AUTOMATING REVIEW OF FORMS FOR INTERNATIONAL TRADE TRANSACTIONS: A NATURAL LANGUAGE PROCESSING APPROACH

Vasant Dhar
and
Padmanabhan Ranganathan

March 1986

Center for Research on Information Systems
Computer Applications and Information Systems
Graduate School of Business Administration
New York University

Working Paper Series

CRIS #117

GBA #86-20

Appears in the **Proceedings of the Third International Conference on Office Information Systems**, Providence, Rhode Island, October 1986.

ABSTRACT

A major challenge in Office Automation is one of automating routine jobs that involve large-scale processing of ill-formed natural language data. Such data are often present in documents such as forms where it is necessary and/or practical to allow latitude in how the forms may be filled. In this paper, we describe a computational model designed to process free-form textual data in application forms for Letters of Credit (LC), which represent a common vehicle for initiating international trade transactions. The model is based on a variation of the case-frame or thematic-role frame instantiation methods. We describe the implementation of the model, report empirical results with real LC applications, and indicate directions we are currently pursuing to improve its performance.

Keywords: Office Automation, Forms Processing, Natural Language Processing, Artificial Intelligence

Table of Contents

1. INTRODUCTION 2
2. MODEL OF THE LC DOMAIN 3
  2.1. Goals for the Applicant 4
  2.2. Goals for the Bank 4
3. THE LC APPLICATION FORM 5
  3.1. The structured parts of an LC 5
  3.2. Unstructured parts of the LC application 6
4. A MODEL FOR AUTOMATED LC REVIEW 7
  4.1. Motivation behind our approach 8
  4.2. Syntactically-driven approaches 8
  4.3. Towards a more semantically-driven approach 9
    4.3.1. An Object oriented Representation of key features 11
    4.3.2. Parsing Strategies 13
5. RESULTS 15
6. DISCUSSION 17

1. INTRODUCTION

The field of Office Automation (OA) is concerned with the problem of improving office procedures and task performance through the use of technology. Much of the OA work has concentrated on the design of managerial workstations (Barber, 1983; Hammer and Zisman 1979), communications systems (Fikes and Henderson, 1980), time management systems (Goldstein and Roberts, 1977), and systems for integrating generic office *activities* (Zloof, 1982; Ladd and Tsichritzis, 1980; Attardi et.al, 1980). In addition many business *tasks* have also been automated to a large extent. These include transactions processing oriented tasks such as payroll, accounts receivable, and inventory control. Automation of these tasks is primarily based on accounting theory and generally accepted accounting principles -- the role of the theory underlying transactions processing systems has been one of identifying a core vocabulary of terms that occur commonly in business transactions, establishing unambiguous meanings for such terms, and defining clear-cut relationships among them.

A feature of most accounting tasks is that they involve repetitive manipulation of numerical data and/or large amounts of search through data stores, making them prime candidates for automation. In contrast there are many office tasks that, although repetitive, involve processing data that are not easily interpreted by computer. Such tasks involve visual and unstructured textual data that require a basic level of common-sense or human intelligence. Automating these office tasks will require developing formal models for them in terms of specialized vocabularies, although such models are likely to be significantly more complex than accounting models.

This paper deals with one such generic office task in the banking industry that involves processing of application forms for the issuance of Letters of Credit (LC), which represent a common method for conducting most international trade transactions. This task currently requires a significant amount of human input in order to interpret large amounts of free-form text that appear in such forms. This research has involved an empirical investigation of the LC processing problem in a large New York bank with the objective of understanding the meanings of the basic terms involved in LC transactions and the relationships among them. These have been formalized into an ontological model of the LC domain. The pragmatic goal of the research is the development of a natural-language processing system that handles

LCs with the objectives of eliminating human error, increasing efficiency, and resulting in reduced processing of paper documents. A prototype system is currently being tested with real LC applications.

The remainder of this paper is organized as follows. The next section provides a descriptive model of the LC process, introducing the core vocabulary that is used in international transactions. Section 3 describes the LC application form and discusses our approach toward structuring the review of such a form. Section 4 provides a formalization of our model and its implementation. Section 5 summarizes experimental results to date with sample LC applications. We conclude with a discussion on the current limitations of the system and how we are are addressing them.

2. MODEL OF THE LC DOMAIN

International trading introduces a level of complexity into the notion of exchange of goods because of the geographical separation between the buyer and seller. This complexity requires a credible third party, usually a bank, to ensure that the transaction meets certain standards and that the exchange of goods for money meets with the "widely accepted norms" of international trade. The bank acts as an intermediary in the transaction and in effect guarantees the seller (also known as the Beneficiary) payment if the seller meets the stipulations requested by the buyer (the applicant for the Letter of Credit) which are confirmed by the bank.

In effect, there are 3 "actors" involved, as shown in Figure 1. A trade can be broken up chronologically as follows (the numbered steps correspond to the labeled arrows in Figure 1.).

