Natural Language Processing: A Human–Computer Interaction Perspective

BILL MANARIS

Computer Science Department
University of Southwestern Louisiana
Lafayette, Louisiana

Abstract

Natural language processing has been in existence for more than fifty years. During this time, it has significantly contributed to the field of human-computer interaction in terms of theoretical results and practical applications. As computers continue to become more affordable and accessible, the importance of user interfaces that are effective, robust, unobtrusive, and user-friendly – regardless of user expertise or impediments – becomes more pronounced. Since natural language usually provides for effortless and effective communication in human-human interaction, its significance and potential in human-computer interaction should not be overlooked – either spoken or typewritten, it may effectively complement other available modalities,[1] such as windows, icons, and menus, and pointing; in some cases, such as in users with disabilities, natural language may even be the only applicable modality. This chapter examines the field of natural language processing as it relates to human-computer interaction by focusing on its history, interactive application areas, theoretical approaches to linguistic modeling, and relevant computational and philosophical issues. It also presents a taxonomy for interactive natural language systems based on their linguistic knowledge and processing requirements, and reviews related applications. Finally, it discusses linguistic coverage issues, and explores the development of natural language widgets and their integration into multimodal user interfaces.

Keywords: natural language processing, human-computer interaction, speech recognition, speech understanding, natural language widgets, multimodal user interfaces, user interface development, user interface management systems.

[1] A modality is defined as *channel or form for rendering a thought, concept, or action* (Coutaz and Caelen, 1991).

Appears in *Advances in Computers* (Marvin V. Zelkowitz, ed.), vol. 47, pp. 1-66, Academic Press, New York, 1998.

1. Introduction

The field of natural language processing (NLP) originated approximately five decades ago with machine translation systems. In 1946, Warren Weaver and Andrew Donald Booth discussed the technical feasibility of machine translation "by means of the techniques developed during World War II for the breaking of enemy codes" (Booth and Locke, 1955, p. 2). During the more than fifty years of its existence, the field has evolved from the dictionary-based machine translation systems of the fifties to the more adaptable, robust, and user-friendly NLP environments of the nineties. This evolution has been marked by periods of considerable growth and funding "prosperity," followed by years of intense criticism and lack of funding. This article attempts to provide an overview of this field by focusing on its history, current trends, some important theories and applications, and the state-of-the-art as it relates to human-computer interaction (HCI).

1.1 Overview

Currently, the field of NLP includes a wide variety of linguistic theories, cognitive models, and engineering approaches. Although unrestricted NLP is still a very complex problem (and according to some, an AI-complete problem[2]), numerous successful systems exist for restricted domains of

[2] Similarly to the concept of NP-complete problems, the term AI-complete has been used to describe problems, that can only be solved if a solution to the "general" AI problem has been discovered (Carbonell, 1996). Some argue that such problems are unsolvable (Dreyfus, 1993); others suggest that

discourse. In the context of HCI, NLP applications range from various speech recognition systems, to natural language interfaces to database, expert, and operating systems, to a multitude of machine translation systems. Currently, interactive applications may be classified along the following categories (Manaris and Slator, 1996; Obermeier, 1988):

- Speech Recognition/Understanding and Synthesis/Generation
- Natural Language Interfaces
- Discourse Management, Story Understanding, and Text Generation
- Interactive Machine Translation
- Intelligent Writing Assistants

This article is organized as follows. Initially, it addresses the relationship between natural language processing and human-computer interaction, as well as the potential in the conscious merging of the two fields. It briefly examines the field of linguistics and attempts to provide a concise, yet complete definition for NLP in the context of HCI. It then takes a historical journey through major advances and setbacks in the evolution of the field; specifically, it looks at the early machine translation years, the ALPAC report and its impact, and the field's three major evolutionary phases. Next it examines application areas and gives examples of research and production systems. It presents significant approaches in linguistic modeling, namely symbolic, stochastic, connectionist, and hybrid. Then it attempts to specify a classification for NLP systems in HCI based on depth of linguistic analysis. For each of the categories, it presents examples of related research and development efforts. It discusses linguistic coverage issues, and examines methodologies for developing constrained linguistic models with respect to application domains. Next it looks at natural language as one of the many possible modalities at the user interface and presents multimodal integration issues. Finally, it discusses user interface management systems and development methodologies for effective human-computer interaction with natural language.

1.2 Natural Language and User Interfaces

As the use of computers is expanding throughout society and affecting various aspects of human life, the number and heterogeneity of computer users is dramatically increasing. Many of these users are not computer experts; they are experts in other fields who view the computer as a necessary tool for accomplishing their tasks (Day and Boyce, 1993). Consequently, the user-friendliness and robustness of interactive computer systems is becoming increasingly more essential to user acceptability and overall system performance.

One of the goals of HCI is the development of systems which match (and possibly augment) the physical, perceptual, and cognitive capabilities of users. The field of NLP offers mechanisms for incorporating natural language knowledge and modalities into user interfaces. As NLP tools are becoming more powerful in terms of functionality and communicative capabilities, their contribution to HCI is becoming more significant.

> "Speech is the ultimate, ubiquitous interface. It is how we should be able to interact with computers. The question is when should it begin supplementing the keyboard and mouse? We think the time is now."[3]

---

even partial solutions can be beneficial from both a humanistic and economic perspective (Mostow, *et al.*, 1994; Reddy, 1996).

[3] Bruce Armstrong, Manager of the Speech Technologies Group, WordPerfect, The Novell Applications Group, as quoted in (Markowitz, 1996).

As the relationship between NLP tools and HCI matures, more sophisticated systems are emerging – systems which combine intelligent components with various communicative modalities, such as speech input and output, non-speech audio, graphics, video, virtual environments, and telepresence. As the two fields become more integrated, new developments will make it possible for humans to communicate with machines which emulate many aspects of human-to-human interaction. These new interfaces will transform computers from machines which are visible, and attention-grabbing, to tools that are transparent and embedded; these tools will be so natural to use that they may become part of the context, in such a way that the user may be able focus on the task and not the actual tool (Weiser, 1994; Winograd and Flores, 1986, p. 164).

2. The Field of Natural Language Processing

2.1 An Extended Definition

This section will attempt to answer the question "what is NLP?". As it turns out this is a difficult question to answer as "there are almost as many definitions of NLP as there are researchers studying it" (Obermeier, 1989, p. 9). This is due to the fact that from a linguistics point of view there are many aspects in the study of language; these aspects range from speech sounds (phonetics), to sound structure (phonology), to word structure (morphology), to sentence structure (syntax), to meaning and denotation (semantics), to styles and dialects (language variation), to language evolution, to language use and communication (pragmatics), to speech production and comprehension (psychology of language), to language acquisition, and to language and the brain (neurolinguistics) (Akmajian, *et al.*, 1990). Moreover, there are other disciplines that contribute to NLP, such as philosophy, dealing with questions such as "what is the nature of meaning?" and "how do words and sentences acquire meaning?" (Allen, 1994a).

2.1.1 Linguistics

In order to derive a definition for NLP, one could focus momentarily on the nature of linguistics. Linguistics includes the study of language from prescriptive, comparative, structural, and generative points of view. This study was started at least two thousand years ago by the Sanskrit grammarians (Winograd, 1983); yet we have not made much progress in understanding, explaining, and modeling – the latter being an essential aspect in developing computational entities – this aspect of human existence.

In order to realize the difficulty of the endeavor, as well as the field's fascinating appeal, one could attempt to answer the highly related questions "what is the nature of language?" and "how does communication work?". Some might agree that these questions are equivalent (at least in complexity) to "what is the nature of intelligence?".

The point being made is that such questions may be simply too broad to answer. Yet, it is such questions that fields like linguistics, natural language processing, human-computer interaction, and artificial intelligence are addressing. Although it is quite probable that such questions will never be completely answered,[4] it is through this process of self-study that we have made intriguing

[4] This idea is related to the self-modeling paradox effectively captured by the aphorism: "If the mind was so simple that we could understand it, we would be so simple that we couldn't." Or, as Hofstadter (1979) puts it, "[t]o seek self-knowledge is to embark on a journey which … will always be incomplete, cannot be charted on any map, will never halt, [and] cannot be described" (p. 697).

discoveries and achieved some significant results. The resultant techniques have found their way into commercial and industrial applications, and are quietly changing our lives (Munakata, 1994).

2.1.2 Motivations, Definition, and Scope

There are two motivations for NLP, one scientific and one technological (Allen, 1994a). The *scientific motivation* is to understand the nature of language. Other traditional disciplines, such as linguistics, psycholinguistics, and philosophy, do not have tools to evaluate extensive theories and models of language comprehension and production. It is only through the tools provided by computer science that one may construct implementations of such theories and models. These implementations are indispensable in exploring the significance and improving the accuracy (through iterative refinement) of the original theories and models.

The *technological motivation* is to improve communication between humans and machines. Computers equipped with effective natural language models and processes could access all human knowledge recorded in linguistic form; considering the revolution in information dissemination and communication infrastructure that has been introduced by the World-Wide-Web, one could easily see the importance and potential of such systems. User interfaces with natural language modalities (either input or output, spoken or typewritten) would enhance human-computer interaction by facilitating access to computers by unsophisticated computer users, users in hands-busy/eyes-busy situations (such as car driving, space walking, and air traffic control tasks), and users with disabilities. Actually, the development of this technology for the latter group is motivated by federal legislation and guidelines, such as (a) the US Public Laws 99-506 and 100-542 which mandate the establishment of accessible environments to citizens with disabilities, (b) the 1989 US General Services Administration's guide, *Managing End User Computing for Users with Disabilities*, which describes accommodations for disabled computer users (Shneiderman, 1993), and (c) the 1996 Telecommunication Act. In this context, it does not matter how closely the model captures the complexity of natural language communication; it only matters that the resultant tool performs satisfactorily in a given domain of discourse, or complements/outperforms any alternative solutions. This article adheres to this perspective in presenting and discussing various NLP theories, models, and applications.

In this context, and given the state-of-the-art, NLP could be defined as *the discipline that studies the linguistic aspects of human-human and human-machine communication, develops models of linguistic competence and performance,*[5] *employs computational frameworks to implement processes incorporating such models, identifies methodologies for iterative refinement of such processes/models, and investigates techniques for evaluating the resultant systems*.

NLP is an interdisciplinary area based on many fields of study. These fields include computer science, which provides techniques for model representation, and algorithm design and implementation; linguistics, which identifies linguistic models and processes; mathematics, which contributes formal models and methods; psychology, which studies models and theories of human behavior; philosophy, which provides theories and questions regarding the underlying principles of thought, linguistic knowledge, and phenomena; statistics, which provides techniques for predicting events based on sample data; electrical engineering, which contributes information theory and techniques for signal processing; and biology, which explores the underlying architecture of linguistic processes in the brain.

---

[5] Chomsky (1965) defines *competence* as the linguistic knowledge of fluent speakers of a language, and *performance* as the actual production and comprehension of language by such speakers (Akmajian, *et al.*, 1990).

2.2 Historical Background

This section discusses major milestones in the history of NLP related to human-computer interaction. The information included in this section is somewhat fragmented, due to the non-historical focus of the article, and possibly biased, due to the author's personal path through NLP; the reader is encouraged to follow the references for a more complete historical overview.

2.2.1 The Beginning - Machine Translation

A common misconception is that NLP research started in the late 1960s or early 1970s. Although there was a large body of work that was performed during this period, especially within the symbolic paradigm,[6] research in NLP actually began in the late 1940s with early studies and computational implementations of systems attempting to perform mechanical translations of Soviet physics papers into English (Bar-Hillel, 1960; Slocum, 1986). Specifically, one of the first Machine Translation (MT) projects began in 1946 at the Department of Numerical Automation, Birkbeck College, London, as Warren Weaver and Andrew Donald Booth began work on computational translation based on expertise in breaking encryption schemes using computing devices (Bar-Hillel, 1960; Booth and Locke, 1955; Obermeier, 1988). Specifically, Weaver (1995, p. 18) states:

> When I look at an article in Russian, I say "This is really written in English, but it has been coded in some strange symbols. I will now proceed to decode it".

The work of the Birkbeck group continued until the mid-1960s and made significant contributions to other NLP areas such as preparation of word-indices and concordances (Josselson, 1971). MT research in the US started in 1949 at the University of Washington to be closely followed by The RAND Corporation (1950), MIT (1951), Georgetown University (1952), and Harvard University (1953). As far as other countries are concerned, the USSR joined the MT research effort in 1955, Italy in 1958, and Israel in 1958. As of April 1, 1959, there were eleven research groups in the US, seven in the USSR, and four in other countries employing an estimated 220 full-time researchers and developers (Bar-Hillel, 1960).

2.2.2 The Fall - ALPAC Report

During the first years of MT research, and as considerable progress was being made towards fully-automatic, high-quality translation (FAHQT), even many skeptics were convinced that this goal was indeed attainable, and an operational system was "just around the corner." Significant problems were being solved in quick succession, thus creating the illusion that remaining problems would also be solved quickly. This was not the case, however, as the few remaining problems were the hardest ones to solve (Bar-Hillel, 1960). Consequently, during the 1960s, disillusionment began setting in, as it became apparent that the original enthusiasm and promises for the achievement of FAHQT were unrealistic. Specifically, in 1966 the U.S. National Academy of Sciences sponsored the Automatic Language Processing Advisory Council (ALPAC) Report which, in essence, condemned the MT field, and eventually resulted in the termination of funding for all MT projects in the U.S.

The ALPAC report was criticized by many as narrow, biased, short-sighted, and even as based on inferior analytical work, obsolete and invalid facts, distortion of estimates in favor of human translation, and concealment of data (Josselson, 1971; Slocum, 1986). Perhaps one of the better, and more far-sighted criticisms was the following:

---

[6] The symbolic paradigm is covered in Section 4.1.

> [I]t seems premature to abandon support of machine translation after only 12 brief years, especially if the abandonment is based only on the findings of the ALPAC committee. Even the critics of machine translation admit its contribution to the knowledge of linguists. Who can say what contributions lie ahead? (Titus, 1967, p. 191)

Although the ALPAC report was a major setback in MT, in specific, and NLP, in general, its effects were only temporary. The contributions of early MT research to NLP were significant and long lasting in that they identified new problems for linguists and computer scientists, and thus laid the foundation for subsequent NLP research. This is clearly evident from the areas of emphasis of the pre-ALPAC conferences of federally-sponsored MT groups (Josselson, 1971):

- **Princeton, 1960:** Dictionary design, programming strategies, compatibility/portability of materials, code, and data formats of research groups.
- **Georgetown, 1961:** Grammar coding, automatic conversion of coded materials, investigations of Russian grammar.
- **Princeton, 1962:** Theoretical models for syntactic analysis, consideration of syntax problems in Russian, Arabic, Chinese, and Japanese.
- **Las Vegas, 1965:** Semantic analysis.

This climate of cooperation contributed to the founding of the Association of Computational Linguistics (ACL)[7] in 1962, to the organization of subsequent meetings in the U.S. and elsewhere, and to considerable growth in NLP research and development in both academia and industry throughout the world (Josselson, 1971; Obermeier, 1988; Slocum, 1986).

During these fifty years of development, NLP research has been in a state of constant flux due to shifting goals and objectives, end-user expectations, technical and theoretical advances, predictions on the potential of hypothesis/theory/technique, and heated debates on the ability of computing devices to model certain linguistic phenomena. Possibly, the history of NLP can be seen as evolving through three phases, namely the engineering phase, the theoretical phase, and the user-centered phase. The borderlines among these phases are fuzzy and sometimes overlap. These phases are examined in the next sections.

2.2.3 Engineering Phase

This first phase in NLP started in the mid 1940s and lasted until the early 1960s. It is characterized by an emphasis on algorithms relying mostly on dictionary-lookup techniques used in conjunction with empirical and stochastic methods. Due to the success of the latter in other fields, such as engineering and psychology, such methods were used for constructing linguistic models using corpora of natural language material. These models classified words not only based on meaning, but also on their co-occurrence with other words. They had been heavily influenced by Shannon's work on information theory (Shannon, 1948). These techniques were applied to MT systems with considerable early results, thus causing an overestimation of the techniques' promise for unconstrained NLP (Bar-Hillel, 1960; Church and Mercer, 1993).

During this phase, other NLP application areas began to emerge. An example is speech recognition, employing speaker-dependent, template-based architectures for digit recognition and limited phoneme classification (Fatehchand, 1960).

---

[7] Originally, Association for Machine Translation and Computational Linguistics (AMTCL).

Other events, which were unrelated to NLP in this phase, but which would considerably influence the field in subsequent phases included (a) the organization of the Dartmouth Conference in 1956, which introduced the term "Artificial Intelligence"– a field that would produce many NLP-shaping results, and (b) the publication of seminal works in connectionism, namely McCulloch and Pitts' paper on formal neurons (1943), Hebb's work on cell assemblies and synapses (1949), and Rosenblatt's paper on perceptrons (1958) (Cowan and Sharp, 1988).

Finally, Noam Chomsky's work on the structure of language set the stage for the next NLP phase through its criticism of stochastic techniques, such as n-grams in language modeling, and its introduction of a significant theoretical framework for the analysis and generation of language (Chomsky, 1956; Chomsky, 1957; Chomsky, 1959).

2.2.4 Theoretic Phase

The second phase in NLP spanned approximately from early 1960s until late 1980s. It is characterized by (a) a strong emphasis on theoretical topics including grammatical, logical, semantic, and pragmatic theories, (b) the construction of "toy" systems that demonstrated particular principles, and (c) the development of an NLP industry which commercialized many of this phase's theoretical results. In terms of application foci, this phase can possibly be subdivided into three eras characterized by the early question-answering type systems and database interfaces (1960s), the broadening of the application domain to include interfaces to other interactive systems (1970s), and the commercialization of NLP research (1980s) (Bates, 1994; Obermeier, 1988).

According to Grosz, *et al.* (1986), significant early work in this phase includes:

- a syntactic parser able to effectively handle multiple analyses of syntactically ambiguous sentences (Kuno and Oettinger, 1963), and
- the BASEBALL question answering system which incorporated coding of words by attribute-value pairs – a technique used throughout this phase, a separate syntactic analysis, and constrained linguistic domain coverage (Green, *et al.*, 1963).

Following these systems, researchers focused on the sentence and its meaning. This was done either in isolation, with respect to the human-computer dialog, or in the immediate (preceding) context. This work laid the foundation for domain-specific applications, and commercial systems which emerged towards the end of this phase. In the late 1970s, attention shifted to semantic issues, discourse phenomena, communicative goals and plans, and user models.

In 1975, in their ACM Turing Award lecture, Allen Newell and Herbert Simon formalized the *physical symbol hypothesis* which laid the foundation for symbolic NLP in specific, and symbolic AI in general (see Section 4.1) (Newell and Simon, 1976).

Many systems were developed to demonstrate the effectiveness of various theories in addressing various linguistic issues (see Section 5.4). Landmark systems include ELIZA (Weizenbaum, 1966), SHRDLU (Winorgad, 1972), REL (Thompson and Thompson, 1975), LUNAR (Woods, 1973), SOPHIE (Brown *et al.*, 1975), LIFER (Hendrix *et al.*, 1978), to name a few. Thousands of publications were produced to describe theories, incremental results, and resultant systems; Gazdar *et al.* (1987), provides a partial listing (1764) of these references appearing between 1980 and 1987.

During this phase, it became clear that isolated solutions to NLP problems did not scale up well, when attempts were made to widen their linguistic coverage, or apply them to a different domain (Grosz *et al.*, 1986; Jacobs, 1994). This realization motivated the re-examination of non-symbolic approaches which had been abandoned earlier by the NLP mainstream. In the stochastic arena, Baum and his colleagues developed the basic theory of Hidden Markov Models (HMMs), Viterbi

devised an algorithm which could be applied in estimating probabilities for phonetic, lexical and other linguistic classes, and the Brown and TIMIT corpora were constructed and made available to the research community (Allen, 1994a; Chruch and Mercer, 1993; Rabiner and Juang, 1993). In 1985, Church used a stochastic technique based on letter trigram modeling to identify the national origin of proper names for pronunciation purposes in the context of text-to-speech systems (Church, 1985). The success of this application reintroduced stochastic techniques to the mainstream of traditional NLP (Marcus, 1994).

