Ontological Semantics

by

Sergei Nirenburg and Victor Raskin

Draft: Do Not Quote or Distribute

Preface

This book introduces ontological semantics, a comprehensive approach to the treatment of text meaning by computer. Ontological semantics is an integrated complex of theories, methodologies, descriptions and implementations. In ontological semantics, a **theory** is viewed as a set of statements determining the format of **descriptions** of the phenomena with which the theory deals. A theory is associated with a **methodology** used to obtain the descriptions. **Implementations** are computer systems that use the descriptions to solve specific problems in text processing. Implementations of ontological semantics are combined with other processing systems to produce applications, such as information extraction or machine translation.

The theory of ontological semantics is built as a society of microtheories covering such diverse ground as specific language phenomena, world knowledge organization, processing heuristics and issues relating to knowledge representation and implementation system architecture. The theory briefly sketched above is a top-level microtheory, the ontological semantics theory *per se*. Descriptions in ontological semantics include text meaning representations, lexical entries, ontological concepts and instances as well as procedures for manipulating texts and their meanings. Methodologies in ontological semantics are sets of techniques and instructions for acquiring and manipulating knowledge as well as for running specific applications.

Ontological semantics is not a finished product. It is constantly evolving: new implementations are developed in response to needs for enhanced coverage and utility. Some such needs arise from the internal logic of the field; some others are due to the requirements of practical applications. In any case, ontological semantics is driven by the necessity to make the meaning manipulation tasks, such as text analysis, text generation and reasoning over text meaning representations, work. As a result, our approach places no premium on using a single method, engine or formalism in developing the procedures themselves or acquiring the data that support them. Instead, ontological semantics develops a set of heterogeneous methods suited to a particular task and coordinated at the level of knowledge acquisition and runtime system architecture in implementations.

The methodology of ontological semantics has a hybrid character also because it allows for a variable level of automation in all of its processes—both the runtime procedures and the important knowledge acquisition tasks. Asymptotically, all the processes of ontological semantics will become fully automated. However, to make ontological semantics applicable before it reaches this ideal, human participation must be grudgingly and judiciously built in. In the various extant implementations, the runtime procedures have always been automated, while knowledge acquisition has involved a controlled and channeled human effort. The levels of automation will continue to increase across the board as the approach evolves.

It is arguable that the human level of performance in processing language is a goal that is unattainable by computer programs, either today or, quite possibly, ever. This realization may lead some to rejecting this goal and focusing instead on what is perceived as necessary components of a future NLP system. Often, the focus is further narrowed to methods, formalisms and tools and excludes broad descriptions of phenomena or procedures. Ontological semantics takes the opposite view: it considers the development of implementations and comprehensive applications the main challenge of NLP. We fully realize that, at any given time, these implementations fall short

on quality and coverage. While improving specific methods is important, ontological semantics is more interested in developing all the necessary processing and knowledge modules and combining them in a comprehensive system for a class of real-life NLP applications, at the current stage of their attainability.

We appreciate the attraction in setting and reaching attainable local goals, such as the exploration of a single language phenomenon or the perfection of a processing method or a formalism. We are concerned that such efforts are not at all certain to facilitate any future comprehensive systems. When a potential component of a system is developed in isolation, its integrability, if at all considered by the developers, is assumed. Experience in integrating component microtheories in ontological semantics has demonstrated that it is a major resource drain. It follows that minimizing this effort through coordinated development of microtheories is desirable. In practice, of course, there is always a trade-off between importing a microtheory, which would require an integration step, and developing it in house.

Methodological versatility in ontological semantics helps to avoid the fallacy of trying to apply a method of choice to too many tasks. Such misplaced optimism about the utility of any method often results in increased complexity of implementation and/or lower-quality output. In other words, one has to avoid being burned by the old adage, “If all you have is a hammer, everything looks like a nail.” It is a methodological tenet of ontological semantics that every class of phenomena may require a dedicated method. As a result, the approach always addresses options for treatment instead of promulgating the one “correct” way.

Ontological semantics is content-oriented. It puts a premium on acquiring all the knowledge required for the left-hand sides of the many heuristic rules that it uses in processing. Heuristics presuppose abductive reasoning, that is, they are defeasible. The reason for choosing abduction is the realistic expectation that text inputs may at any time violate a recorded constraint in the knowledge base.

The book consists of two parts. In Part I, ontological semantics is positioned vis-a-vis cognitive science and the AI NLP paradigm (Chapter 1), the philosophy of science (Chapter 2), linguistic semantics and the philosophy of language (Chapter 3), computational lexical semantics (Chapter 4) and studies in formal ontology (Chapter 5). Part II describes the content of ontological semantics. Chapter 6 defines and discusses text meaning representation as a process and as a structure. Chapter 7 is devoted to the static knowledge sources in ontological semantics—the ontology, the fact database, the lexicon and the onomasticon. Chapter 8 sketches the ontological semantic processes involved in text analysis. Chapter 9 deals with acquisition of static knowledge in ontological semantics. The content of the various chapters is highly interrelated, which results in a large number of cross-references.

We believe that the book will be of interest to a variety of scholars and practitioners in our field and adjacent areas. NLP specialists and computational linguists will find here a variety of proposals for computational treatment of specific language phenomena; for the content and format of knowledge resources and their acquisition; and for integration of microtheories into a single implementation system. For AI specialists and cognitive scientists, ontological semantics can be seen as a realization and a practical argument for knowledge-based processing in general, for

example, within the model of an intelligent agent; it also provides a detailed instance of a complex and multifaceted knowledge base augmented with browsing, acquisition and testing tools. Theoretical linguists (especially semanticists) and the philosophers of language will benefit from exposure to a number of suggested solutions for natural language meaning representation and processing. Descriptive linguists may find specifications for convenient tools to enhance the efficiency of their work. The philosophers of science may find the discussion of the philosophy of linguistics a useful case study for their deliberations. Cognitive psychologists and psycholinguists may wish to consider whether our model of language processing may have any validity for humans; additionally, our system may be considered as a substrate for psychological and psycholinguistic experimentation in human language processing. Specialists in human factors may be interested in the particulars of the division of labor between humans and computers that ontological semantics proposes for knowledge acquisition.

