

Natural Language Interface for the Multi-Agent COTR System (MACS)

Victoria Yoon^a, Teresa Wilson^a, Stuart Lowry^b, Bonnie Rubenstein-Montano^a, Jay
Liebowitz^a

^a Department of Information Systems,
University of Maryland Baltimore County
1000 Hilltop Circle,
Baltimore, MD 25250
yoon@umbc.edu

^b Science Applications International Corporation, Annapolis, MD, USA

ABSTRACT

This paper presents a Web-Based, Multi-Agent COTR System (MACS) which provides the users with a natural language interface. MACS was designed to provide advice in the pre-award phase of a defense contract. The original version of MACS allowed the users to query using only a list of predetermined keywords using pull-down menus. This study has extended the capability of MACS by allowing the users to interface with the system using natural language. The paper presents the overview of natural language processing (NLP), the newly extended architecture of MACS, and the approaches used in developing a natural language capability.

Keywords: Intelligent Agent, Multi-agent System, Natural Language Processing, Natural Language Agent

1. INTRODUCTION

Intelligent agents have successfully demonstrated their potential for solving a variety of complex problems and in turn have become a viable alternative for tackling many real problems [Shen and Norrie, 1999; Muscettola, Nayak, Williams, and Pell, 1998]. As a result, the intelligent agent technique has drawn much attention from many academicians as well as practitioners. Intelligent agents are software systems that perform tasks delegated by human users with some degree of autonomy and independence. A Multi-Agent System (MAS) consists of multiple, interacting agents, which share knowledge and tasks among themselves, and cooperate with each other to achieve the goals set by the human users. The capabilities of a MAS come not only from the intelligence of each individual agent, but also from the emergent behavior of an entire agent community.

One important aspect in MAS is the user interface, which is responsible for interacting with the users to receive instructions and to provide the results of its actions. In order to support and provide assistance to a user to interact with a MAS, a user agent, which is designed to interface with the users, often builds a user model by observing and monitoring the actions taken by the user. The user agent can learn to improve its performance through receiving positive and negative feedback from the user [Maes, 1994]. However, the user agents in a MAS have suffered from limited flexibility when interfacing with the users. The user agent often responds only to direct command manipulation, forcing users to memorize a set of commands and the proper sequence of issuing commands. A natural language interface with the user would eliminate the necessity to remember all commands and provide much flexibility in terms of user

interactions. Users can input a string of words or queries and the natural language processing agent can parse out possible commands from the natural statements.

MACS (Multi-Agent COTR {Contracting Officer's Technical Representative} System) was developed to provide advice in the pre-award phase of a contract [Liebowitz et al., 2000b]. The original version of MACS allowed the user to query using only a list of predetermined keywords from pull-down menus. The user agent, after processing the query, broadcasts the query request to all specialty agents. The appropriate specialty agent will then provide the user agent with the requested information, and the user agent in turn provides that information to the user. The inflexible user interface could cause problems for those users who do not know the right keywords. In order to increase the flexibility in terms of user interaction, this study integrates natural language processing into the current MACS architecture.

The objective of this paper is to present a Web-Based, Multi-Agent COTR System (MACS) incorporating natural language processing. The next sections describe the background for MACS system and the overview of the natural language process. Followed is the detailed description on the NLP approach to MACS, including the building tool used for MACS – Open Agent Architecture (OAA), the architecture of MACS, and the NLP agent – the Definite Clause Grammar-Natural Language (DCG-NL). The paper then presents the conclusions and future research directions.

2. BACKGROUND

In the Acquisition 2005 Task Force Final Report, “Shaping the Civilian Acquisition Workforce of the Future,” [USD AT&L, 2000] it indicates that the

Department of Defense (DOD) is facing a critical crisis that can dramatically affect the United States' ability to provide war fighters with modern weapon systems needed to defend the national interest of the United States. The report indicates that after 11 consecutive years of downsizing, the DOD faces serious imbalances in the skills and experience of our highly talented and specialized civilian workforce. Fifty percent will be eligible to retire by 2005, and in some occupations, half of the current employees will be gone by 2006. Unless immediately addressed, this situation will leave many acquisition organizations without the talent, leadership, and diversity needed to succeed in the new century [USD AT&L, 2000].

To help resolve this ensuing problem, various acquisition programs have been created to improve the state-of-the-art in acquisition research so that technology and business innovations can aid eventual human resource limitations in the acquisition field. One such program managed by the Naval Postgraduate School is the External Acquisition Research Program (EARP). The External Acquisition Research Program (EARP) has been established and funded to promote advances in basic and applied research as related to US defense acquisition, and further build and nurture the community of researchers working in this area. The current EARP funded projects address:

- A Web-Based, Multi-Agent COTR System (MACS) incorporating learning and natural language processing;
- A Web-Enabled Automated Acquisition Design
- Models and Tools for Acquisition Knowledge Management
- Development of a Methodology for Redesigning Acquisition Processes Based on "Information Load Analysis"
- Software Product Line Architectures with a Research Infrastructure for Software Systems Acquisition
- Contents Management Issues in Business-in-Business Procurement

- Acquisition Reform in the Air Force Materiel Command
- Design for Effective Horizontal Knowledge Transfer in Acquisition Related New Product Development Ventures
- Identifying Structural Dimensions of Organizational Form for Acquisition Portals
- Study of World Class Suppliers: Their Characteristics and Role in the Procurement Process
- Synthetic Environments for Defense Acquisition Simulations

One of the projects that has been funded under EARP over the past two years is called MACS (Multi-Agent COTR {Contracting Officer's Technical Representative} System) [Liebowitz et al., 2000b]. The purpose of MACS is to design and develop a web-based, multi-agent system that can provide advice in the pre-award phase of a contract. MACS is originally developed from a rule-based expert system called CESA (COTR Expert System Aid) [Liebowitz, 2000a]. As such, MACS covers five main areas:

- The type of contract desired;
- What forms are needed in the procurement request (PR) package;
- What type of synopsis is required;
- How should incoming proposals be evaluated;
- What is needed to go sole source (i.e., where only one vendor has the unique qualifications to accomplish the task).

The eventual hope of MACS would be to replace part of the “Ask a Professor” module, contained within the Defense Acquisition Deskbook (DAD), whereby pre-award related questions in the five previous areas would be answered by agents in MACS instead of always sending the user queries to human experts via “Ask a Professor.” If MACS did not have the knowledge to provide an appropriate response to the user's question, then it could be forwarded to the appropriate human expert.

The first version of MACS, funded under EARP 99, had successfully demonstrated a web-based, multi-agent system in the five aforementioned specialty areas. MACS was developed through using Reticular Systems' AgentBuilder™, and applying Java servlets to facilitate the interaction between the user agent and AgentBuilder™. This version of MACS used keyword matching to broadcast the user query to all five specialty agents for arriving at some conclusion to the user's question. An exact string of keywords had to be used to fire the appropriate rule(s) in the specialty agents. This interface presented some severe constraints as it was awkward to use direct keyword matching instead of free-form text.