Earlier, in the connectionist arena, a milestone publication by Minsky and Papert (1969) showed that perceptrons may be less powerful than a universal Turing machine. This result inhibited progress in training algorithms for neural networks, as a way to represent linguistic and other information, for many years. Eventually, significant contributions by several researchers, such as Grossberg, Kohonen, Hopfield, and Rumelhart revitalized the significance of connectionism and set the stage for its influence on NLP. An intriguing application of this period, which demonstrated the utility of neural networks in NLP, is NETtalk. This is a system that performed text-to-speech conversion in English and was capable of handling some coarticulation effects. It worked surprisingly well, but could not handle syntactic/semantic ambiguities effectively (Clark, 1993; Cowan and Sharp, 1988; Obermeier, 1988).

By the end of this phase, it had been shown that symbolic, stochastic, and connectionist approaches could address many significant problems in NLP. Moreover, stochastic and connectionist approaches had recovered from earlier criticism and were shown to be complementary in many respects to the mainstream symbolic approach. The results of this era coupled with concepts from HCI research set the stage for the final, user-centered phase.

2.2.5 User-Centered Phase

In the late 1980s, NLP entered the current empirical, evaluative, "user-centered" phase. Major advances and tangible results from the last fifty years of NLP research are being reinvestigated and applied to a wide spectrum of tasks where NLP can be applied. These tasks require "real-life" models of linguistic knowledge, as opposed to the models incorporated in earlier "toy" systems. Consequently, many successful NLP applications are emerging including spelling checkers, grammar checkers, and limited-domain, speaker-independent, continuous-speech recognizers for various computer and telephony applications.

During this phase, the field of human-computer interaction enters the mainstream of computer science. This is a result of the major advances in graphical user interfaces during the 1980s and early 1990s, the proliferation of computers, and the World-Wide-Web (Manaris and Slator, 1996). Accordingly, the evolution of NLP concerns and objectives is reflected in the continued growth of research and development efforts directed towards performance support, user-centered design, and usability testing. Emphasis is placed on (a) systems that integrate speech recognition and traditional natural language processing models, and (b) hybrid systems – systems that combine results from symbolic, stochastic, and connectionist NLP research (see Section 4.4). Developers are focusing on user interface development methodologies and user interface management systems (see Section 6.4).

Due to the "user-centeredness" of this phase, theories as well as applications are being judged by their ability to successfully compete in structured evaluations or in the marketplace (Chinchor and Sundheim, 1993; Hirschman and Cuomo, 1994; Moore, 1994b; Pallett *et al.*, 1994; Spark Jones, 1994). An emerging, promising methodology for NLP system development is the Star Model (see Section 6.4.3). This has been a popular development methodology in human-computer interaction, and it is now becoming relevant to NLP research, due to the emphasis on rapid-prototyping and

incremental development – necessitated by the current industry focus on speech understanding and other interactive NLP applications geared towards the everyday user.

3. Application Areas

Important NLP application areas include speech understanding and generation systems; natural language interfaces; discourse management, story understanding and text generation; interactive machine translation; and intelligent writing assistants (Bates, 1994; Church and Rau, 1995; Manaris and Slator, 1996; Obermeier, 1988). These areas are examined in the following sections.

3.1 Speech Understanding and Generation

The goal of *speech recognition systems* is to convert spoken words captured through a microphone to a written representation. *Speech understanding systems*, on the other hand, attempt to perform a more extensive (semantic, pragmatic) processing of the spoken utterance to "understand" what the user is saying, and act on what is being said – possibly by executing some command in an underlying system such as a database, or modifying their particular knowledge of the world. Major issues in this area include speaker independence vs. dependence, continuous vs. discrete speech, complexity of the linguistic model, and handling of environment noise (Markowitz, 1996).

Speech generation or *synthesis systems* deal with the opposite problem, namely to convert written representations of words to sounds. Compared to speech understanding, speech generation is considered by some to be a solved problem. This is because there exist several imperfect, yet effective speech synthesizers for many application domains and for a number of languages including English (American and British), Japanese, and Swedish (Kay *et al.*, 1994). Major techniques utilized for speech synthesis include

- *concatenation* of digital recordings, as in the output produced by US telephone directory assistance systems,
- *synthesis by rule*, where sounds are being generated electronically through the utilization of a grammar providing information on tone, intonation, and phonetic coarticulation effects, and
- *training of connectionist architectures*, as in the NETtalk system mentioned in Section 2.2.4 (Priece *et al.*, 1994).

An interesting example of a system which combines speech recognition with speech generation is Emily (Mostow *et al.*, 1994). Emily is an experimental speech understanding system that acts as a reading coach for children. It provides passages for reading, and listens making corrections whenever necessary – for instance it ignores minor mistakes such as false starts, or repeated words. Reddy (1996) estimates that this project could save U.S. taxpayers over $45 million if it could reduce illiteracy in the U.S. by as little as 20%.

Examples of speech processing (recognition, understanding, and generation) systems that have been marketed include Apple's Plain Talk, BBN's Hark, Decipher, DECtalk, DragonDictate, IBM VoiceType, Kurzweil Voice, Listen, Naturally Speaking, Phonetic Engine (Meisel, 1994; Obermeier, 1988; Rash, 1994; Scott, 1996).

3.2 Natural Language Interfaces

When examining the evolution of software systems, we observe a definite transition from the languages understood by the underlying hardware, i.e., languages based on binary alphabets, to the natural languages of humans (Feigenbaum, 1996; Firebaugh, 1988). In terms of programming

languages, this transition is manifested as a shift from machine languages to assembly languages, to high-level languages, up to non-procedural languages (also known as fourth generation languages). The goal of *natural language interfaces* is to bridge the gap between the linguistic performance of the user and the linguistic "competence" of the underlying computer system. These systems deal with typewritten as opposed to spoken language. They usually perform much deeper linguistic analysis than traditional speech recognizers (see Section 5.4).

Applications have been built for various domains including operating systems, databases, text editors, spreadsheets, and Internet navigation and resource location (Manaris, 1994; Manaris and Slator, 1996). Section 5.4 discusses several of these applications.

Examples of natural language interfaces that have been marketed include Battelle's Natural Language Query, BBN's Parlance, EasyTalk, English Query Language, INTELLECT, Intelligent Query, Language Craft, Natural Language, Symantec's Q+A Intelligent Assistant, and Texas Instrument's Natural Access (Church and Rau, 1995; Obermeier, 1988).

3.3 Discourse Management, Story Understanding, and Text Generation

The objective of discourse management and story understanding systems is to process natural language input to elicit significant facts or extract the essence of what is being said. These systems require access to linguistic knowledge ranging from lexical to, possibly, world knowledge relevant to the domain of discourse (see Section 5.3). Specific applications range from indexing (text segmentation and classification), to summarization ("gisting"), to retrieval (natural language search engines, data mining), to question and answer dialogs. In order to perform these tasks such systems may incorporate text generation components. These components utilize the collected linguistic knowledge to generate various forms of text, such as news summaries (skimming), and special documents. Given the latest advances in integration between speech and traditional natural language processing techniques, it should be straightforward to extend such systems to incorporate spoken input and output.

One example of a discourse management application that incorporates a text generation component is the patent authoring system designed by Sheremetyeva and Nirenburg (1996). This system is intended to interactively elicit technical knowledge from inventors, and then use it to automatically generate a patent claim that meets legal requirements. Another example is the system developed by Moore and Mittal (1996) which allows users to ask follow-up questions on system-generated texts. Specifically, users can highlight portions of the narrative available at the interface and, in response, the system will identify a set of follow-up questions that it is capable of handling. (Also, see the discussion on the DISCERN system in Section 4.4.)

Examples of other systems in this area (many of which have been marketed) include ATRANS, Battelle's READ, BORIS, Clarit, Conquest, Construe, Freestyle, FRUMP, GROK, J-Space, IPP, Oracle's ConText, Savvy/TRS, SCISOR, Target, Tome, and Westlaw's WIN (Church and Rau, 1995; Obermeier, 1988).

3.4 Interactive Machine Translation

This is the earliest area involving NLP in human-computer interaction. The goal of machine translation systems is to map from a source language representation to target language representation(s). Although no MT system can handle unconstrained natural language, there exist many success stories in well-defined sublanguages. In many cases, pre-editing of the input and post-editing of the output may be required (Hovy, 1993; Kay *et al.*, 1994). However, even in such cases MT systems can be very useful. This is illustrated by the fact that in 1975, and while all U.S.

government-funded MT projects had been canceled following the recommendations of the ALPAC report,

> [p]aradoxically, MT systems were still being used by various government agencies in the U.S. and abroad, because there was simply no alternative means of gathering information from foreign (Russian) sources so quickly. In addition, private companies were developing and selling (mostly outside the U.S.) MT systems based on the mid-1960s technology (Slocum 1986, p. 969).

As of 1994, the available languages for MT include Arabic, Danish, Dutch, English, Finnish, French, German, Greek, Italian, Japanese, Korean, Portuguese, Russian, Spanish, and Swedish (Kay *et al.*, 1994; Miller, 1993).

An example of a system that uses a well-defined sublanguage is TAUM-METEO, a fully-automatic machine translation system for translating weather reports from English to French. TAUM-METEO has been used extensively in Canada. It encapsulates a sublanguage model, which became stable in 1981. It requires almost no pre-editing of its input, or post-editing of its output. TAUM-METEO has been translating about 54,000 words per day for over a decade (Church and Rau, 1995; Obermeier, 1988). (Also, see the discussion of EUROTRA in Section 5.4)

Examples of systems that have been marketed include Fujitsu's ATLAS I and II, Globalink's GTS, Hitachi's HICATS, Intergraph's DP/Translator, Logos, Language Assistant Series, Siemens Nixdorf's METAL, Sanyo's SWP-7800 Translation Word Processor, Smart Translators, Socrata's XLT, Toltran's Professional Translation Series, Toshiba's AS-TRANSACT, and Tovna MTS (Church and Rau, 1995; Kay *et al.*, 1994; Miller, 1993; Obermeier, 1988).

In addition to translation of written texts, research has been directed on interactive translation of spoken utterances. One recent example of such a project is Janus-II, discussed in Section 5.4.4.2 (Waibel, 1996). Another example is Verbomil, a system designed as a portable simultaneous interpretation machine. Its goal is to mediate a dialog between two people interacting in real-time using different languages, possibly over the telephone (Alexandersson *et al.*, 1997; Kay *et al.*, 1994).

3.5 Intelligent Writing Assistants

Another area where NLP techniques have been successfully employed is in providing "intelligent" support for document preparation. Examples of applications range from spell checking agents, to hyphenation routines, to intelligent spacing/formatting/text-selecting agents, to grammar checkers, to style checkers providing readability statistics, to electronic thesauri, to automated document creation/maintenance environments, to translator-support environments. Some of these applications are so well understood, that many users do not consider them applications of NLP; examples include word processing, simple and approximate string matching, keyword search, and glossary look-up (Church and Rau, 1995; Hall and Dowling, 1980).[8] Other applications are still in an early marketability stage since their linguistic models and associated algorithms are still under development; examples include the grammar/style checkers available independently or bundled with word processing packages (Church and Rau, 1995; Obermeier, 1988).

An interesting example in this category is the Drafter system (Paris and Vander Linden, 1996). Drafter is an interactive document drafting tool. It assists technical writers with the task of managing draft, final, and updated versions of manuals in several languages. The approach is

[8] This might be viewed as another instance of the "AI losing its best children to Computer Science" phenomenon.

different from traditional MT, in that the system maintains a knowledge base of the concepts to be included in the manual in a language-independent form. Consequently, the translation is not dictated by the original text's language and style. Similarly to other intelligent writing assistants, such as spell checking agents, Drafter is not a fully-automatic tool, in that it relies on the user (technical writer) to provide information about the domain of discourse. This tool supports knowledge reuse, propagation of changes throughout documents, simultaneous production of drafts in several languages, accurate and consistently-used terminology, and production of stylistic variants of documents.

Given the success of this area of NLP, the market is flooded with a wide spectrum of independent and embedded systems. Examples of marketed products include popular word processors, such MS Word, WordPerfect, and FrameMaker, grammar checkers, such as Grammatik, and translator workbenches such as the Eurolang Optimizer, IBM's TranslatorManager/2, and Trados' Translation Workbench (Chucrch and Rau, 1995; Obermeier, 1988).

4. Linguistic Knowledge Models

Linguistic knowledge models may be classified along four basic categories, namely symbolic (knowledge-based), stochastic (probabilistic), connectionist, and hybrid approaches. Each approach has advantages and disadvantages which make it more/less suitable for addressing specific segments of the NLP problem space. In general, regardless of the approach, development of "complete," stable linguistic models is still a very difficult problem, especially for certain application areas such as speech understanding interfaces (Manaris and Harkreader, 1997). For this reason, investigation and adaptation of effective development methodologies from the area of human-computer interaction are becoming essential (see Sections 6.4. and 6.5).

The symbolic approach is based on explicit representation of facts about language through well-understood knowledge representation schemes and associated algorithms. The stochastic approach employs various probabilistic techniques to develop approximate generalized models of linguistic phenomena based on actual examples of these phenomena. The connectionist approach also uses examples of linguistic phenomena, but since connectionist architectures are less constrained than stochastic ones, linguistic models are harder to develop (Kay *et al.*, 1994). Finally, hybrid approaches explore different variations of compound architectures and linguistic models attempting to use the best approach (symbolic, stochastic, or connectionist) for a given modeling subproblem in an application. The following sections examine each of these approaches in terms of their foundations, major research results, and their respective strengths and weaknesses.

4.1 Symbolic Approach

The symbolic approach in NLP is best formulated by the *physical symbol system hypothesis* (Newel and Simon, 1976), although it originated in the late 1950s (see Section 2.2.3). This hypothesis states that intelligent behavior can be modeled using a physical symbol system consisting of physical patterns (symbols) which may be used to construct expressions (symbol structures); additionally, this system contains a set of processes that operate on expressions through creation, deletion, reproduction, and arbitrary transformation. This system exists in some larger encompassing environment and evolves through time by modifying its encapsulated symbolic expressions.

A great portion of the work in computational linguistic is based on this hypothesis. Specifically, numerous representation formalisms and associated analysis/generation techniques have been developed that rely on symbolic patterns to represent various notions (see Section 2.2.4). These notions include phonetic, morphological and lexical constituents, as well as syntactic, semantic,

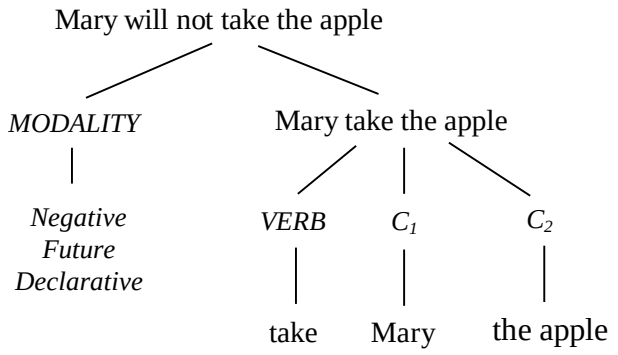

Figure 1. Example of Case Grammar Representation (adapted from Harris, 1985).

and discourse structure, relationships, and constraints. Examples of such formalisms, with approximate dates in parentheses, include Chomsky's theory of formal language grammars (regular, context-free context-sensitive, and unrestricted) (1957), Chomsky's transformational model of linguistic competence (1965), Filmore's case grammar (1967), Halliday's systemic grammar (1967), Wood's augmented transition network (1970), Schank's conceptual dependency theory (1975), Wilks' preference semantics (1975), Burton's semantic grammar (1976), Colmerauer's definite clause grammar (1978), Gazdar's phrase structure grammar (1979), Kay's functional grammar (1979), and Bresnan and Kaplan's lexical-functional grammar (1982).[9]

Figure 1 shows a sample of a case grammar representation of the sentence "Mary will not take the apple." Harris (1985) and Winograd (1983) provide additional details on these formalisms. Such formalisms spawned a wide variety of algorithms and systems; for examples, see Cercone and McCalla, (1986), Grosz *et al.* (1986), Obermeier (1988), and Pereira and Grosz (1994).

Some of the apparent strengths of symbolic formalisms for NLP are as follows (Church and Mercer, 1993; Winograd, 1983):

- They are well understood in terms of their formal descriptive/generative power and practical applications.
- They currently provide the most effective approach for modeling long-distance dependencies, such as subject-verb agreement, and wh-movement.[10]
- They are usually "perspicious," in that the linguistic facts being expressed are directly visible in the structure and constituents of the model.
- They are inherently non-directional, in that the same linguistic model may be used for both analysis and generation.

---

[9] This list is by no means exhaustive or representative of the various schools of thought within the symbolic approach to NLP; moreover, it excludes significant implementations of symbolic formalisms, such as SNePS (Shapiro, 1979).

[10] Winston (1993) provides an introductory presentation on expressing language constraints, such as long-distance dependencies, using a symbolic approach (pp. 575-598).

- They can be used in multiple dimensions of patterning, that is they can be used for modeling phenomena at various linguistic knowledge levels (see Section 5.3).
- They allow for computationally efficient analysis and generation algorithms, as in (Earley, 1970; Marcus, 1978).

Some of the apparent weaknesses of the symbolic approach to NLP are (Church and Mercer, 1993; Kay *et al.*, 1994):

- Symbolic linguistic models tend to be fragile, in that they cannot easily handle minor, yet non-essential deviations of the input from the modeled linguistic knowledge – nevertheless, various flexible (robust) parsing techniques have been devised to address this weakness. Such techniques look for "islands" of structural or semantic coherence and, through heuristics or user intervention, may recover from parsing failures.
- Development of symbolic models requires the use of experts such as linguists, phonologists, and domain experts, since such models cannot be instructed to generalize (learn from example).
- Symbolic models usually do not scale up well. For instance, Slocum (1981) discusses how after two years of intensive development, LIFER's knowledge base grew so large and complex that even its original designers found it hard and impractical to perform the slightest modification. Specifically, the mere act of performing a minor modification would cause "ripple effects" throughout the knowledge base. These side-effects eventually became almost impossible to isolate and eliminate.
- In many practical cases, symbolic-approach techniques perform worse than stochastic and connectionist pattern recognition systems tuned by real-life training data. They cannot model certain local constraints, such as word preferences that can be very useful for effective part-of-speech tagging and other applications.

In summary, symbolic formalisms are the most well-studied techniques for NLP system development. Although they exhibit several weaknesses when compared against stochastic or connectionist approaches, they are still valuable and powerful mechanisms for NLP. They are especially useful, at least, in cases where the linguistic domain is small or well-defined, and where modeling of long-distance dependencies is essential.

4.2 Stochastic Approach

The driving force behind stochastic (statistical) models is their ability to perform well even in the presence of incomplete linguistic knowledge about an application domain. Such models thrive on the inherent inflexibility and fragility of symbolic models, which stem from our lack of complete understanding of many linguistic phenomena – an essential prerequisite to developing successful symbolic models. Stochastic models include a number of parameters that can be adjusted to enhance their performance (Allen, 1994a; Charniak, 1993; Church and Mercer, 1993; Kay *et al.*, 1994; Knill and Young, 1997; Marcus, 1994).

A popular stochastic model is the Hidden Markov Model (HMM). Similarly to a finite-state machine, an HMM consists of a set of states (one of which is the initial state), a set of output symbols which are emitted as the system changes states, and a set of acceptable transitions among states. Additionally, each state in an HMM (as opposed to a finite-state machine) has two sets of probabilities associated with it; one determines which symbol to emit from this state (emission probabilities); the second set determines which state to visit next (transition probabilities). Once the topology of a given HMM has been decided, Baum's (1972) *backward-forward* algorithm can be used to effectively derive these two sets of probabilities using a set of training data; by adjusting

S	→	NP VP	0.85	P	→	like	1.00
S	→	VP	0.15	V	→	like	0.40
NP	→	N	0.40	V	→	flies	0.40
NP	→	N PP	0.35	V	→	jumps	0.20
NP	→	N NP	0.25	N	→	flies	0.45
VP	→	V	0.32	N	→	jumps	0.05
VP	→	V NP	0.31	N	→	banana	0.30
VP	→	V PP	0.19	N	→	time	0.20
VP	→	V NP PP	0.18				
PP	→	P NP	1.00				

Figure 2. Probabilistic Context-Free Grammar
(adapted from Charniak, 1993).

appropriately the model's parameters, this algorithm is guaranteed to either improve or, at least, not worsen the performance of the model. Once the HMM's parameters have been adjusted, the Viterbi (1967) algorithm may be used to recognize specific input against the trained HMM.

