# CRL/BRANDEIS: THE *DIDEROT* SYSTEM

*Jim Cowie, Louise Guthrie, Wang Jin, William Ogden, James Pustejovsky*[†], *Rong Wang, Takahiro Wakao, Scott Waterman*[†], *Yorick Wilks*

Computing Research Laboratory, New Mexico State University
Email: jcowie@nmsu.edu
[†]Computer Science Department, Brandeis University
Email: jamesp@cs.brandeis.edu

## 1. Description of Final System

Diderot is an information extraction system built at CRL and Brandeis University over the past two years. It was produced as part of our efforts in the Tipster project. The same overall system architecture has been used for English and Japanese and for the micro-electronics and joint venture domains.

The past history of the system is discussed and the operation of its major components described. A summary of scores at the 24 month workshop is given.

Because of the emphasis on different languages and different subject areas the research has focused on the development of general purpose, re-usable techniques. The CRL/Brandeis group have implemented statistical methods for focusing on the relevant parts of texts, programs which recognize and mark names of people, places and organizations and also dates. The actual analysis of the critical parts of the texts is carried out by a parser controlled by lexical structures for the 'key' words in the text. To extend the system's coverage of English and Japanese some of the content of these lexical structures was derived from machine readable dictionaries. These were then enhanced with information extracted from corpora.

The system has already been evaluated in the 4th Message Understanding Conference (MUC-4) where it was required to extract information from 200 texts on South American terrorism. Considering the very short development time allowed for this additional domain the system performed adequately. The system was then adapted to handle the business domain and also to process Japanese texts. Further extensions to the system allowed it to process texts on micro-electronics development. Performance at the 12 and 18 month evaluations was good for Japanese, but less good for English where we have been attempting to automate much of the development process. A more pragmatic approach was adopted for the final 24 month evaluation, using the same hand-crafted techniques for English as had been used for Japanese.

We estimate the amount of effort used directly to build the systems described here is around sixty man months.

### 1.1. Technical Approach

Our objectives in this research have been as follows:

- to develop and implement a language-independent framework for lexical semantic representation, and develop and implement a robust integration of that framework into a language-independent theory of semantic processing;
- to investigate and implement language independent techniques for automating the building of lexical knowledge bases from machine readable resources;
- to implement statistical tools for the tuning of lexical structures to specific domains;
- to implement the use of language independent statistical techniques for identifying relevant passages of documents for more detailed analysis;
- to develop and implement a set of robust multi-pass finite-state feature taggers;
- to develop and implement the equivalent methods for Japanese.

### 1.2. Process Flow

An outline of the functions of the main system modules are given here. This is intended to provide a context for the more detailed description of each module which follows. The structures of the Japanese and English systems are very similar. In the examples of intermediate output either Japanese or English may be shown. The system architecture is shown in figure 1.

The input text to the system is processed by three independent pre-processing modules:

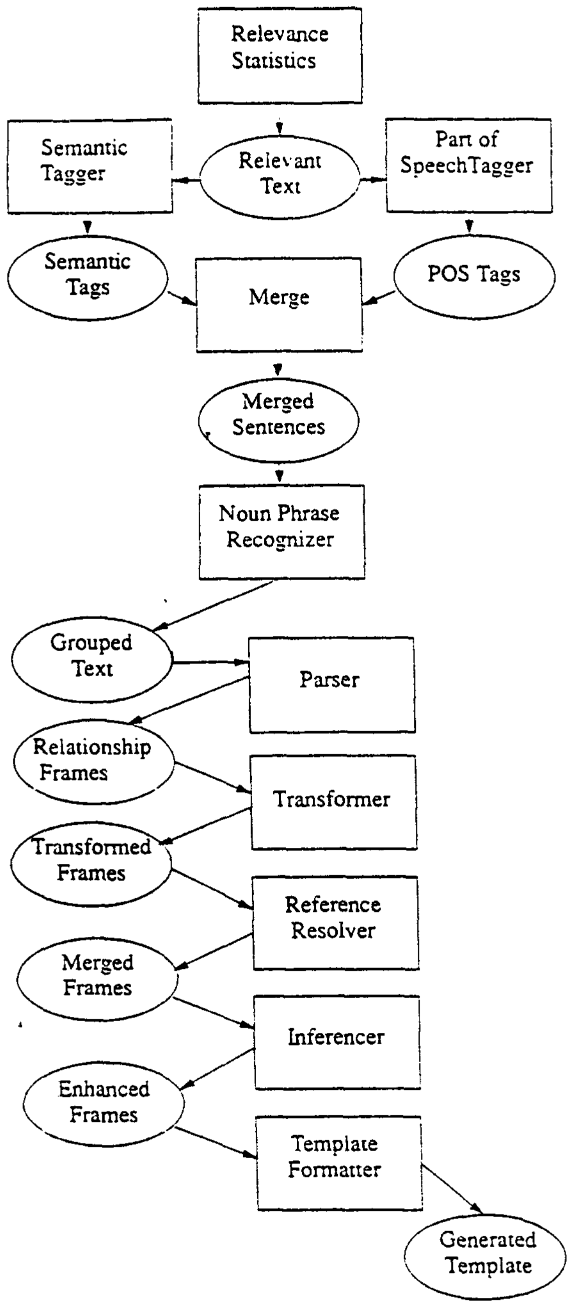


Figure 1: System Overview

- A chain of finite-state feature taggers - these mark: names, organization names, place names, date expressions and other proper names (depending on the domain),
- A part of speech tagger,
- A statistically based determiner of text relevance (micro only).

If the statistical determination rejects the text processing proceeds to the final output stage and an empty template is produced. Otherwise the results of the other two stages are converted to Prolog facts and these then pass into the head of a chain of processes each of which gives rise to further refinements of the text:

- Merge - Here semantic tags, which may mark phrasal units, are merged with POS tags, which mark individual words.
- Compound noun recognizer - this groups words and phrases into compound nouns using POS and semantic information.
- Parser - the relevant paragraph information is used to select which sentences to process further. The sentences containing the marked up noun-phrase groups are then parsed to produce a partially completed representation of the relevant semantic content of the sentence (frames).
- Reference resolver - the frames are then merged based on name matching and noun compounds beginning with definite articles.
- Template formatter - this transforms the resolved frames into the final output form.

## 1.3. Description of key modules and stages

**Statistical Filtering Techniques** Statistical information is used to predict whether a text holds important information that is relevant to completing a template. This allows the parser to skip non-relevant texts. This is based on word lists which are derived from training on relevant and irrelevant texts. The theoretical results on which the method [?] is based assure us that documents can be classified correctly if appropriate sets of words can be chosen for each document type. The method was only applied to the micro domain for MUC-5 as almost all texts in the joint venture domain are relevant and the use of this statistical method is essentially a way of improving precision in text filtering.

The results for the micro electronics domain for text filtering are 84% recall and 90% precision (73 and 83 at 18 month) for Japanese, and 78% recall and 83% precision (77 and 76 at 18 month) for English.

## 1.4. Semantic Tagging

This component is based on a pipeline of programs. These are all written in 'C' or flex. It marks organization names, human names, place names, date expressions, equipment names, process types and a variety of measurements (including money). Many of these have converted forms and additional values attached by the tagger.