1. A (the applicant) wishes to buy goods from B (the beneficiary). To initiate this request, A sends B a purchase order stipulating quantities, prices, quality and other properties of the merchandise being purchased, and conditions about how the exchange is to be conducted.

2. B requests an assurance from A that payment will be received for the goods before a specified date -- which is soon after the goods have been certified as having been shipped.

3. A wants assurance from B that the purchase order details will be met precisely. In order to be sure about this before the goods are actually shipped, A prepares a list of criteria that B must meet. These are usually certified documents confirming the properties of the merchandise being purchased. In addition, certain special instructions peculiar to the transaction might be specified by A.

4. A applies to C (the actual LC application) to guarantee payment to B if the criteria in (3) were satisfied.

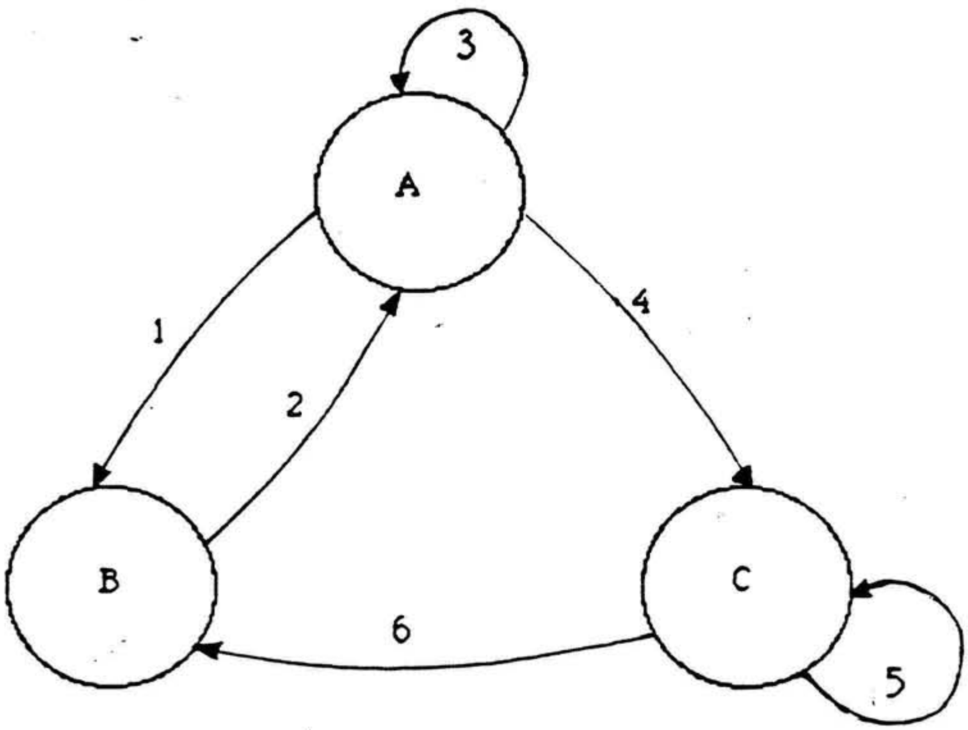

FIG 1. A schematic of the actors and steps (numbered) involved in an international trade transaction. The numbered steps are described in the paper

A. Applicant
B. Beneficiary
C. Bank

5. Bank evaluates the LC application (4 above), making possible modifications to it or rejecting it as unacceptable. Rejection usually occurs if the LC application contains gross inconsistencies of facts, or calls for the bank to perform verifications which it cannot perform without expending excessive resources.

6. C gives an LC to B. This LC is a guarantee that B will be paid if the criteria specified by A and verified by the bank are met.

Steps 3 and 5, involving A and C, are central to the LC process. In order to understand them more fully, it is useful to examine the goals of both the actors. Our objective is to use the knowledge about these goals to automate step 5 -- the evaluation of LC applications.

2.1. Goals for the Applicant

It is in A's best interest to have as much supporting evidence as possible on all aspects of the transaction. At the same time, in requesting this evidence, A is cognizant of the fact that C will only accept terms where the evidence is conveniently verifiable by C. The standard verifiable documents involved in such transactions, which A can ask for are:

1. Airway Bill / Truckway Bill / Ocean Bill of Lading : to ensure acceptable transportation. This is provided by the shipping company.

2. Commercial Invoice, to ensure that the goods requested on the purchase order are the ones actually despatched by B. This is provided by B.

3. Inspection Certificate, to ensure quality of the shipped goods. This is provided by a party stipulated by A, who may be a "responsible person" in B's organization, or a neutral party.

4. Packing List, to facilitate quick identification and forwarding. This is provided by B.

5. Customs Invoice, to ensure legality of export. This is provided by Customs in the sellers country.

6. Other signed statements from B for any additional protection that A may deem necessary. These are usually influenced by the type of good involved and can be provided by various parties.