In the area of linguistic engineering and NLP applications, this book may provide a variety of workers with ideas about content and structure of static knowledge sources, about the knowledge requirements of various processors and complete applications, about practical descriptions of particular phenomena, about organizing the knowledge acquisition effort and about integrating comprehensive systems. We also hope that this book will help the practitioners to realize better that: a) treatment of meaning is a *sine qua non* for attaining a new level of quality in practical applications and that the rather steep price for its inclusion is well worth paying; and b) that the crucial component of success of large applications is content, not formalism.

We would like to express our gratitude to our colleagues who, over the years, have contributed to the various implementations and applications of ontological semantics. Allen B. Tucker worked with us on an early conception of knowledge-based machine translation. James H. Reynolds and Irene B. Nirenburg worked on the POPLAR planning application. James Pustejovsky contributed to an early formulation of the microtheory of aspect. Lori Levin collaborated with us on the issue of syntax-driven and ontology-driven semantics. Ira Monarch and Todd Kaufmann were instrumental in building the first acquisition environment for the ontology and an initial formulation of its top levels. Lynn Carlson helped to research the first guidelines of ontological modeling and contributed to the development of an early version of the ontology itself. Salvatore Attardo and Donalee H. Attardo analyzed and catalogued contributions of linguistic semantics to computational applications. Manfred Stede provided programming support for this effort. Ralf Brown helped with a variety of tools and conceptual and procedural support for knowledge acquisition and representation; in particular, he formulated an early version of set notation for ontological semantics. Ingrid Meyer and Boyan Onyshkevych, together with Lynn Carlson, produced an early statement about lexicon structure and content. Christine Defrise contributed to the specification of the format and content of text meaning representation. Ted Gibson implemented the morphological and syntactic analyzers as parts of the Dionysus implementation of ontological semantics. Eric Nyberg implemented the underlying knowledge representation language in which the original ontology was formulated, as well as contributing, with John Leavitt, to the development of the text generator module inside Dionysus. In the Mikrokosmos implementation of ontological semantics, Kavi Mahesh was responsible for the development of the ontology and shared with Steve Beale the work on the semantic analyzer. Steve also worked on the control structure of the implementation as well as on the generation component. Boyan Onyshkevych developed an algorithm for finding the optimum paths between concepts in the ontology, used as the basis of disam-

biguation in analysis. Evelyne Viegas, Lori Wilson and Svetlana Sheremetyeva provided management and acquirer training support for the knowledge acquisition effort in the Mikrokosmos and CAMBIO/CREST implementations. Eugene Ludovik and Valery Sibirtsev worked on the version of the semantic analyzer for the CAMBIO/CREST implementation. Spencer B. Koehler was responsible for the acquisition tools in this implementation as well as for managing the actual acquisition of the fact database and leading the development of the question answering application in CREST.

We have profited from many discussions, some of them published, of issues in and around ontological semantics with Yorick Wilks, who also read and commented on parts of the manuscript. James Pustejovsky and Graeme Hirst have made useful comments on some of the ideas in the book. Jim Cowie has imparted to us a great deal of his realistic view of NLP in the course of many formal and casual discussions.

There are many people who have, over the decades, have been a source of inspiration and admiration for both or either of us. We have both learned from Igor Melcuk's staunch and fearless refusal to conform to the dominant paradigm as well as his encyclopedic knowledge of linguistics, and ability to mount a large-scale and relentless effort for describing language material. We have always admired Charles Fillmore for having never abandoned an interest in meaning, never sacrificing content for formalism and never refusing to meet the complexities of semantic description head on. We are also grateful to Allen B. Tucker who was a great co-author and enthusiastic supporter in the early days of our joint work in NLP, even before we knew that what we were doing was ontological semantics. We greatly appreciated Jim McCawley's iconoclastic presence in linguistics and mourn his premature death. We agreed early on that Paul M. Postal's treatment of semantic material in his 'remind' article was the early benchmark for our own descriptive work. Roger Schank, a major representative of the 'scruffy' AI tradition of concentrating on the semantic content (no matter how limited the coverage) rather than the formalism, was an important influence. Over the years, we have greatly enjoyed Yorick Wilks' encyclopedic knowledge of philosophy, AI and linguistics as well as his general erudition, so rare in our technocratic times, his energy, his style of polemics, his ever present wit and his friendship. Victor Raskin is forever grateful for the privilege of having worked with Vladimir A. Zvegintsev and Yehoshua Bar Hillel. Sergei Nirenburg would like to thank Victor Lesser for the early encouragement, for the many lessons in how to think about scientific problems, for warmth and wisdom.

I. About Ontological Semantics

1. Introduction to Ontological Semantics

Ontological semantics is a theory of meaning in natural language and an approach to natural language processing (NLP) which uses a constructed world model, or ontology, as the central resource for extracting and representing meaning of natural language texts, reasoning about knowledge derived from texts as well as generating natural language texts based on representations of their meaning. The architecture of an archetypal implementation of ontological semantics comprises, at the most coarse-grain level of description:

- a set of static knowledge sources, namely, an **ontology**, a **fact database**, a **lexicon** connecting an ontology with a natural language and an **onomasticon**, a lexicon of names (one lexicon and one onomasticon are needed for each language);
- **knowledge representation languages** for specifying meaning structures, ontologies and lexicons; and
- a set of processing modules, at the least, a semantic **analyzer** and a semantic **text generator**.

Ontological semantics directly supports such applications as machine translation of natural languages, information extraction, text summarization, question answering, advice giving, collaborative work of networks of human and software agents, etc. For applications other than machine translation, a reasoning module is added that manipulates meaning representations produced by the analyzer to generate additional meanings that can be recorded in the fact database and/or serve as inputs to text generation for human consumption.

Any large, practical, multilingual computational linguistic application requires many knowledge and processing modules integrated in a single architecture and control environment. For maximum output quality, such comprehensive systems must have knowledge about speech situations, goal-directed communicative actions, rules of semantic and pragmatic inference over symbolic representations of discourse meanings and knowledge of syntactic, morphological and phonological/graphological properties of particular languages. Heuristic methods, extensive descriptive work on building world models, lexicons and grammars as well as a sound computational architecture are crucial to the success of this overall paradigm. Ontological semantics is responsible for a large subset of these capabilities.