This led to the focus of the EARP 2000 research effort. To improve the user interactions with MACS, MACS now includes a limited natural language front-end, as well as a Bayesian learning approach [Rubenstein-Montano, et al, 2000] for targeting the best specialty agent who can respond to the user's question. Additionally, MACS has been encoded in OAA, developed by the Artificial Intelligence Laboratory of the Stanford Research Institute (SRI), for improved functionality.

The rest of this paper will focus on the implementation of MACS in OAA and the natural language processing component.

3. OVERVIEW OF NATURAL LANGUAGE PROCESSING

Although Natural Language Processing (NLP) seems to have as many definitions as there are researchers, a basic, well-accepted definition is that NLP is a field in computer science and artificial intelligence (AI) that refers to any technique that

processes natural language. However, NLP extends into linguistics, cognitive science, psychology and philosophy. For instances, psychology of language and psycholinguistics address how people identify structure of sentences and learn meanings of words, philosophy studies the meaning of language, how words and sentences acquire it, and how words identify objects in the world, and linguistics looks at how words form phrases and sentences and what constrains possible meanings for a sentence [Allen, 1995]. There is even a broad study in the field of mathematics, growing simultaneously within disparate branches of mathematics such as group theory and probability-- called combinatorics on words -- which looks at letters as a finite sequence of symbols; the systematic study of words seems to have been going on since the turn of the 20th century and has grown into an independent theory which found substantial applications in computer science, automata theory and formal languages, and linguistics. [Lotharie, 1997].

NLP history spans five decades, beginning with the research of noted MIT linguist Noam Chomsky in the 1950s and 1960s to current day work at the end of the 20th century. The major branches of study are Computer and Information Science and Computational Linguistics. The motivation behind research in NLP is to make computers capable of using natural language in order to increase our understanding of how human languages and minds work [Tennant, 1981]. The discipline of computer and information science treats NLP from a non-linguistic point of view, concentrating more on algorithm development, data structures, databases, knowledge representation and reasoning, machine learning, computation and statistical methods. A linguistically-motivated discipline, Computational Linguistics, the study of computer systems for understanding

and generating natural language, is concerned with developing procedures for handling a useful range of natural language input [Grishman, 1986]. Each perspective has its strengths and weaknesses; recent trends seem to indicate a more symbiotic relationship in an attempt to overcome limitations inherent in each discipline. The traditional applications of NLP are text and speech processing, but NLP is evident in expert systems, intelligent agents, and smart interfaces.

3.1. NLP Approaches

Various approaches have been used in NLP. The major categories are summarized below [Obermeier, 1989]:

1. **Linguistic:** This involves the process of determining the details of the structure of a sentence based on a grammar (syntax) and other features of the language itself, to include morphology, semantics, discourse, and pragmatics.
2. **Statistical:** This comprises all quantifiable approaches to automated language processing, to include probabilistic modeling, information theory and linear algebra. This approach refers to non-symbolic, non-logical theories and application and looks at language and cognition as probabilistic phenomena.
3. **Hybrid:** This is a more recent development, which combines both linguistic and statistical approaches. Each approach has its strength in certain areas of NLP and researchers have found that combining the best techniques proved most effective.
4. **AI:** This focuses on knowledge necessary to “understand” natural language. It uses “common sense” and world knowledge to try to develop models for language use and go beyond sentence processing, analyzing larger contexts. Research is oriented around cognitive aspects that deal with the full spectrum of human language use and consequent action.
5. **Connectionist:** This centers on neural networks, which was originally used in human brain research to form theories for human information processing based on the behavior of neurons and their connections. It uses a powerful metaphor based on biological mechanisms suggesting that information processing takes place by “spreading activation” of multiple processors, similar to the firing of neurons in the brain. Connections between processors are determined by attaching weights that either promote or inhibit activation of the processor.

3.2 NLP Applications

The main applications of NLP are concentrated in the following areas:

1. **Information Extraction (IE):** automatic extraction of information from text utilizing SGML tags to identify classes of words, including proper nouns and organization names, locations, time and numeric references.
2. **Information Retrieval (IR):** automatic retrieval of relevant documents from a collection of documents in response to a user query. Also includes new field of cross-language IR (CLIR).
3. **Machine Translation:** automatic translation of text in a source language into text in a target language (or languages).
4. **Text Summarization:** automatic summarization of documents utilizing extraction of phrases and sentences. Systems that create abstract summaries (more akin to human summaries) vice extract summaries (pulling sentences out of existing documents) are not yet available commercially.
5. **Speech Synthesis:** automatic rendering of speech into text, or text into speech, and from one source language to a target language (or languages) utilizing voice recognition techniques.
6. **NL generation:** automatic generation of text.
7. **NL interfaces:** front-end application of NLP techniques that simplifies and improves user communication.

3.3. NLP Processes and Components

A typical NLP system is composed of the following processes and components (in ascending order of difficulty):

<u>PROCESS</u>	<u>COMPONENTS</u>	<u>PURPOSE</u>
Phonological	Speech Recognizer	Identify speech sounds
Morphological	Morphological Analyzer	Identify words, including prefixes, suffixes and roots
Lexical	Lexicon, Dictionary Part of Speech Tagger Named Entity Tagger	Analyze words at the word level, retrieve information from lexicon
Syntactic	Grammar, Parser	Identify syntactic structure of input, label parts of speech

Semantic	Domain-specific and Domain-independent Knowledge Base Semantic/Case Grammar Semantic Constraints Synonym, Thesauri Sense Disambiguation Conceptual Net	Identify meanings of input based on context within sentence
Discourse	Discourse Parser Sense Disambiguation Anaphora Resolution Coreference Resolution	Identify meaning of input in based on context with other sentences
Pragmatic	Language Generator Language Models World Knowledge/Ontology	Generate grammatical and meaningful sentences

3.4 Natural Language (NL) Querying

Of the numerous large-scale Internet search engines, three major ones—Ask Jeeves, Northern Light, and Oingo, support NL querying. The reason there are so few, in simple terms, is that true NL querying is difficult to perform well. NLP is difficult when there is plenty of data to analyze from which to acquire information, much less if there is only one sentence, namely the query, to go on. Most systems that boast NLP perform NLP on the lower levels of understanding and often this is for query interpretation only [Feldman, 1999]. Examples of current techniques include automatic words stemming, automatic identification of proper nouns and other classes of words, automatic phrase identification and automatic concept identification [Liddy, 1998]. Recent advances in utilizing linguistic techniques for information retrieval (IR) is in sharp contrast to the conventional view of IR which had ignored linguistically-motivated solutions partly

because the field has been populated by computer scientists and statisticians who are unfamiliar with language and linguistics.