2.2. Goals for the Bank

The bank has the following two goals:

1. To ensure consistency between the various parts of the LC. For example, the last date for payment to B must not be earlier than the last date for shipment.

2. To ensure that all the documents requested by A are issued by an "appropriate" party and

are verifiable through a "credible" party (such as customs).

Let us describe how these goals are achieved. In order to do this, it is necessary to examine in more detail, the structure of the LC application form. This structure is generic to virtually all banks.

3. THE LC APPLICATION FORM

Forms represent a means for eliciting and recording information in a structured manner (Ladd and Tsichritzis, 1980). Since forms are typically associated with routine jobs which people perform with little enthusiasm, automating of form handling stands a good chance of being well received and reducing human error (Tsichritzis, 1982). In fact, over the last few years, there have been several efforts at building Form Management Systems (Tsichritzis, 1979; Tsichritzis, 1982; Yao et.al, 1984) which are essentially automated "overseers" which ensure that forms are specified, filled, handled, or routed correctly.

However, forms can also contain free-form text information. This text may emphasize the content of the structured part, or may include ancillary information that is not expressible within a rigid predefined structure. A trained reviewer has little problem in dealing with this data because it is interpreted within a limited context -- shaped by the overall *purpose* of the form. The purpose results in certain *expectations* that a reviewer has of the text content.

In the following paragraphs, we first provide a quick review of the structured part of the LC , then discuss the unstructured parts of the form -- which represent the real challenge in terms of modeling and automation.

3.1. The structured parts of an LC

The structured part of the LC application is divided into several standard fields containing structured data that is interpreted unambiguously. Figure 2 shows the structured parts of one such application, with sample entries. It shows the names and addresses of the applicant and the beneficiary (in appropriate fields), total value of goods, and two dates. Reviewing this part of the form involves simple consistency checks among data items in the various fields. For example, consider the entries in the form in fields 5, 8 and 11:

APPLICATION FOR COMMERCIAL LETTER OF CREDIT

1. ADVISING BANK	**2. FOR ACCOUNT OF - APPLICANT**
TAIWAN EXCHANGE BANK HEAD OFFICE, TAIPEI, TAIWAN	**NAME** HERO INDUSTRIES, INC. **ADDRESS** 235 PARK AVENUE **CITY/STATE** NEW YORK, NY **ZIP** 10017
3. IN FAVOR OF- BENEFICIARY	**4. AMOUNT**
NAME TAIPEI EXPORTERS **ADDRESS** 3 SUN YING STREET **CITY** TAIPEI, **COUNTRY** TAIWAN	US $ 26,254.00 **words** US DOLLARS TWENTY-SIX THOUSAND TWO HUNDRED AND FIFTY-FOUR ONLY

5

[x] **PRESENTATION FOR NEGOTIATION ON OR BEFORE** **DATE** AUGUST 14, 1985	[ ] **PRESENTATION AT NEW YORK BANK, NY, ON OR BEFORE** **DATE** ______

Please issue an irrevocable Letter of Credit substantially as set forth and forward same to your correspondent for delivery to the beneficiary by: A

A = Air Mail only
B = Airmail with preliminary brief Telecom
C = Full Details Telecom

6.
Available by beneficiary's drafts at

[x] **Sight on New York Bank, NY, for** 100 **% of invoice value**
[ ] ______ **for** ______ **% of invoice value**

7. TERMS

[ ] **FAS** ______ [x] **FOB** AIRPORT [ ] **C&F** ______

[ ] **CIF** ______ [ ] **C&I** ______

8. Shipment From TAIPEI **To** NEW YORK
Latest JULY 30, 1985

9. Partial Shipment(s) : [x] **Permitted** [ ] **Prohibited**
Transshipment(s) : [ ] **Permitted** [x] **Prohibited**

10. Insurance effected by applicant with Royal Assurance Corp ______ **under policy number** 34983

11. Documents must be presented to negotiating or drawee bank within 15 **days after shipment, but within validity of the Letter of Credit.** *If number of days left blank, it will automatically be considered 21 days.*

FIG 2. The structured parts of the LC

1. Field 5 contains the entry "AUGUST 15, 1985" which is the last date for presentation of documents by the beneficiary.

2. Field 8 contains "JULY 30, 1985" which is the last date for shipment of the goods.

3. The entry in field 11, "15 days", indicates the time period after shipment within which the beneficiary must present his documents to claim payment for the goods.

The decision rule or constraint that ensures consistency among the three dates is: "Difference between the dates in fields 5 and 8 should not be greater than the entry in field 11".

Similarly, in the "terms" field (number 7), an entry in the box FAS (Free at Source), FOB (Free on Board) or C&F (Cost and Freight) indicates that the goods have not been insured by the seller. This requires that the goods be insured by the applicant. The decision rule for this states that the name of some legitimate insurance company (verifiable via a database) must therefore appear in field 10, otherwise the application will be modified or rejected.