The above generalized application architecture also includes an “ecological,” morphological and a syntactic component, both in the analysis and the generation processes. While realizing the ontological semantic model in applications, such components have been usually developed quite independently of the central ontological semantic component, though the knowledge required for them was often (though not in every implementation) integrated in the overall system lexicons. Thus, for instance, grammar formalisms have remained outside the immediate scope of theoretical work on the ontological semantic model, and indeed several different grammar formalisms have been used to support analysis and generation in the different implementations. Due to this state of affairs, we do not include grammar formalisms and actual rule sets in the core knowledge sources of the model. The interaction between the ontological semantic processing and the rest of

the processing takes place in actual implementations through the specification of the content of the input structures to semantic analyzer and output structure of the semantics-based sentence planner module of the generator.

Historically, ontological semantics has been implemented in several NLP projects, as follows:

Project Name	Content	Dates	Principal Developers	References
Translator	Knowledge-based MT; original formulation	1984-86	Sergei Nirenburg Victor Raskin Allen B. Tucker	Nirenburg *et al*. 1986
Poplar	Modeling intelligent agents	1983-86	Sergei Nirenburg James H. Reynolds Irene Nirenburg	Nirenburg *et al*. 1985; see also Nirenburg and Lesser 1986
KBMT-89	Medium-scale KBMT, Japanese-English	1987-89	Sergei Nirenburg Jaime G. Carbonell Masaru Tomita Lori Levin	Nirenburg *et al*. 1991 Goodman and Nirenburg 1991
Ontos	Ontological modeling and the original acquisition and maintenance toolkit	1988-90	Sergei Nirenburg Ira Monarch Todd Kaufmann Lynn Carlson	Monarch and Nirenburg 1987, 1988 Carlson and Nirenburg 1990
SMEARR	Extension of Ontos; mapping of linguistic semantics into computational semantics	1988-91	Victor Raskin Salvatore Attardo Donalee H. Attardo Manfred Stede	Raskin *et al*. 1994a,b
KIWI	Using human expertise to help semantic analysis	1989-91	Ralf Brown Sergei Nirenburg	Brown and Nirenburg 1990

Project Name	Content	Dates	Principal Developers	References
Dionysus (including DIANA and DIOGENES)	An umbrella project including morphological, syntactic, semantic analysis; text generation and ontological and lexical knowledge acquisition	1989-92	Sergei Nirenburg Ted Gibson Lynn Carlson Ralf Brown Eric Nyberg Christine Defrise Stephen Beale Boyan Onyshkevych Ingrid Meyer	Monarch *et al*. 1989; Nirenburg 1989a,b; Nirenburg *et al*. 1989; Nirenburg and Nyberg 1989; Defrise and Nirenburg 1990a,b; Nirenburg and Goodman 1990; Onyshkevych and Nirenburg 1991; Nirenburg and Levin 1991; Meyer *et al*. 1990
Pangloss	another KBMT application, Spanish-English; hybrid system including elements of ontological semantics	1990-95	Jaime G. Carbonell Sergei Nirenburg Yorick Wilks Eduard Hovy David Farwell Stephen Helmreich	Nirenburg 1994; Farwell *et al*. 1994
Mikrokosmos	Large-scale KBMT; Spanish, English, Japanese; first comprehensive implementation of ontological semantics	1993-99	Sergei Nirenburg Victor Raskin Kavi Mahesh Steven Beale Evelyne Viegas Boyan Onyshkevych	Beale *et al*. 1995; Mahesh and Nirenburg 1995; Mahesh *et al*. 1997a,b; Nirenburg *et al*. 1995; Onyshkevych and Nirenburg 1995; Raskin and Nirenburg 1995; Mahesh 1996; Nirenburg *et al*. 1996; Beale 1997;
PAWS	A patent authoring workstation	1996-97	Svetlana Sheremetyeva Sergei Nirenburg	Sheremetyeva and Nirenburg 1996
Savona	a mixed human-computer agent network for generating reports about emerging crises	1997-98	Sergei Nirenburg James Cowie Steven Beale	Nirenburg 1998a
MINDS	Intelligent information extraction	1998-2000	James Cowie William Ogden Sergei Nirenburg	Ludovik *et al*. 1999; Cowie *et al*. 2000a,b; Cowie and Nirenburg 2000

Project Name	Content	Dates	Principal Developers	References
Expedition	Semi-automatic environment for configuring MT systems and language knowledge to support them	1997 -	Sergei Nirenburg Victor Raskin Marjorie McShane Ron Zacharski James Cowie Rémi Zajac Svetlana Sheremetyeva	Nirenburg 1998b; Nirenburg and Raskin 1998; Oflazer and Nirenburg 1999; Sheremetyeva and Nirenburg 2000a, b; Oflazer *et al*. 2001.
CAMBIO	Mikrokosmos-lite	1999-	Sergei Nirenburg Spencer B. Koehler	Nirenburg 2000a
CREST	Question answering	1999 -	Sergei Nirenburg James Cowie Spencer B. Koehler Eugene Ludovik Victor Raskin Svetlana Sheremetyeva	Nirenburg 2000b

In this book, we will refer to three implementations of ontological semantics, Dionysus, Mikrokosmos and CAMBIO/CREST, which are the major stages in the development of the approach, dating from, roughly, 1992, 1996 and 2000.

Our theoretical work in semantics is devoted to developing a general semantic theory that is detailed and formal enough to support natural language processing by computer. Therefore, issues of text meaning representation, semantic (and pragmatic) processing, the nature of background knowledge required for this processing and the process of its acquisition are among the central topics of our effort. Ontological semantics shares the commitment to these foundational issues with a number of approaches to processing meaning in artificial intelligence, among them conceptual dependency, preference semantics, procedural semantics and related approaches (e.g., Schank 1975, Schank and Abelson 1977, Schank and Riesbeck 1981, Wilensky 1983; Wilks 1975a,b, 1977, 1982; Charniak and Wilks 1976; Woods 1975, 1981, Lehnert and Ringle 1982, Waltz 1982, Charniak 1983a, Hirst 1987). Moreover, the influences go beyond purely computational contributions back to cognitive psychology and cognitive science (Miller and Johnson-Laird 1976, Fodor, Beaver and Garrett 1974; see also Norman 1980). The foundational issues in this research paradigm, in fact, transcend natural language processing. They include the study of other perceptors (e.g., speech, vision) and effectors (e.g., robotic movement, speech synthesis) as well as reasoning (e.g., general problem solving, abductive reasoning, uncertainty and many other issues). Newell and Simon (1972) provide a seminal formulation of the overall paradigm that underlies all the abovementioned work as well as many contributions that were not mentioned (see also Newell *et al*. 1958; Miller *et al*. 1960; Newell 1973; Newell and Simon 1961, 1976; McCarthy and Hayes 1969; McCarthy 1977). This paradigm certainly underlies ontological semantics.

