concepts.

Sumo2loom makes all concepts primitive by default then uses a recursive heuristic to decide which ones should be nonprimitive. Any concept that has multiple parent concepts, at least one of which is either the Abstract concept or a nonprimitive concept, is made nonprimitive. Exceptions are made for ObjectAttitude and PropositionalAttitude. They are disjoint concepts with identical definitions. Making them nonprimitive causes them to be merged into a single, incoherent concept.

3.4.2 Subrelation domain resolution

There is a minor conflict of expressiveness between the SUMO and LOOM syntaxes for declaring subrelations. In LOOM it is not possible to explicitly override some domains while implicitly inheriting others from the parent relation. This conflict is resolved by making all inherited domains explicit.

3.5 Output generation

Output generation is mostly a matter of traversing the various maps that were built up previously and printing out the corresponding LOOM expressions. However, some final adjustments are made during the generation of output.

- In SUMO, only functions have ranges. But in LOOM, the first N-1 arguments of an N-ary relation are domains and the Nth one is the range. (This allows any relation to be used as a function over its first N-1 arguments.) When a relation that is affected by this difference is being output, the declaration for the last argument is shifted from the domains to the range.

- The heuristic for identifying nonprimitive relations is trivial, so it is handled during output generation. If a relation has multiple parent relations, a nonprimitive declaration is output; otherwise it is primitive.

4 Output

Sumo2loom emits to standard output a LISP source file that has been saved as sumo++.lisp. The file consists of two parts. The first part contains definitions that were built up from possibly many SUO-KIF statements. The second part contains the translation of the remainder of the SUO-KIF input, including commentary and asserted relations, in the same order that these appeared in the input.

Sumo++.lisp has only been tested with LOOM 4.0 (1999-07-12) patch level 64 (2002-10-21) running under Allegro Common Lisp Enterprise Edition 5.0.1 for the Sparc platform. In other environments, modification may be required.

To load the ontology, start up the LOOM environment and issue the `load` command:

```
USER(1): (load "sumo++.lisp")
```

Many diagnostics are output during the loading process. Information about them can be found in Section 6 (Unresolved issues).

Loading can take over an hour, so it is highly recommended to save a snapshot of the LISP environment after loading is complete. In Allegro, this is accomplished with the `dumplisp` command:

```
USER(2): (dumplisp :name "baseline8.dxl")
```

After that, the SUMO+LOOM environment can be started up in seconds with a command like the following:

```
> lisp –I baseline8.dxl
```

5 Using the ontology

First, it is necessary to start up the SUMO+LOOM environment and gain access to the SUMO package.

```
> lisp –I baseline8.dxl
```
[… startup messages deleted …]
```
USER(1): (in-package "SUMO")
#<The SUMO package>
SUMO(2):
```

Then, SUMO concepts can be instantiated and SUMO relations can be asserted using the LOOM command `tell`. Similarly, queries can be executed using the LOOM command `retrieve`.

Note: NIST Special Publication 811, "Guide for the Use of the International System of Units (SI)" [8], specifies "In the expression for the value of a quantity, the unit symbol is placed after the numerical value and a *space* is left between the numerical value and the unit symbol." Unfortunately, the measures package of LOOM does not work that way. In the script below, the values of quantities have been left in the form that is required by LOOM so that readers will be able to reproduce the results shown. However, it should be understood that this form is deprecated by NIST.

```
SUMO(2): (tell (Product Widget))
QQrrr*
Recognition changes for state 216 in context SUMO-THEORY:
```

```
   entry:  WIDGET   |C|PRODUCT
           WIDGET   |C|ARTIFACT
           WIDGET   |C|CORPUSCULAROBJECT
           WIDGET   |C|SELFCONNECTEDOBJECT
           WIDGET   |C|OBJECT
           WIDGET   |C|PHYSICAL
           WIDGET   |C|ENTITY
OK
SUMO(3): (tell (side BoojumSide Widget))
QQQQrT.+^!
#sealing operations = 1
rrrrrr*
Recognition changes for state 217 in context SUMO-THEORY:
   entry:  BOOJUMSIDE    |C|SELFCONNECTEDOBJECT
           BOOJUMSIDE    |C|OBJECT
           BOOJUMSIDE    |C|PHYSICAL
           BOOJUMSIDE    |C|ENTITY
*
Recognition changes for state 217 in context SUMO-THEORY:
   entry:  WIDGET   |C|REGION
OK
SUMO(4): (tell (side SnarkSide Widget))
QQQrrr*
Recognition changes for state 218 in context SUMO-THEORY:
   entry:  SNARKSIDE     |C|SELFCONNECTEDOBJECT
           SNARKSIDE     |C|OBJECT
           SNARKSIDE     |C|PHYSICAL
           SNARKSIDE     |C|ENTITY
OK
SUMO(5): (tell (distance SnarkSide BoojumSide 42cm))
OK
SUMO(6): (retrieve ?x (width Widget ?x))
```
[... lots of "thinking" deleted ...]
```
(42cm)
SUMO(7):
```

The above example makes use of LOOM's built-in measures package to express 42 cm. SUMO includes its own definitions of measures, but having lost some axioms in translation they are not as functional as the LOOM built-in.