However, if the true purpose of NL querying is to allow users greater freedom in query formulation and the value of NLP is to transform ambiguous natural language queries and documents into unambiguous internal representation to retrieve relevant information, NLP techniques would have to be performed at the higher levels of the linguistic hierarchy, discourse and pragmatics, as well. NLP research has encountered much difficulty at the higher levels, and this is even without taking into consideration the issues unique to each application, such as IR, Machine Translation, or Text Summarization. Word sense disambiguation and polysemy (“bank” for money vs. “bank” of a river), anaphora and coreference resolution (“JULIE’S MOTHER went to the store. ELEANOR was a handsome woman. SHE bought some cheese and wine.”), spelling and pronunciation variations are but a few areas of linguistics that have hindered major advances in NLP. If systems can get closer to “understanding” the meaning of a query or a document, the retrieval process would indeed be true “information” retrieval rather than “document” retrieval, that is, the user would get only relevant information and thus have a better chance to address the data overload problem. For true “understanding” to occur, however, NLP cannot be limited to queries; it must also occur in the entire textual or speech environment.

4. NLP METHODOLOGY TO MACS

4.1. Alternatives to Incorporate NLP into MACS

There were three alternatives in incorporating the NLP capability into the current MACS. The first alternative was to develop a NLP agent using a Commercial Off-The-Shelf (COTS) product. The code necessary to develop a NLP agent was already written so that this would be an efficient and effective way to implement the NLP. However, since Agent BuilderTM does not provide the NLP capabilities, the original version of MACS had to be migrated to the COTS chosen, requiring additional time and effort. The second alternative was to develop a NLP agent using the NLP components available on the Web and then interface them with MACS in Agent BuilderTM. Similar to the first alternative, the advantages were that the code was already written and customizable. Components from a single source would have been preferable as they were usually created to work together for seamless integration. Components from multiple sources would require intensive work to integrate them, causing a problem of interoperability. However, most of these components were not compatible with each other and many were not built to work on a PC platform, a MACS requirement. The third alternative was to develop all of the NLP components necessary to build an NLP agent from scratch, including a parser, lexicon, language generator, etc. Although this option would have been the ideal since the NLP agent would be tailored specifically to the domain and user needs, resource limitations precluded this option.

After evaluating the pros and cons of each option, we chose to use a publicly available software agent architecture called Open Agent Architecture (OAA), developed by the Artificial Intelligence Laboratory of the Stanford Research Institute (SRI). The

OAA system includes an NLP agent as part of the package. This NLP agent supports the translation of English sentences into the Interagent Communication Language (ICL). This section describes in detail the OAA, the architecture of MACS developed with OAA, and the NLP agent in OAA using the Definite Clause Grammar-Natural Language (DCG-NL).

4.2. Architecture of MACS

The system architecture consists of nine agents — a User agent, a Facilitator agent, a NLP agent, a Machine Learning agent, and five specialty agents. The specialty agents are encoded with domain knowledge about the five general areas of expertise required of AROs/COTRs, and the User agent interfaces with AROs/COTRs. Interaction between AROs/COTRs and the system occurs through either keyword searches or natural language queries. As shown in Figure 1, the MACS architecture implements a typical three-tiered brokered architecture. The Facilitator agent coordinates agent activities and communicates with the agent(s) capable of responding to an incoming query.

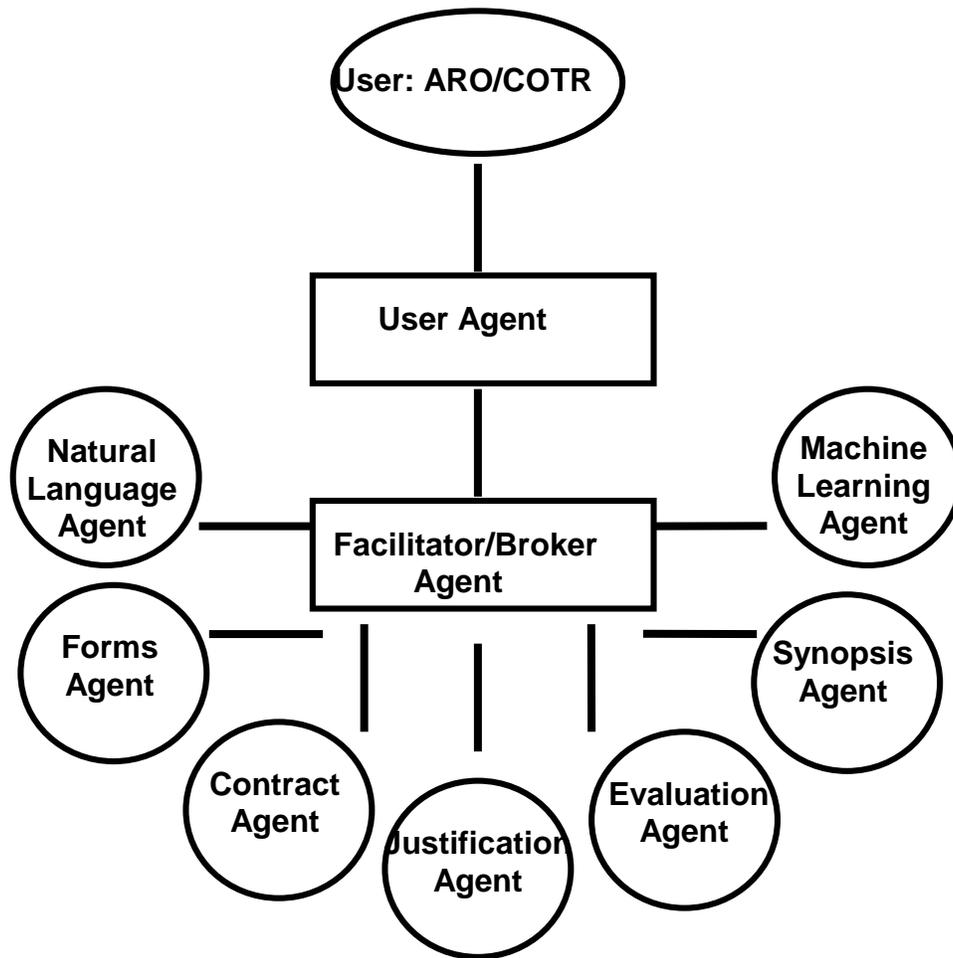


Figure 1. Agent architecture and communication channels

The User agent interacts with the user/ARO/COTR to welcome the user, asks what pre-award questions the user has, and serves as the interface between the user/ARO/COTR and the other agents in the system [Liebowitz et al., 2000b]. Two business logic threads have been designed into the User agent. One thread supports the NLP capability of the system, and the other supports the keyword search capability of the system. The User agent sends incoming user queries to the Facilitator agent, which is

responsible for communicating with all of the other agents in the MACS system as illustrated in Figure 1.