There are other such rules that are used to ensure consistency among various fields. Consequently, automation of the structured parts of the application is relatively straightforward.

3.2. Unstructured parts of the LC application

There are three unstructured sections in an LC application. These involve merchandise description, documents requested by the applicant, and other special instructions peculiar to the transaction. Figure 3 shows examples of each of these unstructured parts of the LC application.

Certain inputs simply emphasize the content of the structured fields of the form. Thus the statement "Terms are FOB Taipei Airport" essentially repeats field 7. Similarly, the statement "Mens 100% silk coats packed 12 to a carton in hangars with plastic bags, as per purchase order No. 256., 100 dozen@US$262.54/doz" is an elaborate merchandise description that cannot be entirely accomodated within the fields. Note, however, that this statement contains data that can be used to calculate the total value of the merchandise, which must be identical to the value in field 4. The following statement, an example of documents required by an applicant, expresses ancillary information that is completely divorced from the structured part of the LC: "Airway bill in original plus three copies consigned to order

COVERING - Merchandise must be described in the invoice as:

Mens 100% silk coats packed 12 to a carton in hangars with plastic bags, as per purchase order No. 256., 100 dozen @US$262.54/doz.

Draft(s) must be accompanied by the following documents:

1. Commercial Invoice, original and 3 copies, indicating net weight per dozen and gross weight of shipment. Date of shipment to be indicated. Invoice must be marked "freight collect".
2. Airway Bill in original plus three copies, consigned to order of New York Bank marked notify applicant as shown above, showing date of shipment, flight number and carrier.
3. Inspection certificate signed by John Chen of Taipei Exporters stating that the goods are of acceptable quality and in agreement with our purchase order #256.
4. Signed statement from beneficiary that one complete set of non-negotiable documents sent to Hero Industries at address shown.
5. One set of original documents must accompany shipment.

Special Instructions, if any:

1. Terms are FOB Taipei airport.
2. Bank to telex applicant as soon as possible after payment is effected.
3. All foriegn bank charges are for beneficiary's account.

FIG 3. The unstructured parts of the LC.

of New York Bank marked notify applicant as shown above, showing date of shipment, flight number and carrier."

Similar statements appear in the special instructions section. In general a reviewer must extract the meanings of several such sentences (or groups of sentences) pertaining to the various types of standard documents that an applicant may request, and decide whether they are verifiable by the bank and hence "acceptable" to the bank. In the following section, we describe a computational model of this review process.

4. A MODEL FOR AUTOMATED LC REVIEW

Before describing the model, we should point out that it is sometimes possible to structure a task through the judicious design of a form. We explored this possibility of redesigning the standard LC form but found it infeasible for three reasons. First, experts at the bank indicated that it is not possible to envision all possible documentary and other requirements a priori, and the *form* that these requirements might take (each applicant has a unique style and preference for documentary requirements). Secondly, even if it were possible to enumerate most possible requirements, the form would become large and cumbersome, with large sections of it becoming irrelevant for most applications. Finally, over the last few years, users have become comfortable dealing with the standard application form used by all banks -- which is exactly one 8.5 by 11 inch page[1] -- and are unlikely to accept changes to it readily.

In summary, we have had to accept the situation that the LC form structure will remain largely unchanged, and will continue to have significant amounts of free-form text. This requires that an automated reviewer must have as its fundamental component, a natural language processing system based on a model of the LC transaction.

[1]A computerized "form-blank" is currently filled out by the applicants. It is similar in structure to the paper application form. The form is filled in by the applicant and transmitted to the bank.

4.1. Motivation behind our approach

Two features of LC applications have shaped our model. First, much of the input is grammatically incorrect. Second, it is often the case that very different superficial sentence structures appear in LC applications, with very similar underlying meaning. The similarity arises because of the limited number of basic concepts involved in such transactions (buying, selling and verification) and a core vocabulary of "semantic primitives" (such as "airway bill") that often appear in the input.

The upshot of the first feature is that a syntactically-driven approach to processing is infeasible. The second feature suggests that it should be possible to map different superficial structures into underlying canonical forms in the spirit of Schank's CD theory (1975). In the following section, we illustrate specifically the deficiencies of the syntactic approach, and how they must be overcome to deal with the variety of ill-formed input in LCs.