If a command refers to a SUMO concept or relation that was not fully computed in the original loading process, there may be a delay of several minutes while that work is finished. However, once that work is done, it is not repeated. These delays can be avoided in the future by saving another snapshot of the LISP environment and using that in subsequent runs.

6 Unresolved issues

This section is effectively the release notes for Baseline 8. These issues may be fixed in future releases, by NIST or by others, if this work is continued.

6.1 Unfinished reconciliation of SUMO concepts with LOOM built-in theory

SUMO includes some concepts that are redundant with and sometimes incompatible with concepts that appear in LOOM's built-in theory. The relations Instance and Equal are notable examples. Sumo2loom replaces some uses of these SUMO concepts with LOOM built-ins but leaves others alone.

If one were to desist from replacing SUMO concepts with LOOM built-ins, LOOM's facilities for reasoning about classes and instances would no longer work in the SUMO context. More work would be needed to determine whether additional rules to make the connection between SUMO concepts and LOOM built-ins would suffice to

repair that damage. On the other hand, if one were to replace SUMO concepts with LOOM built-ins in every possible context, there would be many inconsistencies to resolve. Assertions about LOOM built-ins that are made within the SUMO theory generally cause scoping errors.

6.2 Warnings about bad query expressions

LOOM warns that it cannot find a way to generate bindings for all of the query variables in the relations GreaterThanOrEqualTo, LessThanOrEqualTo, Equal, and OverlapsTemporally. It also warns of questionable definitions and free variables in Measure, Instance, and AfterTaxIncome. The sources of these warnings are not immediately obvious but may become apparent after further testing and use of the ontology.

6.3 Deletion of relation Connected

A variety of fatal errors that occurred while loading the ontology was found to relate to the nontrivial mix of disjointness assertions and multiple inheritance in the definitions of the subrelations of Connected. The least destructive way around the problem was to sacrifice Connected itself. The subrelations remain, but the relationships among them that depended on the parent relation are gone.

Baseline 8 includes a concept Connected whose only role is to hold onto the documentation that was asserted for the relation.

6.4 Deletion of inheritance of Traverses by Penetrates

A variety of fatal errors that occurred while loading the ontology was found to relate to the multiple inheritance by relation Penetrates of relation Traverses and relation MeetsSpatially (notably, a subrelation of Connected). The problem was worked around by deleting the inheritance of Traverses.

6.5 Deletion of MaxResponseTime and AverageResponseTime

The domains of MaxResponseTime and AverageResponseTime were incoherent, which caused loading to fail. The correct resolution was not obvious, so the relations were simply deleted.

6.6 Unfinished business

For the time being, some potentially translatable constraints, *e.g.*, the mutual exclusivity of contrary attributes, have been left as passive SUMO assertions with no mechanism to enforce them. The reconciliation of SUMO concepts with the LOOM built-in theory will impact how this is resolved.

7 Conclusion

The LOOM package output by Sumo2loom contains representations of most SUMO concepts and relations and some SUMO axioms. There is no perfect translation from predicate logic to LOOM's logic because the former is more expressive. However, the LOOM built-in theory replaces the most important functionality. It is thus possible to make practical use of the upper ontology while also enjoying the scalability and performance of LOOM's logic environment.

8 Acknowledgments

The author thanks Adam Pease of Teknowledge for assistance with SUMO and Thomas Russ of the Information Sciences Institute of the University of Southern California for assistance with LOOM.

9 References

[1] Suggested Upper Merged Ontology (SUMO), http://ontology.teknowledge.com/, 2002. Persistent URL http://purl.org/net/dflater/org/sumo.

[2] Ian Niles and Adam Pease, "Towards a Standard Upper Ontology," in Proceedings of the Second International Conference on Formal Ontology in Information Systems (FOIS-2001), ACM Press, October 2001, pp. 2-9.

[3] OpenCyc, http://www.opencyc.org/, 2002. Persistent URL http://purl.org/net/dflater/org/opencyc.