Another stochastic model is the probabilistic context-free grammar (PCFG). Similarly to HMMs, which extend finite-state machines with probabilities, PCFGs extend context-free grammars with probabilities. One approach is to assign probabilities based on rule use; that is, using a set of training data, the probability of each rule's "significance" can be determined based on the frequency of this rule's contribution to successful parses of training sentences (see Figure 2). If one is willing to compromise potential accuracy over efficiency, given a trained PCFG, there exist several effective algorithms that can be employed. For example, one could employ an *N-first* parsing algorithm. This algorithm will not explore every possible alternative, but only a preset number of most promising alternatives, e.g., three or four. This can be accomplished by using a probability cutoff value to prune less promising parse subtrees. Of course, such techniques are not admissible,[11] in that it is possible that they may prune a subtree which, although initially not very promising, would be a much better alternative at the end of a complete search through the parse space. However, assuming that the system explored constituents in the native language order (e.g., left-to-right for English), then the types of sentences that it would probably get confused on would be the same type of sentences that would confuse a native speaker, i.e., *garden-path sentences*, such as "we gave the girl the cake was baked by grandma's recipe."[12]

Some of the strengths of the stochastic approach are (Church and Mercer, 1993; Kay *et al.*, 1994; Knill and Young, 1997; Rabiner and Juang, 1993; Schmucker, 1984):

- Stochastic systems are effective in modeling language performance through training based on most frequent language use. They are useful in modeling linguistic phenomena that are not well-understood from a competence perspective, e.g., speech.
- The effectiveness of a stochastic system is highly dependent on the volume of training data available; generally more training data results to better performance.

[11] A search algorithm is admissible if it is guaranteed to find the optimum path to a solution, if such a path exists (Luger and Stubblefield, 1993).

[12] The embedded clause "the cake was baked by" modifies "girl". For additional discussion on garden-path sentences and examples, see Section 5.4.2.3 and (Akmajian, 1990; Winograd, 1983).

- Stochastic approaches may be easily combined with symbolic models of linguistic constraints, such as dialog structure, to enhance the effectiveness and efficiency of an application (hybrid systems are discussed in Section 4.4).
- Stochastic models can be used to model nuances and imprecise concepts such as *few, several*, and *many*, that have traditionally been addressed by fuzzy logic.

Some of the weaknesses of the stochastic approach are (Church and Mercer, 1993; Kay *et al.*, 1994; Rabiner and Juang, 1993):

- Run-time performance of stochastic systems is generally linearly proportional to the number of distinct classes (symbols) modeled, and thus can degrade considerably as classes increase; this holds for both training and pattern classification.
- In general, given the state-of-the-art in corpus development, producing training data for a specific application domain can be a time-consuming and error-prone process. Thus, since the effectiveness of stochastic systems is tightly bound to extensive, representative, error-free corpora, the difficulty of developing such systems might be, in the general case, similar to that of other approaches.

Stochastic approaches are very effective in addressing modeling problems in application domains where traditional symbolic processes have failed. Although their potential is still being explored, and thus new significant results may be discovered, they have already proven to be a valuable approach to NLP for human-computer interaction.

4.3 Connectionist Approach

Similarly to the stochastic approach, the connectionist approach is also based on employing training data to improve the performance of linguistic models. The difference between the two approaches is in the complexity of the system architecture. Specifically, connectionist models consist of massive interconnected sets of simple, non-linear components. These components operate in parallel, as opposed to the non-parallel systems found in other approaches, such as finite-state machines and context-free frameworks.[13] Acquired knowledge is stored in the pattern of interconnection weights among components. There exist various characteristics that affect the performance and utility of connectionist systems, such as number and type of inputs, connectivity, choice of activation threshold/function, and choice of update function (Caudill and Butler, 1990; 1992; Cowan and Sharp, 1988; Firebaugh, 1988; Kay *et al.*, 1994; Markowitz, 1996; Obermeier, 1988; Rabiner and Juang, 1993).

Although it has been argued that, due to their lack of internal structures, connectionist architectures are not competent in handling natural language (Fodor and Pylyshyn, 1988), such architectures have been used to model various linguistic phenomena, especially in phonology, morphology, word recognition (spoken and written), noun-phrase understanding, prepositional-phrase attachment, script-based narratives, and speech production (Elman, 1991; Miikkulainen, 1993; Wermter, 1996).

---

[13] Nevertheless, from a theoretical perspective, it can be argued that symbolic, statistical, and connectionist approaches are not necessarily distinct; for instance, connectionist and statistical architectures are usually implemented on top of non-parallel computational architectures which conform to the physical symbol system hypothesis. From a practical perspective, however, this distinction is meaningful, since different approaches appear to be best suited to different regions of the NLP problem space – similarly to how different high-level programming languages are best suited to different regions of the problem space solvable by universal Turing machines.

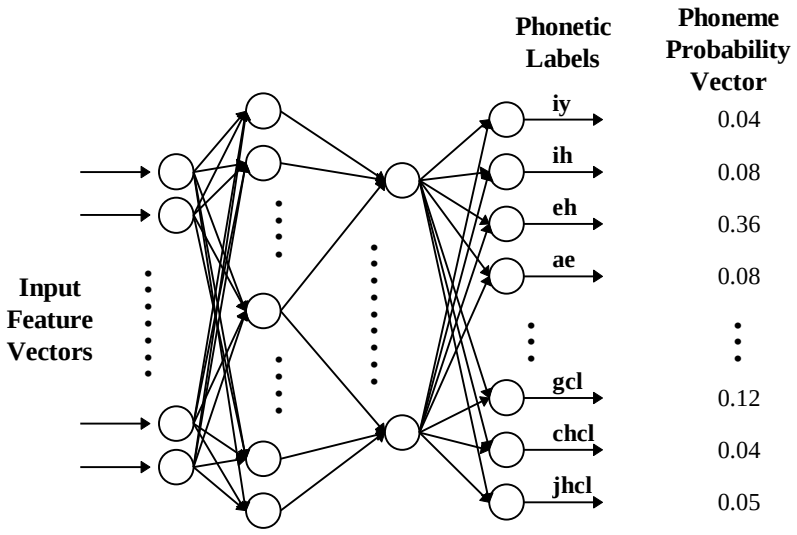

Figure 3. Phoneme Probability Estimator
(adapted from Schalkwyk and Fanty, 1996)

For example, Figure 3 shows a connectionist approach to phoneme classification. The input consists of a collection of feature vectors derived through a temporal context window centered on a target vector. Each of these vectors has been generated from speech input analyzed with a spectral analysis method such as Linear Predictive Coding. The output is a phoneme probability vector in which each phoneme is assigned a probability indicating the likelihood that a given input frame belongs to that phoneme.

Some of the strengths of the connectionist approach are (Caudill and Butler, 1990; Fodor and Pylyshyn, 1988; Rabiner and Juang, 1993):

- Connectionist architectures are self-organizing, in that they can be made to generalize from training data even though they have not been explicitly "instructed" on what to learn. This can be very useful when dealing with linguistic phenomena which are not well-understood – when it is not clear what needs to be learned by a system in order for it to effectively handle such a phenomenon.
- Connectionist architectures are fault tolerant, due to the distributed nature of knowledge storage. Specifically, as increasing numbers of their components become inoperable, their performance degrades gracefully (gradually).
- The weights of a connectionist architecture can be adapted in real-time to improve performance.
- Due to the non-linearity within each computational element, connectionist architectures are effective in modeling non-linear transformations between inputs and outputs.

Some of the weaknesses of the connectionist approach are:

- Once a connectionist system has been trained to handle some linguistic (or other) phenomenon, it is difficult to examine and explain the structure or nature of the acquired knowledge. In many cases, this is not detrimental; however, it is possible for a connectionist system to learn

the wrong type of knowledge, especially if the training set is not well-developed or well-understood.[14]

- It is possible for a system to be over-trained and thus diminish its capability to generalize – only the training data can be recognized.
- Due to their massive parallelism, and their usual implementation on non-parallel architectures, connectionist systems may be ineffective from a runtime complexity perspective for many real-time tasks in human-computer interaction.

Similarly to stochastic approaches, connectionist models are very effective in addressing NLP problems in which traditional symbolic models are ineffective. Although their potential is still being explored, and thus new techniques and applications are being developed, they have already proven to be a valuable tool for NLP in human-computer interaction.

4.4 Hybrid Approach

As seen in the previous sections, each of the above approaches has advantages and disadvantages; accordingly, each approach has advocates and critics, especially on theoretical and philosophical grounds. However, when dealing with natural language systems for human-computer interaction, theoretical debates are not of much relevance unless they contribute to one's understanding of the applicability, or utility of a particular approach with respect to developing an effective human-computer interface. Consequently, researchers have begun developing hybrid architectures which take advantage of the relative strengths of each approach, in an attempt to use "the best tool for the job."

One example of the hybrid approach is the DISCERN system which combines symbolic and subsymbolic (connectionist) techniques to process script-based narratives. Specifically, it reads short stories, stores them in episodic memory, generates paraphrases of the stories, and answers questions related to these stories. DISCERN employs subsymbolic modules to perform each of these subtasks. At a low level, the system is connectionist in nature; however, at a high level, it is symbolic, in that its modules are connected using symbolic information structures, such as scripts, lexicon, and episodic memory (Miikkulainen, 1993; Miikkulainen, 1994).

Another example is the SCREEN speech understanding system. It combines connectionist and symbolic techniques to perform robust analysis of real-world spoken language (Wermter, 1996). Similarly, SpeechActs combines off-the-shelf continuous speech recognizers (employing stochastic or connectionist models) with symbolic modules performing syntactic, semantic, and dialog analysis (Martin *et al.* 1996).

Finally, the SUITE architecture integrates speech recognition and natural language processing components (Manaris and Harkreader, 1997). This is a generic architecture for speech understanding interfaces to interactive computer applications. It is based on the CSLU-C architecture for speech recognition and the NALIGE natural language interface architecture (Manaris and Dominick, 1993; Schalkwyk and Fanty, 1996). It uses connectionist techniques for

[14] One such example is a neural network developed at the Stanford Research Institute which was trained to detect the presence of tanks in photographs. Although the system was successful when presented with testing data derived from the same batch of photographs as the training set, it performed badly otherwise. Eventually, it was discovered that the system had learned to recognize other characteristics of the data set, such as the differences in light intensity and density; it turned out that all photographs in the training set containing a tank had been taken in the morning, whereas the non-tank ones had been taken in the afternoon (Clark, 1993; p. 41).

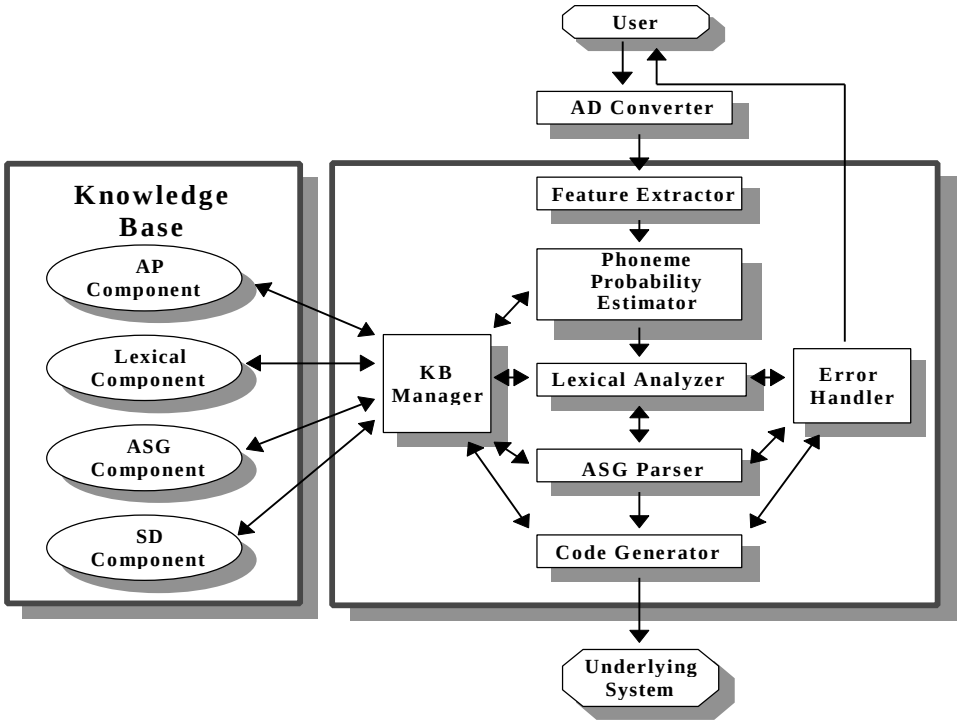

Figure 4. SUITE Speech Understanding Interface Architecture.

phoneme identification, stochastic techniques for creating an N-best sentence hypothesis, and symbolic techniques for additional linguistic analysis of the input. Specifically, It consists of modules which perform acoustic, phonetic, lexical, syntactic, semantic, and pragmatic processing as follows (see Figure 4):

- a feature extractor converts the speech signal to a set of feature vectors;
- a phoneme probability estimator uses a connectionist model to produce a phoneme probability matrix, which approximates the probability of a feature vector being part of a given phoneme;
- a lexical analyzer employs a Viterbi search to determine the N-best paths through the phoneme probability matrix; this search is structurally-driven to focus only on "valid" phonemic transitions, thus enhancing accuracy and efficiency;
- an augmented semantic grammar (ASG) parser identifies the sequence of words that could possibly occur at a given point in a "valid" input, enforce pragmatic constraints, and generate semantic interpretations;
- a code generator converts semantic interpretations to commands to be passed to the underlying system.

Additionally, the architecture includes an error handler, a knowledge-base manager and a system driver.

In summary, hybrid techniques utilize the strengths of symbolic, stochastic, and connectionist approaches in an attempt to (a) minimize the human effort required for linguistic model construction, and (b) maximize the flexibility, effectiveness, and robustness of NLP applications for human-computer interaction.

5. Knowledge and Processing Requirements

According to Feigenbaum (1996), NLP programs exist in one extreme of a continuum which he calls, the "What-to-How" software spectrum.[15] On one extreme of this continuum – the "How" or procedural side – we find general-purpose computing devices. At this level, knowledge is described in terms of binary code (data and instructions) to be executed by the underlying hardware whose processing capabilities are theoretically equivalent to universal Turing machines. On the other extreme of the continuum – the "What" or declarative side – we have the user who wishes to express his/her goals and needs through natural communicative modalities such as gesture, speech, and body language. The history of software evolution is marked by specific milestones in the attempt to bridge the gap between the "How" and "What" ends of this continuum. Such milestones include assembly languages, high-level languages, software development environments, specification languages, intelligent agents, and domain-specific expert systems. The next milestone in this continuum is probably intelligent user interfaces encapsulating knowledge about the domain as well as the user communicative capabilities/preferences – thus bringing software closer to the "What" side of the continuum.

5.1 Computational Issues

It is important to remember that any model of natural language phenomena will eventually have to be communicated to and executed by a computing device. Therefore, assuming that Church's Thesis holds,[16] any theory or representation formalism that is not formally equivalent to a universal Turing machine, will fall short of exploiting all the power of a computing device in its attempt to perform NLP; this might have considerable implications with respect to the utility of any NLP theory.

On the other hand, Wegner (1997) discusses a thought-provoking alternative model of computation based on interaction, namely *interaction machines*, that is more powerful than Turing machines. Specifically, he argues that any system that allows for interaction is capable of exhibiting richer behavior than a Turing machine. That is the "assertion that algorithms capture the intuitive notion of what computers compute is invalid" (p. 83). This supports claims of certain researchers that natural language could be effectively (if not completely) modeled by context-free, or even regular language frameworks (Blank, 1989; Marcus, 1980; Reich, 1969) – especially if such models can be trained through interaction.[17] Actually, such results have contributed to empirical NLP applications in the late 1980s and 1990s based on text or speech corpora, finite-state-machine modeling frameworks, such as HMMs, and neural networks (see Sections 4.2 and 4.3).

5.2 Understanding Natural Language

Let us for a moment consider a hypothetical dialogue taking place between a human and a computer system. This dialog is in the flavor of Apple's "Knowledge Navigator" vision of the future (Lee, 1993):

<Human>: I am getting ready to quit for today.

---

[15] Actually, Feigenbaum focuses on general AI programs. However, assuming that NLP is an AI-complete problem, his ideas hold in the NLP realm.

[16] Church's Thesis proposes that the intuitive notion of what is computable corresponds to the formal notion of computability as expressed by Turing machines (Lewis and Papadimitriou, 1981; Wegner, 1997).

[17] This has a strong intuitive appeal as it is through interaction that humans acquire natural language.

<Computer>: I understand.

<Human>: Please do not delete any of the temporary files I have created, as they contain information that may be useful later for the Advances in Computers article that you and I are working on.

<Computer>: I understand.

Actually, assuming that the underlying computer system did not, by default, delete any temporary files, this "understanding" system could be effectively implemented by the following LISP code (Firebaugh, 1988):

```
(while (not (null? (read)))
       (display "I understand")
       (newline))
```

Obviously, no understanding is taking place here. Nevertheless, this system contains knowledge about natural language that has been implicitly programmed into it by its developer. For example:

- In a dialog there exists two participants.
- An information exchange is taking place between these participants. Actually, this rule does not specifically appear in the above program; it is nevertheless one of the rules known to the designer of the system and thus could be claimed that this knowledge is encapsulated in the design choices made during the system's development.
- Once the first participant has completed a statement, the other may claim that (s)he understands what is being said, even if that is not the case.

One of the philosophical questions which naturally arise in the context of discussing knowledge and processing requirements for NLP systems is "what does it mean to understand natural language?" (Winograd, 1980). From a philosophical perspective, it can be claimed that this system is not much different from state-of-the-art NLP systems, since such systems are also not capable of doing anything more than they have been programmed to do. Hofstadter (1979) devotes much discussion to this and related issues. For instance, given the input:

> "Margie was holding tightly to the string of her beautiful new balloon. Suddenly, a gust of wind caught it. The wind carried it into a tree. The balloon hit a branch and burst. Margie cried and cried" (Rumelhart, 1975, p. 211)

Hofstadter points out that an NLP system could never truly understand what is being said "until it, too, has cried and cried" (p. 675).

Nevertheless, considering systems like ELIZA (see Section 5.4.1.1), which under certain circumstances could pass the Turing Test of intelligence,[18] and the simplistic keyword matching strategies they employ (without any formal representation of syntax, semantics, or pragmatics), how could we possibly expect to distinguish between a system that understands natural language (if such as a system may ever exist) from one that does not?[19] From a human-computer-interaction

---

[18] The Turing Test, proposed by Alan Turing (1950), requires a human interrogator to communicate with a computer via a teletype. The interrogator is not told whether (s)he is communicating with a computer or another human; if the interrogator cannot tell the difference, then the computer has passed the test (Russell and Norvig, 1995; p. 5). The interested reader may also look into the annual Loebner Prize Competition, a formal effort to locate a machine that can pass the Turing Test (Epstein, 1992).

[19] McCorduck (1979) describes how an ELIZA-like program "participated" in a lengthy, intimate conversation with an internationally respected computer scientist, as cited in Firebaugh (1988), p. 223.

perspective, which is the one adopted in this article, such questions are of importance to artificial intelligence and cognitive science researchers; what really matters is the end-result, namely the effectiveness, learnability, user-friendliness, and functionality of the user interface which employs natural language models.

