Université
de Montréal

Benchmarking the Performance of Two Automated Term-Extraction Systems: LOGOS and ATAO

Par

Stacy Love

Département de linguistique et traduction

Faculté des arts et des sciences

Mémoire présenté à la Faculté des études supérieures
en vue de l'obtention du grade de
Maître ès arts (M.A.)
en traduction (option recherche)

avril 2000

Université de Montréal

Faculté des études supérieures

Ce mémoire intitulé :

Benchmarking the performance of two automated
term-extraction systems: LOGOS and ATAO

Présenté par :

Stacy Love

a été évalué par un jury composé des personnes suivantes :

Mémoire accepté le : ……………

Abstract

The present work is a comparison of the performance levels achieved by two commercially-available term extraction software programs: ATAO by the TRADUCTIX company and LOGOS 3.0 by the LOGOS Corporation. A comparison of two such software programs would assist potential users, such as a company or academic institution, considering the purchase of such a tool. Furthermore, comparing the two brings to the forefront the issues related to the automatic or assisted identification of compound terms.

To compare the two software programs, a total of 10 electronic texts (more than 30,000 words) from the field of computer science were submitted to each to create a corpus of term candidates. We then analyzed the machine output to determine recurring themes common to both systems, as well as each term-extraction software's strong and weak points. The data we obtained was stored in a Microsoft Access database running underneath a software solution named TEMS (Term Extraction Management System). This tool was developed in Microsoft Visual Basic specifically for the requirements of this study.

This study's focus is limited to compound nominal terminological units occurring in the Language for Special Purposes (LSP) for computer science. However, terminological units from all parts of speech and subject fields are included in our comparison. We believe that because ATAO and LOGOS do not use the same method to select term candidates, their extraction output will demonstrate accordingly different traits.

The extraction data we obtained by submitting our collection of specialized texts to the two term-extraction software programs allows us to measure the strengths of each program, as well as examine areas of each that need refinement. Software manufacturers could readily use the results of our analysis to fine-tune their respective product.

Comparing the machine output by each software program and a manual scanning we completed ourselves led to our observation of a certain number of tendencies specific to each program. They were also compared and contrasted in the areas of performance and accuracy. This type of information would be helpful for potential users of this type of software because they would be able to compare the strong and weak points of each with respect to their own requirements and terminological environment, thereby simplifying the process of finding the program best suited to their needs.

Sommaire

Dans le cadre de ce projet de recherche, nous examinons la performance de deux logiciels de dépouillement terminologique automatisé : ATAO conçu par la compagnie TRADUCTIX et LOGOS 3.0 conçu par LOGOS Corporation. Une comparaison de deux logiciels de ce genre sera utile pour des utilisateurs potentiels comme une société ou un établissement académique lors de la sélection d'un outil de dépouillement terminologique. De plus, cette comparaison met en lumière les questions reliées a l'identification automatique ou assistée de termes complexes.

Afin de comparer ces deux logiciels, nous leur avons soumis un ensemble de 10 textes (plus de 30 000 mots) provenant du domaine de l'informatique pour créer un corpus de termes potentiels. Par la suite, nous avons analysé les sorties-machine dans le but de trouver des tendances et les points forts et faibles de chaque logiciel de dépouillement terminologique automatisé. Les données obtenues sont stockées dans une base de données de type Microsoft Access gérée par un outil informatique qui s'appelle TEMS (Term Extraction Management System). Cet outil a été conçu en Microsoft Visual Basic expressément pour les fins du présent mémoire.

Le focus de cette étude se limite aux unités terminologiques complexes provenant de la langue de spécialité (LSP) de l'informatique. Cependant, des unités terminologiques appartenant à différentes catégories grammaticales et rattachées à d'autres domaines sont incluses dans notre comparaison. Bien que LOGOS et ATAO soient deux systèmes de dépouillement terminologique qui effectuent la même tâche, soit parcourir les textes qui leur sont soumis et en extraire les candidats terminologiques, les méthodes implantées dans chacun d'eux sont différentes. Nous avons donc avancé l'hypothèse que les résultats du dépouillement terminologique se révéleraient également différents.

Les sorties-machine obtenues après la soumission de notre ensemble de textes spécialisés aux logiciels de dépouillement automatique nous permettent de mesurer les points forts de chaque système, mais aussi de mettre en lumière certains aspects qui nécessitent des améliorations. Les concepteurs de ces logiciels pourraient profiter grandement des résultats de notre analyse afin de perfectionner leur produit respectif.

En comparant les résultats générés par le dépouillement automatique de chaque logiciel et un dépouillement manuel que nous avons effectué, nous avons pu constater quelques tendances propres à chaque logiciel et les comparer sur le plan de la performance et du rendement. Ces informations peuvent très utiles pour un utilisateur potentiel d'un tel logiciel. Il sera alors en mesure de peser le pour et le contre des deux logiciels dans le cadre de ses besoins propres en outils terminologiques, ce qui en définitive lui permettra de faire un choix plus éclairé.

Acknowledgments

A project of this magnitude could not be completed without the assistance of others. These are the key individuals who helped me realize this study.

I would like to express my gratitude first and foremost to my research advisor, Marie-Claude L’Homme, for her attention, enduring patience and understanding throughout the writing of this work. Her input and suggestions are both greatly appreciated and constitute an integral part of the final product.

Thanks go out to Claude Bédard of the TRADUCTIX company for graciously allowing me to use the term extraction software program, ATAO, as part of this study. Similar thanks are extended to Brigitte Orliac and Scott Bennett of the LOGOS Corporation for granting me permission to use the noun phrase search tool included in their machine translation system, LOGOS 3.0, in this study to compare to ATAO.

The long-term contribution made by Benoît Perron, Djamel Kacel and Sylvain Hamel, including invaluable technical insight, design recommendations and brainstorming, helped make the TEMS program into a robust, highly-useful Windows application. Their assistance also stamped out the last few bugs in TEMS and made me a better programmer in the process.

Finally, a special thank you goes out to my colleagues, Carolyn Bastable and Isabelle Meynard, for their encouragement, as well as for taking the time to painstakingly proofread and comment this work from end to end.

Table of Contents

List of Tables

List of Figures

Introduction

The object of our study is to carry out a comparative examination of two software programs (LOGOS 3.0 and ATAO) designed to perform automatic term extraction on texts written in English: one of the methods currently being employed to alleviate the terminologist's task of managing the terminology of a given subject field. Automated term extraction can also be beneficial for translators who need to manage their terminology and create term records as part of their work. By completing this study, we hope to contribute to the efforts already being deployed to improve the performance of future automated term-extraction systems.

Term extraction is a process by which texts are scanned for the terminological units they contain in order to enrich lexicographic resources or assist with translation. Software solutions that can automate the process by scanning texts for terminological units, extracting word combinations that fulfill preset criteria and generating reports for filtering are extremely helpful to terminologists and translators because they automate a task that can otherwise be a time-consuming, and hence costly, undertaking. An example of such a list appears below:

```
title bar
message box
status bar
option button
command button
tooltip
operating system
continuous voice recognition system
desktop hardware
discrete voice recognition system
discrete voice recognition technology
demand-driven economy
commercial product
continuous voice recognition
technical writer
nonprofessional writer
text formatting
font selection
spell checker
training process
enrollment process
```

Two concepts from the field of automated language processing which are frequently confused are "machine translation" and "computer-assisted translation." For the purposes of this study, the distinction between these two types of applications is necessary so that our findings are interpreted correctly.

"Machine translation" is a term covering a range of different sub-procedures that when completed together, a text is translated from one source language to one or several target languages by a specially-designed software program. The software program performs an analysis of the input text that varies in scope from almost nil to multi-phase, in-depth morpho-syntactical and even some elements of semantic evaluation then generates a raw translation in the target language(s).

"Computer-assisted translation" on the other hand, is a term that designates the translation work environment that includes any or all computerized utilities translators can use to accelerate and/or simplify certain tasks. Computerized linguistic databanks such as Termium and Eurodicautom, spell checkers and term-extraction utilities are all examples of tools that can be part of the computer-assisted translation environment.

The key difference between these different approaches to the same situation is who completes the transfer to the target language: the human translator or the computer. In machine translation, the human assists the computer prior to submission to the system by grooming the input text through tagging proper nouns, lexical units not to be translated and unknown words. Once the system has produced its translation, the human post-edits it for accuracy and style. In the computer-assisted translation environment, however, the human translator uses the computer’s lightening-fast data manipulation capabilities as a resource while the actual translation itself takes place in the translator's mind.

Both LOGOS and ATAO are linguistic software solutions designed expressly for translation, and both include a utility for extracting terminological units which are compared in this research work. However, LOGOS is a complete machine translation system; whereas ATAO is part of the computer-assisted translation environment designed to complement the human translator's efforts before and after translation. LOGOS' term-extraction utility was chiefly intended to help enrich its translation databases and reduce mistranslations brought about by spelling errors or variations in the input text. It does not attempt to help human translators with their work like ATAO does. ATAO attempts to provide the human translator with as much information possible by collecting and presenting term information with the human translator's needs in mind. This shift in focus means that LOGOS and ATAO cannot be evaluated in a direct comparison against one another because although they fulfill the same needs, they serve two different masters. Their respective performance can nonetheless be evaluated quantitatively for thoroughness and suitability for the intended purpose of each.

These tools are part of the environment of computerized aids intended to enhance the productivity of translators and terminologists. Computerized language utilities are a promising new concept, and many language professionals are open to the prospect of incorporating them in their daily work. However, there are very few established methods for assessing their performance, and even fewer impartial evaluations currently available to potential users. This work is an attempt to fill this need by documenting a method for comparison and applying it to commercially-available products. A chapter-by-chapter outline of our study is presented below.

Chapter 1 of this study introduces the pertinent concepts and lays the foundation for the rest of this work. We begin by discussing what constitutes a terminological unit, drawing upon the works of several well-known authors, then focus on an introduction to automatic term extraction. Here, we elaborate on human term extraction and the most common automated term-extraction strategies, then explain the issues of “terminological noise” and “terminological silence” and give the principal causes of each.

Chapter 2 provides a tour of a number of key players in the field of automated term-extraction software. Extraction strategies, including the shortcomings of each, are discussed and the two automated term-extraction utilities compared in this study, LOGOS 3.0 and ATAO and examined closely. The chapter concludes with a summary of the current state of automatic term extraction.

Chapter 3 introduces TEMS, the *Term Extraction Management System* developed for this study that was used to manage the terms on the human list, as well as the thousands of term candidates extracted by LOGOS and ATAO. The program’s technical specifications are given and annotated screen captures explain how the system was used to keep the volumes of term candidate data under control.

Chapter 4 is a step-by-step explanation how our experiment was carried out: how we created the human list of terms from our collection of electronic texts, gathered our term-extraction data then interpreted it to generate statistics based on our findings. Special cases, guidelines when processing our data and exceptions are also mentioned.

Chapter 5 presents our findings from the comparison described in Chapter 4. Our initial expectations, the results that actually came about, and any surprises, impressions and observations are all discussed here. Our final results are compared to those from similar investigations. A critical evaluation of each automatic term-extraction package is also made.

Chapter 6 is a summary of our work with propositions for possible applications for this research. We also give our suggestions and personal recommendations for improving term extraction software, and conclude with our ideas for possible future work in this area.

An electronic version of this research study is available
on the compact disc attached to the back cover.

1. THE TERM AND TERM EXTRACTION

1.1. Terminology as a discipline

1.1.1. Why terminology is important

As shown by the respected anthropologists, Edward Sapir and Benjamin Lee Whorf, through the course of their extensive research in the area of language, culture and society, all human cultures and communities on earth have a spoken language [Mandelbaum (1949:7)] or "…a purely human and non-instinctive method of communicating ideas, emotions, and desires by means of a system of voluntarily produced symbols," [Sapir (1921:7)] which they use to share information with one another. According to the Sapir-Whorf theory of *linguistic determinism*, the way we think is largely determined by our language. In his own words this time, Sapir (1958 [1929]:69) summarized his position as follows:

> Human beings do not live in the objective world alone, nor alone in the world of social activity as ordinarily understood, but are very much at the mercy of the particular language which has become the medium of expression for their society. It is quite an illusion to imagine that one adjusts to reality essentially without the use of language and that language is merely an incidental means of solving specific problems of communication or reflection. The fact of the matter is that the 'real world' is to a large extent unconsciously built upon the language habits of the group.

Terminology encourages communication through language by providing speakers with the lexical units they require for effective dialogue. More specifically, terminology is defined by Dubuc (1992:3) as "…une discipline qui permet de repérer systématiquement, d'analyser et, au besoin, de créer et normaliser le vocabulaire pour une technique donnée, dans une situation concrète de fonctionnement de façon à répondre aux besoins d'expression de l'usager"[1]. Terminology science identifies the relationships between concepts that are complex and often interrelated. Furthermore, it is a vital part of all technological and scientific fields. Superficially-speaking, terminology can appear to be mostly semantic and linguistic in nature; when in fact it is it actually rather multidisciplinary, touching on such diverse areas as information science, epistemology, communications and logic.

Although terminology in one form or another has existed for centuries, only in the twentieth century has terminology science as we now know it come of age as a recognized and independent field. Moreover, this fledgling discipline is still undergoing the process of defining its theoretical foundations and scope. For example, the definition of the concept of "terminology" itself is far from unanimous, even among the discipline's own theorists and professionals.

1.1.2. Why terminology is an <u>increasingly</u> important part of our daily lives

The volume of appropriate designations required for emerging concepts, processes and even commercial products is greater than ever before. More specifically, the reason why the need for terminology is growing exponentially can be broken down into four main categories.

Advances in science and high technology's exponential growth: We are currently experiencing exponential growth in human knowledge overall, and every new scientific or technical discovery must be given a name so that it may be discussed at large and published in various media. Highly-specific naming conventions ensure that research is not duplicated between different teams of scientists or even by fellow members of

[1] A practice that involves the systematic collection, analysis, and when needed, the creation and standardization of vocabulary for a given subject field so as to fulfill the expressional needs of the language's speakers. (Our translation)

the same team because such monosemic designation clearly indicates the nature of the work at hand. Facilitating effective communication and comprehension between individuals is a justifiable, cost-effective venture for small private businesses and multinational corporations alike. If a concept does not have a linguistic representation, discussion in its regard is severely hindered, thereby stunting both the concept's development and further innovation.

Globalization of the economy, thereby making the need universal: Companies from all corners of the globe are diversifying their activities on an international scale in an attempt to attain a greater market share and increase cost-effectiveness. Widespread business activity requires terminology in multiple languages so that the workers in all of a given corporation's subsidiaries or local offices can communicate effectively and efficiently. Before a designation may be translated, however, it understandably must exist in the source language.

Availability and affordability of computer hardware and software: The price of computer software and particularly hardware is in a continual downward spiral. Most newly-purchased hardware loses a significant percentage of its value within a short period of time after purchase. On a more positive note, however, this devaluation also means that powerful, fully-adequate computer hardware and peripherals may be purchased at a reasonable price by most businesses and individuals. Specific to terminology and term creation, businesses can now afford to manage their terminology files electronically in a searchable database instead of paper card files, which encourages the standardization of terminology through improved access. Furthermore, the relative low cost and availability of commercial linguistic data banks, such as *Termium* by the Government of Canada's Translation Bureau or *EURODICAUTOM* produced by the Translation Service of the European Commission, make a considerable contribution towards large-scale terminological acceptance and standardization.

Communication is less expensive and easier than ever before: Never before have people been more accessible in more ways: e-mail and the World Wide Web, video-conferencing, faxing, electronic data interchange (EDI) and inexpensive long-distance telephone calls are only a few of the ways in which people may now communicate with one another. As such, new designations for emerging concepts must be coined even sooner than before because details of their discovery reach more people earlier and in less time. Furthermore, the availability of texts in electronic form on the Internet and reduced printing costs make research results highly accessible to massive numbers of readers.

1.2. The terminological unit

1.2.1. Definition of the term

Given that the focus of our research is the extraction of terminological units in free-running text, an in-depth definition of the *term* is a critical part of establishing the foundation upon which our findings will rest. We will begin by presenting an overview of the definitions of the term found in works by many widely-cited authors of terminology textbooks, among them Dubuc (1992), Sager on his own (1991) and co-authoring with Dungworth and M^{c}Donald (1980), and Picht and Draskau (1985). We will then bring to light the differences between simple terms and compound terms according to the authors and provide an in-depth look at compound terms, including a typology of the compound terms found in the English language. We follow with a discussion of which lexical combinations in our corpus texts should rightly be conferred terminological status within the framework of this research, in other words, the human criteria for term extraction.

1.2.1.1. Perspective of Robert Dubuc

Dubuc (1992) provides us with a concise, generalized definition of the terminological unit, then elaborates upon his basic definition in the sections that follow.

> Le terme, encore appelé unité terminologique ou terminologisme, est l'élément constitutif de toute nomenclature terminologique liée à une langue de spécialité. On peut donc le définir comme l'appellation d'un objet propre à un domaine donné (1992:25).

Dubuc's belief of what constitutes a term relies heavily on the "linguistic sign" developed by the Swiss linguist, Ferdinand De Saussure, in his work *Cours de linguistique générale* (1974:99). Saussure's model is comprised of the "signifié" and the "signifiant" and is illustrated as follows:

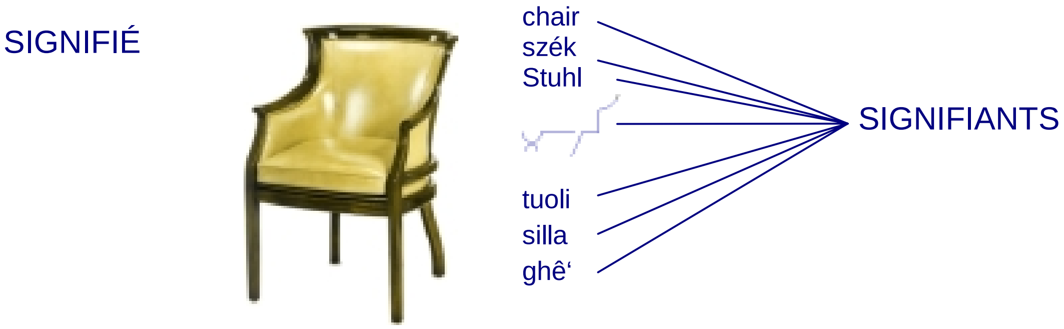


Figure 1-1 Saussure's linguistic sign

The linguistic sign is the result of the relationship that unites a term with its notion. It is this tacitly-accepted bond between an object and its linguistic rendering that has all English-speaking individuals refer to the signifié in our example as a *chair* instead of another signifiant. Furthermore, this bond prevents the signifiant *chair* from being used to designate any signifié except for the one represented in this figure (excepting, of course, instances of homonymy, such as the "chair" of a meeting in this example). This relationship is arbitrary, but generally indissociable once formed.

It is believed that the signifiant is more frequently motivated in *Languages for Special Purposes* (LSPs) than in non-technical vocabulary. According to Dubuc, a signifiant is considered "motivated" when it is comprised of semes that allude to the signifié it designates. This allusion can be made through either the semes' etymology or meaning. Over the course of time, the semes' individual motivation can become obscure or even fall from use completely; however, the term's functional value remains unaffected. Many motivated terms can trace their etymology back to Latin or Greek roots; an example of this type of motivated term is *leukemia* [Greek *leukos* (white) + -*aimia* blood <haima], a type of cancer characterized by the abnormal growth of white blood cells. An example of a motivated term whose semes' individual motivation has been preserved is *lunge whip* [lunge + whip], which is a specific type of whip used when lunging a horse. Dubuc concludes by stating that, although not mandatory, motivation when coining neologisms is highly desirable.

1.2.1.2. Perspective of Juan C. Sager (and Sager *et al.*)

Sager (1991) writes that terms are "the linguistic representation of concepts". The term itself is but an element of a trilogy Sager calls the *Term-Definition-Concept Equation* which he opts for in place of Saussure's linguistic sign. Sager's Term-Definition-Concept Equation is expressed thus:

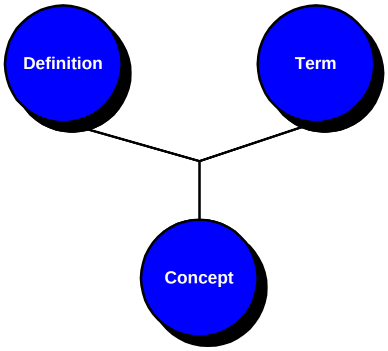


Figure 1-2 Sager's term-definition-concept equation

where the concept is analogous to the signifié and the term is the signifiant.

More specifically, Sager (1991:2) defines terms as "lexical items belonging to a specialized area of usage in one or more languages". The collective of all the terms of a given discipline is known as its *terminology*. The same lexical unit can be both a term in an LSP and a word in general usage at the same time. One example is the lexical unit *salt*. In *Language for General Purposes* (LGP), salt is a white crystalline mineral used to preserve and season foods. In the LSP for chemistry, however, a salt is a usually crystalline chemical compound derived from an acid by wholly or partly replacing the hydrogen by a metal or an electropositive radical. Terminologists require a foundation of subject field knowledge that justifies the existence of LSPs and delimits their boundaries in order to work proficiently with terminology.

Sager defines the definition as "the explanation of the meaning of linguistically-expressed symbols," Sager (1991:39). It is possible to define both tangible and intangible objects, objects that do not exist (ex: *leprechauns*) and even objects that cannot exist (ex: *a three-sided square*). Definitions encapsulate the linguistic description of a concept, which conveys the concept's meaning. The description is expressed through an enumeration of a certain number of the concept's characteristics. Definitions presuppose that the vocabulary they use is understood by the reader. This is especially the case of the definitions found in general language dictionaries, whose purpose is to distinguish between polysemous words and homonyms, and explain the meaning and usage of less common words.

To round out the *Term-Definition-Concept Equation*, Sager summarizes his understanding of what constitutes a concept in the following, albeit provisional, definition as "…constructs of human cognition processes which assist in the classification of objects by way of systematic or arbitrary abstraction," Sager (1991:22). He acknowledges that there exists considerable divergence of opinion in this matter and chooses to leave it more or less undefined and considered as an "axiomatic primitive, like word or sentence" (1991:23). A concept must exist before a term can be created to represent it because terms are the symbols used to represent concepts. Concepts are units that are described, organized and related in a structure Sager refers to as a "model of knowledge". This model is an abstract hierarchy of several axes representing everything known about a particular discipline. Each concept has its own zone of coordinates within the model of knowledge.

He contrasts words and terms by stating that the arbitrariness of the linguistic sign is accepted in general language, thus supporting Dubuc; whereas LSPs strive to systemize principles of designation and name concepts according to pre-defined standards or basic guidelines. He adds that general language makes full use of polysemy, metaphor and adjectival determination, and that true word creation is relatively rare.

Focusing on terms, Sager Dungworth and M^{c}Donald (1980:243) state that, unlike words, whose origin is only sometimes traceable, terms are the result of more or less conscious creation. “The overall aim of designation in special language is to achieve transparency and consistency: a designation often reflects in its structure the major conceptual features it represents and in such a way that related terms have a comparable expression”. Later on in their work, Sager, Dungworth (1980:249) and M^{c}Donald add that “a systematic method of designation will attempt to reflect in the term the essential relationships that operate among the concepts”. On his own this time, Sager writes that the overwhelming majority of concepts present in LSPs are expressed in the form of nouns. He continues by explaining that the English language makes extensive use of both analytic and synthetic[2] methods of term formation.

[2] Synthetic methods modify existing lexical units through the use of “affixes”, which are lexically-meaningful items that cannot stand on their own in English. These items can be placed at the beginning of a lexical unit (prefix), at the end of a lexical unit (suffix) or, in some languages such as German, within the lexical unit (infix).

One difference we have observed between prefixes and suffixes in English is that prefixes often contribute conceptual information to the lexical unit it modifies; whereas suffixes tend to provide merely grammatical attributes, such as changing the part of speech or indicating the “state” of the lexical unit’s nucleus. Some affixes can be used as both a prefix and a suffix, such as *path* in homeo**path** and **path**ological; however, most others are limited to being either a prefix or a suffix. Affixes are often derived from other languages; their function in their language of origin, however, is irrelevant in this context because it often differs from their function in modern English. Other affixes possess no meaning on their own, yet acquire one of several possible meanings when combined with a stem.

Analytic designation methods couple independent lexical units to form larger units, such as when compounding. An example of two simple terms combining to form a compound term is *backup tape*, a magnetic storage device (tape) used specifically for the creation of additional copies of electronic data (backups). An example of a simple unit binding with a complex unit to form a multiple compound is *startup Java-script*, a series of instructions (or script) written in the Java programming language used to initiate the Virtual Machine on the client system (the “startup” event in the code environment). Lastly, an example of a compound binding with another compound to form a multiple compound unit is *backup tape retrieval procedure*, where *backup tape* is added to the term’s nucleus to explain the nature of the retrieval procedure in question.

Terms are most often formed through determination, and combining object and function or characteristic is the most common and important specification in technology, for example, *nutcracker* (an instrument that cracks nuts) and *continuous form feed* (form feed that is continuous). In English, the determining modifier, or “determinant,” usually precedes the concept being modified, the hyperonym called the “nucleus”. Difficulty understanding the term’s meaning may arise if this is not the case. Hyphens are generally used to clarify terms that have multiple modifiers, such as *high-tension wire*. Longer terms sometimes omit one of their elements to shorten the expression. There is no set pattern for such truncations, however.

“Conjunction” is a method of term formation whereby two different characteristics of the same concept are conjoined to create a term where both enjoy equal status and are clearly identifiable within the newly-created term, such as *parent-child agreement* or *student-teacher ratio*. Portmanteaux terms, such as *smog* (formed from *smo*ke and f*og*) or *contone* (formed from *continuous* and *tone*), are another representation of conjunction. Its counterpart, “disjunction,” occurs when two or more concepts are conjoined to form a new hyperonym that presents alternatives to a single concept and hence becomes an “either/or” relationship, such as *input/output device*.

How terms are created to designate concepts generally falls into one of the four following categories:

- Using existing phonemic and morphemic resources
- Modifying existing lexical resources
- Coining neologisms
- Borrowing from other languages

According to Sager *et al.*, English initially tended to favour the first method of designation; however, after the Norman Conquest of 1066, preference shifted to the other three methods listed above, particularly the second.

Today, the most commonly-used method of designation is modifying existing resources through derivation and compounding. What this method attempts to do is to zero in on a specific area of the knowledge structure by juxtaposing notions and concepts. This notional and conceptual juxtaposition creates new concepts through compounding or modifying existing terms through affixation, thereby forming the desired more precise designation.

Sager, Dungworth and M^{c}Donald (1980:257) contend that, while general language feels the need to distinguish between these two methods, it is neither possible nor useful to do so for LSPs because the delimitation of affixation with respect to compounding is fuzzy. Their assertion is based on the premise that the definition of an affix used for general language cannot be properly applied to certain Greek or Latin roots which can both be used as affixes and stand on their own, ex: *graph*, or *gram*.

1.2.1.3. Perspective of Heribert Picht and Jennifer Draskau

Instead of formulating their own definition of the term, Picht and Draskau opted to use the following definition of the term proposed by ISO/R 1087:

> Term (for a concept): Any conventional symbol for a concept which consists of articulated sounds or of their written representation (= of letters). A term may be a word or a phrase (1985:96).

To this basic definition they add that a term may be comprised of a single morpheme, ex: *dipstick* or contain multiple morphemes, ex: *graduated pipette*.

As with Dubuc, Picht and Draskau make use of Saussure's linguistic sign (see Section 1.2.1.1 for a more detailed description) to illustrate the two aspects of the term: the content (or semantic value) and the expression (or the communicable linguistic form). They state that, for a concept to be expressed, it necessarily requires a sign. Furthermore, they advance that a sign without content is invalid, just as is a concept without a sign. A concept and its linguistic realization are like the front and back of a piece of paper: they are indissociable and one is necessary for the existence of the other. Lastly, for a sign to have communicative value, its content must be known to both the sender and receiver (destinataire and destinateur) involved in the communicative act.

Picht and Draskau believe that there exist two types of signs: "natural signs" and "conventionalized signs". Natural signs retain a fundamental and inherent relationship between the signifié and the signifiant, such as smoke → fire, and are based on a causal relationship. Conventionalized signs, on the other hand, are based on an expressed or tacit agreement between a minimum of two individuals. The sign-concept agreement is, for the most part, fixed and relatively stable. In the majority of cases, the sign's linguistic expression consists of a word or term. It can, however, take one of the four following forms:

Type of designation	Example
Ideogram or symbols A stylized figure which conveys simplistic unequivocal messages. Such designations are not language-specific.	The plus sign + on a calculator ➕ The print icon used in software 🖶 The yield sign on highways ▽
Number Arabic or roman numerals	1, 2, 3, 4 … i, ii, iii, iv …
Notation Means of identifying elements of a group by means of a series of alpha and/or numeric characters.	 A licence plate number Notations from the Dewey Decimal System for classifying library books
Denomination Sequences of letters ordered according to convention. This form constitutes the vast majority of designations. When they refer to individual concepts, they are called names; when they refer to general concepts, they are called terms.	HP LaserJet 4M (proper name) laser printer (term)

Table 1-1 Picht and Draskau's typology of linguistic expression forms

To better define what is a term, Picht and Draskau contrast terms (LSP) with words (LGP). They note that, although there are very few characteristics of the linguistic form of terms which cannot equally be observed in words, analyzing the semantic content of a term reveals that terms possess a higher level of precision than do words. Furthermore, terms are part of a larger system of notions, which usually influences the creation of new terms within the same system. This position is supported by Otman (1991:67) who states that the syntactic structures used by LSP and LGP are the same. He notes, however, that LGP enjoys greater freedom regarding linguistic forms; whereas terms in LSP must adhere to the restrictions of the subject field and integrate well in a "coherent and structured" notional system.

1.2.2. Compound terms

Regardless of the language, there are more compound terms in LSP than other lexical formations. Moreover, compounding is undoubtedly one of the most prolific methods of term creation according to Picht and Draskau (1985:108). Most neologisms are nouns, as opposed to verbs, adjectives, adverbs or any other part of speech because the majority of new concepts (i.e. signifiés) that require designation (i.e. by a signifiant) are nouns, and not new actions or new ways to describe nouns. In the following section, we give a detailed look at each author or group of authors' perspective on compound terms. Other areas covered include term formation and proposed naming requirements for effective term creation.

1.2.2.1. Perspective of Robert Dubuc

Dubuc (1992) believes that terms are either simple (i.e. composed of a single lexical unit) or complex. Compound terms are then divided into two subgroups, compound words and syntagms. *Latency*, *ticker*, *thread* and *plotter* are all simple terms. Examples of complex terms are *uptime*, *bottleneck* and *middleware*, and *vending machine syndrome*, *cartridge load/unload time test* and *specialized disk array storage system* are all syntagms.

He then goes on to explain that in French, compound words have hyphens between each element to indicate a single unit, for example, *porte-aiguille*. He mentions, however, that this practice is falling from use.

Complex terms that are not linked by hyphens are referred to as “syntagms”. He defines a syntagm as a series of words linked by an identifiable syntactic relationship called a “rapport”. He adds that it is unnecessary for a syntagm to be absolutely indissociable or fixed to be considered a term. In short, syntagms may accept qualifiers or specifics without losing their terminological status. For example, *networking client* and *TCP/IP networking client* are both equally acceptable terms. The qualifier *TCP/IP* does not in any way cause the syntagm to no longer be considered a term, nor does it cause it to be considered a different term because of this precision.

To clarify which types of objects may have their signifiant considered as a term, Dubuc provides us with the following list of eligible categories to which we have added examples from the LSP for target rifle shooting:

Category	Example
Tangible objects	clay pigeon
Actions or processes	three-position zeroing
States or situations	range-wide cease fire
Phenomena	hang fire
Characteristics	rim fire
Procedures or techniques	muzzle loading

Table 1-2 Dubuc's eligible categories for term creation

Lastly, Dubuc indicates that, ideally, only one term (signifiant) is used to refer to a given signifié and that each signifié has but a single signifiant. This is known as "biunivocity" or "one-to-oneness" that excludes "synonymy," a common occurrence in general language where several signifiants designate a single signifié. He notes that these observations apply equally to simple terms and complex terms.

1.2.2.2. Perspective of Juan C. Sager (and Sager *et al.*)

Sager believes that simple and compound terms which express scientific and technical concepts must fulfill certain conditions, and that this applies to both existing vocabulary and the formation of new terms. Before presenting us with his list of naming criteria, Sager qualifies it with the caveat that it could only be fully realized in a strictly-controlled environment (1991:89). We found Sager's list of naming criteria serendipitously helpful when deciding whether certain lexical combinations were indeed terms that should be extracted for the human list (see Section 4.2 for details on the creation and refinement of the human list). It constitutes a quasi-checklist of possible terminological status indicators for questionable terms. Sager's list was only used as a guide and not followed stringently, however, because, as he conceded himself, it could only be fully applied in an optimized setting. Unfortunately, most terms are not coined under such favourable circumstances.

Sager's list of naming requirements is given below:

1. The term must relate directly to the concept. It must express the concept clearly. A logical construction is advisable.
2. The term must be lexically systematic. It must follow an existing lexical pattern and if the words are of foreign origin, a uniform transcription must be preserved.
3. The term must be conform to the general rules of word-formation and of the language which will also dictate the word order in compounds and phrases.
4. Terms should be capable of providing derivatives.
5. Terms should not be pleonastic (i.e. no redundant repetition, e.g. combining a foreign word with a native word having the same meaning).
6. Without sacrificing precision, terms should be concise and not contain unnecessary information.
7. There should be no synonyms whether absolute, relative or apparent.
8. Terms should not have morphological variants.
9. Terms should not have homonyms.
10. Terms should be monosemic.
11. The content of terms should be precise and not overlap in meaning with other terms.
12. The meaning of the term should be independent of context.

A compound term is defined by Sager, Dungworth and M^{c}Donald (1980:265) as "the combination of two or more words into a new syntagmatic unit with a new meaning independent of the constituent parts".

Compound terms usually follow the pattern where the first unit of the compound narrows down the semantic scope of the second unit. An example of this formation pattern is *flask tongs*, where *flask* indicates exactly which type of *tongs* is in question (tongs used exclusively or primarily for handling flasks in a laboratory), thereby more clearly defining its semantic scope. Although common in both general language and LSPs, the formation of compound terms in LSPs is more systematic.

Compounds are often created as the end product of converting frequently-occurring phrases into succinct fully-terminologized units such as “a board that houses integrated circuits” → *integrated circuit board*.

As a compound gains acceptance, it generally goes through three stages of maturity: it begins written as two separate elements, evolves to become a single hyphenated unit, then matures into a single word or term. Two recent examples of this occurrence are *electronic mail* → *e-mail* → *email*, and *off line* → *off-line* → *offline*. Although this is a common tendency for noun compounds, Sager, Dungworth and M^{c}Donald (1980) note that it is far from consistent in the English language. They concede, however, that phrasal compounds containing parts of speech such as adverbs, articles or prepositions are more often hyphenated than their nominal counterparts, for example *end-of-file function*.

Compounds are similar to simple lexical units in that they both may be combined with other units and can take affixes. Compounds may be composed of five, six or more elements, although such lengthy units are less common.

Compounds with three of more elements usually contain at least one two-element compound and follow patterns such as:

([A + B] + [C + D]) or ([A + B] + C)
two-neck distilling flask *heat-cure method*

Some two-element compounds only exist as a constituent of a larger lexical unit and cannot stand on their own, such as the compound *fixed displacement* which only appears as part of compounds formed with *pump*. This type of compound is more a hybrid between an adjective and an affix that is not placed in direct contact with the term nucleus than it is a compound term per se. This type of adjectival compound is more prevalent in LSPs where a higher level of precision is required (more specific determinants are necessary) than in general language.

Lengthy compounds composed of six or more units are generally made up of compound groupings that often use hyphenation to render them easier to understand. An example of such a term is *UNIX-based firewall management*. Such lengthy terms are frequently truncated because the context they are used in, whether it be a written or a spoken context, prevents any ambiguity from arising for the “destinataire” or receiver of the message. When truncating, speakers often omit the determinants and use only the hyperonymic nucleus, or simply leave out one of the determinants and retain the rest of the compound. In this example, the element in parentheses is frequently omitted in usage, but nonetheless remains an integral part of the lexical unit: *front-end (relational) database application*.

1.2.2.3. Perspective of Heribert Picht and Jennifer Draskau

As stated by Picht and Draskau (1985), new processes are often named using alternative means of term creation, such as terminologization, compounding existing lexical resources, derivation, conversion, borrowing from foreign languages and abbreviation, instead of having totally new designations coined for them. Picht and Draskau propose the following list as the primary methods for term formation for compound and simple terms. They note that the methods of coining neologisms for LSP and LGP are strikingly similar, but add that certain means of creation are more prevalent in LSP than in LGP and vice-versa. Furthermore, they advance that this list applies to all languages; however, the frequency with which each method is employed varies according to the type of language, for example Slavic languages (Russian, Polish, Czech) as opposed to Germanic languages (English, Swedish, Dutch).

Terminologization: where a lexical unit from LGP is conferred an additional (often metaphorical) meaning over and above its existing semantic field, and that this new meaning corresponds to a specific concept in a given subject field. The terminologization process does not in any way change the status or semantic field of the lexical unit in LGP. In some instances of terminologization, there is little change in the signifié from the LGP representation and its LSP counterpart. However, in other more metaphorical instances, the two signifiés can be appreciably different. An example of terminologization is the lexical unit *gate*, which is defined as a “movable part of a frame for closing an opening in a wall or fence. It turns on hinges or slides open and shut” in LGP (Dodds de Wolf, Gaelan *et al.*:640) and as “an electronic switch that is the elementary component of a digital circuit. It produces an electrical output signal that represents a binary 1 or 0 and is related to the states of one of more input signals by an operation of Boolean logic such as AND, OR, or NOT” (Microsoft:214) in LSP.

Terminologization is used when naming new concepts because either the new object resembles the LGP object physically, such as the computer hardware device called a *mouse* after the rodent similar in appearance, or because they perform a similar duty but in a different context, such as a mailbox for postal mail vs. a directory on a computer hard drive where electronic correspondence is delivered and stored — a *mailbox* for virtual letters. Another example of this second reason for terminologization is the *gate* example given above. Their purpose is similar; however, the environment in which they operate is different.

By using a lexical unit from LGP that all speakers of the language understand and whose concept they are familiar with to name an emerging concept, one or several characteristics of the LGP concept that was originally given this designation will be conferred on the new concept. This will form a type of semantic bridge between the two and consequently facilitate retention of the new concept’s designation by speakers, or possibly even elucidate certain aspects of the new concept’s purpose or usage for those who will be concerned by it.

Compounding existing lexical resources: Picht and Draskau (1985:108) define compounding as when several elements are grouped together in various combinations. They note that compounding takes widely varying forms depending on the language being treated. For the purposes of the present study, however, only compounding in English will be discussed. They go on to provide a partial list of compound term formations with examples in several languages, but fail to elaborate any further on them. An interesting point they note regarding compounds is that any of the other alternative means of term formation described here can also be coupled with compounding, and this double coverage leaves us with endless possibilities when creating new designations. For example, a compound may be borrowed from another language, new designations may be derived from already-existing compounds, compounds may be converted and compounds may even take the form of abbreviations. In short, it would be nearly impossible to exclude compounds from the term formation process.

Derivation: where existing lexical resources are altered through the use of derivational elements, more specifically, by employing prefixes, infixes and suffixes (see the footnote in Section 1.2.1.2 for further information regarding affixes). To what was stated by Sager *et al.*, Picht and Draskau add that the vast majority of LSP-specific derivational elements are particular to a specific subject field, such as *chloro-* and *phenyl-*, which signify “of or related to chlorine or phenyl” respectively, when added to an already-existing chemistry term. Moreover, while most derivational elements are used by only some languages or language families, others have taken on a somewhat international flavour and are deemed acceptable in the majority of languages. These extremely versatile derivational elements are often taken from Latin or Greek.

Conversion: where a shift in the part of speech or word class occurs to create a new instance of the lexical unit in a different word class. By creating such designations, the intrinsic qualities of the original term are seamlessly conveyed to the new formation. An example of such a conversion occurs when an adjective is converted to a noun and the quality conveyed by the adjective is immediately apparent, such as the adjective *scaleable* and its nominal form *scaleability*. New designations formed in this way are sometimes enhanced with derivational elements so that the word class is apparent, and also so that its formation pattern conforms to other designations in its class. An example of such an enhancement is the adjective, *kind*, and its nominal counterpart, *kindness*. This type of term formation includes a special subclass reserved for proper names that are used to denote common nouns, such as *Planck's constant* (used to determine a photon's energy) named after its discoverer, the German physicist Max Planck; and the chemical compound *curium*, first isolated by Glenn Seaborg and Albert Ghiorso, but named in honour of the French chemists, Pierre and Marie Curie.

Borrowing from foreign languages: where the designation for a concept is taken from another language as-is instead of being translated. Typically, this occurs when the concept itself is developed in a different country, meaning that both the concept and its linguistic expression are imported. Furthermore, the authors remark that the lapse of time required for the borrowing to become fully accepted into its new language depends on the extent to which it was adapted for integration into the new language. Picht and Draskau elaborate on three types or "degrees" of borrowing with examples.