The tagging programs use three separate methods —

- Direct recognition of already known unambiguous names, using a longest string match.
- Recognition using textual patterns only.
- Two pass method marking ambiguous, but potential names, and subsequently verifying they fit a pattern.
- final pass recognizing short forms and isolated occurrences of names not in a strong context

The system uses the case of letters used when available. The final text is tagged using SGML-like markers.

```
BRIDGESTONE SPORTS CO. SAID FRIDAY
IT HAS SET UP A  JOINT VENTURE  IN
TAIWAN  WITH A LOCAL CONCERN AND A
JAPANESE  TRADING HOUSE TO
PRODUCE GOLF CLUBS TO BE SHIPPED
TO JAPAN.
```

```
<organ> BRIDGESTONE SPORTS CO.
{type([[entity_type,'COMPANY']])} <\endorgan>
said
<date> FRIDAY{type([[date,'241189']])}
<\enddate> it has set up a joint
venture in <country> TAIWAN
{type([[nationality,'TAIWAN']])}
<\endcountry> with a local concern and a
<country>japanese {type([[nationality,'JAPAN']])}
<\endcountry>
trading house to produce golf clubs to be shipped
to <country> JAPAN {type([[nationality,'JAPAN']])}
<\endcountry>.
```

At this point the tags are converted into Prolog facts:

```
organ('BRIDGESTONE SPORTS CO.',
       type([[entity_type, 'COMPANY']])),
res('said',type([[undefined,'said']])),
time('FRIDAY',type([[date_adverb,'UNSPEC'],
     [date,'241189']])),
cs('it',type([[it,[pron]]])),
cs('has',type([[has,[pastv,presv]]])),
gls('set up',type([['set up',v]])),
cs('a',type([[a,[determiner]]])),
gls('joint venture',type([['joint venture',comp]])),
date_adverb('in',type([[date_adverb,during]])),
country('TAIWAN',type([[nationality,'TAIWAN']])),
cs('with',type([[with,[prep]]])),
```

The Japanese system preprocesses the article to change the original encoding (Shift JIS) to EUC for a given article. The original and unsegmented text goes through a series of taggers for known names, i.e. organizations, places, GLS verbs. This process is exactly the same as in the English system. The next step is to tag organization, personal and place names which are not known to the system. These are detected by using local context, using Japanese-specific patterns, which use particles, specific words and the text tags to recognize the unknown names. In addition, date expressions are tagged and changed into the normalized form. Date expressions in the Japanese articles seem straightforward, for example, '20 nichi' (20 day) is used even if the document date is 21st and 20th can be expressed as 'yesterday', and this convention 'XX day' (where XX is a number) to express a date is consistently used in the articles. Era names such as '昭和' (Showa) or '平成' (Heisei) are Japanese specific and the year in the era, e.g. '' (Showa 60th year), is correctly recognized and normalized. Here is the first sentence of a typical article after the tagging process.

```
<\organ> 東京海上火災保険
{type([[entity_type,'COMPANY']])}
<\endorgan> は
<\time> 今月から {type([[date_adverb,after],
[date,'8501']])} <\endtime>
<\organ> 大和証券
{type([[entity_type,'COMPANY']])}
<\endorgan> と
<\gls> 提携して
{type(['提携する',v])} <\endgls>
```

Just as for the English system this is then converted into the form of Prolog facts ready to be read into the merging phase.

**Part-Of-Speech Tagging** English text is also fed through the POST part of speech tagger. This attaches the Penn Treebank parts of speech to the text. The output is converted to Prolog facts. The Japanese text is segmented with part-of-speech information by the JUMAN program, which was developed by Kyoto University. The following is the result for exactly the same sentence. The segmented units are converted to Prolog facts ready to input to the next stage.

```
   juman('東京','proper_noun').
   juman('海上','proper_noun').
   juman('火災','normal_noun').
   juman('保険','normal_noun').
   juman('は','topic_particle').
   juman('今月','normal_noun').
```

```
juman('から','case_particle').
juman('大和','normal_noun').
juman('証券','normal_noun').
juman('と','case_particle').
juman('提携','noun_verb').
juman('して','verb').
```

**Merging** The semantic and syntactic information are merged to give lexical items in the form of triples. The merging is done in such a way that if it is not possible to match up words (eg due to different treatments of hyphens) a syntactic tag of 'UNK' is allocated and merging continues with the next word.

**Noun Phrase Grouping** Noun phrases are identified by scanning back through a sentence to identify head nouns. Both semantically and syntactically marked units qualify as nouns. The grouping stops when closed class words are encountered. A second forward pass gathers any trailing adjectives. The main use of the noun phrase in the present system is to attach related strings to company names to help with the reference resolution. They are also used by a retrieval process which uses the string to determine the SIC code industry type.

A similar grouping is carried out for Japanese.

```
noun_phrase([[undefined,house]],
 [unit(cs,a,type([[a,[determiner]]]),['DT']),
  unit(country,japanese,type([[nationality,'JAPAN'],
       [word_type,sp_noun]]),['JJ']),
  unit(res,trading,type([[undefined,trading]]),
       ['NN']),
  unit(res,house,type([[undefined,house]]),
       ['NN'])])

noun_phrase(money,
 [unit(num,'20',type([[num_value,20]]),['CD']),
  unit(num,million,type([[num_value,1000000]]),
       ['CD']),
  unit(money,'NEW TAIWAN dollars',
       type([[denom,'TWD']]),['NP','NP','NNS'])])
```

**Parsing** The parser has GLS cospecification patterns built into it. It uses these and ancillary rules for the recognition of semantic objects to fill a frame format which was given as an application specific field in the GLS entry. The frame formats provide a bridge between the sentence level parse and the final template output. Semantic objects are named in the cospecification and special rules which handle type checking, conjunction and co-ordination are used to return a structure for the object. The following shows an example of a tie-up between two companies. The child company is unmatched, shown by an underscore. The parser has grouped a date with one of the companies. The tie-up status is provided by the GLS template semantics.

```
prim_tie_up(1,1,[
 [[f(name,_9947,[unit(organ,'東京海上火災保険',
    type([[entity_type,'COMPANY']]),
         [proper_noun])]),
   f(entity_type,_9953,[unit(organ,
     '東京海上火災保険',
     type([[entity_type,'COMPANY']]),
          [proper_noun])])]],
  [[f(name,_10102,[unit(organ,'大和証券',
     type([[entity_type,'COMPANY']]),
          [proper_noun])]),
   f(entity_type,_10108,[unit(organ,'大和証券',
     type([[entity_type,'COMPANY']]),
          [proper_noun])]),
   f(time,_10114,[unit(time,'今月から',
     type([[date_adverb,after],[date,'8501']]),
          [proper_noun])])]]],_,
    [f(tie_up_status,existing,[])]).
```

**Transforming** The transformer module takes input from the parser and does the following things-

- format changes
- generation of values for all the factoids
- frame restructuring (e.g. form a simple set for all manufacturers found in a capability frame produced by the parser).

**Reference Resolution** The task of this component is to gather all the relevant information scattered in a text together. The major task is to resolve reference or anaphora. For the current application only references between tie-up events, between entities, and between entity relations are considered.

Since entities are expressed in noun phrases, the references for entities are resolved by resolving the reference between noun phrases. Since the entity can either be referred to by definite or indefinite noun phrase or by name, it is necessary to detect the reference between two definite or indefinite noun phrases, between two names, as well as between one name and one definite or indefinite noun phrase. All entities are represented as frames of the form:

```
entity(Sen#, Para#, Noun-phrase, Name,
       Location, Nationality,
       Ent-type, alias-list, np-list).
```

The reference between two entities is resolved by looking at the similarity between their names and/or their noun phrases. Since companies are often referred by their nationality or location, the Location and Nationality slot fillers in the entity frame also contribute to the reference resolution. Some special noun phrases which refer to some particular role of a tie-up (*the newly formed venture*in particular) are also recognized and resolved. For example, a phrase which refers to the child entity, such as *'the new company'* or *'the venture'*, will be recognized and merged with the child of the tie-up event in focus. A stack of entities found in the text is maintained.