In general, human-defined models are at best approximations of the natural phenomena they represent – given the limitations imposed on our modeling capabilities by our intellectual capacity and our senses (or lack thereof). Since NLP systems incorporate human-defined models of natural language – a natural phenomenon – we should expect to find at best an approximation to understanding. From a computer science perspective, one possible definition to what constitutes "understanding" of natural language is the following: *A system "understands" natural language, if, in response to some input, it creates a conceptual structure corresponding to that input, updates an existing conceptual structure, or makes an appropriate modification to a knowledge base*. Obviously, this definition excludes minimalistic systems such as the one presented above. Additionally, it makes only a cursory reference to "correctness" which is an essential quality of algorithms in computer science.[20] Finally, it introduces a new question, namely: "what type of knowledge would we expect to find in an NLP system's conceptual structures, or its knowledge base?". This question is addressed in the next section.

5.3 Natural Language Knowledge Levels

The success of any NLP system is highly dependent on its knowledge of the domain of discourse. Given the current state-of-the-art in NLP models, this knowledge may be subdivided into several levels. There exist different schools of thought, but, in general, researchers agree that linguistic knowledge can be subdivided into at least lexical, syntactic, semantic, and pragmatic levels. Each level conveys information in a different way. For example, the lexical level might deal with actual words (i.e., lexemes), their constituents (i.e., morphemes), and their inflected forms. The syntactic level might deal with the way words can be combined to form sentences in a given language. One way of expressing such rules is to assign words into different syntactic categories, such as noun, verb, and adjective, and specify legal combinations of these categories using a grammar. The semantic level might deal with the assignment of meaning to individual words and sentences. Finally, the pragmatic level might deal with monitoring of context/focus shifts within a dialog and with actual sentence interpretation in the given context.

Table 1 shows one commonly used classification which attempts to be as thorough as possible (given our current understanding of the language phenomenon) by accounting for acoustic, as well as general world knowledge (Akmajian *et al.*, 1990; Allen, 1994b; Manaris and Slator, 1996; Sowa, 1984). In this classification, each level is defined in terms of the declarative and procedural characteristics of knowledge that it encompasses.

5.4 Classification of NLP Systems

As seen in the previous section, natural language can be viewed at different levels of abstraction. Based on the application domain, NLP systems may require only subsets of the above knowledge levels to meet their application requirements. For example, a machine translation system, such as the Eurotra prototype which focuses on documents dealing with telecommunications, and covers nine languages of the European Community, namely Danish, German, Greek, English, Spanish, French, Italian, Dutch, and Portuguese (Arnold, 1986; Maegaard and Perschke, 1991), may require only knowledge levels 3 to 7 (or possibly 8). Similarly, a speech recognition system, such

[20] A quality that is not always provable, and, in many cases, not even attainable.

1. *Acoustic/prosodic knowledge*: What are rhythm and intonation of language; how to form phonemes.
2. *Phonologic knowledge*: What are spoken sounds; how to form morphemes.
3. *Morphologic knowledge*: What are sub-word units; how to form words.
4. *Lexical knowledge*: What are words; how to derive units of meaning.
5. *Syntactic knowledge*: What are structural roles of words (or collection of words); how to form sentences.
6. *Semantic knowledge*: What is context-independent meaning; how to derive sentence meanings.
7. *Discourse knowledge*: What are structural roles of sentences (or collections of sentences); how to form dialogs.
8. *Pragmatic knowledge*: What is context-dependent meaning; how to derive sentence meanings relative to surrounding discourse.
9. *World knowledge:* What is generally known by the language user and the environment, such as user beliefs and goals; how to derive belief and goal structures. Currently, this is a catch-all category for linguistic processes and phenomena that are not well understood yet. Based on past evolutionary trends, this knowledge level may be further subdivided in the future to account for new linguistic/cognitive theories and models.

Table 1. Knowledge levels in NLP models.

as Dragon Systems' Naturally Speaking and Kurzweil Voice, may require only knowledge levels 1 to 5, although it would benefit from having access to knowledge in levels 5 to 7 (Manaris and Slator, 1996). Finally, a speech understanding interface to UNIX may use knowledge levels 1 to 6 to produce a semantic representation of a given input, e.g., `delete-file("x")`, and then knowledge level 8 to convert that semantic interpretation to the corresponding realization in the underlying command language, e.g., `"rm -i x"`. Such an interface could benefit from having access to specific knowledge about dialog in the context of communicating with an operating system (see Figure 5). An example of such an interface is UNIX Consultant (see Section 5.4.4.1).

In practice, it may be hard to classify NLP systems based on the types and levels of linguistic knowledge they encapsulate. For example, even for primitive NLP systems, such as the one seen in Section 5.2, it might be argued that they contain implicit knowledge from various knowledge levels. Nevertheless, it may be beneficial to examine NLP systems based on the depth of explicit linguistic analysis they perform, as this may provide clues on their strengths and weaknesses. In the remainder of this section, we will attempt to classify a few representative symbolic modeling methods according to this linguistic analysis classification scheme and provide examples of relevant NLP systems.

5.4.1 Lexical Analysis Systems

Lexical analysis systems, also known as keyword matching systems, employ a pattern matching mechanism designed to recognize or extract certain predefined keywords from the user input. Compared to the minimalistic system of Section 5.2, which implicitly encapsulates a limited amount of natural language knowledge, it can be claimed that lexical analysis systems directly

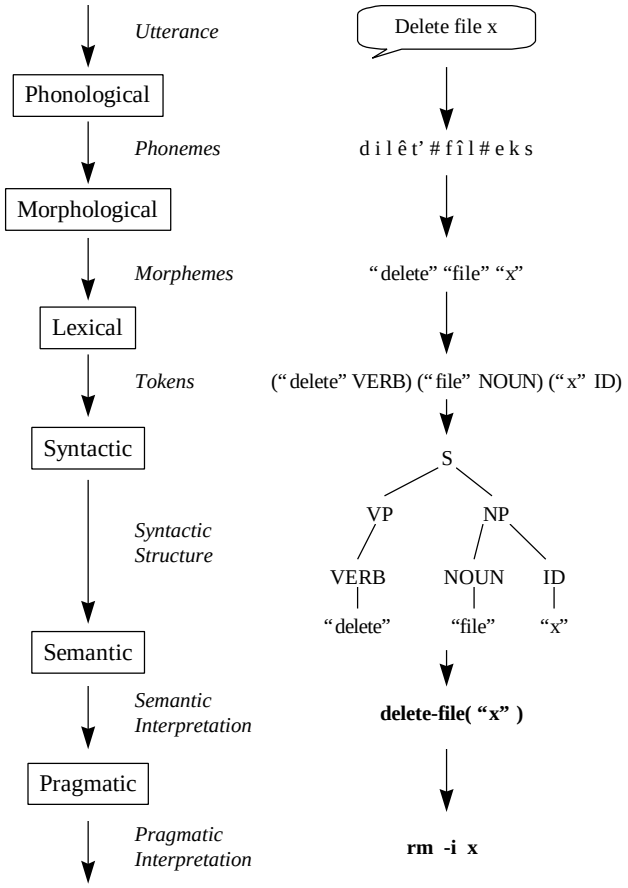

Figure 5. Knowledge Levels in NLP Systems

understand a small subset of natural language. These systems can be compared to a human visiting a foreign country who can recognize only a few words and/or phrases from the native language and can only respond using a small number of previously memorized sentences; additionally, once a certain keyword has been recognized, (s)he may perform some complex action which is appropriate in the given context, such as following the directions in the sentence: “To find the *bank* go *left*, *right*, and then *left*.” In this example, the highlighted words are the actual keywords which convey the most important information.

5.4.1.1 ELIZA

A program using a lexical approach to natural language understanding is ELIZA. This program was written by Weizenbaum at MIT in an attempt to study issues relevant to natural language communication between humans and computers (Weizenbaum, 1966; 1967). ELIZA assumes the role of a Rogerian psychotherapist and, under many circumstances, manages to misguide the user into believing that it actually understands all that is being said. Its knowledge base consists of a collection of predefined patterns against which the user’s input is compared. Each of these patterns has a generic template associated with it, which is used in constructing ELIZA’s response. However, after some interaction with this system the user starts realizing the limits of this program’s intelligence. Incidentally, Hofstadter (1979, p. 621) claims that humans get bored

interacting with such an "intelligent" system not when they have exhausted its repertoire of behavior, but when they have intuited the limits of the space containing this behavior.

5.4.1.2 NLDOS

One NLP application that performs its function relying on strictly lexical analysis techniques is NLDOS (Lane, 1987). NLDOS is a natural language interface to operating systems. The main motivation for building this interface was the syntactic differences between the VAX/VMS and the MS-DOS command languages, which the designer was using interchangeably. He states (p. 261):

> After changing default directories several dozen times on a DEC VAX with the `set def` command, I invariably type the same command to change directories on my IBM PC. Then, I throw a mental "Read my mind!" at the machine and edit the command to `cd` (short for `chdir`).

An additional consideration was the syntactic inflexibility of the MS-DOS `backup` command, especially if the user wants to include subdirectories in the actual request.

NLDOS was implemented in PROLOG using simple lexical rules to identify tokens in the natural language input, such as disk drive, filename, and date specifications. This system's task coverage is very limited and its linguistic knowledge is organized in an *ad hoc* fashion. This is because, the interface development was mainly motivated by the designer's personal difficulties and specifically tailored with respect to his needs. For instance, Lane decided that, for implementation simplicity, certain punctuation marks within the natural language input, such as colons, periods, and hyphens, may only appear inside disk drive, filename, and date specifications, respectively.

NLDOS allows for flexible matching to the point where careless or obtuse wording may "confuse" the system into executing a command different from the one intended. For instance, assuming that the system was trained to recognize any pattern of the type `(* delete * <FILESPEC>)` as a request to delete file <FILESPEC>, where * matches zero or more words and <FILESPEC> stands for a filename token, an input such as "Do not delete file REPORT.DOC" would have the opposite effect. Actually, during the testing phase, Lane reports that the system inadvertently erased the complete contents of one of his computer's disk drives. This is a general problem with systems that perform flexible (or robust) parsing.

5.4.2 Syntactic Analysis Systems

Syntactic analysis systems, in addition to recognizing certain input keywords, attempt to derive a unique structure, namely a parse tree, which directly corresponds to the syntactic information encapsulated in the input sentence.

5.4.2.1 The Chomsky Hierarchy

One of the early influential figures in the field of linguistic analysis is MIT's Noam Chomsky (1957; 1965). His theories depended on a rigorous approach to studying language developed by earlier researchers, but went further by deriving a collection of grammars which describe the structural relations which are acceptable within language (Harris, 1985). Chomsky explains that *a generative grammar is a system of rules that in some explicit and well-defined way assigns structural descriptions to sentences* (Chomsky, 1965, p. 8). This implies that generative grammars offer a formal framework to be used in implementing computer systems which perform syntactical analysis on statements of a specific language.

Additionally, Chomsky classified languages into four categories according to the restrictions imposed on the form of the actual grammar describing them, namely *recursively-enumerable*,

context-sensitive, *context-free*, and *regular* languages. These restrictions reflect the descriptive power of the corresponding grammars. Moreover, these restrictions reflect the computational power needed by a computer system using such a grammar to perform syntactic analysis on a specific language. Analytically, the most powerful grammars, i.e., grammars describing recursively enumerable languages, require the power of a universal Turing machine to be interpreted. Grammars describing context-sensitive languages require the computational power of a linear-bounded automaton (a variation of a Turing machine constrained by the fact that the amount of storage which is available for its processing needs is finite).[21] Grammars corresponding to context-free languages need the processing power of a pushdown automaton. Finally, regular grammars can be handled by a finite-state machine.

Chomsky claimed that due to the recursive-embedding nature of clauses, natural language cannot be modeled by finite-state machines. This claim has been subsequently challenged by other researchers such as Blank (1989), Marcus (1980), and Reich (1969) (see Section 5.4.2). Incidentally, Chomsky developed a formalism equivalent to a universal Turing machine, namely *transformational grammars*, which can be used to model natural language. However, although this formalism is appropriate for generating subsets of natural language, it is extremely inefficient for practical natural language analysis (Woods, 1970).

5.4.2.2 Augmented Transition Networks

In "Transition Network Grammars for Natural Language Analysis", Woods (1970) presents a formalism called augmented transition networks (ATNs). This formalism is actually a computational mechanism equivalent in power to a universal Turing machine. An ATN is not by definition a natural language understanding mechanism, but it may be used to define a process which can recognize subsets of natural language.[22] An ATN is similar to a finite-state machine, in the sense that it consists of a set of nodes, corresponding to the states of the computational process, and a set of named directed arcs connecting these nodes, corresponding to the input symbols which may cause specific transitions between computational states. Additionally, ATNs have the following features:

- Arcs may be named with state names, thus allowing for recursive invocation of complete ATNs, including the caller ATN.
- A set of registers each of which may be assigned to constituents of the parse tree being built.
- Arbitrary tests may be associated with any given arc; these tests are built in terms of registers and/or input symbols.
- A set of actions may be associated with any given arc; these actions provide the mechanism to incrementally construct the appropriate parse tree.

The main advantage of ATNs resides in the immense generative power they offer to the NLP application designer. Consequently, several NLP systems have been implemented using this formalism (either exclusively, or as an underlying programming mechanism), such as LUNAR (Woods, 1970), SOPHIE (Brown and Burton, 1975), GUS (Bobrow *et al.*, 1977), and LIFER (Hendrix *et al.*, 1978). Some of these systems will be examined in Section 5.4.3, since, in addition

---

[21] Actually, the amount of storage available to a linear bounded automaton is linearly proportional to the storage occupied by its original input, as opposed to the infinite storage capacity characterizing universal Turing machines (Moll *et al.*, 1988).

[22] The ATN formalism can be thought of as a special-purpose high-level language.

to performing strictly syntactic analysis, they attempt to construct a semantic interpretation of their natural language input.

Due to their non-deterministic nature, ATNs tend to be very expensive from a computational point of view. This problem is intensified when dealing with large grammars, since, in such a case, input sentences tend to appear locally highly ambiguous thus increasing the amount of backtracking that is necessary to successfully parse them.

5.4.2.3 PARSIFAL

A system employing an interesting approach to NLP which is diametrically opposed to the non-deterministic approach characterizing ATN based systems, is Marcus's PARSIFAL (Marcus, 1980; Winograd, 1983). This system was built to demonstrate that there is a theoretical significance to the *determinism hypothesis*. This hypothesis states that *given certain well-defined mechanisms, syntactical analysis [may be performed] deterministically*. (Winograd, 1983, p. 410.)

Most natural language understanding systems employ strategies intended to explore existing alternatives in an attempt to handle apparent syntactic ambiguity. For example, consider the sentences "Have the boxes in the boiler room thrown away!" and "Have the boxes in the boiler room been thrown away?". These two sentences appear structurally similar until the words "thrown", and "been", respectively, come into focus.

The idea on which PARSIFAL is based is that one may procrastinate assigning a syntactic structure to some given constituent until encountering an input word which resolves any existing ambiguity. This system uses a small, fixed-size buffer in which constituents are stored until their syntactic functions can be determined.

Although the resulting mechanism allows for extremely efficient parsing, one major drawback is that if the amount of look-ahead needed to resolve the apparent ambiguity is greater than the buffer size, then the system may choose an incorrect syntactical structure (or simply fail to produce one) (Winograd, 1983). Moreover, once a disambiguating word is read from the input, there is no way of retracing earlier steps in an attempt to correct a wrong decision (thus resulting in a parsing failure, although there may exist at least one possible syntactic interpretation).

Nevertheless, the sentences on which PARSIFAL fails to assign an appropriate syntactic structure are exactly those on which humans have trouble, also known as *garden path sentences* (Akmajian, 1990; Blank, 1989). A classic example of such a sentence is "The horse raced past the barn fell down." The part "raced past the barn" modifies the noun "horse." The alternative interpretation, which happens to be the one chosen by most human readers, is to initially treat "raced" as the main verb of the sentence. Although such sentences may not be compatible with the linguistic competence of some English speakers, they are nevertheless treated as grammatical by mainstream linguists (Akmajian, 1990, p. 371).

Concluding, PARSIFAL demonstrates that we may be capable of designing systems which understand a significant part of natural language, and whose formal power is equivalent to that of a finite-state machine. In addition to Marcus, several other researchers follow a similar deterministic, finite-storage approach to natural language understanding. These results are of major importance, since they have set the stage for stochastic approaches based on HMMs – a probabilistic version of a finite-state machine (see Section 4.2).

5.4.3 Semantic Analysis Systems

Semantic analysis systems differ from lexical and syntactic analysis systems in their ability to perform processing at the semantic level, in addition to the lexical and syntactic levels. This ability is a direct result of their encompassing knowledge related to the functionality or tasks associated with the application domain (semantic domain). These systems can be further subdivided according to the form in which they represent this semantic knowledge.

5.4.3.1 Implicit Semantic Representation

Systems acting as natural language interfaces to various interactive computer systems are classifiable under this category if they represent their semantic domain in terms of the underlying system's command language. Examples of such interfaces include LUNAR (Woods, 1973), and LIFER (Hendrix *et al.*, 1978). However, command language syntax is not as suitable for representing the semantic domain of an application, as a predicate or lambda calculus based knowledge representation language. This is because the former, since it is equivalent to a procedural description, disallows representation of semantic domain meta-knowledge which could be used in error detection, reasoning and recovery operations. Consequently, NLP systems employing implicit semantic representation are incapable of, or extremely ineffective in producing "intelligent" error messages regarding conceptual errors committed by the user.

5.4.3.2 Explicit Semantic Representation

Systems under this category represent their semantic domain in terms of an intermediate level, which facilitates the explicit identification of semantic domain elements known to the system. More analytically, the actual knowledge base of such a system encompasses declarative (or procedural) knowledge regarding the conceptual entities known to the system, the actions which manipulate these entities, and the relationships that exist among them. Examples of such NLP systems include SHRDLU (Winograd, 1983), and UNIX Consultant (Wilensky *et al.*, 1984; 1988).

5.4.3.3 LUNAR

One of the first systems to be implemented based on the ATN formalism is Wood's LUNAR system (Woods, 1973). LUNAR is a natural language interface to a database system built to translate a subset of English into the corresponding database queries. It interfaces with a database containing data about the mineral samples obtained from the Apollo-11 mission to the moon. LUNAR consists of an ATN used to store natural language knowledge, a lexicon containing approximately 3,500 words, and a target query language based on predicate calculus. An example of the translation process performed by this system follows.

```
(Do any samples have greater than 13 percent aluminum)
                        ⇓
(TEST (FOR SOME X1 / (SEQ SAMPLES) :T;
      (CONTAIN 'X1 (NPR* X2 / (QUOTE AL203))
                   (GREATERTHAN 13 PCT))))
```

5.4.3.4 Augmented Semantic Grammars

The semantic grammar formalism was invented by Burton for use in the SOPHIE NLP system (Brown and Burton, 1975; Burton, 1976). These grammars are equivalent in descriptive power to context-free grammars. The only difference is that in semantic grammars the syntactic and semantic aspects of the linguistic knowledge have been incorporated into a single framework. Although, this formalism has been effectively used for knowledge representation in limited application domains, it is incapable of dealing effectively with context-sensitive linguistic phenomena, such as parallel-association constructs, e.g., "Peter, Paul, and Mary like Bach, Debussy, and Stravinsky, respectively."

Hendrix *et al.* (1978) developed augmented context-free grammars (ACFGs), which augment semantic grammars by allowing for actions to be associated with each production rule in the grammar. These actions are equivalent to the actions associated with arcs in the ATN formalism. The semantics of associating an action with some production rule is that the specified action is executed if and only if the corresponding production rule is successfully matched against a given input constituent. This augmentation is shown to make ACFGs equivalent to a Turing machine in terms of generative/descriptive power. Although this formalism can effectively deal with various linguistic phenomena, it lacks conditions to be tested prior to examining a production rule. This deficiency results in reduced parsing efficiency.[23]

Manaris and Dominick (1993), introduced an additional augmentation to the ACFG formalism, namely the association of preconditions with production rules in the grammar. The resultant formalism, namely augmented semantic grammars (ASGs) has been shown to be powerful enough to recognize any language recognizable by a computing device, yet maintain a degree of "perspicuousness," and provide an effective mechanism for controlling combinatorial run-time behavior in parsing. [24] ASGs have been used to develop a variety of NLP applications. Examples include a natural language interface to operating systems (see next section); a natural language interface for Internet navigation and resource location; a natural language interface for text pattern matching; a natural language interface for text editing; and a natural language interface for electronic mail management (Manaris, 1994).