In some cases, the term is adopted into its new language without any modification. This, of course, is more easily accomplished when the two languages involved possess inherent similarities, such as German and Dutch or Portuguese and Spanish. Two examples of this type of borrowing are the nouns *Download* and *Freeware* that were borrowed from English by German.

A more moderate approach to adopting new concepts without translating their designations is to adopt the foreign-language designation, then modify the spelling or form to a certain degree so that it appears more "naturalized" in its new linguistic surroundings. This is a common practice between German and English, two languages that share similar origins, but use different spelling conventions. A few examples of this type of borrowing between English and German are *downloaden* (in English *download* — the suffix *-en* was added to the infinitive form to give it a verb-like feel), *Chipsatz* (in English *chipset* — an uppercase letter was added because it is a noun and *Satz* is a translation for set) and *online-Hilfe* (in English *online help* — *online* was linked to *Hilfe* by a hyphen to follow the German trend of linking words together and an uppercase letter was added to *Hilfe* because it is a noun).

The least-conspicuous type of borrowing occurs when the term formation pattern remains intact when borrowed into the new language, but each element the borrowing contains is then individually translated verbatim into the new target language. Also known as a “calque”, this method of borrowing fashions a new designation that could justifiably be classified as a linguistic hybrid: such terms are moulded using the original language, yet composed exclusively of lexical units from the language by which it was adopted. This particular type of borrowing occurs most frequently between languages that come into close contact but do not share similar word formation patterns, such as English and French. A few examples of English terms that were calqued by French are *glace noire* (in English, *black ice*), *largeur de la bande* (in English, *bandwidth*) and *gratte-ciel* (in English, *sky-scraper*).

Abbreviation: where a new designation is formulated using the abbreviated form of one or several of its elements. This particular method of term formation is extremely common in LSP because terms tend to be longer than words in LGP, and simply omitting elements of a compound to make them shorter and faster to pronounce quickly could occasion vagueness and ambiguity between speakers and potentially lead to miscomprehension. Examples of this type of term formation are *Rh-factor* (abbreviated from *Rhesus blood group*), *ping* (abbreviated from *packet Internet groper*), *CPU-intensive proxy server* (abbreviated from *central processing unit*) and *adaptive RAID technique* (abbreviated from *random array of inexpensive disks*).

1.2.3. Types of compound terms found in the English language

For the purposes of our research, we examined two typologies of compound terms found in the English language to decide which one was better suited to our work. These typologies were found in Sager, Dungworth and M^{c}Donald (1980) and Selkirk (1982).

1.2.3.1. Perspective of Sager *et al.*

According to Sager, Dungworth and M^{c}Donald (1980:267), no analysis or description that could be deemed fully acceptable had been produced when their work was published, despite the numerous attempts that had been made to analyze compound word formation and describe compounding. The authors do state, however, that regardless of the method of analysis adopted, knowledge of both the context and the subject field are instrumental to successfully analyzing compounds. They continue that understanding a compound's structure hinges on correctly parsing it for the part of speech and function of each of its constituents. In conclusion, they indicate that interpreting compound terms is facilitated by the frequent presence of neoclassical or Romance words where the part of speech is evident.

With respect to the nucleus, there exist three main types of compounds:

Compound type	Example
Those that designate objects	restricted data
Those that designate properties	platform specific
Those that designate processes and procedures	URL accounting

Table 1-3 The three main types of compounds defined by Sager *et al.*

The determinant in these cases can add a higher level of precision to the nucleus by indicating a purpose, making the nucleus more exact, indicating how an operation is carried out, to which objects a procedure is applied, the time or place or other information relevant to the operation.

Compounds that designate objects constitute the largest group and also demonstrate the widest variety of relationships between the determinant and nucleus. Sager, Dungworth and M^{c}Donald (1980) elaborate on ten such relationships.

Relationship	Example
The determinant compares the nucleus to another object.	proxy server horseshoe crab
The nucleus' formation or composition is described by the determinant.	silicone chip star network
The determinant represents an inherent property of the new concept which is not inherent in the nucleus alone.	LAN backup spring-form pan
The intended and principal use of the nucleus is indicated by the determinant.	wood-burning stove error handler
The determinant is the item generally associated with the nucleus.	juice machine bus terminal
The instrument is the nucleus and the determinant is the object on which the nucleus operates.	Fortran compiler hair dryer
The determinant may indicate the method by which the nucleus functions.	hacksaw freeze-drying
The nucleus may be polysemic or insufficiently specific and the determinant supplies additional information to increase clarity.	hardware adapter login profile
The nucleus' identity is reinforced by the determinant (semiotic relationship).	air pump nail file
The determinant indicates the place where the nucleus takes place or is situated.	inter-city gang capillary hemorrhage

Table 1-4 The determinant-nucleus relationship according to Sager *et al.*

They concede that this list is not exhaustive, however.

Compounds that designate properties are often formed by the determinant specifying the concept to which the property term is related. The determinant in these cases are often adjectives instead of nouns, but are nonetheless considered compound terms for the purpose of our research. A few examples of this type of compound are as follows: *thrust force*, *thrust load*, etc.

Finally, compounds that designate processes and procedures are normally formed by the nominalization of a phrase containing either *of* or *by means of* where the subject, object or instrument of the corresponding verbal action is specified. An example of each appears below:

Nominalization type	Example
Subject	colour server data-warehouse
Object	data-processing mainframe configuration
Instrument	sandblasting centrifugal casting

Table 1-5 The Sager *et al.* typology of procedural compounds

Compound nouns designating processes are often created from deverbal nouns and converted verbs, for example *bacteria collection* and *network maintenance*. Conversely, compounds designating procedures/operations are created from verbal nouns, for example, *urban planning* and *stress testing*.

1.2.3.2. Perspective of Elisabeth Selkirk

Instead of trying to provide an exhaustive report on compounding in English (she states that this has already been done), Selkirk (1982) focuses her attention on what she believes to be the fundamental aspects of English-language compounding and how compound word structures are generated. She stresses the need for precise rules for compounding that adequately characterize the compounding possibilities offered by the English language.

Selkirk states that the elements of a compound may be either nouns, adjectives, verbs or prepositions. Moreover, compounds as units are either nouns, verbs or adjectives. Most compounds are “endocentric,” that is, they have a head or principal element, as opposed to being “exocentric” or having no head. Given the positioning of the head in the majority of English compounds, Selkirk argues that the word category can generally be deduced by analyzing the grammatical category of the rightmost element of the compound. Selkirk (1982:20) calls this postulate the “Right-Hand Head

Rule" and attributes it to morphology. This theory is supported by Williams (1981), who states that the head is defined according to the positioning of the elements in the compound instead of the relationship between the compound constituents. Selkirk notes that of the terms which have a head, only the vast majority are right-headed, such as *internal database architecture*. One group of exceptions to this rule is the left-headed compound formation type verb + particle, examples of which being *sit down* and *grow up*. In her revised model of the right-hand rule, Selkirk makes an allowance for inflectional affixes that appear after the head so that all situations in English are covered.

Concerning the structure of compounds, Selkirk (1982:14-15) provides us with a typology that is summarized in the table below:

Part of speech	Composition	Examples
Nouns	noun + noun adjective + noun preposition + noun verb + noun	bottleneck, dataglove real-time, sharpshooter downtime, afterburner runtime, supply-chain, password
Adjectives	noun + adjective adjective + adjective preposition + adjective	mainframe-specific, credit-hungry well-muscled, double-blind outboard, overactive
Verbs	preposition + verb	underachieve, upgrade

Table 1-6 Selkirk's compound structure typology

These "low-level" compound types can then in turn be combined to form more complex structures. On the whole, Selkirk considers compounding to be recursive, meaning that there is no theoretical limit to the number of modifiers that can be added within the framework of a prepositional phrase. Recursiveness can be either left-handed or right-handed in English. Selkirk calls the formations listed in Table 1-6 above to be "context-free word structure rules" and expresses them using the notation N → N N, which translates to "a noun is formed of a noun coupled with another noun".

To complement her list of context-free word structure rules, Selkirk proposes the following list of “context-free rewriting rules” that comprise the compounding formation grammar in the English language. She stressed that there are possible compound formations not covered in her rules, such as verb + verb and verb + [verb + noun] + verb, but adds that this is because English simply does not make use of these particular formations. In addition, she states that if her set of rules were relaxed enough to encompass all formations for all languages, it would be overly generalized and less effective.

$$N \rightarrow \left\{ \begin{matrix} N \\ A \\ P \end{matrix} \right\} N$$

Nouns

$$A \rightarrow \left\{ \begin{matrix} N \\ A \\ P \end{matrix} \right\} A$$

Adjectives

$$V \rightarrow PV^{-}$$

Verbs

Figure 1-3 Selkirk’s context-free rewriting rules

Selkirk concedes that N V and A V type verbs can exist (such as *to load-balance* or *to hot-walk*), but counters they are what are referred to as “pseudo-compound verbs” by Marchand (1969:58-69) because he argues that they are simply back-formations of already-existing nominal or adjectival compounds and would never be formed naturally on their own. These two compound verb formation types rely wholly on back formation; whereas the other types presuppose the existence of neither nouns nor adjectives. Consequently, she omitted N V and A V type verbs from her list of context-free rewriting rules.

1.2.4. Summary

After examining each author’s perspective on what constitutes a term, we have concluded that Dubuc’s definition of the term is better applied to general language (LGP) than to LSPs because it lacks a reference to the semantic aspects intrinsically bonded to the linguistic sign by means of the signifié. This element is highly important to specialty language because most technical objects named in LSPs have a single, specific task that cannot, whenever possible, be misidentified; whereas an item from general language is precisely that — general. Its meaning can be applied to a variety of situations and is subjective. We will consequently use the three-tiered definition proposed by Sager (1991) and supported by Picht and Draskau (1985) as a base for determining which lexical units in our corpus are terms.

When analyzing the structure of the compound terms in our corpus, we will use the theory of English-language compounding proposed by Selkirk (1982) because her so-called “simple context-free grammar for generating compound word structures” makes no reliance on semantic or extra-linguistic information and brings her approach closer to the concerns of automated term extraction. Furthermore, it is the most complete and best explained typology we found. We consider it to be more easily applied and less theoretical than the typology proposed by Sager *et al.* (1980). Semantic criteria, supported by morpho-syntactic criteria, were employed when scanning our collection of electronic texts to compile our list of manually-extracted terms.

1.3. Term extraction

1.3.1. Why automate term extraction?

According to Otman (1991), compiling a list of documentation to be scanned for terms, generating lists of candidate terms and then selecting the terminological units to retain is the most time-consuming (hence, costly) aspect of terminology processing. Furthermore, Otman (1991:60) raises the interesting point that the curve expressing the number of terms retained in a given subject field per number of pages scanned is logarithmic as opposed to linear or exponential. In other words, as the quantity of texts scanned for terms increases, the number of new terminological units that are retained by the human terminologist decreases until it becomes almost null[3].

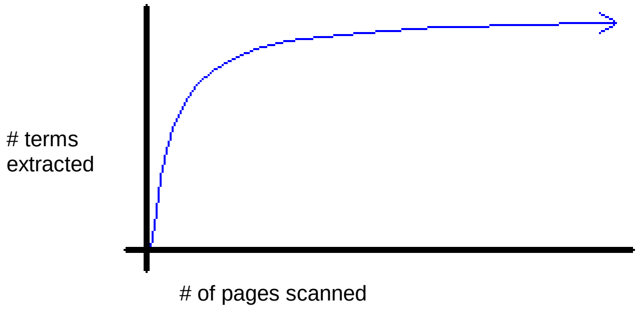


Figure 1-4 Logarithmic curve representing the number of terms extracted vs. the quantity of texts scanned

The net result of this curve is that the cost-effectiveness of giving a human terminologist this task becomes less and less clear as the quantity of texts scanned for terms increases. Otman suggests sharing the task between human and machine, with the human terminologist commencing the activity with the most terminology-intensive texts and leaving the documents with less potential to the automated term-extraction system. He emphasizes that turning to a computerized solution such as automated term extraction is only a secondary step in the process of collecting terminological units and constitutes a way of better using the human terminologist's time and expertise.

Not surprisingly, some subject fields are better suited to automated term extraction. In disciplines that are served by a somewhat open and loosely-standardized terminology, term extraction is hindered because the meaning of certain terms is fuzzy at best for the human terminologist, and it is next to impossible for the term extractor to decipher the vocabulary's use then determine its terminological status. Conversely, in other subject fields where a relatively stringent standardization approach has been implemented for an extended period of time, term extraction is facilitated.

[3] This phenomenon could be likened mathematically to an asymptote parallel to the x-axis.

1.3.1.1. Criteria for term extraction by humans

In order to define a set of custom criteria for which lexical units would be extracted as terms for the human list, we compared the authors' perspectives to create guidelines that are realistic to the constraints of the real world without becoming exceedingly vague. It is important at this point to mention that it is human criteria we are analyzing at this point, not how LOGOS or ATAO extract terms. This information appears in a different section of this work focusing on the respective software packages. Some criteria are common between humans and one or both of the software term-extraction packages; however, at this point we are focusing uniquely on the human perspective of which criteria justify extraction of a lexical combination in free-running text and consideration as a term.

Based on criteria elaborated in Sager *et al.* (1980:268), the following is an enumeration of elements that were considered as indicators to look for when scanning the corpus texts for term candidates to extract for inclusion on the human list. When we were unsure whether a term candidate should be retained, this list was consulted in a sequential fashion, i.e. giving priority to semantic criteria, to assist with the final decision. This orderly approach kept the "guesswork" down to a minimum and promoted consistency in our data because the same criteria were applied to all term candidates. It must be mentioned here, however, that human terminologists apply these criteria subconsciously, at least to a certain extent, whenever they analyze a term candidate's level of terminologization. The first few methods at the beginning of this list illustrate the semantic relationships that exist inside the formal syntactic structure. Other methods rely on morphological and/or syntactic criteria.

1. Is the compound a back-formation — a complex verb derived from a compound noun term?

    - *To burn-shine* (formed from *burn shining*: a method of shining leather by passing an oxyacetylene or similar flame over the area)

2. Is it possible to paraphrase the compound in such a way that the paraphrase can take the place of the compound in the sentence?

    - The worker used *welding studs* to join the pieces together.
      — Studs used for welding

3. Can the compound be interpreted as a shortened sentence independent of its context?

   - *Welding stud*
     — A stud used for welding

4. Is the lexical unit formed of an “object + function”?

   - *Queue controller* (a mechanism used to monitor and control queues)
   - *High-tension wire* (an electrical wire designed to withstand high-tension transmission)

5. Does the lexical unit exhibit conjunction or is it a portmanteaux word?

   - *Black and white TV* (both black and white)
   - *Contone* (a portmanteaux word for “continuous tone,” an image that displays continuously varying tonal values)

6. Does the lexical unit exhibit disjunction (an either/or relationship)?

   - *I/O device* (input / output device)

7. Is the lexical unit composed of a hyponym of an existing hyperonym?

   - *Multi-headed screwdriver* (*multi-headed* indicates exactly which type of screwdriver)

8. Does the lexical unit contain any affixes that render it more specific?

   - *Cool* vs. *coolant* (the suffix *-ant* indicates a noun whose purpose is the term’s nucleus)

1.3.2. Human vs. automated term-extraction methods

When a human terminologist extracts a lexical unit as a term, the degree of terminologization is based primarily on semantic and pragmatic criteria; whereas automated term-extraction systems have only just begun functioning at the linguistic analysis level. Consequently, the performance of an automated term-extraction program cannot be compared qualitatively to the work of a human terminologist. Lists of term candidates generated by extraction programs are precisely that — *possible* terms whose terminological status has not yet been verified — and must be carefully filtered by a human to eliminate non-terms. Term-extraction systems are intended to facilitate and accelerate the human terminologist's task of identifying terminological units, not assume full responsibility for the task on its own. Many of the difficulties commonly faced by automated term extraction are so fundamental that they are not even apparent to terminologists experienced in manual term extraction. On a more positive note, however, some automated term-extraction strategies appear to be better adapted to certain applications than others.

1.3.3. Automated term-extraction strategies

There are two main strategies for identifying complex terminological units in running text that are currently in use: one is based on a "linguistic analysis" of the text, and the other uses a "statistical approach". Some systems focus on one of these methods; whereas others attempt a hybrid method that incorporates aspects of both approaches in the identification strategy. For instance, a system based on linguistic analysis might filter its output list using frequency statistics, or a statistical program might incorporate a low-level morphosyntactic analysis as part of the term identification process. An overview of each approach, as well as a hybrid method, is given in Chapter 2 of this work.

1.3.4. Introduction to noise

By their very nature, lists of term candidates generated by both types of term-extraction systems tend to be much longer than those a human terminologist would compile. As a result of their limitations, term-extraction systems add word combinations to their list that would not be considered as terminological by a human, also called "non-terms". This type of error is classified as "noise“.

Noise can be caused by a number of factors, which vary depending on whether the term-extraction system being used employs linguistic analysis, statistical analysis or a mixture of both. The main causes for noise in linguistic-based automated term-extraction systems such as LOGOS and ATAO are listed below [L'Homme *et al.* (1996:306)]:

- Category ambiguity
- Matching pattern, but non-terminological combination
- Incorrect delimitation of terminological units
- Other (graphical issues, unknown type or morphological analysis applied by a system)

Category ambiguity represents a considerable challenge when attempting to recognize compound terms in English. Where appropriate, word forms are tagged in the system dictionary as belonging to more than one part of speech, and character strings that meet the criteria for being considered a compound unit may fall under several morphosyntactic categories. This form of noise occurs when a word form is tagged as belonging to more than one part of speech and the system cannot determine which one is correct in the local context. Lauriston (1993:150) provides us with the following three main types of category ambiguity that occur when identifying compound terms in English:

- Noun or adjective: *legacy application*
- Noun or verb: *figure*
- Verb or adjective or gerund: *kerning*

Certain occurrences of category ambiguity can be resolved through morphological analysis, thereby emphasizing the importance of lemmatizing and tagging input texts as part of the term identification process. “Lemmatization” or “stemming” is a procedure whereby separable endings such as plural indicators are stripped from extracted term candidates to reduce them to their canonical form. Only using terms' canonical form in the term candidate extraction lists generated by the system limits redundancy and consequently reduces their relative length. Lemmatization has the negative side effect, however, of potentially introducing ambiguity in the text if it is completed before tagging the word category. For example, *plums* with an S can only be considered a noun; however, after lemmatization, *plum* can either be interpreted as a noun or as an adjective.

A major difficulty encountered by term-extraction systems while lemmatizing is that no word category (nouns, adjectives, verbs, adverbs, etc.) has an exclusive right to a specific morphology. This means that several parts of speech, such as nouns and adjectives, can share similar or even identical morphemes, for example *parent* (noun) and *equivalent* (adjective) both share the same ending *-ent*. Conversely, some forms can belong to multiple categories; an example would be the lexical unit *firm* which may be used as either a noun, verb or adjective. Another example of a word that belongs to multiple categories is the lexical unit *round*, which may be used as a noun, adjective, verb, adverb and preposition. Otman estimates that, in French, approximately 25% of all lexical units in a given text may be classified as categorically ambiguous before lemmatization occurs. He adds that term-extraction systems require a syntactic analyzer that is able to eliminate category ambiguity in virtually all situations.

Otman (1991:70) illustrates that morphological analysis on its own is insufficient for term-extraction if it is not supported by syntactic analysis and interpretation of the immediate context to considerably decrease the number of ambiguities without necessarily reducing it to zero. Moderately ambiguous contexts can be clarified through syntactic analysis, coupled with a certain amount of contextual information. Otman provides us with the French example of “Il cassa un vase de Chine,” (C’est le *vase de Chine* qu’il a cassé) as compared to “Il rapporta un vase de Chine,” (C’est de Chine qu’il a rapporté le vase). Two examples that illustrate this same ambiguity in English are *white paper* (either a “detailed technical report summarizing a company’s product or service whose publication is intended to publicly and officially disseminate impartial information” or any sheet of paper that happens to be white, depending on the context) and *small disk* (a hard disk with a modest storage capacity used with several other small disks to form a RAID, as opposed to a *single large expensive disk* or any small-sized disk, depending on the context). Contexts such as these, however, are often too ambiguous to resolve without extra-linguistic information and must be left to the human terminologist. They must be disambiguated through comparison with the local context or by “prosody” (intonation and other characteristics of speech that can provide extra insight into the meaning behind words and sentences).

Furthermore, syntax alone is not sufficient when disambiguating cases of polysemy, anaphora or idiomatic expressions.

At the syntactic level, the initial difficulty when identifying term candidates in the input text consists in simply recognizing whether or not a given collection of words represents a lexical unit. Such identification is made more difficult by the fact that compound terms commonly share the same syntactic pattern as structures brought together coincidentally through sentence syntax. This source of noise is known as “structural ambiguity”. Identifying these non-terminological units that match term formation patterns as term candidates is another prevalent cause of noise. If a term-extraction system is designed to extract without exception every lexical combination in the input text that adheres to a given list of term formation patterns, it can in no way distinguish between genuine terminological units and non-terms that are formed using the same pattern. *Wrong choice* and *powerful feature* are two examples of non-terms extracted by ATAO that follow common term formation patterns, in this case ADJ + N. As shown by these two examples, differentiating between terminological collocations and free collocations is often unclear and relies on extra-linguistic criteria. As such, it is impossible to resolve such cases by morphosyntactic analysis alone. Recognizing lexical units is extremely difficult for existing automated term-extraction systems, and represents an area in which considerable advancement is still required.

Once the collection of words has been designated as a lexical unit, the system must decide whether it is a term or a non-term. Matters are further complicated for the term-extraction utility by the fact that there are no set syntactic rules for creating compound terms. The prevailing vagueness when searching for terminological units in running text (not quite knowing exactly what to look for or the form it will take, all the while contending with structural ambiguity) makes it reasonable for us to expect that the system will omit some compound terms and include some additional non-terms in with its list of candidates. As mentioned previously, current term-extraction systems can at best only propose a list of promising term candidates for subsequent filtering by a human terminologist. Dubuc (1992:29) and Sager (1991:61) both agree that there is not a clear distinction separating compound units and free phrases. Auger (1979:18) proposes that formal syntactic properties are insufficient for determining absolutely whether a phrase is “terminological” or “free”. Lauriston opines that either analysis at a higher level (i.e. semantic or pragmatic) or statistical evidence of a candidate term in the input text is required to proceed further with separating compound terms from freely-formed phrases.

Another source of noise, incorrect term delimitation, occurs when the system extracts a terminological unit that is in fact a part of a larger terminological group. Deciding where terms begin and end is not a task to be taken lightly, however, for it is a challenge to both the human terminologist and automated term-extraction systems alike [L’Homme (no date given:35)]. A basic example of such an error would be if a term-extraction program identified *Plug-and-Play network card* as a term, when in fact the entire term properly delimited is *SCSI Plug-and-Play network card*. The system is unaware that the head *network card* is modified by two separate adjectives and cuts the term after the first one, which is physically closer to it. Otman (1991:72) notes that term delimitation occurs once non-terms have been eliminated from the list of term-candidates. Some complex nominal groups are comprised of embedded subgroups that may or may not be considered independent terminological units. An example of such a grouping is the terminological unit found in Termium, *scalable transportable intelligence communications system*, which contains the following lexical subgroups:

1. *Scalable transportable intelligence communications*
2. *Scalable transportable intelligence*
3. *Transportable intelligence communications system*
4. *Transportable intelligence communications*
5. *Transportable intelligence*
6. *Intelligence communications system*
7. *Intelligence communications*
8. *Communications system*

Not all compound terms are this easy to process, however. A compound term of similar length, *Paasche current weighted import price index*, contains different lexical subgroups:

1. *Paasche current weighted import price*

2. *Paasche current*
3. *Current weighted import price index*
4. *Weighted import price index*
5. *Weighted import price*
6. *Import price index*
7. *Import price*
8. *Price index*

Of the above list of plausible term candidates, (2) and (3) would not logically be retained by the human terminologist. To eliminate non-terminological combinations such as these, the term-extraction system would employ a process of elimination based on comparing plausible term candidates with its list of possible matches to determine that the head of this compound is *price index*. Some of these recognition difficulties for automated term-extraction systems can be resolved by careful programming and enhancing the rules used by the system for term delimitation.

Other noise-producing factors can also arise, depending on the system's configuration for handling special situations during processing. When some systems encounter an unknown word form, for example, they automatically assign them a part of speech (in most cases, they are considered to be nouns). If the new word form is something other than a noun, noise may have been produced.

Special typographical characters and text attributes can also be a source of noise. Although this lowest level of analysis poses little or no problem to human terminologists (they may not even be aware that the analysis is taking place), there do exist recognition and disambiguation difficulties that the term-extraction software must overcome. Formatting and graphic difficulties can cause either noise or silence, depending on the situation. Software developers offer a partial solution to the problem of recognition by restricting the number of characters permitted by the system and implementing a translation table to convert less-frequent characters to a more common alternative, for example, translating the symbol *½* to *1/2*. The disadvantage of such an approach is that some semantic information can be lost through actions such as converting all input text to lowercase letters and stripping diacritical characters from a text, for example, translating the accents causes *résumé* to become *resume* (thereby potentially resulting in silence). Ambiguity at the graphical level generally stems from an individual character (most often a punctuation mark) lending itself to more than a single interpretation. For example, a hyphen can be interpreted as a bullet character in an enumeration, a device for indicating an aside in a longer sentence, a link between two like elements or as a type of linguistic adhesive connecting two lexical units that could otherwise not be joined.

1.3.5. Introduction to silence

When a term-extraction system fails to identify terminological units in the input text, the error is classified as “silence”. In general, automated term-extraction programs are designed in such a way as to reduce silence to a minimum. It is even preferable to increase the percentage of noise to help keep silence as low as possible because it is far easier for the human terminologist to reject inappropriate strings from the list of term candidates than it is to manually scan for omitted terms. This situation is compounded in circumstances where the volume of text to process is high.

Otman (1991:66) notes that the amount of noise generated by a term-extraction system is in direct relation with the number of terminological patterns it attempts to extract. Conversely, he mentions that silence can be appreciably increased if rules to exclude certain lexical formations from the list of term candidates are added to the system (1991:73). Such a rule might state that groups of words starting with an adverb such as *very* or *ultra* are to be excluded from the list of potential terms. However, this rule would overlook the terms *Very High Frequency* and *Ultra High Frequency* and simply add *High Frequency* to the list of term candidates, which is an instance of incorrect term delimitation (noise). A terminological unit may be created using virtually any possible formation; so this type of elimination rule should be used with extreme caution.

The causes of silence differ, depending on whether the method implemented in the system being evaluated is linguistic or statistical. In linguistic systems such as LOGOS and ATAO, the main causes for silence are as follows:

- Pattern not implemented in the system
- New word form
- Absence of a given word tagging
- System configuration

A terminological unit may be formed according to a pattern that is not implemented in the term-extraction system. As such, the extraction system does not realize that it has happened upon a term and passes over it undetected. Developers of such software must try to strike a delicate balance between including the highest number of term formation patterns possible and keeping noise to a minimum. As mentioned above, every new term formation pattern added to the system is accompanied by a varying quantity of noise.

On a more lexical level, if an element of a compound term is not found in the system dictionary, silence will result because the system will not be able to parse the text adequately for lack of linguistic information. Furthermore, if a word tagging is absent from the dictionary used by the term-extraction utility, for example, a word is tagged as a noun but not as a verb, the system will not be aware of all the different ways a given word can be used and therefore potentially omit a certain percentage of occurrences of the word from the list of term candidates.

Some configuration settings in the term-extraction utility can result in noise, as was stated earlier, and others may result in silence. Some user settings can entail higher incidences of silence, such as the minimum frequency of occurrences for a terminological unit to be added to the list of term candidates and how special graphical characters like hyphens and uppercase letters are to be processed by the system.

1.4. Summary

In this chapter, we took a closer look at the main types of automated term extraction; our findings and impressions are summarized below.

First and foremost, we believe that there is much more to automated term extraction than meets the eye; it is far more complicated than we originally estimated. This could explain in part why there is no commercially-available term-extraction system that boasts 100% accuracy when scanning electronic documents for terms. Lauriston (1994:156) supports our observation by adding that, “There is today no commercial system available for full automatic term recognition. Existing systems are either experimental or semi-automatic”.

Secondly, the criteria for humans when extracting terminological units differ from those used by either linguistic- or statistical-based methods. The errors made by machines consequently differ from those made by humans (overlooking a term in the text, lack of subject matter knowledge or experience). Although they take the factors used by machines into consideration to a certain extent, humans have the luxury of extra-linguistic information when choosing or delimiting terms. More importantly, however, they can acquire a “linguistic-cohesiveness hunch” or intuition that says that a given series of lexical units “belong together” and that their occurrence alongside one another in free-running text is more than just a chance happening, i.e. whether a phrase is “terminological” or “free”. As stated by L”Homme *et al.* (1996:293) “Term identification [by humans] relies chiefly on knowledge and experience”.

Although great advances in this field have already been made, continued research efforts in the area of artificial intelligence will be necessary to bring the effectiveness and accuracy of automated term extraction to a level where it is used more extensively by institutions and companies who wish to improve quality of their terminological resources.

In the next chapter, we turn our attention to the more tangible side of the world of automated term extraction and investigate the various systems that are in use today. Although our focus is primarily on the two utilities used as part of this study, LOGOS and ATAO, we will also briefly examine other, similar term-extraction systems.

2. EXISTING AUTOMATED TERM-EXTRACTION STRATEGIES

As discussed in Section 1.1.2, the need to coin appropriate designations for emerging concepts is currently experiencing a surge never seen before, due in part to advances in science and high technology’s exponential growth. Some of these same technologies are in turn being applied to facilitating terminology extraction and management. In addition to automated term-extraction utilities, translators, terminologists and other language specialists now have a wide range of recent computerized tools to assist them with their work: online linguistic databanks, concordance software, machine translation systems and translation memory systems are only some of the aids that propose to lighten linguists’ tasks and complement their training, experience and skill. Continued research initiatives in the field of terminology automation will hopefully bring researchers to a heightened understanding of what exactly constitutes a term and increase the level of effectiveness of terminology software and hardware devices.

Focusing once again on automated term-extraction utilities, it must be stressed that no commercial system on the market today is capable of performing fully-automated term extraction[4]. Current systems are either experimental or semi-automated, with LOGOS and ATAO being of the latter variety. More precisely, ”semi-automated” term-extraction utilities identify a large number of ”term candidates” from the text submitted to it which then must be filtered by a human terminologist.

In this Chapter, we will describe the linguistic and statistical approaches to automated term extraction, including the strong points and shortcomings of each. We will also look briefly at a more recent hybrid approach that draws its strength from a

[4] This observation was made by Lauriston (1993:156) based on a statement by Ananiadou (1988:63 ff).

combination of the linguistic and statistical approaches. Although our investigation will extend beyond the LOGOS and ATAO systems to include an overview of Nomino and TermCruncher (two linguistic term-extraction programs), we will centre our focus primarily on a detailed discussion of many aspects of LOGOS and ATAO because they are the two being evaluated and compared later on in this research project.

2.1. The linguistic approach to term extraction

The linguistic approach attempts to identify complex terminological units by matching word combinations in the input text against the system's own list of specific patterns for compound term formation. Linguistic systems scan the input text, count the elements they locate and compare strings of characters against a list of preset formats and patterns, i.e. the term formation patterns discussed at length in Chapter 1 of this work. They cannot, however, comprehend the input text at either the semantic level or the morphological level. This is why term-extraction systems absolutely require linguistic descriptions of which lexical patterns to extract that are expressed in such a way that the machine can easily process them. The system's compound term combination patterns with which the lexical units in the input text are parsed for comparison are really descriptions of typical noun phrases and other term formations expressed as rules. The long list of possible compound term formation patterns is not cast in stone, however, and varies to a certain extent from one system to the next.

The drawback of establishing a vast array of lexical patterns for the term-extraction system to check for is that, although such non-specific models will cover the vast majority of the terminological units contained in an input text, some units will not be identified by the system because they do not adhere to any of the system patterns (see Section 1.3.5 for more information on silence). Pragmatically speaking, however, it is inconceivable to attempt to multiply exponentially the number of models used by the system in the hopes of identifying every possible term in the input text because the number of possible patterns and the number of elements a terminological unit may contain are limited only by human comprehension and usage. Moreover, for every model implemented in the system, there exist word sequences that match perfectly but are not terminological. These non-terms increase the human terminologist's workload by lengthening the system's output list considerably (through the added noise[5]). Two examples of non-terms extracted by ATAO that fit a terminological pattern are *widespread skills availability* (terminological pattern ADJ + [N + N]) and *capacity and data rate* (terminological pattern [N + CONJ + [N + N]]).

On a more positive note, equipping the term-extraction system with patterns for potential terms offers the advantage of being able to extract neologisms and terms in the input text that do not appear in the system dictionary. Otman (1991:68) states that the capacity of a term-extraction system to extract neologisms is how to measure its potential effectiveness because such systems must go beyond simply relying on the contents of their built-in electronic lexicons to identify terms. Term-extraction systems that rely heavily or entirely on the contents of a system dictionary to identify terms are less effective because they face the double handicap of being perpetually incomplete and by far too colossal in size to be practical or consulted quickly during term extraction. Besides, the primary purpose behind term-extraction systems is to enrich terminological databases. If they cannot identify terms other than those which have previously been identified, little progress is achieved by the system. They would appear to offer little more than the spell checking capabilities of an advanced word-processor.

For successful term extraction by a linguistic system, two steps must be achieved: “recognition” of the units to be analyzed and “disambiguation” or understanding what the author is trying to communicate by using them. The level of difficulty represented by recognition and disambiguation for the term-extraction software varies relative to the complexity of the input text. Optimal performance by a linguistic system also requires information on the part of speech for each lexical unit in the input text. Part of speech information can be added to the text in one of two ways: either the term-extraction system includes a highly-descriptive dictionary containing all lexical units

[5] A more detailed discussion of noise is found in Section 1.3.4 of this work.

along with usage information, such as the part of speech, or a third-party utility “tags” the text prior to processing by the system[6].

Early term-extraction systems used pattern matching as the sole means of identifying term candidates. This strategy proved to be unsatisfactory because it did not take into account the ambiguity caused by lexical units that belong to more than one part of speech. More recent systems, however, tend to incorporate at least a basic syntactical analysis of the input text to counter the difficulties raised by ambiguous forms. The level of syntactical analysis depends on the system, ranging from simply analyzing the immediate context in which the ambiguous form appears to a complete parsing of the sentence. Unfortunately, however, even the most advanced syntactical analysis systems are still struggling with the problems caused by ambiguous word forms.

An alternative linguistic approach to automated term extraction is proposed by Bourigault (1993) and utilized by the LEXTER system (also designed by Bourigault). Instead of attempting to identify term candidates through typical term formation patterns, i.e. the term formation patterns discussed at length in Chapter 1 of this work, LEXTER uses the process of elimination to discard all items that clearly cannot be considered for terminological status (ex: pronouns, conjugated verbs, etc.), coupled with syntactical analysis. For more information on the LEXTER system, refer to L’Homme (1996) or Lauriston (1994).

2.1.1. Shortcomings of the linguistic approach

The linguistic approach is, by its very nature, extremely language dependent. Consequently, extension to each additional language involves an intensive program overhaul and redesign. Furthermore, the linguistic approach is not very effective when

[6] “Tagging” a text is a procedure based primarily on morphological analysis, whereby a marker indicating the part of speech is assigned to all lexical units it contains. The tagging process facilitates term identification. An example of tagging is *The/def.art. server/n is/v currently/adv unavailable/adj*

it comes to extracting simple terminological units. This is understandable because purely linguistic systems cannot discern between new single unit terms and, for example, a known lexical unit that simply contains a keying error or spelling mistake. The system is also unaware of the single term's part of speech; so it is left no choice but to skip over the unknown unit.

Another drawback of linguistic term-extraction systems is that not all noun phrases are compound terms, even noun phrases that perfectly match the preset term formation patterns used by the system can be non-terminological. Lastly, some terminological units may be formed using patterns that are not entered in the extraction system; so they are not identified as term candidates during processing.

2.1.2. Nomino

Formally known as TERMINO, Nomino was developed at the Centre ATO (Analyse de Texte par Ordinateur), a computational linguistics research centre affiliated with the Université du Québec à Montréal. According to Lauriston (1993:156), Nomino is two generations more advanced than systems that identify terms using only lexical-level categorization, which is based primarily on morphological analysis. It goes beyond simply tagging the input text by applying a specialized morphosyntactic approach.

Designed to process French-language texts only, Nomino's term-extraction system is made up of three parts:

1. A **pre-processing module** that prepares documents for extraction by separating the text into words and sentences, then identifying proper nouns. These preliminary measures help the program yield more accurate results when extracting.

2. A **morphosyntactic analyzer** which handles categorization and syntactic difficulties occurring at the lexical level.

3. A **term-record utility** that allows the human terminologist to use the output from Nomino to generate records for newly-extracted items.

Nomino's morphosyntactic analyzer contains three subcomponents: a morphological analyzer, a parser and a "synapsy" detector[7].

The morphological analyzer's role is to tag, then lemmatize the input text to facilitate further processing. Instead of consulting an immense part-of-speech lexicon, Nomino uses lists of affixes and exceptions when tagging. This novel approach means that unusual words that would not otherwise figure in the system dictionary can be treated, and also that the issue of the system dictionary being voluminous and perpetually incomplete no longer applies. In cases of ambiguity, the morphological analyzer presents all possible taggings to the parser.

The parser then receives the text from the morphological analyzer as a series of tagged lexical entries. Nomino's parser was designed for unrestricted text and does not abort processing even if a full parse of the sentence is not achieved. The parser's main responsibility is to resolve any remaining instances of lexical ambiguity still present in the input text.

The synapsy detector is the component of Nomino that actually identifies the term candidates in the input text. Within the synapsy detector itself are two sub-elements: a synapsy builder and a synapsy comparator. The synapsy builder locates and identifies synapsies in a five-step process of syntactical analysis outlined in Otman (1991:83-84). Lastly, the synapsy comparator attempts to filter out the most obvious instances of noise from the list of term candidates.

[7] A term coined by Benveniste in 1966, a "synapsy" is defined by David and Plante (1990:145) as a complex lexical unit that constitutes the nucleus of a noun phrase.

Both Otman (1991) and Lauriston (1993) have published quantitative assessments of Nomino’s performance when extracting terms. Otman reported that Nomino version 1.0 recognized 70% of terms in running text and added that some performance improvements had been made in version 1.1 of the software program.

Lauriston’s results for Nomino version 1.2.3 were not always as positive, however. Using two standards, one tolerant and one rigorous, Lauriston recorded the following statistics. With the tolerant standard, 74% of terms were detected and 26% went undetected (thereby resulting in silence). The tolerant standard also produced a 28% incidence of noise. Using the rigorous standard, 51% of terms were recognized. Noise in this case was 52% and silence reached approximately 50%.

Lauriston listed acronyms, capitalized LGP nouns and how the terms were presented in the input text [for example, terms interrupted by coordinate conjunctions (*menu or button bar*), slashes (*cartridge loading/cleaning*) or punctuation marks (*"plug-in", extendible architecture*)] as the three principal causes for Nomino to fail, that is, the three main silence-producing factors. Noise, on the other hand, was generally caused by either an incorrect parse resulting from category ambiguity or incorrect term delimitation (syntactic noise), or non-terminological LGP combinations being extracted as term candidates.

2.1.3. Term Cruncher

Produced by the translation firm CLC ltée, Term Cruncher is a software package that scans an English-language text, extracts the collocations[8] it contains, then prepares a number of term candidate reports sorted by frequency or alphabetical order. In addition, this term-extraction program has the interesting feature of being equipped

[8] For the creators of Term Cruncher, a “collocation” is defined as “...un mot ou suite de mots répétés au moins une fois dans un document. À proprement parler, un terme peut être une collocation, mais l’inverse n’est pas nécessairement vrai” Taken from Normand (1993:29).

with the ability to automatically search a term databank selected by the user for equivalents to the collocations it extracts, and presenting the corresponding pairs of collocations together in a report format. Term Cruncher appears to be designed to scan for repetitive non-terminological expressions, as well as terms. This might be because it was developed by a translation firm to assist translators with translating whole documents and not to facilitate enriching a linguistic databank such as Termium or the BTQ.

When Term Cruncher has completed processing a text, seven different reports are generated. They are described below:

Report name	Description
TC1	All the words the submitted text contains. Collocations are sorted alphabetically and accompanied by their frequency rating.
TC2	The contents of the TC1 report after the transparent words have been removed. Collocations are sorted in descending order by their frequency rating.
TC3	All the collocations the submitted text contains with a frequency rating of two or more.
TC4	All the collocations the submitted text contains sorted by the number of lexical elements they comprise. The frequency rating is also given.
TC5	All the collocations the submitted text contains sorted in descending order by their frequency rating.
TC6	All the collocations in the submitted text for which Term Cruncher was able to find an equivalent in the linguistic databank selected by the user, accompanied by the target-language equivalent for each. Collocations for which no official match in the linguistic databank was found, but Term Cruncher was able to propose an approximate translation or “fuzzy match”, also appear here.
TC7	All the collocations in the submitted text for which Term Cruncher was unable to propose a fuzzy match or a translation from the user-defined linguistic databank.