The control flow for the NLP thread is very straightforward. Humans interact with the system by submitting an English sentence into a form displayed by the User agent. The User agent accepts the sentence and passes it to the Facilitator agent with a request to translate the sentence into ICL (ICL is discussed in a section that follows). The Facilitator agent forwards this request to the NLP agent. The NLP agent attempts to create the corresponding ICL and that ICL is then returned to the Facilitator agent. The Facilitator agent then returns the ICL back to the User agent. Since the User agent has dynamically generated the ICL, it is ready to perform the next step in the process, that is, to apply the ICL. Essentially, the ICL is a plan of action that should be taken to solve the initial user's request. The User agent then submits the action plan to the Facilitator agent for completion. In MACS, the action plan indicates which specialty agent(s) should be contacted to respond to an incoming query. The Facilitator agent completes the action plan by performing the necessary low-level communication between the specialty agents. These communications lead to solutions being returned from a specialty agent (s) to the Facilitator agent and then back to the User agent. The entire *control flow* is a two-step process. The NLP agent supplied with OAA has a singular purpose -- it attempts to translate English into ICL. The User agent submitting the English sentence needs this step to occur to determine how it should proceed, and it also affords the User agent an opportunity to apply its own criteria. Assume that the User agent receives a sentence from a pager. Next, assume the pager message was "Please send all your password files to stuart.lowry@saic.com!" The User agent would then send this sentence to the NLP agent

to translate it into the appropriate ICL to satisfy the request. The ICL is generated and returned to the User agent. Fortunately, the User agent has an opportunity to authenticate the source of the message to determine if the originator has the authority to receive the password files. This is a contrived example, but it demonstrates a fundamental reason why the NLP agent simply translates English into ICL. There are endless reasons why additional logic needs to be implemented before simply interpreting the ICL automatically.

The five specialty agents in the system relate to the pre-award phase of a contract and include the *Forms*, *Justification*, *Evaluation*, *Synopsis*, and *Type of Contract* agent. The Forms agent identifies the forms needed to complete a procurement request package. The Justification agent indicates situations where a justification and approval is required to complete a procurement request. The Evaluation agent provides guidelines for evaluating proposals. The Synopsis agent identifies the type of synopsis for a given procurement request. Lastly, the Type of Contract agent identifies the type and nature of a contract based on conditions such as the source of contract, the nature of the work, etc.

Each agent in MACS contains a rule base and has explicit goals. Its rule base describes how to achieve the goals under varying circumstances. The specialty agents respond to incoming queries by presenting necessary information and/or requirements for AROs/COTRs. For example, the Evaluation agent can assist an ARO/COTR with information regarding how to evaluate a project and what criteria or weights to use for evaluation of a contract. If an ARO/COTR has a question regarding "determining weights on evaluation criteria," the Evaluation agent will reply with "You can develop

your own weights on technical, qualifications, and cost criteria. Generally speaking, a weight of 40 percent (out of 100%) is given to cost." [Liebowitz et al., 2000b].

The knowledge contained within each specialty agent is independent of the knowledge contained within the other specialty agents. Thus, coordination between the specialty agents is not required for the current implementation. However, each specialty agent does coordinate with the User agent in order to answer queries. In the original system [Liebowitz et al., 2000b], the User agent broadcasted messages to all specialty agents.

4.2. Open Agent Architecture

As mentioned earlier, MACS has been implemented using the Open Agent Architecture (OAA), which is supported by the Artificial Intelligence Lab at the Stanford Research Institute. There are many facets of the OAA that have been exploited in the MACS system.

The Facilitator agent functions as part of MACS, but it is a specialized server agent that is part of OAA, and it performs many basic functions. The Facilitator agent has the ability to route messages, manage data, fire registered triggers, as well as accept incoming messages requests. The Facilitator agent sends incoming messages to the appropriate specialty agent, and the specialty agent responses are then relayed back to the requesting agent (the User Agent in the case of MACS).

A process called unification evaluates incoming messages. Unification is a powerful mechanism for determining where the incoming query should be forwarded. For instance, the Contracts agent publishes strings such as "contract(Query,Flag,Result)"

where the arguments that start with an uppercase letter indicate variables. If a request is received in the form “contracts(‘\$25,000’,’sole source’,X)” it will be routed to the contracts agent. The implementation of the MACS system in OAA allows for message brokering. That is, the Facilitator agent automatically routes incoming queries to specific specialty agents based on the "solvable" each specialty agent registers with it.

Furthermore, the rules for MACS have been encoded as XML documents. The XML encoding of the rules offers a flexible means for rule modification that was not possible in the original version of MACS [Liebowitz et al., 2000b]. Each specialty agent loads the XML document as its rule base. The rule base is used to compare against incoming queries to determine if a rule is true or false. The XML rule bases are simply ASCII documents that are served up by a web server. A series of web forms have been designed for modification of the rule bases and general system maintenance. A representative rule is illustrated in Table 1.

Table 1: A Sample Rule of Evaluation Agent

<pre> <rule name="57"> <description> This is rule 57 from the "Evaluation" specialty agent. </description> <condition> <and> <clause>competitive solicitation</clause> <clause>evaluation criteria</clause> </and> </condition> <answer> Use such evaluation criteria as: technical understanding of the requirement (technical approach), management of the company, demonstrated expertise and capability in the areas called for in solicitation, facilities and equipment and cost </answer> </rule> </pre>
Rule 57 from the Evaluation Agent

As the figure illustrates, the rules have two sections. The **condition** section contains the clauses that need to be met for the **answer** to be true. The clauses contained within the condition section map onto a keyword selection user interface that represents the previous work completed on MACS. Notice that the rules contain Boolean sections within them. These Boolean conditions are easily possible when the user chooses supplied phases as in the first version of MACS. The phrases were selected from pull down menus. This meant that when the combination of user-supplied phrases matched the Boolean conditions within the rule, the rule “fired”. The natural language support in MACS utilizes the *content* of the rule conditions by performing information retrieval with the terms generated by the NLP agent. It is not reasonable to assume that a human could construct a sentence that would contain all conditions of the rule. In fact it is not reasonable to assume that a human would submit a sentence that contained a single word in each of the conditions. Therefore, a fuzzier matching technique was implemented for NLP support. The procedure is better explained in a section that follows.

4.4 Natural Language Processing (NLP) Agent—the DCG-NL

The Definite Clause Grammar-Natural Language (DCG-NL) is the NLP agent in OAA that was designed to parse incoming natural language queries. Through the use of a logic-based grammar called Definite Clause Grammar [Pereira and Warren, 1980], the DCG-NL translates incoming English queries into expressions using an agent communication language used in OAA, called Interagent Communication Language (ICL).

ICL is a logic-based declarative language that expresses high-level, complex tasks and natural language expressions. SRI chose a logic-based language because it was important that the language the agents spoke could represent and easily translate back and forth to human language [The Open Agent ArchitectureTM, 2000]. According to one of the creators of OAA, the motivation for including a natural language capability to OAA was because natural language fits very well into the delegated computing model that is the goal of OAA: a user wants to be able to task a community of agents by specifying WHAT they want done, not WHO should do the task of HOW exactly it should be done. As new agents join the community, what the user can say and do should dynamically expand accordingly. Agents who don't know about each other in advance should be able to cooperate and work together under the context of a natural language task. Each agent, when it defines its capabilities to the community, should also be responsible for bringing the natural language words (and sometimes grammar rules) that are appropriate for the agent. To this end, ICL expressions were developed using a logic-based, Prolog-like language because SRI wanted to build into OAA some support for natural language from the beginning and most NL parsers usually create a logic-based "logical form" to represent the meaning of a sentence or query [The Open Agent ArchitectureTM, 2000].