5.4.3.5 Natural Language Interfaces to Operating Systems

Manaris (1994) presents a natural language interface to UNIX and its subsequent porting to MS-DOS, VAX/VMS, and VM/CMS. Although it has some pragmatic-level knowledge incorporated into its knowledge base, it is best classified as a semantic analysis system as it does not maintain any knowledge on dialog structure. This system has been developed as a demonstration of the NALIGE user interface management system which can be used to construct natural language interfaces to interactive computer systems (see Section 6.4.1). The interface to UNIX handles a variety of user tasks including:

- **File Tasks:** copy, delete, display, edit, print, and send via e-mail
- **Directory Tasks:** create, delete, list contents, and display name

---

[23] Conditions to be tested as well as actions to be executed are part of other formally equivalent, but potentially more efficient formalisms such as augmented phrasal structure grammars (Sowa, 1984) and ATNs (Woods, 1970).

[24] Woods (1970), uses the term *perspicuity* to describe the inherent readability of context-free grammars, that is being able to directly determine the consequences of a production rule for the types of constructions permitted by the linguistic model (as opposed to other formalisms such as regular grammars and pushdown automata).

```
<MAKE-DIRECTORY> :
   [CONSTRUCT-V] <MAKE-DIRECTORY-SPEC> || `(create-directory :directory ,(V 2))
 | [CREATE-V]    <MAKE-DIRECTORY-SPEC> || `(create-directory :directory ,(V 2))
 | [MAKE-V]      <MAKE-DIRECTORY-SPEC> || `(create-directory :directory ,(V 2))
 | [MKDIR-V]     <MAKE-DIRECTORY-SPEC> || `(create-directory :directory ,(V 2))
 | [OPEN-V]      <MAKE-DIRECTORY-SPEC> || `(create-directory :directory ,(V 2))
 | [PRODUCE-V]   <MAKE-DIRECTORY-SPEC> || `(create-directory :directory ,(V 2)) ;

<MAKE-DIRECTORY-SPEC> :
   [*DIR-NAME*] || (V 1)
 | [DIR-N] [*DIR-NAME*] || (V 2)
 | [DIR-N] [NAMED-REF] [*DIR-NAME*] || (V 3)
 | [DIR-N] [REF-V] [*DIR-NAME*] || (V 3)
 | [*DIR-NAME*] [AS-REF] [DIR-N] || (V 1)
 | [NEW-ADJ] [DIR-N] [*DIR-NAME*] || (V 3)
 | [A-INDEF] [NEW-ADJ] [DIR-N] [*DIR-NAME*] || (V 4)
 | [A-INDEF] [DIR-N] [NAMED-REF] [*DIR-NAME*] || (V 4)
 | [A-INDEF] [DIR-N] [CALLED-V] [*DIR-NAME*] || (V 4)
 | [BLANK-ADJ] [DIR-N] [CALLED-V] [*DIR-NAME*] || (V 4)
 | [A-INDEF] [NEW-ADJ] [DIR-N] [CALLED-V] [*DIR-NAME*] || (V 5)
 | [A-INDEF] [DIR-N] [AND-CONJ] [CALL-V] [IT-N] [*DIR-NAME*] || (V 6)
 | [ME-PN] [AN-DEF] [EMPTY-ADJ] [DIR-N] [CALLED-V] [*DIR-NAME*] || (V 6)
 | [A-INDEF] [BRAND-ADV] [NEW-ADJ] [DIR-N] [CALLED-V] [*DIR-NAME*] || (V 6) ;
```

Figure 6. ASG Excerpt from UNIX Natural Language Interface
(reprinted with permission of World Scientific).

- **Other Tasks:** change user password, display user information, display users on system, list e-mail messages, and send e-mail messages

Figure 6 shows an excerpt from the augmented semantic grammar included in this interface.

5.4.4 Pragmatic Analysis Systems

Pragmatic analysis systems improve on the natural language understanding capabilities of semantic analysis systems. This is because the main objective of these systems is to participate in extended dialogues with users over some specific area of world knowledge. These systems employ a significant subset of world knowledge associated with a given application, in order to facilitate a deeper understanding of a given natural language input. More specifically, pragmatic analysis systems attempt to derive the deeper meaning or implications of natural language utterances by performing inference on pragmatic discourse elements, such as the goals of dialogue participants, social protocols associated with a given situation, and facts derived from earlier parts of the dialogue/story.

For example, consider the sentence “The gas is escaping!” in a story understanding application. Although the semantics of this sentence is clear, its pragmatics is ambiguous. More specifically, if the sentence is uttered by a chemistry instructor to a student performing an experiment in a chemistry lab, then a pragmatic analysis system might infer the following facts:

1. The student has been careless.
2. The instructor is displeased.
3. The student may receive a low grade in this lab.

On the other hand, if the dialogue is taking place in a San Francisco building following a major earthquake, the system might infer:

1. The earthquake has ruptured a gas line.
2. There is imminent danger of an explosion.
3. The building has to be evacuated immediately.

Pragmatic analysis is necessary for story understanding and discourse analysis. In the last few years, pragmatic analysis has been incorporated to a wide variety of applications including natural language and speech understanding interfaces to a variety of applications including electronic appointment scheduling, battlefield simulation, currency exchange information, electronic calendar, electronic mail, real-time machine translation, Rolodex, stock quote access, voice mail, and weather forecast access (Alexandersson *et al.*, 1997; Busemann *et al.*, 1997; Martin, 1996; Moore *et al.*, 1997; Waibel, 1996; Wauchope *et al.*, 1997).

5.4.4.1 UNIX Consultant

UNIX Consultant (UC) is a natural language understanding system whose objective is to advise users of the UNIX operating system (Wilensky *et al.*, 1984; 1988). This system allows users to obtain information about the usage of commands, such as command language syntax, on-line definitions of general UNIX terminology, and command-line debugging problems.[25] UC employs a pattern-action formalism for processing of natural language inputs, which is formally equivalent to an ATN. It has been implemented using the LISP and PEARL programming languages.

Actually, the objectives of UC are of a wider scope than those of the natural language interface applications discussed in the previous sections. This is because, instead of simply providing a natural language front-end to a particular operating system, UC attempts to improve a user's understanding of the UNIX environment by participating in possibly extended question-and-answer type of interactive sessions. Consequently, in addition to natural language understanding, it combines concepts from a number of AI areas, such as natural language generation, situation planning, and problem solving (Chin, 1983). The rationale for developing UC is to provide naive users (having to phrase some specific request in an unfamiliar command language) with a more attractive alternative than locating a knowledgeable user consultant and/or searching through some esoteric manual. UC employs a natural language analyzer able to produce user-friendly error feedback when faced with ill-formed input. This system incorporates an extensible knowledge base of facts about UNIX and the English language.

5.4.4.2 JANUS-II

Janus-II is a speech translator that operates on spontaneous conversational dialog in limited domains (Waibel, 1996). It currently includes vocabularies of 10,000 to 40,000 words and accepts input in English, German, Japanese, Spanish, and Korean. Its output can be any of the these languages.

The system consists of modules for speech recognition, syntactic and semantic parsing, discourse processing (contextual disambiguation), and speech generation. It incorporates HMMs and HMM-neural network hybrid techniques to generate the most promising word hypotheses. For parsing it employs semantic grammars within a pattern-based chart parser (Phoenix) and a stochastic, fragment-based generalized LR* parser. The result is a language-independent representation (an *Interlingua*) that is used by the generation part of the system to produce a spoken translation of the

[25] A presentation of a similar natural language understanding system, namely UNIX-TUTOR, appears in (Arienti *et al.*, 1989).

input in the desired output language. Although the Interlingua approach dates from the early MT days of NLP, it continues to be a very viable approach for the following reasons:

- It disassociates the syntactic structures of the input and output languages.
- It facilitates introduction of new languages to the system, as the linguistic knowledge which is specific to a given language can be contained within only a few modules of the system. [26]
- It allows generating output in any language, including the original input language. This facilitates feedback to the user, since, as the input gets converted to the Interlingua representation it is possible that errors, may have been introduced. This approach allows the user to verify that the system has "understood" the input correctly.

Janus-II has been used to develop several speech-translation application prototypes including a videoconferencing station with a spoken-language interpretation facility, and a portable speech translator running on a portable computer incorporating headphones and a wearable display for output.

As systems move up the knowledge-level classification, they acquire more capabilities in handling natural language phenomena. But what are some of these natural language phenomena and associated problem areas? The next section addresses these issues.

5.5 Problem Areas

Cercone and McCalla (1986) and Grosz *et al.* (1987) discuss many issues that arise in the context of processing natural language. These issues include augmenting linguistic coverage, generalizing parser capabilities and robustness, incorporating pragmatics and meta-knowledge (knowledge about the domain and the system), devising comprehensive knowledge representation schemes, porting linguistic knowledge, modeling the user, and handling various discourse phenomena. The latter include reference resolution, ambiguity resolution, handling ellipsis, monitoring user focus, handling incomplete knowledge, representing exceptions, summarizing responses, handling time dependencies, and dealing with hypothetical queries, references to system generated concepts, system knowledge updates, and user goals.

Weizenbaum (1976, p. 204) indicates that, similarly to Einstein's ideas on the relativity of motion, intelligence is also meaningless without a frame of reference. In everyday interaction, we provide such frames of reference, based our own cultural, educational, and social background and the situation at hand. The same can be argued for language competence and performance. Although it would be wonderful to have a modeling theory and associated algorithms that account for and handle the complete spectrum of linguistic issues arising in human-human interaction, this is not necessarily required for achieving effective human-computer interaction. This is because, human-computer interaction applications are always developed within specific frames of reference, that is specific application domains. Therefore, one should focus on methodologies for developing models which provide effective linguistic coverage in specific application domains. The next section discusses three methodologies which provide significant insights into the problem at hand.

---

[26] This point is best illustrated in the context of the Eurotra MT project which covers 72 language pairs (Arnold, 1986; Maegaard and Perschke, 1991); by isolating the language dependencies from as many modules of the architecture as possible, only the language dependent modules have to be re-written, as new languages are introduced.

5.6 Linguistic Coverage

Furnas *et al.* (1987; 1983) discuss issues related to spontaneous language use at the user interface. Although they focus on the selection of natural language words to describe semantic elements (objects, actions) in command language interfaces, their study provides intuitions regarding the potential for unconstrained natural language in human-computer interaction, as well as the development of constrained linguistic models. They indicate that the variability of spontaneous word choice is "surprisingly large," in that in all cases they studied, the probability that the same natural language word is produced by two individuals at the interface to describe a single semantic entity is less than 0.20.

Specifically, they discuss six general schemes for assigning natural language descriptions to semantic elements at the interface. Additionally, they present results from several experiments designed to evaluate each of these schemes with respect to the resultant systems' performance measured against expected user behavior. They conclude that, out of the six schemes, the following three are the most important: *weighted random one-description-per-object* scheme, which they also refer to as the *armchair* model, *optimized one-description-per-object* scheme, which will be subsequently referred to as the *optimal single-description* model, and *optimized multi-description-per-object* scheme, which will be subsequently referred to as the *optimal multi-description* model.

5.6.1 Armchair Model

The armchair model corresponds to the standard approach used in deriving command names and syntactical structures for input languages to interactive computer systems, such as operating environments, and information systems (Furnas *et al.*, 1987). The underlying idea is that each semantic element available in the system is associated with a single surface description. Moreover, this description is chosen by a system expert, usually the system designer, according to his/her personal belief ("armchair" introspection) as to what constitutes a proper naming convention.

The data derived from the conducted experiments suggest that this popular method is highly unsatisfactory. More analytically, it is observed that, if the natural language description of a semantic element known to a system has been derived using the armchair method, untutored subjects will fail 80 to 90 percent of their attempts to successfully access this semantic element.[27] It should be noted that the experiments conducted by Furnas *et. al.* concentrate on semantic element descriptions which consist of a single word. However, they suggest that usage of multi-word descriptions is certain to result in even lower performance. Clearly, this is highly undesirable for human-computer interaction.

Actually, the source of the problem is that:

> [T]here are many names possible for any object, many ways to say the same thing about it, and many different things to say. Any one person thinks of only one or a few of the possibilities. Thus, designers are likely to think of names that few other people think of, not because they are perverse or stupid, but because everyone thinks of names that few others think of. Moreover, since any one person tends to think of only one or a few alternatives, it is not surprising that people greatly overrate the obviousness and adequacy of their own choices (Furnas *et al.*, 1983, p. 1796).

The usual solution has been for system designers to rely on the fact that, through practice, users will eventually learn this linguistic model – a prescriptive linguistics approach. When the semantic coverage of the system is relatively small, this method works fairly well. Actually, it has been

[27] Furnas *et. al.* (1987, p. 966) point to similar results reported in other studies.

shown that, for small systems, using a semantically unrelated and/or random naming convention has little or no significant effect on initial learning by untrained users (Landauer *et al.* 1983). But, if the system is large and its use is intermittent, this approach is unacceptable from a human factors point of view.

5.6.2 Optimal Single-Description Model

The next alternative is to discard the convenience sampling method of the armchair model, i.e., focusing on the personal taste or intuition of a single individual, and instead attempt to employ a more objective selection method. This method takes into consideration the preferences of numerous subjects and use them to derive an optimal single description for each element in the system's semantic domain based on frequency distribution. This approach has been popular in the context of command-line interfaces where there is usually only one way to describe one action (Furnas *et al.*, 1987; 1983).

It has been shown that this method approximately doubles the chances of serendipity over the armchair model. That is an untrained user has double the chances in generating the appropriate language understood by the underlying system. Although this is a significant improvement over the armchair method, it is not good enough to be used in practice, since, even with this method, subjects fail to communicate successfully with the underlying system 65 to 85 percent of the time.

5.6.3 Optimal Multi-Description Model

The main problem with the previous models is that they fail to take into consideration the wide variety of language that users spontaneously produce with respect to a specific semantic element. As a result, the performance of the resultant systems with respect to untrained users is highly unsatisfactory.

The optimal multi-description model differs from the armchair and optimal single-description models in that it provides the resultant system with an "unlimited" number of aliases describing individual semantic elements (Furnas *et al.*, 1987; 1983). More specifically, the major characteristics of this model are the following: The system has the capability to acquire over time all descriptions which users may use to refer to a specific semantic element. Data is also collected to identify the alternate meanings associated with each of the above descriptions. In the case of semantic ambiguity, the system presents the user with the list of alternative interpretations.

The results of the conducted experiments indicate that there is significant improvement over the previous models. Specifically, when the number of aliases reaches 20, the percentage of success is within 60% to 95%. The significant improvement over the previous models is due to the fact that most descriptions produced spontaneously by users are "rare" in nature, in the sense that there exists a very small probability that other users will choose the exact same description to refer to a given semantic element.

This spontaneous generation of natural language descriptions tends to follow Zipf's distribution (Zipf, 1949). Specifically, if we plot the logarithm of the frequencies of such descriptions against the logarithm of the corresponding ranks, we produce a straight line having a slope of negative one. The interpretation of this observation is that few linguistic constructs are used very frequently, whereas the vast majority of them are used only rarely. A similar observation is made in the context of the Brown corpus, a representative corpus of American English, where about half of the world types appear only once – approximately 32,000 out of 67,000 total word types (Marcus, 1994).

Due to the complexity of natural language, it is practically impossible to expect that an NLP system will ever be able to collect a "complete" model of natural language even with respect to a single user. Nevertheless, it could always improve its performance in an asymptotic manner.

The optimal multi-description model is consistent with the current success of corpus-based stochastic and connectionist approaches. These approaches, in addition to being statically trained through corpora, which contain a wide variety of linguistic data, are also capable of dynamic adaptation based on the actual linguistic performance of users. An example of this is found in many speech recognition systems which improve their performance by constantly updating their stochastic model of phonetic knowledge during interaction with the user. Lately, it has been shown that corpora may also be utilized in the automatic development of symbolic models; this might lead to results demonstrating that the relative success of stochastic and connectionist approaches, over traditional symbolic ones, may be due to their corpus-based nature and interactive learning capabilities, as opposed to some other inherent characteristic (Marcus, 1994). This possibility is supported by Wegner's (1997) claim on how "all forms of interaction transform closed systems to open systems and express behavior beyond that computable by algorithms" (p. 83).

5.6.4 Controlled Languages

A similar approach has been to find a compromise between unconstrained natural language and the need for single unambiguous descriptions required by the underlying computer systems through the use of *controlled languages* (Church and Rau, 1995). This compromise has been to restrict the natural language available at the interface in a systematic and easily remembered fashion while simultaneously allowing for some freedom of linguistic expression. These restrictions may be imposed at any level of linguistic knowledge, such as phonetic, lexical, syntactic, or semantic. Actually, in a sense, the functionality of the underlying applications already restricts the pragmatics. Examples include certain speech recognition systems (phonetic, lexical), spelling checkers (lexical), style checkers (lexical, syntactic, semantic).

This approach is very successful in developing effective linguistic models for human-computer interaction. This is because the interface can "subliminally educate" users through feedback (Slator *et al.*, 1986), while the linguistic model of the interface gradually adapts to users' linguistic preferences through appropriate extensibility mechanisms. Of course, these mechanisms are tightly bound to the type of linguistic modeling approach employed, namely symbolic, stochastic, connectionist, or hybrid. The only weakness of this approach is that in some cases, due to the size of the linguistic model, or the deviance of the user from the linguistic model, data collection (adaptation) may continue for a long time before asymptotic performance is approached.[28]

6. Multimodal Interaction

Until the early 1980s, the prevalent interactive style was command entry. However, in the mid 1980s another user interface paradigm became popular, namely Windows, Icons, Menus, and Pointing (WIMP). This introduced new possibilities, such as direct manipulation, which have resulted in today's graphical user interfaces.

It is now clear that the user interface designer has several building components (interaction styles, input/output devices) available from which to develop an effective interface. In terms of interaction

[28] This is the case in many current speech recognition systems as their linguistic models make implicit assumptions regarding the users' educational, cultural, and native-language backgrounds. For instance, users with non-American accents, or speech impediments cannot fully benefit from such applications.

styles these include command-line entry, menus and navigation, question and answer dialogs, form fill-in, natural language, and direct manipulation. In terms of devices these include keyboard, display, mouse, trackball, joystick, touch screen, microphone, speakers, video camera, dataglove, datasuite, 3D tracker, and various other types of sensors and actuators (Priece *et al.*, 1994; Shneiderman, 1993). Naturally, this list is constantly evolving, as it depends on the state-of-the-art in technology for interfacing with human communication channels, namely sight, touch, taste, smell, and hearing.

According to Coutaz and Caelen (1991), *a multimodal user interface combines various interaction styles, is equipped with hardware for acquiring and rendering multimodal expressions in real time, must select appropriate modality for outputs, and must "understand" multimodal input expressions.* This is contrasted to a multimedia interface which *acquires, delivers, memorizes, and organizes written, visual, and sonic information, but ignores the semantics of the information it handles.* [29]

Additionally, Coutaz and Caelen (1991) identify a taxonomy for multimodal user interfaces as follows:

- an *exclusive* multimodal user interface allows one and only one modality to be used in rendering a given input/output expression;
- an *alternative* multimodal user interface allows alternative modalities to be used in rendering an input/output expression;
- a *concurrent* multimodal user interface allows several input/output expressions, possibly in different modalities, to be rendered in parallel; and
- a *synergic* multimodal user interface allows components of input/output expressions to be rendered in different modalities.

Since natural language is one of the prevailing modalities for human-human interaction, it is a natural modality for effective, user-friendly, human-computer interaction. This has lead to an extension of the WIMP paradigm, namely WIMP++. The latter incorporates additional modalities, such as natural language and animation (Hirschman and Cuomo, 1994). However, some HCI researchers indicate that although natural language at the interface has several advantages, it is not a panacea. Shneiderman (1993, p. 167), for example, states that "[p]eople are different from computers, and human-human interaction is not necessarily an appropriate model for human operation of computers." So, a question that arises is "is there a benefit from having natural language modalities available at the interface?". The next section reports on three studies that address this issue.