Table 2-1 Post-processing reports generated by Term Cruncher

The following observations were made in an article summarizing an evaluation of Term Cruncher by Normand (1993:29). A 3000-word text containing 80 terms was submitted to Term Cruncher for processing. The output reports generated by Term Cruncher revealed that a total of 193 collocations were extracted, and approximately 20 collocations were left undetected in the input text.

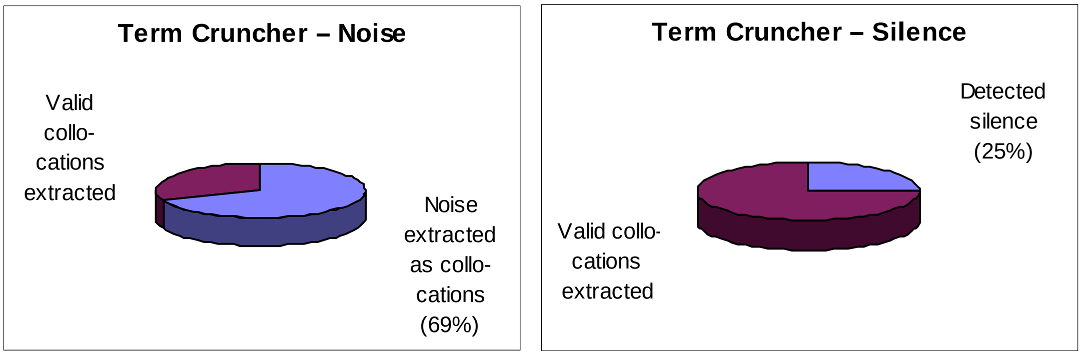


Figure 2-1 Term Cruncher's performance

The unusually high incidence of silence is explained by the author as being caused in part by the small size of the sample text submitted to Term Cruncher for processing. Many of the collocations that went undetected by Term Cruncher only appeared once in the input text. The author advances that, had the sample text been longer, the frequencies would have been higher and more collocations might have been extracted.

Based on our limited observations of this automated term-extraction package, Term Cruncher appears by and large to target translators as their main focus instead of lexicographers or terminologists. It extracts frequently-occurring LGP expressions as readily as it does terminological units and provides an automatic lookup in the user's selected termbank to facilitate translation. Furthermore, Term Cruncher places greater emphasis on a collocation's frequency in the input text than do the other automated term-extraction programs.

2.1.4. LOGOS

A product by the Logos Corporation of Germany, the LOGOS TranslationControlCenter is an integrated commercial machine translation solution that may be customized to suit individual or corporate needs. The LOGOS TranslationControlCenter is first and foremost a machine translation environment, not a dedicated term-extraction utility. It contains a rich suite of language technology and translation-related tools, one of which being term extraction. Term-extraction functionality was added to LOGOS in order to enrich its system dictionaries and hence improve the overall quality and accuracy of its translation output.

Several translation techniques are currently used by commercialized machine translation systems or by those either still under study or in the prototype phase. Of the many techniques available, LOGOS uses the “transfer approach,” a second-generation advanced core technology where an association is made in the system dictionary between the same signifiant in the source language and the target language for a given signifié, and the system generates its translations by transferring the lexical units in the source text to the corresponding target language unit indicated in the system dictionary. More specifically, LOGOS translates by parsing the lexical units and phrasemes contained in the source text in order to determine the part of speech and function of each element in the sentence. Once this has been completed to the fullest extent possible, the system then consults its dictionary for the equivalent associated with each target language entry, taking into consideration the part of speech and function in the sentence to avoid category ambiguity; for example, confusing a verb with its nominal form because they share the same spelling, such as *format* (the verb *to format*) and *format* (an LGP noun).

Configuring LOGOS to produce a list of unknown words, i.e. those that do not appear in the system dictionary, is simple. An option button in the Preferences tab of the LogosClient interface is used to enable or disable producing this list. In the same tab, it is also possible to specify whether or not found terms are to be included on the term candidate report, which parts of speech to extract for the list, as well as the thresholds for the minimum number of occurrences before a lexical unit appears in the list. A screen capture of this tab of the LogosClient interface appears in Appendix IV of this work.

To complement and support its main function of machine translation, the LOGOS TranslationControlCenter offers a number of linguistic and translation-oriented tools that help to accelerate some of the tedious or time-consuming tasks the translator must carry out. The main system components that comprise the TranslationControlCenter are described below:

> **LogosServer:** the machine translation engine that actually completes all processing of submitted texts, including the transfer from the source to the target language(s). LogosServer extracts the unknown terms the texts contain and generates a report for the human terminologist to verify. We did not have direct access to this component as part of our study.
>
> **LogosClient:** Web-enabled and written in Java for cross-platform capabilities, the LogosClient has a tabbed graphical user interface for submitting texts to LogosServer for processing. We used LogosClient to submit our collection of texts to LogosServer for term extraction. Application screen shots, a description of how the system is used, as well as our general impression are provided in Appendix IV of this work.
>
> **Alex, the Automatic LEXicographer:** a wizard-style utility that creates customized subject-matter dictionaries. The **Terminology Verification** utility that extracts term candidates for importation to the system dictionary is a subcomponent of Alex. More about Alex is explained in Section 2.1.4.2.
>
> **LOGOS Dictionary:** the multi-part translation dictionary that contains the customized subject-matter dictionaries created via Alex, as well as words from LGP, including grammatical units such as conjunctions and definite articles. More about the LOGOS dictionary is explained in Section 2.1.4.1.
>
> **LTM:** is a translation memory system that reuses previously translated and post-edited sentences to save time and ensure consistency amongst translated documents. LTM scans the inputted text on a sentence-by-sentence basis for similar sentences in the translation memory database and proposes identical or similar matches for approval by the user. If no match in the translation memory database is found, the sentence is translated outright by

> LogosServer and inserted in the machine output for post-editing. Every time LTM is used, the new post-edited sentences are added to the LTM database to increase efficiency.

The TranslationControlCenter ensures formatting is preserved while translating into the target language(s). It supports a variety of common file formats, including SGML, RTF, HTML and plain text files. Furthermore, multiple documents may be submitted at the same time for batch processing.

2.1.4.1. The LOGOS dictionary

When scanning a submitted text for term candidates, LOGOS consults its system dictionary first to ensure that the list of term candidates to be filtered by the human terminologist does not contain terms that already figure in its dictionary. The following is a detailed look at this component of the TranslationControlCenter.

The LOGOS dictionary is in fact three separate, but closely-linked dictionaries that include all lexical units a submitted text might contain. A segmented, three-tiered dictionary architecture is preferable over a single, immense dictionary because it allows the system to scan the text in multiple passes, quickly eliminating the basic simple grammatical words before attempting to process the more complex lexical units the text contains. Moreover, this approach accelerates the system performance because it is smaller for the system to maintain and handle. The basic dictionary eliminates a significant percentage of easily-translated basic grammatical elements and assists when parsing the other, more complicated parts of the sentence. A diagram illustrating the relationship between the components of LOGOS' dictionary is given in Figure 2-2.

Separating the second and third passes by the dictionary makes it possible to distinguish between the dictionary entries made by the user (via Alex) and those that were shipped with the system at the time of purchase. As such, the system can choose between the two and apply the entry from the appropriate dictionary, as indicated by the user preferences, when processing the input text. The same term may quite conceivably be entered in both the LOGOS Company Dictionary and one or several Client Dictionaries, each with different subject fields, company codes and/or equivalents. Using the subject field and/or company code as a guide, LOGOS must be able to choose between all available equivalents so that it selects the best translation possible and not simply the first match it encounters in its dictionary.

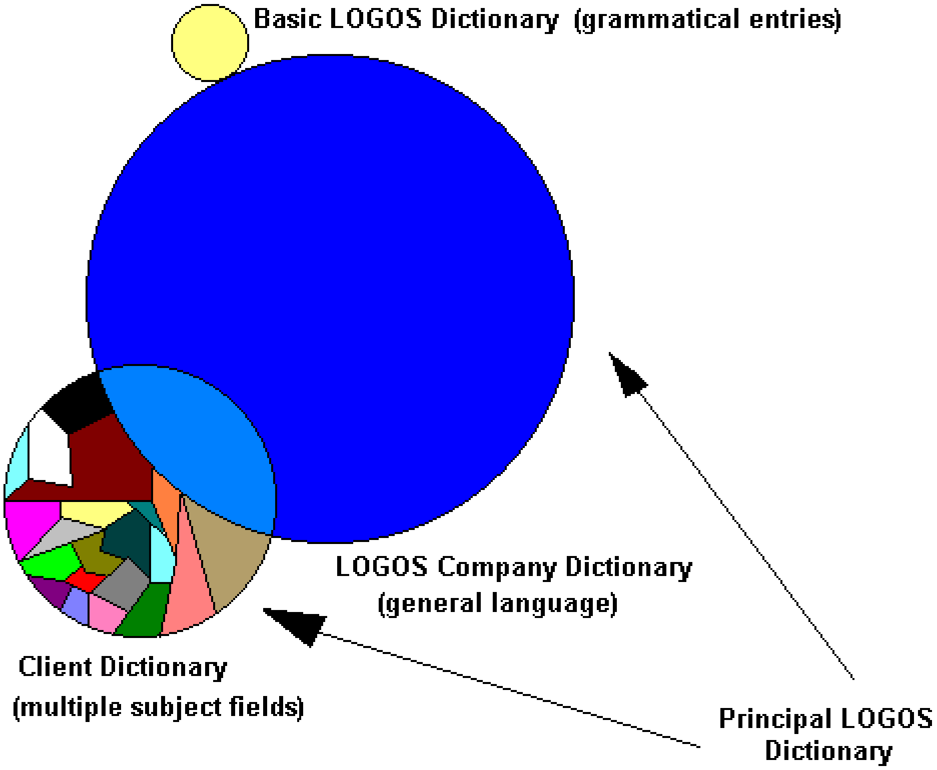


Figure 2-2 LOGOS' dictionaries

The "basic LOGOS dictionary" is a read-only grouping of a few hundred primarily grammatical words. This dictionary is consulted first when texts are submitted to

LOGOS for translation and consequently saves processing time by taking care of fundamental words in the source language text.

The principal LOGOS dictionary is made up of two parts: the "LOGOS Company Dictionary" and the "Client Dictionary”. The LOGOS Company Dictionary contains many thousands of entries and is in a constant state of evolution. Most of them are from LGP, and none are assigned a company code because they are packaged with the software and not added by the user for a specific subject field. Although this dictionary's entries are set as read-only, i.e. they cannot be modified by the user, they may be enabled, disabled and re-enabled at the user level as needed. The entries of the "Client Dictionary," on the other hand, are wholly created and maintained by the user. This dictionary contains all entries for all subject fields inputted by all users of the same LOGOS system. Entries in this dictionary are assigned subject field codes to simplify use by the system and are maintained via the ALEX interface described below.

Although all dictionary entries contain grammatical, semantic and translation information in the form of numeric codes, they are listed in canonical form, leaving the application to make the grammatical agreements and inflections in the target language text. Depending on the sub-dictionary in which they appear, these numeric codes are assigned by either the LOGOS Corporation or the user by means of the Alex interface.

Each entry in the dictionary is assigned a subject field code and a company code to ensure that each company's specific terminology choices are respected. Such markers are necessary because some companies have preferences for certain equivalents over others when translating the same term, such as opting for the equivalent *customer* instead of *client* when translating the French term *client* in to English. Furthermore, instances of homonymy can create the situation where a company might choose to refer to the entities with whom it conducts business as *customers*, but would refer to *client software* (as opposed to *server software*) and not to *customer software* in a computer-related text.

2.1.4.2. ALEX, the customizable system dictionary

ALEX is the interface between the user and the dictionary used by LogosServer to process texts for translation and/or term extraction. When using ALEX to add or modify terms in the system's dictionary, ALEX's wizard interface asks the user questions about the linguistic nature of the entry: semantic characteristics, usage information, irregularities, part of speech, syntax, etc., then ALEX encodes this information for processing in a numeric format understood by LogosServer. Through ALEX, dictionary entries or rules for analyzing the source text can be added, modified or deleted from the system. A reporting function is also available, through which the contents of the client dictionary can be viewed and/or sorted by a company code, subject field code or part of speech.

A subcomponent of ALEX, the "Terminology Verification" utility searches the input text for lexical units that are unknown to the system and proposes a list of term candidates for inclusion in the system dictionary. This type of analysis enables an input text to be cleaned of spelling errors and variations, and assists the user when protecting the proper names and abbreviations (if appropriate) in the source text from being translated before submission to LogosServer for processing. Furthermore, scanning the source text with LOGOS' Terminology Verification utility can sometimes reveal terminological inconsistencies within the same text and reduce post-editing time because entries on the term candidate report are accompanied by their frequency in the submitted text.

This feature can be used either generally or in a restrictive manner to generate unknown terms reports. When used generally, the system will look under all company and subject field codes to ensure the lexical unit in question does not appear anywhere in the system dictionary before adding it to the term candidate report. Conversely, the restrictive method has the system limit its research to the company and subject field codes listed in the current profile. It is also possible to have the system only search for certain parts of speech, i.e. only nouns and verbs. This method of term scanning helps the user determine which codes to use when submitting the source text to the system because the report results will indicate which subject matter dictionaries are most complete and best suited for processing the input text.

This is the aspect of the LOGOS system that is the most pertinent to our research. We used this utility to generate the list of candidate terms that were quantified, evaluated and compared in our study. Sample output from the LOGOS Terminology Verification utility is presented and explained in detail in Appendix V of this work.

2.1.5. ATAO

Produced by the Montreal-based firm, TRADUCTIX Inc., ATAO is a "machine pre-translation environment" (MPT) that generates partial translations of inputted source texts for completion by the human translator (Bédard:1992). MPT does not attempt a full translation, leaving this task to the human translator. The resulting mixed-language text can be edited on-screen by the translator with efficiency comparable to conventional MT post-editing. Despite the unusual appearance of the output text, this partial solution does offer several advantages, such as terminological consistency throughout a text and reduced incidence of omissions.

ATAO features two operational modes: "Full MPT" and "Light MPT”. In Full MPT mode, all confirmed words in the text are translated, including general vocabulary, and basic grammatical agreements are made in the French output text to assist the translator with the post-editing process. Light MPT, on the other hand, only translates any technical terms the text contains, then marks them in bold for consultation either on-screen or in printed form. This mode is similar to an automatic dictionary lookup and targets translators who dictate their work.

In addition to the MPT environment, ATAO also includes several utilities. ATAO attempts to assist the translator throughout the translation process with components to accelerate both the preparatory measures and post-processing. Each of ATAO’s utilities are listed below, followed by their French equivalent in parentheses.

Pre-editing module (*Prééditeur*): pre-editing includes standardizing spelling and terminology throughout the source document.

Term search module (*Dépouilleur*): presented as a report, the term search module analyzes the source text then extracts repetitive word strings and sorts them by head word. These term candidates can in turn be used as a starting point for the creation of a pre-translation dictionary or glossary. An example of a term search module report may be viewed in Appendix III at the end of this work.

KWIC (Key Word In Context) tool (*Concordancier*): being able to view multiple instances of a given term candidate in context is a crucial part of determining its terminological status. The KWIC tool presents all instances of the term candidate found in the submitted text in their immediate context so as to allow the translator to evaluate its terminological status in its natural surroundings.

Reference dictionary (*Dictionnaire de référence*): ATAO includes a basic bilingual dictionary covering general vocabulary which is used in the Full MPT mode.

Pre-translation dictionary (*Dictionnaire de prétraduction*): this customizable dictionary can accommodate entries measuring up to 30 words in length, as well as support variables and formatting codes.

Given that the focus of the present work is term extraction and not pre-translation or machine translation per se, our exploration of the ATAO program is mostly limited to these utilities and excludes the pre-translational (MPT) functionalities of the software program.

2.1.5.1. ATAO’s term search module

It must be stressed that *pre-translation* is the primary focus of the ATAO software package, and not term extraction. ATAO was enabled with term-extraction functionality in order to assist the user with enriching the pre-translation dictionary. The term search functionality was included in ATAO to assist with updating the system’s pre-translation dictionary.

ATAO can only process unformatted text documents with the *.TXT file extension. Texts to be scanned cannot be submitted in batch mode, only individually. It is recommended in the user manual, however, to always concatenate all texts from the same subject field before submission to the term search utility because this gives a better overall view of the subject field's terminology. The manual adds that the greater the number of texts scanned, the easier it is for ATAO to identify potential terms because of the higher rate of repetition.

There are five user-definable settings for submitting texts to ATAO's term search module: simple term extraction, word string extraction, thresholding, purging terms on the output list that are already present in the project or reference dictionary, and terminological comparison with an existing reference text. Each setting may be enabled or disabled as needed.

In a nutshell, the term search module is designed to research the input text for repetitive strings of words that could potentially become dictionary entries. After the initial scan of the document, ATAO compares its output with the contents of the pre-translation dictionary to avoid entering the same term in its dictionary twice. Double-checking for redundancy also reduces the length of the list of term candidates the terminologist must filter. When the scanning is completed, ATAO provides separate lists for single words and extracted word strings. The list of word strings includes both lexical units that represent potential terms and phrasal units that are either full sentences or segments thereof. Refer to Section 2.1.5.3 for descriptions of the many lists and sub-lists generated by the term search module.

Simple terms are extracted by ATAO and listed in a separate file with the *.MS (Mots Simples) file extension. The main list produced by ATAO, the strings of repetitive words, is placed in a separate file with the *.CH (Chaînes de Mots) extension. ATAO can optionally be configured to clean the *.MS file of transparent words, as well as doubles in the pre-translation dictionary to further reduce its size for the human translator. More information and an example of the complex nominals list is given in Appendix III.

“Thresholding” is a user-definable property used by ATAO to eliminate words from the list of term candidates that have infrequent or isolated instances in the input text. As a general rule, combinations with a low frequency constitute a major source of noise. Purging them from the list of identified term candidates makes it much shorter and easier to read for the human terminologist. It has been noted, however, that thresholding can sometimes remove otherwise-valid term candidates from the output list (thereby creating silence) because the only criterion for purging entries is the detected frequency in the input text. The thresholding setting can be raised or lowered to suit the user’s needs. However, the maximum number of occurrences of a word in a submitted text that ATAO will indicate is 999, regardless of how frequent the word actually appears in the text.

The terms contained in ATAO’s reference dictionary and any user project dictionaries are stored as term records. The linguistic information ATAO needs to process texts is entered in the ATAO field. The source language entry in ATAO’s reference or project dictionary cannot exceed 12 words. Strings over 12 words in length must be entered as phrasal units. Each punctuation mark counts as a word in this particular case.

All characters are interpreted literally by ATAO except for asterisks, which are interpreted as reference marks; semi-colons, which are treated as term separators; foreslashes (in some situations only) and pointy brackets < > (the other kinds of brackets are supported, however). If, for whatever reason, a section of the input text is not to be scanned, **<<!** and **!>>** symbols can be inserted before and after it to have ATAO skip to the next unmarked section. Sections to be skipped cannot be greater than one paragraph in length. Consequently, each paragraph to be skipped must be marked separately.

Spelling variations are recorded in ATAO by means of semi-colons. As such, a headword entry might appear as follows: *cesarean section; caesarean section; caesarian section; cesarian section*. A separate entry is not required for each variation. Moreover, abbreviations can be included in the same term record with the full word counterpart using the code */a/* to separate them, for example *direct access* *storage device /a/ DASD*. Bold, underline and italic text attributes can be hard-coded in the term record headword, if needed. Subject field codes can be entered directly in a term record headword using the pipe character **|** before and after the appropriate code.

2.1.5.2. Lexical coding of dictionary entries

ATAO uses several codes to describe the words in its dictionary:

- Nouns
- Adjectives (including past participles and ordinal numbers)
- Determinants (including definite articles, indefinite articles, possessive pronouns, demonstrative pronouns and numbers)
- Adverbs
- Standard and auxiliary verbs
- Invariable words or expressions
- Conjunctions
- Prepositions
- Personal and relative pronouns
- Coordinate conjunctions

ATAO extracts all strings of words from the input text in which all the elements are either nouns (including most gerunds) or adjectives (including past participles) for inclusion on the list of complex nominals. The entries on the list of term candidates are sorted by the last element they contain, then the headword is lemmatized, leaving the other elements of the term candidate in their original form. Sorting the list by their final element sorts term candidates by their head word and groups conceptually-similar terms together.

2.1.5.3. Output lists generated by ATAO

In addition to the main complex nominals list, ATAO generates five other sub-lists of term candidates. The nature of the different texts submitted to ATAO for processing can cause the length of the sub-lists it produces to vary considerably. Moreover, some lists can be decreased to next to nothing through thresholding. The contents of each output list are described below.

1. One sub-list is similar to the complex nominals list, except that it contains verbs as well as nouns and adjectives. This list is very similar to the main list of term candidates.
2. Term candidates are sorted by the second element in the “noun + preposition + other elements” format by their preposition.
3. Term candidates are sorted by the second element like in the second list, except that it sorts terms in the “adjective + preposition + other elements” format.
4. Term candidates are sorted by the “auxiliary verb + other words” format.
5. Strings of words that do not follow the other nominal linguistic patterns are sorted together. This commonly occurs because one of the elements of the complex nominal is not coded with all its possible syntactic categories.

ATAO also produces three sub-lists of phrasal units. Unlike the sub-lists of term candidates, entries in the three phrasal unit sub-lists are not lemmatized.

1. One sub-list uses the “preposition or conjunction + other elements” pattern to sort phrasal units.
2. One sub-list chops the text into units by punctuation marks.
3. The last sub-list sorts phrasal units by the first element per sentence.

2.2. The statistical approach to term extraction

In contrast to the linguistic approach, the statistical approach to automated term extraction focuses on the frequency of word combinations in the input text, and functions primarily on the recognition of repeated strings of characters. This technique is based on the premise that raw frequency in the input text has a direct relationship with the probability of a sequence of lexical units being a term. In other words, the more often a specific candidate string is detected in an input text, the greater the chances that it is a valid term. More specifically, lexical units which appear in the input text more frequently than what could be normally expected are assumed to be central to the text's subject matter. Similarly, a large-volume corpus covering a specialty subject field would normally contain more than just a single occurrence of most key terms.

The statistical approach is further supported by the observation that noun modifiers in LGP are mostly used for describing the headword; whereas compound noun terms use modifiers to distinguish from other, similar concepts in the same LSP. This means that after being used once or possibly twice in a text, non-terminological noun phrases tend to omit descriptors to promote conciseness and simplified reading. LSP nouns, on the other hand, are more likely to repeat the full noun phrase throughout a text to ensure clarity and understanding. This observation supports the belief that a higher incidence of repetition tends to indicate terminological status. Used in many areas of information retrieval and Natural Language Processing, this principle asserts that it is more than a chance happening when lexical items co-occur at an uncharacteristically high frequency, and that such co-occurrence must be indicative of a particular use within the bounds of the given context.

Statistical term-extraction systems are less dependent on dictionaries and linguistic rules to identify term candidates and are hence easier to mange according to Habert and Jacquemin (1993:20). Because of their more simple design, they are highly-powerful tools in that they can process large volumes of text in a relatively short period of time. Moreover, they are language independent to a large degree, which makes them more easily extended to process a new language than a term-extraction system based on morphosyntactic analysis. Another advantage of term-extraction systems based on statistical analysis is that, in theory, they can be extended to include the identification of simple terms, such as *failover*, *downtime* and *machination*. Lastly, depending on how a statistical term-extraction system has been configured for the minimum number of occurrences before appearing on the list of term candidates (thresholding), the frequency indicators can automatically eliminate term candidates that are comprised of a frequently-occurring head with an expansion that only occurs once inside the corpus. This is advantageous because most such occurrences are examples of noise caused by category ambiguity, such as “Many *vendors offer storage solutions* similar to this setup”. The N + V + [N + N] combination *vendors offer storage solutions* extracted erroneously as an N + N + [N + N] combination is naturally not a valid compound term.

2.2.1. Shortcomings of statistical-based systems

The main shortcoming of the statistical approach to term extraction is that, working on its own, it is insufficient without a minimum of linguistic information to supplement and filter through the raw statistics. One such combined term-extraction method uses syntactic evaluation to optimize the list of candidate terminological combinations extracted by statistical analysis. Part of the linguistic analysis consists of eliminating strings which posses neither a linguistic status (a plausible term formation pattern) nor semantic value, such as strings that start with articles or contain pronouns. These systems must also be able to go beyond analyzing for simple lexemes and search for co-occurring sequences of lexical units (i.e. compound terms).

The frequency threshold used to determine whether or not a lexical combination’s frequency is sufficient for inclusion on the list of identified units is in itself extremely difficult to establish. Although it removes a great number of non-terminological combinations from the list to filter, setting the threshold higher than one has the major drawback of eliminating all terms that only appear once in the input text. Human terminologists tend to couple frequency with extra-linguistic information to assist them when evaluating the terminological status of a sequence of lexical elements, something which is not currently an option for automated term-extraction utilities. In an effort to palliate this situation to a limited extent, some commercial term-extraction packages give the user the option of setting the minimum number of occurrences to record for a lexical unit before it appears on the list of term candidates. Also, the number of occurrences of extracted term candidates must be evaluated with respect to the length of the input text, which is not a factor that systems using linguistic analysis must contend with.

Lastly, certain aspects of how the input text has been written will have a greater impact on the results achieved by a statistical-based term-extraction system than with a system that uses linguistic analysis. For instance, a lengthy text with a high incidence of anaphora (demonstrative pronouns, etc.) will yield poor results as compared to a text where the subject is repeated on a more frequent basis.

2.3. The mixed approach — linguistic filters and statistics

The need to incorporate statistical measures in the term identification progress emerged from the fact that morphosyntactic analysis proved to be inadequate when used unassisted. This hybrid approach to automated term extraction has been studied and proposed in one form or another by several authors, among them Daille (1995), Drouin and Ladouceur (1994), and Drouin on his own (1996).

In the approach adopted by Drouin and Ladouceur (1994:21), repetitive strings from the input text are calculated according to their frequency, and then two sets of filters (morphological and sub-chain frequency) are applied to the raw output in order to cull unlikely items and non-terms from the term candidate report. This raw list is then passed over one last time to determine the terminological potential of each term candidate through an analysis of its immediate context.

Daille, on the other hand, has a different angle of approach. This two-step method has the input text follow an algorithm, whereby tagged text is submitted to a sequence of linguistic filters to isolate potential term candidates and sorted by statistical frequency to refine the selection process.

Once the list of high-frequency phrases has been created, the unlikely term candidates are culled from it by submitting the list for processing by a series of language filtering rules. The following flowchart illustrates a number of such rules that this type of system might choose to apply.

The results from the flowchart filtering process are sorted by frequency and the term candidates with the highest frequency are considered to be the most likely to be genuine terminological units. In order to keep the list of candidate terms to filter from becoming unmanageably long, certain measures can be taken to restrict its length, such as imposing a minimum threshold of occurrences (established in keeping with the length of the input text) before a string is added to the list of phrases to be filtered.

With the exception of simple terms, analyzing an input text using this approach would appear to generate very little silence, provided that all lexical units were properly coded in the system database and the system contained an appropriate number of term filtering rules, because the vast majority of terms would fit the patterns and be extracted for the term candidate list. The drawback of this approach is that it runs a greater risk of extracting common non-terminological combinations (noise), particularly when the non-terminological combination contains a modifier that can be equally applicable in both fixed LGP expressions and LSP. One such modifier is the adjective *high*, which can appear in LSP (high resolution) as frequently as in LGP (high prices).

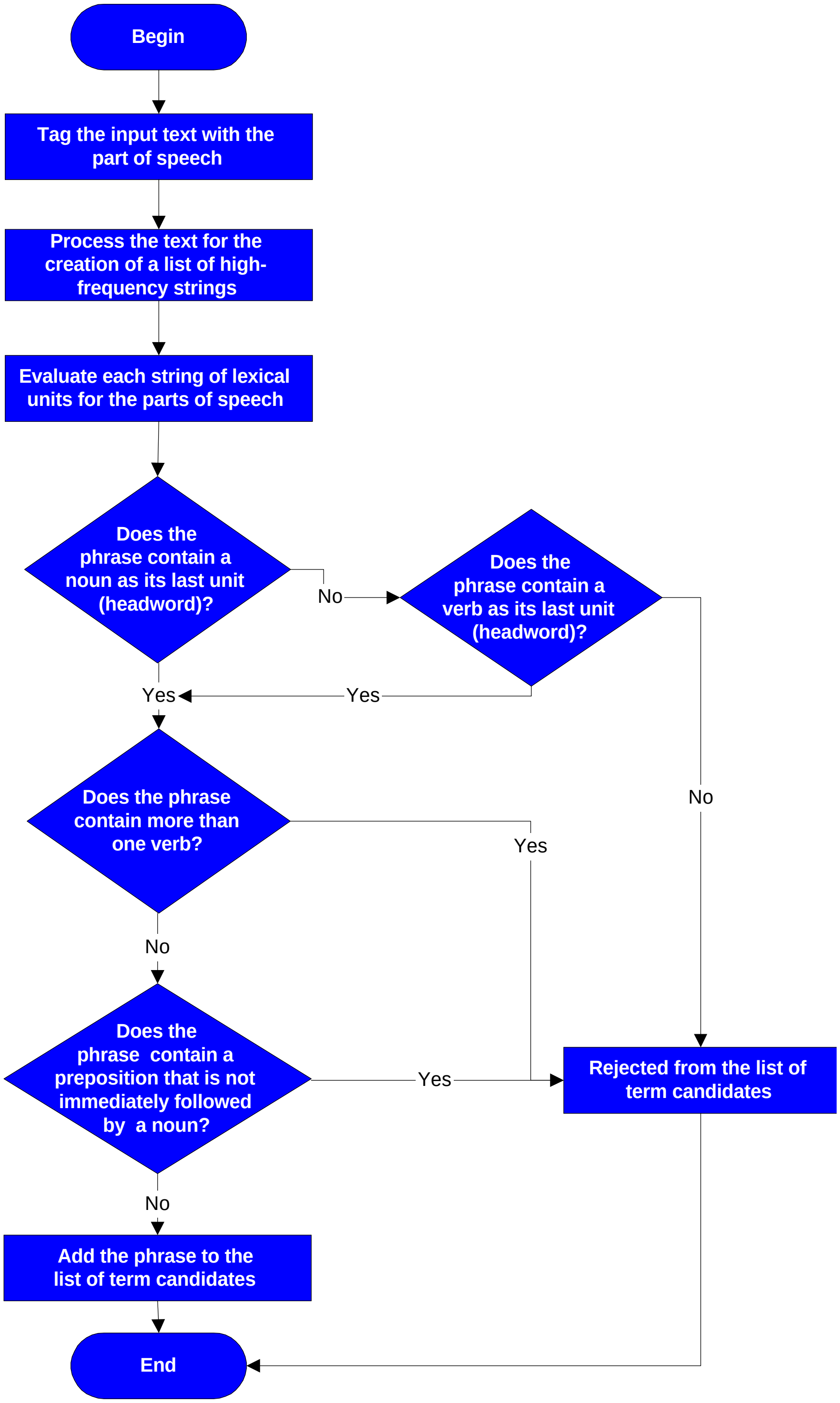

Begin
Tag the input text with the part of speech
Process the text for the creation of a list of high-frequency strings
Evaluate each string of lexical units for the parts of speech
Does the phrase contain a noun as its last unit (headword)?
No
Does the phrase contain a verb as its last unit (headword)?
Yes
Yes
No
Does the phrase contain more than one verb?
Yes
No
Does the phrase contain a preposition that is not immediately followed by a noun?
Yes
Rejected from the list of term candidates
No
Add the phrase to the list of term candidates
End

Figure 2-3 Mixed approach to automated term extraction

2.4. The current state of automated term extraction

Several observations can be made after our review of a selection of commercially-available term-extraction utilities. We were surprised to find, for example, that term extraction is but a part of the much larger field of terminology automation and most terminology extraction programs are encapsulated in larger, multi-purpose applications. In particular, this is the case for both LOGOS and ATAO, which comprise the focus of our study. In keeping with this observation, we noticed that many term-extraction programs also deliberately attempt to extract frequently-appearing phrasal units that are admittedly not terminological. Such units are undoubtedly useful to the translator (who must translate the entire text, and not simply the technical vocabulary it contains), but are of little use or even counterproductive to terminologists or lexicographers. We are uncertain whether this is a “software feature” to justify to a certain extent the noise created by non-terminological combinations or if it is intentional in order to assist translators with their work.

Moreover, the subjectivity of what exactly constitutes a terminological unit (see Chapter 1 for a review of this situation) emerged as a key obstacle to perfecting term-extraction utilities such as those investigated in this study. Associated with this obstacle is the fact that no currently available commercial system we encountered can boast term candidate reports that do not require post-editing by a human terminologist.

We found that the systems we investigated approached the task of scanning submitted texts for terms in a variety of manners: counting term candidates' frequency in the input text and statistics, parsing and analyzing the function of each element of a sentence, linguistic filtering, matching the input text against a set of term formation patterns predefined in the system, and any combination thereof. A common point of all these systems, however, was the absence of semantic tagging to indicate the relationship between the constituents of compound terms or other extra-linguistic information to help the term-extraction system select its term candidates in a manner more like that of a human terminologist. As proposed by Hannan (1996) and Lauriston

(1994), we agree that the addition of simple semantic information to the term-extraction methodology would be a welcome addition to the term-extraction systems available in the near future. We advance that this is the best, and possibly the only, way to improve the accuracy of term-extraction systems much beyond their current performance.

In order to obtain the highest level of accuracy and efficiency when gathering and analyzing our data on the term candidates extracted by LOGOS and ATAO for the purposes of this study, we developed a computerized management tool named Term Extraction Management System, or TEMS. A detailed investigation of this software program is the focus of Chapter 3.

3. TEMS, THE TERM EXTRACTION MANAGEMENT SYSTEM

3.1. System overview

TEMS is a computerized solution to the difficulties of managing the thousands of term candidates extracted by LOGOS and ATAO, as well as the contents of the human list. Developing the TEMS system was necessary to the evolution of this project because it facilitates comparing and contrasting LOGOS and ATAO. TEMS was an invaluable aid to our study because our data was stored in a format that was highly intuitive and easily exploited through querying the TEMS database with Microsoft Access. By merging our two data sets on the same tab of TEMS' interface, similarities and differences between the two rise to the surface and become more visible when calculating statistics. Furthermore, statistics comparing any two systems are more easily calculated and tested when their data is located in a single database. Our statistical reports on term composition and extraction performance results were generated using TEMS' raw data and the query builder of Microsoft Access. A working version of TEMS, including all source code and a sample database, is available on the compact disc attached to the back cover of this work.

Developed with Microsoft Visual Basic 5.0 for the Windows 98/NT 4.0 platforms, TEMS is a stand-alone executable linked to a Microsoft Access database through the Data Access Object (DAO). TEMS requires approximately 8 megabytes of free hard disk space, 256-colour display mode and a minimum 640 x 480 resolution for optimal performance. Rounding out the list of TEMS' key points is a complete setup and uninstall utility accessible through the Windows Control Panel. Microsoft Access does not need to be installed for TEMS to function properly; however, its query builder is the method of choice for compiling statistics based on TEMS' raw data.

Each output list of term candidates was carefully entered in TEMS, with each unique term candidate becoming a record in the system database. Once the term candidates were entered in TEMS, specific details about each were entered in the appropriate tabs. Once the records for the entire list of term candidates and the human list were completed, the Microsoft Access database running in the background was ready to be queried using Microsoft Access' advanced SQL (Structured Query Language) environment.

All five tabs in the TEMS main screen share a common menu bar, as shown in Figure 3-1.

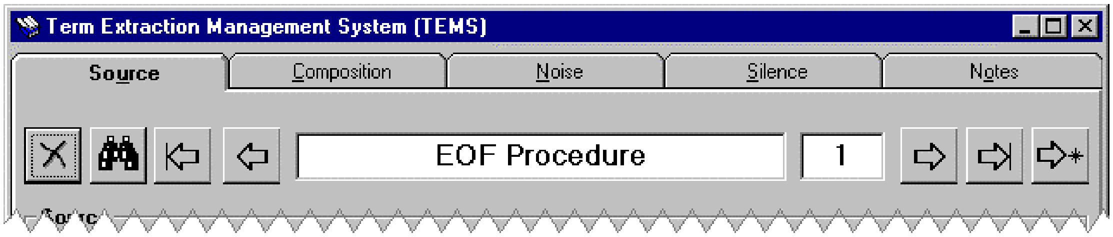


Figure 3-1 The TEMS menu bar

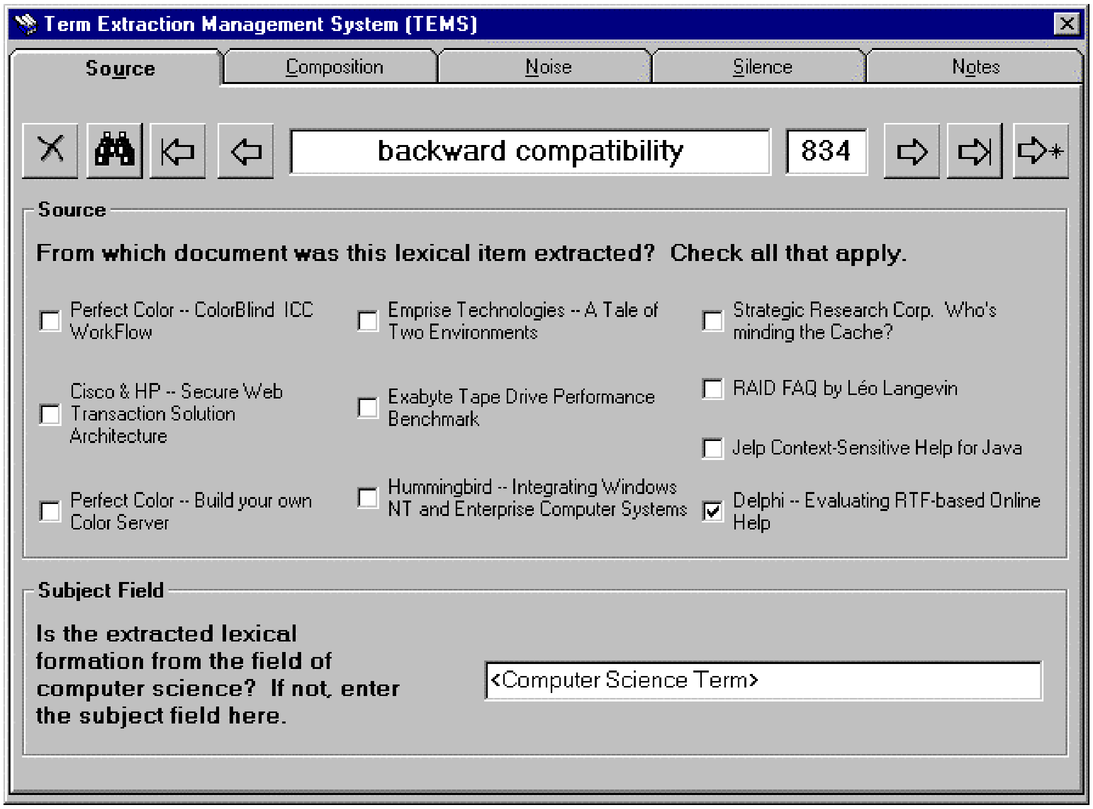
In addition to displaying the extracted term candidate (the current record) and the record number, the menu bar allows the user to scroll through the database records, add new records to the system database, delete an existing record or open the TEMS Search screen to locate a specific term candidate or group of records.

Measures were taken to minimize avoidable errors while using TEMS, such as inadvertently deleting a term candidate or modifying its TEMS record number (the primary key used internally for managing the database records and searching TEMS). The record number field is auto-generated upon the creation of a new database record to ensure it is unique, and confirmation from the user is required before a record is deleted from the database.

3.1.1. Source tab

When TEMS is first opened, the Source Tab is visible to the user (see Figure 3-2). In the upper portion of the screen, the names of the documentation sources that were submitted to ATAO and LOGOS for term extraction are listed. For each new term candidate that is added to TEMS, the user simply clicks the name of the source where it appears and the information is recorded in TEMS' database for statistical purposes and querying.

In the lower section of the screen, the user enters the subject field to which the extracted term candidate belongs. If no entry is made, TEMS assumes by default that the term candidate belongs to the field of computer science. If a term candidate could belong to multiple subject fields, only the subject field most pertinent to the candidate's immediate context is entered.



backward compatibility | 834

Source

From which document was this lexical item extracted? Check all that apply.

☐ Perfect Color -- ColorBlind ICC WorkFlow
☐ Emprise Technologies -- A Tale of Two Environments
☐ Strategic Research Corp. Who's minding the Cache?
☐ Cisco & HP -- Secure Web Transaction Solution Architecture
☐ Exabyte Tape Drive Performance Benchmark
☐ RAID FAQ by Léo Langevin
☐ Jelp Context-Sensitive Help for Java
☐ Perfect Color -- Build your own Color Server
☐ Hummingbird -- Integrating Windows NT and Enterprise Computer Systems
☑ Delphi -- Evaluating RTF-based Online Help

Subject Field

Is the extracted lexical formation from the field of computer science? If not, enter the subject field here.