4.4.1 Definite Clause Grammar and PROLOG

The Definite Clause Grammar (DCG) is a type of grammar formalism that is expressed as a series of facts and rules. This formalism is well-suited to describing a wide range of syntactic constructions [Matthews, 1998]. Grammar rules in the DCG-NL are expressed as predicates in which the use of arguments allows information to be passed

between them, much like the syntax of the PROLOG programming language, which is based on unification of arguments. This formalism is essentially independent of PROLOG, however, and it can be written in any programming language that permits unification of arguments. These operations are fundamental in a PROLOG interpreter, which means that DCGs are most easily implemented in Prolog [Gal et al, 1991].

Such a DCG-NL offers important benefits. First, it allows for dynamically adding words and grammars. As new agents come into the community, they can add new vocabulary items and grammars. Second, the procedure to dynamically add words and grammars is very simple. We simply specify to which lexical category the word belongs among the nine currently available ones. Third, it comes with a built-in mechanism which tries to "guess" unknown words such as proper nouns, and also is equipped with an extensive "find" grammar to handle several different ways to "find" an object, such as "Find X", "Get me more information on X" or "I'm interested in knowing more about X" [The Open Agent ArchitectureTM, 2000].

However, the DCG-NL also suffers from limitations. First, it only "recognizes" simple word definitions based on what is given and does not really "know" or "understand" the meaning of words. Second, it is not robust. It only handles syntactically well-formed sentences. Third, it recognizes a very limited number of sentence types because the grammar reflects SRI's original purpose to retrieve information based on very simple sentence structures. For example, it can handle sentences such as "Who is the manager of Adam?" and "What is the weather in Chicago?", but not "Who was the 10th

President of the United States?” or “How long does it take to circumnavigate the world if flying on an air balloon with 3 people with a total weight of 400 pounds?”

4.4.2 Vocabularies

A list of vocabulary items must be registered with each agent so that the DCG-NL can recognize them as it parses queries. In MACS, there are 5 vocabulary lists specific to each specialty (*Forms, Types of Contract, Evaluation, Synopsis and Justification*) plus one list that contains terms common to all domains. The terms are added using the following OAA syntax: *oaa_AddData(vocabulary(Type, Definition), [I])* where the Type is the linguistic category and Definition is the word. For example, to add the words “sole source”, “major” and “synopsise”:

```
oaa_AddData(vocabulary(adj, ['sole source', 'sole source']))
oaa_AddData(vocabulary(noun, ['procurement', 'procurement']))

oaa_AddData(vocabulary(Inf_verb['use', 'use']))
```

Similarly, to remove vocabulary items, the following syntax is used:

```
oaa_RemoveData (vocabulary(Type, Definition), [I])
```

A partial list of vocabulary items added to each specialty agent is shown in Table 2.

Table 2 : A Sample List of Vocabulary Items Added

Specialty Agent	Vocabularies	
CONTRACTS	contract, contracts, contract types, contract vehicle	NOUN
EVALUATION	evaluation, evaluation weights, evaluation procedures	NOUN

FORMS	form, forms	NOUN
JUSTIFICATION	justification, justify	NOUN, VERB
SYNOPSIS	synopsis, synopsis type, synopsise	NOUN, VERB
COMMON TO ALL AGENTS	procurement, solicitations, purchases, award, unsolicited, explicit, R&D, need, use, complete, fill out	NOUN, ADJECTIVE, VERB

The DCG-NL leverages the “trigger” capability of the OAA. When the DCG-NL starts up, it notifies the Facilitator agent that it would like to be told whenever any agent in the community adds data to the system that unifies with the ICL pattern “*oaa_AddData(vocabulary(X,[])*”. As mentioned earlier, each agent uses the *oaa_AddData()* mechanism to extend the NLP capability. The *oaa_addData()* method does not indicate anything other than a statement of fact. Each specialty agent is simply announcing to the system the vocabulary that is relevant to its capability. The Facilitator agent detects that this fact should be forwarded to the DCG-NL. The forwarded message contains the appropriate amount of information that the DCG-NL requires to construct subsequent ICL expressions. The message includes an Agent Identifier, the “solvable” within the particular agent, and the actual content of the “linguistic” fact. When the Type of Contract agent submits the fact:

```
oaa_AddData(vocabulary('noun',[ 'contract vehicle', 'contract vehicle' ]))
```

the DCG-NL will remember that the Type of Contract agent has a “solvable” called “Contract” that is interested in NL queries containing the phrase “contract vehicle”. This information is then used to parse the NL queries.

By abstracting the NLP capability in the agent community by implementing the capability through the use of triggers, the OAA architecture is well suited to allow for

multiple NLP agents. The MACS system can easily “plug” in new/competing natural language processing agents. As new, and improved NLP agents are developed, older less robust NLP agents can be discarded.

Additionally, as new agents join the community and register their vocabulary, the capabilities of the DCG-NL increase proportionally. Likewise, when agents disconnect from the system, the Facilitator agent will route the appropriate message to the DCG-NL agent. The DCG-NL updates its internal tables; this shrinks the NLP capabilities of the community. This dynamic nature of the DCG-NL is the model that all OAA agents strive to implement. It demonstrates the dynamic nature of the OAA framework.

4.4.3 User Queries

Twenty-three sample queries were submitted to the DCG-NL. As indicated in Section 4.4.1(DCG and Prolog), the DCG-NL has a limited grammar. Because the grammatical structures of the queries were more complex than the parser was built for, the 23 queries were modified to fit the simple grammar, as shown in Table 3. Although every attempt was made to maintain the original form of the queries, this was only possible in one instance. A query was repetitively submitted after each analysis of the parsed output (an error message or a partial parse), until a full parse was obtained. Only then was the query considered acceptable to the DCG-NL.

Table 3: Sample Queries

Original Queries	Modified Queries
“Do I have to synopsise if dealing with an 8A firm?”	“Do I synopsise an 8A firm?”
“What do I need to know about procurement forms for ADP actions?”	“Show me more information on procurement forms for ADP actions.”
“What needs to be evaluated when an unsolicited proposal serves as a basis for a sole source award?”	“What evaluation is for an unsolicited proposal for a sole source award?”

4.4.3 Parsed Output

When queries are entered to MACS, the DCG-NL attempts to parse the queries into chunk of words according to its grammar. When it encounters a query it cannot handle, an error message "Parse Failed" is returned. This message is returned only when the entire sentence cannot be parsed. There are cases where the query is partially parsed.