6.1 Effects of Natural Language on User Performance

Ledgard *et al.* (1980) report on an experiment conducted at the University of Massachusetts, Amherst, in association with Digital Equipment Corporation. One of this study's assertions is that an interactive system should facilitate use of familiar, descriptive, everyday words and legitimate English phrases at the interface. The experiment's objective was to test the above hypothesis and, simultaneously, demonstrate the effects that human engineering can have on commercially available software in terms of human efficiency, performance, and satisfaction.

The study involved users with varying degrees of computing experience. Users were divided into inexperienced, familiar, and experienced groups. Additionally, the study utilized two semantically

[29] Another term for multimodal user interfaces is *intelligent multimedia interfaces* (Maybury, 1993).

equivalent versions of an interactive computer system – a text editor. The differences between the two versions were confined at the input syntax level. One employed a natural language front-end which accepted flexible, yet constrained natural language – this was the only input modality provided. The other understood the standard computer system syntax which consisted of abbreviated, cryptic keywords and rigid syntactical rules. One half of the subjects were randomly assigned to the natural language equipped system, whereas the other half were assigned to the standard system. Subsequently, the subjects were given a number of tasks to be performed by the assigned computer system. These tasks were compound in nature, in the sense that each required a sequence of actual system commands in order to be accomplished. On all measures, and regardless of user expertise, performance using the natural language front-end proved superior to performance using the regular command language. Finally, they conducted a post-experimental study to test asymptotic performance. They found that significant performance differences between the two versions remained even after a long time of use.

Similarly, Napier *et al.* (1989) report on a closely related study comparing the performance of novice users using two syntactically different (but semantically equivalent) versions of the Lotus 1-2-3 spreadsheet system. The first version employed a restricted natural language front-end, whereas, the second one employed the standard user interface, i.e., command language, and menu selection. They conclude that there is a clear and consistent advantage in favor of the natural-language equipped system.

Finally, Pauch *et al.* (1991) compare the use of discrete speech input against menu selection. They found considerable performance increase (21%) in users who had access to both voice selection and mouse, as opposed to those with access only to mouse.

Although all the above studies indicate that natural language is beneficial at the interface, one should not assume that this always the case. For instance, Napier *et al*. (1989) point to other studies that are critical of natural language being the only available modality at the interface. Since natural language is not appropriate for every type of information exchange in human-human interaction, it is not surprising that this may also be the case in certain types of information exchange in human-computer interaction. Obviously, investigation of the application domain, as well as good design of the interface are essential to user acceptance – regardless of whether the interface incorporates natural language or not. That is developers should not religiously employ a single modality, such as natural language, at the interface, but instead select the best combination among the set of available modalities for building an interactive system.

6.2 Natural Language Widgets

There already exist several popular toolkits and environments, such as X-Window and MS-Windows, which facilitate development of user interfaces by making available a collection of user interface building elements. How could we possibly extend such collections to incorporate natural language building blocks?

Hall *et al.* (1996) introduce the concept of *natural language edit controls* (NLECs). NLECs are special user interface building elements which facilitate use of natural language for specific input expressions in multimodal user interfaces. NLECs are an extension of traditional text entry controls, in that they facilitate rendering a portion of a user input in natural language. When combined with the other traditional controls available at the interface, such as buttons, check boxes, slidebars, and menus, they can result in the most effective interface for a given application. Although NLECs are a excellent idea, they are nevertheless limited in that they deal only with typewritten natural language.

One could extend this concept to a more general natural language widget, of which an NLECs is only an instance. Specifically, a *natural language widget* is defined as *a user interface widget that incorporates a generic NLP engine capable of handling a limited domain of discourse*. Natural language widgets may come in different flavors, as they may employ any of the approaches for modeling natural language (see Section 4), may incorporate as many or as few knowledge levels as necessary (see Section 5.3), and may implement any kind of natural language application that is relevant to the task at hand, such as speech recognition, voice identification, word spotting, and spoken or written natural language generation.

6.3 Modality Integration

Oviatt and Cohen (1991, p.70) indicate that modalities

> physically constrain the flow and shape of human language just as irresistibly as a river bed directs the river's current. … Although communication modalities may be less visually compelling than the terrain surrounding a river, it is a mistake to assume that they are less influential in shaping the [information] transmitted within them.

Spoken or written natural language is clearly not appropriate for all tasks at the user interface. This is supported by the fact that human-human interaction includes additional modalities such as pointing, drawing and various other gestures (Oviatt *et al.* 1997). Specifically, some things can be best pointed at, or drawn, or selected from a list of alternatives. For this reason, natural language needs to be combined with other traditional or non-traditional modalities in order to construct effective user interfaces. This introduces the question "what tasks (or portions of tasks) are best specified through natural language?".

6.3.1 Effective Use of Natural Language

Hall *et al.* (1996) provide a decision procedure for when to employ natural language over *deictic controls* – controls utilizing a pointing device, such as mouse, pen or finger. Extending on their ideas, in order to accommodate both actions and objects, it appears that natural language is best for input tasks where the set of semantic elements (entities, actions) from which to choose is

- large, unfamiliar to the user, or not well-ordered; or
- small and unfamiliar, with no *bijective mapping*[30] between that set to a set of familiar elements.

Figure 7 shows the extended decision procedure for selecting between natural language widgets and traditional deictic controls.

6.3.2 Synergic Multimodal Interfaces with Natural Language

Oviatt *et al.* (1997) discuss integration patterns of input modalities in the context of synergic multimodal interfaces that include natural language. They report that multimodal interaction is most frequent for tasks involving spatial location, and somewhat frequent for tasks involving selection. Integration patterns consist of sequential, simultaneous, point-and-speak, and compound rendering of input expressions. In temporal terms, written input may overlap with spoken input subexpressions – with written (pen) input providing location information at the beginning of an

[30] A bijective mapping is an one-to-one mapping between two sets which includes all elements from both sets.

```
IF (familiar set) THEN
   IF (large set) AND
      (not well-ordered set) THEN

      use a text entry control // user types familiar object

   ELSE

      use deictic control      // user selects familiar object

ELSE                            // unfamiliar
   IF (exists bijective mapping) AND
      ((small set) OR (well-ordered set)) THEN

      use a deictic control    // user selects familiar object
                               // system maps to unfamiliar object
   ELSE

      use a natural language widget
```

Figure 7. Extended Decision Procedure for Use of Natural Language Widgets (adapted from Hall *et al.*, 1996)

expression. For example, in the context of a user interface to an operating system, a user may circle two file icons while saying "delete these". Additionally, spoken and written modalities supply complementary (as opposed to redundant) semantic information. For instance, a user might type in a file name and say "edit this".

In a different study, Cohen *et al.* (1997) have found that multimodal interaction makes users generate simpler linguistic constructs than unimodal speech interaction. For example, in order to create a red line between two <x,y> grid coordinates a user might say "create a red line from one three five point eight to four six seven point nine".; whereas (s)he could draw a line on the map while saying "red".

One example of an multimodal interface which includes speech input is CommandTalk (Moore *et al.*, 1997). This is a user interface to the ModSAF battle-field simulator which allows the use of natural language to create forces, assign missions, modify missions during execution, and control various simulation functions, such as map display control. Another example of a multimodal interface that incorporates natural language is QuickSet (Cohen *et al.*, 1997). QuickSet combines pen and speech to manage distributed interactive simulations. It combines widgets (agents) that handle speech recognition, written natural language, and gesture recognition (pen input). Finally, MedSpeak is a multimodal interface for creating radiology reports (Lai and Vergo, 1997). It accepts input via speech (dictation and command modes), mouse, and keyboard.

6.4 User Interface Management Systems

User interface management systems (UIMSs) are environments which facilitate the specification, design, implementation, evaluation, and run-time support of user interfaces (Bass and Coutaz, 1992). Although they have been used extensively in developing graphical user interfaces, only recently have they been used in the context of natural language interfaces. Such systems are extremely significant since they facilitate the development of natural language widgets (see Section 6.2).

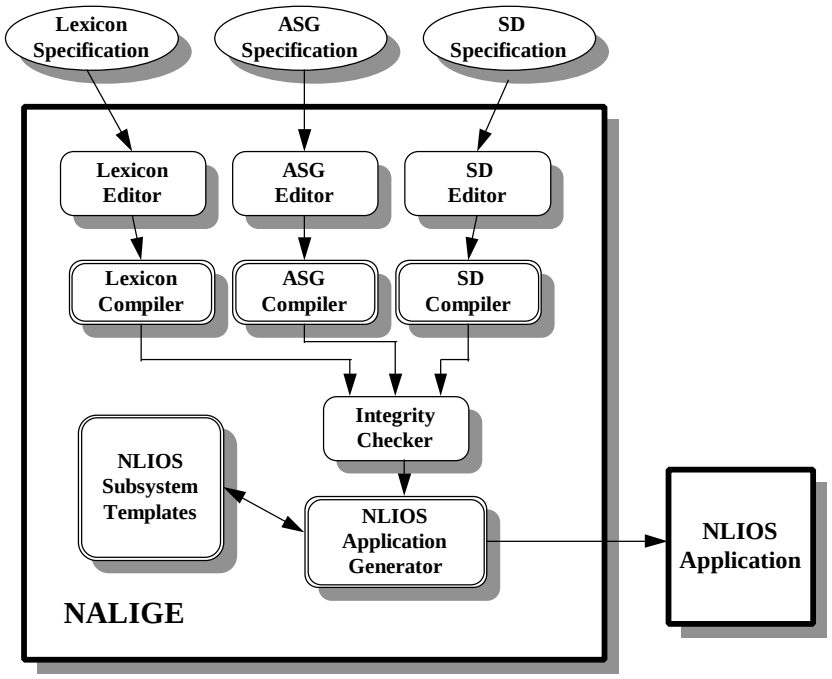

Figure 8. NALIGE Architecture
(reprinted with permission of World Scientific).

6.4.1 NALIGE

An example of such an environment is NALIGE (Manaris and Dominick, 1993; Manaris, 1994). NALIGE facilitates the development of natural language interfaces to interactive computer systems through the use of high-level specifications. These specifications describe the linguistic model to be incorporated in the interface in terms of lexical, syntactic, semantic, and pragmatic knowledge.

As shown in Figure 8, the NALIGE environment consists of several subsystems, namely:

- **Specification editor modules:** These editors provide context-sensitive assistance in the development of the NALIGE input specifications.
- **Specification compiler modules:** These compilers accept the input specifications and convert them to an efficient internal representation (declarative and procedural knowledge).
- **Specification integrity-checker module:** As the input specifications sometimes have to reference the same entity, e.g., a token, this module performs checks to enforce inter-specification integrity.
- **Subsystem template module:** This module contains generic procedural components, such as code templates for a lexical analyzer, a parser, a target-code generator, a low-level interface to the underlying system, a knowledge-base manager, and an error handler.
- **Application generator module:** This module combines the necessary code templates with the declarative and procedural knowledge produced by the compilers to generate an operational natural language interface.

6.4.2 SpeechActs

Another example of such a system is SpeechActs (Martin *et al.*, 1996). This is a user interface management system for developing loosely-coupled speech understanding applications. The architecture facilitates the incorporation of commercial speech recognizers as front-ends to interface applications. Currently, the system is compatible with the following continuous-speech recognizers: BBN's Hark (Smith and Bates, 1993), Texas Instruments' Dagger (Hemphill, 1993), and Nuance Communications (Digalakis and Murveit, 1994).

6.5 Development Methodologies

There exist many different system development methodologies employed in human-computer interaction (Priece *et al.* 1994). One of the most promising is the Star model (Hartson and Hix, 1993) (see Figure 9).

This model encompasses a user-centered approach,[31] since it provides for constant evaluation of all aspects of system development by users and experts. It stresses the idea that system development activities should follow a flexible order. It facilitates top-down (analytic), bottom-up (synthetic), and iterative-refinement types of development – the latter in tighter, smaller loops than spiral development methods. Finally, it emphasizes rapid prototyping and incremental development (Priece *et al.*, 1994).

In the context of developing natural language widgets, the Star model may be adapted as follows:

- **Task analysis:** Identify the tasks to be performed through the natural language widget. This could be performed in various ways, including study of corpora collected through Wizard-of-Oz techniques.[32] This phase is essential in that it provides information about the functionality that the widget is to provide through its "understanding" of natural language.
- **Linguistic analysis:** Specify (or automatically capture) the sublanguage to be modeled. Depending on the expected size of the sublanguage and the knowledge modeling approach (symbolic, stochastic, connectionist, hybrid), this phase might include specification of vocabulary, syntax, semantic elements (entities, actions), or dialog structure.
- **Conceptual design:** Identify the internal architecture of the widget in terms of knowledge components, procedural modules, and their interaction. A UIMS can greatly assist during this phase in that it may provide design support based on existing linguistic knowledge and processing modules.
- **Prototyping:** Develop an operational prototype of the system. A UIMS can greatly assist during this phase in that it may combine the sublanguage model captured during the linguistic analysis phase with existing knowledge and processing modules to construct a functional prototype.
- **Evaluation:** Evaluate the deliverable of any of the above phases. A UIMS can greatly assist during this phase by providing specific benchmarks and automated evaluation tools to be used

[31] The basic idea behind user-centered design is to incorporate end-users throughout the development of a product or application. This is somewhat intrinsic in the process of collecting data for stochastic or connectionist architectures, since users are involved in the development by contributing to the linguistic model to be incorporated in the NLP application; however, user involvement needs to extend to every step in the system development process.

[32] Wizard-of-Oz techniques employ simulated NLP systems, which collect linguistic data from users. The systems' natural language understanding capabilities are simulated by remotely located humans.

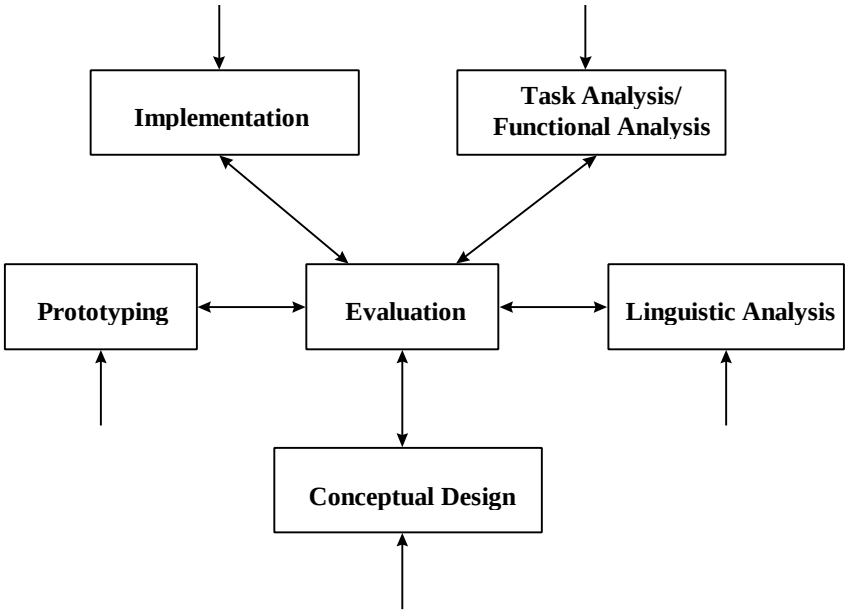

Figure 9. The Star Model for NLP System Development
(adapted from Priece, *et al.*, 1994)

in formative and summative evaluation. These benchmarks and tools should facilitate utilization of user expertise (possibly through collected corpora) to test and measure the effectiveness or relative completeness of any of the above deliverables.

Additional information on user interface development, including task analysis, design, and prototyping may be found in (Day and Boyce, 1993; Priece *et al.*, 1994; Shneiderman, 1993). Significant work has been carried out in NLP system evaluation (Hirschman and Cuomo, 1994; King, 1996; Moore, 1994b; Pallett *et al.*, 1994; Spark Jones, 1994). However, existing techniques and methodologies require additional development, so that they (a) take into account user tasks and environments, in terms of the influence that these have on system development and performance, and (b) address natural language as one of many available modalities, in the context of multimodal human-computer interaction (Spark Jones and Galliers, 1996).

7. Conclusions

The field of natural language processing has entered its sixth decade. During its relatively short lifetime, it has made significant contributions to the fields of human-computer interaction and linguistics. It has also influenced other scientific fields such as computer science, philosophy, mathematics, statistics, psychology, biology, and engineering by providing the motivation for new ideas, as well as a computational framework for testing and refining existing theoretical assumptions, models, and techniques. Finally, it has impacted society through applications that have shaped and continue to shape the way we work and live our lives.

But other events have happened in these fifty years. For instance, it was initially believed that no more than a handful, so to speak, of computers would ever be needed around the world. It was also predicted that enabling these machines with (artificial) intelligence would be only a matter of a few years' worth of research and development. Both predictions were wrong. They did, however, set

the stage for a ping-pong effect felt throughout the history of the field: On one side, enter overzealous critics who interfere with creative thinking and vision, e.g., the ALPAC report and the dampening effect it had on the field's evolution; on the other side, enter overzealous enthusiasts who make unrealistic, full-of-hype claims and promises, e.g., the translation bureaus which had opened up in big cities promising fully-automatic, high-quality MT services – only to close down a in few months (Josselson, 1971). Numerous examples of this effect can be seen throughout the evolution of the field. Assuming that historical patterns can provide insights about the future, it is clear that we need to support visionary, far reaching, yet well-founded research ideas, while keeping our feet on the ground and maintaining a critical view with respect to our intellectual and modeling limitations and their effect on the capabilities and potential of technology.

In these fifty years, computers have become ubiquitous. Consequently, the field of human-computer interaction has become extremely important as it focuses on bridging the communicative gap between humans and machines. This is accomplished by studying the nature of human communication and by deriving models which attempt to augment the flexibility, learnability, robustness, and overall habitability of computing tools. It is hoped that results from this field will increase our quality of life by facilitating the seamless integration of computing devices into the fabric of society; computer users will be able to focus more on the things they want to accomplish, as opposed to the actual user interfaces.

In terms of natural language in human-computer interaction, we now have several linguistic modeling approaches available, namely symbolic, stochastic, connectionist, and hybrid. Currently, we have a relatively good understanding of the symbolic approach, in that it seems that we have mapped out the limits of the region it best addresses within the NLP problem space. Based on our relatively incomplete understanding of stochastic and connectionist approaches, it appears that these address a different region of the NLP problem space, although some overlap might exist with respect to the symbolic approach. Moreover, symbolic techniques traditionally adhere to the armchair and optimal single-description models, and thus have been generally ineffective from the perspective of users – that is users other than the system developer(s). On the other hand, corpus-based stochastic and connectionist techniques inherently adhere to the optimal multi-description model, and thus have already produced a wide variety of useful speech- and text-based systems.

Although corprus-based stochastic and connectionist models appear to be more effective than traditional symbolic ones, they are nevertheless bound to some notion of "statistical average" with respect to users' linguistic performance, and thus their effectiveness depends on a given user's proximity to that "average." Such models are usually capable of adapting dynamically, and thus are considerably more flexible and robust than symbolic ones. Nevertheless, their adaptation effectiveness is also bound to the user's proximity to the modeled "average." For instance, users with speech impediments cannot benefit from the latest advances in speech recognition technology, because such technology is geared towards the "average" user – for obvious marketability reasons. Therefore, as we focus on natural language-enabled interfaces dealing with a wide spectrum of user needs and instances of linguistic competence/performance, our attention needs to shift from general corpora, whose development is expensive and error-prone, to effective methodologies for developing specialized corpora. Such methodologies may be incorporated into user interface management systems that facilitate, among other things, effective development of linguistic models for well-defined user groups – a truly user-centered approach.