4.2. Syntactically-driven approaches

Syntactic approaches deal with ways by which elements of a sentence such as phrases and clauses are related in the overall sentence. Augmented Transition Network (ATN) interpreters, the primary example of syntactically-driven parsers, make use of a grammar that embodies the rules of English, represented as a recursive transition network, to obtain a "parse tree" for an input sentence. In addition tests can be performed at nodes in the network in order to determine the subject, object etc. of a sentence. The rules specify the legal relationships among the constituents of the language. For example, a sentence can be decomposed according to the following rules:[2]

```
     <S> ::= <NP> <VP>
    <NP> ::= <NOUN> <VP>
  <NOUN> ::= <PRONOUN> | <PROPNOUN>
    <VP> ::= <AUX> <VERB> <NP> <PP>
    <PP> ::= <PREP> <NP>
```

A complete ATN consists of a set of such rules that are capable of decomposing any legal sentence into its lowest level elements. The entire decomposition is referred to as the parse tree. To illustrate, if an ATN

---

[2] More specifically, there are additional tests and assignments that are made. For further discussion of this, the reader is referred to [Woods, 1970].

interpreter is given the sentence:

"Beneficiary must send Airway-Bill to XYZ-Trading-Company"'

it could generate a parse-tree which in Lisp-like notation would take the form:

```
(S (NP (NOUN Beneficiary))
   (VP (AUX must)
       (VERB send)
       (NP (NOUN Airway-Bill)
           (PP (PREP to)
               (NP (NOUN XYZ-Trading-Company))))))
```

where syntactic elements are capitalized and sentence-parts are in lower case. By performing a few additional tests in the ATN (for the mechanics of this see Hayes & Carbonell, 1983), the subject and direct object of the sentence can be determined.

The advantage of this approach is that it captures the syntactic regularities of the language in a concise and uniform fashion. However the approach fails if given even slightly ungrammatical inputs such as:

" Airway bill to XYZ Trading Company "

Such terse inputs are more the norm rather than the exception in LCs presumably because of the context provided by the section titles, and an implicit set of properties that are part of specialized terms such as "airway bill". For example, since the above input appears in the "Documents required" section, it is understood that the applicant is requesting the beneficiary to send the documents. This context automatically enables a reviewer to understand the input, whereas an ATN based syntactic interpreter would fail. The solution to this problem is a method that focuses more on the *semantic* composition of the input.

4.3. Towards a more semantically-driven approach

The simplest way to focus on the semantics of an input is to cast the grammar in terms of semantic elements instead of syntactic elements (Burton, 1976; Hendrix, 1977). A semantic element in the LC domain could be something like a transport document (such as Airway Bill or Ocean Bill of Lading), instances of which occur commonly in the application domain. The resulting grammar is known as a semantic grammar. To illustrate, a semantic grammar that can be used to successfully parse the above grammatically incorrect input would be:

```
<sentence> ::= <copy-indicator> [of] <transport-document>
               [consigned to] <receiving-entity> [marked]
               <freight-indicator>

<copy-indicator> ::= [1 | 2 | 3 | 4] copy|copies

<transport-document> ::= airway bill |
                         ocean bill of lading |
                         truckway bill

<receiving-entity> ::= applicant | <applicant-name> |
                       <bankname> | <forwarder>

<applicant-name> ::= xyz trading company

<bankname> ::= getrich bank | new york bank

<forwarder> ::= abc forwarding company | jetson movers

<freight-indicator> ::= freight prepaid | freight collect
```

where angle brackets enclose semantic elements, square brackets enclose optional terms, and the vertical bar is a disjunction.

It should be apparent that the size of the grammar can grow rapidly with even marginal variations in superficial sentence structure. To illustrate, consider the sentence:

"Airway bill, copy, must be marked freight collect and consigned to order of applicant",

whose meaning is essentially the same as the previous one. Yet it requires specifying an additional grammar rule, namely:

```
<sentence> ::= <transport-document> <copy-indicator>
              [must be] [marked] <freight-indicator>
              [and] <consigned-to-indicator>
              <applicant>

<consigned-to-indicator> ::= consigned to [order of]
```

There are basically two problems within this approach. Firstly, it is still overly dependent on the position of the semantic elements within an input. There are too many variations in positions of the elements (in part because of the lack of constraints on the grammatical structure of the input), making it impossible to enumerate all semantic rules a priori. A related problem is that certain parts of the input, such as "must", "be", and even "and" rarely contribute toward the real meaning of what is being expressed.

The solution to this problem is therefore to be able to express the meaning of an input in terms of certain "key features" that are recognized in the input regardless of the position in which they occur, with the expectation that the input as a whole can be mapped onto an underlying canonical form. In the following section, we describe such a solution based on a structured object representation. The basic idea is to represent the key features in terms of standard properties -- encoded as structured objects -- that are recognized regardless of their position.

