LOJBAN AS A TOOL FOR ENCODING PROSE ON THE SEMANTIC WEB

A Thesis

Presented to

The faculty of California Polytechnic State University

San Luis Obispo

In Partial Fulfillment

Of the Requirements for the Degree

Master of Science in Computer Science

By

Brandon Wirick

November 2005

AUTHORIZATION FOR REPRODUCTION
OF MASTER'S THESIS

I grant permission for the reproduction of this thesis in its entirety or any of its parts, without further authorization from me.

_____

Signature

_____

Date

APPROVAL PAGE

TITLE:	Lojban as a Tool for Encoding Prose on the Semantic Web
AUTHOR:	Brandon Wirick
DATE SUBMITTED:	21 November 2005

Adviser	Signature
Committee Member	Signature
Committee Member	Signature

ABSTRACT

Lojban as a Tool for Encoding Prose on the Semantic Web

Brandon Wirick

As the Web evolves, the problem of enabling software to extract semantics from prose becomes increasingly important. Current solutions for representing semantic concepts mostly involve large concept hierarchies called ontologies in which the classes have more or less arbitrary names based on English words. Parsing English prose into documents that reference these ontologies is problematic and would rely heavily on inadequately accurate techniques based on artifical intelligence.

I explore a new avenue, which involves an artificial, logical, spoken language called Lojban, a language that is special in that it can be used to write both prose and unambiguous logical statements. I demonstrate that, for at least a simple but non-trivial subset of Lojban, parsing prose into documents that reference a Lojban ontology is automatic, even trivial by some comparisons.

Work that builds on this would include, but not be limited to, expanding the subset of Lojban I present and finding ways to use Lojban as an intermediate step for semantic parsing of certain types of prose in natural languages.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF LISTINGS

1. Introduction

Ontology, as a study, is "the branch of metaphysics that deals with the nature of being." [1] ("Ontology.") In artificial intelligence and related fields of study, however, an ontology is "an explicit formal specification of how to represent the objects, concepts and other entities that are assumed to exist in some area of interest and the relationships that hold among them." [2] They are a critical part of the Semantic Web effort, an "extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation." [5] As the members of the Web Ontology Working Group put it, "Ontologies figure prominently in the emerging Semantic Web as a way of representing the semantics of documents and enabling the semantics to be used by web applications and intelligent agents." [39]

An upper ontology is an ontology that is "limited to concepts that are meta, generic, abstract and philosophical, and therefore are general enough to address (at a high level) a broad range of domain areas." [34] Borgo et al claim that "the most important upper ontologies," at least as of 2002, are DOLCE[1], OPENCYC[2], SUMO[3], and BFO[4]. [6] (s. 3) All four of these upper ontologies consist of English-based terms and English descriptions, as any cursory exploration of their components shows.[5]

The problem with using English for naming and describing ontologies, however, is its syntactic ambiguity: many words have multiple, distinctly different meanings (e.g. *set*, *fire*, *sick*, etc.), many words have multiple, similar meanings (e.g. *love*, *burn*,

1 See project website, "http://www.loa-cnr.it/DOLCE.html"
2 See project website, "http://www.opencyc.org/"
3 See project website, "http://ontology.teknowledge.com/index.html"
4 See project website, "http://ontology.buffalo.edu/bfo/BFO.htm"
5 For instance, view the OWL file for SUMO, at "http://reliant.teknowledge.com/DAML/SUMO.owl"

glasses, etc.), and (although not quite as relevant to ontology development) many words even gain meanings from use in idiomatic expressions and slang (e.g. “The apple of my eye,” “You sure clean up nicely,” “Peep home-boy's pimp ride,” etc.) Determining which meaning applies to which instances of the words is often a matter of making sense of how they are used (e.g. “Set it down nicely,” vs. “I bought the twelve-piece set.”). Borgo et al say, however, that for an ontology to serve useful purposes such as facilitating negotiation of meaning among agents of different communities, it requires “the explicit representation of ontological commitment in order to exclude terminological and conceptual ambiguities bound to unintended interpretations,” [6] (s. 1) so using English to name the terms in an ontology defeats some of its useful purpose.

English is a difficult language to parse as well, mostly for the same reason. Church and Patil suggest about English that “Sentences are far more ambiguous than one might have thought. Our experience [...] indicates that there may be hundreds, perhaps thousands, of syntactic parse trees for certain very natural sentences of English.” [7]

Lojban is an artificial human language that its fluent speakers claim to be “at least as expressive as English” [27] (p. 13) but “based on the principles of logic,” (p. 1) without burdening its speakers with ambiguity. (p. 9) Both Lojban and English are “human languages,” meaning they are both intended to facilitate written and spoken communication among humans. English however is a “natural language,” meaning that it evolved over time to satisfy the needs of a particular culture, while Lojban is an “artificial language,” meaning that people designed it and created it. There are many artificial languages (e.g. Esperanto [16], Klingon [29], Sindarin [33], etc.), but what makes Lojban special is that it was designed according to principles of predicate logic [27] (p. 131) in

order to help evaluate the Sapir-Whorf hypothesis, the "notion that the language you speak affects the way you think." (p. 113) In fact, *lojban* is the Lojban word for "logical language." Those who are highly involved with Lojban have made some very interesting claims about it:

- It is completely syntactically unambiguous [27] (p. 9)
- It is easier to learn than natural languages [27] (p. 9)
- Linguists are interested in its potential as an *interlingua,* or intermediate language in translation of natural languages, for computer-aided translation. [18]

Finding an appropriate interlingua is one of several challenges in one of several approaches to developing sophisticated "machine translation" techniques, [19] and is not necessarily related to ontologies, although they sometimes come into play. [24]

There are other logical human languages; Hennings maintains a directory of several dozen of them, [13] but Lojban stands out among them: it is the only one on the list submitted by a corporation (the Logical Language Group[21]) rather than an individual, Hennings describes it as "the most professional and thought-provoking of the modern logical languages," and some of the languages in the directory even use Lojban as a source along with other natural languages.

1.1. Motivation

My work begins with the assumption that an ontology created from Lojban *brivla*, predicate words, rather than English adjective-noun combinations, would retain more of its intended purpose because both the concepts and the terms would be unambiguous. In other words, not only would each concept covered by the ontology map to exactly one

word, but each word would map to exactly one concept as well. This raises the question of whether plain Lojban text could be automatically translated into logical statements about the terms of an ontology on the Semantic Web.

This comes close to solving the problem of enabling people to have non-trivial conversations with their computers, except for the fact that the estimated number of Lojban speakers is currently in the range of hundreds, [27] (p. 15) so it would not solve the problem for many people. English speakers would only be able to have non-trivial conversations with computers that could translate English prose into some formal representation, which is a rich problem in the area of machine translation. [4]

Arnold explains “understanding” by differentiating among three different kinds of knowledge that it involves: [4] (pp. 47-8)

- Semantic, knowledge about expressions' meanings independent of context
- Pragmatic, knowledge about expressions' meanings across various usages
- Real world, knowledge about how to reasonably disambiguate unclear references

In Arnold's terms, I intend to separate the task of extracting semantic knowledge from a sample of prose from the task of extracting pragmatic and real world knowledge from it. My work starts from the imaginary point in the development of machine translation technology at which parsers are available that can translate from natural languages into Lojban without relying significantly on semantic knowledge. At that point, assuming that the claims made about this language are accurate, then it should only be a matter of syntactic parsing to generate documents for the Semantic Web that represent the logical relationships that samples of prose represent, in terms of a Lojban-based upper ontology.

My speculations led me to the Jorne Project, an open-source project that holds such goals as, “creating APIs for automated translation of Lojban to and from data in the Semantic Web.” [10] It appears to be the only significant effort within the currently active Lojban community (besides what Speer and Havasi mention as future work for the JIMPE system [35]) to join Lojban with part of the Semantic Web. In fact, *jorne* is the Lojban word for “joining” or “unifying.”

We on the Jorne project currently are not concerned with machine translation from any natural language into Lojban or vice versa. We currently focus our efforts on developing tools to extract important logical relationships from Lojban prose for use in the Semantic Web. Some of our tools that are relevant to this problem include a Lojban-based upper ontology and a predicate extractor.

1.2. Problem Statement

Consider this situation: A ten-year-old girl is reading a hardback copy of *Alice in Wonderland* in English when she comes upon a phrase with which she is not familiar. She notes the page number, goes to her computer, looks up a certain website and queries for the page number she had noted. The text comes up in her browser, she selects the sentence that contains the unfamiliar phrase, and the page promptly displays a dynamically generated diagram that represents an interpretation of the logic behind that sentence, and many of the elements in the diagram link to other places in the book, and in other literary works, where similar predicate relationships are used, sorted by relevance.

What would drive this website would be a database, built from collecting translations of literary works into Lojban from all over the Semantic Web, and a

reasoning engine based on an ontology of Lojban terms that would be capable of measuring the similarities between two pieces of Lojban prose, not just by textual similarity, but by conceptual similarity.

This dynamically generated logic representation would be typical of systems that employ Lojban-based ontologies, save one question:

Can the predicate relationships contained within an arbitrary piece of prose, written in some non-trivial subset of Lojban, be automatically extracted and made available to the Semantic Web in a format that complies with a single, static ontology?

1.3. Evaluation Criteria

This question slightly open-ended: even if a non-trivial subset can be parsed as described, there are many levels of sophistication above that, but handling all those levels of sophistication (or determining the point at which solutions do not exist) is a very large amount of work. Understanding how to address this issue requires familiarity with some details of Lojban, which appears in Section 5. Therefore, Section 6 addresses this issue.

I present here a proof of concept. Either there exists a non-trivial subset of Lojban that can be parsed as described, in which case I provide a small number of examples of how one might extend the base-level functionality of a parser, or only very limited Lojban can be parsed, in which case I present an analysis of why Lojban is not the solution I thought it was.

A working solution requires a prose parser and a compatible ontology, so I evaluate the question in Section 1.2 by attempting to implement both until I have either produced a working solution or an unavoidable failure. The pertinent questions are:

- Can an upper ontology be created entirely out of Lojban terms that have the same meanings as the classes they represent?
- Can a parser be created that converts Lojban prose into documents defined using this ontology?
- Can a non-trivial subset of Lojban be defined over which this parser can correctly parse all statements?

Summary of Section 1:

- Ontologies are important components of the Semantic Web
- Lojban may serve useful purposes not only for speech but the Semantic Web
- Questions arise whether Lojban has significant practical advantages over natural languages like English for automatically extracting semantics from prose

2. Background

As the analysis is specialized and technical, some background on the problems and technologies mentioned so far may be helpful at this point. It is important to understand the motivations behind the Semantic Web, upper ontologies, and Lojban, as well as some of the details of their implementations.

2.1. The Semantic Web

Tim Berners-Lee invented the World Wide Web (WWW or “the Web”) in 1989 and has directed the World Wide Web Consortium (W3C) since it was founded in 1994. [15] The Web has been quite a successful technology. It now provides people all over the world with a means of asynchronously posting and viewing other people's public data, and it has even infiltrated English vocabulary (e.g. “What website should I go to?” or, “I was just surfing the Web.”) Even people who aren't very computer savvy can use this amazing technology!

Problems arise, however, when people try to get their computers to “surf the Web” for them. Berners-Lee et al identify the crucial issue to be that “Most of the Web's content today is designed for humans to read, not for computer programs to manipulate meaningfully.” [5] (This is not difficult to understand, considering that many websites are never spell-checked, grammar-checked, or edited to make sure their content makes sense.)

A good solution would be to add some structure on top of the existing Web by which computers could navigate what already exists. People who care about computers

being able to process the data on their website can provide it separately in a special, standardized format. This standard would necessarily include some means for specifying large vocabularies of terms so that computers can communicate about large sets of concepts. In fact, this is exactly what the W3C is trying to make possible.

It is called the Semantic Web, and it is one of the W3C's chief projects. Berners-Lee *et al* claim it to be that which "the World Wide Web of today... will evolve into tomorrow," and they describe it as "an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation." [5]

The Semantic Web is characterized by several W3C recommendations, each of which plays some part in enabling people to author computer-parseable information, including large vocabularies, known as ontologies, that consist of terms, descriptions, and relationships. The recommended top-level language for writing ontologies is known as the Web Ontology Language or OWL, but OWL is built on several lower-level recommendations, including:

- Extensible Markup Language (XML), a "simple, very flexible text format," [32] which allows for the expression of textual data with a tree structure consisting of named nodes with attributes and is the preferred medium for writing OWL ontologies
- XML Schema, an XML-based language for "defining the structure, content and semantics of XML documents" and provides the definitions of primitive data-types used by OWL ontologies [36]
- Resource Description Framework (RDF), a "general-purpose language for representing information in the Web," [26] which allows for the identification of

resources, any concepts that can be unambiguously identified by the W3C-endorsed addressing scheme, Uniform Resource Identifiers (URI), and the relationships among those resources, upon which OWL ontologies are based

- RDF Schema, a "standard which describes how to use RDF to describe RDF vocabularies on the Web," such as an OWL ontology [26]

The W3C describes OWL as an ontology definition language "designed for use by applications that need to process the content of information instead of just presenting information to humans," and builds upon XML, RDF, and RDF Schema. [25]

2.2. Ontologies in OWL

That the W3C recommends OWL as the language for writing ontologies for the Semantic Web is clearly sufficient to consider it good practice to use OWL to write ontologies. There are other languages for writing ontologies (e.g. KIF, DAML+OIL), but OWL is the new W3C recommendation. (It became a recommendation in February 2004 [25].) Here is a brief overview of some of the aspects involved in writing an OWL ontology.