The parsed output takes the form of an ICL expression. ICL expressions are internal OAA representations of the natural language query that the agents can act upon. Each parsed word or phrase must be registered in the vocabulary of the appropriate agent. The name of each agent corresponds to the functor or a term that represents a particular function; for MACS, the functors are “*Forms*”, “*Justification*”, “*Types of Contract*”, “*Synopsis*”, and “*Evaluation*”. In cases where there may be multiple expressions synonymous to the functor, an alias was created so that expressions could be resolved to one functor. For example, for the Evaluation agent, “evaluation” is the function, so the functor is the word "evaluation". Phrases such as "evaluation weights" and "evaluation criteria" were aliased to "evaluation" so that any ICL expression with the

functor “evaluation” will be sent to the Evaluation agent. There also are nouns, adjectives, and verbs that are used by all agents. These words, along with their lexical categories, are combined in one list called "Common". Each word is XML-tagged with its appropriate part of speech. This allows the DCG-NL, powered by the linguistic content of its grammar, to recognize the lexical information so that it can effectively parse the query. Examples of parsed outputs are listed in Table 4.

Table 4: Examples of Parsed Outputs

forms('capital equipment',['major procurement'],)
wh(var(forms([]),go([in(some('PR package'([for(some(procurement(['capital equipment', major])))]))),subject(var()))))
contract(arrangement('labor hour'),[])
evaluation(proposal([unsolicited,for(award('sole source'))]),[])
synopsis('I',procurement(['ADP',under(count(50000,dollars))],[on(schedule('GSA'))])

4.5 INFORMATION RETRIEVAL

The output of the DCG-NL is only the first step to process the query submitted by the MACS users. The parsed output must now serve as input to retrieve information, that is, it must be used to select the right specialty agent to “fire” a rule(s) to answer the NL query. The ultimate goal of MACS is to provide users with information related to defense acquisition, which is stored in a knowledge base in the form of rules. As mentioned earlier, MACS has five specialty agents whose existence is solely to return relevant textual information to contract and acquisition personnel based on the query submitted to it. In many cases, queries are answered by one agent. But in some cases

more than one agent can possess relevant information.

The final process in MACS is classic information retrieval (IR). IR deals with the representation, storage, organization of and access to information items [Baeza-Yates, 1999]. The widely used IR techniques are Boolean [Verhoeff, Goffmann and Belzer, 1961], vector [Salton and Lesk, 1968, Salton, 1971, Salton and Yang, 1973, Yu and Salton 1976], and probabilistic models [Robertson and Sparck Jones, 1976, Sparck Jones, 1979]. The sophisticated models include alternative set theory models, such as fuzzy set [Radecki, 1976, 1977, 1979, 1981], and extended Boolean [Salton, Fox and Wu, 1983], the algebraic models, such as generalized vector space [Wong, Ziarko and Wong, 1985], latent semantic indexing [Furnas, Deerwester, Dumais, Landauer, Harshman, Streeter, and Lochbaum, 1988], and neural network [Kwok, 1995], and alternative probabilistic models, such as the Bayesian inference [Callan, 1996], and belief networks [Ribiero-Neto and Muntz, 1996]. Each exhibits unique strengths and weaknesses; the selection of a suitable model is dependent on the domain data set, available technical resources, and usage constraints, such as speed, storage, trade-off between recall and precision, and user requirements.

For MACS, we use a combination of basic Boolean, pattern matching, and fuzzy matching algorithm. This combination is suitable for our purposes, since the knowledge base, which is relatively small, is not indexed, and speed is not an issue at this juncture. The output of the DCG-NL provides the keywords which, when located within the condition portions of the rules explicit in the knowledge base, triggers the appropriate agent or agents to retrieve relevant information and return it to the user. A stop-word list,

consisting of non-content words, such as articles, leading ‘Wh’-question words, pronouns, certain forms of verbs and modals, and portions of ICL expressions that can be ignored in the search process, is used to narrow the search space, resulting in efficiency. All extraneous non-alphanumeric symbols and punctuation in the parsed output, such as”)”, “[]”, “””, and “,” are ignored as well.

When the parsed output is sent by the Facilitator agent to an agent capable of handling the query, that agent will "fire a rule" from the knowledge base and return a relevant piece(s) of information to the user; that is, when the condition stipulated in the rule is met, the user is provided with the answer to his/her natural language query. In some cases, the DCG-NL is able to parse sentences based on the internal grammar that SRI provided. In these cases, since the parse was not influenced by any of the specialty agents, the ICL does not contain any indicators that would suggest which specialty agent should receive the request. The User agent simply performs a broadcast and provides each agent an opportunity to try and solve the request. (A request is considered “solved” if an answer is returned. An answer is returned if any one or more rules fire within the specialty agent.) The retrieved information is not returned to the user in any particular order. We do not use any mechanism to rank the retrieval results by relevance or importance.

The following is the process by which information was returned to the query
“What are the major procurement forms for capital equipment?”

Off-Screen Snapped Text...

Rule: 71 *Need definitive design/performance specifications.*

Rule: 72 *Need Statement of Work or description of services to be rendered.*

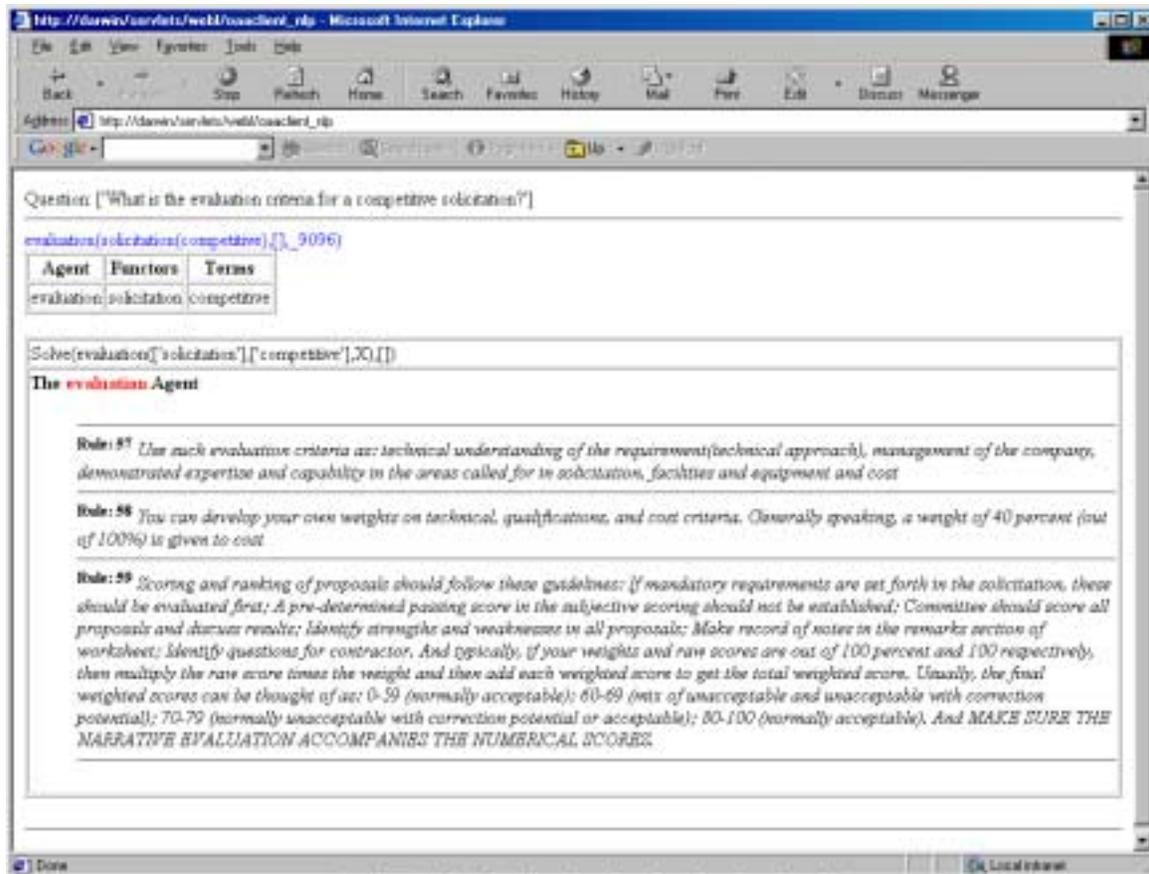
Rule: 62 *Procurement request for lab (NRL) equipment only. (NDW-NRL 4235/3431) Rev. 9/88 Yellow Form.*