4.3.1. An Object oriented Representation of key features

The central question in adopting an object or "frame" oriented representation is one of deciding on what are to be the key features of the domain. One approach in the literature, called the case frame approach (Fillmore, 1968) employs the idea of a head-concept which is a verb, as the key feature of any sentence, related to a set of subsidiary concepts in a well defined manner (Hayes and Carbonell, 1983). The subsidiary concepts are called the cases of the verb. Fillmore defined verbs in terms of 8 standard cases, namely,

1. agent: the active causal agent instigating the action
2. object: the object upon which the action is done
3. instrument: an instrument used to assist in the action
4. recipient: the receiver of an action -- usually the indirect-object
5. directive: the target of a (usually physical) action
6. locative: the location where the action takes place
7. benefactive: the entity on whose behalf the action is taken
8. co-agent: a secondary or assistive active agent

This approach involves identifying the key verb in the input, and then parsing the remaining input (using a semantic and/or syntactic grammar) into the standard cases. Thus the input:

"Applicant asks beneficiary to send Airway-bill with flight number, airline and date, marked freight-collect"

could, upon identifying send as the head verb, parse into the following structured object representation for SEND :

```
[ SEND
    case-frame
        agent      : beneficiary
        object     : airway-bill
        instrument :
        recipient  : bank
        directive  :
        locative   :
        benefactive :
        co-agent   :
]
```

While this representation begins to capture the relationships among the elements in the input, it has two basic drawbacks in the context of the LC domain.

The first problem occurs in situations where multiple verbs or no verbs at all are present in the input. In the former case it becomes difficult to establish the head-verb and its relationships with the other, often "nested" verbs. When no verb is present, the approach breaks down completely. The second drawback is that the properties (such as "flight number", "freight collect" etc.) of the object (in the above example- "airway-bill") which are central to what the applicant is trying to express, are not represented in the above scheme. Recall that it is actually these properties that will influence whether what is being expressed in the input is going to be accepted by a reviewer, i.e. an input that requests the airway bill to be cleared by customs would be rejected since the purpose of an airway bill is simply to express transaction details and has nothing to do with customs.

These problems associated with using verbs as head-concepts have led to a representation where the head-concepts are nouns and correspond to the basic document types that are used in all LC transactions. The structured objects corresponding to each of these have slots (the equivalent of cases) relevant to the document-type. Such a representation has allowed us to capture the semantics of a wide range of ungrammatical inputs. As an example, consider the following structurally different inputs:

1. Airway Bill, original and 3 copies consigned to XYZ Trading Company, indicating flight number, airline, and date, and marked freight-collect.

2. Original plus three copies of Airway Bill consigned to order of XYZ Trading Company. Bill must clearly indicate flight number, airline, and date of flight, and must be marked freight-collect.

3. Airway Bill in original and 3 copies. Consigned to XYZ Trading Company. Must be marked freight-collect. Airway Bill must also indicate date of flight, name of airline, and the flight number.

Semantically, they are equivalent, in that they require the same properties of a certain document, namely Airway Bill. As we mentioned earlier, in reviewing such inputs, it is these properties that are checked for their acceptability. The representation that captures the semantics of the three inputs is as follows:

```
[AIRWAY-BILL

   original : YES
   num-copies    : 3
   consigned-to : XYZ Trading Company
   flight-number: YES
   airline : YES
   shipment-date : YES
   marked  : freight-collect

]
```

There are similar structures for Beneficiary-Signed-statements, Commercial-Invoice, Packing-List, Inspection-Certificate, and Certificate-of-Origin, which are frequently called-for documents. These are shown in figure 4. Given this representation, what is of interest is the parsing strategies by which these structured objects are instantiated correctly. In the next subsection, we illustrate the parsing strategy for Airway-Bill although the strategies for the other documents are conceptually very similar.

4.3.2. Parsing Strategies

In order to parse an input, the system first recognizes -- based on the possible head concepts -- the document being referred to in the input. Thus, an appropriate head-concept frame is instantiated and its slots need to be filled with the appropriate parts of the text.