OWL comes in three flavors: OWL Lite, OWL DL, and OWL Full. OWL Full allows for a custom meta-classes. In OWL Lite and OWL DL, the only meta-class is "owl:Class," but in OWL Full, classes can be instances of other classes; this allows classes to have both extension-based and instantiation-based relationships with other classes.

Consider an example of two interlocking ontologies, one about fruits and one about word lengths, as shown in Figure 1. In the fruits ontology, there are only extension-

based relationships. A Lemon is a Citrus, which in turn is a Fruit. The ontology based on word lengths is a meta-ontology for fruits, because the fruit classes have instantiation-based relationships with the word classes. A Lemon is not a Five Letter Word, but *Lemon* is; note the careful use of italicization.

Meta-ontologies could also be used to arrange classes according to the parts of speech under which their names fall, (e.g. *Lemon* is a noun) although problems may arise in cases where classes have names that fall into two or more parts of speech (e.g. *Orange* is a noun in one sense, but an adjective in another.) This is one problem that syntactic ambiguity poses for ontologies. One solution to this problem is to label classes with arbitrary identifiers that have single, static, artificial meanings but bear similarities to words in a natural language that represent similar concepts.

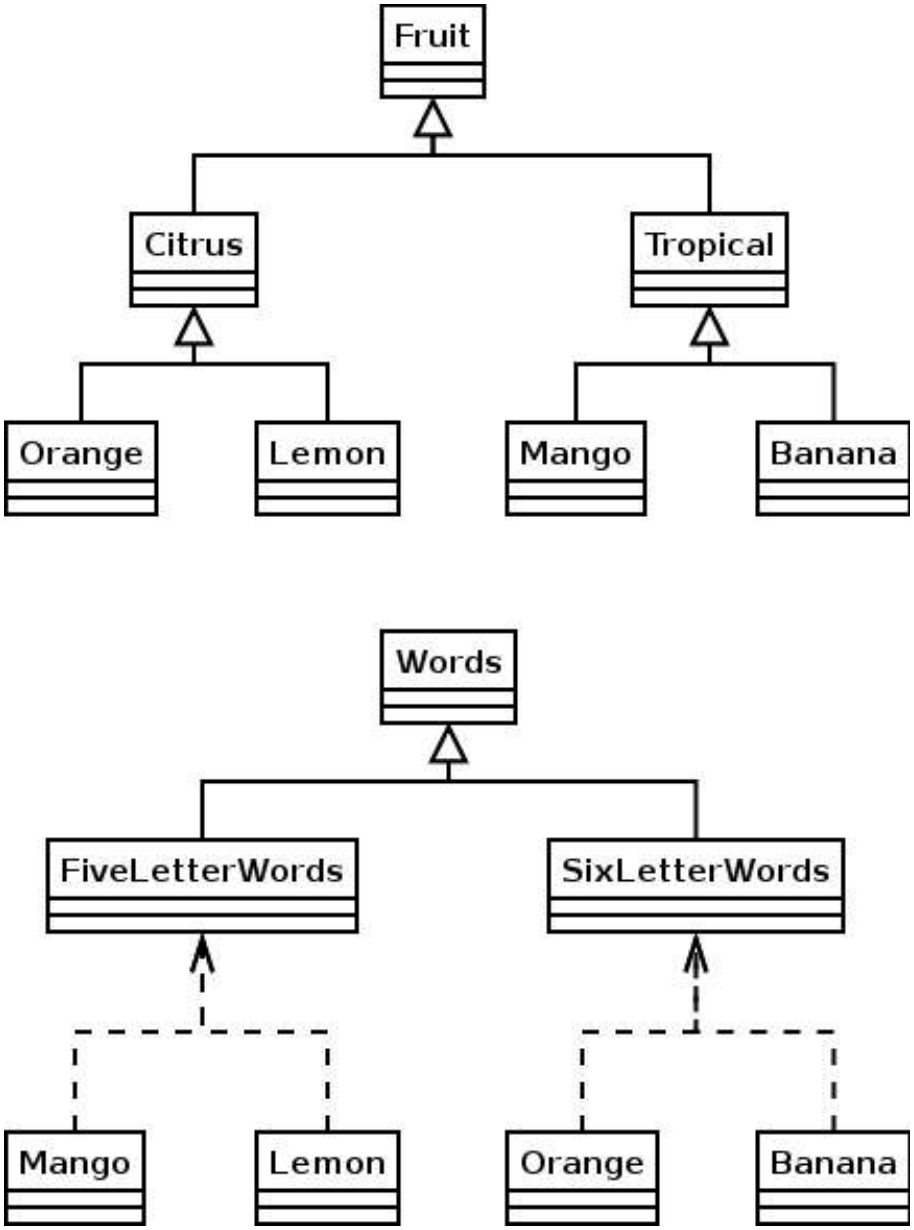

Figure 1: Ontology and Corresponding Meta-Ontology

2.3. Lojban

Another solution is to use a language for which words already exist that fulfill the desired requirements for naming classes in an ontology. Lojban is an artificial, logical, human language derived from a language called Loglan, which started in 1955 as an experiment to test the Sapir-Whorf Hypothesis. [17] In 1987, the Logical Language Group (LLG) was formed in order to develop Lojban. The LLG maintains <http://lojban.org>, which provides not only resources for learning Lojban, but complete word-lists and formal grammars as well. [23]

LeChevalier claims, “Unique features of Lojban remove constraints on language in the areas of logic, ambiguity, and expressive power, opening up areas of thought that have not been easily accessible by human language before.” He goes on to say, “Lojban is undoubtedly the most carefully designed and defined AL ever created. All aspects of its design have been carefully engineered by several people encompassing expertise in a variety of disciplines, including linguistics.” [18]

Lojban reads and sounds somewhat like a Slavic or a Romance language. Lojban has an audio-visually isomorphic nature, meaning that there is only one way to write a spoken sentence, and there is only one way to read a written sentence, although there is a little bit of room for accent and whitespace modification, but only when there is no syntactic ambiguity. For instance, Lojban allows for slightly different pronunciations for the letter *ry* ('r'), so not everyone who learns Lojban needs to have an American English accent, but none of the pronunciations can overlap with the pronunciations of any other letters, like *ly* ('l').

Every letter has one sound, which does not get modified in combination with other letters. Lojban vowels are pronounced like Spanish vowels, except for *ybu* ('y'), which is a schwa (i.e. a relaxed, middle vowel as in the English word *bush*). The letter *cy* ('c') is always pronounced like the English 'sh', never like the English 'k' or 's', and the letter *jy* ('j') is its voiced counterpart, always pronounced like the 's' in *pleasure*. The letter *xy* ('x') is an unvoiced, velar fricative, a sound that doesn't occur in English, but does appear in Scottish (e.g. “loch”). The sound that 'j' makes in *joke* becomes 'dj' in Lojban, and the sound that 'ch' makes in *such* becomes 'tc' in Lojban. Very importantly, the character that looks like a period is the letter *denpa bu*, which is pronounced as a glottal stop like the hyphen in *uh-oh*; the character that looks like a comma is the letter *slaka bu,* which prevents two vowels from forming a diphthong; and the character that looks like an apostrophe is the letter *y'y*, which is pronounced like the 'h' in *behind*. [27] (p. 29)

Table 1 lists some important Lojban terminology. Familiarity with these terms will facilitate understanding of the forthcoming analysis of the language. Lojban uses a predicate structure for sentences. Every clause (*bridi*) has exactly one predicate (*selbri*), which behave like verbs in English, although Lojban words that translate readily to English nouns can also be used as predicates with an implied “is a” predicate relationship. The Lojban word for “human” is *remna*, so the Lojban sentence, “*mi remna*” means, “I am a human.”

Lojban Term	***Meaning***
bridi (bree-dee)	A predicate, similar to a simple sentence
selbri (sell-bree)	The relationship on which a *bridi* is based, similar to a verb
sumti (soom-tee)	An argument to a *bridi*, similar to a subject or a direct object
cmene (shmeh-neh)	A name, similar to a proper noun
cmavo (shmah-vo)	A structure word (many kinds)
brivla (breev-lah)	A word that can be a *selbri*
gismu (geese-moo)	The most basic *brivla*; a five-letter root word
lujvo (loozh-vo)	A *brivla* made from a combination of *gismu* and perhaps *cmavo*
fu'ivla (foo-heave-lah)	A *brivla* formed by borrowing a word that is foreign to Lojban
rafsi (rahf-see)	An affix that abbreviates a *gismu* or *cmavo* (used to form *lujvo*)
selma'o (sell-mah-ho)	A class of *cmavo* that encapsulates some behavior

Table 1: Important Lojban Terminology

Nicholas and Cowan claim that “Lojban is actually much simpler than natural language... Because Lojban's grammar is simple, it is easier to learn than other languages.” [27] (p. 9)

According to the Lojban Brochure, “Lojban's predicate structure is similar to AI, suggesting it as a powerful tool in AI processing, especially in the storing and processing of data about the world and people's conceptions of it. Linguists are interested in Lojban's potential as an intermediate language in computer-aided translation of natural languages. Other people are interested in Lojban as an international language.” [18]

Summary of Section 2:

- The Semantic Web is an emerging technology that provides descriptions of meaningful relationships among resources on the World Wide Web
- These relationships are defined in part by ontologies, written in OWL
- Identifiers in an ontology that make sense can be defined in meta-ontologies
- Resources can be named meaningfully and unambiguously with Lojban

3. Literature Review

Here is a summary of projects in the literature that are closely related to the Jorne Project, sorted roughly by prominence. Note the lack, however, of Lojban-based ontologies in these projects. The Jorne Project can easily find a place along side of these.

3.1. IEEE SUO

On 14 June 2003, the IEEE P1600.1 Standard Upper Ontology Working Group (SUO) passed a motion to commence work on a project to develop a standard for ontology specification and registration, based on the contributions of three SUO candidate projects: IFF, OpenCyc, and SUMO. [37] All three of these ontologies consist of classes named and commented in English. Importantly, as some of their designers admit, these names are arbitrary, not informative.

From the page on the description of CycL, the language for interacting with Cyc, “It's also very important never to assume that you, the observer of the CYC® KB, can know with certainty what a constant denotes to the system, just from seeing its name and nothing else.” [9] Similarly, from the SUMO FAQ page, “A term [in SUMO] means what its axioms say it means; no more, no less. [...] There is no objective basis for deciding on a name. Better to treat each name like an arbitrary symbol, such as GENSYM345432, if the term name doesn't seem evocative [...]” [30]

What this indicates in terms of the stated metrics is a high learning curve for these English-based ontology languages which stems from the low precision of English. The terms in English are not reliable sources for the ontology languages, so the languages

don't have a one-to-one mapping. Even English speakers would have to look up the axioms of every term in order to understand what they mean, not to mention speakers of other natural languages.

3.2. SUMO

SUMO stands for "Suggested Upper Merged Ontology" and is a very important part of the IEEE SUO's developing standard. Niles and Pease sum up the justification for creating SUMO:

> In order to enable continued progress in e-commerce and software integration, we must give computers a common language with a richness that more closely approaches that of human language. Unfortunately, there is now, as things stand, a trade-off between precision and expressiveness. On the one hand, computer-readable languages are impoverished – they permit computers to represent only very specific and limited things. On the other hand, human languages can state almost anything anyone would ever want to say. However, so many of the terms and structures of human languages are vague or ambiguous that these languages are not very useful for specifying meanings to a computer. [28]

In terms of the metrics stated, Niles and Pease say that human languages have low precision and high actual expressiveness while computer-readable languages have the opposite. Interestingly, Lojban is both a human language and a computer-readable language, but it has both high precision and high actual expressiveness. In other words, it transcends the trade-off.

3.3. Robin Lee Powell

Robin Lee Powell is a member of the LLG and head of its *baupla fuzykamni*, a committee in charge of a variety of tasks related to the development of Lojban [22] . He

dedicates several pages on his personal website to Lojban grammar, on which he notes some shortcomings in the state of affairs and provides some tools aimed at eventually correcting the problems.

He claims that the grammars the LLG provides on their site (one written in YACC and one written in BNF [20]) are "non-standard," and in places, "*very* non-standard." He feels that it is "*extremely* misleading to say that Lojban is formally parseable," as long as parts of the grammars provided are not formalized. [31]

He is using Parsing Expression Grammars (PEGs) [11], which he claims to seem "perfect for what Lojban needs," to create a more adequate formal representation for Lojban. He does not claim on his page that Lojban is not formal or not computer-readable, only that computers can't quite yet parse **all** of it **perfectly**. (This is not comparable with the degree to which computers cannot parse natural languages perfectly —as of 2005, many of the remaining problems with parsing Lojban center around details of how the parser handles the words *si* and *sa*, which are backspace-like words, similar to the English expressions, "I mean..." and "Let me start over," which would be extremely difficult for an non-human English parser to handle reliably.)