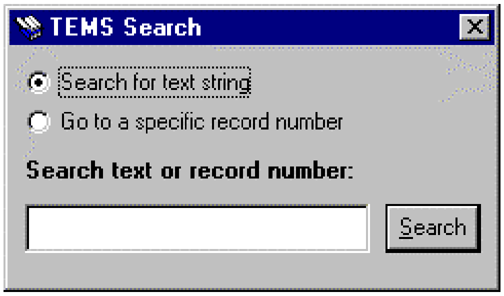
<Computer Science Term>

Figure 3-2 The TEMS Source tab

When TEMS is initialized, the current record is always the first record in the system database. However, this is not necessarily the record required by the user. To search for a particular record or string of characters in the TEMS database, the user simply clicks the search button on the TEMS menu bar to open the search interface where the criteria is entered and the query results returned. See Figures 3-3 and 3-4 for representations of the TEMS Search screen.

3.1.2. TEMS search screen

When the TEMS Search screen opens, it is initially displayed in an abbreviated view (see Figure 3-3). After a query is executed, however, the window lengthens to accommodate a listbox comprising the query result set. This window was designed as such because the user has no need for the listbox until a query has been executed, and the very size of the listbox obscures a significant portion of the TEMS interface.



TEMS Search

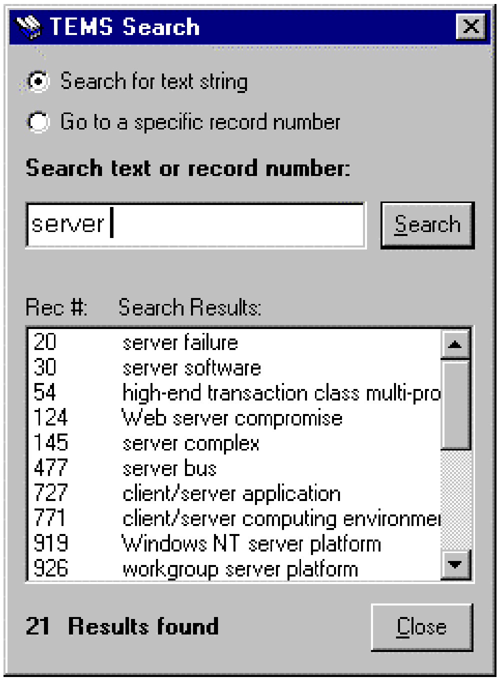
● Search for text string

○ Go to a specific record number

Search text or record number:

Figure 3-3 The TEMS Search screen (abbreviated view)

There are two methods of searching through the TEMS database of term candidates: searching for a text string throughout the entire list of term candidates or entering a specific record number to move to. Searching for character combinations in TEMS is carried out using the most general method possible. When the search string is entered, TEMS automatically adds the SQL operator LIKE before and after the string so that every possible occurrence of the search string will be returned in the query's result set. As such, there is a much greater chance that the exact record being sought will be returned. Furthermore, this method of searching does not require the desired term candidate to be typed out in full before executing the search. With respect to the subject of this work, one might say that this search approach returns a list with a higher incidence of noise, but no incidences of silence.



● Search for text string
○ Go to a specific record number

Search text or record number: server

Rec #:	Search Results:
20	server failure
30	server software
54	high-end transaction class multi-pro
124	Web server compromise
145	server complex
477	server bus
727	client/server application
771	client/server computing environme
919	Windows NT server platform
926	workgroup server platform

21 Results found

Figure 3-4 The TEMS Search screen (full view)

A powerful feature of the TEMS Search screen is the support of the wildcard characters ***** and # when searching the database by text string. The ***** character can optionally replace one or several characters in the search string. As such, searching for *COLO*R* will find all instances of both *color* and *colour* in TEMS' database at the same time. Similarly, searching for *ON*LINE* will return "online," "on line" and even "on-line" for the same query. This type of wildcard searching is practical when working with a significant number of acronyms. For example, entering *H*T*M*L* will return both the full term *HYPERTEXT MARKUP LANGUAGE* as well as its abbreviated form *HTML*. Finally, entering only ***** as a search string will return the entire contents of the TEMS database, practical for scrolling quickly though a number of term candidates or calculating the total number of TEMS records. Entering only # as a search string will return all entries in TEMS' database that contain a numeric character (1 - 9) such as *32-bit processing*. All in all, flexibility when searching proved to be one of TEMS' strongest functionalities.

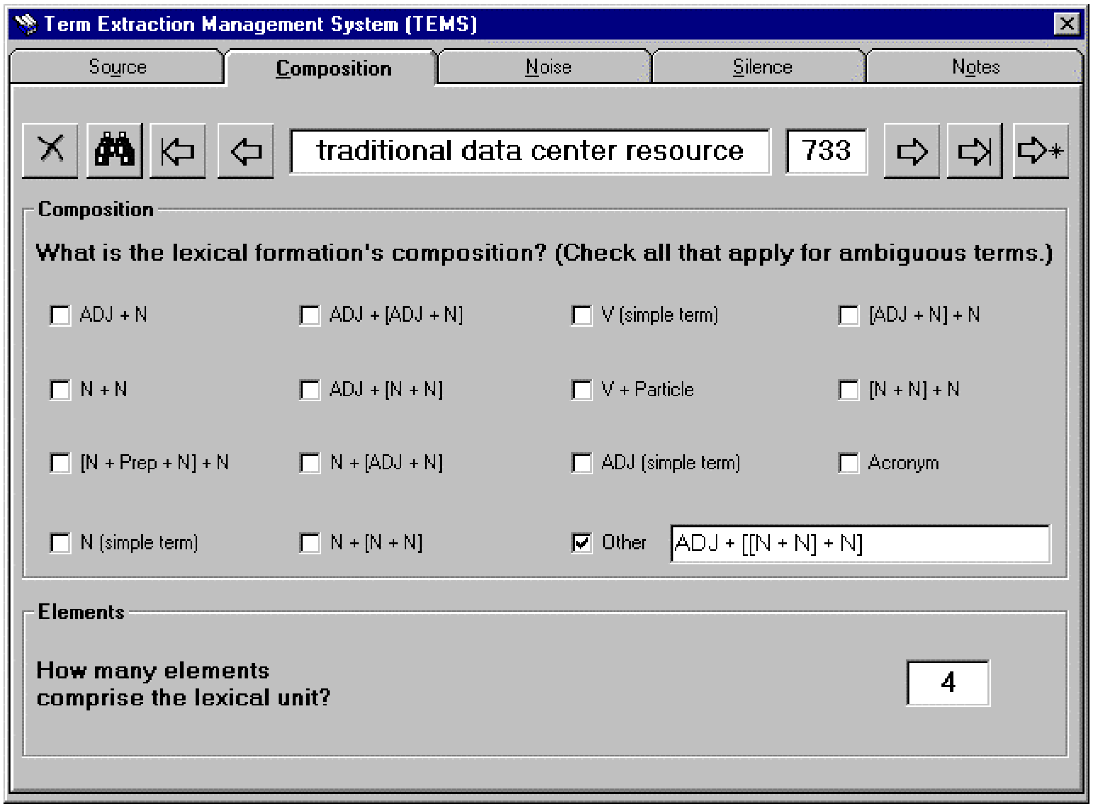
To improve usability, the TEMS Search screen lists the record number as well as the term candidate in the listbox for subsequent use as a search key. Searching in TEMS is not case-sensitive, although accented characters and symbols are supported. Unlike Termium, the search string can be pasted in TEMS by the clipboard instead of forcing manual entry. The number of matches in the listbox is given at the bottom of the screen to help make sense of queries that return overly large result sets. Moreover, searching is possible without using the mouse because simply pressing Enter executes the query.

Once the desired term candidate has been located, double-clicking its reference in the listbox updates the current record in the TEMS main window for consultation. The search window remains visible until it is closed manually, however, in case the record initially selected turns out not to be the correct one. The results from the last executed query remain in the TEMS Search screen until either another query is executed or TEMS is closed by the user. TEMS was not designed to store queries to disk for later use.

3.1.3. Composition tab

The next area of interest in the TEMS interface is the Composition tab (see Figure 3-5), where information pertaining to the term candidate's lexical composition is entered. Fourteen different term formation patterns are listed in the upper section of the TEMS interface, with the fifteenth formation pattern left blank for cases where the term candidate's composition was indiscernible (for instances of noise) or unlike any of the on-screen options available. It is possible to check off more than a single term formation per term candidate because simple terms and some elements of compound lexical units are ambiguous and subject to interpretation, for example, some verb forms might also be used for nouns, etc.

In the lower section of the screen, the number of elements that comprise the term candidate is entered. This information was recorded to discern between compound terms and single units, and also to investigate the occurrences of longer terminological units in our corpus.



traditional data center resource 733

Composition

What is the lexical formation's composition? (Check all that apply for ambiguous terms.)

□ ADJ + N □ ADJ + [ADJ + N] □ V (simple term) □ [ADJ + N] + N

□ N + N □ ADJ + [N + N] □ V + Particle □ [N + N] + N

□ [N + Prep + N] + N □ N + [ADJ + N] □ ADJ (simple term) □ Acronym

□ N (simple term) □ N + [N + N] ☑ Other ADJ + [[N + N] + N]

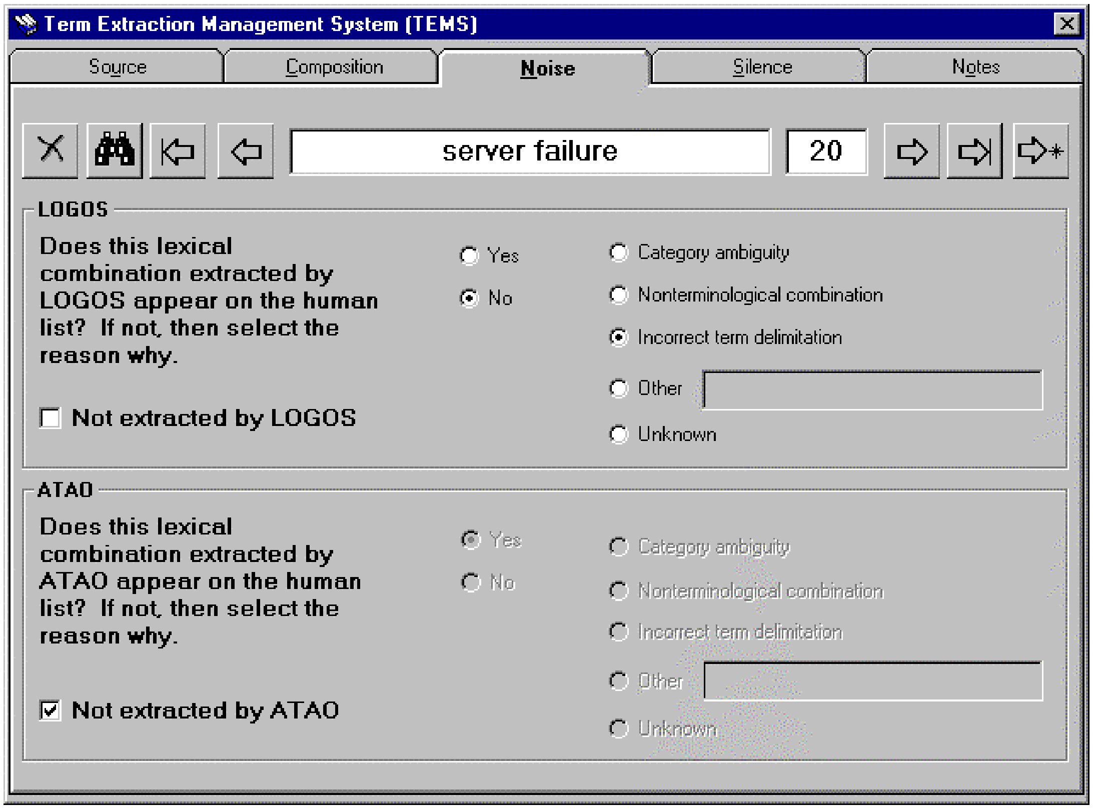
Elements

How many elements comprise the lexical unit? 4

Figure 3-5 The TEMS Composition tab

3.1.4. Noise and Silence tabs

Following the Composition tab are the Noise and Silence tabs, where any incidences of noise or silence are documented (see Figures 3-6 and 3-7). One tab focuses on noise for both extraction programs and the other does the same for silence, instead of one tab for LOGOS (both noise and silence) and a second for ATAO (both noise and silence) because it seemed more logical to group the extraction results by the type of error instead of by the program that generated the list of term candidates. Organizing noise and silence in this manner in TEMS also proved to be a better choice for analyzing the term extraction results afterwards. On the other hand, it is admittedly more tedious for the user when initially entering the term candidates because effort is required to switch between the Noise and Silence tabs when entering data on the same term candidate.



server failure 20

LOGOS

Does this lexical combination extracted by LOGOS appear on the human list? If not, then select the reason why.

○ Yes
● No

○ Category ambiguity
○ Nonterminological combination
● Incorrect term delimitation
○ Other
○ Unknown

□ Not extracted by LOGOS

ATAO

Does this lexical combination extracted by ATAO appear on the human list? If not, then select the reason why.

● Yes
○ No

○ Category ambiguity
○ Nonterminological combination
○ Incorrect term delimitation
○ Other
○ Unknown

☑ Not extracted by ATAO

Figure 3-6 The TEMS Noise tab

For any given term candidate, there are three possible scenarios concerning extraction by one of the software programs and potential inclusion on the human list. They are summarized as follows:

The lexical unit is a valid term which was extracted by the software program and is included on the human list.

1. The lexical unit is not a valid term and is not included on the human list. It was however, extracted as a term candidate by the software program. (Noise detected)

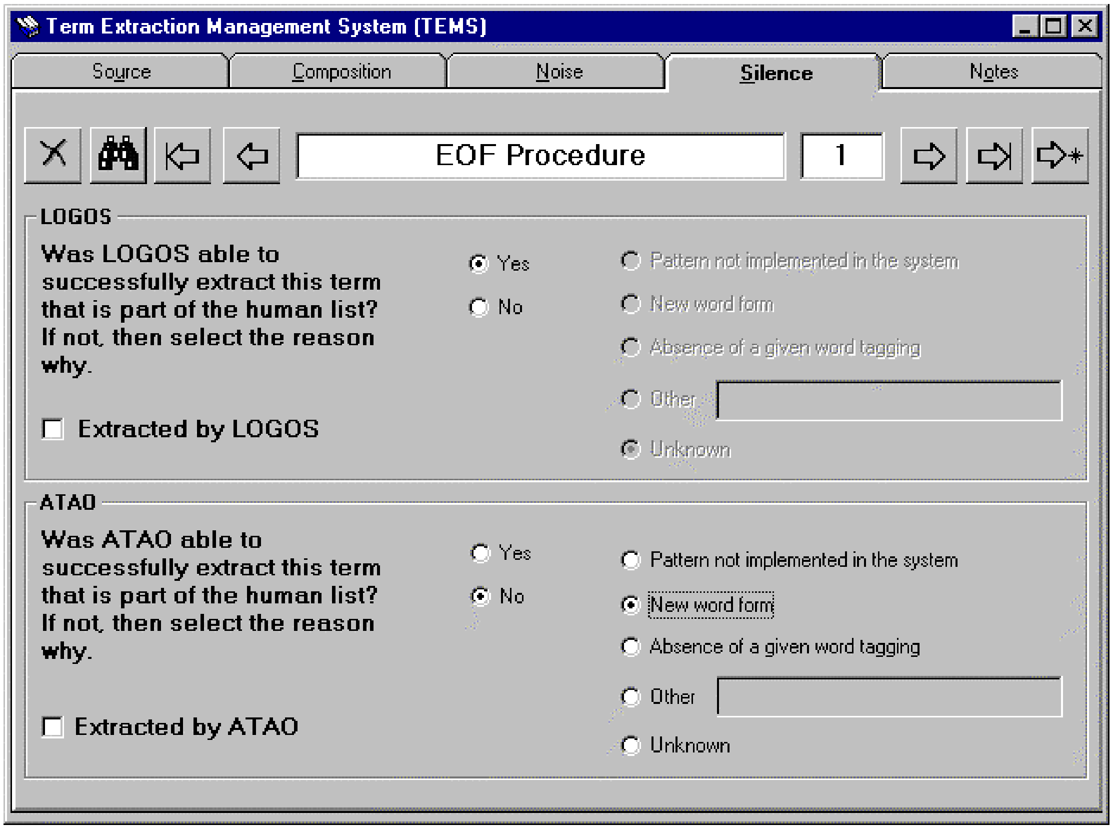
2. The lexical unit is a valid term and appears on the human list, but was not extracted as a term candidate by the software program. (Silence detected)

The purpose behind the Noise and Silence tabs of the TEMS interface is to accurately and unequivocally reflect these three possibilities for every term candidate in the TEMS database. Both tabs must be filled out for each term candidate in order for processing to be considered complete.

In cases of scenario #1 (where the term was extracted by the software program and appears on the human list), the checkmark is cleared from the checkbox in the Noise tab labeled “Not extracted by LOGOS (or ATAO)” to enable the first array of option buttons. Since we already know that the term candidate was in fact included on the human list, we click the “Yes” option button in the Noise tab, leaving the second array of option buttons disabled.

In cases of scenario #2 (noise), the checkmark is cleared from the checkbox in the Noise tab labeled “Not extracted by LOGOS (or ATAO)” to enable the first array of option buttons. Since the term candidate was not included on the human list, we click the “No” option button in the Noise tab to indicate that this is an incidence of noise, which in turn enables the second array of option buttons. From the second array of option buttons, a probable reason why the system erroneously extracted the non-term is indicated once the incident has been analyzed.

Lastly, in cases of scenario #3 (silence), the checkmark is cleared from the checkbox in the Silence tab stating “Extracted by LOGOS (or ATAO)” to enable the first array of option buttons. Since the term candidate is included on the human list but was not extracted by the software program, we click the “No” option button in the Silence tab to indicate that this is an incidence of silence, which in turn enables the second array of option buttons. From the second array of option buttons, a possible reason why the system failed to extract the term that appears on the human list is indicated once the incident has been analyzed.



EOF Procedure

1

LOGOS

Was LOGOS able to successfully extract this term that is part of the human list? If not, then select the reason why.

● Yes
○ No

○ Pattern not implemented in the system
○ New word form
○ Absence of a given word tagging
○ Other
● Unknown

□ Extracted by LOGOS

ATAO

Was ATAO able to successfully extract this term that is part of the human list? If not, then select the reason why.

○ Yes
● No

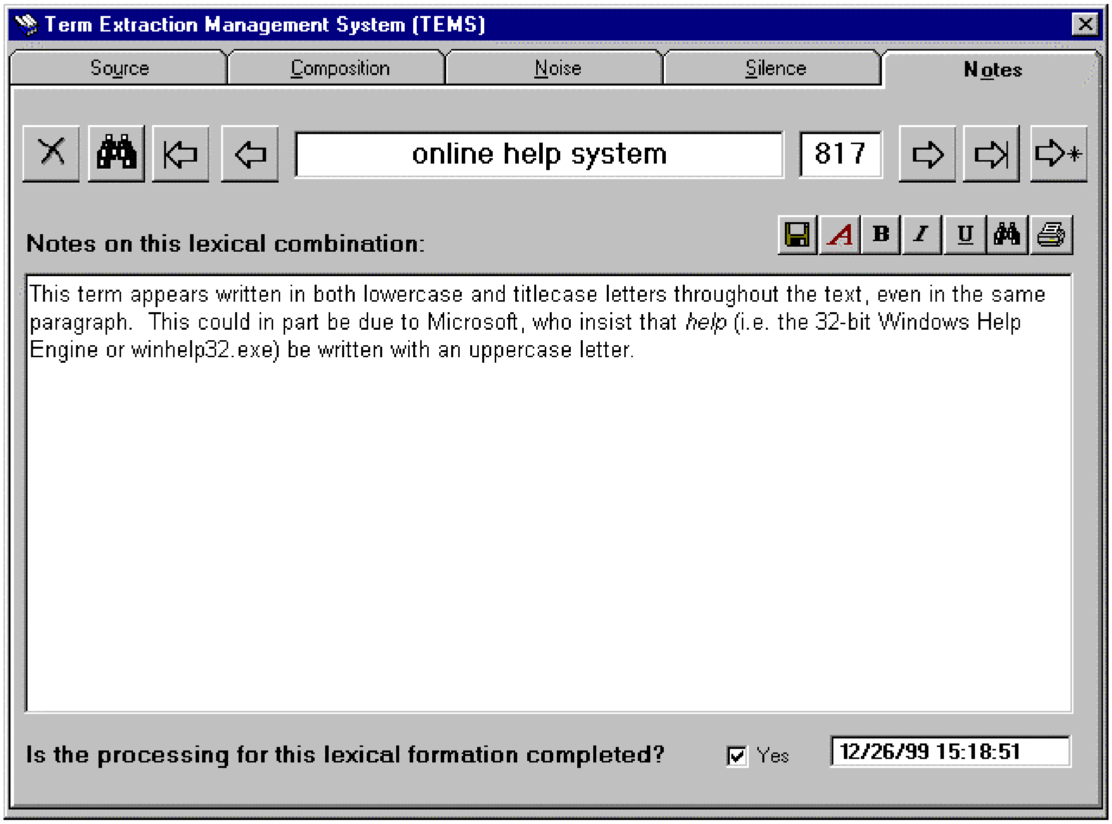
○ Pattern not implemented in the system
● New word form
○ Absence of a given word tagging
○ Other
○ Unknown

□ Extracted by ATAO

Figure 3-7 The TEMS Silence tab

3.1.5. Notes tab

The fifth and final tab of the TEMS interface is the Notes tab, where any other pertinent piece of information related to the term candidate that was not documented elsewhere in TEMS may be stored for further reference. This includes observations, comments, interim conclusions and even the context in which the term candidate was located. It also indicates the date and time at which processing for the current record was completed, which is a useful reminder when following up on thousands of term candidates.



online help system

817

Notes on this lexical combination:

This term appears written in both lowercase and titlecase letters throughout the text, even in the same paragraph. This could in part be due to Microsoft, who insist that *help* (i.e. the 32-bit Windows Help Engine or winhelp32.exe) be written with an uppercase letter.

Is the processing for this lexical formation completed? ☑ Yes 12/26/99 15:18:51

Figure 3-8 The TEMS Notes tab

The notes on the current record are presented in an RTF box that includes many features found in commercial word processors. The TEMS RTF box supports font attributes such as bold, italics, underline, font and point-size definition, and offers other features such as text string search, contextual menu, undo, clipboard support, drag-and-drop text editing, printing the notes and even saving the notes to an RTF file for subsequent editing in another application. The rationale behind the decision to incorporate a limited word processor in TEMS is two-fold. By doing so, all additional information about a term candidate may be stored within the database itself without the hassle of maintaining a separate source for term candidate notes. Furthermore, storing the extra information about the term candidates directly inside TEMS makes this data more quickly and easily accessible.

Our evaluation of the appropriateness of the TEMS program, as well as our impressions of using this approach to data management are given in Section 5.6. of this work.

In Chapter 4, we explain the methodology employed to contrast and evaluate the two term-extraction utilities in this research study: LOGOS and ATAO.

4. DESCRIPTION OF OUR RESEARCH METHODOLOGY

4.1. Introduction

This chapter is concerned with the methodology of the experiments that were carried out to observe the differences between the term extraction output generated by LOGOS and ATAO respectively.

Simply put, in our experiment we manually scanned a series of electronic texts for terminological units, entered the information in the TEMS program[9] then statistically evaluated the composition of the results, thereby creating the human list of terms to be compared with the output from the two term-extraction utilities. Once the human list was completed, the electronic texts were submitted to the term-extraction utilities for automatic processing. The machine output data was entered in TEMS, the performance figures in a number of categories for each utility were calculated and the results obtained for each utility were analyzed and compared for trends and disparity. Greater detail of how our research was carried out is given in the sections below, and a graphical representation appears on the following page.

[9] TEMS is a *Terminology Extraction Management System* that was developed expressly for this study. Please refer to Chapter 3 for an in-depth discussion of the TEMS program, as well as screen captures of the program interface.

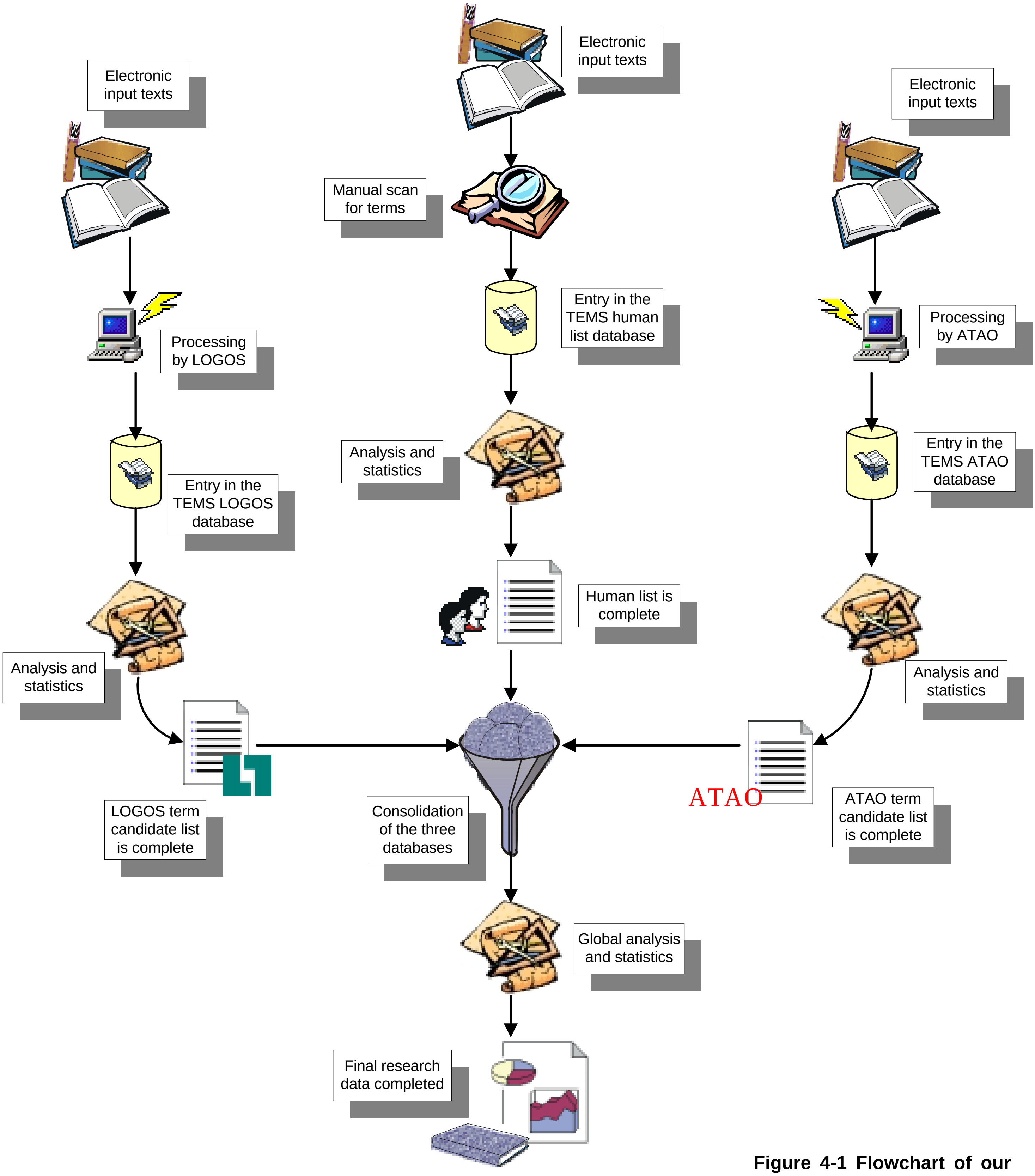

Electronic input texts
Electronic input texts
Electronic input texts
Manual scan for terms
Entry in the TEMS human list database
Processing by LOGOS
Processing by ATAO
Analysis and statistics
Entry in the TEMS LOGOS database
Entry in the TEMS ATAO database
Human list is complete
Analysis and statistics
Analysis and statistics
LOGOS term candidate list is complete
Consolidation of the three databases
ATAO
ATAO term candidate list is complete
Global analysis and statistics
Final research data completed

Figure 4-1 Flowchart of our research methodology

When defining the syntactic patterns of the manually-extracted terms on our list, we used the term formation model elaborated by Selkirk (1982:44)[10]. This typology was selected because it was the most complete and best explained. The author calls her theory of compounding “a simple context-free grammar for generating compound word structures,” (1982:13). Furthermore, her approach was also more easily applied (that is, closer to the real-life concerns of automated term extraction and less theoretical) than the typology proposed by Sager *et al*. because there was no reliance on semantic or extra-linguistic information. Other term typologies we examined for possible use in this experiment, such as that in (Levi:1978), were rejected because, while interesting, they seemed to focus almost exclusively on nouns; whereas we needed coverage of all types of terms because running text is not by any means limited to noun terms.

L’Homme *et al.* (1992:294) advance that in order to perform a full comparative evaluation of LOGOS and ATAO, they must be examined in the two main areas of focus: pre-processing and post-processing. Both phases of the evaluation process are described below.

Pre-processing evaluation: includes areas such as the targeted user group (i.e. translators, terminologists and possibly document managers), the automatic term extraction strategy applied by the program (i.e. linguistic, statistical or a hybrid approach), the types of terminological units identified by the program (i.e. only noun phrases or other types of terms as well), various system options (i.e. automatic storage in a database and morphological analysis) and the reports generated by the software (i.e. the number of different types of lists and the form that these lists take).

[10] A detailed look at Selkirk’s typology of English compound terms is given in Section 1.2.3.2.

Post-processing evaluation: consists of evaluating the actual performance of the system or the quality of its raw output; in other words, its ability to identify all the terminological units in a text with a minimal amount of noise.

After reviewing the programs, we decided to only carry out a post-processing evaluation of the two term-extraction utilities instead of a full evaluation because we believe that evaluating pre-processing criteria is subjective and depends almost entirely on the individual user's needs. On the other hand, all users, regardless of their discipline, subject field and the particulars of their working environment, want the highest level of accuracy when identifying terms coupled with the lowest incidence of noise. Consequently, our post-processing evaluation will apply equally well to all users.

Other than choosing to base our study on a single term formation typology and limiting our comparison to a post-processing evaluation, we did little else to narrow our focus for the experiment other than choose our corpus of texts to scan from one specific subject field — computer science. When creating the list of manually-extracted terms, however, we extracted all terms, not only terms from our chosen subject field.

4.2. Preparing the list of manually-extracted terms

4.2.1. Text selection

We began by gathering a selection of texts in electronic form from the Internet to be used for manual and electronic scanning. An initial sampling of 15 texts was compiled, from which five texts were culled, leaving a final cut of 10 texts. All texts that were chosen are highly technical documents from various branches of the field of computer science and vary in length from 778 to over 6000 words. None of the texts directly focus on computer programming however, because such texts necessarily include code examples and the names of program-specific variables, functions, etc. which could negatively affect the performance of the term-extraction utilities by generating an uncharacteristically high level of noise.

An attempt was made to choose texts that were similar in nature; we avoided texts that offered little more than technical specifications or were drafted using a format of excessive lists and point form instead of free-flowing sentences and paragraphs. By and large, the chosen texts are what are known as *white papers*, detailed technical reports regarding a company's product or service whose publication is intended to publicly disseminate impartial information in an official manner. Consequently, the texts scanned for the creation of our corpus were all published by large corporations; none were written by students, academic researchers or members of professional associations such as the Institute of Electrical and Electronics Engineers (IEEE)[11]. This decision was made for two reasons. First, in real-life situations, term-extraction utilities would be used most often to process non-academic texts; so it was expected that our choice of texts would produce more realistic and reliable results. Second, we were under the impression that a major multinational corporation would hire professional technical writers to compose, edit and proofread widely-published corporate literature; hence we believed we would obtain the highest possible quality of writing from such texts.

4.2.2. Spotting terms

For the purposes of collecting our corpus of terms, we extracted all terms that fulfilled the criteria to be considered as such. We did not omit genuine terminological units from other subject fields that appeared in our electronic texts because the term-extraction software we are evaluating cannot discern which terms belong to the subject field we are treating and which inadvertently found their way into the text as either examples or digressions. This approach is similar

[11] Please refer to Appendix I for an annotated list of all electronic texts used to compile our corpus, as well as their source.

to that adopted by translators, who must translate all terminology in the source text, not only the terms from one main subject field.

We did not apply very stringently to our corpus the naming requirements advanced by Sager (1991) in Section 1.2.2.2 for what constitutes a term because we were not forming new vocabulary and feared that otherwise acceptable terms might be discarded, although he states that existing vocabulary should also adhere to these same principles. We do believe, however, that his list of criteria should be followed as closely as the situation permits when forming new terms.

While scanning our electronic texts, we looked for characteristics of potential terms that increase their level of lexicalization, such as the following:

- Transparency
- Motivation
- An identifiable syntactic relationship between the components of the potential term
- Consistency (if it appears that similar terms are being extracted)
- Use of either the analytic or synthetic method of term formation[12]
- Coherence between terms[13]

A lexical unit‘s level of lexicalization was high enough to be considered a term when its meaning in the text was not that belonging to general language, for example *robustness* (when referring to a software application’s stability, flexibility and scalability). We did not limit our corpus to the extraction of compound terms and retained all types of valid complex term formations, not simply noun phrases.

[12] See the footnote in Section 1.2.1.2 for more information regarding these term formation methods.

[13] By “coherence”, we mean that if there exists a term such as *open system architecture*, there necessarily must be a term to express the notion of a *closed system architecture*. An excellent discussion of this point can be found in L’Homme (1991:33).

When creating the human list, we included terms that were not implicit because of the way the sentence was presented (referred to as “coordinate terms” in ATAO), regardless of the fact that the term extraction software would not be able to extract such terms individually because of a lack of conceptual or extra-linguistic information. For example, *TCP/IP utilities and applications* would generate two separate terms for the corpus, *TCP/IP utilities* and *TCP/IP applications* because the modifier *TCP/IP* is equally applicable to both nouns was only omitted in the second instance to avoid wordiness.

The potential for term redundancy was an issue to which we paid special attention because our corpus of manually-extracted terms cannot contain doubles or even variations of the same term, since such variations would skew our results. The following list of nine such areas for potential duplication, proposed by Bédard (1992:745), is admittedly not exhaustive. This weakness in the term-extraction process is palliated, at least in part, by the pre-editing module in ATAO and the Terminology Verification component of LOGOS.

Type of redundancy	Example
The presence or absence of a hyphen when the term retains the same function and part of speech	single-pole switch vs. single pole switch
Whether two units are fused together or simply juxtaposed	fuse holder vs. fuseholder
Random use of endings for modifiers	regulator device vs. regulating device vs. regulation device
The use of variable elements, often numbers, inside frequently-occurring strings	in row X of column Y
The presence of absence of certain punctuation marks from one instance of a lexical unit to the next	Hallowe’en vs. Halloween, lily-o-the-valley vs. lily-o’-the-valley
Typographical errors, homonyms or transfer errors that may slip into a source text	affect vs. effect, insure vs. assure vs. ensure, remuneration vs. renumeration
Spelling variations between different dialects of the source language, such as British, Canadian and American English, which can be found from one document to the next or even within the same text	colour vs. color, jewellery vs. jewelry

The use of dashes, whether they are em dashes or en dashes, and whether or not a space is inserted between the dash and the text it offsets	high-end system vs. high - end system
Random variations in how otherwise identical strings are presented in the same text. Since term extraction in general relies on exact pattern matching, such fluctuations cause the software program to present each variant as a totally different term candidate.	

Table 4-1 Types of redundancy when extracting terms

All the electronic texts were printed out for manual scanning in order to improve accuracy and to allow working from an easy-to-use, visual representation. Once the manual scanning was complete, data on the manually-extracted terms was entered in TEMS, with each term corresponding to a separate record in TEMS' database[14].

At this stage of the experiment, each TEMS term record contained the following information:

1. **The full term as it appeared in the electronic text:** Terms were recorded in TEMS exactly as they appeared in the source text. Any extraneous characters such as slashes, quotation marks, commas and hyphens were left intact to preserve realistic conditions. Capitalization was left as it was in the electronic text, except when the term was capitalized because it was the first word in a sentence. Text attributes such as italics or boldface type, however, were omitted from the term record.

2. **The unique record number used to identify each term:** Each term record was assigned a unique number to facilitate searching for exact records and managing the thousands of records in TEMS' database. Once a term was assigned a record number, the number could not be modified. Also, record

[14] Full screen captures illustrating typical data records from TEMS are given in Chapter 3 of this research project.

numbers for terms that were purged from the database could not subsequently be reassigned to another term record.

3. **The electronic source(s) where the term was spotted:** If exactly the same term was spotted in more than one of the electronic texts used to compile our corpus, more than one source checkbox was ticked to indicate the duplicate sighting instead of creating a second term record. Hence, each record in our database is unique. The same applies to variant presentations of a given term, such as the same term written with and without a hyphen, the British and American spellings for the same term or the same term appearing twice in the electronic texts, but with different capitalization. Such differences were noted on an individual basis in the affected term record.

4. **The subject field to which the term belongs:** The term's subject field was entered here, with computer science given as the default entry. In cases where the term was from another subject field, the term was researched in Termium for the appropriate subject field and then, where possible, validated with a subject matter expert. Once the terms from all the electronic texts had been entered in TEMS, the non-computer science subject fields were compared for consistency in how they were recorded. For example, all graphic arts terms was coded as being from the field of "Printing and Graphic Arts" instead of some being coded as simply "Graphic Arts" and others entered as "Graphic Arts and Printing". Standardizing the subject field names reduced the number of "other" subject fields and made our data easier to interpret.

5. **The term's syntactic formation:** Each term was analyzed for its syntactic formation, with ambiguously-formed terms occasionally being checked off as having more than one possible syntactic pattern. All terms that contained an abbreviation or acronym were also identified at this time. The full form of all acronyms and abbreviations was researched and added to the Notes section of the term record to verify that it was indeed a true term. Any term formation that did not appear among the list of available choices was entered manually using our standardized nomenclature.

We elected to represent all our term formation patterns using symbols and brackets for readability and efficiency, given the high number of lexical units to encode. The coding method we adopted is given in Table 4-2.

Part of speech	Symbol used for classification
nouns	N
adjectives	ADJ
conjunctions	CONJ
verbs	V
adverbs	ADV
prepositions	PREP
particles	PART

Table 4-2 Symbols used in TEMS to indicate syntactic patterning

For terms comprising more than two units, brackets were added to indicate which elements of the term had the closest semantic relationship, sometimes revealing the headword of a compound term. For example, the formation pattern of the term *heterogeneous computer system* was expressed as ADJ + [N + N] because the elements *computer* and *system* are a complex unit modified by the element *heterogeneous*. Longer terms sometimes contained several sets of brackets; for example, *centralized multihost shared storage* was encoded as ADJ + [ADJ + [ADJ + N]].

6. **The number of elements comprising the term:** A number between 1 and 7 was entered for each term to represent the number of elements it contained. For the purposes of this experiment, an element was considered as any string of characters with a blank space before and after it. Two elements joined by a hyphen (ex: *non-UNIX*) or slash (ex: *TCP/IP*) were considered as a single unit for the purposes of this experiment. We felt this was the best alternative because in such cases, both parts generally form a single noun or adjective.

7. **Any notes or points to remember regarding this term:** While processing the manually-scanned electronic texts, we often had observations, definitions, ideas or points to remember regarding the term being entered. All such remarks were documented in the Notes section so they were not lost and remained easily accessible beside the term to which they applied.

4.2.3. Perfecting the list

Once the entire list of manually-scanned terms was entered in TEMS, Microsoft Access was used to ensure that our data was accurate, reliable and contained no extraneous records. Through querying our list of manually-extracted terms, we identified and corrected the following list of problem areas:

- Duplicate records of the same term found in the same text were compared and then one was eliminated from the database.
- Duplicate records of the same term spotted in different texts were compared and consolidated, then one was eliminated from the database.
- Multiple records containing spelling variations of the same term were consolidated, and the variations were documented in the Notes section of the record.
- Records of the same term differing only in capitalization or the use/omission of a hyphen, comma or quotation marks were consolidated and the variations were documented in the Notes section of the record.
- Every record was checked to ensure it had at least one term formation pattern indicated and that this pattern was appropriate.
- All records that had been checked off as having a term formation pattern not listed in TEMS were verified to ensure that the new term formation pattern had been entered in the system. Conversely, all terms that had a new term formation pattern in their record were verified to ensure that the *Other Term Formation Pattern* checkbox had been ticked.

- All abbreviations and acronyms were verified that they had been noted as such in the system.
- All entries were exported to RTF format then spellchecked in Microsoft Word.

Checking this list of potential problems and correcting inconsistencies and transfer errors helped ensure that our data could be easily queried and was highly reliable for calculating statistics and comparing against the machine output from the two automatic term-extraction utilities. Once we had finalized our list of manually-extracted terms, we compiled a statistical profile of it for a better look at the list of terms the utilities would be facing. Numerous statistics were calculated for each of the electronic texts that was manually scanned for terms on its own, as well as global figures for all the texts together. A presentation of our corpus statistics is given in Appendix II of this work.

4.2.4. Special cases when coding the manually-extracted terms

As could be expected in a corpus of large magnitude, some terms we extracted did not exactly fit the encoding rules and patterns we had defined and had to be dealt with on a case-to-case basis. How we handled these exceptions, which are mostly on the graphical or character level, is described below. Once a new exception was discovered, we searched our corpus retroactively to ensure that no previously-entered term with the same particularity had been treated differently. Consistency was of capital importance when gathering our data.