What sets this knowledge-based paradigm apart is the reliance on the glass-box, rather than black-box approach to modeling understanding. In other words, instead of attempting to account for meaning in terms of the fully observable (though, interestingly, not necessarily correctly understood!) phenomena, namely, pairs of inputs and outputs (stimuli and responses, see Section 3.4.1) to a language processor, understood as a black box, these theories aspire to come up with hypotheses about what processes and what knowledge is needed in order to recreate the human ability to process language using computers. This is done by modeling the contents of the black box, necessarily using notions that are not directly observable.

Ontological semantics subscribes to a version of this tenet, the so-called "weak AI thesis" (see Section 2.4.2.2), that avoids the claim that computer programs directly model human semantic capacity. Instead, this hypothesis suggests functional equivalence, that is, that computer programs can attain human-quality results, though not using the exact methods that humans use.

The tenets of ontological semantics are compatible with those of semantic theories developed within the generative paradigm in linguistics (Fodor 1977, see also Section 3.5). There are also important differences, along at least the following two dimensions:

- the purview of the theory (ontological semantics includes all of: lexical and compositional semantics, pragmatics, reasoning); and
- the degree to which the theory has been actually both developed and implemented through language description and computer system construction.

A number of differences exist between the mandates of general semantic theory and semantic theory for NLP. In what follows, we suggest a number of points of such difference (this list is an extension of the discussion in Nirenburg and Raskin 1986; see also Raskin 1990—cf. Chapter 4).

While it is agreed that both general and NLP-related theories must be formal, the nature of the formalisms can be quite different because different types of reasoning must be supported. A general linguistic theory must ensure a complete and equal grain-size coverage of every phenomenon in the language; an NLP-related theory must be sufficiently flexible and robust to adapt to the purposes of any application. The ultimate criterion of validity for a general linguistic theory is explanatory adequacy; for an NLP-related theory, it is the success of the intended applications. A general linguistic theory can avoid complete descriptions of phenomena once a general principle or method has been established: a small number of clarification examples will suffice for its purposes. In NLP, the entire set of phenomena present in the sublanguages of applications must be covered exhaustively. A general linguistic theory has to be concerned about the boundary between linguistic and encyclopedic knowledge. This distinction is spurious in NLP-oriented semantic theories because in order to make semantic (and pragmatic) decisions, a system must have access equally to both types of data (Raskin 2000).

While a general linguistic theory can be method-driven, that is, seek ways of applying a description technique developed for one phenomenon in the description of additional phenomena (this reflects the predominant view that generalization is the main methodology in building linguistic theories), an NLP-related theory should be task-driven—which means that adequacy and efficiency of description takes precedence over generalization (Nirenburg and Raskin 1999).

1.1 A Model of Language Communication Situation for Ontological Semantic Theory

Ontological semantics, as a mentalist approach to building NLP-related language processing theories, is centered around the metaphor of the model of an intelligent agent.[1] An NLP-related theory must account for such properties of intelligent agents as goal- and plan-directed activity, of which language activity is a part—verbal actions, together with perceptual, mental and physical actions, comprise the effector inventory of an intelligent agent. It must also take into account the knowledge of the agent's attitudes to the entities in the world model as well as to remembered instances of events and objects in its own episodic memory. Not only are these attitudes often the subject of a discourse but they also influence the form of discourse on other topics.

Building nontrivial natural language processing systems that manipulate meaning is best done using the metaphor of modeling intelligent agents immersed in a language communication situation. In other words, we prefer to ground our meaning representation theory on cognitive premises rather than on purely logical ones. In most basic and simplified terms, we define our model of an intelligent agent as follows. An intelligent agent is a member of a society of intelligent agents. The agent's actions are goal-directed. It is capable of perception, internal symbol manipulation and action. Its actions can be physical, mental or communicative. The communicative actions are used for communicating with other agents. An agent's perceptual mechanism is a model of the perceptual mechanism of humans. The peculiarities of the perception and action sides of the agent are less central to a discussion of ontological semantics, so we will concentrate on the agent's resident knowledge and the processing environment for the treatment of natural language.

We model the communication situation as follows. It involves at least two intelligent agents—a discourse (text, speech) producer and a discourse consumer. The communication situation also involves the discourse itself, in our case, a text. More precisely (though this is not a crucial distinction from the standpoint of text processing), discourse producer and consumer are roles played by intelligent agents, as each agent can play any of these roles at different times. The message conveyed by a text can be viewed as an action which the discourse consumer perceives as a step in a discourse producer's plan to achieve one of his or her active goals.[2] These plans take into account the knowledge the producer has (or assumes it has) about the target audience. A theory of discourse goals must, therefore, follow the prior introduction of a model of a participant in a language communication situation.

1.1.1 Relevant Components of an Intelligent Agent's Model

The following components in an agent's model are relevant for its language processing ability:[3]

- Knowledge about the world, which we find useful to subdivide into:
  – an ontology, which contains knowledge about types of things (objects, processes, prop-

1. This assumption follows the well-established views of Newell and Simon (1972) and Miller and Johnson-Laird (1976).
2. It is the presence of the notions of goal and plan that makes this communication model nontrivially distinct from Jakobson's (1960) original speaker-message-hearer scheme, a linguistic adaptation of Shannon and Weaver's (1949) classical model.
3. Agent models for other AI applications, such as general planning and problem solving, may require additional facets and not need some of the facets we list.

erties, intentions) in the world; and
  – a fact database, an episodic memory module containing knowledge about instances (tokens) of the above types and about their combinations; a marked recursive subtype of this knowledge is a set of mental models of other agents (see, for instance, Ballim and Wilks 1991, for an analysis of the "artificial believers"), complete with their own components—these models can be markedly different from the "host" model;
- Knowledge of natural language(s), including, for each language:
  – ecological, phonological, morphological, syntactic and prosodic constraints;
  – semantic interpretation and realization rules and constraints, formulated as mappings between lexical units of the language and elements of the world model of the producer;
  – pragmatics and discourse-related rules that map between modes of speech and inter-agent situations, on the one hand, and syntactic and lexical elements of the meaning representation language, on the other;
- Emotional states that influence the "slant" of discourse generated by an agent (Picard 2000)
- An agenda of active goal and plan instances (the intentional plane of an agent).