Definite noun phrases can only be used for local reference. So they can only be used to refer to entities involved in the tie-up event which is in focus. On the contrary, names can be used for both local and global reference, so they can refer to any entity referred to before in the text.

When a reference relation between two entities is resolved they are merged to create one single entity which contains all the information about that particular entity.

Since a tie-up is generally referenced by an entire sentence rather than a single noun phrase, the reference of tie-up events is handled by resolving the reference between its participants and some other information mentioned about the event. Other heuristics are also applied. These mostly block the overapplication of merging. For example, two tie-ups cannot be merged if their dates are different; similarly, entities with different locations will not be merged. There are currently two types of text structures which are considered. In the first type, one tie-up-event is in focus until the next one is mentioned and after the new one is mentioned the old one will not be mentioned again. In the second type, a list of tie-up-events are mentioned shortly in one paragraph, and more details of each event are given sequentially later. Finally, when the reference between two tie-ups is resolved they will also be merged to form a single tie-up event. The final result is a set of new frames which are linked in such a way as to reduce the requirement on the final stage of maintaining pointers to the various objects.

With the exception of the use of definite articles —an obvious cross-linguistic difference between the languages studied— the reference resolution process for Japanese is identical to English. The resolved entities, entity-relation, and tie-up for a typical text are shown below.

```
final_entity(2,
 [f(name,['大','和','証','券'],'UNSPEC'),
  f(entity_type,'COMPANY','UNSPEC'),
  f(industry_product,'(63 "財形年金")',wj),
  f(time,[after,'8501'],wj),
  f(entity_relationship,1,inf),
  f(entity_relationship,3,inf)]).

final_entity(9,
 [f(name,['東','京','海','上',火,災,'保',険],
    'UNSPEC'),
  f(entity_type,'COMPANY','UNSPEC'),
  f(name,['東','京','海','上'],'UNSPEC'),
  f(entity_relationship,1,inf),
  f(entity_relationship,3,inf)]).

final_rel(1,[9,2],'UNSPEC','PARTNER','UNSPEC').

final_tie_up(1,[9,2],'UNSPEC','UNSPEC',
    'UNSPEC',existing,'UNSPEC',1,'UNSPEC').
```

The Japanese system uses character-based rules for identifying aliases. The followings are examples of rules used in the system.

- First two characters used for an alias. '日立' (Hitachi) for '日立製作所' (Hitachi Manufacturing).
- First and third characters used. '日航' (Nikkou) for '日本航空' (Nihonnkoukuu or Japan Airlines).
- First and last characters for an alias of a foreign company name. 'ア社' (A sha or A Co) for 'アプライド・マテリアル社' (Applied Material Co).
- The system has a knowledge base for difficult aliases. 'ＪＡＬ' for '日本航空' (Japan Airlines) and 'ＧＥ' for 'ジェネラル・エレクトリッ ク' (General Electric).

**Template Formatting** The final stage generates sequence numbers and incorporates document numbers into the labels. It also eliminates objects which are completely empty. The final output from the English system example text, #0592, is shown below.

```
<TEMPLATE-0592-1> :=
   DOC NR: 0592
   DOC DATE: 241189
   DOCUMENT SOURCE: "Jiji Press Ltd."
   CONTENT: <TIE_UP_RELATIONSHIP-0592-1>
<TIE_UP_RELATIONSHIP-0592-1> :=
   TIE-UP STATUS: existing
   ENTITY: <ENTITY-0592-3>
   JOINT VENTURE CO: <ENTITY-0592-1>
   OWNERSHIP: <OWNERSHIP-0592-1>
<ENTITY-0592-1> :=
   NAME: BRIDGESTONE SPORTS TAIWAN CO
   ALIASES: "BRIDGESTONE SPORTS"
   TYPE: COMPANY
   ENTITY RELATIONSHIP:
<ENTITY_RELATIONSHIP-0592-1>
<ENTITY-0592-3> :=
   NAME: BRIDGESTONE SPORTS CO
   ALIASES: "BRIDGESTONE SPORTS"
   TYPE: COMPANY
   ENTITY RELATIONSHIP:
<ENTITY_RELATIONSHIP-0592-1>
<ENTITY_RELATIONSHIP-0592-1> :=
   ENTITY1: <ENTITY-0592-3>
   ENTITY2: <ENTITY-0592-1>
   REL OF ENTITY2 TO ENTITY1: CHILD
   STATUS: CURRENT
<OWNERSHIP-0592-1> :=
   OWNED: <ENTITY-0592-1>
   TOTAL-CAPITALIZATION: 20000000 TWD
   OWNERSHIP-%: (<ENTITY-0592-3> 75 )
```

## 1.5. Hardware and Software Requirements

**Hardware** The system runs on Sun 4 Workstations. It should run on any Unix machine with the appropriate compilers and has in fact been ported onto an IBM RS6000 system.

**Software**

1. Operating System
   The system runs under UNIX. Currently we are using SunOS Release 4.1.

2. Segmentation programs
   POST (BBN) : 24 Megabytes
   JUMAN (KYOTO/MCC version) : 8 Megabytes

3. Programming languages
   Quintus Prolog : Release 3.1.1, requires 64 Megabytes of disk space.
   C
   CMU Common Lisp : 16 Megabytes of memory and 25 Megabytes of disk space are recommended.

4. Unix tools
   flex/lex

5. Size of the data and programs

   English Total 103 Megabytes
   Data 16 Megabytes, Code 87 Megabytes

   Japanese 49 Megabytes
   Data 0.7 Megabytes, Code 48 Megabytes

## 1.6. Speed/Throughput Statistics

On average, the time for the English systems to process one article is 3 minutes. The Japanese systems are much faster, taking about 40 seconds per article.

## 1.7. Key Innovations of Final System

The methods used in the Diderot system have not changed significantly since the original system was assembled for the MUC-4 terrorist message evaluation. Our conviction has always been that simple, easily configurable, modular methods were the only approach which would work in the short term on general text. Four aspects of the system have proven to be key to its operation. These are – finite state tagging methods, semantic partial parsing, domain and language specific reference resolution and statistical judgement of relevance.

**Finite State Tagging Methods** These are an essential component of our extraction system. They allow a text to be marked up with semantic classes of all the objects mentioned in it by the use of patterns and data base files.

This component is language specific and to some extent domain specific. It would seem likely that as more extraction systems are built a growing number of recognizers will become available. For micro electronics we developed specific recognizers for equipment and device names.

We also tested the performance of our organization and human name recognizers by scoring them automatically against human tagged text. This allowed us to enhance the performance of the taggers independent of the rest of the system. Development of specific evaluation methods for components is time consuming and expensive, but it has enormous paybacks in terms of measuring the performance of specific components. (The scoring software and data is available to members of the Consortium for Lexical Research, as it much other data and software developed by Tipster contractors. Mail lexical@nmsu.edu for further information.)

**Semantic Partial Parsing** The parser has two levels of operation. The first is a set of rules for identifying appropriate semantic objects in a text. The second is a lexical pattern driven parse which identifies the roles of the objects in a specific sentence. These two operate together to produce frames closely related to the final semantics of a template.

