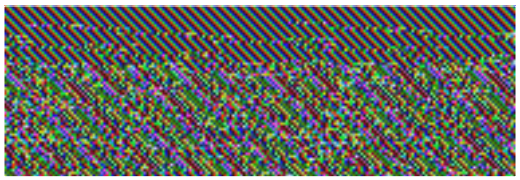

NetOwl(TM) Extractor Technical Overview
March 1997

1 Overview

NetOwl Extractor is an automatic indexing system that finds and classifies key phrases in text, such as personal names, corporate names, place names, dates, and monetary expressions. NetOwl Extractor finds all mentions of a name and links names that refer to the same entity together. NetOwl Extractor combines dynamic recognition with static look-up to achieve high accuracy and coverage at very high speed.

NetOwl Extractor is adaptable to information technology applications. Text retrieval engines can accept proper names as special index terms to achieve higher accuracy. Textual databases can store the proper names in special fields to improve access to on-line documents. Hypertext browsers can link documents using names found across multiple documents, allowing web browsers to connect web sites to on-line text.

2 Background

Proper names play a crucial role in information management, both in specific applications and in the underlying technologies that drive the applications [1]. For example, the Seymour on-line photo retrieval system from Picture Network International [2], and its cousin, Publisher's Depot, recognize proper names in photo captions and in user queries and automatically search for variations of each name, because so many users require photographs of specific individuals or locations. Nearly every business newspaper and magazine provides a special company name index, and many print names in boldface, because this allows readers to spot articles of interest and identify who's in the news. Information extraction systems [3,4,5,6,7] use name and phrase recognition as the first step toward more detailed analysis of text, and custom news ("clipping") systems often provide special features for users to track particular people and companies.

In spite of the recognized importance of names in applications, most text processing applications, including search systems, spelling checkers, and document management systems, do not treat proper names correctly. This is because proper name identification and interpretation is actually a very complex task: there are an infinite variety of possible names, which can be highly variable in structure; names can overlap with other names and with other words, and even simple clues like capitalization can be misleading. Table 1 provides many examples of the problems encountered.

Description	Examples
Names overlapping other names	Murphy Oil vs. Murphy Department Stores
Names overlapping words	Prime Computer vs. prime beef
Organization / place ambiguity	State College, PA vs. Imperial College, London
Corporations containing person names	J. C. Penney Co. Cray Computer
Corporations containing place names	Sante Fe Southern Pacific Corp. Bethlehem Steel Corp.
Names containing AND	Atlantis Mill and Lumber Co. vs. Honda and Toyota Motor Corp.
Ambiguous first names	Chip Roth vs. chip price Rich Bond vs. rich investment
Misleading capitalization	J. F. Kennedy vs. P. O. Box
Generically-named organizations	Bank of Japan First Bank World Bank

Table 1: Name Recognition Problems

A basic approach to name recognition is to perform a simple table look-up. This approach requires *all* names must be known and unambiguous. Both of these assumptions prove problematic for most general information technology applications, where there are always new and ambiguous names. Carnegie's NameFinder product [8] incorporated this approach and had some success in limited applications. Oracle's ConText product [9] creates an index of proper names along with many other words and phrases. NetOwl Extractor differs from NameFinder because it identifies names that aren't on any known list, and it differs from ConText because it can correctly classify names (e.g., person, place, company) and resolve references to avoid ambiguity in the index.

NetOwl Extractor's approach is to recognize names using two major knowledge sources: (1) a representation of the structure of names, (e.g. people names have first names and last names, company names often include "Inc." or "Corp.") and (2) detailed linguistic knowledge that identifies the context in which names can appear (e.g. corporate executives are often described along with a title, other descriptive information, and the name of their company).

3 System Description

NetOwl Extractor consists of a software engine that applies name recognition rules to text, supported by lexical resources and limited lists of proper names, as illustrated in Figure 1. Users implement an application driver that invokes the engine and configures the processing. NetOwl Extractor can either generate a document that has the names annotated with SGML (Standard Generalized Markup Language) [10] or provide a table of the names with indices to the text.