1.1.2 Goals and Operation of the Discourse Producer

The discourse producer goals will be formulated in terms of these different components. Thus, a producer may want to achieve the following types of inter-agent communicative goals:

1. Modify the discourse consumer's ontology, for example, by giving a definition of a concept.
2. Modify the discourse consumer's episodic memory, for example, by stating a fact, describing an object or relating an event.
3. Modify the discourse consumer's model of the producer, for example, by expressing its attitude towards some fact (e.g., *Unfortunately, Peter will come too*).
4. Modify the discourse consumer's attitudes to facts of the world.
5. Modify the discourse consumer's agenda, for example, by threatening, giving an order or asking a question.
6. Modify the discourse consumer's emotional state.

A discourse producer can achieve these goals by choosing not only what to say, but also how to say things. Usually, one element of discourse will achieve several goals at the same time. For instance, if the producer has any authority over the hearer, the fact of simply stating its own opinion about a fact (a goal of Type 3) may very well affect the hearer's opinions, thus achieving a goal of Type 4 (e.g., Wilensky 1983). Goal types are represented in the world model of an agent as postconditions (effects) of complex events (see Carlson and Nirenburg 1990, for the description of the formalism and the motivation behind it; cf. Section 7.1.5).

The producer's processing during generation can be sketched as follows. Given an input stimulus, the producer will activate a goal, choose a rhetorical plan to realize that goal and generate a text. This is done with the help of its knowledge about the world, about the consumer, about the target language (at both the sentence and the discourse level), and the relevant pragmatic constraints.

1.1.3 Operation of the Discourse Consumer

The discourse consumer's processing during analysis can be very roughly sketched as follows. Given an input text, the consumer must first attempt to match the lexical units comprising the text,

through the mediation of a special lexicon, with elements in the consumer's model of the world. To facilitate this, it will have to analyze syntactic dependencies among these units and determine the boundaries of syntactic constituents. The next step is filtering out unacceptable candidate readings through the use of selectional restrictions, collocations and special heuristics, stored in the lexicon. The consumer must then also resolve the problems of co-reference by finding referents for pronouns, other deictic lexical units and elliptical constructions. Furthermore, information on text cohesion and producer attitudes has to be determined, as well as, in some applications, the goals and plans that lead the producer to produce the text under analysis.

Many additional processes are involved in interpretation. A semantic theory for natural language processing must also account for their interaction in a computational model, that is, the overall architecture and control of the semantic and pragmatic interpretation process. Control considerations, we believe, must be an integral part of semantic theories for natural language processing, of which ontological semantics is an example. However, many of the current semantic theories, notably those relying on unification as the main processing method, essentially relinquish control over control. A whole dimension of modeling is thus dispensed with, leading to reduction in expressive power of a theory and extra constraints on building applications. Why not accept unification as one of a number of possible control structures? And, for every processing module, choose a control structure most responsive to the peculiarities of the phenomenon which is treated? In AI, there is a long tradition of looking for the most appropriate representation of a problem, which will "suggest'" the most appropriate algorithm for processing it. It is clear that different representations must be preferred for different problems. (see, e.g., Newell and Simon 1972) Adopting a single type of representation and a single control method for all tasks means putting method before phenomena.

1.2 Ontological Semantics: An Initial Sketch

As any semantic theory for natural language processing, ontological semantics must account for the processes of generating and manipulating text meaning. An accepted general method of doing this is to describe the meanings of words and, separately, specify the rules for combining word meanings into meanings of sentences and, further, texts. Hence the division of semantics into lexical (word) semantics and compositional (sentence) semantics. Semantics for NLP must also address issues connected with the meaning-related activities in both natural language understanding and generation by a computer. While the semantic processing for these two tasks is different in nature—for instance, understanding centrally involves resolution of ambiguity while generation deals with resolution of synonymy for lexical selection—the knowledge bases, knowledge representation approaches and the underlying system architecture and control structures for analysis and generation can be, to a realistic degree, shared. This view is a departure from our earlier views (Nirenburg and Raskin 1987a,b), brought about by practical experience in description and implementation of non-toy applications.

In ontological semantics, the meaning representation of a text is derived through:

- establishing the lexical meanings of individual words and phrases comprising the text;
- disambiguating these meanings;
- combining these meanings into a semantic dependency structure covering

  – the propositional semantic content, including causal, temporal and other relations among individual statements;
  – the attitudes of the speaker to the propositional content; and
  – the parameters of the speech situation;
- filling any gaps in the structure based on the knowledge instantiated in the structure as well as on ontological knowledge.

It is clear from the above description that ontological semantics incorporates the information that in some approaches (e.g., Lascarides 1995, Asher and Lascarides 1995) has been delegated to pragmatics.

The final result of the process of text understanding may include some information not overtly present in the source text. For instance, it may include results of reasoning by the consumer, aimed at filling in elements required in the representation but not directly obtainable from the source text. It may also involve reconstructing the agenda of rhetorical goals and plans of the producer active at the time of text production and connecting its elements to chunks of meaning representation.

Early AI-related natural language understanding approaches were criticized for not paying attention to the halting condition on meaning representation (a criticism of the same kind as Weinreich's attack on Katz and Fodor, see Section 9.3.5). The criticism was justified to the extent that these approaches did not make a very clear distinction between the information directly present in the text and information retrieved from the understander's background knowledge about the entities mentioned in the text. This criticism is valid when the program must apply all possible inferences to the results of the initial representation of text meaning and not when a clear objective is present, such as resolution of ambiguity relative to a given set of static knowledge sources, beyond which no more processing is required.