Periods in abbreviations and acronyms were left untouched when entered in TEMS. However, very few terms we extracted actually contained periods; most were written as acronyms (either in uppercase or lowercase letters), even if every letter was pronounced separately. This may be attributable to the fact that these texts were all written by computer specialists and abbreviated terms in this subject field, of which there is an abundance, tend to be written without periods. This move away from period-delimited abbreviations in the field of computer science may itself stem from the fact that periods have special meaning[15] in computer code and leaving them in abbreviations could be ambiguous for the reader who associates them with a specific purpose.

Hyphenation was an unexpected and unique problem when manually scanning for terms. We noted an extremely high incidence of omitted hyphens in the scanned texts, even for terms that clearly required hyphenation. In such cases, the extracted term was coded with two separate syntactic patterns: one as it appeared in the text, and another as it should have been written with the hyphenation. The hyphens were not added manually to either the human list or the electronic text to improve scanning results or readability. We were not exposed to the other difficulty posed by hyphens because none of our texts were hyphenated manually. In other words, all hyphens that appear in our electronic texts were added for grammatical purposes and not present solely to ease text justification. Another situation we observed is that on a couple of occasions, a space was inadvertently inserted either before or after the hyphen, thereby leaving a blank space in the middle of the hyphenated term. In these particular cases, we entered the term in TEMS without the space, but did not fix the error in the electronic text. We felt we should extract the term without the space because it was clear that the additional space was unintentional and the way the term should really appear was apparent.

Terms that contained fore- or backslashes were entered with the slash and when counting the number of lexical units, the characters before the slash, the slash itself and the characters following the slash were all considered to be a single unit, for example *client/server* was coded as a single unit. In one case, we noted that the author had employed the incorrect slash in a term (a backslash entered

[15] In Microsoft Visual Basic, for example, periods serve several purposes, such as separating the name of an object and a property, or the name of an object and an associated event or method. Periods are also used to identify the distinction between an instance of a structure and one of its components, or between a class and its members. In the area of the World Wide Web, periods delimit the elements of universal resource locaters, such as www.umontreal.ca

instead of a foreslash). Once again, we left the error in the electronic text untouched and did not extract the term a second time (its properly-written counterpart with the foreslash had already been extracted earlier on in the same text).

None of the terms in our list of manually-scanned terms contained parentheses; however, a small number of terms containing square brackets were noted. These terms gave the full form of an acronym by presenting the longer form and enclosing each letter of the acronym in square brackets, such as *[m]ean [t]ime [b]efore [f]ailure*. These terms were entered as-is in the database. However, if we were creating a glossary instead of evaluating automated term-extraction software, we would eliminate the square brackets from the terms because they are only present to indicate the abbreviated form and are not part of the term itself.

Our electronic texts contained a fair number of terms that included commas. Some, but not all, of these terms were long terms of chained adjectives that separated the adjectives with commas, such as *commercial, off-the-shelf operating system* and *active content, transaction-oriented application*. Other terms with commas contained as few as three lexical units, such as *scaleable, modular system*. Our initial reaction was to remove the commas because they are unnecessary and appear to actually diminish the term's cohesiveness because the term delimitation is hampered by a series of commas within the boundaries of a single terminological unit. However, when we realized the quantity of terms written with commas, we felt it best to leave them intact because they were more significant than they would be if it were simply a case of human error or personal writing style.

We encountered a few other miscellaneous symbols while scanning the electronic texts for terms. The greater-than and less-than symbols were used to indicate a function key on the computer keyboard (ex: the *<F1> key*) and the + symbol was used as part of a proper name (ex: *the Visual C++ compiler*). Such symbols were necessarily retained in our list of manually-scanned terms. The greater-than and less-than symbols do not affect the part of speech or the way the term is pronounced, but they have semantic meaning and offer extra-linguistic information to the context because it is common practice to enclose references to the keyboard function keys along with their number between these symbols.

Some terms had single letter elements, such as *X Windows* and references to the *C* programming language, that were not joined to the rest of the term by a hyphen. These minimalist term elements were treated in exactly the same manner as their longer counterparts. No distinction was made because these single letters are lexical units, and not acronyms. They carry as much semantic information as words much greater in length and were treated as such.

No scanning of this scope would be complete without a few oddities. We found a few such isolated incidences and treated them with the special consideration and extra-linguistic knowledge that only a human terminologist could possess. For instance, one of our texts contained a high incidence of typographical errors (ex: *equivilant* instead of *equivalent*). We entered the misspelled terms in our database by their proper spelling. In one electronic text, the author once mentioned *inline help facility* instead of *online help facility*. Given that this is not a term and the text contained dozens of references to *online help*, we simply chose to ignore the newly-coined, yet erroneous term.

Now that our list of manually-scanned terms was finalized and perfected in the TEMS database, it was time to submit our texts to the two term-extraction utilities, LOGOS and ATAO.

4.3. Submission to the term-extraction utilities

4.3.1. Preparing the texts and utilities for submission

The electronic texts to be scanned were not retouched in any way prior to submission to the term-extraction utilities in order to improve accuracy or increase performance. For example, some of the electronic texts contained diagrams or charts that were in graphical form and could not be read by either term-extraction utility. The terms that appeared in this manner were left as-is for submission to the term-extraction utilities. No attempt was made to enter them manually at the end of the text so they would not be “lost” during automatic processing because in a real-life situation, time would not be spent scanning hundreds of pages of a corpus to extract text embedded in images. Furthermore, the electronic texts were not run through a spellchecker in a word processor in an effort to emulate real-world conditions as closely as possible. None of the texts used for this study were scanned copies of paper originals; so any typographical errors they contain can be attributable to human error.

Before submitting the electronic texts to LOGOS and ATAO, we concatenated them to form a single, long text and saved it as a raw text file. Furthermore, we defined our customized scanning settings for each term-extraction utility. When submitting our texts to LOGOS, we indicated that they were written in English (LOGOS is equipped to handle many different languages), that the subject field was computer science and that the default dictionary was the technical dictionary, and left the other settings at their default values. ATAO was run using all default settings except for the reduction feature, which was disabled because we are interested in the entire output, not merely which terms from our electronic texts could be added to ATAO’s pre-translation dictionary. We did not need to indicate the language of our electronic texts because ATAO only processes English texts for term scanning.

4.3.2. Machine output from the utilities

Once the automatic term scanning was completed by both utilities, data from the machine output was inputted in TEMS, with each newly-extracted term candidate that was not already present in the system (in other words, all instances of noise) corresponding to a new separate record in TEMS' database. Whenever the utility extracted a term that was part of the human list, hence already recorded in TEMS, the already-existing record was updated to include data from the machine processing instead of creating a duplicate record. This approach to storing the machine output data meant that all information on a given term candidate was conveniently stored in the same location, which reduced oversights when analyzing the results and calculating statistics.

When coding the subject fields of the machine output, we coded term candidates with a subject field whenever possible, and only used <General Vocabulary> as a last resort. Some of the noise term candidates did not clearly belong to one specific subject field, but we were able to determine the subject field because we had already read and scanned these same electronic texts ourselves beforehand. Two examples of noise that did not clearly belong to one specific subject field are *personal productivity applications* (which could be from any number of subject fields) and *significant hardware investment* (which could be coded as either computer science or handtools, depending on the sense given to the element *hardware*). From consulting the texts in which these term candidates appear, however, we were able to decide that <Computer Science Term> would be the most appropriate subject field coding for both.

Fixed expressions from LGP and extracted term candidates that had no discernible subject field, such as *Mathematics selected Hummingbird's NFS Maestro*, were assigned the subject field <General Vocabulary>.

When indicating the term formation patterns of the term candidates extracted by ATAO, the true part of speech was assigned to element comprising the term candidate after consulting the electronic text, regardless of how the term-extraction utility chose to tag the elements of the term candidate. As such, some term candidates that seemed to be plausible terms when viewed out of context in TEMS were coded as noise brought about by category ambiguity. We used this approach even when the “accidental term” extracted by the utility was in fact a valid and genuine term, but not the term being used in the electronic text. These term candidates are a coincidence and perhaps serendipitous, but not actual terms when examined in their immediate context. Consequently, a human terminologist scanning the term would not have extracted them for inclusion in a term databank. An example of this phenomenon taken from the electronic text “A Tale of Two Environments... LAN Backup to the Mainframe” by Emprise Technologies is *server technology addresses*, which is in reality an instance of noise brought about by category ambiguity when viewed in its immediate context: “Linking host and *server technology addresses* the need for cost-efficient capacity by using the mainframe environment to store the increasing mountain of network data”. We can observe that in this instance, ATAO incorrectly determined that the verb *address* in the third-person singular was a noun in the plural.

Proper names were handled differently, depending on whether it was the name of a person (*Leo Langevin*) or a corporation (*Exabyte Corporation*) or a product name (ex: *Silicon Graphics OpenGL* or *Microsoft Windows NT*). Names of individuals were coded as <General Vocabulary>; whereas names of corporations were coded as <Corporate Policy> and names of products were coded on the type of product. Since the focus of all the electronic texts submitted to the term-extraction utilities concerned computer science, virtually all the products were coded as a <Computer Science Term>.

4.3.2.1. LOGOS

When we initially received the machine output from LOGOS, it was in the form of several Microsoft Excel spreadsheet files, each with the file extension *.xls. Although we concatenated our electronic texts to form a single document prior to submission to LOGOS because we hoped to yield better-quality results (like it was supposed to with ATAO), we were forced to separate our large text into ten shorter, subdocuments because the concatenated text proved to be too long for processing. Scott Bennett of the Logos Corporation explained that this situation was attributable to the fact that we were using a version of LogosServer that was not yet fully tested and that such occurrences can often happen with pre-release software.

To begin the importation to TEMS' database, each of the comma-delimited output texts was imported into Microsoft Access using the standard import filter. Microsoft Access was chosen to complete the conversion to a TEMS-compatible format instead of another software solution or entering the machine output in TEMS' system database manually because TEMS and Microsoft Access share a compatible database format. Possibly once again linked to the sheer size of the machine output, it took several attempts and methodologies to convert the files (particularly the output generated by the longer documents submitted for term extraction) to a format easily manipulated by Microsoft Access. We were surprised by the size of the challenge this represented. Once the data had been converted, we meticulously checked the newly-created records to ensure that all the fields and data had been imported without error or omissions. After the raw machine output had been imported into Microsoft Access and thoroughly verified for accuracy, we manipulated the database structure so that the data records it contained could be read by TEMS.

At this point, we had created a database that could be read and modified by TEMS, but only contained the term candidates extracted by LOGOS. We used this opportunity to define this new database as TEMS' system database and launch TEMS using it in order to view our LOGOS output data via the TEMS interface. Each term was completed with a unique record number identifier, information on the term candidate's source, the subject field, the term candidate's syntactic formation, the number of elements comprising the term candidate, noise and silence information (see the following section for more information), as well as any noteworthy points on the term candidate.

We then completed the steps listed in Section 4.2.3 to groom the machine output for analysis, only this time we applied them to the term candidate report generated by LOGOS instead of the human list. The LOGOS-only database was then combined with the human list database and the doubles between the two databases were consolidated to form a single unit.

4.3.2.2. ATAO

When we initially received the machine output from ATAO, it was in the form of a list contained in a single Microsoft Word document. A sample of the raw output generated by ATAO appears in Appendix III of this work.

Importing the ATAO data into Microsoft Access was straightforward, especially after the experienced gained with the LOGOS raw machine output. Similar to the procedure for LOGOS, once the data had been converted to Access format, we checked the newly-created records to ensure that all the fields and data had been imported correctly, then manipulated the database structure for use by TEMS.

We launched TEMS with the ATAO-only database of term candidates and entered the appropriate information for each entry. The machine output was prepared for analysis in the same manner as the LOGOS data, then combined with the TEMS database of the human list and LOGOS term extraction data to form the final version of the database that would be used for analysis.

4.3.2.3. Noise and Silence data

Now that both automatic term-extraction utilities had processed our electronic texts and the resulting data was entered in TEMS, each TEMS term candidate record was complete and contained the following additional information:

1. **Data on Noise:** For each term candidate extracted by one of the term-extraction utilities, the checkmark was removed from the *Not extracted by* <*system name*> checkbox in the *Noise* tab to indicate this reality and expose

the second level of questions on the term extraction. If the extracted term candidate was part of the human list, the option button was left in the default selection to indicate that a valid term had been extracted and processing was complete for this term candidate. If, however, the term did not appear on the human list (thereby constituting an instance of noise on the system's term candidate report), the *No* option button was selected to enable the third array of option buttons where the assumed explanation for the error was selected[16]. If none of the proposed reasons for the instance of noise (category ambiguity, non-terminological combination, incorrect term delimitation) were deemed appropriate for the term candidate, a different reason was entered instead or, as a last recourse, *Unknown* was selected as the explanation and processing was considered complete.

2. **Data on Silence:** If the term on the human list was not extracted by a term-extraction utility (thereby constituting an instance of silence on the system's term candidate report), the checkmark was removed from the *Extracted by <system name>* checkbox in the *Silence* tab to indicate this reality and expose the second level of questions on the term extraction. If the term from the human list was not extracted by the system, the *No* option button was selected to enable the third array of option buttons where the assumed explanation for the error was selected. If none of the proposed reasons for the instance of silence (pattern not implemented in the system, new word form, absence of a given word tagging) were deemed appropriate for the term candidate, a different reason was entered instead or, as a last recourse, *Unknown* was selected as the explanation and processing was considered complete.

3. **The date at which processing for the term was completed:** A checkbox was ticked upon completion of each entry in TEMS database, which printed the system date and time in a text box underneath the Notes section. Recording the time at which the term's entry was finished made it obvious

[16] Our principal causes for noise and silence were taken from L'Homme *et al*. (1996); however, we found that the same reasons were repeated more or less in other sources we consulted.

which records were completed and which others needed further work. (Refer to Chapter 3 of this work for an example).

4.3.3. Calculating Statistics

Using the information collected in the TEMS, statistics for the level of noise and silence were calculated for each of our electronic texts, as well as global statistics for all our texts together and statistics for the overall performance of each of the systems. Other areas of our corpus that were analyzed and measured were the syntactic composition and length of the term candidates, as well as the various subject fields whose terms were extracted as part of the corpus of terms.

In the next chapter, we present our findings for noise and silence for each system, then evaluate the appropriateness of TEMS as a computerized tool for managing term candidates.

5. RESULTS OF OUR EXPERIMENT AND OBSERVATIONS

This chapter interprets and analyzes the results of the experiments we carried out in order to observe the differences between the term-extraction output generated by ATAO and LOGOS respectively. We also evaluate the appropriateness of the TEMS system as a computerized tool for gathering information on our term candidates and interpreting our results.

In a nutshell, our experiment led us to find that LOGOS had a better overall performance when extracting compound nominal term candidates from our collection of electronic texts, although ATAO had its own strong points worth mentioning. Greater detail of our research findings and analysis are laid out in the sections below.

5.1. The electronic texts

5.1.1. Linguistic quality of the electronic texts

As mentioned in Section 4.2.1, efforts were made to only select highly-technical electronic texts from various branches of the field of computer science for the purposes of this study. We looked for white papers in electronic form from known companies and multinational corporations because we assumed they would release carefully proofread, well-composed literature by professional writers. Instead, we were surprised by the overall mediocre diction of the electronic texts.

- The same text would have two spellings for certain words (*imbedded* vs. *embedded*), and sometimes use a hyphen, sometimes two words and others times only one word to express the same concept, even within the bounds of a single paragraph (ex: *multi vendor* vs *multi-vendor* vs. *multivendor*).

- Some texts had inverted words and spelling mistakes that were detected by the spellcheck utility of our word processor, such as *writting data* instead of *writing data* and *equivilant* instead of *equivalent*. It must be stressed that we did not correct the spelling or typographical errors in our electronic texts before submission to LOGOS or ATAO.

Consistency in the texts was very low in general, although some texts we scanned were better than others. It was obvious to us that these texts were not written or at least revised by professional technical writers because language professionals would not have allowed errors of this magnitude to pass.

Regardless of the relative literary quality of the input documents and the errors they contained, this situation is more than likely typical of the quality of internal corpora submitted to term-extraction software such as LOGOS and ATAO. Moreover, we cannot report that it skewed our results. Although the term-extraction utilities may have missed some genuine terms because of typographical errors, spelling mistakes and variations in hyphenation, this type of error occurred in all our texts to a certain extent. The observed “consistent inconsistency” confirms our belief that this is indeed representative of the type of input text routinely fed to term-extraction programs as part of their daily processing. In retrospect, it might even be fortunate (i.e. realistic) that we did not use picture-perfect texts for our research because it is assumed that the texts used to benchmark other term-extraction utilities also contained similar errors.

5.1.2. Other subject fields

In addition to terms from the field of computer science, our corpus contained lexical units from a variety of different subject fields, with a total of 24 disciplines. Our input texts touched on such diverse areas as engineering, printing machines and equipment, economics and corporate policy, although the overwhelming majority of the non-computer science vocabulary we identified was from the field of *Printing and Graphic Arts.* Appendix II of this work contains the complete listing of the subject fields found in our corpus. A pie chart illustrating the breakdown of the other subject fields encountered in our corpus of extracted terms is given below.

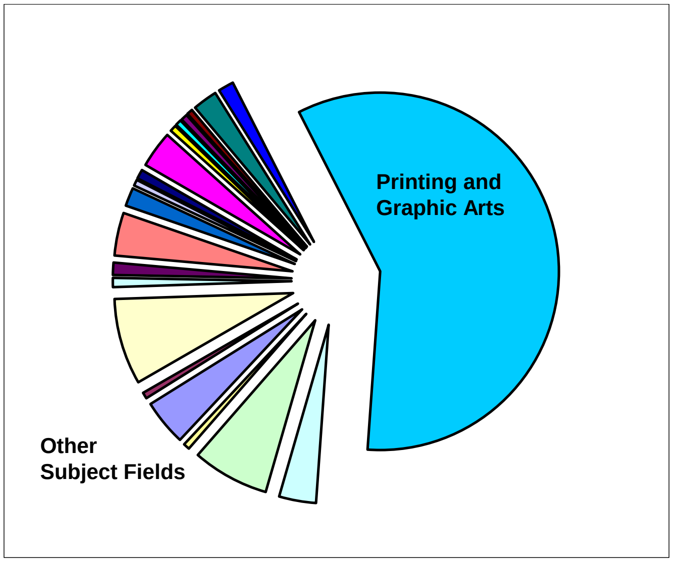


Figure 5-1 Other subject fields identified in our corpus

The wide array of subject fields is understandable because our white papers were by and large more than simply programmer’s guides containing no contact with outside subject matter or issues.

5.1.3. Terminological content of the electronic texts

The percentage of compound and simple terminological units identified in the electronic documents varied between 3% and 6% (see Figure 5-2 for a visual representation of the different percentages). This rather significant margin might be attributed to the fact that not all the electronic texts used to create our corpus of extracted terms shared the same purpose. Some of the texts were highly focused and developed one idea in great detail; whereas others were longer and discussed broader concepts and compared all the possible solutions to a given situation, thereby introducing far more terminology in the text. An example is the RAID FAQ versus the white paper by Strategic Research Corporation discussing data warehouse performance tuning. The RAID FAQ limited its scope to discussing the five RAID levels in detail and comparing them against one another. Since there is little difference between these five levels, the article contained fewer terms than eight of the other electronic texts used for this study. However, for the most part, these same terms had an extremely high usage frequency. This approach differs considerably from that adopted by the authors of the data warehouse performance tuning paper, who elected instead to summarize a number of contrasting solutions to the performance issues raised in the electronic text. Results from the scanning by both LOGOS and ATAO indicate a corresponding increase in the number of term candidates identified in this particular text, even when the document length is taken into consideration.

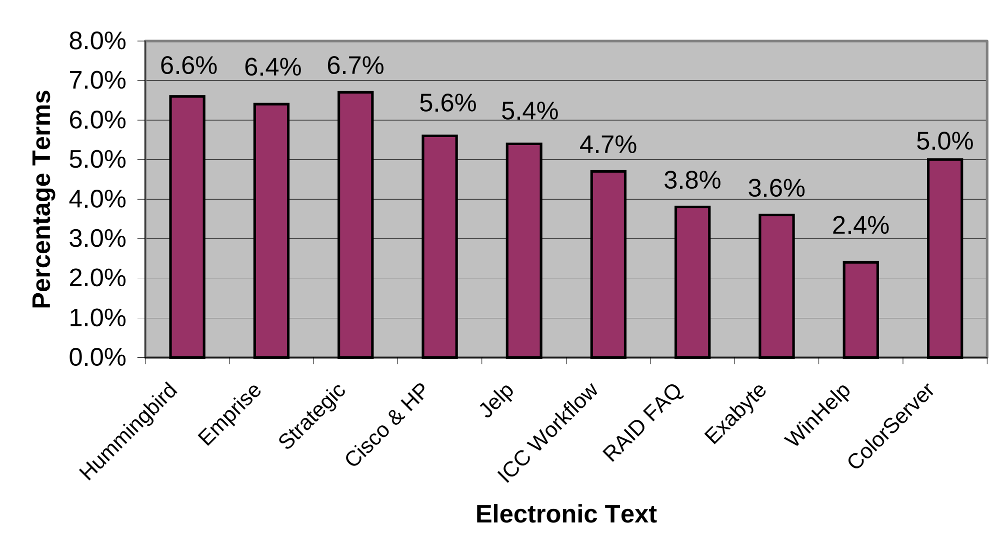


Figure 5-2 The percentage of terminological units contained in the texts submitted to the automated term-extraction utilities

The statistic given by Hannan (1996:133), whereby 40% of two-element terms are of the ADJ + N formation held perfectly to the electronic texts we scanned manually before submission to LOGOS and ATAO. Furthermore, the figure of 6% given by Lauriston (1994:162) for the percentage of terms in his 8500-word text applied to our texts of similar length, with the exception of Delphi Corporation's white paper "Evaluating RTF-based Online Help for Microsoft® Windows® 95 and NT". This text was somewhat different from the other electronic texts we scanned in that it contained extensive amounts of marketing and historical information, disclaimers and lengthy summaries. If these sections are discarded and the percentage of terms recalculated, it too approaches the 6% recorded by Lauriston. These observations confirm our belief that the electronic texts were scanned accurately and that our electronic texts were typical with regards to terminological content. Consult Appendix II for many other statistics on our corpus of manually-extracted terms.

5.2. ATAO

5.2.1. Processing ATAO's raw output

As stated in Chapter 2 of this work, ATAO sorts its list of complex nominals by the last element they contain because it is supposed to present conceptually-similar entries together in the list (an example of this list is given in Appendix III). We found this powerful design feature of the complex nominals list to be extremely helpful when processing our machine output from ATAO, particularly when assigning the subject field tags to the term candidate records in TEMS' database.

We observed that ATAO's parsing engine did not always function as expected. This meant, for example, that ATAO's raw output would contain the same term candidate with and without the *S* for pluralization. We purged dozens of duplicate term candidate records from TEMS' database when it became apparent to us that ATAO had left some term candidates on its report twice or even three times with various combinations of the plural and singular.

Sometimes ATAO's lemmatizer trimmed two instances of the same term candidate differently when reducing it to canonical form: we extracted both *format offers help author new feature*<u>*s*</u> and *format offers help author*<u>*s*</u> *new feature* from the same text and do not understand why. Another example of this occurrence is *level use multiple disk*<u>*s*</u> versus *level use*<u>*s*</u> *multiple disk*.

5.2.2. Calculating ATAO's success rate

When we originally calculated ATAO's proficiency, we arrived at a figure of 36.4% (meaning that ATAO correctly extracted 36.4% of the terms present on the human list), which we found to be both rather disappointing and well below the term-extraction performance figures advanced by L'Homme *et al.* (1996:303), Lauriston (1994:163) and Otman (1991:94) for other automated term-extraction systems.

We then recalculated our statistics after having removed the simple terms from the human list. We judged that this was only fair because ATAO does not attempt to extract simple terms, which are beyond the scope of this study. Simply limiting the human list to compound terms increased ATAO's percentage accuracy to 41.9%. Further refining the human list by culling the terms we extracted from image objects (table captions, labels and the like) embedded in the electronic texts submitted to the two automated term-extraction utilities raised ATAO's final accuracy for our collection of electronic texts to 42.7%. We felt that culling the terms found in embedded image objects was necessary and reasonable because neither ATAO nor LOGOS could scan the rasterized text. In other words, we removed these terms from the human list because they were not in machine-readable format, hence both term-extraction systems understandably passed over them.

In his evaluation of the Nomino term-extraction system used for processing French texts, Lauriston (1994:163) used what he referred to as both *rigorous criteria* and *tolerant criteria*[17] for determining whether or not an extracted term candidate was considered to be a match for an entry on the human list. Our criteria for assessing whether or not ATAO's term candidates corresponded to an

[17] On the same page of his article, Lauriston states that, "The first yardstick, a tolerant standard, considered as acceptably recognized any complex term of which at least one constituent was detected, regardless of any extraneous words taken by the system as forming part of the term. The second yardstick, a rigorous standard, considered as recognized only those complex terms that were fully detected, e.g., in their entirety and without extraneous lexical items".

entry on the human list fell in the middle-ground between Lauriston's two criteria types, but held closer to his rigorous standard.

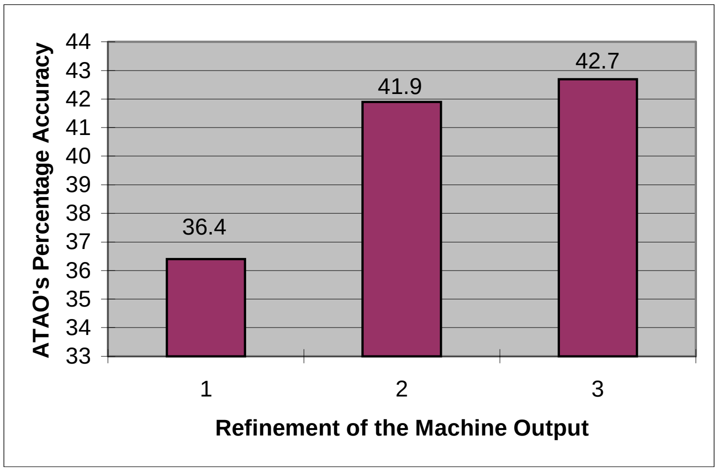


Figure 5-3 ATAO's percentage accuracy through refinement of its machine output

For example, we gave ATAO the benefit of the doubt when evaluating term candidates that were the same as entries on the human list except for a hyphen (ex: *cross-platform file sharing* extracted as *cross platform file sharing* by ATAO and *run-time environment* extracted by ATAO as *run time environment*) or the letter *S* that was frequently dropped to lemmatize inflected term candidates (ex: *actual working conditions* extracted as *actual working condition* by ATAO and *network news client* extracted by ATAO as *network new client*). We did not, however, permit extraction errors that involved numbers, whether written out in full text or appearing as digits (ex: *16 bit Windows application* that was extracted by ATAO as *bit Windows application*). Two different approaches were necessary here because the letter S truncated from the end of words, hyphens and apostrophes were errors committed by the parser when reducing the term candidate to its canonical form; whereas the way numbers and digits are processed is handled earlier on in the process when analyzing the syntax of the sentence. This study does not attempt to evaluate the proficiency of the system's parser, only the system's quantitative results for noise and silence.

Overall, we feel that the percentage accuracy of 42.7% valid terms is satisfactory for the machine output generated by ATAO. We concede that our results report 5.9% more noise than those gathered by L'Homme *et al.* (1996:303); however, we believe that this slight performance variation can be attributed, at least in part, to the fact that our electronic texts were more technical than those used by L'Homme *et al.* Furthermore, being highly technical, our electronic texts had a different, more specialized target audience.

Upon completion of their analysis of ATAO's corpus of term candidates, L'Homme *et al.* observed that their term candidate report contained 48.6% valid terms, i.e. correct forms that were extracted for the human list, and 51.4% erroneous forms (noise), i.e. non-terms that were not part of the human list. Our research on ATAO yielded a term candidate report containing 40% valid terms and 60% noise, as shown in Figure 5-4.

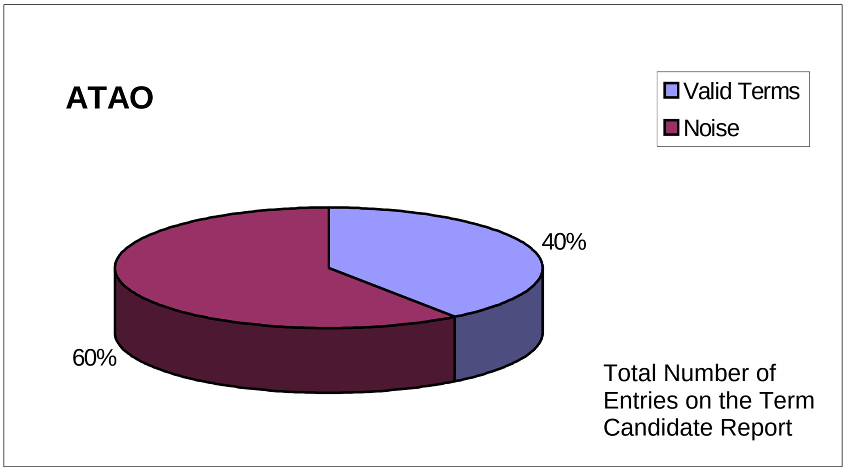


Figure 5-4 ATAO's percentage accuracy after refinement of the machine output

5.2.3. Output generated by ATAO

ATAO was designed to list coordinate terms (ex: *TCP/IP utilities and applications*) intact as a single unit on the complex nominals list. This system characteristic was taken into account when calculating silence for ATAO because these nested structures were present on the term candidate report, only not quite in the form originally expected. Nested terms were manually split apart and entered in separate records in TEMS' database to remain consistent with the treatment of nested terms for the human list. *System font and colour* is an example of one of the many nested terms that were extracted by ATAO as a single unit, but identified as two separate terms for the human list. This approach to coordinate terms can be considered acceptable because, after all, the machine output generated by

ATAO is designed to be filtered by a human terminologist. In some respects, this behaviour is simply a compromise that reveals ATAO's inability to determine that the term candidate in question is two terms joined together and not a single unit.

Similarly, we noticed on more than one occasion that ATAO extracted the same two terms containing the conjunction *and* as one of their elements, but in reverse order (ex: *Windows NT workstation and server* versus *Windows NT server and workstation*). We are still uncertain whether we feel this constitutes an error or merely thoroughness on the part of ATAO that entails additional filtering time for the human terminologist.

An appreciable amount of the noise generated by ATAO was caused by proper nouns (both people's names, as well as those of products and corporations) and numbers being improperly handled (for instance, *16 bit Windows application* was extracted without the number). Some, not all, of these cases might have been avoided if the authors of our electronic texts had used hyphens to indicate the adjectival use of the digits in the affected terms that were overlooked by the term-extraction software.

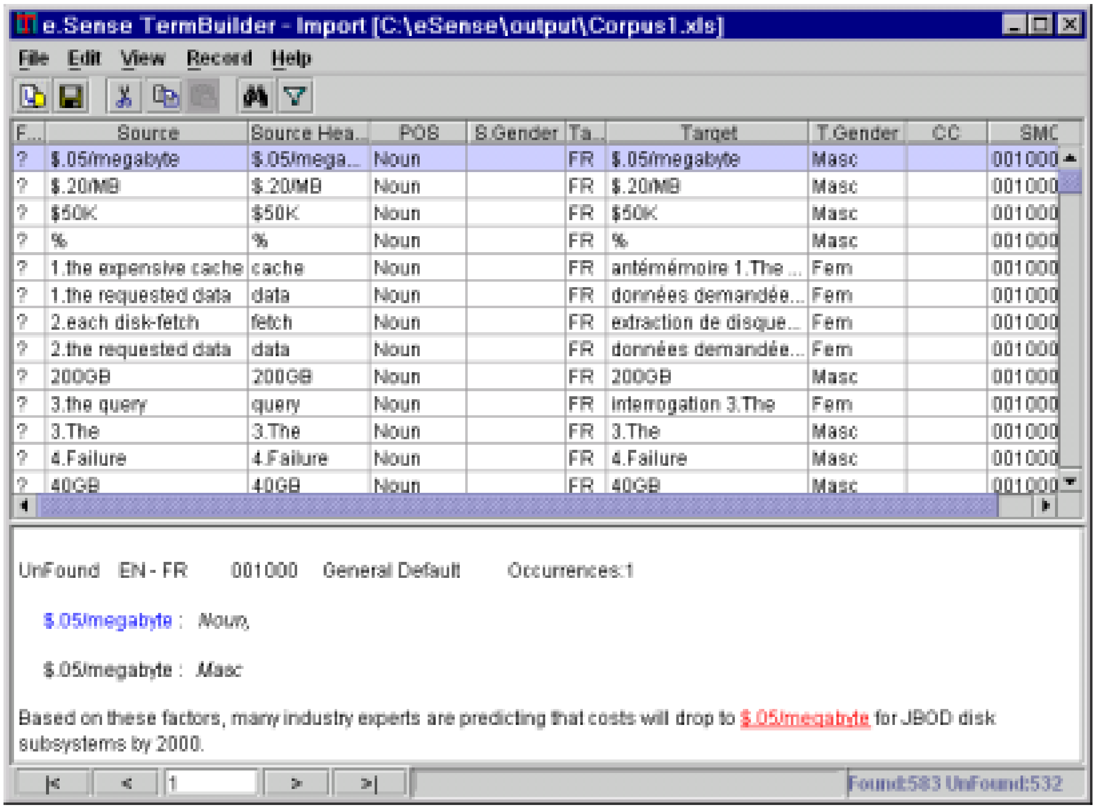
Furthermore, we observed that ATAO systematically removed the hyphens from term candidates, which complicated merging ATAO's raw output with the human list. In cases where the same term appeared in the electronic text both with and without the hyphen, ATAO usually extracted it both ways, then removed the hyphen from one term candidate. This action had the side-effect of creating even more duplicates in the term candidate report generated by the system.

Another issue we encountered was the letter *S* dropped by the lemmatizer from the name of the operating system, *Windows*, which normally would be retained because it is a proper name. This new canonical form made it so that *window* and *windows* appeared in the same format on the term candidate report generated by ATAO. By making it impossible to distinguish between single and plural nouns, we were obliged to double-check many term candidates in the original context to ensure we did not falsely code valid terms as noise or vice-versa.

5.3. LOGOS

5.3.1. Processing LOGOS' raw output

Once LOGOS has finished scanning its input texts for term candidates, the raw machine output is customarily viewed by the human terminologist in a dictionary data maintenance program called *TermBuilder* for potential inclusion in the LOGOS Client Dictionary. However, we imported our data directly in TEMS to imitate the conditions used when evaluating the raw output produced by ATAO. A screen capture of the *TermBuilder* interface appears below.



F...	Source	Source Hea...	POS	S.Gender	Ta...	Target	T.Gender	CC	SMC
?	$.05/megabyte	$.05/mega...	Noun		FR	$.05/megabyte	Masc		001000
?	$.20/MB	$.20/MB	Noun		FR	$.20/MB	Masc		001000
?	$50K	$50K	Noun		FR	$50K	Masc		001000
?	%	%	Noun		FR	%	Masc		001000
?	1.the expensive cache	cache	Noun		FR	antémémoire 1.The ...	Fem		001000
?	1.the requested data	data	Noun		FR	données demandée...	Fem		001000
?	2.each disk-fetch	fetch	Noun		FR	extraction de disque...	Fem		001000
?	2.the requested data	data	Noun		FR	données demandée...	Fem		001000
?	200GB	200GB	Noun		FR	200GB	Masc		001000
?	3.the query	query	Noun		FR	interrogation 3.The	Fem		001000
?	3.The	3.The	Noun		FR	3.The	Masc		001000
?	4.Failure	4.Failure	Noun		FR	4.Failure	Masc		001000
?	40GB	40GB	Noun		FR	40GB	Masc		001000

UnFound EN - FR 001000 General Default Occurences:1

$.05/megabyte : *Noun,*

$.05/megabyte : *Masc*

Based on these factors, many industry experts are predicting that costs will drop to $.05/megabyte for JBOD disk subsystems by 2000.

Figure 5-5 LOGOS dictionary data maintenance program, *TermBuilder*

One field included in the LOGOS term candidate report that was absent on the report generated by ATAO was the immediate context of each extracted term candidate. When importing the LOGOS raw machine output in the TEMS system, we transferred the immediate context to the *Item Notes* field of each term candidate record. Having the immediate context readily at hand proved to be both handy and a significant time saver when assessing the terminological status of the term candidates identified by the system. We were not obliged to wade through pages of text to locate the rest of the sentence from which the term candidate was extracted because it was already present in TEMS' system database.

5.3.2. Calculating LOGOS' success rate

Given that we refined ATAO's machine output to improve the system's percentage accuracy (see Section 5.2.2 for a detailed description of the refinement process), we made the same refinements to the LOGOS machine output to promote consistency and an environment for fair comparison. The initial percentage accuracy recorded for LOGOS was 52.3%, which translates to LOGOS correctly extracting 52.3% of the terms present on the human list. This figure is higher than that recorded by ATAO, but plausible all the same. Limiting the human list to compound terms boosted LOGOS' accuracy to 60.2% — a rise of 7.9%. This increase is in line with the 5.5% observed for ATAO. The slightly higher figure for LOGOS indicates that this system extracted more valid terms initially and that the percentage increase is proportional to its original accuracy figure. Culling the terms from the human list that were embedded in graphics, charts and other objects that were not readable by either system further increased LOGOS' score by another 1.4% to attain a final percentage accuracy of 61.6%.

We observed that the parser used by the LOGOS system to reduce its inflected term candidates to canonical form was far more accurate than that employed by ATAO. As such, we identified far fewer errors by the system that were traceable to the parser lemmatizing the extracted term candidates (hyphenation, apostrophes and pluralization) compared to ATAO. To remain consistent, any error we believed was caused by an incorrect parse of the sentence in the source text was recorded as noise; whereas errors due to lemmatization were overlooked.

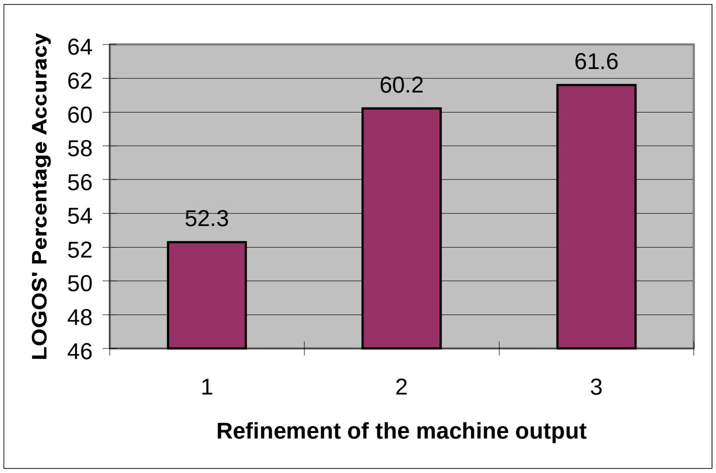


Figure 5-6 LOGOS' percentage accuracy through refinement of its machine output

A final percentage accuracy of 61.6% was better than what we had hoped for and higher than the percentage recorded for ATAO; however, this variation is not substantial enough to have us reconsider our methodology for calculating the percentage accuracy. Once again, the higher level of technical content in the electronic texts we chose to use for our research explains in part the difference between our scores and those reported by L'Homme *et al.* for the same term-extraction system. Furthermore, the marked improvement demonstrated by LOGOS might be attributed to the fact that we did not use the same legacy version of the software as L'Homme *et al.* to analyze our collection of electronic texts[18].

When L'Homme *et al.* analyzed their corpus of term candidates generated by LOGOS, they found that the system's term candidate report contained a total of 32.9% valid terms, i.e. correct forms identified for inclusion on the human list, and 67.1% erroneous forms (noise), i.e. non-terms that were not part of the human list. Our research with the LOGOS system yielded machine output with results of 24.1% valid terms and 75.8% noise respectively, as shown in Figure 5-7.

[18] After L'Homme *et al.* had completed their research on LOGOS, the system underwent a complete overhaul and the source code for several areas was totally rewritten.

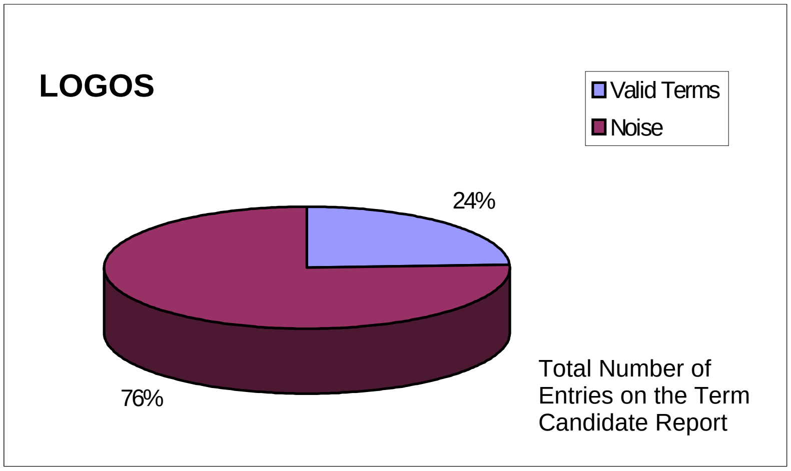


Figure 5-7 LOGOS' percentage accuracy after refinement of the machine output

5.3.3. Output generated by LOGOS

When processing the term candidate report produced by LOGOS, we remarked that the system often truncated term candidates at unusual places, for example *restricted run-time environmen* (trimmed before the final *T* on *environment*). This is probably a programming limitation, i.e. a limit of 512 characters per sentence; however, it created some curious-looking term candidates that would require manual correction by the human terminologist while filtering the machine output. Overall, this is the most noteworthy point observed while examining LOGOS' term candidate report.