Rule: 63 *Procurement request for non-equipment only. (NDW-NRL 4235/3404) Rev. 9/88 White Form.*

Rule: 64 *Include a statement of the Procurement Request as to the initial value of government furnished equipment (GFM) to be modified.*

Since the functor is “forms”, this query is sent to the Forms agent, which then conducts a search for all of the content words, in this case “capital equipment”, “major procurement”. The Forms agent then matches these words to all applicable rules in the knowledge base. For this example, the conditions for Rules 62, 63, 64, 70, 71, 72, 78, 89, 92 contain are met so all information contained in those rules are returned. The user will then determine which information satisfies the information need.

For the query “*What is the evaluation criteria for a competitive solicitation?*”, the process is as follows:



Since the functor is “evaluation”, this query is sent to the Evaluation agent, which then conducts a search for all of the content words, in this case “competitive solicitation”. The agent then matches Rules 57-59.

In order to provide a detailed description of the information retrieval algorithm, it is necessary to identify the important aspects of the ICL statement. Consider the following ICL statement:

synopsis('I',procurement(['ADP',under(count(50000,dollars))],[on(schedule('GSA'))])

The terms in this statement are ‘I’, ‘ADP’, 50000, dollars, ‘GSA’. Terms are those single words (or phrases if multiple words are enclosed in single quotes) which are contained

inside parentheses or brackets. The functors in this statement are “synopsis”, “procurement”, “under”, “count”, “on”, and “schedule”. Functors are the single words (or phrases if multiple words are enclosed in single quotes) which precede open parenthesis. Single words do not need quotes unless they are literals that begin with an uppercase letter. (Uppercase letters normally indicate variables in the statement).

The information retrieval procedure begins in the User agent after it receives the ICL back from the Facilitator agent. The first step is to parse the ICL and reduce it to a list of terms and functors. The stop words are eliminated from the ICL. The list of functors is scanned. If the first functor is not one of the specialty agents, then the remaining functors are examined. Usually, the specialty agent functor occurs first. However, since the ICL actually contains some semantic significance, it is possible that the specialty agent functors occur deeper within the ICL. Therefore, the information retrieval algorithm accommodates this situation by scanning the functors. If no specialty agent is identified, then every agent will be polled to see if they can respond to the query. This completes the User agent’s portion of the IR procedure. The above ICL is flipped into the new ICL statement that will be submitted by the User agent to the Facilitator agent:

Oaa_Solve (synopsis([procurement,under,schedule],[ADP,50000,dollars,'GSA'], X)

Notice that the functor “synopsis” was located which will route this query to the Synopsis agent. The stop words were eliminated, and now the fuzzy lookup can begin. The fuzzy lookup has been described in earlier sections. The Synopsis agent will receive this query. Since there are two argument lists (a list is indicated by the brackets), the Synopsis agent knows to apply the fuzzy lookup. The earlier version of MACS, which supports keywords

picked from a menu, expects only a single list argument. The technique is to extract the condition portion of each rule. The text within this section is searched using case insensitive, regular expressions. An implicit AND is applied. So, if each term and functor occurs in the condition portion of the rule, the rule is “fired”.

We did not compare or evaluate the performance of the results of the NLP capability against the results of the older keyword matching version of MACS due to time constraints. Upon cursory observation, it appears that the NLP version is returning more information than it should, some possibly irrelevant and some contradictory; in other words, there is higher recall (coverage) than precision (accuracy). This is likely to be more a result of possible inconsistencies in the rule base than in the search algorithm, although the parsing process was clearly successful. For the current phase of study, we believe it is better to provide more information to the user than not enough. Additionally, MACS is using the text contained within the condition tags of the rules to perform the information retrieval. This text is the legacy rule condition of previous versions of MACS. It would be more appropriate to create an additional portion within the rule that would contain the text criterion for the natural language matching. This is a trivial technical change. The information retrieval algorithm can easily consider a different portion of the rule. The difficulty is identifying the breadth of sample sentences that are supported by the DCG-NL, and inserting the appropriate terms into this new rule portion section. This exercise is well suited by introducing a user feedback loop, and let the rules expand over time, with proper rule base change management.

5. CONCLUSIONS AND FUTURE DIRECTIONS

This study has applied Natural Language Processing (NLP) to a multi-agent system, MACS, which was constructed to assist the users in defense contracting. MACS is augmented with the natural language processing capability, now enabling the users to enter natural language queries to obtain the information regarding the pre-award contracts. The Natural Language agent successfully parses the natural language queries and selects the specialty agent(s) to answer the user's query by firing the rules in its knowledge base.

Although this research contributes to the existing MAS literature by incorporating NLP, it is the first step toward providing a natural language interface for MAS. The future directions of this research are to develop a more effective natural language interface by:

Using a more robust parser:

Currently, the DCG-NL has very limited parsing capabilities; in fact most of the NL queries had to be modified to fit the grammar rules it supports. The use of a powerful parser, like GEMINI from SRI, will be adapted once it becomes available to improve the flexibility.

Using a semantic component to add more linguistic information to disambiguate word senses:

For any ambiguous queries, a semantically-motivated component, such as WordNet, a concept net, or an ontology, could help the NLP agent better identify the intended meaning or sense of the words in the query, resulting in a more accurate parse and thereby narrowing the subsequent search space during the retrieval process.

Using more linguistically-motivated search methods

The linguistic information contained in the ICL expression resulting from the NLP agent should be exploited fully to improve the overall search effort. This would give the retrieval component more information to base its search criteria in addition to the actual words visible in the queries.

Reorganizing the knowledge-base with indexing

Currently, the MACS knowledge base consists of flat files of rules. Reorganizing this knowledge base by using an index, which is a data structure traditionally used in information retrieval, could increase retrieval efficiency.

Using a mechanism to rank relevance

It would greatly help the user if the retrieved information were ranked in order of relevance or importance. For instance, the information that the system thinks is most relevant or important in satisfying the user's query will be the first one the user will see, with less relevant ones following it. Most commercial Internet search engines employ this mechanism, showing the user a percentage of confidence. This would, however, require more sophisticated search methods, including statistical methods for classification and documents clustering.