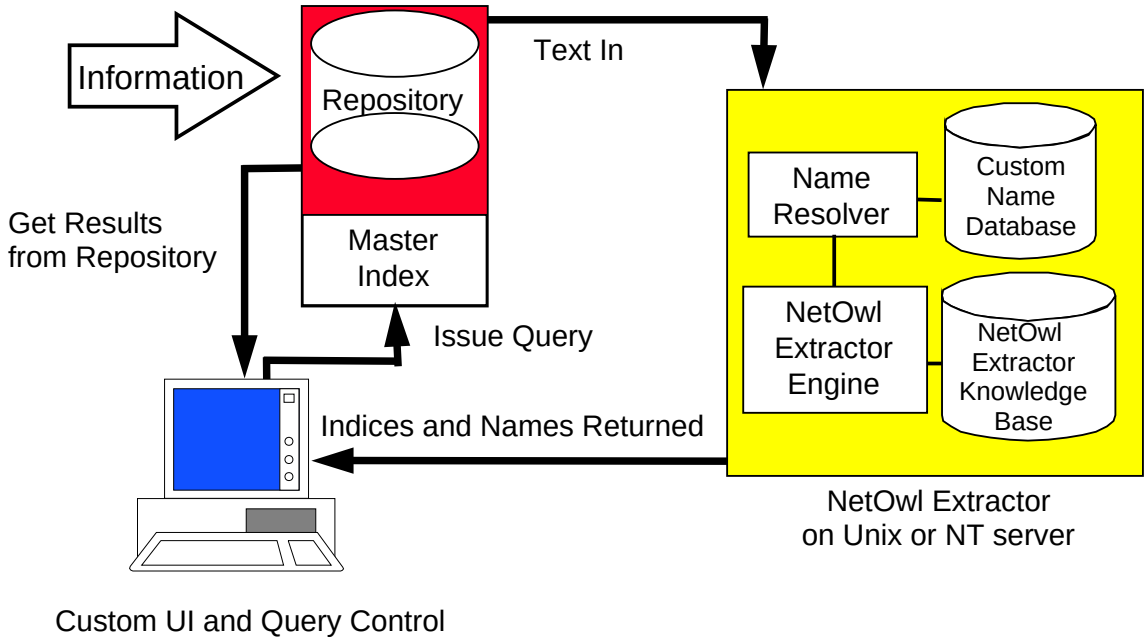

Figure 1: NetOwl Extractor System Components

A *name recognition rule* consists of a pattern and an action. The pattern is similar to a regular expression and consists of special operator and operands that match portions of text. Typically, patterns recognize structural or contextual indicators of names and thus perform dynamic recognition, as illustrated in Table 2. The action performs operations on the text, such as tagging a name with a classification. The rules are partitioned to form *processing phases* that primarily recognize one class of name. For example, NetOwl Extractor has separate phases for recognizing personal names and organizational names. This feature allows the recognition of certain classes of names to impact the recognition of other names. The *lexical resources* contain information about words, such as their part-of-speech (e.g. noun) and their meaning (e.g. personal title).

Type of Rule	Pattern	Action	Example
structural	capitalized personal first name + capitalized word	Tag match as PERSON	George Bush
contextual	personal title + capitalized word sequence	Tag match as PERSON, excluding title	Mr. George Bush
structural	caitalized word sequence + corporate indicator	Tag match as ENTITY, company subtype	Digital Equipment Corp.
contextual	job position + OF + capitalized word sequence	Tag match as ENTITY, excluding job positon and OF	president of Digital Equipment
structural	capitalized word sequence + caapitalized location noun	Tag match as PLACE, assign subtype using location noun	Orange County
contextual	capitalized word sequence + , U.S. state name ,	Tag match as PLACE, city subtype, excluding state,	Bethlehem, PA.,

Table 2: Name Recognition Rules

The *name lists* contain those names that can be tagged without recognition rules and thus constitute a static look-up of names. These lists primarily contain common or problematic names, such as nations, U.S. states, and household acronyms (e.g. "IBM", "3M"), and can be customized by the user. The dynamic recognition rules over-ride the static look-up, thus allowing local or global context to confirm or invalidate the name lists.