The approach bypasses the normal two stage approach of parsing to a tree structure and then applying inference mechanisms to derive the final logical form for the sentence.

The recognition of objects uses two lists of allowable and required semantic types for each object. Thus a location is allowable as part of an organization semantic object, but either an organization name or an organization noun phrase must be found to satisfy the semantic constraints for an organization. These constraints are specified in a declarative form. It is this level of the parser which recognizes conjunctions and lists of objects. These are nested according to a set of precedence rules and the resulting tree is unwound to produce lists for each object identified for the parse. Thus the pattern `<entity> manufacture <product>` will recognize a list of organizations in the subject position and one or more products in the object.

The hand development of patterns for the parser is relatively simple as there is a clear mapping to the final template. A very small number of frames were used to represent these template semantic structures. The definition of these frames was the same for Japanese and English.

**Reference Resolution and Domain Independence**
The task of reference resolution module in Diderot is to sort the partially filled frames produced by the parser from single sentences in the text and to search for coreferential frames and merge them. Frames are used to represent entities (e.g. companies and persons) as well as events (e.g. tie-ups and relations). Frames are defined recursively such that some frames might have other frames to fill their slots. Frames contain not only the information that needs to be finally extracted from the text but also other information (includes syntactic information) that will help to resolve the reference (e.g. noun phrases). The resolution program consists of the following parts:

1. a set of conditions such that if two frames meet a condition then they are considered to be coreferential.

2. a bottom-up syntax driven algorithm to find all the coreferential frames and merge them into a single frame.

3. methods on how to merge two coreferential frames.

The coreferential conditions can be categorized into syntactic constraints and semantic constraints. The syntactic constraints are harder to specify as declarative conditions and they are coded as procedures that guide the search for coreferential frames. On the other hand, these constraints are domain independent. Semantic constraints are mostly domain dependent and they are specified for each type of frame. Since different syntactic constraints suggest different search patterns and put different requirements on the semantic constraints, the semantic constraints associated with different syntactic constraints may also be different.

The recursively defined frames suggest a frame hierarchy. Our resolution algorithm works from the lowest level frames upwards. At each level, all the search schemes suggested by different applicable syntactic constraints are tried for each frame. If the associated semantic constraints are also satisfied, a coreferential pair is found. Finally, the coreferential frames get merged into one single frame. Since the merge of higher level frames may cause lower level frames to be merged, the merge process is recursive. Here a set of contradiction conditions that resist two frames being merged are used.

The domain independent parts of our reference resolution module are the resolution algorithm and syntactic constraints. The domain dependent parts are semantic constraints, merge methods and contradiction conditions. Trying to make semantic constraints domain independent, we believe, is very difficult if not impossible. For instance the set of conditions that indicate two company frames (such as the ones for name or aliases) are coreferential are very different from that for equipment frames. Besides, unless we have a semantic interpretation module that is intelligent and rich enough, it is impossible to have a domain independent mechanism that can correctly interpret, say, definite descriptions (consider possessive modifiers for company and equipment). To make things even worse, it is also very difficult to specify some of these conditions declaratively. A good example is the company names and device names where different naming conventions force us to write different procedures to manipulate name strings in order to find out alias relations.

So, we believe the best solution to make adapting to a new domain easier is a yacc/lex type of precompiler. Here, to port the system to a new domain, we only need to provide domain dependent conditions and merge methods for each frame type and/or each syntactic constraint. We can write our own predicates/procedures or use ones provided in a system library to specify the conditions and the methods. The precompiler will combine them together with the resolution algorithm and syntactic constraints to produce a reference resolution program for that domain.

**Statistical Relevance Judgement** We have continued to work on a procedure for detecting document types in any language. The system requires training texts for the types of documents to be classified. The method is developed on a sound statistical basis using probabilistic models of word occurrence [?]. This may operate on letter grams of appropriate size or on actual words of the language being targeted and develops optimal detection algorithms from automatically generated "word" lists.

For the Japanese micro-electronics system, texts were filtered to decide whether or not they were relevant to the domain. The decision was based on whether an incoming document "resembled" a set of documents judged "relevant" by human analysts (i.e. human analysts produced a corresponding non-empty template for the document). We varied the meaning of "resemble" in a series of statistical experiments using the frequencies of words, bigrams, trigrams and four grams found in the document to be classified, and found to be good "distinguishing" words/grams in the texts which were judged relevant by humans. All experiments used a multinomial model for the problem and maximum likelihood ratio test for the decision. Similar experiments were performed on the English micro-electronics texts. The entire set of documents judged relevant by humans was used for training since it was felt that the number of texts of this type which were available was relatively small, and for this same reason, the decisions in both systems are based on words rather than grams at this time.

## 2. Original Project Goals

We list our original project goals and comment briefly on how far our present effort has gone in achieving these goals and how they have been modified based on the realities of the Tipster information extraction task.

1. *language modularity: allowing the addition of new languages with a minimum of effort through use of a limited interlingual representation for lexical and domain knowledge;*

   Since the English and Japanese systems use the same system architecture in both domains and the same internal representation is used in English and Japanese system, the conversion from English system to corresponding Japanese system was relatively easy. The English Joint Venture system was converted to give the Japanese JV system and the English Micro-Electronic system was converted to give the Japanese ME system by one native speaker of Japanese. The differences between English and Japanese systems are as follows:

   - Data for tagging. Company, human, title, and place names and time expressions are language specific.
   - Patterns for GLS cospecification. There is a set of Japanese verbs for indicating various kinds of tie-ups such as import tie-up, sales tie-up, and business tie-up. Besides, the majority of the tie-ups in Japanese Joint-Venture articles involve only two parent companies and there is no mention of the JV company. Thus this fact is relfected on the cospec patterns of these verbs.
   - Patterns for recognizing company name aliases. As explained above, the Japanese system uses character-based and language-specific rules for recognizing aliases.

2. *acquisition of benefits of scale through the addition of lexical information automatically from existing machine-readable dictionaries;*

   We used the Longman Dictionary of Contemporary English to generate the initial verb patterns using verb subcategorization information, which is supplied in the dictionary, supplemented by example definitions which sometimes preferred subject and object information in the form of bracketed example subject and object types. This was then extended by finding additional pattern information in the Wall Street Journal Corpus. The dictionary, however, did not prove rich enough to provide all the possible ways of expressing information found in newspaper text, For example *team up with, join forces*, and so on. These have been added using patterns for equivalent senses found in the dictionary.

   Additionally the dictionary was used to generate semantic classes of nouns, for example all the words like *factory* which represent an industrial site. This was done for several classes of noun. The other source of this type of information was the keys provided for training data.

3. *the use of well-motivated Lexical Structures (LS's) to capture the presuppositional and anaphoric aspects of texts structures, essential for successful extraction;*

   The lexical structures used in Diderot specify possible patterns occurring in the text and the types of appropriate objects found at specific locations in the patterns. By allowing noun phrases with appropriate heads to satisfy these constraints the lexical structures allow the generation of partially completed *frames* which can then be processed by the reference resolution module.

4. *the initial seeding of structures automatically by the techniques of (2) above, and the tuning of the LS's against corpora for particular languages (e.g. Japanese);*

   Tuning of lexical structures against the corpus has been a major effort in our project. This has not produced the results we had hoped for. This may be partially due to the lack of specificity of the corpus we were using. In addition some of the methods developed depended on having corpora tagged with reasonably accurate semantic information. Our semantic tagging module has increased in accuracy during the course of the project. During the initial development phase it was probably not of sufficient quality to support the corpus development effort.