3.4. WordNet

WordNet is a widely known project of the Cognitive Science Laboratory at Princeton University. It is an "online lexical reference system whose design is inspired by current psycholinguistic theories of human lexical memory," in which "English nouns, verbs, adjectives and adverbs are organized into synonym sets, each representing one underlying lexical concept." [3] The Global WordNet Association [38] has extended the

WordNet project and intends to “promote the standardization of the specification of wordnets for all languages in the world,” including:

- The “standardization of the Inter-Lingual-Index for inter-linking the wordnets of different languages, as a universal index of meaning”
- The “development of a common representation for wordnet data”

WordNet serves as a lexicon rather than an ontology. Hirst describes a lexicon as something that is very similar to an ontology but that “depends, by definition, on a natural language and the word senses in it.” [14] Whereas the purpose of a lexicon is to provide semantics for all the words in a language, the purpose of a general-purpose ontology is to provide semantics for all concepts, regardless of language.

Gomez notices some shortcomings in WordNet (possibly related to Hirst's notion of “overlapping word senses”):

> The concept *written-communication*, which has many subconcepts, is categorized in Wordnet 1.6 only as an *abstraction*. Thus, the interpreter failed to interpret such simple sentences as "She burned the letter/She put the letter on the table," because "letter" does not have *physical-thing* as one of its hypernyms (superconcepts). In "The fish frequently hides in a crevice," the interpreter failed to assign meaning to "hides" because "crevice" is categorized in WordNet 1.6 only as an *abstraction*. In "Blood poured from the wound," the interpreter fails to assign meaning to "poured" because "wound" and its hypernym, "injury," are not as a *physical thing* in WN. The examples are many. [12]

Hirst suggests that a lexicon can be viewed as “an index that maps from the written form of a word to information about that word.” He claims, however, that this is not a one-to-one correspondence: “Words that occur in more than one syntactic category will usually have a separate entry for each category.”[14]

In Lojban, words cannot appear in more than one syntactic category, because each word's category can be uniquely determined by the spelling of the word. If Hirst were aware of the Lojban lexicon, perhaps he would call it a one-to-one correspondence.

3.5. JIMPE

Lojban's potential for application to the Semantic Web has not gone unnoticed. Speer and Havasi are developing a Lojban parser called JIMPE that can make inferences about given Lojban predicates without needing supplemental contextual information. (s.3.7) They plan eventually to add “a module that uses Semantic Web inference engines to draw conclusions” about Lojban sentences. They intend to take advantage of WordNet and other information available on the Semantic Web by supplying translation rules that map to them from Lojban terms. [35]

Summary of Section 3:

- There are several large, English-based upper ontologies
- With a small and decreasing number of exceptions, Robin Lee Powell's parser can parse any Lojban prose
- WordNet fails in some cases because it is based on a language in which the lexicon is not a one-to-one correspondence, unlike the Lojban lexicon
- Lojban has received some attention in the Semantic Web community

4. Solutions

A solution for a very basic subset of Lojban is somewhat obvious, given that technology is already available that can parse Lojban prose into a hierarchical, rule-based structure. From that, solutions involve designing ways to infer and extract various kinds of information about the logical relationships within parsed prose.

4.1. Powell's Parser

The Lojban parser that Robin Lee Powell wrote does the bulk of the work necessary in transforming Lojban prose into something out of which the Semantic Web can make sense, but it does not employ Semantic Web formats. It does, however, create a hierarchical structure, which facilitates simple querying for information about logical relationships contained in the prose.

4.1.1. Morphological Parsing

The first task Powell's parser performs is to determine the part of Lojban speech to which each word in a line of text belongs. If the text contains words that are not valid in Lojban, it parses them as “non-Lojban words,” but does not fail. Assume the parser receives this text as input (Listing 1):

```
loi cicyge'u cu batci
```

Listing 1: Sample Input Text

It is a Lojban sentence which means, “Part of the mass of those that are wild dogs are things that bite,” or simply, “Wild dogs bite.” As a proposition, it can be read, “Some

wild dogs are biters." The parser processes each token according to rules specified in its grammar and produces the following transformation (Listing 2, emphasis added):

```
CMAVO=( LE=( loi ) ) spaces=( ) BRIVLA=( lujvo=( cicyge'u ) )
spaces=( ) CMAVO=( CU=( cu ) ) spaces=( ) BRIVLA=( gismu=(
batci ) )
```

Listing 2: Results of Morphology Parse

In this example the parser recognizes *cmavo* of two different *selma'o* (*LE* and *CU*) and recognizes two *brivla*: one *gismu* and one *lujvo*.

4.1.2. Structural Parsing

The second task Powell's parser performs is to attempt to find a structural hierarchy for the words in the text that is valid according to the rules of its grammar. If such a hierarchy does not exist (i.e. the text is not valid Lojban), the parser fails and produces no output. Given the input text in Listing 1, the parser produces the following transformation (Listing 3, emphasis added):

```
text=( text1=( paragraphs=( paragraph=( statement=( statement1=
( statement2=( statement3=( sentence=( terms=( terms1=( terms2=
( term=( term1=( sumti=( sumti1=( sumti2=( sumti3=( sumti4=
( sumti5=( sumti6=( LEClause=( LEPre=( LE=( CMAVO=( LE=( loi ) )
) ) ) sumtiTail=( sumtiTail1=( selbri=( selbri1=( selbri2=
( selbri3=( selbri4=( selbri5=( selbri6=( tanruUnit=( tanruUnit1=(
tanruUnit2=( BRIVLAClause=( BRIVLAPre=( BRIVLA=( BRIVLA=( lujvo=(
cicyge'u ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) )
) ) ) ) ) ) ) CUClause=( CUPre=( CU=( CMAVO=( CU=( cu ) ) )
) ) bridiTail=( bridiTail1=( bridiTail2=( bridiTail3=( selbri=
( selbri1=( selbri2=( selbri3=( selbri4=( selbri5=( selbri6=
( tanruUnit=( tanruUnit1=( tanruUnit2=( BRIVLAClause=( BRIVLAPre=
( BRIVLA=( BRIVLA=( gismu=(
batci ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) ) )
) ) ) ) )
```

Listing 3: Results of Structural Parse

Even though the example is a sentence with only four words, its hierarchy is over two dozen nodes deep. Listing 4 is an XML-based equivalent to this transformation; Powell's Parser does not produce XML like that, but the transformation is academic.

```
<?xml version="1.0" encoding="UTF-8"?>
<text>
 <text1>
  <paragraphs>
   <paragraph>
    <statement>
     <statement1>
      <statement2>
       <statement3>
        <sentence>
         <terms>
          <terms1>
           <terms2>
            <term>
             <term1>
              <sumti>
               <sumti1>
                <sumti2>
                 <sumti3>
                  <sumti4>
                   <sumti5>
                    <sumti6>
                     <LEClause>
                      <LEPre>
                       <LE>
                        <CMAVO>
                         <LE>loi</LE>
                        </CMAVO>
                       </LE>
                      </LEPre>
                     </LEClause>
                     <sumtiTail>
                      <sumtiTail1>
                       <selbri>
                        <selbri1>
                         <selbri2>
                          <selbri3>
                           <selbri4>
                            <selbri5>
                             <selbri6>
                              <tanruUnit>
                               <tanruUnit1>
                                <tanruUnit2>
                                 <BRIVLAClause>
                                  <BRIVLAPre>
                                   <BRIVLA>
                                    <BRIVLA>
                                     <lujvo
                                      >cicyge'u</lujvo>
                                    </BRIVLA>
                                   </BRIVLA>
                                  </BRIVLAPre>
                                 </BRIVLAClause>
                                </tanruUnit2>
                               </tanruUnit1>
                              </tanruUnit>
                             </selbri6>
                            </selbri5>
                           </selbri4>
                          </selbri3>
                         </selbri2>
                        </selbri1>
                       </selbri>
                      </sumtiTail1>
                     </sumtiTail>
                    </sumti6>
                   </sumti5>
                  </sumti4>
                 </sumti3>
                </sumti2>
               </sumti1>
              </sumti>
             </term1>
            </term>
           </terms2>
          </terms1>
         </terms>
         <CUClause>
          <CUPre>
           <CU>
            <CMAVO>
             <CU>cu</CU>
            </CMAVO>
           </CU>
          </CUPre>
         </CUClause>
         <bridiTail>
          <bridiTail1>
           <bridiTail2>
            <bridiTail3>
             <selbri>
              <selbri1>
               <selbri2>
                <selbri3>
                 <selbri4>
                  <selbri5>
                   <selbri6>
                    <tanruUnit>
                     <tanruUnit1>
                      <tanruUnit2>
                       <BRIVLAClause>
                        <BRIVLAPre>
                         <BRIVLA>
                          <BRIVLA>
                           <gismu
                            >batci</gismu>
                          </BRIVLA>
                         </BRIVLA>
                        </BRIVLAPre>
                       </BRIVLAClause>
                      </tanruUnit2>
                     </tanruUnit1>
                    </tanruUnit>
                   </selbri6>
                  </selbri5>
                 </selbri4>
                </selbri3>
               </selbri2>
              </selbri1>
             </selbri>
            </bridiTail3>
           </bridiTail2>
          </bridiTail1>
         </bridiTail>
        </sentence>
       </statement3>
      </statement2>
     </statement1>
    </statement>
   </paragraph>
  </paragraphs>
 </text1>
</text>
```

Listing 4: XML Representation of a Lojban Sentence

4.2. Semantic Web Formats

The parser I developed for the Jorne Project builds on Powell's parser by adding two steps to the parsing process. First, it takes transformations that result from the structural parse (e.g. Listing 3) and transforms that into XML, similar to the format of Listing 4, except that it also traverses the document, calculates certain properties, and annotates several elements in the document with attributes that reflect those properties. Second, it transforms the annotated XML into RDF statements (in N3 format[6]) using a style-sheet transformation (XSLT[7]). The final results for the example text in Listing 1 are as follows, in Listing 5:

```
@prefix : <#> .
@prefix jon: <http://jorne.org/lojban#> .
@prefix gism: <http://jorne.org/lojban/gismu#> .
@prefix lujv: <http://jorne.org/lojban/lujvo#> .

:b105FA7CA682 jon:selbri gism:batci .
:b105FA7CA682 jon:fa lujv:cicyge'u .
```

Listing 5: Results of RDF Parse

The first statement after the prefix statements means that *bridi* #105FA7CA682 has as its *selbri* a *gismu*, *batci*, and the second statement means that *bridi* #105FA7CA682 has as its first *sumti* (marked in lojban by the word *fa*) a *lujvo*, *cicyge'u*. The hexadecimal number 105FA7CA682 is a time-stamp that indicates when the *bridi* was processed, with millisecond precision, as a partial means of uniquely identifying the *bridi* on the Semantic Web.

6 See website, “http://www.w3.org/2000/10/swap/Primer”
7 See website, “http://www.w3.org/Style/XSL”

The prefixes (e.g. "`jon`," "`gism`," "`lujv`") correspond to namespaces, as specified by the prefix statements. While two resources might have the same local identifier, as with `jon:selbri` and `lujv:selbri`, they can refer to different resources, even different kinds of resources. In this example, `jon:selbri` refers to an OWL property while `lujv:selbri` refers to an OWL class. The empty prefix, used before the *bridi* identifiers in the example, refers to the current document.

4.3. References to an Ontology

Admittedly, the parser does not check whether the words in a sentence (besides *cmavo*) are officially supported words. It simply interprets all words that follow the morphological rules for *gismu* as *gismu*, *lujvo* for *lujvo* and so on. The parser transforms the sentence, "ti pardi" into statements about a resource called `gism:pardi`, even though *pardi* is not a word in Lojban, so there should not be a resource named *pardi* in the *gismu* name-space.