For each document, NetOwl Extractor segments the text into tokens such as words, punctuation marks, and numbers. NetOwl Extractor attaches lexical information to the token using the lexical resources. Tokens are then grouped into sections, such as paragraphs, to form a document structure. If the document is annotated with SGML, the document structure is hierarchical, as illustrated in Figure 2.

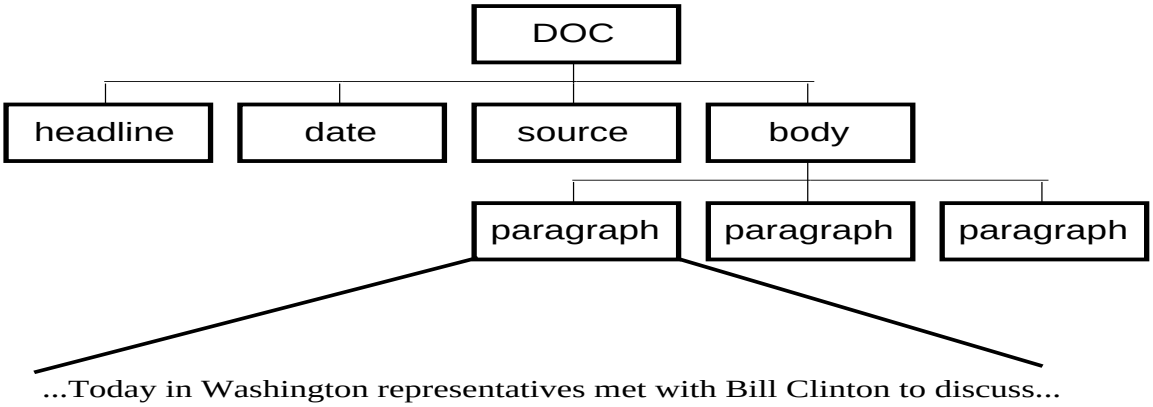

Figure 2: NetOwl Extractor Document Structure

NetOwl Extractor processes each section according to a user-assigned *processing class.* A processing class defines what pattern matching phases NetOwl Extractor should use to analyze the text. This feature allows NetOwl Extractor to process sections of a document using only those rules that are appropriate for the type of text contained in the section. For example, headlines require special treatment because all words are often capitalized and they often contain abbreviated forms of names fully mentioned in the text body.

As names are recognized, NetOwl Extractor records them in order to support the recognition of other forms of the same entity, such as abbreviations, acronyms, and personal last names. NetOwl Extractor links names as they are recognized, and uses a special processing phase to find other forms that were missed. For example, NetOwl Extractor will recognize and record "George Bush" as a person, then recognize "Mr. Bush" as a person and link it to "George Bush", and then look for other mentions of "Bush" missed by the contextual recognition rules.

4 Tag Definitions

NetOwl Extractor classifies names according to a general-purpose definition. The definition consists of a top-level category and a secondary classification. The top-level category represents the basic class of phrase, as listed below:

PERSON	named person or family
PLACE	political or geographical place name
ENTITY	named organizational entity, including facilities
TIME	dates, days of week, time of day, and other temporal expressions
NUMERIC	percentages, monetary values, measures
OTHER	other named things

The secondary classifications are self-explanatory and are as follows:

PERSON	none
PLACE	CONTINENT, COUNTRY, PROVINCE, COUNTY, CITY, REGION, DISTRICT, WATER, LANDFORM, ROADWAY
ENTITY	ORGANIZATION, COMPANY, GOVERNMENT, UNION, MILITARY EDUCATION, FACILITY, PUBLICATION
TIME	DATE, TIMEOFDAY, AGE, TEMPORAL
NUMERIC	PERCENT, MONEY, PHONENUM, MEASURE
OTHER	PRODUCT, EQUIPMENT, OTHER

The OTHER categories are open-ended and reflect the fact that there are large numbers of named things that are not persons, places, or entities.