[4] Michael Genesereth, "Knowledge Interchange Format," in Proceedings of the Second International Conference on the Principles of Knowledge Representation and Reasoning (KR-91), Morgan Kaufman Publishers, 1991, pp. 238-249. See also http://logic.stanford.edu/kif/kif.html, persistent URL http://purl.org/net/dflater/org/kif.

[5] Adam Pease, "Standard Upper Ontology Knowledge Interchange Format," http://suo.ieee.org/suo-kif.html, 2002. Persistent URL http://purl.org/net/dflater/org/kif/suo-kif.

[6] LOOM, http://www.isi.edu/isd/LOOM/LOOM-HOME.html, 2002. Persistent URL http://purl.org/net/dflater/org/loom.

[7] GNU's Not Unix – the GNU Project and the Free Software Foundation, http://www.gnu.org/, 2002. Persistent URL http://purl.org/net/dflater/org/gnu.

[8] Barry Taylor, "Guide for the Use of the International System of Units (SI)," NIST Special Publication 811, http://physics.nist.gov/Pubs/SP811/contents.html, April 2001. Persistent URL http://purl.org/net/dflater/org/nist/sp811.

10 Change log

10.1 Baseline 8, 2003-01-22

Thomas Russ of the Information Sciences Institute of the University of Southern California provided assistance that was instrumental in locating the sources of the incoherent concepts remaining in Baseline 7.

The sumo++.kif source file was modified to eliminate the incoherent concepts. No modifications to the sumo2loom program were required.

Changes to this document:

- Corrected a typo in the example in Section 5 (Using the ontology)
- Removed Section 6.2 (Incoherent concepts)
- Revised Section 2 (Input) to reflect the changes to sumo++.kif
- In Section 4 (Output), noted the patch level of LOOM that was tested
- Added Section 8 (Acknowledgments)
- Added Section 10 (Change log)
- Changed references from Baseline 7 to Baseline 8

10.2 Baseline 7, 2003-01-09

First public release, approved by WERB 2003-01-09.

11 Appendix 1: Section 508 compliance statement

§ 1194.21 Software applications and operating systems.

(a) When software is designed to run on a system that has a keyboard, product functions shall be executable from a keyboard where the function itself or the result of performing a function can be discerned textually.

> Sumo2loom operates entirely in a text mode from the command line. This requirement is satisfied.

(b) Applications shall not disrupt or disable activated features of other products that are identified as accessibility features, where those features are developed and documented according to industry standards. Applications also shall not disrupt or disable activated features of any operating system that are identified as accessibility features where the application programming interface for those accessibility features has been documented by the manufacturer of the operating system and is available to the product developer.

> Sumo2loom does not knowingly disrupt other functions.

(c) A well-defined on-screen indication of the current focus shall be provided that moves among interactive interface elements as the input focus changes. The focus shall be programmatically exposed so that assistive technology can track focus and focus changes.

> Sumo2loom operates entirely in a text mode from the command line. This requirement is not applicable.

(d) Sufficient information about a user interface element including the identity, operation and state of the element shall be available to assistive technology. When an image represents a program element, the information conveyed by the image must also be available in text.

> Sumo2loom operates entirely in a text mode from the command line. This requirement is not applicable.

(e) When bitmap images are used to identify controls, status indicators, or other programmatic elements, the meaning assigned to those images shall be consistent throughout an application's performance.

> Sumo2loom operates entirely in a text mode from the command line. This requirement is not applicable.

(f) Textual information shall be provided through operating system functions for displaying text. The minimum information that shall be made available is text content, text input caret location, and text attributes.

> Sumo2loom operates entirely in a text mode from the command line using normal system input and output functions. This requirement is satisfied.

(g) Applications shall not override user selected contrast and color selections and other individual display attributes.

> Sumo2loom operates entirely in a text mode from the command line. This requirement is satisfied.

(h) When animation is displayed, the information shall be displayable in at least one non-animated presentation mode at the option of the user.

> Sumo2loom uses no animation. This requirement is not applicable.

(i) Color coding shall not be used as the only means of conveying information, indicating an action, prompting a response, or distinguishing a visual element.

> Sumo2loom uses no color coding. This requirement is satisfied.

(j) When a product permits a user to adjust color and contrast settings, a variety of color selections capable of producing a range of contrast levels shall be provided.

> Sumo2loom operates entirely in a text mode from the command line. This requirement is not applicable.

(k) Software shall not use flashing or blinking text, objects, or other elements having a flash or blink frequency greater than 2 Hz and lower than 55 Hz.