The set of RDF resources that are defined for a given name-space can be represented with an OWL ontology. Using OWL Full, name-spaces can be gathered hierarchically, as meta-classes, into a meta-ontology. Listing 6 shows a hypothetical OWL file that defines a hierarchy of Lojban parts of speech. The gismu class, for instance, is a subclass of the brivla class because all *gismu* are *brivla*.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE rdf:RDF [
  <!ENTITY jon "http://jorne.org/lojban#">
  <!ENTITY owl "http://www.w3.org/2002/07/owl#">
  <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#">
  <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#">
  <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#">
]>

<rdf:RDF xmlns="&jon;"
    xmlns:owl="&owl;"
    xmlns:rdf="&rdf;"
    xmlns:rdfs="&rdfs;"
    >
  <owl:Ontology rdf:about=""/>

  <owl:Class rdf:ID="tcita">
    <rdfs:subClassOf rdf:resource="&owl;Class"/>
  </owl:Class>
  <owl:Class rdf:ID="rafsi">
    <rdfs:subClassOf rdf:resource="&jon;tcita"/>
  </owl:Class>
  <owl:Class rdf:ID="selmaho">
    <rdfs:subClassOf rdf:resource="&jon;tcita"/>
  </owl:Class>
  <owl:Class rdf:ID="valsi">
    <rdfs:subClassOf rdf:resource="&jon;tcita"/>
  </owl:Class>
  <owl:Class rdf:ID="brivla">
    <rdfs:subClassOf rdf:resource="&jon;valsi"/>
  </owl:Class>
  <owl:Class rdf:ID="fuhivla">
    <rdfs:subClassOf rdf:resource="&jon;brivla"/>
  </owl:Class>
  <owl:Class rdf:ID="gismu">
    <rdfs:subClassOf rdf:resource="&jon;brivla"/>
  </owl:Class>
  <owl:Class rdf:ID="lujvo">
    <rdfs:subClassOf rdf:resource="&jon;brivla"/>
  </owl:Class>
  <owl:Class rdf:ID="cmavo">
    <rdfs:subClassOf rdf:resource="&jon;valsi"/>
  </owl:Class>
  <owl:Class rdf:ID="cmene">
    <rdfs:subClassOf rdf:resource="&jon;valsi"/>
  </owl:Class>
</rdf:RDF>
```

Listing 6: A Meta-Ontology for Lojban Terms

A helpful convention for upper ontologies to make use of this meta-ontology would be for all *gismu* to be defined in classes that are instances of `http://jorne.org/lojban#gismu` and that are in the `http://jorne.org/lojban/gismu#` namespace.

4.4. Using a Meta-Ontology

There is no single reason to use meta-ontologies, but for Lojban, there are a few clear benefits. First, although an upper ontology may consist of *gismu* that are subclasses of *lujvo* and vice-versa, *gismu* and *lujvo* are defined differently. For instance, *gismu* have *rafsi* abbreviations, but *lujvo* don't; *lujvo* have underlying hierarchies, but *gismu* don't. It may be desirable to create different infrastructure for defining words of either type although they play the same role in upper ontologies.

Another, more obscure benefit comes from higher-order logic. Upper ontologies sometimes have classes for concepts that are closely related to the concepts on which ontologies themselves are built, like a class for classes, or for ontologies, or for words. In Lojban, the word for "word" is *valsi*. In the Jorne Project, there would be an OWL class, `http://jorne.org/lojban/gismu#valsi` and an OWL meta-class, `http://jorne.org/lojban#valsi`; the question is whether there is any semantic difference between the two. Setting them equal to each other, as demonstrated in Listing 7, has far-reaching implications and may lead to undesirable results in RDF-based tools because it bridges an ontology with its meta-ontology.

```
@prefix jon: <http://jorne.org/lojban#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix gism: <http://jorne.org/lojban/gismu#> .