5 Performance

NetOwl Extractor recognizes names and key phrases with high accuracy and coverage. Its performance can be quantified by comparing its results with manually tagged text and computing measures of performance to represent its score. Continual training on many different samples of text with manual tags has helped NetOwl Extractor to achieve near-human performance, and also ensures that the engine can be improved and further refined in specific domains or applications. Table 3 defines the three primary measures of performance. Typically, F-Measure is used with an equal weighting of recall and precision, and the resulting, simplified definition is also shown in Table 3.

Measure	Description	Definition
Recall	the percent of total names correctly found by the system *(also know as coverage)*	correct names / total possible names
Precision	the percent of the system's names that are correct *(also known as accuracy)*	correct names / total system names
F-Measure	the weighted combination of recall and precision	$\frac{(1 + ß^2) \times \text{precision} \times \text{recall}}{(ß^2 \times \text{precision}) + \text{recall}}$
F-Measure	recall and precision weighted equally (ß = 1.0)	$\frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$

Table 3: Measures of Performance

NetOwl Extractor attempts to maximize both recall and precision. However, high recall requires more processing time, which may be more important to some users. Therefore, NetOwl Extractor provides three configurations that trade performance for speed, as shown in Table 4. NetOwl Extractor achieves the trade-off by reducing the number of recognition rules, which is also listed in the table.

Configuration	Rules	Approximate Speed (Meg/hour)	Performance
BASE	226	80	high recall and precision
FAST	86	90	slightly lower recall, with slightly higher speed
FASTEST	59	110	lower recall, with significantly higher speed

Table 4: NetOwl Extractor Performance Configurations

The approximate speed measurements assume the processing of news stories on a Sun SPARCstation 20/71 running SunOS 4.1.4.

NetOwl Extractor's overall performance depends greatly on the quality, type, and format of the text. NetOwl Extractor achieves very high performance on well-edited, formatted news stories, such as the *Wall Street Journal* or *Associated Press* newswire. NetOwl Extractor will perform worse on poorly edited, unformatted text, such as E-mail or transcripts of interviews. Some publications are well-edited and formatted, but they make vast assumptions about the reader's knowledge and can be problematic to NetOwl Extractor. For example, computer technical reviews may assume that "Bill" is understood to mean "Bill Gates of Microsoft", and that "Apple" is understood to mean "Apple Computer Co." Such assumptions can cause problems for NetOwl Extractor's dynamic recognition rules, since the surrounding context may not be enough to effectively classify the name.

6 Government-Sponsored Benchmarks

The Sixth Message Understanding Conference (MUC-6) [11], sponsored by the Advanced Research Projects Agency, conducted an evaluation of name recognition as one of its benchmarks, called the Named Entity task. This task required systems to find and classify names and other key phrases according to a government-defined specification. The specification represented a subset of the NetOwl Extractor tag definition, and the task did not require the linking of co-referential names. However, this task represented a valid test of NetOwl Extractor's performance [12].

Fourteen other sites participated in the Named Entity task. Participants obtained the task specification and training data, received a blind test set, and submitted their results. The government used a scoring program to calculate the measures of performance, using F-measure (an equally weighted combination of recall and precision, shown in Table 3) as the primary result. The test consisted of 30 *Wall Street Journal* articles related to changes in management posts. The small size of the test, its source, and its topic focus all can influence the results. NetOwl Extractor out-performed all other sites, as illustrated in Table 5.

Measure	NetOwl Extractor	Next Best System	Human
F-measure	96.42	94.00	96.68
Recall	96	94 (with 93 precision)	95
Precision	97	96 (with 92 recall)	98
Total Errors	115	219	146

Table 5: MUC-6 Named Entity Test Results

The government conducted an inter-annotator comparison to estimate human performance on the test set, which also appears in Table 5. The comparison was not scientifically controlled, and the two human annotations occurred at different times, with slightly different task specifications. Although the actual quantitative results are approximate, the experiment did prove that human performance is not perfect. NetOwl Extractor achieved the estimated human performance, and actually committed fewer errors. The government also conducted statistical significance testing to determine which scores are essentially equal (i.e., their difference could have been obtained by chance). This testing determined that NetOwl Extractor's F-measure was significantly different from all other sites, and thus clearly superior.