In contrast to its tendency of trimming some term candidates prematurely, LOGOS often extracted lengthy chains of lexical units for its machine output report. We found unwieldy strings containing over a dozen elements on LOGOS' term candidate report, such as *table 1 test matrix platforms Platform*Operating SystemBackup application Software1.Sun Ultra1Solaris 2.5.1legato*. These entries seem rather exaggerated because even the longest noun phrases on the human list contained at most seven units. A possible improvement to the system might be to limit the maximum number of units allowed per term candidate to a more feasible number, such as nine or ten units.

Lastly, LOGOS was observed to systematically remove the quotation marks and capital letters from the term candidates it extracted. In some instances, this treatment might result in a certain loss of information if the capital letters or quotation marks were added to indicate an inhabitual semantic relationship between the term elements involved.

5.4. Contrast of the systems' performance

5.4.1.1. Bédard's areas for potential duplication

In a table presented in Section 4.2.2, Bédard (1992:754) notes nine situations that can lead to redundancy in the term candidate report generated by an automated term-extraction program. He estimates that these circumstances represent the primary areas for duplication and variations in the corpus of extracted terms. The following is a summary of how LOGOS and ATAO handled each of these points when preparing their term candidate reports. This listing is complemented by the information on the output produced by LOGOS and ATAO given in Sections 5.2.3 and 5.3.3 respectively.

- **The presence or absence of a hyphen when the term retains the same function and part of speech**

  This point was difficult to evaluate for ATAO because the system usually stripped the hyphens from term candidates while lemmatizing. As such, ATAO's term candidate report contained several doubles. However, we were unsure in certain instances whether the system had extracted the term candidate's unhyphenated form twice and skipped the hyphenated form or extracted the term candidate once with the hyphen and once without, then lemmatized them to make them appear the same. One example is the term *cross-platform support*, which appeared throughout its electronic text with and without the hyphen in approximately equal proportions. It was one of many such terms that were extracted multiple times by ATAO, thereby creating a number of duplicates on its term candidate report.

Terms containing the element *third-party* were a special case for ATAO. *Third-party* was represented with and without a hyphen in a ratio of 3:4 in our electronic texts[19]. Regardless of the presentation of the term, not once did ATAO extract it with the hyphen for the list of term candidates. See the following section on how numbers and digits were handled for more information on the issue of *third-party* terms.

General-purpose operating system, *high-speed backup* and *front-end application* are examples of the many term candidates ATAO extracted, then removed the hyphen before adding them to the term candidate report. These terms were only extracted once and did not create duplicates on the term candidate report.

LOGOS retained all hyphens when identifying term candidates. It also extracted the same term with and without the hyphen, creating doubles on the term candidate report. In sum, hyphens were not an issue for LOGOS.

- **Whether two units are fused together or simply juxtaposed**

  This reality (ex: *stand alone* versus *standalone*) was not found in our collection of electronic texts submitted to LOGOS and ATAO. Consequently, we do not know how either system would react when faced with this area for duplication. Furthermore, because ATAO only extracts compound units for its term candidate report, any two-unit terms that were fused to become simple terms would necessarily be skipped by ATAO when scanning the input texts.

[19] We found it interesting to note that the authors of the electronic texts were more likely to add hyphens to their writing if the term was long.

For example, *third party* is written without a hyphen in the following instances:

- *third party vendor*
- *third party technology*
- *third party supplier*
- *third party software vendor*
- *third party enabling technology*

However, as the term increases in length, hyphens are included:

- *Windows NT third-party internetwork connectivity solution*
- *third-party terminal emulation software*
- *third-party PCX server software*
- *third-party X Windows encapsulation technology*

It would appear as though the authors do not realize that the two lexical units are in fact a single adjective until the term becomes more complex.

- **Random use of endings for modifiers**

  A fitting example of this potential area for duplication taken from our collection of electronic texts submitted to LOGOS and ATAO is *symmetrical multiprocessing* and *symmetric multiprocessing*. ATAO failed to extract *symmetrical multiprocessing*, although it did extract *symmetric multiprocessing* for its term candidate report. LOGOS, on the other hand, correctly extracted both terms from the human list when processing the same electronic text.

  Going out on a limb, we could advance that the terms *revenue-enhancing activity* and *revenue-generating activity* are also of this type because they are similar endings for the modifier *revenue*. ATAO did not extract either one of these terms from the human list, however, and LOGOS only extracted the latter for its term candidate report. We are uncertain why *revenue-enhancing activity* slipped by both systems.

- **The use of variable elements, often numbers, inside frequently-occurring strings**

  Numerical term elements, including digits, ordinal and cardinal numbers, appeared to pose a particular problem for ATAO. A flagrant example is the series of term candidates containing the element *third-party*. Not once did ATAO extract *third* with *party*, leaving a total of 17 incorrectly-delimited term candidates, among them *party software product*, *party help development tool*, *party enabling technology* and *party vendor*. In contrast, LOGOS had little trouble with these items, extracting 13 of the 17 *third-party* terms.

  The handling of digits in term candidates, such as *64-bit application support* and *3-D visualization tool*, is discussed in Section 5.2.3.

- **The presence or absence of certain punctuation marks from one instance of a lexical unit to the next**

  ATAO correctly extracted the term *"write-only" data warehouse* from the collection of electronic texts with the quotation marks intact. As such, we can conclude that ATAO does handle quotation marks appropriately. However, of the seven term candidates containing quotation marks that were extracted by ATAO, only two were on the human list, leading us to the observation that the system’s performance was hindered, at least to a certain extent, by the presence of quotation marks in the input text.

  LOGOS, on the other hand, stripped quotation marks from the term candidates on its machine output list. This action by the system obliged us to double-check its term candidate report to ensure we did not code any valid terms as silence simply because they did not appear in precisely the same format as the terms on the human list. In some instances, this handling of quotation marks in the term candidates might result in a certain loss of information, such as the newness indication for neologisms or allusions to humour or sarcasm.

The term candidate report generated by ATAO contained a total of 52 entries with single apostrophes indicating possession, such as *applet's specific need*, indicating that they did not hinder extraction by the system. However, not one of these terms was on the human list, which translates to a considerable amount of noise added to ATAO’s term candidate report. LOGOS followed suit by extracting 128 term candidates with apostrophes, none of which were on the human list.

Both LOGOS and ATAO had a tendency to append a possessive acronym like *NT’s* when extracting term candidates. This means that the machine output for both systems contains many terms such as *NT’s preemptive multitasking*, when the actual term is *preemptive multitasking*.

- **Typographical errors, homonyms or transfer errors that may slip into a source text**

  The two spelling errors we found in our collection of electronic texts, *Equivilant Data Availability* instead of *Equivalent Data Availability* and *inline help facility* instead of *online help facility*, were treated differently by ATAO: the first was identified by the system, but the second went undetected. It is unclear to us why ATAO failed to identify *inline help facility* because it was able to successfully extract the similar term *online help engine*, as well as 18 other term candidates from the same electronic text containing the element *online*. LOGOS, on the other hand, managed to extract both these erroneous terms for its term candidate report.

  There was a typographical error in the term *fault tolerant disk array*. (In the original text, the term appears as *fault tolerant3 disk array*). This term was not extracted by ATAO, although we cannot determine whether or not the digit inadvertently placed in the term was the cause of the oversight. LOGOS extracted this erroneous term with the digit left untouched at the end of *tolerant*. This behaviour on the part of each system is in keeping with the manner in which numbers and digits were handled in general.

  ATAO extracted both *writing data* and *writting data* from one of the electronic texts; however, LOGOS only extracted *writting data*, perhaps because it identified *writing* as a verb and not a noun.

  A spacing error was made in the term *Windows Ntplatforms*, fusing together the elements *NT* and *platforms* as a consequence. This situation was exacerbated by the fact that most word processors today correct atypical capitalization, and the software program used to compose this text switched the *T* in *NT* to a lowercase letter. ATAO did identify this term for its term candidate report, in addition to extracting similar term candidates, *native Windows NT application* and *Windows NT desktop*. LOGOS did not extract this erroneous term candidate; however, we cannot judge if it were aware of the spacing error or not.

A word inversion error occurred in one of the electronic texts. The sentence reads, "to develop a of list critical considerations," instead of "to develop a list of critical considerations”. We predicted that the term-extraction utilities would be tempted to extract *list critical considerations* instead of the true term *critical considerations* because of this mix-up. LOGOS did in fact extract this erroneous term, but ATAO did not. It is possible that the ambiguous term element *list* was identified as a verb by ATAO instead of a noun, causing it to pass over the element. The true term, *critical considerations*, was not extracted by either system, resulting in another instance of silence for each.

- **Spelling variations between different dialects of the source language which can be found from one document to the next or even within the same text**

  Only one of our electronic texts was written in a variety of English other than North American English. Spelling variations such as *imbedded* versus *embedded*, *scalable* versus *scaleable* and many more were detected in this text from Ireland. ATAO and LOGOS extracted all spelling variations without either converting the spelling to the North American English convention or skipping entirely term candidates that had already been extracted, but with the other spelling convention.

- **The use of dashes, whether they are em dashes or en dashes, and whether or not a space is inserted between the dash and the text it offsets**

  This reality was not found in our collection of electronic texts submitted to LOGOS and ATAO. Consequently, we do not know how either system would react when faced with this area for duplication.

- **Random variations in how otherwise identical strings are presented in the same text**

  Secure, real-time, embedded system and *scaleable, modular system* are two examples of terms from the human list that escaped LOGOS and ATAO solely because of the commas separating the adjectives they contain. ATAO did extract the term *real-time embedded system*, which appeared later on in the same text without the hyphen, but LOGOS failed to identify this term on both occasions. The fact that they were unable to extract these terms is not surprising because we believe that no currently available utility could determine that they were valid terms without extra-linguistic information.

  Another example of punctuation making term identification difficult is *[m]ean [t]ime [b]efore [f]ailure*. This term was challenging for the utilities because of the brackets showing the acronym's formation. This term slipped by ATAO, but was extracted by LOGOS.

The term *end user* appeared most often written with a hyphen in our collection of electronic texts; however ATAO ignored the relative frequency of the two and removed the hyphen when extracting this term for the term candidate report. LOGOS extracted this term candidate in both forms.

Neither *<F1> topic display*, nor *<F1> context-sensitive help* were extracted by either system, although they did identify *context sensitive help* (with and without the hyphen) for their term candidate report. Pointy brackets are uncommon in terms; so this behaviour by the systems is understandable.

5.4.1.2. Lauriston’s list of noise- and silence-producing factors

Lauriston (1994:164) listed acronyms, capitalized LGP nouns and how the terms were presented in the input text (such as nested terms interrupted by coordinate conjunctions, fore- or backslashes and punctuation marks) as the three principal causes for Nomino to fail, that is, the three main silence-producing factors. Noise, on the other hand, was generally caused by either an incorrect parse resulting from category ambiguity or incorrect term delimitation (syntactic noise), or non-terminological LGP combinations being extracted as term candidates. Below is an overview of how ATAO and LOGOS handled these silence-producing factors when preparing their term candidate report. This summary is complemented by the information on the output produced by ATAO and LOGOS given in Sections 5.2.3 and 5.3.3 above, as well as the points mentioned in Section 5.4.1.1 addressing Bédard’s areas for potential duplication. Lastly, noise-producing factors are covered in Section 5.4.2.

It appears that acronyms were handled differently by the two automated term-extraction systems because LOGOS was able to correctly identify approximately one and a half times as many acronym terms as ATAO. See Figure 5-13 for a graphical representation of the results obtained by both systems. When both LOGOS and ATAO encounter an unknown lexical unit in the input text, they tag it as a noun when parsing the sentence. We observe in Figure 5-13 that LOGOS systematically extracted more term candidates with the N + N syntactic pattern. In keeping with this finding, we remark that the comparative results for acronym terms bear a striking resemblance to those for terms with the N + N formation pattern.

Closely related to the issue of acronyms, capitalized LGP nouns were detected throughout our collection of electronic texts. LGP nouns were often capitalized as paragraph titles (ex: *Related Papers*) and table headings (ex: *New Macro*). Names of products (ex: *HP VirtualVault*) and people (ex: *Michael Peterson*) were also noted. We suppose that the proper names not in the system dictionary were taken to be nouns by the term-extraction systems, which explains in part their high incidence on the term candidate reports. One instance of an LGP noun phrase being extracted presumably because it was written with a capital letter is *Old source file*. However, the capital letter in this case was only added because this non-term appears at the beginning of a sentence. Quantitative performance results when extracting capitalized LGP nouns were noted to be similar to those obtained for acronym terms.

Terms containing foreslashes, such as *load/unload time*, *TCP/IP Server Suite* and *intelligent I/O channel*, were not a difficulty for ATAO. The system extracted 40 units with slashes, as compared to 38 identified for the human list. However, only approximately one-quarter of the terms containing foreslashes on the human list were detected by ATAO (75% noise is below-average performance), meaning that slashes could be considered a minor hindrance when processing the input texts. LOGOS, on the other hand, extracted an impressive 91 term candidates containing the slash character for its term candidate report. Of these, only 15 were present on the human list of manually-extracted terms, which leads us to believe that the slash character did indeed hamper the systems' analysis of the electronic texts. Only one (erroneous) incidence of a backslash was detected in our collection of electronic texts, *TCP\IP application suite*, and it was extracted by ATAO and LOGOS.

A detailed discussion of how punctuation marks were handled by the two automated term-extraction systems is given in the last point of Section 5.4.1.1.

Two of the primary causes of noise, category ambiguity and incorrect term delimitation are treated below in Sections 5.4.2.1 and 5.4.2.3 respectively.

5.4.2. Noise

As stated in the introduction section on this topic, automated term-extraction systems generate noise when they extract term candidates for their machine output report that would not normally be retained by a human terminologist. Refer to Section 1.3.4 of this work for background information on the definition of noise, as well as a typology and the principle noise-producing factors.

Overall, when analyzing our corpus of term candidates, we found that deciding how to encode the systems' performance was neither easy nor clear-cut. Moreover, we remarked that the noise generated by both ATAO and LOGOS was more difficult to code than genuine terms. This only stands to reason because noise is often ambiguous or exhibits characteristics from more than one type of noise. Naturally, calculating statistics purely based on the systems' quantitative performance was relatively straightforward.

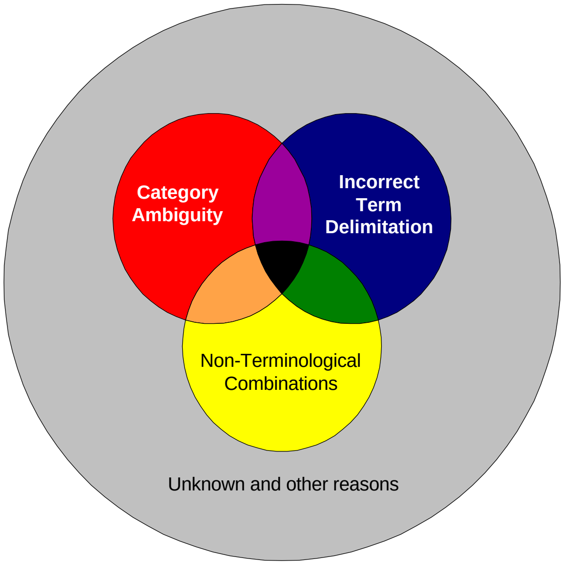

Category Ambiguity
Incorrect Term Delimitation
Non-Terminological Combinations
Unknown and other reasons

Figure 5-8 The causes behind terminological and syntactic noise

It must be said that we are only making a so-called “educated guess” when we indicate in TEMS the reasons behind the systems’ extraction errors. Occasionally, the causes for the systems to fail were evident. More often, however, there would be two conceivable reasons why the systems extracted certain term candidates incorrectly. For example, there were often cases of incorrect term delimitation, but the term the system was attempting to delimit was a non-terminological combination. In such situations, we could only indicate what we believed was the most probable cause without being 100% certain of our choice. Similarly, we regularly found that category ambiguity caused the system to consider a verb at the end of a term candidate as a noun. This situation made the system extract a non-terminological combination, thereby creating a joint category ambiguity/non-terminological combination error. A diagram illustrating the inter-relationship between the causes of noise appears in Figure 5-8.

Given the complex correlation linking the noise-producing factors and the subjective nature of this type of observation, our findings for how and why the systems failed to extract certain valid terms from the human list should be taken with a grain of salt.

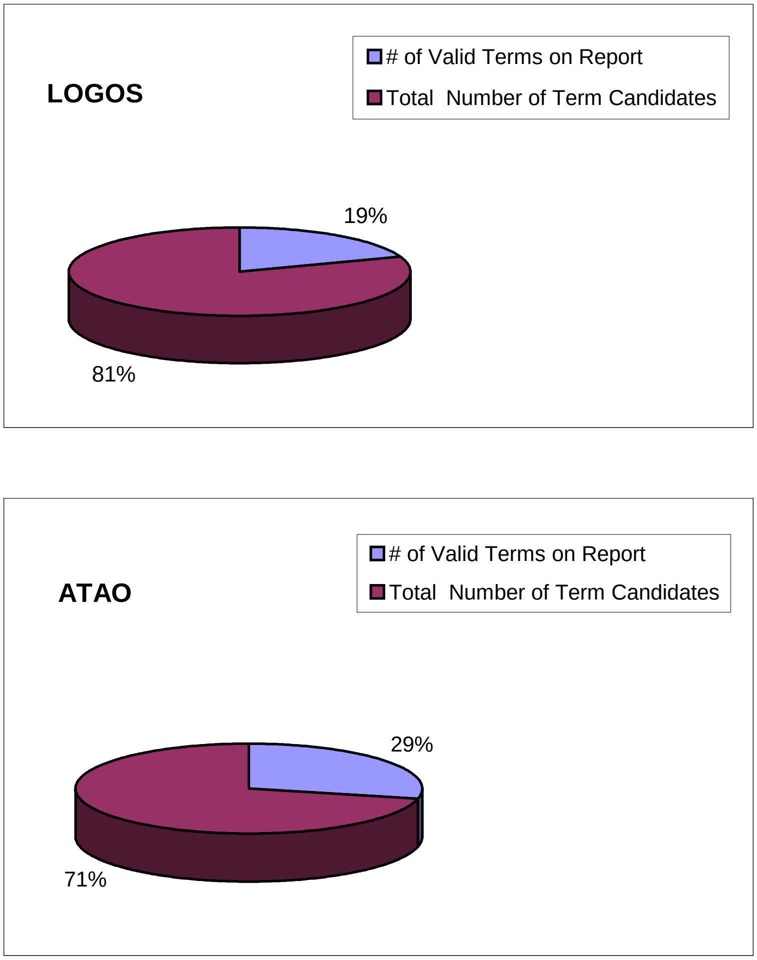


Figure 5-9 Percentage of valid terms as compared to the overall number of extracted lexical units

After completing our calculations on the percentage of valid terms extracted by each system, we learned that LOGOS had extracted far more valid terms on the human list than did ATAO. This being said, we also noticed that the machine output report created by LOGOS contained a corresponding higher number of entries than ATAO's term candidate report. To compare the accuracy of each system, we felt the need for a comparison that took into account the length of the raw machine output. This distinction is necessary because the longer the term candidate report, the more valid terms it will contain; however, it will be more time-consuming to filter once processing by the automated term-extraction system has been completed. As shown in the two pie charts in Figure 5-9, both systems demonstrated approximately the same level of accuracy, with ATAO slightly ahead of LOGOS. Only 37% of the entries on the term candidate report generated by ATAO were erroneous, as compared to 43% erroneous entries for the report by LOGOS for the same electronic texts. These results indicate to us that although LOGOS appeared to demonstrate an appreciably better performance, in reality it only produced a longer term report than did ATAO.

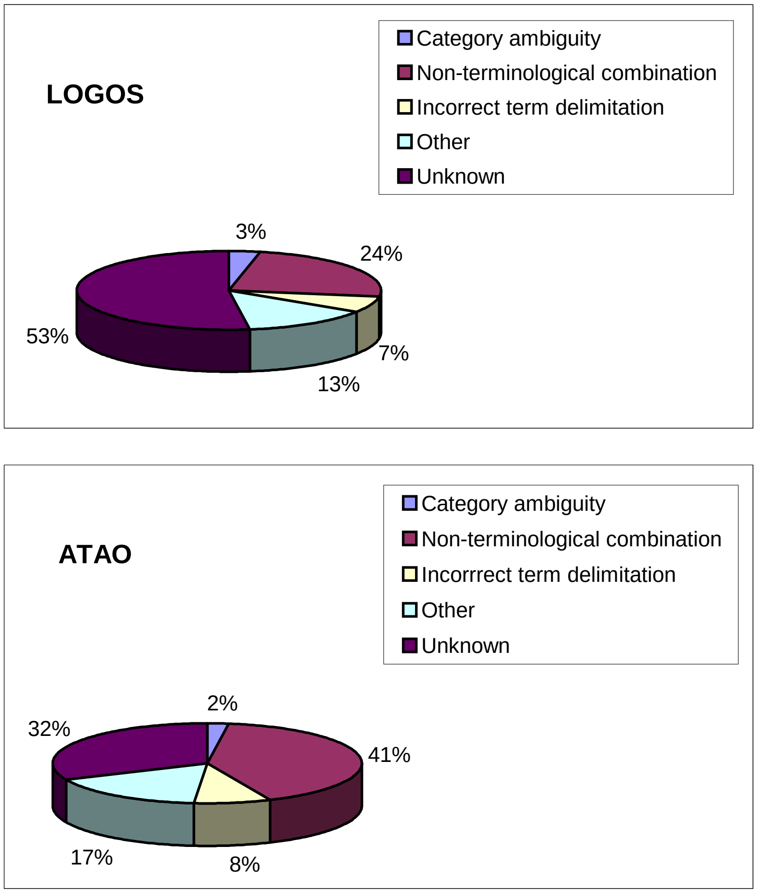


Figure 5-10 Incidence of noise-producing factors detected for each system

We then analyzed the results we obtained for each system to learn the source of the noise entries on their respective term candidate report. Although we were aware of the fact that ATAO and

LOGOS each had approximately the same percentage of noise in their machine output, it was not assumed that the reasons for the erroneous terms would be the same. We were consequently not surprised when we calculated the results shown in Figure 5-10. The results are similar for the two systems because the order in which the noise-producing factors were detected in both term candidate reports was identical, albeit with slightly varying proportions. For example, although we report that both systems show non-terminological combinations as the prime source of noise on their term candidate report, LOGOS had fewer such entries, with the reason for extracting many more term candidates coded as *Unknown*.

A summary of our observations for each of the principle types of noise is given in the following sections.

5.4.2.1. Category ambiguity

“Category ambiguity” is an example of syntactic noise resulting from an incorrect parse. Simply put, the entries in the automated term-extraction system’s dictionary can sometimes be coded as belonging to more than one part of speech. If the system cannot determine through syntactic analysis which is the appropriate part of speech, this form of noise may occur.

As illustrated in Figure 5-8, the causes of terminological and syntactic noise are all closely interrelated. Our results for the incidence of each of the noise-producing factors allude that category ambiguity was less prevalent in our corpus than the other causes (incorrect term delimitation and non-terminological combinations); however, we feel that this interpretation is misleading and requires an explanation. Because so many lexemes in the English language are ambiguous, we realize now that we tended to code noise as being caused by one of the other noise-producing factors unless the extracted non-term was a flagrant example of category ambiguity and could not be traced to any other factor. Looking back, we do not feel that this was unjustified, but it does mean that category ambiguity is underrepresented in our statistics. TEMS was designed to only support one cause for noise per term candidate. If multiple selections had been possible, the majority of our entries would have been assigned more than one noise-producing factor, which ultimately defeats the purpose of carrying out such a detailed analysis.

Some noteworthy examples of category ambiguity were detected while analyzing our corpus. We identified several different types of category ambiguity in the term candidate reports produced by LOGOS and ATAO.

Ambiguity type	Example	Context
N ↔ V	DLLs remains	Extensibility via DLLs remains relatively tricky but represents a very powerful customisation feature if you have access to the appropriate programming resources.
ADJ ↔ N	current price	Current prices today range from $2.00 to $.20/MB of storage, and will probably drop 50% in the next year or two, especially at the low end.
ADJ ↔ ADV ↔ V	even customers	Down-time can lead to a loss of revenue, productivity and even customers.
ADJ ↔ N ↔ V	key question	Is it a better format for distributing online information is the key question.

Table 5-1 Examples of category ambiguity

Lauriston (1994:150) revealed that 40% of the lexemes in running text are ambiguous in English. In this case, compounds such as *server support* would be counted as two ambiguous units. The cited figure of 40% does not apply fully to our corpus of terms because we evaluated our terms in their full compounded form instead of assessing category ambiguity on a per-lexeme basis. We concede, however, that category ambiguity is a significant issue facing automated term-extraction systems and human terminologists alike.

5.4.2.2. Non-terminological combination

A “non-terminological combination” is a type of structural ambiguity that is caused by the system being unable to discriminate between true noun phrases and free collocations that bear the same syntactic pattern. For the purposes of this study, we understand non-terminological combinations to include plausible-sounding LGP combinations (ex: *client workstation connection*, *virtually invincible defense* and *single desktop device*), fixed LGP expressions and collocations (ex: *on the fly*, *wide array of* and *all rights reserved*), and names of companies, products and people (ex: *ESB International*, *Microsoft Windows NT* and *Leo Langevin*). Non-terminological combinations do not include random word sequences or sentence fragments (ex: *day a week*, *only a matter of* and *past ten years*), e-mail addresses, the names of FTP sites or URLs (ex: infor@sresearch.com, ftp://ftp.mcs.com/mcsnet.users/VSE/text/RAID.FAQ, and www.raid-avisory.com). Although these items do have meaning and a standardized nomenclature, they are not lexical units in the strictest sense and thus were coded otherwise in TEMS.

We looked for indications of coherence and cohesiveness with the term candidates to guide us when assessing their terminological status. In other words, we tried to look for term candidates that a human terminologist with absolutely no subject field knowledge (i.e. extra-linguistic information) might unknowingly include when manually extracting terms from the same electronic text. Other key points we used when evaluating the term candidates are summarized in Section 1.3.1.1 of this work.

Analyzing the machine output generated by LOGOS and ATAO led us to report that non-terminological combinations were the most prevalent cause of noise for both systems. We calculated that 24% of the noise produced by LOGOS and 41% of the noise produced by ATAO stemmed directly or indirectly from non-terms being identified by the system as term candidates on its term-extraction report.

Some instances of non-terminological combinations were evident. For borderline cases, we studied the immediate context to ensure that they were not incorrectly-delimited terms, and avoided misreporting the extraction error. Several observations were made while analyzing the non-terminological combinations identified by the two automated term-extraction systems. *Effective*, *best*, *high* and *huge* are examples of adjectives that were never part of valid terms, yet were extracted on multiple occasions by both LOGOS and ATAO. They caused the systems to extract free collocations on a regular basis. We also noted that some of the chart headings in our electronic texts were extracted by ATAO and LOGOS as noun phrases because they could not judge that the headings were not part of the body of the document being scanned for terms.

5.4.2.3. Incorrect term delimitation

“Incorrect term delimitation” is another example of syntactic noise resulting from an incorrect parse. This type of noise occurs when the automated term-extraction system reports a long phrase that contains both a valid term, as well as part of the valid term’s immediate context as a term candidate. Conversely, incorrect term delimitation also occurs when the term-extraction system truncates elements from the beginning or end of the noun phrase it is attempting to identify.

An example of incorrect term delimitation taken from the term candidate report generated by LOGOS is *secure real-time embedded system* extracted from the sentence: *Unlike typical CPU-intensive proxy servers that perform extensive processing on each data packet, the Cisco PIX Firewall uses a non-UNIX* ***secure real-time embedded system****.* LOGOS should have included the element *non-UNIX* along with the term candidate it identified in order to extract the entire term: *non-UNIX secure real-time embedded system*.

For the machine output generated by LOGOS and ATAO, incorrect term delimitation was the second-leading cause of noise after non-terminological combinations. We calculated that 7% of the noise produced by LOGOS and 8% of the noise produced by ATAO could be traced to terms being delimited incorrectly by the system.

For the purposes of this study, incorrect term delimitation did not include instances where LOGOS truncated the term candidate in unusual places such as *online docum* or *gital camera* because we are of the opinion that such errors can be traced to either the lemmatizer or the importation of the LOGOS data to Excel, then to TEMS’ system database, and not to the term-extraction system’s parsing engine per se. In cases of abnormal truncation, we handled the affected term candidates as though they were extracted correctly.

Some cases of noise caused by incorrect term delimitation were easily categorized; whereas others required more extensive analysis before a final decision could be made. For example, in situations where the system incorrectly delimited the term candidate it was extracting, and it ended up being a non-terminological combination, we alternated between coding these term candidates as non-terminological combinations and instances of incorrect term delimitation. Each case was judged on an individual basis; however, the majority of them could easily have been coded as both. The last-resort option when coding the causes of noise, *Unknown*, was strictly reserved for term candidates that exhibited no traits of any of the three main noise-producing factors, not candidates that could be placed in more than one category.

Several observations were made while analyzing the noise that resulted from incorrect term delimitation. We routinely encountered valid terms that were extracted with the verb which immediately followed it. These cases involved incorrect term delimitation including a categorically-ambiguous verb. Such instances were divided between incorrect term delimitation and category ambiguity. More information on category ambiguity is given in Section 5.4.2.1.

As was the case with non-terminological combinations, some instances of incorrect term delimitation were obvious. Others, however, required a second round of analysis to evaluate which of the three main causes of noise was the most appropriate. In cases where no decision was clear, an arbitrary choice among the possible causes was made.

5.4.3. Silence

The opposite of noise (see Section 5.4.2), silence is produced when the automated term-extraction system fails to identify valid terminological units for its term candidate report. Refer to Section 1.3.5 of this work for a broader discussion of silence, including a typology of silence-producing factors for linguistic-based term-extraction systems.

Detecting and processing instances of silence must naturally take place after the valid terms and noise have been dealt with because we must know what is present on the term candidate report before we can ascertain which units are missing from it (i.e. silence). As such, we proceeded with our methodology described in Chapter 4 without incident until we arrived at the stage where the silence generated by LOGOS and ATAO was to be coded in TEMS' system database. Here, we drew a complete blank. We detected hundreds of items absent from the term candidate report of one or both systems, yet we did not have even the faintest idea how to explain the silence in the vast majority of cases. This turn of events led us to research the two term-extraction systems further, but to no avail. Unlike noise, where we permitted ourselves to make an "educated guess" when indicating the reasons behind the systems' extractions errors, when processing silence, we were unable to even speculate except in a few isolated cases.

On a more positive note, we were able to make a few quantitative observations on the systems' performance with respect to silence.

- LOGOS seemed to miss terms of all formation types and lengths equally. It was more difficult to identify tendencies by this system because LOGOS overlooked a uniform percentage of all term formation patterns clear across the board. ATAO, on the other hand, demonstrated more specific traits regarding the nature of the terms it frequently missed.

- ATAO experienced difficulty extracting term candidates that included colour elements, such as *grey scale* and *white paper*.

- A possible example of category ambiguity, the term candidates starting with the elements *read* or *write*, such as *read request* and *write throughput rate*, posed a particular challenge for ATAO. We believe that this might be because the elements *read* and *write* were mistaken for verbs when parsing the sentence.

- On the syntactic level, ATAO had considerable difficulty with terms containing the elements *physical* (ex: *physical fetch* and *physical address*), as well as terms containing the element *multi*- (ex: *multithreaded 32-bit operating system* and *multi-processor computer*) [20].

- A graphical issue, ATAO and LOGOS seemed somewhat to have trouble extracting term candidates containing slash characters, such as *I/O device* and *TCP/IP protocol*.

A breakdown of the percentage silence for each term candidate length recorded in our corpus is given in Figure 5-11.

In Figure 5-11, we remark a direct relationship whereby both systems had increasing difficulty identifying term candidates as the length of the terms grew longer, with the exception of ATAO and two-unit terms and five-unit terms for both systems. Furthermore, we notice that both systems began struggling when terms increased to six units in length; whereas they remained strong until this point. LOGOS demonstrated its greatest proficiency (i.e. the lowest incidence of silence) when extracting two-unit terms, and ATAO was more or less consistent until its threshold of six-unit terminological units was reached.

[20] Terminological units including the element *multi*- tended to exhibit inconsistencies of their own. Sometimes they were hyphenated, sometimes not. Both forms were problematic for ATAO.

Some of the terms from the human list that included this lexeme are:

- multi-user
- multi-vendor
- multitask

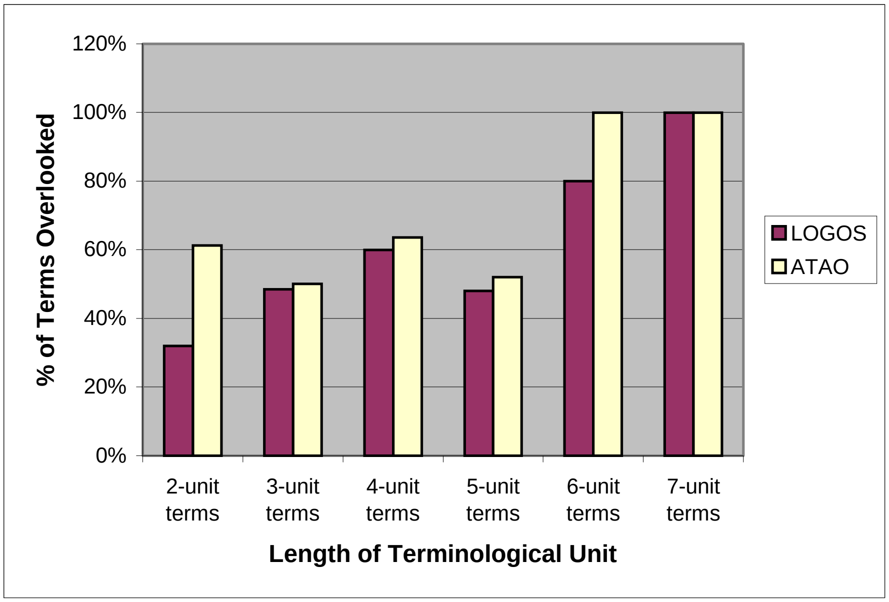


Figure 5-11 Percentage silence for each term candidate length recorded in our corpus

Discussions focusing on the three main silence-producing factors appear below.

5.4.3.1. Pattern not implemented in the system

The algorithms or list of term formation patterns used by LOGOS and ATAO are unknown to us. As such, we do not know with certainty if any items from the list of manually-extracted terms were absent from either system's term candidate report simply because its lexical formation was unknown. On a few rare occasions, we did encode the reason for the system failing to extract a given term as the pattern not being implemented in the system; however, this is only a guess and we based this assertion on instinct and nothing more.

Both systems had difficulty extracting terms with the syntactic formation patterns N + CONJ + N and [N + CONJ + N] + N (ex: *look and feel* and *end of file mark*). We opine that this is the result of an incorrect parse and not because the pattern is absent from the system, but we cannot be sure. Comparable difficulties were manifested when presented with terms formed in the following manner: [N + PREP + N] + [N + N], such as *peer to peer file service*. Neither system extracted a single term candidate that corresponded to this syntactic pattern, bringing us to the assumption that this pattern must be missing from both systems.

Naturally, some of the more lengthy entries on the human list were formed according to infrequent syntactic patterns that we believe absent from both systems (ex: *fault tolerant enterprise class disk array system* formed using the syntactic pattern [N + ADJ] + [[N + N] + [[N + N] + N]]). The absence of this sort of long, uncommon pattern is understandable because it would introduce many times more noise on the term candidate report than it would valid terms.

We are disappointed by the fact that we are unable to comment further on this aspect of the performance of either automated term-extraction system.

5.4.3.2. New word form

Similar to Section 5.4.3.1, we are unaware of the exact term formation patterns used by LOGOS and ATAO. Consequently, we can only estimate if any items from the list of manually-extracted terms did not appear on the term candidate report of either system because the word form was new. We unfortunately cannot comment further on this aspect of the performance of either automated term-extraction system.

5.4.3.3. Absence of a given word tagging

Similar to Sections 5.4.3.1 and 5.4.3.2, we could not judge whether a lexeme had been fully documented in the automated term-extraction systems; therefore, any indications of such in the TEMS database were purely speculation. However, we opine that this silence-producing factor would be less prevalent than the other two factors discussed above because lexical units are rarely assigned new parts of speech and both systems used for this study are recent.

5.5. Overall system ability

After comparing LOGOS and ATAO separately on several different specific points, we only felt it appropriate to distance ourselves from the systems for a moment to analyze them in more general terms.

In an attempt to yardstick the performance of each system and look for trends, we calculated the percentage of genuine terms that each automated term-extraction system was able to identify per electronic text. For most of the texts used for this study, the variation between LOGOS and ATAO was relatively consistent. We were extremely surprised to learn, however, that ATAO actually extracted a few more valid terms in the Exabyte tape drive performance text than did LOGOS. We also noticed that ATAO's performance when processing the RAID document was closer to that of LOGOS, and that the results from the Strategic data warehouse performance tuning white paper were poorer than average. These findings lead us to believe that ATAO fared better when processing texts with a narrow focus (i.e. higher frequency), than broader texts that covered a wide range of ideas and vocabulary.

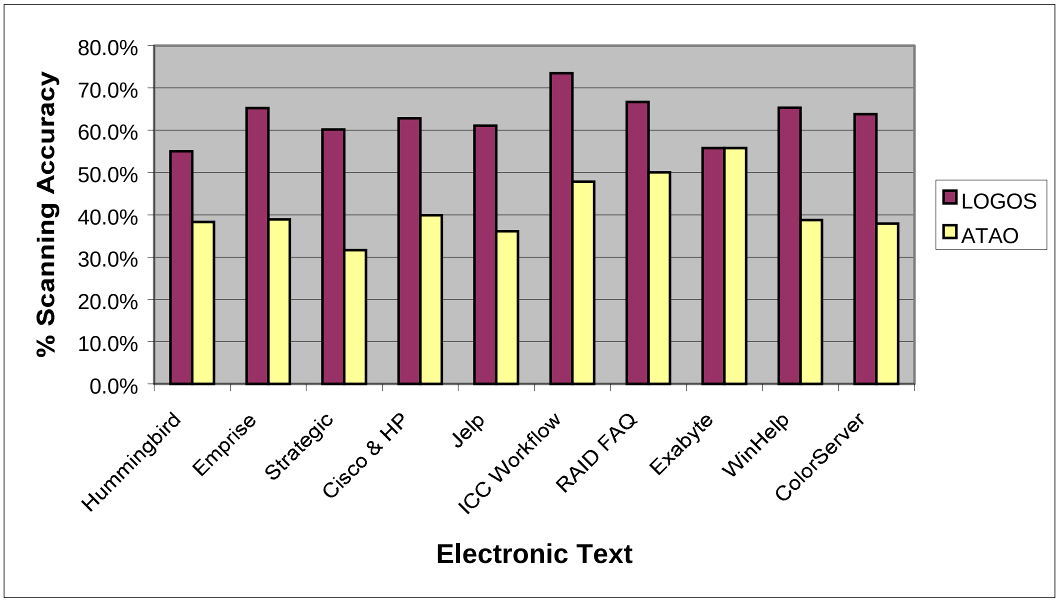


Figure 5-12 Quantitative performance values for each automated term-extraction system per electronic text

We then took a closer look at the performance of each automated term-extraction system when extracting different syntactic formation patterns present in our collection of electronic texts. These figures are limited to valid terms and do not include noise. Our findings are summarized in Figure 5-13[21].