jon:valsi owl:sameAs gism:valsi .
```

Listing 7: Bridging Ontologies and Meta-Ontologies

For the Jorne Project, this is an optional feature. That is to say, a file that declares statements similar to the one in Listing 7 can either be imported or ignored, depending on which order of logic would be appropriate to foster with technologies that make use of the Jorne Project.

4.5. Ontology Considerations

Powell's parser deals largely with syntactic rather than semantic issues. RDF and OWL, however, focus on semantic relationships. All the listings in this section deal with a single semantic relationship, wild dogs being biters (“loi cicyge'u cu batci”). The sentence, “lai reks. cu batci,” which means, “some things named Rex are biters,” would be parsed in a very similar way, but *cicyge'u* (“wild dog”) has a clear place in an ontology, because all wild dogs are dogs, wild things, animals, objects, and so forth. Only some things named Rex are dogs, however. Some things named Rex are people. The name Rex can be given to all sorts of things, even things that are ideas rather than objects.

The Lojban word *reks* only has a trivial static meaning: It refers to the set of things named *reks*, which is immediately evident from the word itself. Besides that, its meaning is dynamic: in one sentence, *reks* could refer to a dog, while in another sentence it could refer to a computer program. The Lojban word *cicyge'u* has a static meaning that is not immediately evident from the word itself, and no matter what sentence uses it, it

will always refer to some sort of canine with some sort of wild attribute. This is the nature of words that make desirable classes.

This limits the usefulness of including the word *reks*, that is, `http://jorne.org/lojban/cmene#reks`, in the Jorne Project ontology. Once things named Rex are accounted for, what about things named Bob or Grandpa or Main Street? Every *cmene* is just an identifier for something with an arbitrary meaning, so they don't fit well in an ontology where every class has a static meaning of its own. This raises the question of which parts of Lojban are appropriate class material. (This is not the same question as of which parts of Lojban can be parsed.)

Recall that Lojban has three parts of speech: *brivla*, the predicate words; *cmene*, the name words; and *cmavo*, the structure words. From a high level, it looks like all *brivla* are candidates for an ontology, but no *cmene* or *cmavo*. An in-depth study of each, however, reveals some nuances.

Summary of Section 4:

- Through Powell's parser, free Lojban prose statements can be parsed into hierarchies
- Many if not all of those statement hierarchies can be translated into RDF graphs
- Those graphs can correspond to classes in a Lojban-based ontology
- That ontology should contain mostly *brivla*, with few if any *cmavo* or *cmene*

5. Details of Lojban

Not all Lojban words have exactly one meaning all the time: Lojban has variables, words with dynamic meanings. Ontologies are meant to be static references that define how terms relate to each other. If a language existed in which these relationships were dynamic, the language itself could change with every sentence. Before consulting an ontology for the meanings of terms, variable terms must be dereferenced into static terms. Words with multiple meanings are like variables: their meanings change with context. What makes Lojban a suitable languages from which to draw terms for an ontology is that there are many words with single, static meanings.

This section features an analysis of Lojban parts of speech, highlighting groups of words that have single, static meanings. These words will be the only candidates for classes in the Jorne Project ontology. Recall Table 1 in Section 2.3 for a review of important Lojban terms.

5.1. Basic Predicate Words

The highest classes (i.e. closest to the root) in the SUMO ontology tree[8] include *Entity*, *Physical*, *Abstract*, *Object*, *Attribute*, *Proposition*, and other such abstract concepts. Their meanings are somewhat straightforward: *Entity* encompasses everything, *Physical* encompasses instances of *Entity* that have location in time-space, *Abstract* encompasses instances of *Entity* that are not instances of *Physical*, and so forth.

8 These are available via an OWL file at "http://reliant.teknowledge.com/DAML/SUMO.owl"

There are Lojban terms with similar meanings, such as *xanri*, which means "imaginable" in a technical sense and "imaginary" in a metaphorical sense and would take a very similar role to *Entity* in an ontology. Things that could be described as *xanri* could generally be an instance of *Entity*, and vice versa. Likewise, *zasti* is similar to *Physical* and *mucti* is similar to *Abstract*, even to the extent that *zasti* and *mucti* have a similar exclusive relationship to the one between *Physical* and *Abstract*. The list goes on: *dacti* resembles *Object*, *ckaji* resembles *Attribute*, *sidbo* resembles *Proposition*, etc. Figure 2 shows part of the SUMO tree and a parallel tree with corresponding Lojban concepts.

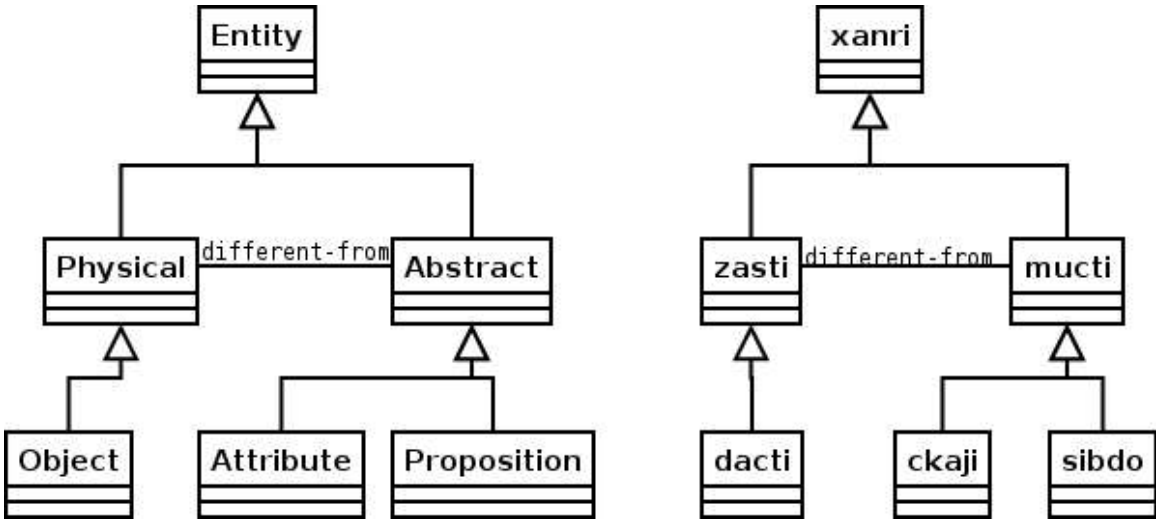

Figure 2: Parallel Ontologies for English (SUMO) and Lojban

These Lojban words are *gismu*, or root words. Each *gismu* has exactly five letters, starts with a consonant, ends with a vowel, and has exactly one other vowel as either its second or third letter. Any Lojban word that fits this structure (and that obeys other, quite technical rules) can be a *gismu*, but no other kind of Lojban word. Every *gismu* has one or

more affixes, called *rafsi*, which serve to abbreviate the *gismu*. These *rafsi* can be systematically combined with *rafsi* for other *gismu* (and for some *cmavo* as well) to create *lujvo*, compound words that serve essentially the same purpose as *gismu*, but tend to be more specific.

Some SUMO entries that have close resemblance to *lujvo* include *Argument* and *FieldOfStudy*. The Lojban word *nibypoi* is a combination of the word *nibli*, which refers to things that logically necessitate or imply something, and the word *porsi*, which refers to sequenced or ordered lists. Literally, *nibypoi* means "sequence of logical implications," or a formal argument, which is very similar to the meaning of *Argument*. There is no *gismu* that much resembles *FieldOfStudy*, but to say, ".i mi djuno fi ta" means roughly, "I know some fact about that subject." Part of using the word *djuno* includes specifying something that resembles *FieldOfStudy*. To say, ".i ta te djuno fa mi" means roughly, "That subject is one about which I know some fact." The *cmavo* (structure word) *te* swaps the first and third arguments in a predicate; in this case it extracts the concept of a subject (or field of study) from *djuno*. The word *terju'o* is short for *te djuno* and therefore closely resembles *FieldOfStudy*.

For the most part, *gismu* and *lujvo* have very static meanings and therefore correspond very clearly to classes in ontologies. Typical translations for a *gismu* include:

- dunku - x1 is anguished/distressed/emotionally wrought/stressed by x2
- fraso - x1 reflects French/Gallic culture/nationality/language in aspect x2
- nejni - x1 is energy of type x2 in form x3
- notci - x1 is a message/notice/memorandum about subject x2 from author x3 to intended audience x4

Although broad in scope, and sometimes vague (i.e. characterized by fuzzy logic), there is no ambiguity to speak of. Whereas dictionary entries for English words typically have several definitions, there is no entry like the following in a Lojban lexicon[23]:

- pardi[9] (dfn. #1) - x1 is a gathering/social event involving persons x2 at location x3
- pardi (dfn. #2) - x1 is a division of government x2 with agenda x3

Some words do have two separate translations, but they are always very closely related so that the word can apply to anything in between, such as:

- pezli - x1 is a leaf of plant x2; x1 is foliage of x2

Some words have two or more translations with different numbers of arguments but with compatible and very similar meanings, such as:

- stagi - x1 is the edible x2 portion of plant x3; x1 is a vegetable

It is possible for *gismu* to modify other *gismu* in order to create more specific terms; these modifications are called *tanru* and are constructed by placing what modifies immediately before what it modifies, as in *gerku zdani*, which means "dog/canine type of nest/house/den." With *tanru*, any relationship can do, so all doghouses, all houses that are shaped like dogs, and all houses that are dogs (i.e. dogs with fleas) are *gerku zdani*. In fact, for a *zdani* not to be a *gerku zdani*, neither it nor its inhabitants can have any relationship to any dog or a breed of dogs. This does not rule out many *zdani*.

There's very little need to create short, convenient expressions for concepts that don't get much attention or that don't have much salient meaning, and this is what motivates the use of *lujvo*, words that encapsulate one of the relationships among the words in a *tanru*, preferably the most obvious, appropriate, or otherwise significant

9 Note: *pardi* is **not** a Lojban word. It is an example of what the translation for a syntactically ambiguous Lojban word would look like if such a thing existed.

relationship. The most significant relationships for some *tanru* are more obvious than others, and there is a chance that in some contexts, for some *tanru*, some readers may perceive a different relationship to be the most significant than other readers do. For the *lujvo* based on such *tanru* to be class candidates for the Jorne Project ontology, then, there must be some sort of official definition of what relationship is the most significant (as well as place structure). The LLG provides such definitions for many *lujvo*. [23]

According to Cowan, "All *tanru* are ambiguous semantically," [8] (p.85) and the primary reason for having *lujvo* is to reduce this semantic ambiguity. [8] (p.273) Some *lujvo* are just as simple and unambiguous as *gismu*, the best example being *lujvo* (less than nine letters long) that begin with "sel-," "ter-," "vel-," or "xel-," which correspond to *cmavo* in *selma'o SE*: *se*, *te*, *ve*, and *xe*. These *cmavo* just reorder the place structure of words they modify without adding or subtracting any place-structure meaning, so the word *selbri*, which is short for *se bridi*, is no more ambiguous than *bridi*. No *lujvo* that follows this format, or that is based on only one *gismu* for that matter, needs any special *tanru*-to-*lujvo* relationship definition to be a class candidate for the Jorne Project ontology.

5.2. Variable Predicate Words

Very importantly and very exceptionally, there are five *gismu* that have no meanings of their own, but are variable predicate words: *broda*, *brode*, *brodi*, *brodo*, and *brodu*. They are unique in that their context determines their meanings and place structures. Since ontologies cannot depend on variable context, these five *gismu* cannot correspond to classes. These five *gismu* alone make it incorrect to say that all *gismu* have

single, static meanings. All other *gismu* have meanings that are static enough to be class candidates for the Jorne Project ontology. They can be known as "static *gismu*."

These words have *rafsi*, meaning they can be used to form *lujvo*. Therefore, while there are many *lujvo* with static meanings, some depend on the context in which they are used. No *lujvo* that contains *rafsi* of any of the *gismu* in the *broda* series can be a class candidate for the Jorne Project ontology.

As an aid to further investigation into this matter, the LLG has made public their comprehensive lists of the official *gismu* and of the *lujvo* that have some precedence in Lojban literature as well as an explicit *tanru* relationship definition and corresponding argument place structure. [23]

5.3. Names

Not all Lojban words are vague. Besides very long *lujvo* that are specific because they combine so many *gismu* and *cmavo*, (e.g. *blarulkemymletcepurdi*, which refers to very beautiful gardens of blue flowers,) *cmene* function like proper nouns. Using Lojban to refer to people or places often involves transliterating their non-Lojban names into valid *cmene*. Table 2 lists some names of people and places and possible corresponding *cmene*. Note that if a syllable (or just its vowels) are capitalized, that syllable is stressed. If no syllables in a word are stressed, the penultimate syllable is stressed, and *denpa bu* ('.') breaks up words. All names must start with a consonant or a *denpa bu* and end with a consonant and a *denpa bu*. (In some cases, *denpa bu* are implied.)

English Names	***Lojban* cmene**
Isaac Newton	.aizek.nutn.
Cher	ceir.
Billy Graham	byligrAm. / byl.gram.
Indiana Jones	.Endi,anydjOnz / .Endi,anys.djonz.
New York	nu,iork.
Boston	bastn.
Chicago	cykagos.
Houston	xiustn.

Table 2: Examples of Transliteration into Lojban

Ultimately, *cmene* have no inherent meanings. They are arbitrary labels for existing concepts. This is also true of RDF identifiers: there is nothing inherent about the name “http://www.w3.org/2002/07/owl#Class” that makes it the unique identifier for the OWL “Class” resource, but it is part of a consistent, universal naming system that disambiguates it from all the other things it could represent. Like natural languages, Lojban does not have such a naming system. The name *djan* could refer to anything: John Brown, John the Baptist, Timbuktu, planet Jupiter, or anything else that can be named. By convention it refers to the one named “John” within the reasonable domain of discourse, but there are no hard rules for what names mean.

If an ontology were to contain classes based on *cmene*, it would have to support a system for unambiguous naming and only allow *cmene* that comply with that system to be associated with classes. An example might be a *cmene* registry where users register unique *cmene* for themselves or for anything else they would like to name by providing some sort of RDF-based description of the entity named. It could either be an unrestricted

naming system that allows any entity to register any available name, or it could be structured so that only certain kinds of entities could register certain kinds of names. Perhaps people could only be registered with names that start with 'rem.' whereas places could only be registered with names that start with 'tut.': the word *remna*, which means "human" or "human being," has *rem* as one of its *rafsi,* which can be made into a *cmene* by appending a *denpa bu*; the word *tutra*, which means "territory," "domain," or "space," has *tut* as one of its *rafsi*. Someone named Sherman Johnson, for instance, may wish to register the name *rem.djansn.cyrmn.* on this system. Such a system is left as future work. (The Lojban community does maintain a dynamic dictionary called "Jbovlaste" that contains some *cmene*[10], but there are currently less than 100 of them, their definitions are not restricted to any sort of format, and very few of them refer to identifiable people, arguably only Bob LeChevalier, James Cooke Brown, and some characters from the Bible.) Regardless of the availability of an adequate registry, *cmene* that do not have exactly one, static meaning are not class candidates for the Jorne Project ontology.

5.4. Borrowed Words

All Lojban sentences (*bridi*, or predicates) require a *brivla*, or predicate word. The three classes of words that comprise *brivla* are *gismu*, *lujvo*, and *fu'ivla*. Section 4.1 describes *gismu* and *lujvo*, and this section describes *fu'ivla*, borrowed words. There are four varieties of *fu'ivla*, corresponding to the four stages available in Lojban to borrow a non-Lojban word.

10 See project website, "http://jbovlaste.lojban.org"

5.4.1. Quoting Foreign Words

Using a few *cmavo*, any text can become a *brivla*, even text that contains non-Lojban letters. The word *la'o* means roughly, “that which is called...,” and is always followed by some sort of quote-word, often the Lojban word for the Lojban letter that initiates the Lojban word for the language from which the text originates (an example may be necessary). The Lojban word for English is *glico* (“glee-show”), and the Lojban word for the first letter of *glico* is *gy* (which, importantly, is pronounced with a relaxed, middle vowel sound like *good* in English.) Therefore, “la'o gy. OWL document .gy.,” although lacking any inherent meaning, would conventionally refer to something called “OWL document” in English. This construct is restricted by the choice of quote-word, however, in that the quoted text cannot contain the quote-word visually or audibly, so neither “la'o gy. gyroscope .gy.” or “la'o gy. negativity .gy.” would be valid *sumti*. (The words *gy* and *negativity* have syllables that sound very similar.)

The Lojban word *me* converts arguments like *la'o*-phrases (or any *sumti* for that matter) into *brivla*; for instance, “.i ta me la'o gy. OWL document .gy.” means roughly, “That is/was/will be some quantity of that which is called, 'OWL document,'” or more concisely, “That's an OWL document.”

These “stage one” *fu'ivla* are clearly much too ambiguous to be class candidates for the Jorne Project ontology. Only a parser with a vast amount of cultural and linguistic information could make use of just the stage one *fu'ivla* that are readily recognizable to humans as terms that make sense, but *fu'ivla* that don't make any cultural sense are still valid. These terms definitely do not have static meanings.

5.4.2. Names as Predicate Words

"Stage two" *fu'ivla* are simply *cmene* that are decorated in the same way as quotations are in the first variety. Instead of *la'o*, the word *la* precedes *cmene* to make them into *sumti* (arguments), and *me* makes the *sumti* into *brivla.* For example, ".i mi me la brandn." means, "I am/was/will be a quantity of that which is called *brandn.*," or more consicely, "I am Brandon."

Because stage two *fu'ivla* are based on *cmene*, they are just as applicable to the Jorne Project ontology as *cmene* are, which depends on the naming system available.

5.4.3. Restricted Names

"Stage three" *fu'ivla* involve qualifying a *gismu* with a suffix that reflects a foreign word. These words include an abbreviated *gismu*, a hypen letter, and a Lojban-pronounceable suffix that starts with a consonant, ends with a vowel, and contains no *ybu* (the letter 'y'.) For instance, *bangrsperanto*, a *fu'ivla* that evidently refers to the Esperanto language, consists of a *rafsi* for *bangu*, which means "language," the hyphen letter *ry*, and a suffix, *speranto*, which sounds like "Esperanto."

Stage three *fu'ivla* do not require the use of *cmavo* to make *brivla*; they can be used just like *gismu* or *lujvo*. The sentence, ".i ta bangrsperanto" means "That's Esperanto."

The suffixes in these *fu'ivla* are like *cmene* in that they have no inherent meaning. The complete *fu'ivla* do have inherent meanings, however, because they are based on *gismu*. At the very least, stage three *fu'ivla* can be categorized according to their *gismu*,

making *gismu* and stage three *fu'ivla* identical in the Jorne Project ontology, which adds no benefit. At best the LLG could publish an authoritative list of accepted *fu'ivla*, but that's not a reality at this point. Until the LLG publishes static meanings for stage three *fu'ivla*, (not just a list of how often *fu'ivla* have been used in Lojban literature, [23]) they are not class candidates for the Jorne Project ontology.

5.4.4. Borrowed Words Integrated into Lojban

"Stage four" *fu'ivla* include words that are too important or have become too commonly used to have a long, *gismu*-based name. Being short and atomic, they have much in common with *gismu*, except they can't be used to form *lujvo*. (Actually there has been a proposal, still experimental in 2005, to reserve a special form of this class of words to represent cultural concepts, and these words would be able to have *rafsi* and thus contribute to *lujvo*.) Semantically they are more like *cmene*, though, because without places in an official lexicon or *gismu* bases they have no inherent meanings.

In theory there should be some sort of process for determining which of the stage three *fu'ivla* are important enough or used often enough to earn a stage four variant, which would then be added to some authoritative list and defined specifically, like *gismu*. It would appear, however, that the LLG has made no such list public, since it is not listed as an official publication, [23] and it may be that no such list (or process) yet exists. Given the state of affairs, even stage four *fu'ivla* cannot be class candidates for the Jorne Project ontology, which rules out all *fu'ivla*, at least until they appear in authoritative lexicons.

5.5. Control Words

As would be expected of a language designed to be syntactically unambiguous, many words in Lojban have single, clear meanings and would map naturally to one place in an ontology. Some words lack enough rigorous definition to be useful, but no word can be defined to mean two different things and thereby occupy two separate places in an ontology. Even vagueness is not a problem, because (as is evident from analyzing SUMO) terms in ontologies tend to be vague.

There is one more part of Lojban speech besides *brivla* and *cmene*: *cmavo*, the structure words. They serve a very wide range of purposes and are therefore subdivided into over 100 *selma'o*, or *cmavo* classes. Some *cmavo* serve purposes similar to English punctuation; others behave like English articles (e.g. a, an, the); others behave like pronouns. There are even *cmavo* to indicate the mood of the expression, to erase previous utterances, and to pause in the middle of a thought.

Most of these do not represent concepts that can be represented as classes, (i.e. that can be instantiated) just as the English word *the* is not a likely class in an English-based ontology. Most of these words, rather than having meanings that are relevant to an ontology, facilitate the use of words that do. It does not make sense to speak about a *the* or a *beyond*: these words cannot be meaningfully instantiated.

Not all English prepositions are out of the scope of ontologies, however. The preposition *near* is closely related to the noun *nearness*, for which it is not inconceivable to have an entry in an ontology. One *selma'o*, *BAI*, is (almost) entirely devoted to preposition-like *cmavo* that extract their meanings from *gismu*.

At first glance, words in *selma'o BAI* seem to warrant their own classes in an ontology. For example, *ca'i* means “by authority...” and can be used in a *bridi* to introduce elements about authority even when the *brivla* has nothing to do with authority, as in, “.i do mi ti minde ca'i ma,” which roughly means, “You command this of me by what authority?”

It is possible, however, to rewrite *cmavo* from *selma'o BAI* as *gismu*, preceded by the word *fi'o*. The word *ca'i*, for instance, is short for *fi'o catni*, which means that it doesn't need its own class. The static meaning comes from the *gismu*, but *fi'o* cannot even be meaningfully instantiated. There is one exceptional word, *do'e*, which belongs to *selma'o BAI* without corresponding to a *gismu*. It is analogous to using *fi'o* with a null *gismu*[11]; accordingly, it lacks any static meaning whatsoever.

There are a few *cmavo* that can be instantiated; they resemble English pronouns and exist primarily in *selma'o GOhA*, the so called “pro-*bridi*” and *selma'o KOhA*, the so called “pro-*sumti*.” The word *mi*, for instance, is a pro-*sumti* which means “me/we” or “the speaker(s)/author(s).” Like all the *cmavo* in these two *selma'o*, the meaning of *mi* depends on the context in which it is used, which means that it has no single, static meaning. Since no *cmavo* that can be instantiated have static meanings, *cmavo* are not class candidates for the Jorne Project ontology.

These *cmavo* that can be instantiated but have variable meanings have the same relationship to *lujvo* as *gismu* in the *broda* series. No *lujvo* that contains any *rafsi* for these *cmavo* can be a class candidate for the Jorne Project ontology.

11 There is a word, *co'e*, which can be used in place of an unspecified *bridi*, so *do'e* is similar to *fi'o co'e*.

5.6. Summary

Currently the Jorne Project ontology can only consist of static *gismu* and of *lujvo* that meet the following conditions:

- They cannot contain *rafsi* for *gismu* or *cmavo* that have variable meanings
- They must either contain only one *rafsi* of a *gismu* or their *tanru* relationship and place structure must be explicitly defined by an authoritative source.

If lexicons of *fu'ivla* and registries of *cmene* ever become available, then the ontology may be expanded to accommodate them, but not until then.

Section 5.1 contains hints as to how a Lojban-based ontology may be constructed in parallel to SUMO. Table 3 lists *gismu* and *lujvo* that map well to the classes in SUMO. The Jorne Project ontology would just consist of several classes named after Lojban words, instantiated from meta-classes like in Listing 6, and defined with owl:sameAs statements that equate them to the classes defined for the OWL version of SUMO. (Unfortunately, *lujvo* that contain the letter *y'y* (apostrophe) cannot be valid XML identifiers, so the letter *h* is used instead.)

For brevity, Table 3 is all that is necessary to describe the Jorne Project ontology. As it is currently a proof of concept, it does not map completely to SUMO.

SUMO	***Jorne Project Ontology***
Entity: The universal class of individuals. This is the root node of the ontology.	*xanri*: x1 [concept] exists in the imagination of/is imagined by/is imaginary to x2
Physical: An entity that has a location in space-time	*zasti*: x1 exists/is real/actual/reality for x2 under metaphysics x3
Abstract: Properties or qualities as distinguished from any particular embodiment of the properties/qualities in a physical medium	*mucti*: x1 is immaterial/not physical/without material form
Object: Corresponds roughly to the class of ordinary objects. Examples include normal physical objects, geographical regions, and locations of Processes, the complement of Objects in the Physical class	*dacti*: x1 is a material object enduring in space-time; x1 is a thing
Attribute: Qualities which we cannot or choose not to reify into subclasses of Object	*ckaji*: x1 has/is characterized by property/feature/trait/aspect/dimension x2 (*ka*); x2 is manifest in x1
Proposition: Abstract entities that express a complete thought or a set of such thoughts	*sidbo*: x1 (idea abstract) is an idea/concept/thought about x2 (object/abstract) by thinker x3
Quantity: Any specification of how many or how much of something there is	*klani*: x1 is a quantity quantified/measured/enumerated by x2 (quantifier) on scale x3 (*si'o*)
Relation: The Class of relations. There are three kinds of Relation: Predicate, Function, and List	*selki'i*: x2 is related to/associated with/akin to x1 by relationship x3
Class: *SetOrClass* that are not assumed to be extensional, typically have an associated `condition' that determines the instances of the Class, and cannot contain duplicate instances	*klesi*: x1 (mass/*si'o*) is a class/category/subgroup/subset within x2 with defining property x3 (*ka*)
Set: *SetOrClass* that are extensional and can be an arbitrary stock of objects	*selcmi*: x2 is a member/element of set x1; x2 belongs to group x1; x2 is amid/among/amongst group x1
Agent: Something or someone that can act on its own and produce changes in the world	*zukte*: x1 is a volitional entity employing means/taking action x2 for purpose/goal x3/to end x3
Collection: Collections have members like Classes, but, unlike Classes, they have a position in space-time and members can be added and subtracted without thereby changing the identity of the Collection	*gunma*: x1 is a mass/team/aggregate/whole, together composed of components x2, considered jointly
Region: A topographic location. Regions encompass surfaces of *Objects*, imaginary places, and *GeographicAreas*	*sefta*: x1 is surface/face [bounded shape/form] of [higher-dimension] object x2, on side x3, edges x4
Argument: A set of premises which, it is claimed, imply a conclusion	*nibypoi*: logical implications (*nibli+porsi*)
FieldOfStudy: An academic or applied discipline with recognized experts and with a core of accepted theory or practice	*terju'o*: x3 knows fact(s) x2 (du'u) about subject x1 by epistemology x4

SUMO	***Jorne Project Ontology***
Procedure: A sequence-dependent specification	*tadji*: x1 [process] is a method/technique/approach/means for doing x2 (event) under conditions x3
Number: A measure of how many things there are, or how much there is, of a certain kind	*namcu*: x1 (*li*) is a number/quantifier/digit/value/figure (noun)
List: A particular ordered n-tuple of items	*porsi*: x1 [ordered set] is sequenced/ordered/listed by comparison/rules x2 on unordered set x3
Predicate: A sentence-forming Relation. Each tuple in the Relation is a finite, ordered sequence of objects	*bridi*: x1 (text) is a predicate relationship with relation x2 among arguments (sequence/set) x3
ProbabilityRelation: The Class of Relations that permit assessment of the probability of an event or situation	*tercu'o*: x3 is random/fortuitous/unpredictable under conditions x2, with probability distribution x1
RelationExtendedToQuantities: A Relation that, when it is true on a sequence of arguments that are RealNumbers, it is also true on a sequence of ConstantQuantites with those magnitudes in some unit of measure	*xelkarbi*: x5 [observer] compares x2 with x3 in property x4 (ka), determining comparison x1 (state)
SingleValuedRelation: *Relation* ...an assignment of values to every argument position except the last one determines at most one assignment for the last argument position	*fancu*: x1 is a function/single-valued mapping from domain x2 to range x3 defined by expression/rule x4

Table 3: A Lojban Ontology as a Parallel to SUMO

Although not very interesting to the Jorne Project ontology, *cmavo* create a lot of work for the Jorne Project prose parser.

Summary of Section 5:

- Although some Lojban words depend on context for meaning, many do not
- A pertinent ontology should contain most *gismu* and *lujvo*, but nothing else
- It could also contain *fu'ivla* if appropriate word registries existed

6. Parsing Prose According to Lojban Semantics

It would require a very simple transformation to produce RDF statements from an XML document like the one in Listing 4 if sentences could only use *gismu* and *lujvo* that are class candidates, no *tanru*, and the *cmavo cu*, *lo*, and *loi*. The words *lo* and *loi* function much like existential quantifiers, except *lo* quantifies individuals whereas *loi* quantifies masses, and *cu* signals an upcoming *selbri*. The ontology in Listing 8, when combined with the properties in Listing 9, would be sufficient to represent, in RDF, the essential logic behind any statement in that subset (ignoring actual differences between *lo* and *loi.*)

This is not a trivial subset: there are quite a large number of meaningful statements (infinitely many, given that some *gismu* can take any number of *sumti*) that can be represented in this subset of Lojban, including all statements of the form, "Some S are P," where *S* and *P* are classes in the Jorne Project ontology. It is not a very sophisticated subset, however, because Lojban speakers would often want to make statements that use more *cmavo* than just *cu*, *lo*, and *loi*.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE rdf:RDF [
  <!ENTITY jon "http://jorne.org/lojban#">
  <!ENTITY owl "http://www.w3.org/2002/07/owl#">
  <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#">
  <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#">
  <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#">
]>

<rdf:RDF xmlns="&jon;"
    xmlns:owl="&owl;"
    xmlns:rdf="&rdf;"
    xmlns:rdfs="&rdfs;"
    >
  <owl:Ontology rdf:about="">
    <owl:imports rdf:resource="http://jorne.org/lojban.owl"/>
  </owl:Ontology>

  <owl:Class rdf:ID="bridi"/>

  <!-- The constructs that can be {sumti} -->
  <owl:Class rdf:ID="sumti">
    <owl:unionOf rdf:parseType="Collection">
      <owl:Class rdf:about="&jon;gismu"/>
      <owl:Class rdf:about="&jon;lujvo"/>
    </owl:unionOf>
  </owl:Class>

  <!-- The {selbri} for a {bridi} -->
  <owl:ObjectProperty rdf:ID="selbri">
    <rdf:type rdf:resource="&owl;FunctionalProperty"/>
    <rdfs:domain rdf:resource="&jon;bridi"/>
    <rdfs:range rdf:resource="&jon;sumti"/>
  </owl:ObjectProperty>
</rdf:RDF>
```