It follows that text meaning is, on this view, a combination of

- the information directly conveyed in the NL input;
- the (agent-dependent and context-dependent) ellipsis-removing (lacuna filling) information which makes the input self-sufficient for the computer program to process;
- pointers to any background information which might be brought to bear on the understanding of the current discourse,
- records about the discourse in the discourse participants' fact database.

Additionally, text understanding in this approach includes detecting and representing a text component as an element of a script/plan (in Schank-Abelson-Cullingford-Wilensky's terms—see Schank and Abelson, 1977, Cullingford, 1981, Wilensky, 1983, see also Section 7.1.5) or determining which of the producer goals are furthered by the utterance of this text component. We stop the analysis process when, relative to a given ontology, we can find no more producer goals/plans which can be furthered by uttering the sentence. But first we extract the propositional meaning of an utterance using our knowledge about selectional restrictions and collocations among lexical units. If some semantic constraints are violated, we turn on metonymy, metaphor and other "unexpected" input treatment means. After the propositional meaning is obtained, we actually proceed to determine the role of this utterance in script/plan/goal processing. In doing so, we extract speech act information, covert attitude meanings, and eventually irony, lying, etc. The extant

implementations of ontological semantics make no claim about including all these features.

There is a tempting belief among applied computational semanticists that in a practical application, such as MT, the halting condition on representing the meaning of an input text can, in many cases, be less involved than the general one. The reason for this belief is the observation that, when a target language text is generated from such a limited representation, one can, in many cases, expect the consumer to understand it by completing the understanding process given only partial information. Unfortunately, since, without human involvement, there is no way of knowing whether the complete understanding is, in fact, recoverable by humans, it is, in the general case, impossible to posit a shallower (and hence more attainable) level of understanding. To stretch the point some more, humans can indeed correctly guess the meaning of many ungrammatical, fragmentary and otherwise irregular texts (e.g., Charniak's (1983b:159) example of "lecture, student, confusion, question"). This, however, does not mean that an automatic analyzer, without specially designed extensions, will be capable of assigning meanings to such fragments—their semantic complexity is of the same order as that of "regular" text.

1.3 Ontological Semantics and Non-Semantic NLP Processors

Ontological semantics takes care of only a part, albeit a crucial part, of the operation of the major dynamic knowledge sources in NLP, the analyzer and the generator. These processors also rely on syntactic, morphological and "ecological" information about a particular language. Syntactic processing establishes the boundaries and nesting of phrases in the text and the dependency structures at the clause and sentence levels by manipulating knowledge about word order and grammatical meanings carried by lexical items. Morphological processing establishes the grammatical meanings carried by individual words, which helps the syntactic processor to decide on types of grammatical agreement among the words in the sentence, which, in turn, provides heuristics for determining syntactic dependencies and phrase boundaries. The "ecology" of a language (Don Walker's term) includes information about punctuation and spelling conventions, representation of proper names, dates, numbers, etc.

Historically, the integration of all these steps of processing into a single theory and system has been carried out in a variety of ways. Thus, the Meaning↔Text model (Apresyan *et al.* 1969, 1973, Mel'čuk 1974, 1979) dealt with most of these levels of processing and representation at a finer grain size. However, that approach did not focus on semantic representation, and its computational applications (e.g., Kittredge *et al.* 1988) did not address semantics at all, concentrating instead on deep and surface syntax and morphology. Conceptual dependency (Schank 1975) did concentrate on semantic representations but neglected to consider syntax or morphology as a separate concern: most of the application programs based on Conceptual dependency (and all the early ones) simply incorporated a modicum of treatment of syntax and morphology in a single processor (e.g., Riesbeck 1975, Cullingford 1981). Ontological semantics, while concentrating on meaning, enters into a well-defined relationship with syntactic, morphological and ecological processing in any application.

The most immediate and important element supporting the relations between ontological semantics and the non-semantic components of an NLP system is content of those zones of the ontological semantic lexicon entry that support the process of linking syntactic and semantic

dependencies (see Section 7.3). Specifically, what is linked is the syntactic dependency and the semantic dependency on clause and phrase heads. This essentially covers all words in a language that take syntactic arguments, which suggests that their meanings are predicates taking semantic arguments. The dynamic knowledge sources use this information to create and/or manipulate a text meaning representation (TMR). The dynamic knowledge sources, however, also use morphological, syntactic and other non-semantic information in their operation.

1.4 Architectures for Comprehensive NLP Applications

The ideal state of affairs in NLP applications (as in all the other complex multi-module software systems) is when each component produces a single, correct result for each element of input. For example, a morphological analyzer can produce a single citation form with a single set of inflectional forms for a given input word, e.g., given the English *lain* it produces "*lie;* Verb, Intransitive, past participle; *'*be prostrate,'" while disambiguating it at the same time from "*lie* 'to make an untrue statement with intent to deceive.'"

Unfortunately, this state of affairs does not always hold. For example, given the Russian *myla* as input, a Russian morphological analyzer will (correctly!) produce three candidate outputs:

1. *mylo* 'soap'; Noun, Neuter, Genitive, Singular;
2. *mylo* 'soap'; Noun, Neuter, Nominative, Plural;
3. *myt'* 'to wash'; Verb, Transitive, Past, Feminine.

In context, only one of the multiple outputs will be appropriate. Conversely, the English morphological analyzer will (correctly!) fail to produce a candidate for the input string *mylo*, as it is not a word in the English language. Or, to use another example, a standard semantic analyzer for English will not be able to interpret the English phrase *kill the project* if the lexicon entry for *kill* (reasonably) lists its meaning as something like "cause not to be alive." Indeed, as projects are not living beings, the combination does not work.

The history of NLP can be viewed as the fight against these two outcomes: underspecification, that is, being unable to cut the number of candidate solutions down to exactly one, and failure to produce even a single candidate solution, due to overconstraining or incompleteness of static knowledge sources. The big problem is that it is difficult, if at all possible, to develop static knowledge sources (lexicons, grammars, etc.) with information that is correct in all contexts that can be attested in running text. Selecting an appropriate computational architecture is one of the methods of dealing with these difficulties as well as of improving the efficiency of the overall process.