For MUC-6, optional test runs were submitted to illustrate performance trade-offs of NetOwl Extractor, as listed in Table 6. The BASE configuration is the official MUC-6 test result for NetOwl Extractor. The FAST and FASTEST configurations reduce the name recognition analysis to increase the processing speed. The NO-NAMES

configuration is a variant of BASE in which approximately 500 household personal and organizational names were removed from NetOwl Extractor's name lists.

Configuration	CPU Time (seconds)	Speed (Meg/hour)	F-Measure
BASE	3.72	78.68	96.42
FAST	3.33	87.73	95.66
FASTEST	2.62	111.76	92.61
NO-NAMES	3.67	79.75	94.92

Table 6: MUC-6 Optional Test Results

7 Implementation

NetOwl Extractor is implemented in C++, and provides an ANSI C-linkable library that can be incorporated into applications. NetOwl Extractor provides versions for UNIX Workstations (SunOS, Solaris, and others) and PCs (e.g. Windows NT, 95, 3.1). NetOwl Extractor provides an easy-to-use API that allows the developer to set parameters, load documents, output SGML-annotated documents, and access the name table.

Creating an application program with the NetOwl Extractor API is very simple and straight-forward. The program first makes function calls to initialize the system, set up text processing parameters, and determine the tags to extract. The program can then load documents from file or memory, process the document, and iterate over the tags, extracting information such as character offset, the character string, and tag classification.

8 Summary

NetOwl Extractor is an automatic indexing system that finds and classifies names and other key phrases in text. NetOwl Extractor finds all mentions of a name and links names that refer to the same entity together. NetOwl Extractor combines dynamic recognition with static look-up to achieve high accuracy and coverage at very high speed. NetOwl Extractor's adaptable design provides a solution to a critical problem for information technology applications.

References

[1] Church, K. and Rau, L. Commercial Applications of Natural Language Processing, *Communications of the ACM*, November 1995.

[2] Flank, S., et al., Photofile: A Digital Library for Image Retrieval, *Proceedings of the IEEE International Conference on Multimedia Computing and Systems*, May 15-18, 1995, Washington, DC.

[3] Aone, C. et al., SRA: Description of the SRA System as Used for MUC-5, *Proceedings of the Fifth Message Understanding Conference (MUC-5)*, August 1993.

[4] Aone, C. et al., The Muraski Project: Multilingual Natural Language Understanding, *Proceedings of the ARPA Human Language Technology Workshop*, 1993.

[5] Appelt, D. et al., FASTUS: A Finite-State Processor for Information Extraction from Real World Text, *Proceedings of the 13th International Joint Conference on Artificial Intelligence (IJCA1-93)*, August 1993.

[6] Jacobs, P., Krupka, G., Rau, L., Lexico-semantic pattern matching as a companion to parsing in text understanding, *Fourth DARPA Speech and Natural Language Workshop*, February 1991.

[7] Jacobs, P., et al., GE-CMU: Description of the SHOGUN System Used for MUC-5, *Proceedings of the Fifth Message Understanding Conference (MUC-5)*, August 1993.

[8] Hayes, P., *NameFinder: Software that finds Names in Text*, Carnegie Group, Inc., May 1994.

[9] *ConText: Introduction to Oracle ConText*, Oracle Corporation, September 1993.

[10] Goldfarb, C., *The SGML Handbook*, Oxford, 1990.

[11] *Proceedings of the Sixth Message Understanding Conference (MUC-6)*, November 1995.

[12] Krupka, G., SRA: Description of the SRA System as Used for MUC-6, *Proceedings of the Sixth Message Understanding Conference (MUC-6*), November 1995.