5. *the use of strong semantic resolution techniques (based on Wilks' Preference Semantics [?]) for the resolution of lexical ambiguity, and the imposition of appropriate structure on real (i.e. potentially ill-formed, multi-sentence) input text;*

   Semantic constraints are applied to the structures which occupy the various fields in the cospecification pattern. These impose necessary conditions on the information gathered for each field. This proved sufficient to disambiguate the uses of the forms found in both domains.

6. *given that full parsing of very large-scale text samples is out of the question in the current state of the art, in the sense of parsing every sentence of a large text into a formal structure of any depth and content, we propose a set of alternative partial parsers and segmenters, all parsing to a canonical interlingual representation for selected sentences;*

   This statement is almost a thumbnail sketch of our current system. Our system essentially operates with patterns at a variety of levels. These produce a very specific domain dependent canonical representation containing the essential information required for the construction of a set of templates.

7. *we shall define a set of "minimalist AI techniques" to connect inferentially the information carried by the slot-names of the TIPSTER templates: among these will be Finite State Acceptor demons that know about, e.g., the structures of dates, places, person names in English and Japanese and have access to large publicly-available word lists;*

   Our system is dependent on a multiplicity of finite state machines which recognize the basic building blocks of a template. These processes often rely on large lists of terms for the specific class of item being recognized. In other case they rely on patterns derived by using corpus analysis tools such as Keyword in Context (KWIC) indexes (for example for equipment names).

8. *although statistical techniques used alone and unaided for traditionally AI tasks give poor results and seem to offer no clear path to optimization, the use of some such techniques is now firmly established in conjunction with symbolic techniques and we shall propose statistical techniques for gathering what we shall refer to as the "true lexicon" of the texts, and using these to locate relevant "text points" for detailed analysis;*

   Our statistical techniques have been used in a variety of ways during the development of Diderot. In the original MUC-4 system they were used to identify specific paragraphs, for Tipster microelectronics they marked relevant texts. These methods have already been discussed. In addition the methods allow us to identify important vocabulary for a domain. This has been less important for the well defined domains we have worked on, but would prove useful to an analyst moving into a new domain who already had a collection of relevant and irrelevant texts.

9. *closely connected to (7) will be Metallel procedures that determine standard metonymic and hierarchical relations between text items and other items available to the domain knowledge base (e.g. Moscow often should be replaced by Soviet Government). Like the procedures of (7) it has access to an automatically-generated tangled genus hierarchy from the methodology of (2).*

   A study of the metaphor and metonymy occurring in the joint venture domain was made at an early stage in the project. Various classes of metaphors were identified. However, the large majority of these proved to occur in standard ways and could be classified as *dead* metaphors. The most appropriate approach seemed to be to code these explicitly into the lexicons used by the system.

## 2.1. Machine Assisted Human Information Extraction

In addition to work on the automatic extraction of information from documents, CRL was also involved in the human side of the Tipster project. To prepare the Tipster data, human analysts performed the information extraction task on over five thousand documents. CRL created and maintained software tools to aid in this task for each of the domains and languages. These window-based tools allow human analysts to build the key templates by selecting pieces of the original text, or picking standardized field information from menus. These tools were used by all of the analysts and all of the sites performing this task.

Based on this experience with the human extraction task, and our own automatic extraction system, our vision for the future is one of integrated extraction components which aid *human in the loop* analysis. For many applications the current information extraction systems are insufficiently accurate and have too long a development time. Even in cases where the technology is adequate there is still a need for some completed keys both to 'prime the pump' and to allow objective testing of system performance. In both cases this means a human analyst carrying out the template filling task.

We have developed an initial version of a system which supports integrated machine assisted human information extraction, with fills for fields being both suggested and converted to standard forms by automatic extraction modules. This system, *Tabula Rasa*, is an interactive design tool and interface code generator which allows an analyst to define a new domain and to produce a matching machine assisted information extraction tool in minutes. This is intended to allow a more rapid development of the definition of the extraction task and an integration of automatic extraction techniques in a tool used by human analysts.

With Tabula Rasa an analyst can define windows for each data object which is to be extracted from the text. The fields in these objects are created and labeled by the analyst and a definition of the type of information they can hold is specified. Other attributes can also be set, for example if it is a required or optional fill. Some fields can be set-up with automatic extraction capabilities. For example, a field can be specified as a 'name' field and if the texts are preprocessed by the *Diderot* system, a list of automatically extracted names are presented as candidate fill values. The structured data specification is controlled with an interactive graphical user interface and is used to produce a tool which can be used immediately to test if the output specified is appropriate. A definition of the data structure developed (in standard BNF form), and a set of texts describing specific fields and objects in the template are automatically produced. These can be used as the basis of both on-line and paper documentation and we intend to build a simple generator which will create the first draft of this documentation automatically.

Tabula Rasa is an attempt to reduce two of the major bottlenecks of information extraction; the definitions of the text extraction task and the production of tools intergrating automatic extraction to aid the human analyst in the production of structured data. We intend to investigate how successful Tabula Rasa is by researching its actual use by analysts. This investigation will focus on the usefulness of automatically extracted data for *human in the loop* analysis systems. Future versions will embody ways of integrating well tested improvements in automatic techniques that will aid the analyst as suggested by the actual use of the tool.

## 3. Evolution of system over two years

The Diderot system was developed from scratch for the Tipster information extraction project. A diagram showing the chronology of the system can be found at the end of this paper.

The first version of the system was developed in five months and was evaluated in the 4th Message Understanding Conference (MUC-4) where it extracted information from 200 texts on South American terrorism. At this point the system depended very heavily on statistical recognition of relevant sections of text and on the ability to recognize semantically significant phrases (e.g. a car bomb) and proper names. Much of this information was derived from the keys.

The next version of the system used a semantically based parser to structure the information found in relevant sentences in the text. The parsing program was derived automatically from semantic patterns. For English these were derived from the Longman Dictionary of Contemporary English, augmented by corpus information and these were then hand translated to equivalent Japanese patterns. The Japanese patterns were confirmed using a phrasal concordance tool. A simple reference resolving module was also written. The system contained large lists of company names and human names derived from a variety of online sources. This system handled a subset of the joint venture template definition and was evaluated at twelve months into the project.

Attention was then focused on the micro-electronics domain. Much of the semantic information here was derived from the extraction rules for the domain. A single phrase in micro-electronics can contribute to several different parts of the template, to allow for this a new semantic unit the *factoid* was produced by the parser. This produced multiple copies of a piece of text, each marked with a key showing how the copy should be routed and processed in subsequent stages of processing. This routing was performed by a new processing module, which transformed the output from the parser. The statistical based recognition of text relevance was used for microelectronics only, as a much higher percentage of articles in the corpus are irrelevant. This system was evaluated at 18 months.

Finally the improvements from micro-electronics were fed back to the joint venture system. An improved semantic unit recognizer was added to the parser. This handles conjunctions of names, possessives and bracketing. An information retrieval style interface to the Standard Industrial Classification Manual was linked into the English system. The reference resolving mechanism was extended to handle a richer set of phenomenon (e.g. plural references). This, current, version was evaluated at 24 months.