1.4.1 The Stratified Model

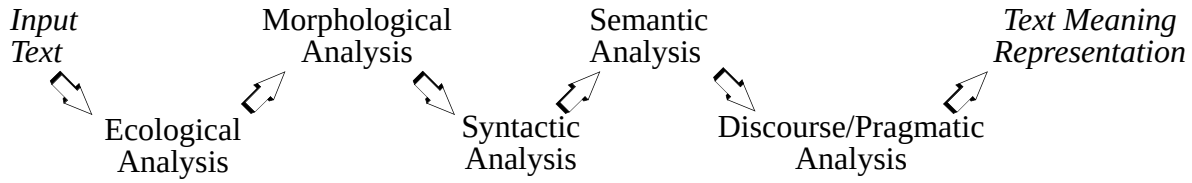

Figure 1. Stratified Model I: Analysis. A schematic view of a traditional pipelined architecture for the analysis module of a comprehensive NLP system (e.g., an MT system). Results of each processing stage are used as input to the next processing stage in the order of application.

The most widely used NLP system architecture conforms to the stratified model (see Figures 1 and 2): the task is modularized, and the modules are run on a text one by one, in their entirety, with the cumulative results of the earlier modules serving as inputs to the later modules. This architecture has been a step forward compared to the early architectures which were not modular in that they heaped all the processing knowledge together rather indiscriminately; see, for instance, the early MT systems or the early AI NLP systems, such as, e.g., Margie (Schank 1975). One of the reasons for introducing the modularity is the difficulty of acquiring static knowledge sources for an "integral" system. Indeed, each of the standard analysis stages—morphology, syntax, semantics and, later, pragmatics and discourse and, still later, ecology—was (and still is) a complex problem which is difficult to study even in isolation, let alone taking into account its connections with other language analysis problems.

It is clear that this architecture was designed for processing without underspecification, overconstraining or knowledge lacunae. Indeed, it presupposes that each module can successfully complete its processing before the later modules take over. While it was not clear what can be done architecturally to counteract possible overconstraining—or other reasons for a failure to find a solution for an element of input, such as lack of necessary background knowledge,—modifications were introduced to the architecture to deal with underspecification.

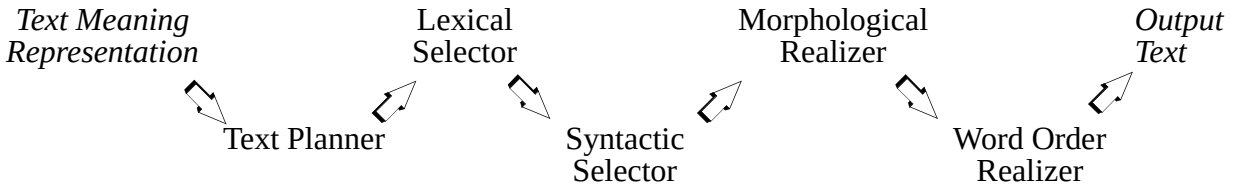

Figure 2. Stratified Model II: Generation. A schematic view of a traditional pipelined architecture for the generation module of a comprehensive NLP system (e.g., an MT system). Results of each processing stage are used as input to the next processing stage in the order of application.

The most prominent deficiency of the strictly pipelined architecture is the systematic insufficiency of knowledge within a single module for disambiguating among several output candidates. To try to alleviate this problem, the basic architecture can be modified by allowing underspecification of the outputs of individual modules, with the exception of the last one. Underspecification,

then, essentially, amounts to postponing decisions of a particular module by allowing it to produce, instead of a single solution, a set of candidate solutions and subsequently using information obtained through the operation of later modules to filter this set (or these sets, if several instances of underspecification occurred). Figure 3 illustrates this kind of architecture for the case of text analysis.

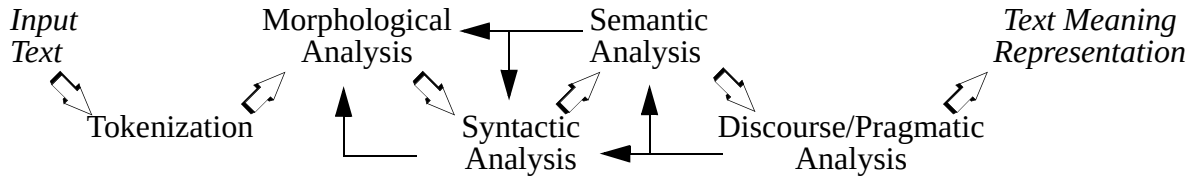

Figure 3. Stratified Model Modified: A schematic view of an enhanced pipelined architecture for the analysis module of a comprehensive NLP system (e.g., an MT system). Thin arrows represent knowledge from later modules which is used to disambiguate results of a prior module.

1.4.2 The "Flat" Model

The stratified architecture of language processing is, in many ways, constraining. Thus, even in the model with feedback, such as that of Figure 3, no use is made of the fact that findings of each of the modules can contribute to text meaning specification directly, not necessarily through the operation of other (later) modules. In addition, the individual results from any module can contribute to determining more than one text meaning element. Conversely, it may be a particular combination of clues from a variety of sources that makes possible the determination of a text meaning element. None of the above is directly facilitated by the stratificational architecture. Underspecification may be difficult to implement efficiently.

A "flat" architectural model (see Figure 4) represents a swing of the pendulum back from pipelining but not back to the lack of modularity. In the flat module, all processing modules operate simultaneously, without waiting for the results of an "earlier" module, e.g., semantic analyzer does not wait till the syntactic analyzer finishes with an input element before starting to work on the latter. Of course, in isolation, the analyzer modules will not be able to complete their processing. However, they will succeed partially. For instance, morphologically uninflected words will be found in the lexicon and the set of their senses instantiated by the semantic processing module irrespective of the results of the syntactic analyzer. If it so happens that only one sense is recorded for a word in the lexicon, this sense becomes a strong constraint which is used to constrain further the realization choices of other text components.