Listing 8: Definitions for a Simple Subset of Lojban

6.1. Parser Functionality

Each *selma'o* to be implemented requires some sort of software module in the prose parser that modifies the RDF output based on the logic of the *cmavo*. For example, consider *selma'o FA* and *selma'o SE*. The *cmavo* in these *selma'o* provide a means of writing prose by providing *sumti* in a *bridi* in a different order than the *selbri* specifies. For instance, the sentence, “.i mi prami do,” is logically equivalent to, “.i do se prami

mi," as well as, ".i fe do prami mi." Although the prose parser produces different XML for each, it must ultimately provide equivalent RDF outputs.

```
<!-- The first {sumti} for a {bridi} -->
<owl:ObjectProperty rdf:ID="fa">
  <rdf:type rdf:resource="&owl;FunctionalProperty"/>
  <rdfs:domain rdf:resource="&jon;bridi"/>
  <rdfs:range rdf:resource="&jon;sumti"/>
</owl:ObjectProperty>

<!-- The second {sumti} for a {bridi} -->
<owl:ObjectProperty rdf:ID="fe">
  <rdf:type rdf:resource="&owl;FunctionalProperty"/>
  <rdfs:domain rdf:resource="&jon;bridi"/>
  <rdfs:range rdf:resource="&jon;sumti"/>
</owl:ObjectProperty>

<!-- The third {sumti} for a {bridi} -->
<owl:ObjectProperty rdf:ID="fi">
  <rdf:type rdf:resource="&owl;FunctionalProperty"/>
  <rdfs:domain rdf:resource="&jon;bridi"/>
  <rdfs:range rdf:resource="&jon;sumti"/>
</owl:ObjectProperty>

<!-- The fourth {sumti} for a {bridi} -->
<owl:ObjectProperty rdf:ID="fo">
  <rdf:type rdf:resource="&owl;FunctionalProperty"/>
  <rdfs:domain rdf:resource="&jon;bridi"/>
  <rdfs:range rdf:resource="&jon;sumti"/>
</owl:ObjectProperty>

<!-- The fifth {sumti} for a {bridi} -->
<owl:ObjectProperty rdf:ID="fu">
  <rdf:type rdf:resource="&owl;FunctionalProperty"/>
  <rdfs:domain rdf:resource="&jon;bridi"/>
  <rdfs:range rdf:resource="&jon;sumti"/>
</owl:ObjectProperty>

<!-- Extra {sumti} for a {bridi} -->
<owl:ObjectProperty rdf:ID="fai">
  <rdfs:domain rdf:resource="&jon;bridi"/>
  <rdfs:range rdf:resource="&jon;sumti"/>
</owl:ObjectProperty>
```

Listing 9: OWL Properties for Referencing Arguments

Appendix A contains source code for parser logic that handles both *selma'o FA* and *SE*. Recall from Section 4.1 that when Powell's parser receives sentence text, it parses it by morphology into a string which it then parses by a grammar into another string that

represents a trees structure. The high-level parser then converts that string into XML and traverses it, organizing its *sumti* into data structures. Then, without changing the order of the elements in the tree, it sorts the *sumti* according to the order specified by any *FA* or *SE cmavo* present, and annotates the XML with attributes based on how the *sumti* should be sorted. The parser then passes the entire XML document, along with style-sheet information, to a Web server. An XSLT-compliant Web browser then renders the XML as RDF in N3 format.

Consider the sentence, "lo zukte cu te sibdo fe lo gunma." It uses only *gismu* from the ontology in Table 3, the *cmavo cu* and *lo*, and some *cmavo* from *FA* and *SE*. It means, roughly, "Some agent has an idea about a group." The place structure for *sibdo* calls first for the idea, second for what the idea is about, and third for who has the idea. Using *cmavo fe* causes the following *sumti* to take the second place of the *selbri*. and using *cmavo te* causes the *sumti* assigned to the first and third places of the *selbri* to switch. The third place in this sentence (the first place of *sidbo*) is unspecified, but the *cmavo* push it to the end of the sentence where it can be ignored. This sentence could also be written as, "lo gunma cu se sidbo fi lo zukte," which translates more awkwardly as, "Some group is the subject of an idea that some agent has," but the relationships are identical.

For the first sentence, “lo zukte cu te sibdo fe lo gunma,” the parser produces the following (Listing 10) by the process just described (steps omitted for brevity):

```
@prefix : <#> .
@prefix jon: <http://jorne.org/lojban#> .
@prefix gism: <http://jorne.org/lojban/gismu#> .
@prefix lujv: <http://jorne.org/lojban/lujvo#> .

:b106293FDDDF jon:selbri gism:sibdo .

:b106293FDDDF jon:fe gism:gunma .

:b106293FDDDF jon:fi gism:zukte .
```

Listing 10: Results of Parsing a Lojban Sentence

For the second sentence, “lo gunma cu se sidbo fi lo zukte,” the parser produces the following (Listing 11):

```
@prefix : <#> .
@prefix jon: <http://jorne.org/lojban#> .
@prefix gism: <http://jorne.org/lojban/gismu#> .
@prefix lujv: <http://jorne.org/lojban/lujvo#> .

:b10629403565 jon:selbri gism:sidbo .

:b10629403565 jon:fe gism:gunma .

:b10629403565 jon:fi gism:zukte .
```

Listing 11: Results of Parsing a Reorganized Lojban Sentence

The only difference between Listings 10 and 11 are the timestamps on the *bridi*. In both cases, *gunma* takes the second place of *sidbo* and *zukte* takes the third place. These sentences are arbitrary, but the prose parser automatically extracted the correct semantics. Appendix B contains the output from several test cases that illustrate how completely the parser implements this functionality.