In terms of multimodal interaction, we need to continue research on integrating natural language with other modalities. This has already been recognized by funding agencies around the world, since in the last few years, they began providing support for basic and applied research in human multimodal communication utilizing among other modalities speech, text, and images (Strong, 1996). Moreover, we need to focus on techniques for intermodal translation of communicative information. This will be of great benefit in situations were some modalities are temporarily or

permanently unavailable, either because of the task at hand, or because of the user's capabilities (or lack thereof). Specifically, major results have been achieved in the context of converting text to speech; such results have been incorporated into text-reading software utilizing speech synthesis technology. Nevertheless, certain problems still remain, such as dealing with the two-dimensional nature of text (and the ease of scanning it affords), compared to the one-dimensional nature of speech. Such research will be extremely valuable to users with certain motor and/or visual disabilities in that it will facilitate interaction with various computing or computer-controlled devices. The benefits of related applications are immense, considering the new possibilities they open to users with disabilities in terms of access to information and control/manipulation of immediate and remote physical environments (Muller *et al.*, 1997).

In terms of the future, in the short-term, we should experience benefits in various aspects of everyday life from the momentum of the latest research and development efforts, such as dialog-based speech-understanding telephony applications. Actually, several development environments are already beginning to emerge for dialogue-based speech processing telephony applications. Additionally, we could see various speech-enabled and voice-related applications appear dealing with security, voice dialing, voice identification, language identification, and language translation (Flanagan, 1994; Wilpon, 1994).

As we continue to improve our appreciation of the strengths and weaknesses of linguistic modeling approaches, and our understanding of the nature and use of language, we will become more effective in addressing NLP problems. Considering the boom of the NLP industry within the last decade, it is safe to assume that we have reached a critical point in our understanding of modeling approaches, linguistic phenomena, application requirements, and compromises that we can afford. Clearly, some approaches appear more suitable to specific applications over other approaches, e.g., stochastic approaches to speech recognition. Nevertheless, there seems to be a complexity associated with linguistic model development that transcends modeling approach. Some believe that this is due to the language phenomenon itself (Ristad, 1993). It is highly probable that achieving truly effective, natural human-computer interaction will be the next bottleneck, in that our inability to understand "how it is that we do what we do" gets in the way of accomplishing this field's ultimate goal. This has been the case with other scientific endeavors that study aspects of human existence, such as biology, psychology, and cognitive science. Nevertheless, although our models of natural phenomena – such as language – will most probably always remain approximations, it is through this quest for knowledge about self that we are finding out more about who we are, how to be more effective in doing what we do, and thus contribute to the evolution of society and ourselves.

ACKNOWLEDGMENTS

The author would like to thank István Berkeley, Adrienne Broadwater, Subrata Dasgupta, Anthony Maida, Renée McCauley, and Brian Slator for their invaluable comments and suggestions; Brian Slator is acknowledged for his contribution to an early outline – especially on the phases of NLP evolution; István Berkeley for discussions on philosophical and connectionist issues; and Eleni Efthimiou for providing several important references.

REFERENCES

In addition to the references cited above, the following list contains supplementary references, which are closely related to the topics raised, and thus may be of interest to the reader.

Abney, S. (1997). Part-of-Speech Tagging and Partial Parsing. In "Corpus-Based Methods in Language and Speech Processing" (S. Young, and G. Bloothooft, eds.), pp. 118-136. Kluwer Academic Publishers, Dordrecht, The Netherlands.

Akmajian, A., Demers, R. A., Farmer, and A. K., Harnish, R. M. (1990). "Linguistics – An Introduction to Language and Communication," 3rd ed. The MIT Press, Cambridge, Massachusetts.

Allen, J. (1994a). "Natural Language Understanding," 2nd ed. Benjamin/Cummings, Redwood City, California.

Allen, J. (1994b). Linguistic Aspects of Speech Synthesis. In "Voice Communication Between Humans and Machines" (D. B. Roe, and J. G. Wilpon, eds.), pp. 135-155. National Academy of Sciences, Washington, District of Columbia.

Alexandersson, J., Reithinger, N., and Maier, E. (1997). Insights into the Dialogue Processing of Verbomil. *In* "Proceedings of Fifth Conference on Applied Natural Language Processing," Association for Computational Linguistics, Morgan Kaufmann, San Francisco, California, pp. 33-40.

Arienti, G. and Cassaniga, T., Gardin, F., and Mauri, M. (1989). UNIX-TUTOR: An Experiment for the Use of Deep Knowledge for Tutoring. *In* "INFORMATION PROCESSING 89" (G. X. Ritter, ed.), pp. 569-574, Elsevier Science Publishers B. V., North-Holland.

Arnold, D. (1986). Eurotra: A European Perspective on MT. *Proceedings of IEEE* 74(7), 979-992.

Bar-Hillel, Y. (1960). The Present Status of Automatic Translation of Languages. *In* "Advances in Computers (F. L. Alt ed.), pp. 91-163. Academic Press, New York.

Bass, L., and Coutaz, J. (1991). "Developing Software for the User Interface." Addison-Wesley Reading, Massachusetts.

Bates, M. (1994). Models of Natural Language Understanding. In "Voice Communication Between Humans and Machines" (D. B. Roe, and J. G. Wilpon, eds.), pp. 238-254. National Academy of Sciences, Washington, District of Columbia.

Baum, L.E., and Petrie, T. (1966). Statistical Inference for Probabilistic Functions of Finite State Markov Chains. *Ann. Math. Stat.* 37, 1554-1563. Cited in (Rabiner and Juang, 1993).

Baum, L.E., Petrie, T., Soules, G, and Weiss, N. (1970). A Maximization Technique Occurring in the Statistical Analysis of Probabilistic Functions of Markov Chains. *Ann. Math. Stat.* 41(1), 164-171. Cited in (Rabiner and Juang, 1993).

Baum, L. E. (1972). An Inequity and Associated Maximization Technique in Statistical Estimation of Probabilistic Functions of Markov Processes. *Inequalities* 3, 1-8.

Biermann, A. W. (1976). Approaches to Automatic Programming. *In* "Advances in Computers (M. Rubinoff and M. C. Yovits, eds.), pp. 1-63. Academic Press, New York.

Blank, G. D. (1989). A Finite and Real-Time Processor for Natural Language. *Commun. ACM* 32(10), 1174-1189.

Bobrow, D. G., Kaplan, R. M., Kay, M., Norman, D., Tompson, H., and Winograd, T. (1977). GUS, A Frame-Driven Dialog System. *Artificial Intelligence* 8, 155-173. Reprinted in (Grosz, *et al.* 1986).

Bod, R., and Scha, R. (1997). Data-Oriented Language Processing. In "Corpus-Based Methods in Language and Speech Processing" (S. Young, and G. Bloothooft, eds.), pp. 137-173. Kluwer Academic Publishers, Dordrecht, The Netherlands.

Bonarini, A. (1993). Modeling Issues in Multimedia Car-Driver Interaction. In "Intelligent Multimedia Interfaces" (M. T. Maybury, ed.), pp. 353-371. The MIT Press, Cambridge, Massachusetts.

Booth, A. D., and Locke, W. N. (1955). Historical Introduction. *In* "Machine Translation of Languages," (W. N. Locke and A. D. Booth, eds.), pp.1-14. The Technology Press of MIT and John Wiley, New York.

Brown, J. S., and Burton, R. R. (1975). Multiple Representations of Knowledge for Tutorial Reasoning. *In* "Representation and Understanding" (D. G. Bobrow, and A. Collins, eds.), pp. 311-349. Academic Press, New York.

Burger, J. D., and Marshall, R. J. (1993). The Application of Natural Language Models to Intelligent Multimedia. *In* "Intelligent Multimedia Interfaces" (M. T. Maybury, ed.), pp. 174-196. The MIT Press, Cambridge, Massachusetts.

Burton, R. R. (1976). Semantic Grammar: A Technique for Efficient Language Understanding in Limited Domains. Ph.D. Thesis, University of California, Irvine.

Busemann, S., Declerck, T., Diagne, A. K., Dini, L., Klein, J., and Schmeier, S. (1997). Natural Language Dialog Service for Appointment Scheduling Agents. *In* "Proceedings of Fifth Conference on Applied Natural Language Processing," Association for Computational Linguistics, Morgan Kaufmann, San Francisco, California, pp. 25-32.

Carbonell, J. (1996). Foreword. *In* Franz, A. "Automatic Ambiguity Resolution in Natural Language Processing." Springer-Verlag New York.

Caudill, M. and Butler, C. (1990). "Naturally Intelligent Systems." The MIT Press, Cambridge, Massachusetts.

Caudill, M. and Butler, C. (1992). "Understanding Neural Networks." The MIT Press, Cambridge, Massachusetts.

Cercone, N., and McCalla, G. (1986). Accessing Knowledge through Natural Language. *In* "Advances in Computers (M. C. Yovits, ed.), pp. 1-99. Academic Press, New York.

Charniak, E. (1993). "Statistical Language Learning." The MIT Press, Cambridge, Massachusetts.

Chin, D. N., (1983). A Case Study of Knowledge Representation in UC. *In* "Proceedings of the Eighth International Conference on Artificial Intelligence," IJCAI, Karlsruhe, Germany, pp. 388-390.

Chinchor, N., and Sundheim, B. (1993). MUC-5 Evaluation Metrics. *In* "Proceedings of Fifth Message Understanding Conference," Morgan Kaufmann, San Francisco, California, pp. 69-78.

Chomsky, N. (1956). Three Models for the Description of Languages. *IRE Transactions on Information Theory* 2(3), 113-124.

Chomsky, N. (1957). "Syntactic Structures." Mouton, The Hague, Holand.

Chomsky, N. (1959). On Certain Formal Properties of Grammars. *Information and Control* 2(2), 137-167.

Chomsky, N. (1965). "Aspects of the Theory of Syntax." The MIT Press, Cambridge, Massachusetts.

Church, K. W., and Mercer, R. L. (1993). Introduction to the Special Issue on Computational Linguistics Using Large Corpora. *In* "Using Large Corpora," (S. Armstrong, ed.), pp. 1-24. The MIT Press, Cambridge, Massachusetts.

Church, K. (1985). Stress Assignment in Letter to Sound Rules for Speech Synthesis. *In* "Proceedings of the 23rd Annual Meeting of ACL," Morgan Kaufmann, San Francisco, California, pp. 136-143. Cited in Marcus, 1994.

Church, K. W., and Rau, L. F. (1995). Commercial Applications of Natural Language Processing. *Commun. ACM* 38(11), 71-79.

Clark, A. (1993). "Associative Engines – Connectionism, Concepts, and Representational Change." The MIT Press, Cambridge, Massachusetts.

Cohen, P. R., and Oviatt, S. L. (1994). The Role of Voice in Human-Machine Communication. In "Voice Communication Between Humans and Machines" (D. B. Roe, and J. G. Wilpon, eds.), pp. 34-75. National Academy of Sciences, Washington, District of Columbia.

Cohen, P., Johnston, M., McGee, D., Oviatt, S., Pittman, J., Smith, I., Chen, L., and Clow, J. (1997). QuickSet: Multimodal Integration for Simulation Set-up and Control. *In* "Proceedings of Fifth

Conference on Applied Natural Language Processing," Association for Computational Linguistics, Morgan Kaufmann, San Francisco, California, pp. 20-24.

Cowan J. D. and Sharp, D. H. (1988). Neural Nets and Artificial Intelligence. *In* "The Artificial Intelligence Debate – False Starts, Real Foundations," (S. R. Graubard, ed.), pp. 85-121.

Crangle, C., and Suppes, P. (1994). "Language and Learning for Robots." Center for the Study of Language and Information, Stanford, California.

Day, M. C., and Boyce, S. J. (1993). Human Factors in Human-Computer System Design. *In* "Advances in Computers (M. C. Yovits, ed.), pp. 333-430. Academic Press, New York.

Digalakis, V., and Murveit, H. (1994). Genomes: Optimizing the Degree of Mixture Tying in Large Vocabulary Hidden Markov Model Based Speech Recognizer. *In* "Proceedings of International Conference on Acoustics, Speech, and Signal Processing," IEEE Press, Piscataway, New Jersey, pp. 1,537-1,540.

Earley, J. An Efficient Context-Free Parsing Algorithm. *Commun. ACM* 13(2), pp. 94-102. Reprinted in (Grosz, *et al.* 1986).

Elman, J. L. (1991). Distributed Representations, Simple Recurrent Networks, and Grammatical Structure. *Machine Learning* 7, 195-225.

Epstein, R. (1992). The Quest for the Thinking Computer. *AI Magazine* 13(2), 80-95.

Fatehchand, R. (1960). Machine Recognition of Spoken Words. *In* "Advances in Computers" (F. L. Alt ed.), pp. 193-229. Academic Press, New York.

Feigenbaum, E. A. (1996). How the "What" Becomes the "How" – ACM Turing Award Lecture. *Commun. ACM* 39(5), 97–104.

Firebaugh, M. W. (1988). "Artificial Intelligence – A Knowledge-Based Approach." PWS-Kent, Boston, Massachusetts.

Flanagan, J. L. (1994). Speech Communication – An Overview. In "Voice Communication Between Humans and Machines" (D. B. Roe, and J. G. Wilpon, eds.), pp. 76-104. National Academy of Sciences, Washington, District of Columbia.

Fodor, J. A., and Pylyshyn, Z. W. (1998). Connectionism and Cognitive Architecture: A Critical Analysis. *Cognition* 28, 3-71.

Furnas, G. W., Landauer, T. K., Gomez, L. M., and Dumais, S. T. (1983). Statistical Semantics: Analysis of the Potential Performance of Key-Word Information Systems. *The Bell System Technical Journal* 62(6), 1753-1806.

Furnas, G. W., Landauer, T. K., Gomez, L. M., and Dumais, S. T. (1987). The Vocabulary Problem in Human-System Communication. *Commun. ACM* 30(11), 964-971.

Furui, S. (1994). Toward the Ultimate Synthesis / Recognition System. In "Voice Communication Between Humans and Machines" (D. B. Roe, and J. G. Wilpon, eds.), pp. 450-466. National Academy of Sciences, Washington, District of Columbia.

Garvin, P. L., and Spolsky, B., eds. (1966). "Computation in Linguistics." Indiana University Press, Bloomington, Indiana.

Gazdar, G., and Mellish, C. (1989). "Natural Language Processing in LISP – An Introduction to Computational Linguistics." Addison-Wesley, Reading, Massachusetts.

Gazdar, G., Franz, A., Osborne, K., and Evans, R. (1987). "Natural Language Processing in the 1980s." Center for the Study of Language and Information, Stanford, California.

Giachin, E. P. (1992). Automatic Training of Stochastic Finite-State Language Models for Speech Understanding. *In* "Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing." Cited in (Harper, *et al.*, 1994).

Giachin, E., and McGlashan, S. (1997). Spoken Language Dialogue Systems. In "Corpus-Based Methods in Language and Speech Processing" (S. Young, and G. Bloothooft, eds.), pp. 69-117. Kluwer Academic Publishers, Dordrecht, The Netherlands.

Grosz, B. J. (1987). TEAM: An Experiment in the Design of Transportable Natural-Language Interfaces. *Artificial Intelligence* 32, 173-243.

Grosz, B. J., Spark Jones, K., and Webber, B. L., eds. (1986). "Readings in Natural Language Processing." Morgan Kaufmann, Los Altos, California.

Hall, G., Popowich, F., and Fass, D. (1996). Natural Language Edit Controls: Constrained Natural Language Devices in User Interfaces. *In* "Proceedings of IEEE 8th International Conference on Tools with Artificial Intelligence," IEEE Computer Society Press, Los Alamitos, California, pp. 475-477.

Hall, P. A. V., and Dowling, G. R. (1980). Approximate String Matching. *Computing Surveys* 12(4), 381-402.

Harris, M. D. (1985). "Introduction to Natural Language Processing." Reston Publishing Company, Reston, Virginia.

Harper, M.P., Jamieson, L. H., Mitchell, C. D., Ying, G., Potisuk, S., Shrinivasan, P. N., Chen, R., Zoltowski, C. B., McPheters, L., L., Pellom, B., and Helzerman, R. A. (1994). Integrating Language Models with Speech Recognition. *In* "Proceedings of AAAI-94 Workshop on Integration of Natural Language and Speech Processing," (P. McKevitt, ed.), pp. 139-146, Seattle, Washington.

Hayes-Roth, F., and Jacobstein, N. The State of Knowledge-Based Systems. *Commun. ACM* 37(3), 27-39.

Hebb, D. O. (1949). "The Organization of Behavior." John Wiley, New York.

Hemphill, C. (1993). DAGGER, Directed Acyclic Graphs of Grammars for Enhanced Recognition. User's Guide and Reference Manual, Texas Instruments, Dallas, Texas.

Hendrix, G. G., Sacerdoti, E. D., Sagalowicz, D., and Slocum, J. (1978). Developing A Natural Language Interface to Complex Data. *ACM Transactions on Database Systems* 3(2), 105-147.

Herdan, G. (1964). "Quantitative Linguistics." Butterworths, Washington, District of Columbia.

Hill, D. R. (1971). Man–Machine Interaction Using Speech. *In* "Advances in Computers (F. L. Alt, and M. Rubinoff, eds.), pp. 165-230. Academic Press, New York.

Hirschman, L. (1994). The Roles of Language Processing in a Spoken Language Interface. In "Voice Communication Between Humans and Machines" (D. B. Roe, and J. G. Wilpon, eds.), pp. 217-237. National Academy of Sciences, Washington, District of Columbia.

Hirschman, L., and Cuomo, D. (1994). Report from the ARPA Workshop on Evaluation of Human Computer Interfaces. Tech. Rep. MP 94B0000259, MITRE, Bedford, Massachusetts.

Hofstadter, D. P. (1979). "Gödel, Escher, Bach: An Eternal Golden Braid." Random House, New York.

Hollan, J., Rich, E., Hill, W., Wroblewski, D., Wilner, W., Wittenburg, K., and Grudin, J. (1991). An Introduction to HITS: Human Interface Tool Suite. *In* "Intelligent User Interfaces," (J. W. Sullivan, and S. W. Tyler, eds.), pp. 293-337. ACM Press, New York.

Hovy, E. (1993). How MT Works. *Byte* 18(1), 167-176.

Jackendoff, R. (1990). "Semantic Structures." The MIT Press, Cambridge, Massachusetts.

Jacobs, P. S. (1994). Text-Based Systems and Information Management: Artificial Intelligence Confronts Matters of Scale. *In* "Proceedings of Sixth International Conference on Tools with Artificial Intelligence," IEEE Computer Society Press, Los Alamitos, California, pp. 235-236.

Josselson, H. H. (1971). Automatic Translation of Languages Since 1960: A Linguist's View. *In* "Advances in Computers (F. L. Alt, and M. Rubinoff, eds.), pp. 1-58. Academic Press, New York.

Kamm, C. (1994). User Interfaces for Voice Applications. In "Voice Communication Between Humans and Machines" (D. B. Roe, and J. G. Wilpon, eds.), pp. 422-442. National Academy of Sciences, Washington, District of Columbia.

Kay, M., Gawron, J. M., and Norvig, P. (1994). "Verbmobil: A Translation System for Face-to-Face Dialog." Center for the Study of Language and Information, Stanford, California.

King, M. (1996). Evaluating Natural Language Processing Systems. *Commun. ACM* 39(1), 73-79.

Knill, K., and Young, S. (1997). Hidden Markov Models in Speech and Language Processing. In "Corpus-Based Methods in Language and Speech Processing" (S. Young, and G. Bloothooft, eds.), pp. 27-68. Kluwer Academic Publishers, Dordrecht, The Netherlands.

Koons, D. B., Sparrell, C. J., and Thorisson, K. R. (1993). Integrating Simultaneous Input from Speech, Gaze, and Hand Gestures. *In* "Intelligent Multimedia Interfaces" (M. T. Maybury, ed.), pp. 257-276. The MIT Press, Cambridge, Massachusetts.

Krause, J. (1993). A Multilayered Empirical Approach to Multimodality: Towards Mixed Solutions of Natural Language and Graphical Interfaces. *In* "Intelligent Multimedia Interfaces" (M. T. Maybury, ed.), pp. 328-352. The MIT Press, Cambridge, Massachusetts.

Kuno, S., and Oettinger, A. G. (1963). Multiple-Path Syntactic Analyser. *In* "Information Processing" 62, (C. M. Popplewell, ed.), pp. 306-312. North-Holland, Amsterdam. Reprinted in (Grosz, *et al.* 1986).