## 4. Accomplishments: What worked and what failed, and why

The Tipster task is an extremely complex one in terms of the number of components involved and the volume of data needed to support the task. It is extremely difficult to point at individual components of the system and say this works, and this does not. Throughout the processing each component is dependent on the performance of previous stages.

Our main accomplishment was in the construction of five working extraction systems over the two years of the project. We are particularly pleased with the performance of our two Japanese systems.

For the English systems we adhered to our plan of attempting to automate as much as possible the development of the system, in particular the lexicon and associated semantic patterns. This work is going to continue, but at the moment the performance of a system developed in this manner is unlikely to match one which depends on careful hand tuning.

Our name and object recognizing software is a stand alone component and has now reached levels of precision and recall of 75% for both languages.

The automatic generation of our parser from the GLS lexical entries is also a useful method developed in the system. However, we need more sophisticated debugging techniques to enable us to track parse failures and errors.

We feel that we have explored the problems involved in implementing a linguistic theory (Pustejovsky's Generative Lexical Semantics) in an operational system. This has lead to additions to the theory to support the specifics of extraction and also to ignoring interesting aspects which did not support the task. In particular we have failed to achieve the *generative* aspect of the theory which allows the lexical attributes of nouns to be incorporated in the more general sense of a verb. We have relied on a much simpler semantic typing for proper nouns and noun phrases.

Our other main research theme was to develop lexical entries from corpora. This proved to be a very time consuming process and based as it is in a kind of averaging may not produce data specific enough for the task. An analyst with some knowledge of how the system operates could write patterns for actual sentences that fill templates more specifically than those we generated for our English systems. The contrast here is clear between our English and Japanese systems.

We have advocated partial parsing and regular expression based pattern matching methods since the project began. This approach certainly appears to be the most appropriate for the information extraction task.

## 5. Evaluation Summary

### 5.1. Official Tipster/MUC Scores

The summary scores for each system are given in the appendix to this paper. Graphs are also given showing the improvement of the final systems compared to those at the eighteen month evaluation. The systems were all designed to attempt to fill all the possible slots in the template. For the joint venture domain, in particular, where many slots occurred only a few times in the training keys this made developing accurate systems very much harder.

It is also clear from our experience of system development that the interaction between the parts of a system is complex and that modifications at one level can often, due to bugs or changes in the representation, lead to a significant drop in performance. The ideal approach would seem to be to iteratively test small changes on a relatively stable system, by scoring performance against a series of test sets. This is the approach adopted for both our Japanese systems. The English systems received no detailed hand tuning at this level, although the micro electronics was improved by producing appropriate lexical entries for all short texts in the test collection, which originally had no template output produced by the system.

**English Joint Venture** This system was the most reliant on automatic development and least on human tuning. The recall in particular was very low 24%, with a precision of 51% for the *all objects* measure. In particular some of the simpler slots *entity location* and *nationality*, should have been subjected to much stronger inspection.A large number of fills were generated for these, but with very low precision. Other slots such as the product service code, which produced 818 entries, were much harder to fill correctly depending as they did on a correct analysis of the relevant sentences, a correct coreference match to the appropriate entities and finally the correct identification of the product string and SIC code.

Our performance lies somewhere in the middle of the MUC-5 systems and is the lowest of the Tipster systems.

**English Micro-electronics** This system had a similar precision to our English joint venture system, but had higher recall. This was largely due to a last minute attempt to produce a greater coverage by hand coding lexical entries. There is a great deal of variation in the accuracy of the recognizers for the variety of fields found in EME. Further tuning would focus first on this aspect of the system. That is until *etchants, materials, equipment names* can be identified accurately there is no possibility of extracting this information in the present system. The other significant problem we faced was the roles of the organizations mentioned in the text. Our precision for these was far lower (19% - 34%) than the precision we obtained for the process object(58%). The actual identification of appropriate entities was much higher (60%) and for entity name recognition (54%).

**Japanese Joint Venture** Our performance in Japanese is significantly better than English, with the CRL system lying in second place behind the extremely high performing GE system. The differences between the two systems are that the GE system has better recall with high precision. The CRL system has lower recall and slightly higher precision. In fact, in terms of the precision, the CRL system has the best score. The error rate and under generation for the GE system is lower than that of CRL system. Thus the GE system has shown good recall with good precision, which means lower scores in error rate and under generation.

**Japanese Micro-electronics** Again, the GE system is the top performer with the CRL system coming second. In JME, GE's system has lower precision than its JV system. It seems that recall was emphasized in GE's ME system. On the other hand, CRL's ME system focused on precision. The CRL system has the highest precision. The GE system has lower scores in error rates and under generation, and the CRL system has lower scores in over generation and substitution.

### 5.2. Explanation and Interpretation of Results

The scores for Japanese, using an identical architecture, but with much more intensive human tuning, are much higher. We feel the huge difference between performance in Japanese and English is principally due to one person being dedicated for Japanese to running and tuning the system. All other personnel were working on particular components to be used first in the English and then in the Japanese system and no one person was repeatedly testing the operation of the English System. Another difference might be due to the focus of effort on automatic and semi-automatic pattern generation for the English systems, a process which was not attempted for Japanese development.

## 6. Conclusions

We have learned a great deal over the past two years, partly through the many mistakes we have made. The project has depended a great deal on the skill and care of the people working on it to ensure consistency in our data and code. Given the large number of knowledge bases in our system this is an onerous task and one task needed for the future is a system which allows this knowledge to be integrated and held in one central data-base, where consistency can be maintained. The second is to develop an easily configurable and portable reference resolution engine.

There are no major differences in the structure of the English and Japanese systems. It would seem that a critical part of achieving high precision and recall is to have at least one person with a reasonable knowledge of the whole system to carry out repeated test/improve cycles.

The current system is robust and provides a good starting point for the application of more sophisticated techniques, some of them simply refined versions of the current architecture. Given appropriate data it should be possible to produce a similar system for a different domain in a matter of months. Many parts of the system are portable in particular the semantic tagging mechanisms, the statistical filtering component. Dates, companies and people - all of which occur in many kinds of text - we now handle with good levels of accuracy.

## 7. Acknowledgements

The system described here has been funded by DARPA under contract number MDA904-91-C-9328.