6.2. Contributions

The value of this work to the field of computer science is not in a new product, service, or algorithm, but rather in a model for approaching problems involving extracting semantics from prose. The model builds upon existing technologies, solutions, and standards to reveal a new way to consider solving these problems. Given the potential to improve software that handles natural language content, this model (or others that are similar) deserve further exploration.

Lojban was not designed primarily for computer use, and it's certainly conceivable that a more appropriate language be designed, but what makes this model practical is that Lojban works well with computers and comes with decades of refinement, dozens of contributors, and several educational resources. At least it is a decent starting point for approaching problems involving extracting semantics from prose using logical spoken languages.

On the other hand, this work demonstrates that Lojban is not a silver bullet in itself: although the fantastic claims about the language, when taken together, (see Section 1) can make it sound like an ambiguity-free code for all human thought, free Lojban prose is not guaranteed to be semantically unambiguous. Prose restricted to certain subsets of Lojban can be guaranteed to be semantically unambiguous, and as it appears, some of those subsets can be quite expressive.

6.3. Future Work

There are many more *selma'o* than just *CU*, *LE*, *FA*, and *SE*. Some *selma'o* that should certainly be implemented for a Lojban prose parser include:

- *A*, *GIhA*, and *JA*, the afterthought logical connectives for various parts of sentences
- *GI* and *GA*, the forethought logical connectives for various parts of sentences
- *JOI* and *GAhO*, the non-logical connectives
- *FIhO,* the converter from *selbri* to modal *sumti*, as per *selma'o BAI*
- *GOI*, the relative phrase markers
- *POI*, the relative clause markers
- *NA*, *NAI*, and *NAhE*, the negators for various parts of sentences
- *NU*, the abstractors
- *PA*, the numbers
- *VEI*, *VEhO*, and *VUhU*, the mathematical operators

This list does not contain all the *selma'o* that might be desirable, just ones that very probably will not cause problems of compatibility with the ontology. In addition to this list, *LE* should be fully implemented to include both veridical and non-veridical descriptors. This would probably require a more complex *sumti* model than the one provided, but so would some of the other *selma'o*. Note that although *PA* should be implemented, *LI* (the *selma'o* for the numerical descriptor, *li*) possibly should not, because numbers themselves should not be classes: ontologies should not be infinitely large.

Some *selma'o* that certainly should not be implemented, either because of total irrelevance to the Jorne Project ontology or potential danger to the parser, include:

- *CAI* and *UI*, the emotional indicators
- *GOhA* and *KOhA*, the pro-*bridi* and pro-*sumti* (except maybe the *da* series)
- *LA*, the name descriptors (unless a *cmene* registry becomes available)
- *Y*, the hesitation noise
- *ZOI*, the non-Lojban input delimiters

Some *selma'o*, like *LU*, would be interesting and challenging to implement. The word *lu* starts grammatical quotations. Implementing it would require some sort of recursive parsing to first make sense of the quoted text and then to make sense of the quoting text in terms of the quotation. This may make for interesting studies of the impact of higher-order logic on machine translation.

Summary of Section 6:

- The presented parser does technically handle a non-trivial subset of Lojban, albeit not necessarily a very sophisticated one
- The model of using a logical language with an ontology and a prose parser provides a significant contribution to the field of computer science
- There remains much to be explored about Lojban parsing

7. Concluding Analysis of Results

The question was, “Can the predicate relationships contained within an arbitrary piece of prose, written in some non-trivial subset of Lojban, be automatically extracted and made available to the Semantic Web in a format that complies with a single, static ontology?”

The answer is yes. Section 6.1 and Appendix B demonstrate the results of a piece of software that parses several arbitrary pieces of prose into RDF documents that maintain the essential logical relationships and refer to elements in a single, static ontology. That prose comes from a subset of Lojban that allows for an infinite number of meaningful relationships, as Section 6 describes.

7.1. Strengths

Lojban terms have both inherent meaning as found in natural languages and uniqueness as found in computer-based naming schemes. Lojban can be written as prose, enabling people who are not familiar with computer science to translate text between it and a natural language. Lojban may be the best currently available language to use to represent the thoughts behind natural language prose in an unambiguous, culturally-neutral fashion, a particularly desirable function of the Semantic Web.

Using Lojban with the Semantic Web might create opportunities to establish Lojban as an interlingua for machine translation between two natural languages, at least for technical content that doesn't convey information through humor or idiomatic expressions.

7.2. Weaknesses

Lojban is not widely known. Consequently, its literature base is quite small. Convincing people that an artificial spoken language is an elegant solution to a problem would require overcoming the mental barrier that such languages are somehow trite or silly. In addition, most useful applications of this Lojban-based parsing method would require translation between Lojban and a natural language, which is bound to be a very difficult task to automate.

As an Interlingua for machine translation between two natural languages, Lojban may be ineffective in many cases, for instance when translating between two very similar languages that are very different from Lojban.

Lojban was not designed for the Semantic Web, so it is conceivable that a better language could be developed for ontologies. As investing in a language like this is time-expensive, it is important to invest in the best language available.

7.3. Final Assessment

This is a small but crucial step towards the fulfillment of a vision: sophisticated pieces of software, operating in conjunction with exhaustive ontologies, that parse meaningful semantic data even from natural conversations. Only time, research, and prototyping will answer the question of how much Lojban can contribute to this end.

Appendix A: Source Code for Prose Parser

This is the most relevant file from the prose parser, the code for representing a *bridi*. One method, which adds a time-stamp, has been removed for clarity.

```
//Bridi.java - J2SE 1.5.0 compliant
package lp2rdf;

import java.io.IOException;
import java.io.PrintWriter;
import java.io.Reader;
import java.io.StringWriter;
import java.util.LinkedList;
import java.util.List;
import java.util.SortedMap;
import java.util.TreeMap;

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.ParserConfigurationException;
import javax.xml.transform.Result;
import javax.xml.transform.Source;
import javax.xml.transform.Transformer;
import javax.xml.transform.TransformerConfigurationException;
import javax.xml.transform.TransformerException;
import javax.xml.transform.TransformerFactory;
import javax.xml.transform.dom.DOMSource;
import javax.xml.transform.stream.StreamResult;
import javax.xml.xpath.XPath;
import javax.xml.xpath.XPathConstants;
import javax.xml.xpath.XPathExpression;
import javax.xml.xpath.XPathExpressionException;
import javax.xml.xpath.XPathFactory;

import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;
import org.w3c.dom.ProcessingInstruction;

public class Bridi {

  public static final String DOM_FEATURES = "XML 1.0";

  public static final String NAMESPACE_URI = "http://www.jorne.org/lojban#";

  public static final String QUALIFIED_NAME = "jbo:bridi";

  public static final String SCHEMA_LANGUAGE = "W3C XML Schema 1.0";

  private static final DocumentBuilder DOCUMENT_BUILDER;

  private static final Transformer TRANSFORMER;

  private static final XPath XPATH;
```

```
public enum CmavoFA {
  FIHA(0), FA(1), FE(2), FI(3), FO(4), FU(5), FAI(0);

  private static final CmavoFA[] INDICES = { FIHA, FA, FE, FI, FO, FU };

  public final int index;

  public final String valsi;

  private CmavoFA(int index) {
    this.index = index;
    this.valsi = this.toString().toLowerCase();
  }

  public static CmavoFA getCmavoFaForIndex(int index) {
    CmavoFA c = null;
    if (index >= FA.index && index <= FU.index) {
      c = INDICES[index];
    }
    else if (index > FU.index) {
      c = FAI;
    }
    else {
      c = FIHA;
    }
    return c;
  }
}

public enum CmavoSE {
  SE(2), TE(3), VE(4), XE(5);

  public final int index;

  public final String valsi;

  private CmavoSE(int index) {
    this.index = index;
    this.valsi = this.toString().toLowerCase();
  }
}

static {
  try {
    DocumentBuilderFactory dbf = DocumentBuilderFactory.newInstance();
    dbf.setIgnoringComments(true);
    dbf.setIgnoringElementContentWhitespace(true);
    dbf.setNamespaceAware(true);
    dbf.setValidating(false);
    DOCUMENT_BUILDER = dbf.newDocumentBuilder();
  }
  catch (ParserConfigurationException e) {
    throw new RuntimeException(e);
  }
  try {
    TransformerFactory tf = TransformerFactory.newInstance();
    TRANSFORMER = tf.newTransformer();

  }
  catch (TransformerConfigurationException e) {
    throw new RuntimeException(e);
  }
  XPathFactory xpf = XPathFactory.newInstance();
  XPATH = xpf.newXPath();
}
```

```
private final Reader source;

private final Document dom;

private boolean initialized = false;

private String xmlString = new String();

public Bridi(Reader source) {
  this.source = source;
  this.dom = DOCUMENT_BUILDER.newDocument();
}

public boolean initialize() throws IOException, XPathExpressionException {
  boolean success = false;
  if (!this.initialized) {
    parseStream(this.dom);
    identifyTerms();
    initXmlString();
    success = true;
  }
  return success;
}

public boolean isInitialized() {
  return this.initialized;
}

public void transformTo(Result result) throws TransformerException {
  synchronized (this.dom) {
    DOMSource source = new DOMSource(this.dom);
    TRANSFORMER.transform(source, result);
  }
}

public void transformTo(Result result, String xsltLocation)
    throws TransformerException {
  synchronized (this.dom) {
    ProcessingInstruction xslt = this.dom.createProcessingInstruction(
        "xml-stylesheet", "type=\"text/xsl\" href=\"" + xsltLocation + '"');
    this.dom.insertBefore(xslt, this.dom.getDocumentElement());
    transformTo(result);
    this.dom.removeChild(xslt);
  }
}

@Override
public String toString() {
  return this.xmlString;
}

private void initXmlString() {
  StringWriter w = new StringWriter();
  Source in = new DOMSource(this.dom.getDocumentElement());
  Result out = new StreamResult(w);
  try {
    TRANSFORMER.transform(in, out);
  }
  catch (TransformerException e) {
    e.printStackTrace(new PrintWriter(w));
  }
  this.xmlString = w.toString();
}
```

```
private void parseStream(Node parent) throws IOException {
  boolean done = false;
  boolean firstSpace = false;
  boolean secondSpace = false;
  boolean children = false;
  String name = null;
  StringBuffer b = new StringBuffer();
  while (!done) {
    int i = this.source.read();
    if (i < 0) {
      done = true;
    }
    else {
      char c = (char) i;
      switch (c) {
        case ' ':
          if (!firstSpace) {
            firstSpace = true;
          }
          else if (b.length() == 0) {
            secondSpace = true;
          }
          break;
        case '=':
          if (secondSpace) {
            name = b.toString();
            b.delete(0, b.length());
          }
          else {
            b.append(c);
          }
          break;
        case '(':
          if (name != null) {
            Element child = this.dom.createElement(name);
            firstSpace = false;
            secondSpace = false;
            children = true;
            name = null;
            parent.appendChild(child);
            parseStream(child);
          }
          else {
            b.append(c);
          }
          break;
        case ')':
          if (!children) {
            parent.appendChild(this.dom.createTextNode(b.toString()));
          }
          done = true;
          break;
        default:
          b.append(c);
          break;
      }
    }
  }
}
```

```
protected void identifyTerms() throws XPathExpressionException {
  List<Element> plainTerms = new LinkedList<Element>();
  SortedMap<CmavoFA, Element> faTerms = new TreeMap<CmavoFA, Element>();
  List<Element> faiTerms = new LinkedList<Element>();
  List<Element> fihaTerms = new LinkedList<Element>();
  XPathExpression x;
  NodeList s;
  final int termCount;

  x = XPATH.compile("//term1");
  s = NodeList.class.cast(x.evaluate(this.dom, XPathConstants.NODESET));
  termCount = s.getLength();
  x = XPATH.compile(".//CMAVO/FA");
  for (int i = 0; i < termCount; i++) {
    Element e = Element.class.cast(s.item(i));
    String f = x.evaluate(e);
    if (f.length() == 0) {
      plainTerms.add(e);
    }
    else {
      CmavoFA key = null;
      for (CmavoFA c : CmavoFA.values()) {
        if (c.valsi.equals(f)) {
          key = c;
          break;
        }
      }
      switch (key) {
        case FA: case FE: case FI: case FO: case FU:
          faTerms.put(key, e);
          break;
        case FAI:
          faiTerms.add(e);
          break;
        case FIHA:
          fihaTerms.add(e);
          break;
      }
    }
  }
  int nonFihaCount = Math.max(5, termCount - fihaTerms.size());
  Element[] nonFiha = new Element[nonFihaCount + 1];
  for (int i = 1; i <= nonFihaCount; i++) {
    CmavoFA c = CmavoFA.getCmavoFaForIndex(i);
    Element e = null;
    if (c == CmavoFA.FAI) {
      if (faiTerms.isEmpty()) {
        e = plainTerms.remove(0);
      }
      else {
        e = faiTerms.remove(0);
      }
    }
    else {
      if (faTerms.containsKey(c)) {
        e = faTerms.remove(c);
      }
      else if (!plainTerms.isEmpty()) {
        e = plainTerms.remove(0);
      }
    }
    nonFiha[i] = e;
  }
  //continued...
```

```
    //identifyTerms (cont'd.)
    x = XPATH.compile("//selbri//tanruUnit2//CMAVO/SE");
    String se = x.evaluate(this.dom);
    for (CmavoSE c : CmavoSE.values()) {
      if (c.valsi.equals(se)) {
        Element temp = nonFiha[c.index];
        nonFiha[c.index] = nonFiha[1];
        nonFiha[1] = temp;
        break;
      }
    }
    for (int i = 1; i <= nonFihaCount; i++) {
      CmavoFA c = CmavoFA.getCmavoFaForIndex(i);
      Element e = nonFiha[i];
      if (e != null) {
        e.setAttribute("order", Integer.toString(i));
        e.setAttribute("fa", c.valsi);
      }
    }
  }
}
```

Here is the XSLT style-sheet used to transform the XML generated by Bridi.java into RDF statements.