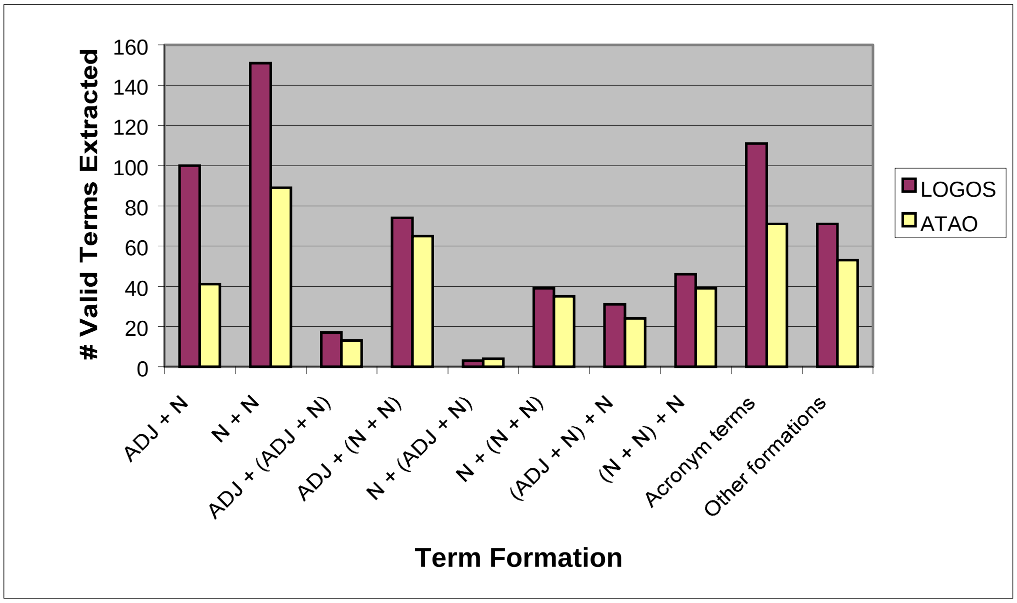


Figure 5-13 Quantitative performance values for each automated term-extraction system per grammatical formation

Figure 5-13 illustrates that LOGOS outperformed ATAO when extracting terminological units with the ADJ + N and N + N syntactic patterns, and the quantitative performance results between the two systems remained relatively consistent for the other term formation patterns. How acronyms were handled is discussed in Section 5.4.1.2.

All automated term-extraction systems, whether they be linguistic-based such as LOGOS and ATAO, statistics-based or a hybrid of the two approaches, share a common goal — reducing the human terminologist's workload by completing

[21] Note that we were obliged to reduce by 50% the raw number of valid terms extracted by both systems for the ADJ + N and N + N term formation patterns because they were so high compared to the other types that they reduced them to the point of being illegible.

certain scanning tasks automatically. This being the case (a software application attempting to emulate the work of a trained human professional) one of the manners to measure the success of these systems would be to compare the raw results obtained with those of a human terminologist to see if the two have the same look-and-feel. Our results comparing the contents of the human list with the raw output lists produced by LOGOS and ATAO are shown in Figure 5-14.

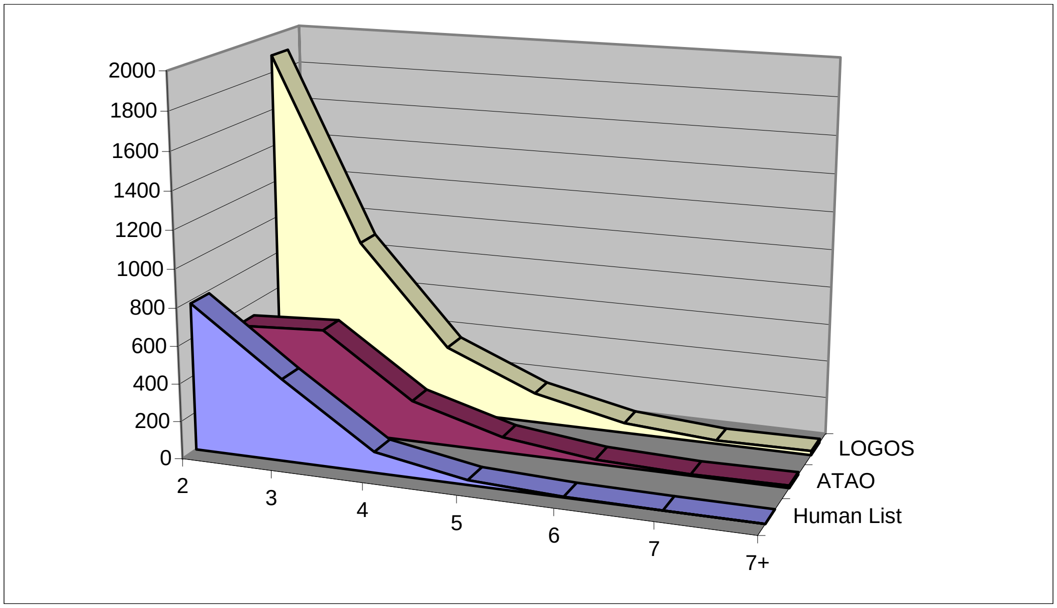


Figure 5-14 Curve representing the distribution of term candidates sorted by length

Analyzing our term candidate reports for closeness in composition to the human list led us to the discovery that the output produced by LOGOS was more analogous to the human list (i.e. its curve in Figure 5-14 has an ergonomic human-like "feel" to it) and consequently more effective in general. In comparison, the list produced by ATAO has a flattened top indicating that it extracted far too few two-unit terms or far too many three-unit terms. This novel method of assessing the term candidate report underlines the importance of the human-machine interface. Even if computers can provide humans with impressive quantities of data, we are not any farther ahead if this data is difficult for us to assimilate, apply or manipulate.

On another more human note, we found the term candidate report created by ATAO to be more user-friendly and easier to work with. However, it must be noted that the term candidate report produced by LOGOS was not intended for viewing outside the specially-designed software program, TermBuilder. Consequently, our observation is pertinent and understandable. ATAO is also a more straightforward software application in general; however, it attempts to fulfill fewer purposes and only treats French and English; so it is understandable that it is simpler to understand and use.

5.6. Evaluation of TEMS' performance and suitability

Once we had completed entering the human list and the term candidate reports generated by ATAO and LOGOS in TEMS' system database, we reflected on our work and evaluated the TEMS application for its time-saving potential and suitability to this research project. Was developing TEMS worth the time investment? Was it easy to use and did it meet our needs? Would another type of application have been a more appropriate choice? The answers to these questions are developed below.

In the early stages of our research, we did not feel that TEMS would save us any time at all; in fact, it took longer to develop than we had ever imagined it would. Each feature we added increased development time accordingly. We began to assume that a computerized solution would not be a time-saver, but would nonetheless ensure that our statistics were accurate (as compared to managing the data on paper only). It was a consolation to know that our data would be of the highest quality, even if it were long and onerous to obtain.

Then, for a brief period of time, we were concerned that we had not accurately analyzed how to represent the incidences of noise and silence in the system database, meaning that our results for recording noise and silence would have been skewed. A joint investigation with an experienced Visual Basic developer led us to insert an additional column in the database's main table before we began entering the noise and silence data, thereby correcting this situation and reassuring us that the data was truly representative of what we needed.

When the time came to analyze our data, however, TEMS' value to us became quite clear. Because we are familiar with query building using SQL, we were able to refine our data (eliminating doubles, standardizing entries, correcting tiny errors with respect to capitalization, etc.) in very little time, as compared to using a card file system to represent our term candidates. Furthermore, we were able to generate far more statistics in a relatively short period of time through TEMS than if we had used an alternative approach. We were able to easily save our data then import it into another application for pie charts and bar graphs. Compiling our statistics did not take half as long as we expected. It must be said that our expectations may have been affected by the exaggerated period of time that was required to finish developing the TEMS application itself.

In sum, we believe that TEMS was indeed worth the investment in time and effort overall. By creating this application, our knowledge of both term record management and software development was heightened considerably. The TEMS program, once corrected, proved to be 100% pertinent to this research project. Moreover, it was straightforward and easy to use when entering our data in the system database. In our opinion, no commercial application would have been a more appropriate choice for our needs. If we were to complete a similar research project in the future, we would definitely use TEMS or a program similar to TEMS if at all possible.

5.6.1. Other applications of the TEMS system

After producing such a high-level, elaborate software program for a single, albeit important, purpose (i.e. this research project), it only stands to reason that we make an attempt to find ways to apply this development to other activities, whether professional or academic, in order to obtain the highest possible return on our investment of time and effort. We took it upon ourselves to investigate how to reuse TEMS once this study was over. We felt that its flexible and scaleable architecture would lend itself well to multiple professional or academic purposes. After analyzing portability issues, evaluating the time required to update TEMS, the possible benefits of making the changes and what features the new version of TEMS could offer, we concluded that TEMS could be converted into an application for managing terminological records for either a university faculty, a free-lance translator or the translation department of a company. We then completed the face-lift and modified TEMS' underlying database.

The following changes were made to TEMS to create the professional version:

- The TEMS interface was enlarged to a minimum resolution of 800 x 600 pixels instead of the original size of 640 x 480 pixels. This change was made to allow more controls per tab and took advantage of the larger monitors available in the workplace.

- The way TEMS handled dates was changed so that the application was Y2K compliant.

- The file-sharing parameters in the system database were modified to allow multi-user access over a network.

- A short help file was written to assist new users with the search screen. This help file also listed the program's keyboard shortcuts and gave examples of what items should appear in the Notes tab.

- The system database was updated (tables renamed, queries removed, etc.) to reflect the modifications made to the system interface.

- The error messages were re-written in fuller detail to reflect the new multi-user environment.

- A field was added to the Source tab to contain a French translation of the English term (where the term candidate was originally located in TEMS). This added feature means that TEMS is now able to house bilingual term records in the same manner as the linguistic database, Termium. The search screen was updated to reflect this addition. It is now possible to search the system database for either French or English terms.

- Additional fields were incorporated in the Source tab to hold the term's definition, a context and/or usage information, as well as the gender of the French equivalent.

- The Composition, Noise and Silence tabs were removed because they are not applicable to term record management in a corporate environment.

- In addition to indicating the date when processing for the term was completed, a field was added to the Notes tab to automatically display the login ID of the individual who made the changes to the record. This is helpful when tracing individual records in a multi-user database.

- A field was added to the Notes tab to indicate whether or not the term had been approved by Management, and by whom. This can be likened to Termium's "research term files" (work still in progress) and "master term files" (approved and finalized).

The TEMS application has now been reincarnated and currently exists under a different name as a term record database organizing the corporate and technical terminology used by the technical publications/linguistic services department of a Montreal-based company. At the time of writing, this new version of TEMS included a database that contained over 800 terminological units. Being able to apply our application to a professional environment with relative ease only confirms once again that our application was a success before and after this research project.

The next chapter serves as a conclusion to this research project.

6. CONCLUSION

The purpose of this study was to carry out a critical comparison of the performance levels achieved by two commercially-available term-extraction software programs: ATAO by the TRADUCTIX company and LOGOS 3.0 by the LOGOS Corporation. This focus of our work is restricted to compound nominal terminological units occurring in the Language for Special Purposes (LSP) for computer science.

To compare the programs, a collection of specialized computer science texts were submitted to each in order to create a corpus of extracted term candidates. We then analyzed the machine output to determine recurring themes common to both systems, as well as each term-extraction software's strong and weak points.

In a nutshell, our experiment led us to find that ATAO had a better overall performance when extracting term candidates from our collection of electronic texts, although LOGOS also had its own strong points worth mentioning. More specifically, we observed that ATAO demonstrated greater ability when extracting longer terms (those with four or five lexical units); whereas LOGOS was more proficient when processing term candidates with fewer lexical units, as well as when analyzing for terms with the terminological formation types N + N and ADJ + N. We reported that the two types of noise most commonly found in our corpus were non-terminological combinations (terminological noise), followed by incorrect term delimitation (syntactic noise) resulting from non-terminological elements being extracted as part of valid terms. Furthermore, we came to the conclusion that our lack of information about the inner workings of the two automated term-extraction utilities kept us from making any specific judgments about the reasons behind the incidences of silence in our term candidate reports.

If we could make one suggestion on how to improve the performance of ATAO, it would be to either disable the lemmatizer that processed the items on the term candidate report or make considerable improvements to it. It often seemed to be more of a hindrance than an assistant when processing the system's term candidate report.

LOGOS, on the other hand, would benefit greatly from incorporating measures in the system architecture to reduce the incidence of noise and overly-long term candidates. The candidate term report produced by LOGOS contained over twice as many entries as that produced by ATAO and was two and a half times as long as the human list.

6.1. Applications for this research

Considerable reflection went into deciding how this research project might best be applied. We see this question to be on two levels, and will address each level accordingly. In the immediate context relative to this research, we will discuss how the quantitative performance results we obtained for LOGOS and ATAO might be applied. In addition, we will extend our vision of applications of this research to beyond the scope of this study and give our opinions on how automated term-extraction technology might be applied to every-day life or to existing technologies that currently do not have any co-relation with automated language processing.

6.1.1. Direct application of our results

A comparison of this nature brings to the forefront the issues related to the automatic or assisted identification of compound terms. The extraction data we obtained by submitting our collection of specialized texts to LOGOS and ATAO allowed us to measure the strengths of the two term-extraction software programs, as well as examine areas of each that need refinement. The companies that developed these systems could readily use the results of our analysis to fine-tune their respective product. Conversely, a comparison of two such software programs would assist potential users, such as a business or academic institution, considering the acquisition of such a tool.

When investigating a number of automated term-extraction systems in Chapter 2 of this work, we learned that term extraction is but an element of the much larger field of terminology automation, and that most terminology extraction programs are encapsulated in larger, multi-purpose applications. Research that sheds light on what automated term-extraction systems are and shows their real-world capabilities has the potential to encourage other software manufacturers to incorporate a similar feature in their application.

6.1.2. Application of automated term-extraction technology

Automated term-extraction is already being applied in one form or another to related disciplines such as machine translation. The LOGOS machine translation system, for example, includes an automated term-extraction system to accelerate maintaining and enriching its system dictionary used for machine translation. The inclusion of semantic filters or tagging to the criteria presently used by LOGOS' Terminology Verification utility when identifying term candidates would most certainly yield more exact results, i.e. shorter lists of term candidates with a reduced incidence of silence. Shorter, better-quality lists would reduce the level of human intervention required and make the utility more efficient on the whole.

Lauriston (1994:167) briefly mentions the application of automated recognition of multi-word descriptors for automating the creation of indexes for electronic texts, another promising application of this technology. Such software (ex: Authex Plus, HyperIndex and wINDEX) is now commercially available according to the American Society of Indexers; however, we are unaware if these applications employ semantic knowledge at any level or if their parsing engines are limited to morpho- syntactic analysis and/or frequency to determine how input texts should be indexed[22].

The possibilities of applying compound term recognition along with semantic information[23] to projects in the area of information science are virtually boundless. Few software programs would not benefit from the advantages of "comprehension". A system that applies some form of semantic information when processing input texts could, for example, be used as a starting point for the development of a new generation of "intelligent" search engines for the Internet (including both the World Wide Web and UseNet newsgroups). In an intelligent search engine environment, the user would be able to enter more detailed, plain-language search queries explaining exactly what type of information was sought. The search engine would be designed to gain a limited understanding of the HTML documents it scanned from the Internet and be able to return concise search results listing only the most pertinent sites, as opposed to the many thousands of results returned when performing multi-keyword searches on current search engines. This technology would either entirely replace the meta tags presently used by Internet search engines to ascertain the contents of HTML

[22] Automated indexing software programs build a concordance from the electronic documents submitted to them, similar to automated term extraction. Our research on this issue has led us to believe that automated indexing was never intended to produce back-of-the-book indexes. We learned that although the manufacturers often claim these packages build complete indexes, the actual results are a list of words and phrases that are sometimes helpful in the preliminary stages of building the index, but nonetheless require filtering just as the term candidate reports created by LOGOS and ATAO.

Furthermore, usability tests for indexing software have shown that the word lists produced omit many key ideas and phrases, and can neither fine-tune terminology for easy retrieval, nor build the needed hierarchies of ideas that human indexers can. Indexing involves understanding and organizing the ideas and information in the text, which is a key point of the research projects currently underway in this field. In conclusion, we believe that these tools do not employ semantic knowledge at the present time; however, recourse to semantic information will become an integral part of these tools in the future.

[23] An enlightening discussion of one application of semantic information to corpora for the purposes of more accurate processing is given in Meyer *et al.* (1996:01).

documents or work jointly with them. Meta tags are a flawed concept to begin with, and any improvement that can be made in this regard would be highly beneficial[24].

An intelligent newsreader would be able to filter the postings to the newsgroups selected by the user and return only those which fulfilled certain criteria, such as subject matter and specific tools or technology, and filter out spam postings. This feature would be extremely practical for high-traffic or unmoderated newsgroups that can generate up to several hundred new postings every day. Another possibility would be to design more effective engines for searching newsgroups aided by semantic information. Traditional search engines are inadequate because the topic discussed by the newsgroup is often so specific that the majority of postings would contain the same keywords, making searching difficult at best.

In addition, incorporating basic semantic information on the World Wide Web could be applied to the fledgling concept of push technology. "Pushing" information instead of having it "pulled" as the result of one-to-one requests for documents to the Web server is a method of information delivery on the World Wide Web that is initiated by the information server rather than by the recipient. Simply put, users enter preferences for push content to the program, who searches for documents or information that corresponds to the users' profile and delivers it on their behalf. By adding a semantic element to the push software, preference profiles could be more precise and the pertinence of the pushed

[24] Desperate to increase the number of hits to their Website, less scrupulous Web authors sometimes fill the meta tag fields in the header section of their HTML documents with an exaggerated number of key words (for instance, a Web site for a junior hockey team might include the names of all the teams and key players of the National Hockey League to attract more visitors).

Other questionable tactics include the use of suggestive and sexually-explicit descriptors on Web pages, or even adding the names of celebrities to cause their site to jump to the top of the search results list. In a move to combat this issue, many Internet search engines have implemented measures such as limiting the maximum number of keywords; however, the situation remains far from ideal.

material would be higher because the push program would have a better understanding of the contents of the text it was interpreting for the user.

These suggestions are but a few of the ways semantic tagging and filters coupled with automated term extraction could tame the sheer size of the Internet and help to bridge the gap between humans and machines. We expect some, if not all, of these applications to come to fruition sometime in the future.

6.2. Possible refinements of our methodology

Reflecting on our methodology once our experiment was completed brought to light a few points that might have been handled differently in order to obtain better-quality results. We opine that, by and large, the areas described herein were only obvious to us once we had already begun work, meaning inevitable. In other words, we do not feel that increased analysis of our needs or enhanced subject matter knowledge would have circumvented these issues.

Occasionally, Selkirk's compound structure typology did not seem to cover all types of compounds: *broadcast* might be considered a verb (or is it a back-formation instead?) Another example that was not easily defined by Selkirk's compound structure typology was *cut-through proxy*. This term is an ADJ + N, but the adjective-like modifier is actually composed of a V + PART. (or is it simply colloquial language and not properly formed?) Perhaps improving on Selkirk's compound structure typology by introducing the element of a particle would be a fitting refinement of our compound term typology, and consequently of our methodology in general.

We did not take frequency into account when judging the lexicalization of the terms on the human list or the term candidates on the reports generated by LOGOS and ATAO. Each candidate was evaluated solely on the term-extraction criteria listed in Section 1.3.1.1 of this research project. Perhaps including frequency as a criterion when judging the terminological status of compounds would be a possible improvement to the research methodology we employed to evaluate the ability of LOGOS and ATAO when identifying compound terminological units in running text. However, in order to use frequency as a criterion, it would be necessary to opt for much longer texts that were closer related conceptually than those used in our research project because most valid terms only appeared once or twice in the shorter electronic texts we treated, not to mention the redundancy factors such as hyphens, spelling variations and the like listed by Bédard (1992:754).

An avenue worth investigating that would further the work completed in this study would be to identify a single text, also from the field of computer science, with a word count equal to the total word count of our collection of electronic texts and submit it to both automated term-extraction utilities and compare the findings for frequency on the two term candidate reports generated. By varying this single aspect of our methodology, the importance represented by frequency of the identified term candidates would be brought to the forefront while leaving the other aspects of our evaluation unchanged and facilitating interpretation of the results.

6.3. Improving automated term extraction

Closely examining two automated term-extraction systems, ATAO and LOGOS, led us to truly understand the formidable challenge developers and linguists face when they set out to design a software program that attempts to gain a limited understanding of the electronic texts submitted to it, then list the compound terminological units they contain. The systems we investigated are both highly complex and the product of considerable resourcefulness and hard work on the part of those who developed them.

One of the key points of this study was our realization of the extent to which the semantic aspect of analyzing technical texts for term candidates was crucial to achieving performance rates that are within acceptable limits. Current term-extraction systems are trying to surpass a nebulous threshold imposed by the inherent limitations of implementing syntactic analysis alone in order to process input texts.

Lauriston (1994:153) argues that syntax alone is not sufficient for determining absolutely whether a phrase is terminological or free. We totally agree with this point after scanning our corpus and seeing the list of valid-sounding free collocations that were extracted by the term-extraction programs. Extra-linguistic knowledge is an absolute must for proper term identification in running text.

A similar point raised by Lauriston (1994:166) was the possibility of making automated term-extraction systems “adaptable to subject field or text type,” whereby term-extraction systems would have a specialty field just as their human counterparts frequently do. Customizing term-extraction systems for a certain genre of text instead of designing generic systems without a specialty could only improve performance, although feasibility of such a venture, financially speaking, is questionable.

We would quite understandably support any move to decrease the incidence of noise in the term candidate reports produced by the term-extraction software systems. However, completing this study allowed us to fully grasp just how problematic it is to limit noise without a corresponding increase in silence resulting from the valid terms being excluded from the term candidate report because of overly-stringent term candidate identification rules. We agree with the designers of the term-extraction programs we studied with regards to increasing noise so as to keep silence to a strict minimum. We feel it is far less work for a human terminologist to filter a longer term candidate report than to go through the input text a second time looking for any terms that the term-extraction system may have missed.

A considerable amount of noise for both systems was caused by ordinal and cardinal numbers and digits being mishandled, such as *16 bit Windows application*. This is one area of automated term extraction that could benefit greatly from further improvement. We must concede, however, that if the authors of the electronic texts we submitted to the term-extraction systems had used hyphens to indicate the adjectival use of the digits, this problem would have been vastly improved, if not solved almost completely.

6.4. Summary

This research project forced us to take a step back and analyze what exactly constitutes a terminological unit, as opposed to a non-term. We learned that many aspects of term identification rely on extra-linguistic information and are subconsciously applied by human terminologists. An improved understanding of what constitutes a term would almost certainly bring about an overall increase in the performance achieved by term-extraction systems because their designers would be placing themselves in the situation of the computer system attempting to identify the term candidates in the input texts. Such “role-playing” would lead to heightened understanding of the computer systems’ specific needs and limitations when processing texts.

In conclusion, we feel that our examination of these two automated term-extraction systems based on syntactic analysis was beneficial and worthwhile. We eagerly look forward to other similar studies when semantic analysis has been fully or partially incorporated in the identification strategy by term-extraction systems available in the future.

BIBLIOGRAPHY

[1] Ananiadou, Sofia (1988). *Towards a Methodology for Automatic Term Recognition*. Ph.D. dissertation. University of Manchester.

[2] Atkins, Beryl T. *et al.* (1993). **Le Robert & Collins Dictionnaire Français▶Anglais, Anglais▶Français Senior**. Paris: Dictionnaires Le Robert.

[3] Auger, Pierre (1979). *La syntagmatique terminologique, typologie des syntagmes et limites des modèles en structure complexe*, **Actes de la table ronde sur les problèmes du découpage du terme**, Montreal, Quebec, 26 August 1978. Quebec City: Office de la langue française, pp. 11-25.

[4] Auger, Pierre, Patrick Drouin and Marie-Claude L'Homme (1991). *Automatisation des procédures de travail en terminographie*, **Meta** 36(1), pp. 120-127.

[5] Bédard, Claude (1992). *La prétraduction automatique, outil de productivité et d'évolution professionnelle,* **Meta** 37(4), pp. 730-760.

[6] Bennett, Paul (1993). *A multilingual translation-oriented typology of compound terms*, **Traitement automatique des langues (TAL)** 34(2), pp. 43-58.

[7] Bennett, Paul (1994). *The Translation Unit in Human and Machine*, **Babel** 40(1), pp. 12-20.

[8] Benveniste, Émile (1966). **Problèmes de linguistique générale**. Paris: Gallimard.

[9] Blessé, Bruno de, Blaise Nkwenti-Azeh and Juan C. Sager (1998). *Glossary of Terms Used in Terminology*, **Terminology** Volume 4(1), pp. 117-156.

[10] Bourigault, Didier (1993). *Analyse syntactique locale pour le repérage de termes complexes dans un texte*, **Traitement automatique des langues (TAL)** 34(2), pp. 105-117.

[11] Bourigault, Didier and Isabelle Gonzalez (1994). *Acquisition automatique de termes complexes en français et en anglais, approche comparative*, **Proceedings of the Workshop on Compound Nouns. Multilingual Aspects of Nominal Composition**, University of Geneva. Geneva, 2-3 December, 1994, pp. 29-43.

[12] Cabré, Maria Teresa, adapted and updated by Monique C. Cormier and John Humbley (1998). **La terminologie : théorie, méthode et applications**. Ottawa: Ottawa University Presses.

[13] Canada. Department of the Secretary of State (1996). **Termium** [CD Rom]. Ottawa: Minister of Supply and Services Canada.

[14] Canada. Public Works and Government Services, Translation Bureau (1997). **The Canadian Style, A Guide to Writing and Editing**. Toronto: Dundurn Press.

[15] Clas, André and Étienne Tifou (1989). **Introduction aux études linguistiques**. Course manual for LNG 1035. Montreal: La librairie de l'Université de Montréal.

[16] Cornell, Gary (1997). **Visual Basic 5 from the Ground Up**. Berkeley, California: Osborne/McGraw-Hill.

[17] Daille, Béatrice (1993). *Repérage et extraction de terminologie par une approche mixte statistique et linguistique*, **Traitement automatique des langues (TAL)** 36(1-2), pp. 101-118.

[18] David, Sophie and Pierre Plante (1990). *De la nécessité d'une approche morphosyntaxique en analyse de textes*, **Intelligence artificielle et sciences cognitives au Québec** 3(3), pp. 140-145.

[19] Desmet, Isabel and Samy Boutayeb (1994). *Terms and words: Propositions for terminology*, **Terminology** 1(2), pp. 303-325.

[20] Dodds de Wolf, Gaelan *et al.* (1983). **The Gage Canadian Dictionary**. Toronto: Gage Educational Publishing Company.

[21] Dorr, Bonnie Jean (1993). **Machine Translation: A View from the Lexicon**. Cambridge, Massachusetts: MIT Press.

[22] Drouin, Patrick (1996). *Définition d'une approche hybride pour l'identification automatique des termes complexes*. Paper given in a seminar entitled *Traductique*, Département de linguistique et traduction, Université de Montréal, March 10, 1996.

[23] Drouin, Patrick (1997). *Une Méthodologie d'identification automatique des syntagmes terminologiques : L'apport de la description du non-terme*, **Meta** 42(1), p.45-54

[24] Drouin, Patrick and Jacques Ladouceur (1994). *L'identification automatique des descripteurs complexes dans des textes de spécialité*, **Proceedings of the Workshop on Compound Nouns. Multilingual Aspects of Nominal Composition**, University of Geneva. Geneva, Switzerland, 2-3 December, 1994, pp. 18-28.

[25] Dubuc, Robert (1992). **Manuel pratique de terminologie**. Brossard, Quebec: Linguatech.

[26] Gaussier, Éric and Jean-Marc Langé (1993). *Modèles statistiques pour l'extraction de lexiques bilingues*, **Traitement automatique des langues (TAL)** 36 (1-2), p. 133.

[27] Gurewich, Nathan and Ori Gurewich (1997). **Teach Yourself Visual Basic 5 in 21 Days**. Indianapolis, Indiana: Sams Publishing.

[28] Habert, Benoît and Christian Jacquemin (1993). *Noms composés, termes, dénominations complexes : problématiques linguistiques et traitements automatiques*, **Traitement automatique des langues (TAL)** 34(2), p. 5.

[29] Hannan, Marie-Louise (1996). *The Use of Semantic Classes for Automatic Terminology Extraction*. Master's thesis. Département de linguistique et traduction, Université de Montréal.

[30] Hoffman, Lothar (1979). *Towards a Theory of LSP — Elements of a methodology of LSP analysis*, **Fachsprache** 1(2), pp. 12-17.

[31] Hoffman, Lothar (1980). *Language for Special Purposes as a Means of Communication: An Introduction*, **Langues de spécialité** (1), Quebec: Girsterm, pp. 3-38.

[32] Hoffman, Lothar (1984). *Seven Roads to LSP*, **Fachsprache** 6(12), pp. 28-38.

[33] Justeson, John S. and Slava M. Katz (1995). *Technical terminology: Some linguistic properties and an algorithm for identification in text*, **Natural Language Engineering** 1(1), pp. 9-27.

[34] Kocourek, Rostislav (1991). **La langue française de la technique et de la science vers une linguistique de la langue savante**. Second edition. Wiesbaden: Brandstetter Verlag.

[35] Lauriston, Andy (1993). *Le repérage automatique des syntagmes terminologiques*. Master's thesis. Département de linguistique, Université du Québec à Montréal.

[36] Lauriston, Andy (1994). *Automatic recognition of complex terms: Problems and the TERMINO solution*, **Terminology** 1(1), pp. 147-170.

[37] Levi, Judith N. (1978). **The Syntax and Semantics of Complex Nominals**. New York: Academic Press.

[38] L'Homme, Marie-Claude (no date given), **Manuel de formation LOGOS**.

[39] L'Homme, Marie-Claude (1991). *Constitution de bases de données textuelles : une définition des unités lexicales complexes*, **Le langage et l'homme** 26(1), pp. 33-45.

[40] L'Homme, Marie-Claude (1994). *Traitement des groupes nominaux en traduction automatique : opportunité d'un codage conceptuel*, **Proceedings of the Workshop on Compound Nouns. Multilingual Aspects of Nominal Composition**, University of Geneva. Geneva, 2-3 December, 1994, pp. 147-161.

[41] L'Homme, Marie-Claude (1997). *Méthode d'accès informatisé aux combinaisons lexicales en langue technique*, **Meta** 42(1), p. 15-23.

[42] L'Homme, Marie-Claude *et al.* (1996). *Definition of an evaluation grid for term-extraction software*, **Terminology** 3(2), pp. 291-312.

[43] L'Homme, Marie-Claude and Geertrudia de Rooij (1998). *Expérimentation du logiciel de dépouillement assisté par ordinateur ADEPTE-NOMINO en contexte de veille terminologique*. Rapport d'évaluation présenté à l'Office de la langue française, Montreal.

[44] Mandelbaum, David G. (1949). **Selected Writings of Edward Sapir in Language, Culture, and Personality**. Berkeley and Los Angeles: University of California Press.

[45] Meyer, Ingrid *et al.* (1996). *How can phraseology help?*, **Terminology** Volume 3(1), pp. 1-26.

[46] Marchand, H. (1969). **The categories and types of present-day English word-formation**. Wiesbaden: Harrassowitz.

[47] Microsoft Corporation (1997). **Microsoft Press Computer Dictionary**. Third Edition. Redmond, Washington: Microsoft Press.

[48] Mounin, Georges (1963). **Les problèmes théoriques de la traduction**. Paris: Gallimard.

[49] Newton, John (1992). **Computers in Translation: A Practical Appraisal**. New York: Routledge.

[50] Normand, Diane (1993). *Quand la terminologie s'automatise*, **Circuit** (36), pp. 29-30.

[51] Opitz, K (1980). *Language for Special Purposes: An Intractable Presence*, **Fachsprache** 2(1), pp. 21-27.

[52] Otman, Gabriel (1991). *Des ambitions et des performances d'un système de dépouillement terminologique assisté par ordinateur*, **La banque des mots**, n^{o} spécial 4, p. 59.

[53] Picht, Heribert (1987). *Terms and their LSP environment — LSP phraseology*, **Meta** 32(2), pp. 149-155.

[54] Picht, Heribert and Jennifer Draskau (1985). **Terminology: An Introduction**. Guildford, England: University of Surrey, Department of Linguistic and International Studies.

[55] Robert, Paul, Josette Rey-Debove and Alain Rey (1993). **Le nouveau petit Robert : dictionnaire alphabétique et analogique de la langue française**. Paris: Dictionnaires Le Robert.

[56] Rondeau, Guy (1984). **Introduction à la terminologie**. Chicoutimi, Quebec: Gaëtan Morin Éditeur.

[57] Sager, Juan C. (1990). **A Practical Course in Terminology Processing**. Amsterdam: John Benjamins Publishing Company.

[58] Sager, Juan C., David Dungworth and Peter F. McDonald (1980). **English special languages: principles and practice in science and technology**. Wiesbaden: Brandstetter Verlag.

[59] Sapir, Edward (1921). **Language: An Introduction to the Study of Speech**. New York: Harcourt, Brace and Company.

[60] Saussure, Ferdinand de (1974). **Cours de linguistique générale**. Paris: Payot. Published by Charles Bally and Albert Sechehaye with the cooperation of Albert Riedlinger.

[61] Selkirk, O. Elisabeth (1982). **The Syntax of Words**. Cambridge, Massachusetts: MIT Press.

[62] Somers, Harold (1996). **Terminology, LSP and Translation — Studies in language engineering in honour of Juan C. Sager**. Amsterdam: John Benjamins Publishing Company.

[63] Sta, Jean-David (1993). *Comportement statistique des termes et acquisition terminologique à partir de corpus*, **Traitement automatique des langues (TAL)** 36 (1-2), p. 119.

[64] TRADUCTIX Inc. (1994). **ATAO Traduction — Manuel de référence Version 1.4.** Montreal: TRADUCTIX.

[65] Williams, E. (1981). *On the Notions 'Lexically Related' and 'Head of a Word'*, **Linguistic Inquiry** (12), pp. 245-274.

Appendix I — Electronic texts used to compile the corpus of term candidates

Each of the texts listed below was scanned manually to create the human list of terms. They were then submitted to the term extraction utilities of LOGOS and ATAO to generate an output list of term candidates.

All these texts were obtained from corporate Web sites via the Internet, instead of being scanned from paper documentation. Consequently, our electronic texts do not contain the errors usually associated with optical character recognition (missing letters, accented characters mistaken for numbers, etc.) In addition, they were not spell checked or retouched in any manner prior to submission. Not grooming the texts in the pre-processing phase was an attempt to emulate a real-life situation as closely as possible.

Cisco and Hewlett-Packard

"Secure Web Transaction Solution Architecture"
Taken from Website http://www.hp.com

This highly-technical text discusses the use of firewalls and other security devices to create a secure runtime environment where it is possible to make safe transactions over the Internet or an intranet/extranet. It also shows how the companies' proposed solution can help corporate Web sites to better manage traffic and make them impenetrable by crackers.

Color Solutions

"Perfect Color — ColorBlind ICC Workflow"
Taken from Website http://www.color.com

This document gives an in-depth look at the many features of the ColorBlind software program and shows why it is indispensable to graphics professionals. This text presupposes that the reader is familiar with graphic arts software and specialty printing.

Color Solutions

“Perfect Color — Build Your Own Color Server”

Taken from Website http://www.color.com

This white paper discusses why a colour server program should be used as part of graphic development and printing. As with the other document from the same company, it is assumed that the reader is familiar with graphic arts and printing.

Delphi Software

“WinHelp 4.0 — Evaluating RTF-based Online Help for Microsoft Windows 95 and NT”

Taken from Website http://www.delphi.ie

In this white paper, a critical comparative evaluation is made between the two principal methods used to produce Windows Help files: compiled RTF files and HTML. Intended for beginning Help authors and managers faced with choosing between the two platforms, this text is only semi-technical and contains a substantial amount of non-technical content, such as disclaimers, author biography and sales information on the products Delphi software recommend and sell.

Emprise Technologies

“White Paper: A Tale of Two Environments… LAN Backup to the Mainframe”

Taken from Website http://www.emprisetech.com

This light-hearted, yet highly-technical document explains how the IS and LAN specialists in the same company can work together instead of against one another to recycle mainframe systems that are falling from use as a new way to backup files. An understanding of how LANs work is required to fully grasp this text.

Exabyte Corporation

“White Paper: Tape Drive Performance Benchmark — Performance of the Exabyte Mammoth, Sony SDX-300, Quantum DLT 7000 compared”

Taken from Website http://www.exabyte.com

In this text, three heavy-duty tape drives are compared in many areas and the results interpreted. This technical article is intended for network administrators and purchasers of corporate computer equipment.

Hummingbird Communications

“White Paper: Integrating Windows NT and Enterprise Computer Systems”

Taken from Website http://www.hummingbird.com

Beginning with a detailled description of their solution and why it is effective at uniting UNIX and Windows NT systems in the same internetwork, then providing us with three case studies to prove their point, this is a technical marketing document aimed at corporate decision makers who understand the MIS side of business, as well as the technical managers in charge of implementing such a solution in the company.

MCS Corporation

“Frequently Asked Questions — RAID”

Taken from Website http://www.mcscorp.com

A highly-technical FAQ with a narrow focus, this document gives specific information on the different types of RAID (Redundant Array of Inexpensive Disks) and discusses the strong and weak points of each. A certain amount of domain-specific knowledge is required to understand this text.

Strategic Research Corporation

“Who’s Minding the Cache?

A White Paper Discussing Data Warehousing Performance Tuning through Storage”

Taken from Website http://www.sresearch.com

This document outlines methods corporations can use to fine-tune the performance of their data warehouse, such as caching, duplexing and employing disk array technology. It is assumed that the reader is familiar with how networks operate and has basic notions of datawarehousing.

Sun Microsystems

“Jelp Context Sensitive Help System for Java™”

Taken from Website http://www.sun.com

The Help system used for Java applets (HTML-based Help) does not currently support context-sensitive Help. This product, released by the creator of Java, is a user assistance alternative that supports this type of Help. Highly informative without being overly technical, this documents explains how the product works, as well as what it can and cannot offer the consumer.

Appendix II — Corpus statistics

II.i. Linguistic categories, term formations and subject fields

After scanning our corpus manually, we compiled a number of statistics on our output in order to better understand the scope of our results before attempting to compare them with the machine output generated by ATAO and LOGOS. These statistics are summarized in the second table of this Appendix. A description of each column of the table of corpus statistics based on our term scanning is given in the first table immediately below to help the reader interpret the results.

Consult Chapter 5 of this work for examples of each term formation type and a detailed discussion of our scanning results.

Column name	Description
`Text name`	This is the name of the electronic text that was part of the corpus scanned for terms.
`Words`	This is the word count of the electronic text that was part of the corpus scanned for terms, as calculated by Microsoft Word.
`total # of terms`	This is the total number of terminological units the electronic text contained after manual scanning.
`% terms in text`	This is the number of terminological units the electronic text contained, expressed as a percentage of the text's total word count.
`# non CS-terms`	This is the total number of terminological units not from the field of computer science the electronic text contained after manual scanning.
`% non-CS terms`	This is the total number of terminological units not from the field of computer science the electronic text contained, expressed as a percentage of the text's total word count.
`# of simple terms`	This is the total number of simple (one-word) terms the electronic text contained after manual scanning.
`# of 2-unit terms`	This is the total number of two-unit terms the electronic text contained after manual scanning.
`# of 3-unit terms`	This is the total number of three-unit terms the electronic text contained after manual scanning.
`# of 4-unit terms`	This is the total number of four-unit terms the electronic text contained after manual scanning.

Column name	Description
`# of 5-unit terms`	This is the total number of five-unit terms the electronic text contained after manual scanning.
`# of 6-unit terms`	This is the total number of six-unit terms the electronic text contained after manual scanning.
`# of 7-unit terms`	This is the total number of seven-unit terms the electronic text contained after manual scanning.
`% simple terms`	This is the number of simple (one-word) terms the electronic text contained, expressed as a percentage of the text's total number of terms.
`% 2-unit terms`	This is the number of two-unit terms the electronic text contained, expressed as a percentage of the text's total number of terms.
`% 3-unit terms`	This is the number of three-unit terms the electronic text contained, expressed as a percentage of the text's total number of terms.
`% 4-unit terms`	This is the number of four-unit terms the electronic text contained, expressed as a percentage of the text's total number of terms.
`% 5-unit terms`	This is the number of five-unit terms the electronic text contained, expressed as a percentage of the text's total number of terms.
`% 6-unit terms`	This is the number of six-unit terms the electronic text contained, expressed as a percentage of the text's total number of terms.
`% 7-unit terms`	This is the number of seven-unit terms the electronic text contained, expressed as a percentage of the text's total number of terms.
`# terms ADJ + N`	This is the total number of terminological units formed using the adjective + noun pattern that the electronic text contained after manual scanning.
`# terms N + N`	This is the total number of terminological units formed using the noun + noun pattern that the electronic text contained after manual scanning.
`# [N + prep + N] + N`	This is the total number of terminological units formed using the [noun + preposition + noun] + noun pattern that the electronic text contained after manual scanning.
`# simple N terms`	This is the total number of terminological units solely comprised of a single noun that the electronic text contained after manual scanning.

Column Name	Description
# terms ADJ + [ADJ + N]	This is the total number of terminological units formed using the adjective + [adjective + noun] pattern that the electronic text contained after manual scanning.
# terms ADJ + [N + N]	This is the total number of terminological units formed using the adjective + [noun + noun] pattern that the electronic text contained after manual scanning.
# terms N + [ADJ + N]	This is the total number of terminological units formed using the noun + [adjective + noun] pattern that the electronic text contained after manual scanning.
# terms N + [N + N]	This is the total number of terminological units formed using the noun + [noun + noun] pattern that the electronic text contained after manual scanning.
# simple V terms	This is the total number of terminological units solely comprised of a single transitive, intransitive or reflexive verb that the electronic text contained after manual scanning.
# terms V + part.	This is the total number of terminological units formed using the noun + particle pattern that the electronic text contained after manual scanning.
# simple ADJ terms	This is the total number of terminological units solely comprised of a single adjective that the electronic text contained after manual scanning.
# terms [ADJ + N] + N	This is the total number of terminological units formed using the [adjective + noun] + noun pattern that the electronic text contained after manual scanning.
# terms [N + N] + N	This is the total number of terminological units formed using the [noun + noun] + noun pattern that the electronic text contained after manual scanning.
# acronyms	This is the total number of terminological units in the electronic text that contained an acronym after manual scanning. This figure includes both lone acronyms and complex terms that had an acronym as one of their elements.
# other formations	This is the total number of terminological units formed using any pattern not described above that the electronic text contained after manual scanning.