> Sumo2loom uses no blinking text. This requirement is satisfied.

(l) When electronic forms are used, the form shall allow people using assistive technology to access the information, field elements, and functionality required for completion and submission of the form, including all directions and cues.

> Sumo2loom uses no electronic forms. This requirement is not applicable.

§ 1194.31 Functional performance criteria.

(a) At least one mode of operation and information retrieval that does not require user vision shall be provided, or support for assistive technology used by people who are blind or visually impaired shall be provided.

> Sumo2loom operates entirely in a text mode from the command line. Assistive technology that reads out text from standard system output functions should work. If a user finds this to be a problem, contact dflater@nist.gov for assistance.

(b) At least one mode of operation and information retrieval that does not require visual acuity greater than 20/70 shall be provided in audio and enlarged print output working together or independently, or support for assistive technology used by people who are visually impaired shall be provided.

> Sumo2loom operates entirely in a text mode from the command line. Users should be able to adjust the font in their text windows as needed.

(c) At least one mode of operation and information retrieval that does not require user hearing shall be provided, or support for assistive technology used by people who are deaf or hard of hearing shall be provided.

> Sumo2loom uses no audio. This requirement is satisfied.

(d) Where audio information is important for the use of a product, at least one mode of operation and information retrieval shall be provided in an enhanced auditory fashion, or support for assistive hearing devices shall be provided.

> Sumo2loom uses no audio. This requirement is not applicable.

(e) At least one mode of operation and information retrieval that does not require user speech shall be provided, or support for assistive technology used by people with disabilities shall be provided.

> Sumo2loom uses no speech. This requirement is satisfied.

(f) At least one mode of operation and information retrieval that does not require fine motor control or simultaneous actions and that is operable with limited reach and strength shall be provided.

> Sumo2loom uses only standard keyboard input. Assistive technology for standard keyboards should work.

§ 1194.41 Information, documentation, and support.

(a) Product support documentation provided to end-users shall be made available in alternate formats upon request, at no additional charge.

> This should be possible. Contact dflater@nist.gov for assistance.

(b) End-users shall have access to a description of the accessibility and compatibility features of products in alternate formats or alternate methods upon request, at no additional charge.

> Sumo2loom requires no built-in accessibility and compatibility features. This requirement is not applicable.

(c) Support services for products shall accommodate the communication needs of end-users with disabilities.

> Sumo2loom has no official support. This requirement is not applicable.

12 Appendix 2: legal

12.1 Sumo2loom software

This software was developed at the National Institute of Standards and Technology by employees of the Federal Government in the course of their official duties. Pursuant to Title 17 Section 105 of the United States Code this software is not subject to copyright protection and is in the public domain.

You can redistribute it and/or modify it freely provided that any derivative works bear some notice that they are derived from it, and any modified versions bear some notice that they have been modified. We would appreciate acknowledgement if the software is used.

This software is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. NIST does not assume legal liability or responsibility for anything done with this software.

12.2 Ontologies

SUMO was once submitted to the Standard Upper Ontology (SUO) Working Group of the Institute of Electrical and Electronics Engineers (IEEE) as a suggested starting point for an IEEE standard upper ontology. SUMO is described on the SUO Working Group web page (http://suo.ieee.org/) as an "expert contribution." The text at http://suo.ieee.org/suopapers.htm reads: "These documents are the work of the authors and not approved or otherwise endorsed by this working group."

SUMO did not succeed in entering the standards track, so it never became an IEEE document and the copyright remained with its authors. The terms that Adam Pease of Teknowledge has described for SUMO could be characterized as "the usual and customary unwritten academic license" where use and distribution are permitted without fee provided that "due credit" is given. However, any modified versions cannot be called by the name SUMO.

The domain ontologies, on the other hand, are released under the terms of the GNU General Public License (GPL). Under the terms of the GPL, since sumo++.kif and sumo++.lisp are modified derivatives of SUMO *and* the domain ontologies, they must also be released under the GPL.

A copy of the GPL follows.

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and

modification follow.

GNU GENERAL PUBLIC LICENSE
TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based

on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are

prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free

Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year>  <name of author>

This program is free software; you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation; either version 2 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program; if not, write to the Free Software
Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA  02111-1307  USA
```

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

```
    Gnomovision version 69, Copyright (C) year name of author
    Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
    This is free software, and you are welcome to redistribute it
    under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

```
  Yoyodyne, Inc., hereby disclaims all copyright interest in the program
  `Gnomovision' (which makes passes at compilers) written by James Hacker.

  <signature of Ty Coon>, 1 April 1989
  Ty Coon, President of Vice
```

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.