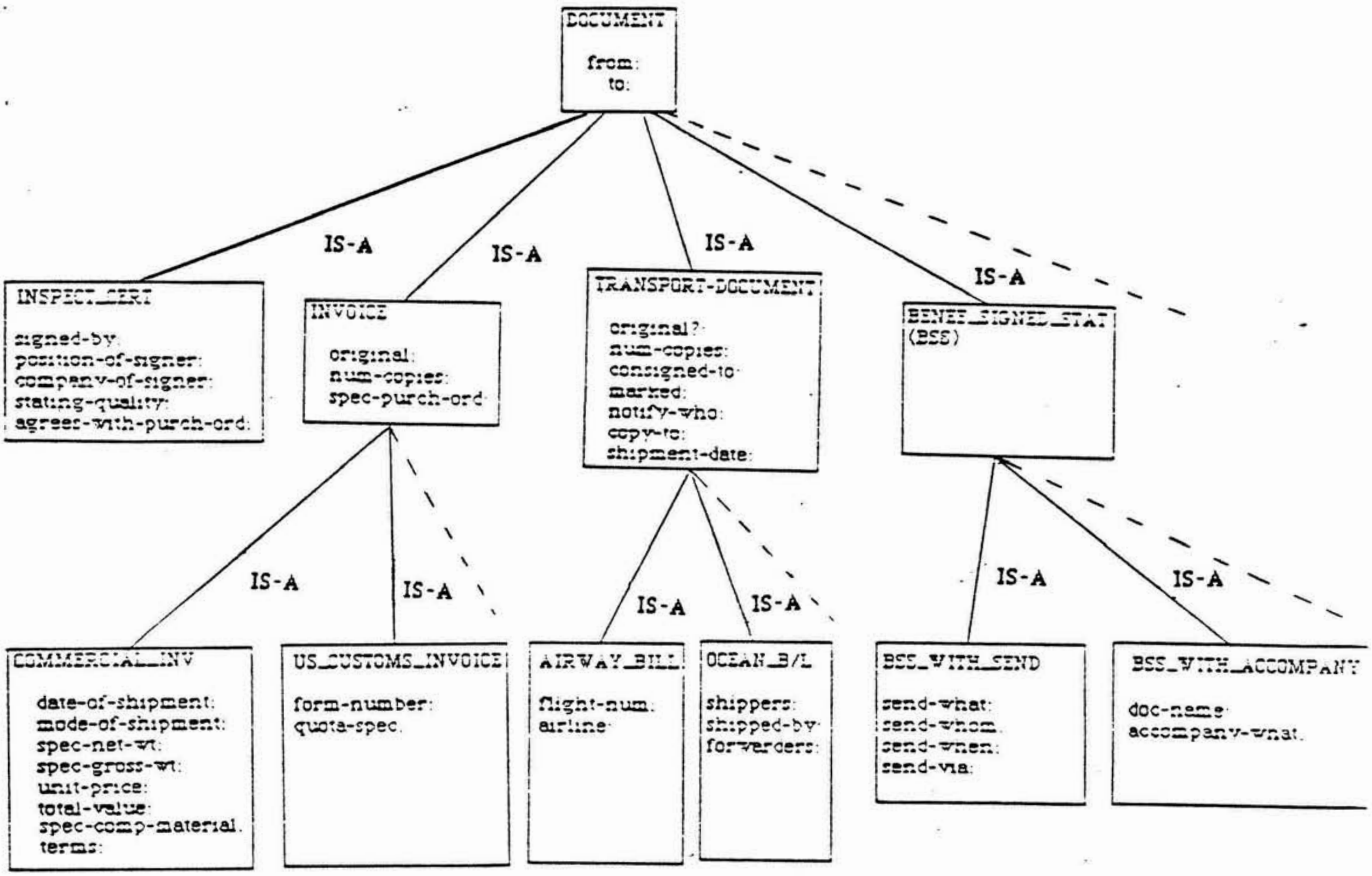

FIG 4. A fragment of the document-type hierarchy.

Associated with each slot is a "filler" and a "marker". A "filler" value is determined from a range of elements or literals relevant to that slot. A "marker" denotes a position in the input following which the filler is to be expected. Both "markers" and "fillers" are defined using a semantic grammar of the form shown in section 4.3. For an example of an object, consider the following:

```
[AIRWAY-BILL

original? :
      marker : nil
      filler : (original)

num-copies    :
      marker : nil
      filler : <num-copies-filler>

consigned-to :
      marker : <consigned-to-marker>
      filler : ((applicant)(new york bank))

notify-who  :
      marker : ((notify)(notification to))
      filler : <entity-in-applicants-country>

copy-to :
      marker : ((copy to))
      filler : <entity-in-applicants-country>

flight-number :
      marker : nil
      filler : ((flight number)(number))

shipment-date :
      marker : nil
      filler : ((flight date)(date of flight)(date))

airline   :
      marker : nil
      filler : ((airline)(carrier))

marked :
      marker : nil
      filler : <freight-info-filler>
]

where:
<num-copies-filler> ::= 1|2|3|4|5
<consigned-to-marker> ::= consigned to [order of]
<entity-in-applicants-country> ::= applicant |
                                   Hero Industries |
                                   New York Bank
<freight-info-filler> ::= freight collect |
                          freight prepaid
```

In addition, several slots also have a "heuristic" which is a LISP function that will be applied to the input if the slot does not have a value.

The algorithm used to determine slot values from the input text is as follows:

1. For each slot of the object associated with the head-concept do:

2. Retrieve the marker associated with that slot.

3. If the marker is "nil", attempt to match each value of filler starting anywhere in the input text (an unanchored match).

4. If the marker is non-nil, then until a successful filler is found or all markers have been tried, do:

5. Attempt an anchored match of the filler to the right of the marker.

6. If all markers and fillers have been tried and the slot has no value, apply the "heuristic" to the input to determine a value for the slot.

In summary, as the head-concept is parsed, the fillers are recognized and stored in the appropriate slots of the object. Since there are not too many head concepts, we have found a top-down un-anchored matching efficient.