Column name	Description
`# N`	This is the total number of terminological units in the electronic text that contained a noun after manual scanning. This figure includes both simple noun terms and complex terms that had a noun as one of their elements.
`# ADJ`	This is the total number of terminological units in the electronic text that contained an adjective after manual scanning. This figure includes both simple adjective terms and complex terms that had an adjective as one of their elements.
`# V`	This is the total number of terminological units in the electronic text that contained a verb after manual scanning. This figure includes both simple verb terms and complex terms that had a verb as one of their elements.
`% Nouns`	This is the total number of terminological units in the electronic text that contained a noun after manual scanning, expressed as a percentage of the text's total number of terms. This figure includes both simple noun terms and complex terms that had a noun as one of their elements.
`% ADJ`	This is the total number of terminological units in the electronic text that contained an adjective after manual scanning, expressed as a percentage of the text's total number of terms. This figure includes both simple adjective terms and complex terms that had an adjective as one of their elements.
`% Verbs`	This is the total number of terminological units in the electronic text that contained a verb after manual scanning, expressed as a percentage of the text's total number of terms. This figure includes both simple verb terms and complex terms that had a verb as one of their elements.

The statistics on the list of manually-extracted terms appear on the following page.

Appendix II — Corpus Statistics

Text Name	Words	total # of terms	% of terms in text	# non-CS terms	% of non-CS terms	# of simple terms	# of 2-unit terms	# of 3-unit terms	# of 4-unit terms	# of 5-unit terms	# of 6-unit terms	# of 7-unit terms	% simple terms	% 2-unit terms	% 3-unit terms	% 4-unit terms	% 5-unit terms	% 6-unit terms
Hummingbird Communications White Paper: Integrating Windows NT and Enterprise Computer Systems	6679	441	6.60%	30	6.80%	70	171	140	50	7	3	0	15.87%	38.78%	31.75%	11.34%	1.59%	0.68%
Emprise Technologies White Paper: A Tale of Two Environments… LAN Backup to the Mainframe	3077	197	6.40%	14	7.11%	35	106	46	8	2	0	0	17.77%	53.81%	23.35%	4.06%	1.02%	0.00%
Strategic Research Corporation “Who’s Minding the Cache?” A White Paper Discussing Data Warehousing Performance Tuning through Storage	4135	275	6.65%	12	4.36%	51	149	54	15	5	0	1	18.55%	54.18%	19.64%	5.45%	1.82%	0.00%
Secure Web Transaction Solution Architecture by Cisco & Hewlett-Packard	2888	159	5.51%	8	5.03%	30	83	42	10	4	1	0	18.87%	52.20%	26.42%	6.29%	2.52%	0.63%
Jelp Context Sensitive Help System for JavaTM	778	42	5.40%	5	11.90%	6	25	8	2	0	0	0	14.29%	59.52%	19.05%	4.76%	0.00%	0.00%
Perfect Color — ColorBlind ICC Workflow	2979	140	4.70%	77	55.00%	27	71	35	5	2	0	0	19.29%	50.71%	25.00%	3.57%	1.43%	0.00%
Frequently Asked Questions — RAID	1768	67	3.79%	0	0.00%	21	32	8	4	2	0	0	31.34%	47.76%	11.94%	5.97%	2.99%	0.00%
An Exabyte white paper:Tape Drive Performance Benchmark — Performance of the Exabyte Mammoth, Sony SDX-300, Quantum DLT 7000 compared	1614	58	3.59%	1	1.72%	6	31	11	9	0	1	0	10.34%	53.45%	18.97%	15.52%	0.00%	1.72%
WinHelp 4.0 Evaluating RTF-based Online Help for Microsoft Windows 95 and NT	4504	108	2.40%	5	4.63%	15	62	28	2	1	0	0	13.89%	57.41%	25.93%	1.85%	0.93%	0.00%
Perfect Color — Build your Own Color Server	1336	66	4.94%	40	60.61%	8	36	21	1	0	0	0	12.12%	54.55%	31.82%	1.52%	0.00%	0.00%
Total:	**29758**	**1563**	**5.25%**	**192**	**12.28%**	**269**	**766**	**393**	**106**	**23**	**5**	**1**	**17.21%**	**49.01%**	**25.14%**	**6.78%**	**1.47%**	**0.32%**

Appendix II — Corpus Statistics

% 7-unit terms	# ADJ + N terms	# N + N terms	[N + prep + N] + N terms	# simple N terms	# ADJ + [ADJ + N] terms	# ADJ + [N + N] terms	# N + [ADJ + N] terms	# N + [N + N] terms	# simple V terms	# terms V + part.	# simple ADJ terms	# [ADJ + N] +N terms	# [N + N] + N terms	# acronyms	# other formations	# N	# ADJ	# V	% Nouns	% ADJ	% Verbs
0.00%	62	115	1	60	10	45	0	33	8	1	6	19	32	112	71	442	9	10	95.88%	1.95%	2.17%
0.00%	49	57	1	34	3	17	1	4	4	0	0	7	13	37	11	197	0	4	98.01%	0.00%	1.99%
0.36%	66	87	0	39	3	18	2	2	6	2	5	8	13	35	30	265	5	11	94.31%	1.78%	3.91%
0.00%	37	46	0	24	6	15	3	3	6	1	1	7	9	16	16	164	3	7	94.25%	1.72%	4.02%
0.00%	8	17	0	5	2	1	1	5	1	1	0	0	0	2	1	40	0	2	95.24%	0.00%	4.76%
0.00%	24	46	0	23	2	10	1	14	3	0	1	1	6	24	11	136	3	3	95.77%	2.11%	2.11%
0.00%	10	22	0	20	0	2	0	0	1	0	1	2	3	15	8	66	2	1	95.65%	2.90%	1.45%
0.00%	13	18	0	6	3	5	0	1	0	0	0	1	2	1	10	59	0	0	100%	0.00%	0.00%
0.00%	33	36	0	13	1	5	1	3	2	0	0	9	7	14	8	116	0	2	98.31%	0.00%	1.69%
0.00%	15	21	0	8	2	6	3	8	1	0	0	0	1	16	1	65	0	1	98.48%	0.00%	1.52%
0.06%	**317**	**465**	**2**	**232**	**32**	**124**	**12**	**73**	**32**	**5**	**14**	**54**	**86**	**272**	**167**	**1550**	**22**	**41**	**96.09%**	**1.36%**	**2.54%**

II.ii. Subject fields recorded in TEMS

As might be expected with a collection of texts exceeding 30,000 words in length, not all terms that were extracted belong to the subject field of computer science. The following is a list of the subject fields to which terms in our corpus belong. As stated in Chapter 4 of this study, the subject fields used as part of our research were verified against those used in the Termium linguistic data bank. A pie chart illustrating the relative proportions of each subject field is given in Section 5.2.2 of this work.

- Accounting
- Biological sciences
- Business and finance
- Business and supply management
- Corporate Policy
- Economics
- Education and training
- Electromagnetic radiation
- Engineering tests and reliability
- General vocabulary (LSP)
- Handtools
- Labour relations
- Law
- Marketing and law
- Marketing and trade
- Occupational titles
- Optical instruments
- Printing and graphic arts
- Printing machines and equipment
- Stock exchange
- Telematics
- Mathematical geography
- Electrical engineering
- Systematic Botany

II.iii. Excerpt from the human list

This list below is a sample from the over 1500 terms manually extracted from our collection of electronic texts. As explained in Chapter 4 of this work, this list of terms was used as a point of comparison for the two automatic term-extraction software programs evaluated in this study: ATAO and LOGOS.

- array feature
- bulk upload
- cache mirror
- complex query
- daisy chain
- database activity
- database administrator
- database cache
- disk capacity
- higher-level cache
- hot spare
- hot swapping
- I/O bandwidth
- load balance
- mainframe administrator
- management tool
- multiple request
- network administrator
- network connection
- OLTP application
- parallel server
- physical "fetch"
- physical failure
- physical I/O
- RAID level
- read-centric operation
- read-intensive operation
- read-mostly nature
- read/only query
- relational database
- remote monitoring
- schedule slippage
- SCSI bus
- sequential processing
- service arrangement
- service relationship
- sustained bandwidth
- swap out
- system rollout
- Ultra-SCSI bus
- update performance
- user concurrency
- volume manager
- wash out
- write cache
- write-intensive operation

Appendix III — Sample output from ATAO

The first and main list in the *.CH document produced by ATAO is the "Complex Nominals List." It is considered to be the most important list the system gives the user. The other lists produced by ATAO contain phrasal units that assist with translation and the MPT environment; however, they are not strictly term extraction lists per se; so they were not included here for discussion.

ATAO takes the name of the file it scanned then assigns the file extension *.CH to differentiate between this and the other lists it produces. The *.CH file is filtered for frequency through the thresholding setting defined by the user.

The *.CH list is in effect a compendium of three sublists:

- Classic nominal strings
- Nominal Strings by last element
- Prepositional nominal strings

The relative length of these three sublistings can vary significantly, depending on factors such as thresholding and the subject of the input text. Sample output from each of these sublists is provided below.

Classic nominal strings: The complex nominals list contains all compound term candidates ATAO extracted from the input text. It also contains coordinate terms such as *backup and restore process*. Term candidates that contain numeric elements are extracted for this list and contain a generic numeric symbol {0} to represent the number.

The headword of each entry on the complex nominals list is lemmetized (or reduced to its canonical form) to facilitate sorting and interpreting the list contents. The other elements of the complex nominals list are left untouched. The entries on the complex nominals list are sorted by the last unit they contain, which is presumed to be the headword. This alternative sorting approach has the advantage of elucidating the relationship between the term candidates on the list because the terms with a common theme (headword) are presented together.

Pointy brackets (**<** and **>**) are used by ATAO to indicate that the lexical unit they offset belongs immediately after the lexical unit by which the complex nominal term candidate was sorted in the list. For example, the term candidate ***average access time*** was sorted by the element ***access*** because ATAO determined that ***access*** could potentially be the term candidate's headword. As such, it was presented in the complex nominals list as ***<time> average access***. In cases where ATAO is unsure which element is the headword of the extracted term candidate, it will present the same term candidate in the complex nominals list expressed in up to four different ways to ensure it is listed with conceptually-related term candidates to assist the human terminologist

```
                                           <time> average access
                                           <time> average access
                                              information access
                                                 Internet access
                                                 Internet access
                                                  printer access
                                                  printer access
                                      Controlling printer access
                                                     read access
                                                     read access
                                                   timely access
                                                   timely access
                                     location transparent access
                    authorized external users transparent access
                                                     user access
                                            speeding user access
                                               multiple actuator
                                                      IP address
                                                      IP address
                                               single IP address
                                       <translations> IP address
                               network manager and administrator
++ management utility enabling network manager and administrator
                                               LAN administrator
                                               LAN administrator
                                               LAN administrator
                                               LAN administrator
```

```
                  freeing LAN administrator
              <knowledge> LAN administrator
                      network administrator
                      network administrator
                 best network administrator
<trying> overburdened network administrator
      <learned years> network administrator
                       system administrator
                       system administrator
            Enterprise system administrator
                             rapid adoption
                 Microsoft's rapid adoption
                             full advantage
                             full advantage
                adaptive security algorithm
                Adaptive Security Algorithm
                               large amount

                                      <...>

                              outside world
                              outside world
                           technical writer
                           technical writer
                   Skilled Technical Writer
        <throughput> lowest sustained write
         <throughput tests> sustained write
               <Throughput> Sustained write
                                  next year
                       <promises> next year
                                  time zone
                                  time zone
```

Nominal strings by last element: The purpose of this list is to detect certain types of expressions that contain a verb. It may contain some of the same entries as the complex nominals list. Its entries are presented in the same manner as those of the Classic nominal strings list.

```
have become key
  Advanced mode
  advanced mode
in a batch mode
     batch mode
 Dangerous mode
 dangerous mode
  Standard mode
  standard mode
      user mode
      user mode
 Storage option
 storage option
```

Prepositional nominal strings: This sublist regroups term candidates formed according to the N + PREP + other word forms pattern. In this list, the headword is the first element of the term candidate instead of the last. Consequently, this list is sorted by the first element of the extracted term candidate and not the last like the complex nominals list.

```
access to network
access to network
amount of data
application to NT
application to NT
College of Natural
College of Natural Sciences
College of Natural Sciences
cost of ownership
cost of ownership
degree of performance
degree of performance for
environment for applications
environment for applications
exceed for Windows NT
exceed for Windows NT
format for online
format for online
launch of Windows
launch of Windows
level of RAID
level of RAID

<...>

Suite for Windows NT
suite for Windows NT
traffic between multiple colocated servers
traffic between multiple colocated servers
wide range of
wide range of
XDK for Windows NT
XDK for Windows NT
XDK for Windows NT
```

Appendix IV — Screen captures from the LogosClient version 3.0 interface

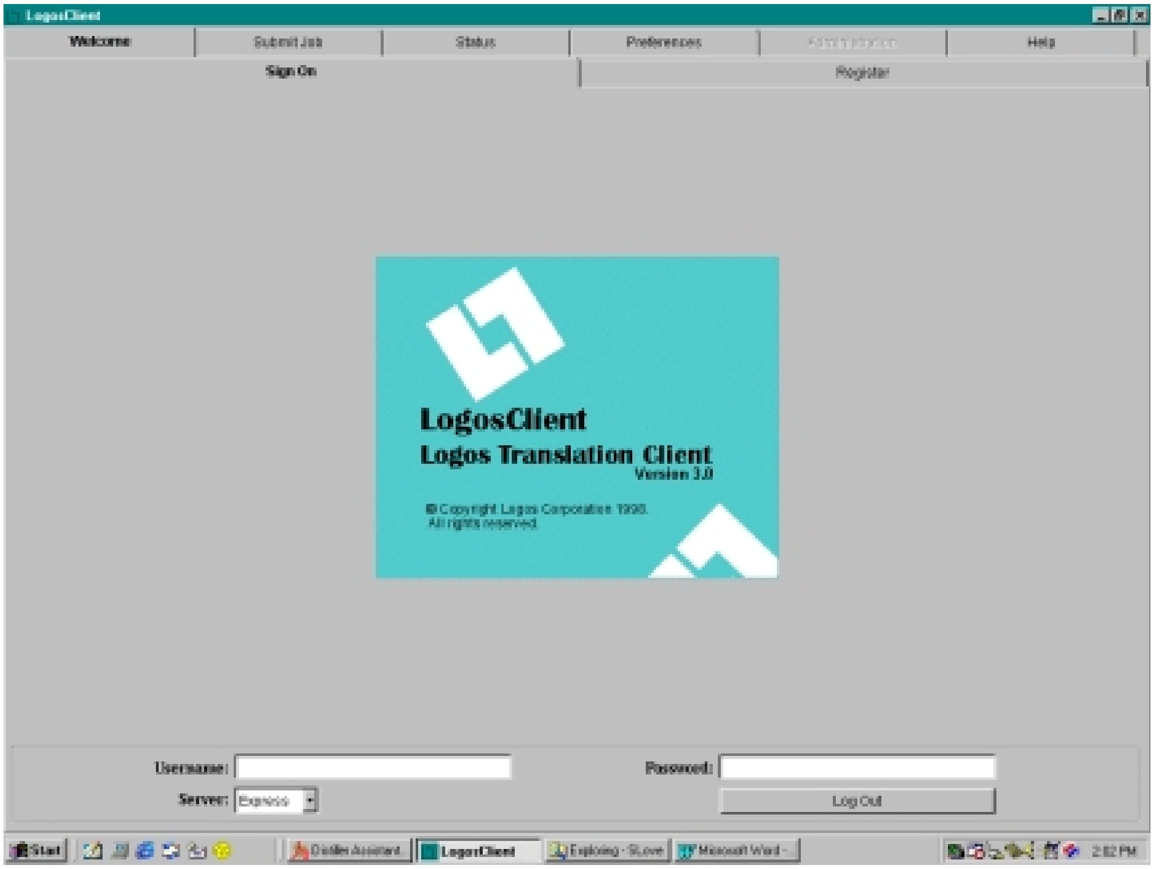
LogosClient is the machine translation interface that is used to send documents to be processed from the user desktop to the Logos machine translation engine called LogosServer. The texts whose terms were extracted and analyzed for the purposes of this study all went through LogosClient for submission to LogosServer.

This appendix contains a collection of LogosClient screen images that were pertinent to this research project. The system settings used for submitting our corpus of electronic texts to LogosClient are depicted in these screen captures to make the images more meaningful to the reader and to show which settings were used for our corpus.

The LogosClient uses a tabbed interface similar to our TEMS application. Unlike TEMS, however, LogosClient employs a two-tiered system of tabs called “pages,” whereby the lower, secondary series of tabs is dynamically and contextually updated to reflect the active tab from the primary series of tabs located at the top of the screen. This is a unique approach (one that we had never seen before) and not a very standard type of graphical user interface (GUI). However, we must concede that this method of presentation avoids creating an interface with an unwieldy number of tabs and limits them to only two rows instead of three or even more. Another surprise when investigating the LogosClient was that the online user assistance was placed in a tab and neither in a standard Windows Help file (*.HLP) or represented by a ? in the toolbar of the application toolbar.

Although the overall look-and-feel of this application was not what we expected, it does not in any way take away from its capabilities as a term extraction utility of multilingual machine translation program.

The **Welcome Page** is where the user logs in to the system before beginning a term extraction job.



LogosClient
Logos Translation Client
Version 3.0

© Copyright Logos Corporation 1998.
All rights reserved.

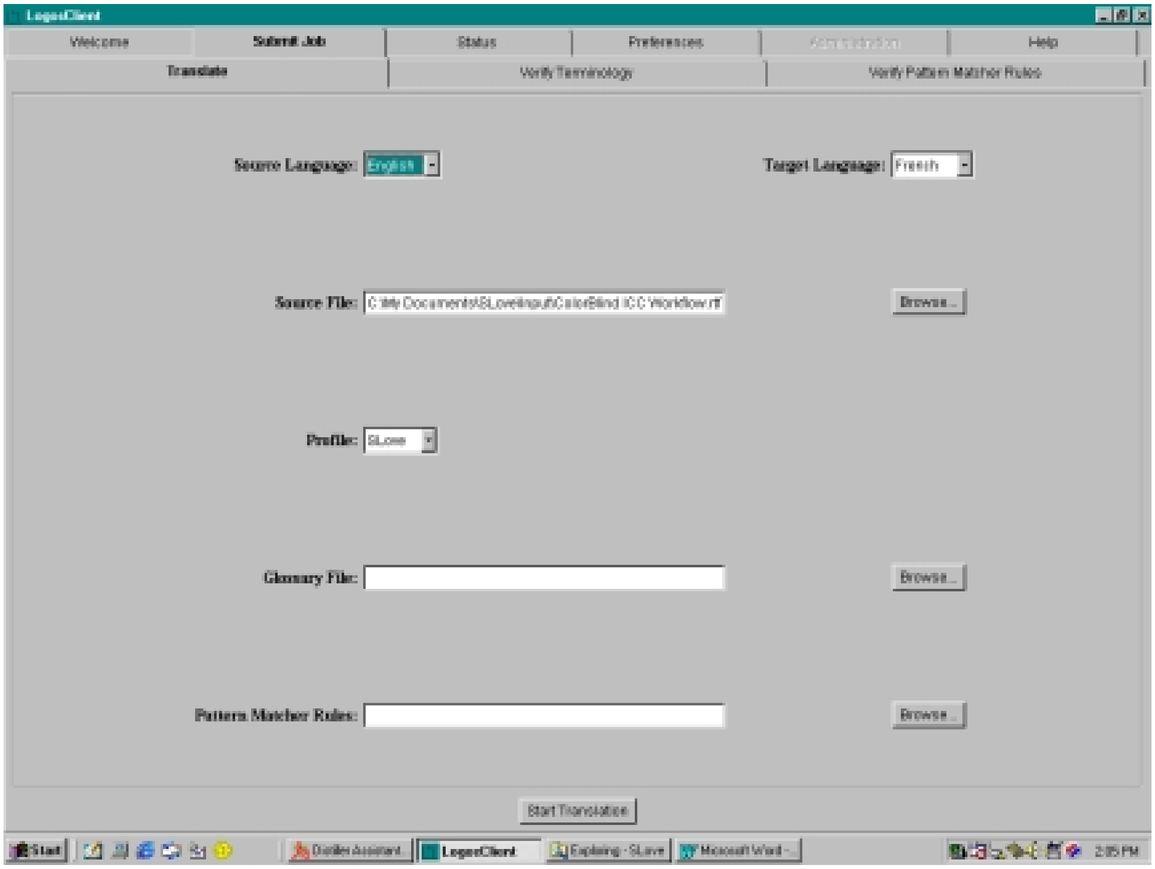
Username:

Password:

Server: Express

Figure V - 1: The LogosClient Login Screen

In the **Translation Submission Page**, the user indicates the source and target language (for translation purposes) and specifies which file (and file format) is to be processed.



Source Language: English

Target Language: French

Source File: C:\My Documents\SLove\input\ColorBlind ICC Workflow.rtf

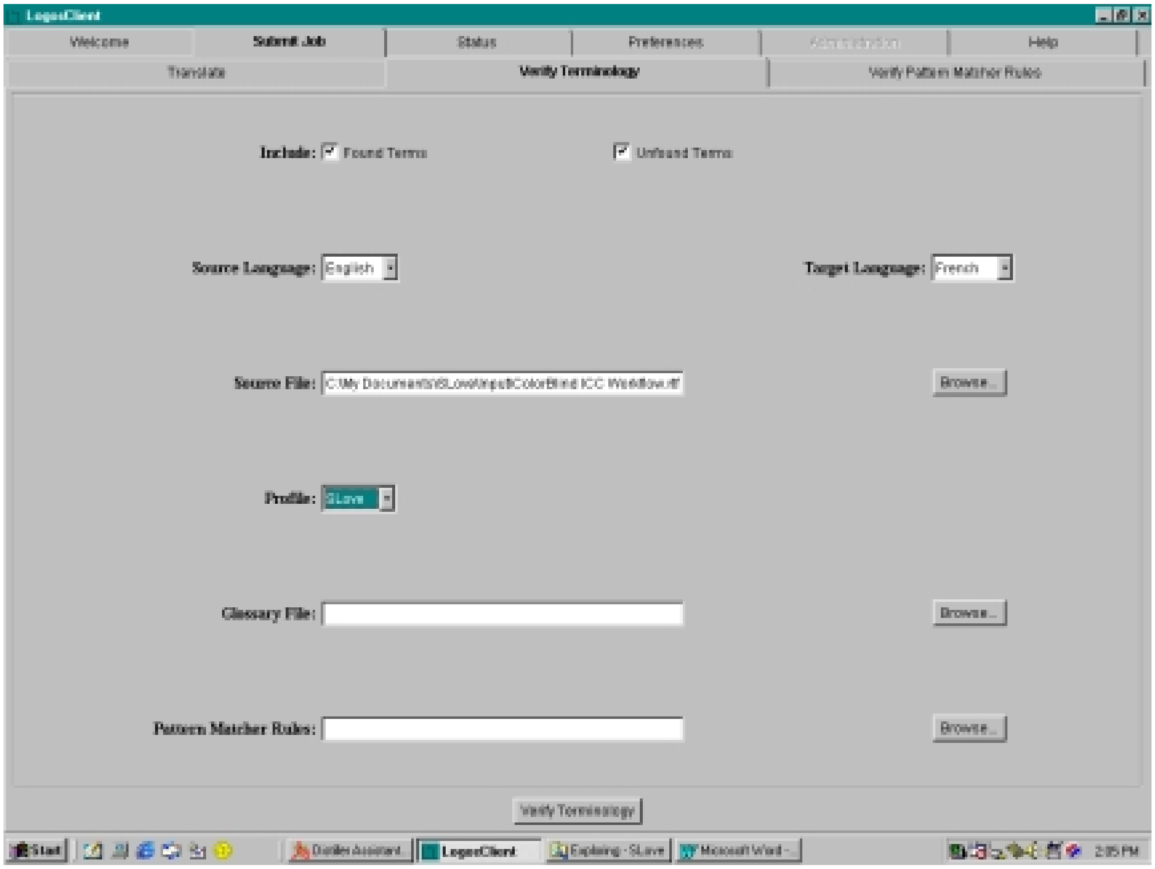
Profile: SLove

Glossary File:

Pattern Matcher Rules:

Figure V - 2: The LOGOS Translation Submission Page

In the **Terminology Search Submission Page**, the user indicates whether or not found terms or unfound or both are to be extracted while examining the source document’s terminology.



Include: ☑ Found Terms ☑ Unfound Terms

Source Language: English

Target Language: French

Source File: C:\My Documents\SLove\Input\ColorBlind ICC Workflow.rtf

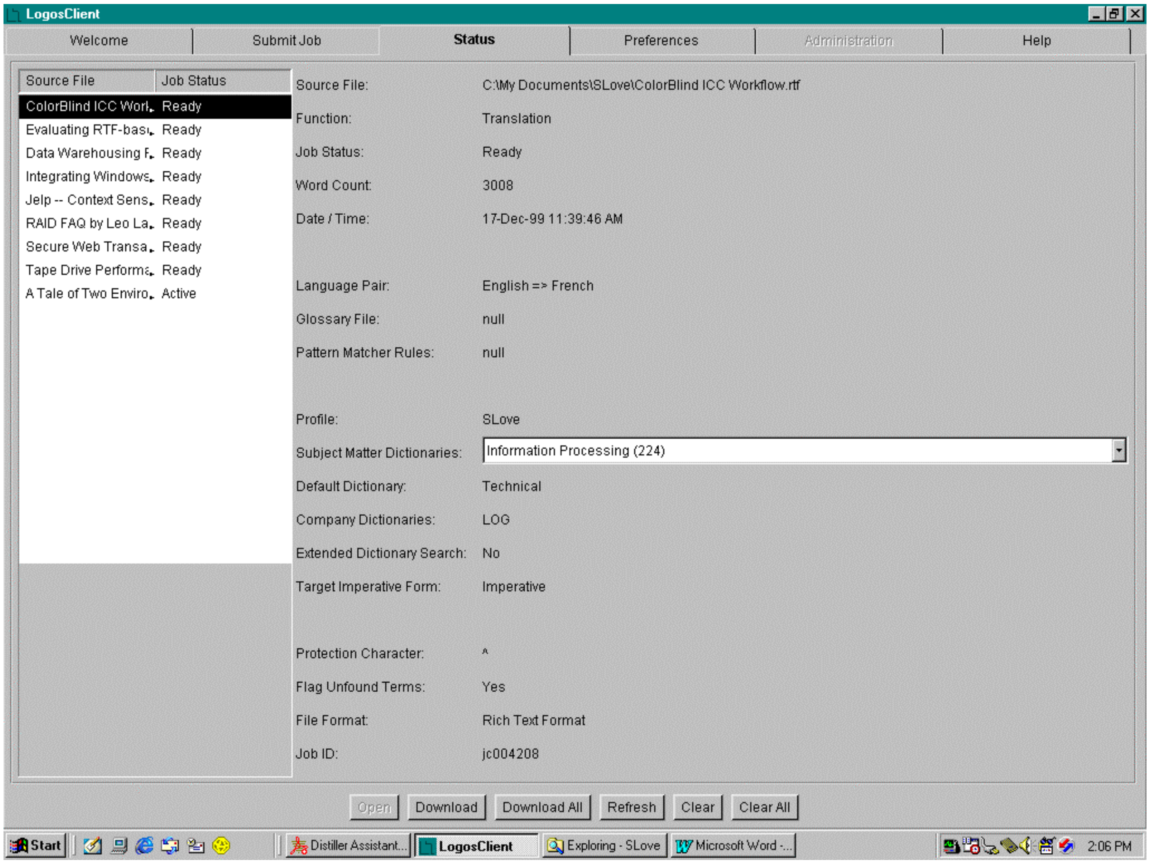
Profile: SLove

Glossary File:

Pattern Matcher Rules:

Figure V - 3: The LOGOS Terminology Search Submission Page

In the **Status Page**, the user can view the status of the submitted job(s), as well as most settings.



Source File	Job Status
ColorBlind ICC Worl	Ready
Evaluating RTF-bas	Ready
Data Warehousing F	Ready
Integrating Windows	Ready
Jelp -- Context Sens	Ready
RAID FAQ by Leo La	Ready
Secure Web Transa	Ready
Tape Drive Perform	Ready
A Tale of Two Enviro	Active

Source File: C:\My Documents\SLove\ColorBlind ICC Workflow.rtf
Function: Translation
Job Status: Ready
Word Count: 3008
Date / Time: 17-Dec-99 11:39:46 AM

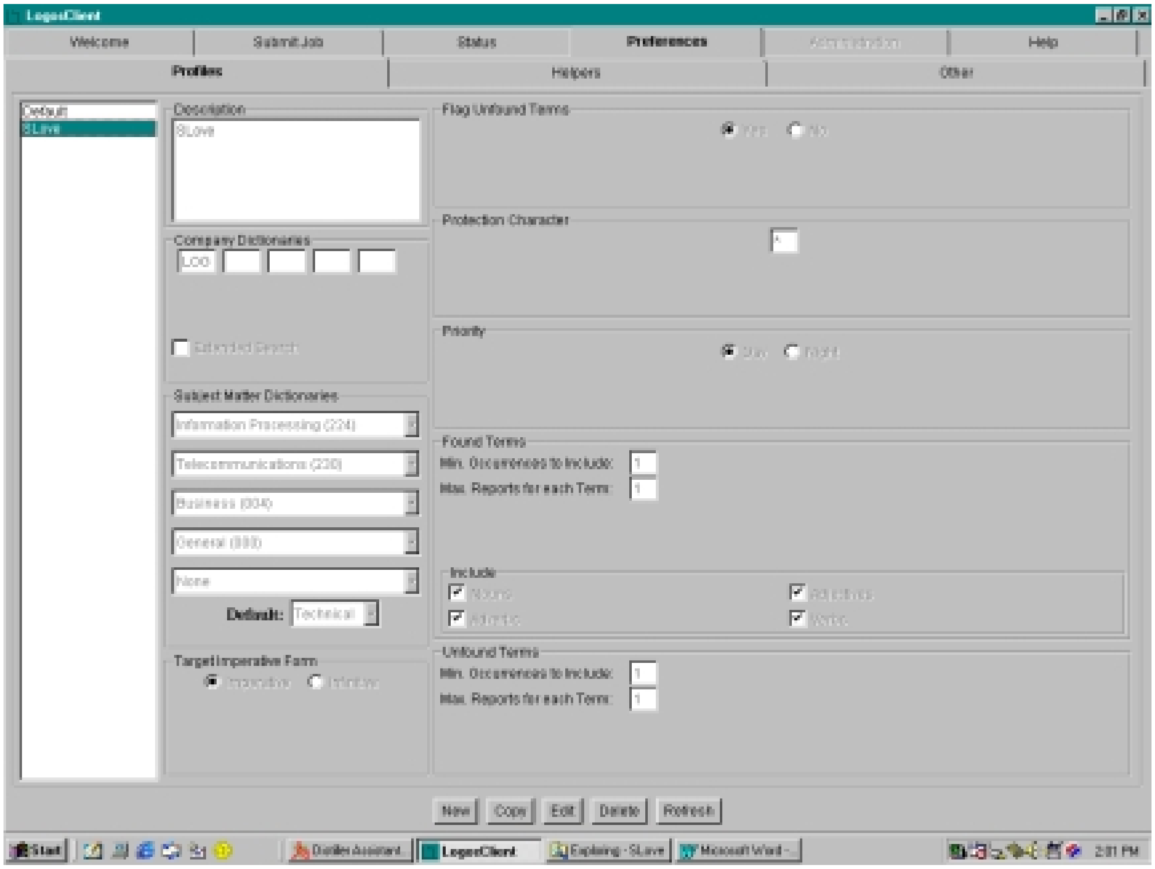
Language Pair: English => French
Glossary File: null
Pattern Matcher Rules: null

Profile: SLove
Subject Matter Dictionaries: Information Processing (224)
Default Dictionary: Technical
Company Dictionaries: LOG
Extended Dictionary Search: No
Target Imperative Form: Imperative

Protection Character: ^
Flag Unfound Terms: Yes
File Format: Rich Text Format
Job ID: jc004208

Figure V - 4: The LOGOS Status Page

In the **Preferences Page**, the user defines the subject matter and general-language dictionaries to use while processing the job, as well as the job priority and other various job settings.



Default
SLove

Description: SLove

Company Dictionaries: LOO

□ Extended Search

Subject Matter Dictionaries:
Information Processing (224)
Telecommunications (230)
Business (004)
General (000)
None
Default: Technical

Target Imperative Form: ● Imperative ○ Infinitive

Flag Unfound Terms: ● Yes ○ No

Protection Character: ^

Priority: ● Day ○ Night

Found Terms
Min. Occurrences to Include: 1
Max. Reports for each Term: 1

Include
☑ Nouns ☑ Adjectives
☑ Adverbs ☑ Verbs

Unfound Terms
Min. Occurrences to Include: 1
Max. Reports for each Term: 1

Figure V - 5: The LOGOS Profile Page

Appendix V — Sample output from LOGOS version 3.0

When our corpus was submitted to LOGOS Version 3.0, the raw machine output resembled the sample output provided in this Appendix. It must be said that the format was initially a comma-delimited text file; however, once properly imported into Microsoft Access, it took a format similar to our sample here. A description of each column of the sample LOGOS data table is given to help the reader interpret the results.

Column name	Description
`Record ID`	This is the number used to identify each extracted term candidate individually
`English`	This column indicates that the input corpus texts were written in English.
`French`	This column indicates that the selected translation language is French. Although we did not utilize any of LOGOS' translation capabilities, this field remained in the machine output.
`Found or Unfound Term`	This field indicates whether or not the extracted term candidate already appeared in the LOGOS default lexicon at the time of the extraction.
`Term Candidate in Original Form`	This is the simple term candidate extracted by LOGOS in its original form.
`Term Candidate in  Canonical Form`	This is the simple term candidate extracted by LOGOS in canonical form.
`Term Candidate in Canonical Form`	This is the simple term candidate extracted by LOGOS in canonical form, if two transformations were required to reduce it to canonical form.
`Part of Speech`	This is the part of speech assigned to the term candidate by LOGOS.
`French Translation`	This is the French-language translation for the extracted term candidate in canonical form. Although we did not utilize any of LOGOS' translation capabilities, this field remained in the machine output.
`Used by LOGOS`	This field was reserved for use by LOGOS.
`French Gender`	This is the gender of the extracted term candidate once it was translated to French by LOGOS. Although we did not utilize any of LOGOS' translation capabilities, this field remained in the machine output.

Column name	Description
System Dictionary	This is the indicator of the system dictionary used by LOGOS to extract the term candidates from our corpus.
Spec. Dictionary	This is the indicator of the principal specialized dictionary used by LOGOS to extract the term candidates from our corpus.
Used by LOGOS	This field was reserved for use by LOGOS.
Used by LOGOS	This field was reserved for use by LOGOS.
Used by LOGOS	This field was reserved for use by LOGOS.
Context	This is the immediate context surrounding the simple term candidate extracted by LOGOS.

The sample data from Logos appears on the following page.

Appendix V — Sample Data from Logos

Record ID	English	French	Found or Unfound Term	Term Candidate in Original Form	Term Candidate in Canonical Form	Term Candidate in Canonical Form	Part of Speech	French Trans-lation	Used by LOGOS	French Gender	System Dictionary	Spec. Dictionary	Used by LOGOS	Used by LOGOS
1	EN	FR	Found	ability	ability	ability	Noun	capacité	V	Fem	LOG	1000	0	2
2	EN	FR	Found	achieve	achieve	achieve	Verb	accomplir	V		LOG	1000	36	1
3	EN	FR	Found	booklets	booklet	booklet	Noun	brochure	V	Fem	LOG	1000	32	2
4	EN	FR	Found	browsing	browsing	browsing	Adj	parcourant	V		LOG	31224	66	2
5	EN	FR	Found	business	business	business	Noun	affaires	V	Fem	LOG	1000	49	1
6	EN	FR	Found	clean	clean	clean	Adj	propre	V		LOG	1000	47	1
7	EN	FR	Found	colors	color	color	Noun	couleur	V	Fem	LOG	1000	41	1
8	EN	FR	Found	video conferencing	video conferencing	video conferencing	Adj	donnant une vidéoconférence	V		LOG	1000	149	1
9	EN	FR	Found	consistency	consistency	consistency	Noun	cohérence	V	Fem	LOG	31224	60	1
10	EN	FR	Found	developers	developer	developer	Noun	révélateur	V	Masc	LOG	1000	32	2
11	EN	FR	Found	easy	easy	easy	Adj	facile	V		LOG	1000	0	2
12	EN	FR	Found	embedded	embed	embed	Verb	incruster	V		LOG	1000	52	2
13	EN	FR	Found	engaged	engage	engage	Verb	engager	V		LOG	31224	19	1
14	EN	FR	Found	features	feature	feature	Noun	caractéristique	V	Fem	LOG	1000	8	2
15	EN	FR	Found	fonts	font	font	Noun	police	V	Fem	LOG	31224	25	1
16	EN	FR	Found	context-sensitive help	context-sensitive help	context-sensitive help	Noun	aide contextuelle	V	Fem	LOG	31224	99	1
17	EN	FR	Found	including	include	include	Verb	inclure	V		LOG	1000	17	1
18	EN	FR	Found	innovations	innovation	innovation	Noun	innovation	V	Fem	LOG	1000	17	2
19	EN	FR	Found	learn	learn	learn	Verb	apprendre	V		LOG	1000	59	1
20	EN	FR	Found	migrating	migrate	migrate	Verb	transférer	V		LOG	31230	30	1
21	EN	FR	Found	multimedia	multimedia	multimedia	Adj	multimédia	V		LOG	1000	137	1
22	EN	FR	Found	recognized	recognize	recognize	Verb	reconnaître	V		LOG	1000	13	1
23	EN	FR	Found	robust	robust	robust	Adj	robuste	V		LOG	1000	0	1
24	EN	FR	Found	cost savings	cost savings	cost savings	Noun	économie	V	Fem	LOG	1000	37	1
25	EN	FR	Found	software	software	software	Noun	logiciel	V	Masc	LOG	1000	0	1
26	EN	FR	Found	tap	tap	tap	Noun	tape	V	Fem	LOG	1000	59	1
27	EN	FR	Found	technology	technology	technology	Noun	technologie	V	Fem	LOG	1000	12	1
28	EN	FR	Found	translate	translate	translate	Verb	traduire	V		LOG	1000	20	1
29	EN	FR	Found	users	user	user	Noun	utilisateur	V	Masc	LOG	31224	15	2
30	EN	FR	Found	written	write	write	Verb	écrire	V		LOG	1000	12	3

Appendix V — Sample Data from Logos

Used by LOGOS	Context
68	Ability to modify system fonts and colors
50	t those application's help files to achieve the same functionality in Java.
63	History tracking for topics and booklets
22	rray of features including a powerful searching capability, topic browsing, history tracking, <F1> context-sensitive help and more.
4	Advances in technology along with the demands of business have ushered in a new era of sophisticated software applic
34	intentionally kept the API for Jelp simple and clean.
69	Ability to modify topic header fonts and colors
79	elp is written entirely in Java, it will allow us to continually enhance it in the future to take advantage of distributed help systems, multimedia, video conferencing and more.
43	look and feel of the standard toolbars and menus to provide consistency with their hosting environment.
7	Unfortunately, users as well as developers are required to learn these new applications as quickly as possib
31	Easy to use and Implement.
19	Because Jelp is written entirely in Java, it can be embedded seamlessly in a Java application or applet.
49	Further, companies engaged in migrating applications to Java could use Jelp to conve
22	rray of features including a powerful searching capability, topic browsing, history tracking, <F1> context-sensitive help and more.
68	Ability to modify system fonts and colors
22	rray of features including a powerful searching capability, topic browsing, history tracking, <F1> context-sensitive help and more.
22	rray of features including a powerful searching capability, topic browsing, history tracking, <F1> context-sensitive help and more.
55	Additionally, as innovations appear in Java, these innovations can be easily incorporated into Jelp.
7	Unfortunately, users as well as developers are required to learn these new applications as quickly as possib
49	Further, companies engaged in migrating applications to Java could use Jelp to conve
79	elp is written entirely in Java, it will allow us to continually enhance it in the future to take advantage of distributed help systems, multimedia, video conferencing and more.
12	CreativeSoft recognized that there was a significant void for a powerful and robust help system for Java applications.
52	Robust and Powerful.
11	curve, all of which translate into a cost savings.
47	software.
54	Our decision to develop Jelp entirely in Java allows us to tap into its power.
4	Advances in technology along with the demands of business have ushered in a new era of sophisticated software applic
11	curve, all of which translate into a cost savings.
7	Unfortunately, users as well as developers are required to learn these new applications as quickly as possib
17	help system written entirely in Java.