Kupiec, J. (1992). Hidden Markov Estimation for Unrestricted Stochastic Context-Free Grammars. *In* "IEEE International Conference on Acoustics, Speech, and Signal Processing," pp. 177–180. Cited in (Harper *et al.*, 1994).

Landauer, T. K., Galotti, K. M., and Hartwell, S. (1983). Natural Command Names and Initial Learning: A Study of Text-Editing Terms. *Commun. ACM* 26(7), 495-503.

Lane, A. (1987), DOS in English. *Byte* 12(12), 261-264.

Lari, K., and Young, S. J. (1991). Applications of Stochastic Context-Free Grammars Using the Inside-Outside Algorithm. *Computer Speech & Language* 5(3), 237–257. Cited in (Harper *et al.*, 1994).

Ledgard, H., Whiteside, J. A., Singer, A., and Seymour, W. (1980). The Natural Language of Interactive Systems. *Commun. ACM* 23(10), 556-563.

Lee, K-F. (1993)."Automatic Speech Recognition. *In* "The Distinguished Lecture Series VI," Video. University Video Communications, Stanford, California.

Levinson, S. E. (1994). Speech Recognition Technology: A Critique. In "Voice Communication Between Humans and Machines" (D. B. Roe, and J. G. Wilpon, eds.), pp. 159-164. National Academy of Sciences, Washington, District of Columbia.

Levitt, H. (1994). Speech Processing for Physical and Sensory Disabilities. In "Voice Communication Between Humans and Machines" (D. B. Roe, and J. G. Wilpon, eds.), pp. 311-343. National Academy of Sciences, Washington, District of Columbia.

Lewis H. R., and Papadimitriou, C. H. (1981). "Elements of the Theory of Computation." Prentice-Hall, Englewood Cliffs, New Jersey.

Locke, W. N., and Booth, A. D. eds. (1955). "Machine Translation of Languages." Technology Press of MIT and Wiley, Cambridge, Massachusetts.

Luger, G. F., and Stubblefield, W. A. (1993). "Artificial Intelligence: Structures and Strategies for Complex Problem Solving." Benjamin/Cummings, Redwood City, California.

Maegaard, B., and Perschke, S. (1991). An Introduction to the Eurotra Programme. *In* "The Eurotra Linguistic Specifications" (C. Copeland, J. Durand, S. Krauwer, and B. Maegaard, eds.), pp. 7-14. Office of Official Publications of the European Communities, Brussels, Luxembourg.

Makhoul, J., and Schwartz, R. (1994). State of the Art in Continuous Speech Recognition. In "Voice Communication Between Humans and Machines" (D. B. Roe, and J. G. Wilpon, eds.), pp. 165-198. National Academy of Sciences, Washington, District of Columbia.

Malmkjær, K., ed. (1991). "The Linguistics Encyclopedia." Routledge, New York.

Manaris, B., and Dominick, W. (1993). NALIGE: a User Interface Management System for the Development of Natural Language Interfaces. *International Journal of Man-Machine Studies* 38(6), 891-921.

Manaris, B. Z. (1994). An Engineering Environment for Natural Language Interfaces to Interactive Computer Systems. *International Journal of Artificial Intelligence Tools* 3(4), 557-579.

Manaris, B. Z., and Slator, B. M. (1996). Interactive Natural Language Processing: Building on Success. *IEEE Computer* 29(7), 28-32.

Manaris, B., and Harkreader, A. (1997). SUITE: Speech Understanding Interface Tools and Environments. *In* "Proceedings of the Tenth International Florida Artificial Intelligence Symposium," Florida AI Research Society, Daytona Beach, Florida, pp. 247-252.

Marcus, M. (1978). A Computational Account of Some Constraints on Language. *In* "Theoretical Issues in Natural Language Processing-2," (D. Waltz, ed.), pp. 236-246. Association of Computational Linguistics, Urbana-Champaign, Illinois.

Marcus, M. P. (1980). "A Theory of Syntactic Recognition for Natural Language." The MIT Press, Cambridge, Massachusetts.

Marcus, M. (1994). New Trends in Natural Language Processing: Statistical Natural Language Processing. *In* "Voice Communication Between Humans and Machines" (D. B. Roe, and J. G. Wilpon, eds.), pp. 482-504. National Academy of Sciences, Washington, District of Columbia.

Markowitz, J. A. (1996). "Using Speech Recognition." Prentice Hall PTR, Upper Saddle River, New Jersey.

Martin, P., Crabbe, F., Adams, S., Baatz, E., and Yankelovich, N. (1996). SpeechActs: A Spoken Language Framework. *IEEE Computer* 29(7), 33-40.

McCorduck, P. (1979). "Machines Who Think." W. H. Freeman, San Francisco, California.

McCulloch W. S., and Pitts, W. H. (1943). A Logical Calculus of the Ideas Immanent in Nervous Activity. *Bulletin of Mathematical Biophysics* 5, 115-137.

Meisel, W. S. (1993). Talk to Your Computer. *Byte* 18(10), 113-120.

Miikkulainen, R. (1993). "Subsymbolic Natural Language Processing: An Integrated Model of Scripts, Lexicon, and Memory." The MIT Press, Cambridge, Massachusetts.

Miikkulainen, R. (1994). Integrated Connectionist Models: Building AI Systems on Subsymbolic Foundations. *In* "Proceedings of IEEE 6th International Conference on Tools with Artificial Intelligence," IEEE Computer Society Press, Los Alamitos, California, pp. 231-232.

Miller, L. C. (1993). Resource Guide: Machine-Translation Software. *Byte* 18(1), 185-186.

Minsky, M., and Papert, S. (1969). "Perceptrons: An Introduction to Computational Geometry." The MIT Press, Cambridge, Massachusetts. Cited in (Cowan and Sharp, 1988).

Moll, R. N., Arbib, M. A., and Kfoury, A. J. (1988). "An Introduction to Formal Language Theory." Springer-Verlag, New York.

Moore, J. D., and Mittal, V. O. (1996). Dynamically Generated Follow-up Questions. *IEEE Computer* 29(7), 75-86.

Moore, R. C. (1994a). Integration of Speech with Natural Language Understanding. In "Voice Communication Between Humans and Machines" (D. B. Roe, and J. G. Wilpon, eds.), pp. 254-271. National Academy of Sciences, Washington, District of Columbia.

Moore, R. C. (1994b). Semantic Evaluation for Spoken-Language Systems. *In* "Proceedings of the Human Language Technology Workshop," Morgan Kaufmann, San Francisco, California, pp. 126-131.

Moore, R., Dowding, J., Bratt, H., Gawron, J. M., Gorfu, Y., and Cheyer, A. (1997). CommandTalk: A Spoken-Language Interface for Battlefield Simulations. *In* "Proceedings of Fifth Conference on Applied Natural Language Processing," Association for Computational Linguistics, Morgan Kaufmann, San Francisco, California, pp. 1-9.

Mostow, J., Roth, S. F., Hauptmann, A. G., Kane, M. (1994). A Prototype Reading Coach that Listens. *In* "Proceedings of Twelfth National Conference on Artificial Intelligence," AAAI Press, Palo Alto, California, pp. 785–792.

Munakata, T. (1994). Commercial and Industrial AI. *Commun. ACM* 37(3), 23-25.

Muller, M. J., Wharton, C., McIver, Jr., W. J., Laux, L. (1997). Toward an HCI Research and Practice Agenda Based on Human Needs and Social Responsibility. *In* "Proceedings CHI'97 – Human Factors in Computing Systems," ACM Press, Atlanta, Georgia, pp. 155-161

Napier, H. A., Lane, D. M., Batsell, R. R., and Guada, N. S. (1989) Impact of a Restricted Natural Language Interface on Ease of Learning and Productivity. *Commun. ACM* 32(10), 1190-1198.

Newel, A., and Simon, H. (1976). Computer Science as Empirical Inquiry: Symbols and Search. *Commun. ACM* 19(3), 113-126.

Newmeyer, F. J. (1983). "Grammatical Theory – Its Limits and Its Possibilities." The University of Chicago Press, Chicago, Illinois.

Ney, H. (1991). Dynamic Programming Parsing for Context-Free Grammars in Continuous Speech Recognition. *IEEE Transactions on Signal Processing* 39(2). Cited in (Harper *et al.*, 1994).

Ney, H. (1997). Corpus-Based Statistical Methods in Speech and Language Processing. *In* "Corpus-Based Methods in Language and Speech Processing" (S. Young, and G. Bloothooft, eds.), pp. 1-26. Kluwer Academic Publishers, Dordrecht, The Netherlands.

Obermeier, K. K., (1988). "Natural Language Processing Technologies in Artificial Intelligence – The Science and Industry Perspective." John Wiley & Sons, New York.

Oettinger, A. G. (1955). The Design of an Automatic Russian-English Technical Dictionary. *In* "Machine Translation of Languages," (W. N. Locke and A. D. Booth, eds.), pp.47-65. The Technology Press of MIT and John Wiley, New York.

Otten, K. W. (1971). Approaches to the Machine Recognition of Conversational Speech. *In* "Advances in Computers (F. L. Alt, and M. Rubinoff, eds.), pp. 127-163. Academic Press, New York.

Oviatt, S., and Cohen, P. R. (1991). The Contributing Influence of Speech and Interaction on Human Discourse Patterns. *In* "Intelligent User Interfaces," (J. W. Sullivan, and S. W. Tyler, eds.), pp. 69-83. ACM Press, New York.

Oviatt, S. DeAngeli, A., and Kuhn, K. (1997). Integration and Synchronization of Input Modes during Multimodal Human-Computer Interaction. *In* "Proceedings CHI'97 – Human Factors in Computing Systems," ACM Press, Atlanta, Georgia, pp. 415-422.

Pallett, D. S., Fiscus, J. G., Fisher, W. M., Garofolo, J. S., Lund, B. A., Martin, A., and Przybocki, M. A. (1994). 1994 Benchmark Tests for the ARPA Spoken Language Program. *In* "Proceedings of the Spoken Language Systems Technology Workshop," Morgan Kaufmann, San Francisco, California, pp. 5-36.

Paris, C., and Vander Linden, K. (1996). An Interactive Support Tool for Writing Multilingual Manuals. *IEEE Computer* 29(7), 49-56.

Pereira, C. N., and Grosz, B. J., eds. (1994). "Natural Language Processing." The MIT Press, Cambridge, Massachusetts. Reprinted from *Artificial Intelligence* 63(1-2), 1993.

Preece, J., Rogers, Y., Sharp, H., Benyon, D., Holland, S., and Carey, T. "Human-Computer Interaction." Addison Wesley, Reading, Massachusetts.

Rabiner, L., and Juang, B-H. (1993). "Fundamentals of Speech Recognition." Prentice Hall PTR, Englewood Cliffs, New Jersey.

Rash, W. (1994). Talk Show. *PC Magazine*, Dec. 20, 203-219.

Reddy, R. (1996) To Dream the Possible Dream – ACM Turing Award Lecture. *Commun. ACM* 39(5), 105–112.

Reeker, L. H. (1976). The Computational Study of Language Acquisition. *In* "Advances in Computers" (M. Rubinoff and M. C. Yovits, eds.), pp. 181-237. Academic Press, New York.

Reich, P. A. (1969). The Finiteness of Natural Language. *Language* 45(4), 831-843.

Revzin, I. I. (1966). "Models of Language." Methuen & Co Ltd, London, England.

Ristad, E. S. (1993). "The Language Complexity Game." The MIT Press, Cambridge, Massachusetts.

Roe, D. B. (1994). Deployment of Human-Machine Dialogue Systems. In "Voice Communication Between Humans and Machines" (D. B. Roe, and J. G. Wilpon, eds.), pp. 373-389. National Academy of Sciences, Washington, District of Columbia.

Rosenblatt, F. (1958). The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain. *Psychological Review* 65, 386-408.

Rudnicky, A. I., Hauptmann, A. G., Lee, K.-F. (1994). Survey of Current Speech Technology. *Commun. ACM* 37(3), 52-57.

Rumelhart, D. E. (1975). Notes on a Schema for Stories. *In* "Representation and Understanding" (D. Bobrow and A. Collins, eds.) Academic Press, New York. Cited in (Hofstadter, 1979, p. 675).

Russell, S., and Norvig, P. (1995). "Artificial Intelligence – A Modern Approach." Prentice Hall, Englewood Cliffs, New Jersey.

Sager, N. (1967). Syntactic Analysis of Natural Language. *In* "Advances in Computers" (F. L. Alt, and M. Rubinoff, eds.), pp. 153-188. Academic Press, New York.

Sager, N. (1978). Natural Language Information Formatting: The Automatic Conversion of Texts to a Structured Data Base. *In* "Advances in Computers" (M. C. Yovits, ed.), pp. 89-162. Academic Press, New York.

Schafer, R. W. (1994). Scientific Bases of Human-Machine Communication by Voice. *In* "Voice Communication Between Humans and Machines" (D. B. Roe, and J. G. Wilpon, eds.), pp. 15-33. National Academy of Sciences, Washington, District of Columbia.

Schalkwyk, J., Vermeulen, P., Fanty, M., and Cole, R. (1995). Embedded Implementation of a Hybrid Neural-Network Telephone Speech Recognition System. *In* "Proceedings of IEEE International Conference on Neural Networks and Signal Processing," IEEE, Nanjing, China, pp. 800-803.

Schalkwyk, J., and Fanty, M. (1996). The CSLU-C Toolkit for Automatic Speech Recognition. Center for Spoken Language Understanding, Oregon Graduate Institute of Science and Technology. http://www.cse.ogi.edu/CSLU/toolkit/documentation/csluc/Cdoc.html .

Schmucker, K. J. (1984). "Fuzzy Sets, Natural Language Computations, and Risk Analysis." Computer Science Press, Rockville, Maryland.

Scott, J.C. (1996). The Voices of Automation. *Computer Shopper* 16(9), 550-555.

Shannon, C. (1948). The Mathematical Theory of Communication. *Bell Systems Technical Journal* 27, 398-403. Cited in (Church and Mercer, 1993).

Shapiro, S. C. (1979). The SNePS semantic network processing system. In "Associative Networks: Representation and Use of Knowledge by Computers," (N. V. Findler, ed.), pp. 179-203. Academic Press, New York.

Sheremetyeva, S., and Nirenburg, S., (1996). Knowledge Elicitation for Authoring Patent Claims. *IEEE Computer* 29(7), 57-63.

Shneiderman, B. (1993). "Designing the User Interface: Strategies for Effective Human-Computer Interaction," 2nd ed. Addison-Wesley, Reading, Massachusetts.

Slator, B. M, Anderson, M. P., and Conley, W. (1986). Pygmalion at the Interface. *Commun. ACM* 29(7), 599-604.

Slocum, J. (1986). Machine Translation: An American Perspective. *Proceedings of IEEE* 74(7), 969-978.

Smith, G., and Bates, M. (1993). Voice Activated Automated Telephone Call Routing. *In* "Proceedings of Ninth IEEE Conference on Artificial Intelligence for Applications," IEEE CS Press, Los Alamitos, California, pp. 143-148.

Sowa, J. F. (1984). "Conceptual Structures – Information Processing in the Mind and Machine." Addison Wesley, Reading, Massachusetts.

Spark Jones, K. (1994). Towards Better NLP System Evaluation. *In* "Proceedings of the Human Language Technology Workshop," Morgan Kaufmann, San Francisco, California, pp. 102-107.

Spark Jones, K., and Galliers, J. R. (1996). "Evaluating Natural Language Processing Systems – An Analysis and Review." Springer-Verlag, New York.

Stock, O., and the ALFRESCO Project Team (1993). ALFRESCO: Enjoying the Combination of Natural Language Processing and Hypermedia for Information Exploration. *In* "Intelligent Multimedia Interfaces" (M. T. Maybury, ed.), pp. 197-225. The MIT Press, Cambridge, Massachusetts.

Strong, G. (1996). Human Language Research and the National Science Foundation. *IEEE Computer* 29(7), p. 31.

Thompson, F. B., and Thompson, B. H. (1975). Practical Natural Language Processing: The REL System as Prototype. *In* "Advances in Computers" (M. Rubinoff and M. C. Yovits, eds.), pp. 109-168. Academic Press, New York.

Titus, J. P. (1967). The Nebulous Future of Machine Translation. *Commun. ACM* 10(3), 190. Cited in (Josselson, 1971, p. 48).

Turing, A. M. (1950). Computing Machinery and Intelligence. *Mind* (59), 433-460.

Wahlster, W. (1991). User and Discourse Models for Multimodal Communication. *In* "Intelligent User Interfaces," (J. W. Sullivan, and S. W. Tyler, eds.), pp. 45-68. ACM Press, New York.

Waibel, A. (1996). Interactive Translation of Conversational Speech. *IEEE Computer* 29(7), 41-48.

Wauchope, K., Everett, S., Perzanowski, D., and Marsh, E. (1997). Natural Language in Four Spatial Interfaces. *In* "Proceedings of Fifth Conference on Applied Natural Language Processing," Association for Computational Linguistics, Morgan Kaufmann, San Francisco, California, pp. 8-11.

Weaver, W. (1955). Translation. *In* "Machine Translation of Languages," (W. N. Locke and A. D. Booth, eds.), pp. 15-23. The Technology Press of MIT and John Wiley, New York.

Weiser, M. (1994). The World is Not a Desktop. *Interactions* 1(1), 7-8.

Weizenbaum, J. (1966). ELIZA – A Computer Program for the Study of Natural Language Communication Between Man and Machine. *Commun. ACM* 9(7), 36-43.

Weizenbaum, J. (1967). Contextual Understanding by Computers. *Commun. ACM* 10(8), 474-480.

Weizenbaum, J. (1976). "Computer Power and Human Reason – From Judgment to Calculation." W. H. Freeman, San Francisco.

Wermter, S., and Weber, V. (1996). Interactive Spoken-Language Processing in a Hybrid Connectionist System. *IEEE Computer* 29(7), 65-74.

Wilensky, R., Arens, Y., and Chin, D. (1984). Talking to UNIX in English: An Overview of UC. *Commun. ACM* 27(6), 574-593.

Wilensky, R., Chin, D. N., Luria, M., Martin, J., Mayfield, J., and Wu, D. (1988). The Berkeley UNIX Consultant Project. *Computational Linguistics* 14(4), 35-84.

Wilks, Y. (1975). An Intelligent Analyzer and Understander of English. *Commun. ACM* 18(5), 264-274 (reprinted in Grosz, *et al.*, 1986).

Wilks, Y. (1996). Natural Language Processing – Introduction. *Commun. ACM* 39(1), 60-62.

Wilks, Y., Slator, B. M., and Guthrie, L. M. (1996). "Electric Words – Dictionaries, Computers, and Meaning." The MIT Press, Cambridge, Massachusetts.

Wilpon, J. G. (1994). Applications of Voice-Processing Technology in Telecommunications. In "Voice Communication Between Humans and Machines" (D. B. Roe, and J. G. Wilpon, eds.), pp. 280-310. National Academy of Sciences, Washington, District of Columbia.

Winograd, T. (1972). "Understanding Natural Language." Academic Press, New York.

Winograd, T. (1980). What Does It Mean to Understand Language?. *Cognitive Science* 4, 209-241.

Winograd, T. (1983). "Language as a Cognitive Process." Addison-Wesley, Reading, Massachusetts.

Winograd, T. and Flores, F. (1986). "Understanding Computers and Cognition." Ablex Publishing, Norwood, New Jersey.

Winston, P. H. (1992). "Artificial Intelligence," 3rd ed. Addison-Wesley, Reading, Massachusetts.

Woods, W. A. (1973). Progress in Natural Language Understanding: An Application to Lunar Geology. *In* "AFIPS Conference Proceedings" 42, 441-450.

Woods, W. A. (1978). Semantics and Quantification in Natural Language Question Answering. *In* "Advances in Computers" (M. C. Yovits, ed.), pp. 1-87. Academic Press, New York.

Zipf, G. K. (1949). "Human Behaviour and The Principle of Least Effort." Addison-Wesley, Cambridge, Massachusetts.

Zue, V. W. (1994). Toward Systems that Understand Spoken Language. *IEEE Expert* 9(1), 51-59.