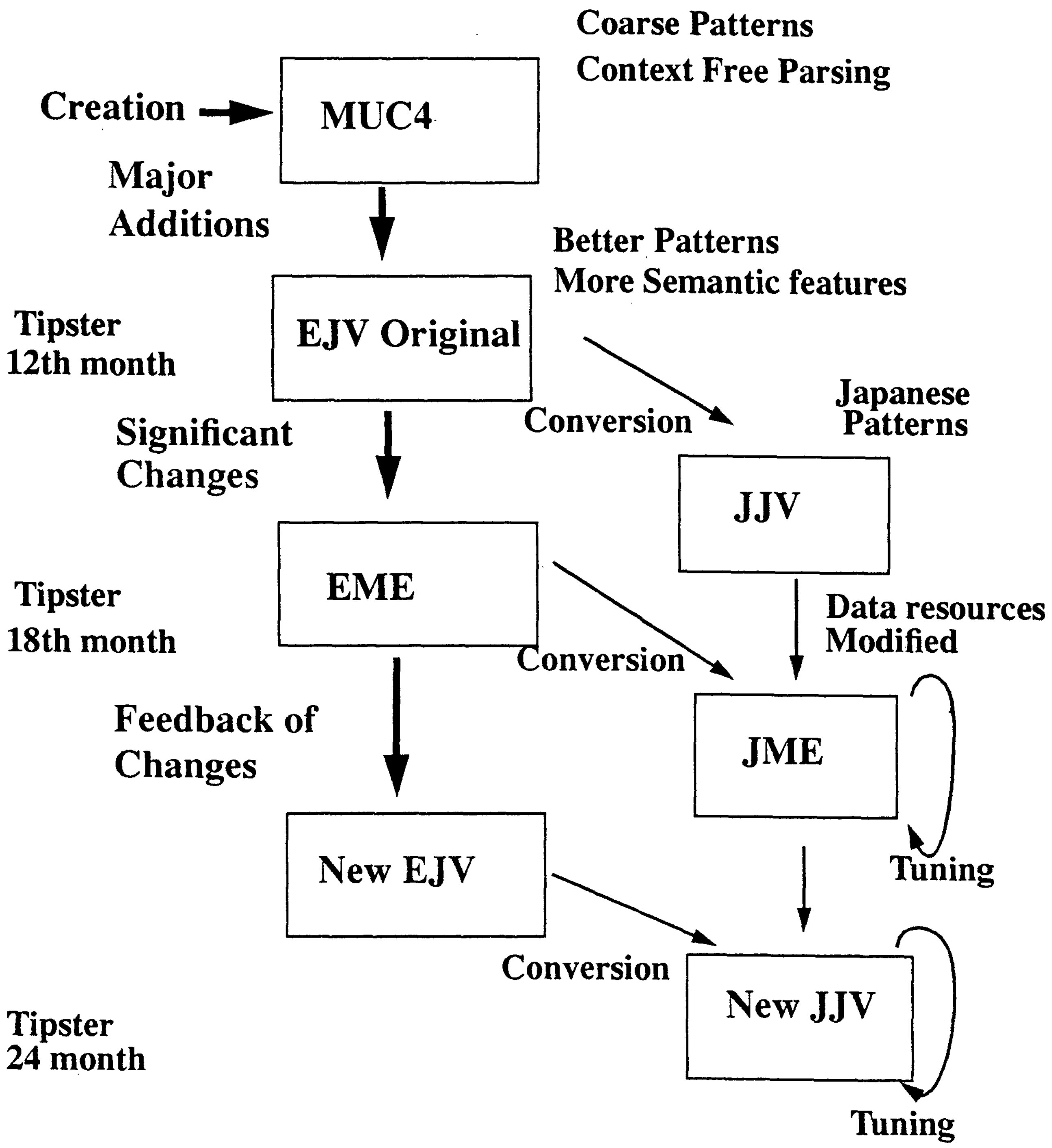
We would like to express our thanks to our colleagues at BBN who have shared their part of speech tagger (POST) with us. Thanks also to Kyoto University for allowing us to use the JUMAN segementor and part of speech tagger.

Diderot is a team effort and is a result of the work of many people. The following colleagues at CRL and Brandeis have contributed time, ideas, programming ability and enthusiasm to the development of the Diderot system; Paul Buitellar, Federica Busa, Peter Dilworth, Steve Helmreich and Fang Lin.

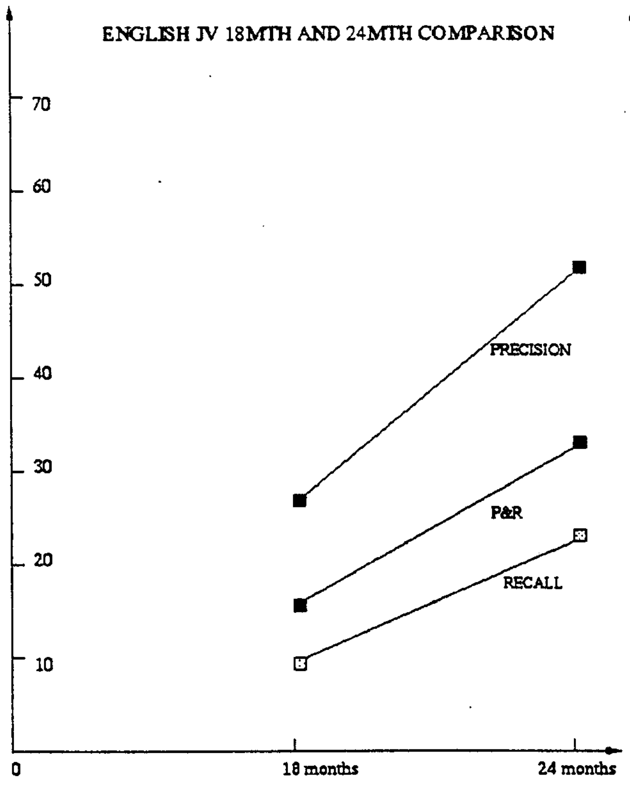
## References

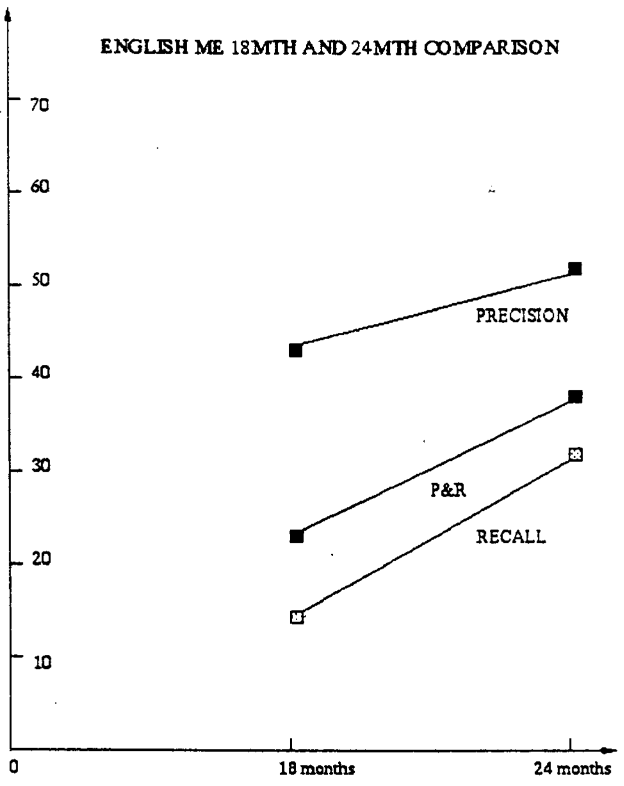
1. DARPA. *Proceedings of the Third Message Understanding Conference (MUC-3)*, San Mateo, CA, 1991. Morgan Kaufmann.
2. DARPA. *Proceedings of the Fourth Message Understanding Conference (MUC-4)*, San Mateo, CA, 1992. Morgan Kaufmann.
3. Cowie, J., Guthrie, L., Wakao, T., Jin, W., Pustejovsky, J. and Waterman, S., The Diderot Information Extraction System. In *Proceedings of the First Conference of the Pacific Association for Computational Linguistics (PACLING93)*, Vancouver, Canada, 1993.
4. Grishman, R., and Sterling, J., Acquisition of selectional patterns. In *Proceedings of the 14th International Conference on Computational Linguistics (COLING92)*, Nantes, France, 1992.
5. Guthrie, L., Bruce, R., Stein, G.C., and Weng, F., Development of an application independent lexicon: Lexbase. Technical Report MCCS-92-247, Computing Research Laboratory, New Mexico State University, 1992.
6. Guthrie, L., and Walker, E., Some comments on classification by machine. Technical Report MCCS-92-935, Computing Research Laboratory, New Mexico State University, 1992.
7. Lehnert, W., and Sundheim, B., An evaluation of text analysis technologies. *AI Magazine*, 12(3):81–94, 1991.
8. Proctor, P., editor. *Longman Dictionary of Contemporary English.* Longman, Harlow, 1978.
9. Pustejovsky, J., The generative lexicon. *Computational Linguistics*, 17(4), 1991.
10. Pustejovsky, J., The acquisition of lexical semantic knowledge from large corpora. In *Proceedings of the DARPA Spoken and Written Language Workshop.* Morgan Kaufmann, 1992.
11. Pustejovsky, J., Bergler, S., and Anick, P., Lexical semantic techniques for corpus analysis. *Computational Linguistics*, 1993.
12. Pustejovsky, J. and Boguraev, B., Lexical knowledge representation and natural language processing. *Artificial Intelligence*, 1993.
13. Pustejovsky, J., *The Generative Lexicon: A Computational Theory of Lexical Semantics* MIT Press, Cambridge, MA, 1994.
14. Schabes, Y. and Shieber, S., An alternative conception of tree-adjoining derivation. In *Proceedings of 30th Annual Meeting of the Association for Computational Linguistics*, 1992.
15. Wilks, Y., A Preferential Pattern-Seeking Semantics for Natural Language Inference. *Artificial Intelligence*, 1975.