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0"
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:variable name="timeStamp" select="/text/@time-stamp"/>
  <xsl:variable name="jorne" select="'jon:'"/>
  <xsl:variable name="cmavo" select="'cmav:'"/>
  <xsl:variable name="cmene" select="'cmen:'"/>
  <xsl:variable name="fuhivla" select="'fyvy:'"/>
  <xsl:variable name="gismu" select="'gism:'"/>
  <xsl:variable name="lujvo" select="'lujv:'"/>
  <xsl:template match="/">
    <html>
      <head>
        <title>Bridi <xsl:value-of select="$timeStamp"/></title>
      </head>
      <body>
        <p>
        @prefix : &lt;#&gt; .
        <br />
        @prefix jon: &lt;http://jorne.org/lojban#&gt; .
        <br />
        @prefix gism: &lt;http://jorne.org/lojban/gismu#&gt; .
        <br />
        @prefix lujv: &lt;http://jorne.org/lojban/lujvo#&gt; .
        </p>
        <xsl:apply-templates/>
      </body>
    </html>
  </xsl:template>
```

```
  <xsl:template match="text//sentence">
    <xsl:for-each select="//bridiTail3/selbri[1]">
      <p>:b<xsl:value-of select="$timeStamp"/>
      <xsl:text> </xsl:text>
      <xsl:value-of select="$jorne"/>selbri
      <xsl:text> </xsl:text>
      <xsl:apply-templates select=".//BRIVLAClause"/>
      <xsl:text> .</xsl:text></p>
    </xsl:for-each>
    <xsl:for-each select="//term1">
      <xsl:sort data-type="number" select="./@order"/>
      <xsl:if test="count(.//selbri) &gt; 0">
        <p>:b<xsl:value-of select="$timeStamp"/>
        <xsl:text> </xsl:text>
        <xsl:value-of select="$jorne"/><xsl:value-of select="@fa"/>
        <xsl:text> </xsl:text>
        <xsl:apply-templates select=".//selbri"/>
        <xsl:text> .</xsl:text></p>
      </xsl:if>
    </xsl:for-each>
  </xsl:template>
  <xsl:template match="text//fragment">
    <p>Fragments not supported</p>
  </xsl:template>
  <xsl:template match="CMAVO">
    <xsl:value-of select="$cmavo"/><xsl:value-of select=".//*[text()]"/>
  </xsl:template>
  <xsl:template match="cmene">
    <xsl:value-of select="$cmene"/><xsl:value-of select="."/>
  </xsl:template>
  <xsl:template match="fuhivla">
    <xsl:value-of select="$fuhivla"/><xsl:value-of select="."/>
  </xsl:template>
  <xsl:template match="gismu">
    <xsl:value-of select="$gismu"/><xsl:value-of select="."/>
  </xsl:template>
  <xsl:template match="lujvo">
    <xsl:value-of select="$lujvo"/><xsl:value-of select="."/>
  </xsl:template>

</xsl:stylesheet>
```

Appendix B: Results of Test Cases

Here are the results for testing some generic sentences. The original prose is written first, followed by the output from the parser; the `@prefix` statements are omitted in the tests, but are given here for completeness:

```
@prefix : <#> .
@prefix jon: <http://jorne.org/lojban#> .
@prefix gism: <http://jorne.org/lojban/gismu#> .
@prefix lujv: <http://jorne.org/lojban/lujvo#> .
```

1	lo zukte cu te sibdo fe lo gunma
	
	`:b106293FDDDF jon:selbri gism:sibdo .`
	`:b106293FDDDF jon:fe gism:gunma .`
	`:b106293FDDDF jon:fi gism:zukte .`
		lo gunma cu se sidbo fi lo zukte
		
		`:b10629403565 jon:selbri gism:sibdo .`
		`:b10629403565 jon:fe gism:gunma .`
		`:b10629403565 jon:fi gism:zukte .`

2	lo selcmi cu te bridi
	
	`:b107556551BC jon:selbri gism:bridi .`
	`:b107556551BC jon:fi lujv:selcmi .`
		bridi fi lo selcmi
		
		`:b10755673BB6 jon:selbri gism:bridi .`
		`:b10755673BB6 jon:fi lujv:selcmi .`

3	sefta `:b107556B9663 jon:selbri gism:sefta .`	ve sefta `:b107556CEE88 jon:selbri gism:sefta .`

4	lo fancu cu xelkarbi lo namcu lo klani fu lo xanri
	
	`:b10755748348 jon:selbri lujv:xelkarbi .`
	`:b10755748348 jon:fa gism:fancu .`
	`:b10755748348 jon:fe gism:namcu .`
	`:b10755748348 jon:fi gism:klani .`
	`:b10755748348 jon:fu gism:xanri .`
		lo xanri lo namcu cu xe xelkarbi lo klani fu lo fancu
		
		`:b1075579603F jon:selbri lujv:xelkarbi .`
		`:b1075579603F jon:fa gism:fancu .`
		`:b1075579603F jon:fe gism:namcu .`
		`:b1075579603F jon:fi gism:klani .`
		`:b1075579603F jon:fu gism:xanri .`

5	lo zukte cu te sibdo fe lo gunma
	
	`:b106293FDDDF jon:selbri gism:sibdo .`
	`:b106293FDDDF jon:fe gism:gunma .`
	`:b106293FDDDF jon:fi gism:zukte .`
		lozukte cutesibdo felogunma
		
		`:b107557C1985 jon:selbri gism:sidbo .`
		`:b107557C1985 jon:fe gism:gunma .`
		`:b107557C1985 jon:fi gism:zukte .`

BIBLIOGRAPHY

1: *American Heritage Dictionary of the English Language*. 4th Ed. Boston: Houton Mifflin, 2000.

2: *Free On-line Dictionary of Computing*, 1993-2005. Edited by Dennis Howe. <http://www.foldoc.org>.

3: Al-Halimi, Reem, et al. Felbaum, Christine, ed. *WordNet*. Cambridge, MA: Massachusettes Institute of Technology Press, 1998.

4: Arnold, Doug. *Machine Translation: an Introductory Guide.* London: Blackwells-NCC, 1994.

5: Berners-Lee, Tim, James Hendler, and Ora Lassila. "The Semantic Web." *Scientific American.* May 2001. <http://www.scientificamerican.com/article.cfm?articleID=00048144-10D2-1C70-84A9809EC588EF21&catID=2>.

6: Borgo, Stefano et al. *Ontology RoadMap.* 2002. <http://wonderweb.semanticweb.org/deliverables/documents/D15.pdf>.

7: Church, Kenneth and Ramesh Patil. "Coping with Syntactic Ambiguity or How to Put the Block in the Box on the Table." *American Journal of Computational Linguistucs*. Vol. 8. Cambridge, MA: Massachusettes Institute of Technology Press, 1982.

8: Cowan, John Woldemar. *The Complete Lojban Language*. Fairfax, VI: The Logical Language Group, Inc., 1997.

9: Cycorp. "The Syntax of CycL." 2002. <http://www.cyc.com/cycdoc/ref/cycl-syntax.html>.

10: Eubanks, Brian. "The Jorne Project." 2005. <http://jorne.sourceforge.net/>.

11: Ford, Bryan. *Parsing Expression Grammars: A Recognition-Based Syntactic Foundation*. Massachusetts Institute of Technology. Venice, Italy: Proceedings of Symposium on Principles of Programming Languages, 2004.

12: Gomez, Fernando. *Grounding the Ontology on the Semantic Interpretation Algorithm*. Masaryk University. Orlando, FL: Proceedings of GWC 2004, University of Central Florida, 2003. <http://www.fi.muni.cz/gwc2004/proc/90.pdf>.

13: Henning, Jeffery. “Conlang Profiles: Logical.” 1996-2005. <http://www.langmaker.com/db/condir_logical.htm>.

14: Hirst, Graeme. *Ontology and the Lexicon.* Toronto, Ontario: University of Toronto, 2003. <http://www.cs.toronto.edu/pub/gh/Hirst-Ontol-2003.pdf>.

15: Jacobs, Ian. “About W3C.” 2005. <http://www.w3.org/Consortium/>.

16: Janton, Pierre. Tonkin, Humphrey, ed. *Esperanto: Language, Literature, and Community*. Albany, NY: State University of New York Press, 1993.

17: Kay, Paul, and Willett Kempton. *What is the Sapir-Whorf Hypothesis*. Tuscon, AZ: University of Arizona Press, 1983. <http://www.u.arizona.edu/~avf/ENGL620/docs/Kay_Kempton.pdf>.

18: LeChevalier, Robert. “Lojban - The Logical Language.” 2002. <http://www.lojban.org/files/brochures/lojbroch.html>.

19: Levin, Lori, et al. *An Interlingua Based on Domain Actions for Machine Translation of Task-Oriented Dialogues.* Sydney, Australia: Proceedings of ICSLP-98, 1998. <http://www-2.cs.cmu.edu/ afs/cs/user/alavie/www/papers/ICSLP-98-cstar.pdf>

20: Logical Language Group. “Lojban Formal Grammars.” 2005. <http://lojban.org/publications/formal_grammars.html>.

21: ---. “Official Baseline Statement.” 2005. <http://www.lojban.org/llg/baseline.html>.

22: ---. “Official LLG Committees Page.” 2005. <http://lojban.com/llg/committees.html>.

23: ---. “Official Publications of the Logical Language Group.” 2005. <http://lojban.com/publications.html>.

24: Mahesh, Kavi. *Ontology Development for MT: Ideology and Methodology*. Madrid Spain: Computing Research Laboratory, New Mexico State University, 1996. <http://www.fdi.ucm.es/profesor/vaquero/DOCT/MikroKosmosOnto (MCCS-96-292).pdf>.

25: Miller, Eric, and Jim Hendler. “Web Ontology Language.” 2005. <http://www.w3.org/2004/OWL>.

26: Miller, Eric, Ralph Swick, and Dan Brickley. “Resource Description Framework (RDF).” 2004. <http://www.w3.org/RDF/>.

27: Nicholas, Nick, and John Woldemar Cowan. *What is Lojban?* Fairfax, VI: The Logical Language Group, 2003.

28: Niles, Ian, and Adam Pease. *Towards a Standard Upper Ontology*. Palo Alto, CA: Teknowledge Corporation, 2001. <http://projects.teknowledge.com/HPKB/Publications/FOIS.pdf>.

29: Okrand, Marc. *Klingon for the Galactic Traveler*. New York: Simon & Schuster, 1997.

30: Pease, Adam. "Ontology Portal – FAQ." 2005. <http://www.ontologyportal.org/FAQ.html>.

31: Powell, Robin Lee. "Issues with the Lojban Formal Grammar." <http://www.digitalkingdom.org/~rlpowell/hobbies/lojban/grammar/>.

32: Quin, Liam. "Extensible Markup Language (XML)." 2005. <http://www.w3.org/XML/>.

33: Salo, David. *A Gateway to Sindarin.* Salt Lake City, UT: University of Utah Press, 2004.

34: Schoening, James. "Standard Upper Ontology Working Group (SUO WG)." 2003. <http://suo.ieee.org>.

35: Speer, Rob and Catherine Havasi. *Meeting the Computer Halfway*. Cambridge, MA: Student Oxygen Workshop, Massachusettes Institute of Technology, 2004. <http://sow.lcs.mit.edu/2004/proceedings/Speer.pdf>.

36: Sperberg-McQueen. C. M. and Henry Thompson, "XML Schema." 2000. <http://www.w3.org/XML/Schema>.

37: Standard Upper Ontology Working Group (SUO WG). "Cumulative Resolutions." 2003. <http://suo.ieee.org/SUO/resolutions.html>.

38: Vossen, Piek and Christiane Fellbaum. "The Global Wordnet Association." <http://www.globalwordnet.org/>.

39: Web Ontology Working Group. "OWL Web Ontology Use Cases and Requirements." 2003. <http://www.w3.org/TR/2003/CR-webont-req-20030818/>.