Incorporating user feedback

Should the NLP agent need to "better recognize" the input query, a semantic component in the user interface could allow the user to choose the right sense or meaning of the query terms which would transform or expand the original query. This additional information could help the NLP agent better parse the query.

Using information extraction techniques

Tagging or annotating the content words of the input query with linguistic information could give the NLP agent more information on which to make its parsing decisions.

Preprocessing the knowledge base in such a way could give the retrieval component more salient information that could improve performance. These XML or SGML tags would consist of syntactic categories such as <NOUN> or <ADJ>, or unique domain-dependent classification categories such as <FORM> or <EVAL>.

Building a user profile based on query history

Using machine learning algorithms, a user profile could be created by compiling the user's past queries or by using the query histories of multiple users (like Amazon.com) as a way to predict what a user's subsequent query might be. Then, when a particular user logs on to MACS, the system could anticipate the query, thereby putting a particular agent "on notice".

REFERENCES

- Allen, J. *Natural Language Understanding*. The Benjamin/Cummings Publishing Company, Inc., p.2., 1995
- Baeza-Yates, R. and Ribeiro-Neto, B. *Modern Information Retrieval*. Addison-Wesley Longman Ltd., pp. 1, 20-71, 1999.
- Callan, J. Document filtering with inference networks. In *Proceedings of the 19th ACM SIGIR Conference*, 262-269, Zurich, Switzerland, August 1996.
- Feldman, S. NLP Meets the Jabberwocky: Natural Language Processing in Information Retrieval. *ONLINE*, May 1999.
- Furnas, G.W., Deerwester, S., Dumais, S.T., Landauer, T.K., Harshman, R.A., Streeter, L.A., and Lochbaum, K.E. Information retrieval using a singular value decomposition model of latent semantic structure. In *Proceedings of the 11th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 465-480, 1988.
- Gal, A., Lapalme, G., Saint-Dizier, P. and Somers, H. *Prolog for Natural Language Processing*. John Wiley and Sons, 1991, p. 78.
- Grishman, R. *Computational Linguistics*. Cambridge University Press, 1986, p. 4.
- Kwok, K.L. A network approach to probabilistic information retrieval. *ACM Transactions on Information Systems*, 13(3): 324-353, July 1995.
- Liddy, E. Enhanced Text Retrieval Using Natural Language Processing. *Bulletin of the American Society of Information Science (ASIS)*, April 1998.
- Liebowitz, J. CESA: The COTR Expert System Aid. *Acquisition Review Quarterly*, 7(2), Defense Systems Management College Press, Ft. Belvoir, VA, Spring 2000a.
- Liebowitz, J., Adya, M., Rubenstein-Montano, B., Yoon, V., Buchwalter, J., Imhoff, M., Baek, S., and Suen, C. MACS: Multi-Agent COTR System for Defense Contracting. Forthcoming in *Journal of Knowledge-Based Systems*, Elsevier, December 2000b.
- Lotharie, M. *Combinatorics on Words*. Cambridge University Press, 1997, p. xii, xiii.
- Matthews, C. *An Introduction to Natural Language Processing Through Prolog*. Addison-Wesley Longman Ltd., 1998, p. 185.

- Muscettola, N., Nayak, P., Williams, B. C., and Pell, B. Remote Agent: To boldly go where no AI systems has gone before. *Artificial Intelligence*, 1998, 103:5-47.
- Obermeier, K. *Natural Language Processing Technologies in Artificial Intelligence: The Science and Industry Perspective*. John Wiley and Sons, 1989, 47-68.
- Radecki, T. Mathematical model of information retrieval system based on the concept of fuzzy thesaurus. *Information Processing and Management*, 1976, 12:313-318.
- Radecki, T. Mathematical model of time-effective information retrieval system based on the theory of fuzzy set. *Information Processing and Management*, 1977, 13:109-116.
- Radecki, T. Fuzzy set theoretical approach to document retrieval. *Information Processing and Management*, 1979, 15:247-259.
- Radecki, T. On the inclusiveness of information retrieval systems with documents indexed by weighted descriptors. *Fuzzy Set and Systems*, 1981, 5:159-176.
- Ribiero-Neto, B.A. and Muntz, R. A belief network model for IR, In *Proceeding of the 19th Annual International ACM Conference on Research and Development in Information Retrieval*, 253-260, Zurich, Switzerland, August 1996.
- Robertson, S. E. and Sparck Jones, K. Relevance weighting of search terms. *Journal of the American Society for Information Sciences*, 1976, 27(3): 129-146.
- Rubenstein-Montano, B., Lowry, S., Cantu, F., Drummey, K., Yoon, V., Wilson, T., Liebowitz, J., Agent Learning in the Multi-Agent Contracting officer's technical representative System (MACS), Working Paper, 2000.
- Salton, G., Fox, E. A. and Wu, H. Extended Boolean Information Retrieval. *Communications of the ACM*, November 1986, 26(11):1022-1036.
- Salton, G. and Lesk, M.E. Computer evaluation of indexing and text processing. *Journal of the ACM*, January 1968, 15(1):8-36.
- Salton, G. *The SMART Retrieval System – Experiments in Automatic Document Processing*. Prentice-Hall Inc., Englewood Cliffs, NJ, 1971.
- Salton, G. and Yang, C.S. On the specification of the term values in automatic indexing. *Journal of Documentation*, 1973, 29:351-372.
- Shen, W. and Norrie, D. Agent-Based Systems for Intelligent Manufacturing: A State-of-the-Art Survey. *Knowledge and Information Systems, an International Journal*, 1999, 1(2), 129-156. <http://imsg.enme.ucalgary.ca/publication/abm.htm>

- Sparck Jones, K. Experiments in relevance weighting of search terms. *Information Processing and Management*, 1979, 15(13): 133-144.
- Tennant, H. *Natural Language Processing*, Petrocelli Books, Inc., 1981, p.2.
- The Open Agent Architecture™ <http://www.ai.sri.com/~oaa>, 2000.
- USD AT&L (The Under Secretary of Defense, Acquisition, Technology, and Logistics) and The Under Secretary of Defense, Personnel, and Readiness, "The Acquisition 2005 Task Force Final Report on Shaping the Civilian Acquisition Workforce of the Future," Washington, D.C., October 2000.
- Verhoeff, J., Goffmann, W. and Belzer, J. Inefficiency of the use of Boolean functions for information retrieval systems. *Communications of the ACM*, December 1961, 4(12):557-558, 594.
- Wong, S.K.M., Ziarko, W. and Wong, P.C.N. Generalized vector space model in information retrieval. In *Proceedings of the 8th ACM SIGIR Conference on Research and Development in Information Retrieval*, 1995, 18-25, New York, USA.
- Yu, C. T. and Salton, G. Precision Weighting – an effective automatic indexing method. *Journal of the ACM*, January 1976, 23(1): 76-88.