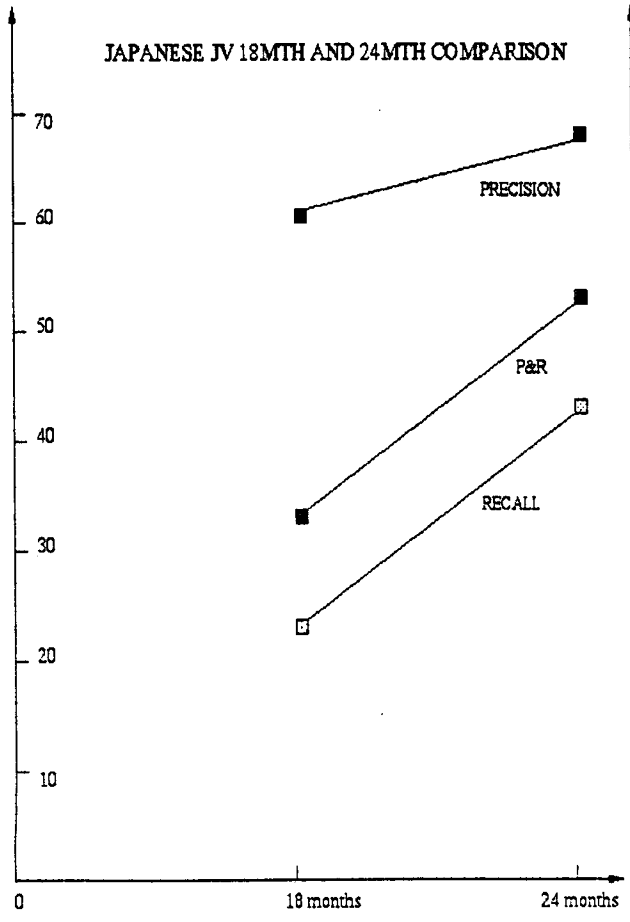
## System Development History

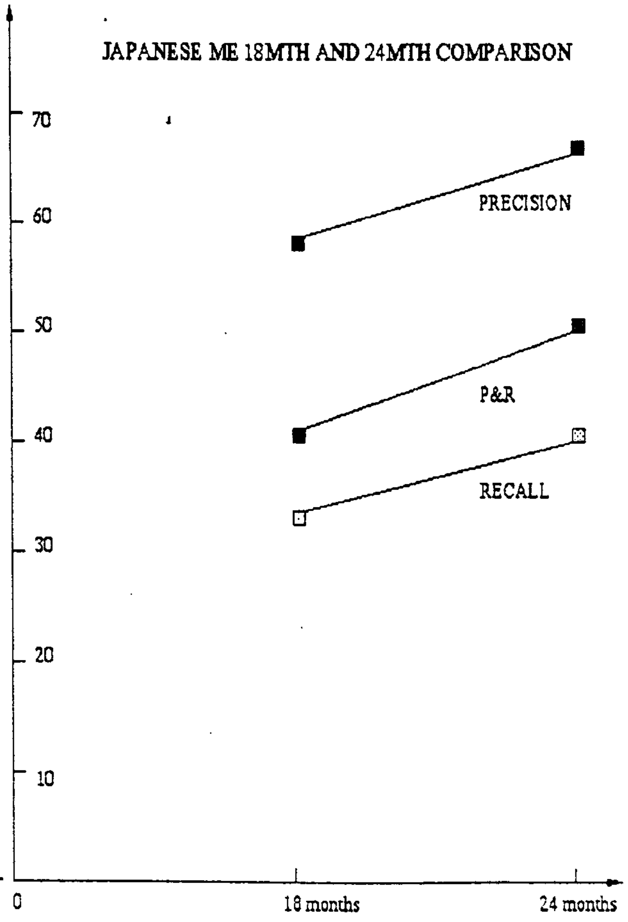


# Progress since 18 month workshop









# Summary of Error-based Scores

## JAPANESE MICRO

| | ERR | UND | OVG | SUB | Min | Max |
|---|---|---|---|---|---|---|
| 18-Month | 72 | 60 | 28 | 18 | .74 | .80 |
| 24-Month | 65 | 54 | 24 | 12 | .69 | .73 |

## JAPANESE JV

| | ERR | UND | OVG | SUB | Min | Max |
|---|---|---|---|---|---|---|
| 18-Month | 79 | 71 | 22 | 22 | .86 | .86 |
| 24-Month | 63 | 51 | 23 | 12 | .70 | .72 |

## ENGLISH MICRO

| | ERR | UND | OVG | SUB | Min | Max |
|---|---|---|---|---|---|---|
| 18-Month | 86 | 76 | 33 | 37 | .87 | .93 |
| 24-Month | 74 | 60 | 33 | 24 | .80 | .84 |

## ENGLISH JV

| | ERR | UND | OVG | SUB | Min | Max |
|---|---|---|---|---|---|---|
| 18-Month | 91 | 76 | 40 | 56 | 1.06 | 1.08 |
| 24-Month | 79 | 67 | 28 | 28 | 0.89 | 0.91 |

# Summary of Recall/Precision-based Scores

## JAPANESE MICRO

| | TF(R/P) | REC | PRE | P & R |
|---|---|---|---|---|
| 18 - Month | 73/83 | 32 | 59 | 41.99 |
| 24 - Month | 84/90 | 40 | 66 | 50.37 |

## JAPANESE JV

| | TF(R/P) | REC | PRE | P & R |
|---|---|---|---|---|
| 18 - Month | 82/99 | 26 | 61 | 32.8 |
| 24 - Month | 88/98 | 42 | 67 | 52.1 |

## ENGLISH MICRO

| | TF(R/P) | REC | PRE | P & R |
|---|---|---|---|---|
| 18 - Month | 77/76 | 15 | 42 | 22.28 |
| 24 - Month | 78/83 | 31 | 51 | 38.49 |

## ENGLISH JV

| | TF(R/P) | REC | PRE | P & R |
|---|---|---|---|---|
| 18 - Month | 67/86 | 10 | 26 | 15.10 |
| 24 - Month | 76/92 | 24 | 51 | 32.64 |