1.4.3 Toward Constraint Satisfaction Architectures

One cannot rely on the partial successes of some modules in an unqualified manner. There are many real-world obstacles for the constraint satisfaction process of this kind. First of all, lexicons can often be incorrect. In particular they may

- contain fewer senses for a word (or a phrase) than necessary for a task; this state of affairs may cause the compositional semantic process of deriving text meaning representation to fail

because of overconstraining—the process may find no candidates that match the constraints specified in the meanings of TMR components with which they must combine; for example, if in a lexicon only the furniture sense is listed for the word *table*, the process will fail on the input *Two last rows of the table had to be deleted*;

- contain more senses for a word (or a phrase) than sufficient for a task; dictionaries compiled for human use typically contain more senses than should be included in the lexicon of an NLP system; thus, for instance, Longman's Dictionary of Contemporary English lists eleven senses of *bank*; should an NLP system use such a dictionary, it will have to be equipped with the means of disambiguating among all these eleven senses, which makes computation quite complex;
- incorrectly interpret the senses or provide incorrect, that is, too relaxed or too strict, constraints on cooccurrence.

While the deficiencies of the lexicon are real and omnipresent in all real-size applications, much more serious difficulties arise from the preponderance in natural language texts, even non-artistic, expository ones, of nonliteral language—metaphors, metonymies and other tropes. In terms of the basic compositional-semantic processing mode, nonliteral language leads to violations of cooccurrence constraints: indeed, you do not really crush your opponent in an argument; or have the orchestra play the composer Bach (e.g., Ballim *et al.* 1991, Martin 1992, see also 8.4.2).

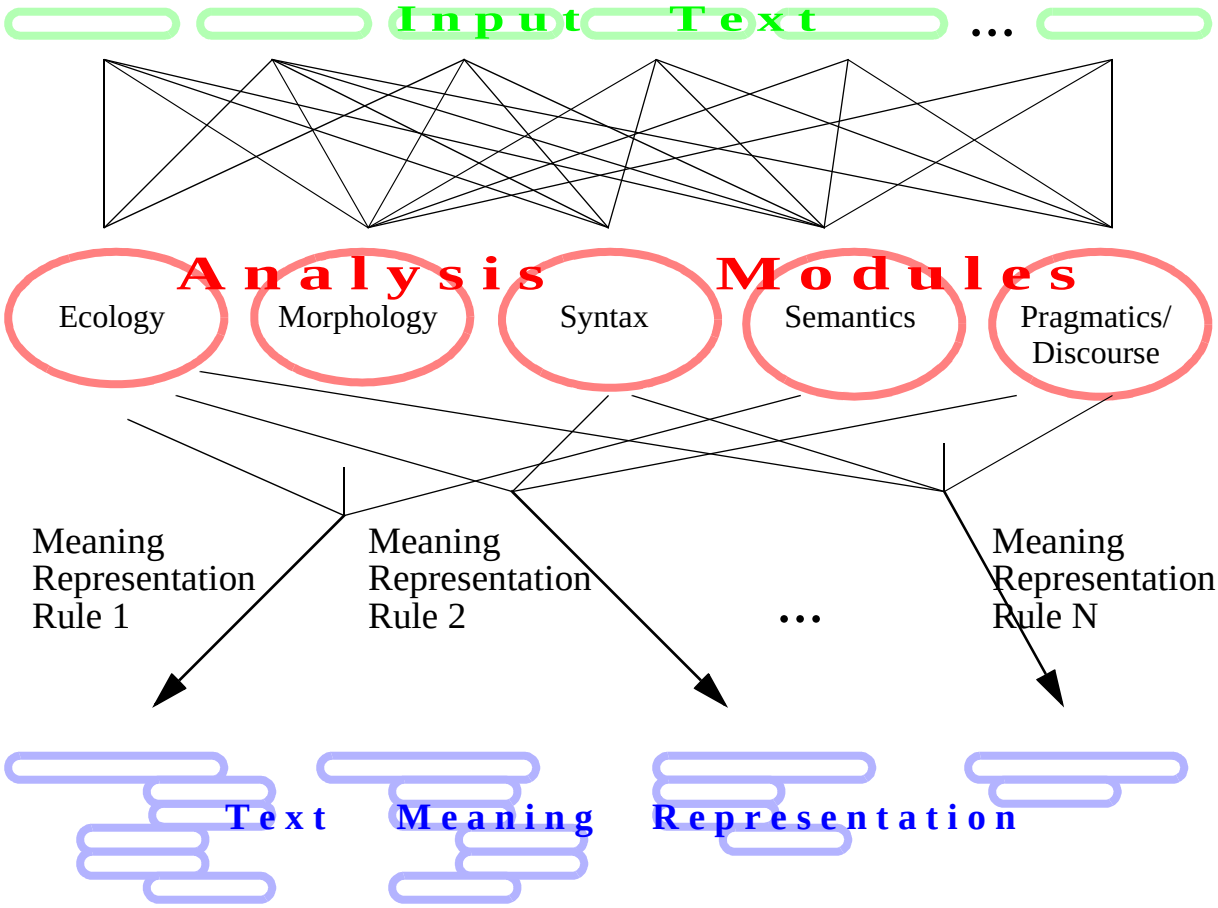

Figure 4. In a "flat" model (illustrated here for the case of text analysis), all modules operate and record partial results simultaneously. An intrinsic ordering remains because "later" modules often need results of "prior" modes to produce results or to disambiguate among candidate analyses. However, partial results are still possible even if an earlier module fails on an element of input. The results of the individual module operation provide clues for the left-hand sides of meaning representation rules. Robustness of the system is further enhanced if the rules are allowed to "fire" even if not all of the terms in their left-hand sides are bound (naturally, this relaxation must be carefully controlled).

One deficiency of the flat model, as sketched above, is that it does not benefit from the intermediate results of its processing, namely, from the availability of the nascent text meaning representation. In fact, intermediate results of analysis, that is, elements of the nascent TMR, can provide reliable clues for the analyzer and must be allowed as constraints in the left hand sides of the text meaning representation rules. Thus, these rules can draw on the entire set of knowledge sources in comprehensive NLP processing: the lexicons, the ontology, the fact database, the text meaning representation and the results of ecological, morphological and syntactic processing. Pragmatics and discourse-related issues are folded in the semantic processing in current implementations of ontological semantics; this, however, is not essential from the theoretical point of view: a single theory covering all these issues can be implemented in more than one application module.

The modified flat model can be realized in practice using the so-called blackboard architecture (e.g., Erman *et al.* 1980, Hayes-Roth 1985), in which a public data structure, a blackboard, is used