5. RESULTS

Twenty five LCs involving a total of 162 individual documentary requests were processed. These 25 LC applications were perceived by bank officials as being more complex than the typical LC. The results can be categorized into 3 types:

1. Cases where the entire text was processed correctly, that is, for which all parts of the text were correctly categorized in the head-concept. 97 cases fell in this category.

2. Cases where a portion of the text was processed correctly and the remainder was not recognized although it should have been. There were 27 such cases.

3. Cases where the system was unable to obtain any representation for the text. This happened in 38 cases.

The cases which fell under category 1 were processed correctly because the requests were generic, dealing with variations of a small set of concepts and their properties.

In category 2, the system was unable to recognize parts of the text for two reasons: (1) The existing set of lexical markers does not cover the complete range of possibilities, and, (2) the unrecognized parts are not associated with any of the slots of the head-concepts. In other words, the particular documentary request has some "properties" that are not recognized. For example, an importer of garments might require that the commercial invoice specify the size and color of the garments, a request peculiar to apparel which is not generic to a commercial invoice.

In order to reduce these errors, it will be necessary for the user to augment the system's knowledge base in cases where it does not recognize parts of the text. To handle the first type of error in category 2 above, we have incorporated mechanisms to let a human reviewer augment the grammar by defining new lexical markers. To handle the second type of error, we have provided machanisms for the reviewer to define new slots for an object and to define markers and fillers for newly defined slots.

In category 3 the system was unable to process the entire text because:

a. The right head-concept was not instantiated, and/or

b. The inability to handle multiple (conjunctive or disjunctive) concepts, and/or

c. A lack of knowledge about the head-concept itself. For example, an importer of apparel may request for a document pertaining to the fiber content of the goods, while an importer of communications equipment might ask for a statement that the goods meet FCC standards. Such documents do not figure in most transactions.

The above results can be summarized as follows:

- There is a large body of generic documentary requests which are independent of the particular commodity involved.
- There is some amount of processing involving product-specific information.

Our approach, of using a product-independent model of the LC transaction process handles a reasonable number of requests. However, it seems clear that more specific models incorporating knowledge about goods are also needed. The system could then do a first pass on an LC application using the generic model, and hand over the uninterpreted parts to a product-specific model. Such multi-strategy approaches are necessary for robust processing (Hayes and Carbonell, 1981) We are currently in the process of identifying classes of such models for the range of goods involved in LC transactions.

6. DISCUSSION

We have presented a model designed to process natural language data that is endemic to large amounts of business transactions. Two features of the problem domain that have motivated our approach are the ungrammatical and terse nature of the text and the stereotypical nature of inputs. The latter feature derives from the fact that the model of the domain restricts the data one can expect in an LC application. The fact that such expectations exist has allowed us to adopt a more knowledge based approach towards "understanding" text than that of domain-independent natural-language systems that engage in large amounts of search (Charniak, 1985). In summary, we have used knowledge about the LC domain to restrict search to localized pieces of text (search for markers and fillers). Formalizing this knowledge has involved identification of a core vocabulary of terms involved in LC transactions, establishing precise meanings for such terms and the relationships among them. As we pointed out at the outset of the paper, this type of theory development for generic office tasks is necessary if they are to be automated.

We have found the Case-frame or thematic-role frame approach to be useful in modelling part of the problem. However, we have found nouns more appropriate as head-concepts than verbs. This stems from the fact that *documents* (nouns) play a central role in commercial transactions Further, much of the processing is directed at ensuring that the properties of these documents are acceptable. Consequently, the subsidiary concepts (slots) of the head-concepts are properties associated with documents. The parsing strategy essentially involves instantiating the object associated with the head-concept, and obtaining values from the input for the various slots of the object.

The main limitation of our system is that it is unable to handle inputs deriving from the idiosyncrasy of specific products involved in the transactions. We have resisted incorporating such knowledge into our model since such data are very diverse and tend to extend the object type definitions to the extent that large parts of the object become irrelevant for individual applications. Yet, from a pragmatic standpoint, it is necessary to be able to process such data. For this reason, we are now working on extending the current architecture to include two models -- the existing one to handle inputs generic to all LC transactions, and industry models that will be used to process data that is specific to type of goods involved in the shipment. We are currently in the process of identifying *classes* of such models for goods such as apparel, foods, furniture, etc., and the *structure* of such models. Once these have been defined, the system should become usable on a routine basis.

REFERENCES

Attardi, G., Barber, G., and Simi, M., 1980. Towards an Integrated Office Workstation, Automazione e Strumentazione, 3, March , 1980.

Barber, G., 1983. Supporting organizational problem solving with a workstation, ACM Transactions on Office Information Systems, Vol. 1, No. 1.

Burton, R.R., 1976. Semantic Grammars: An engineering technique for constructing Natural Language Understanding systems, Bolt, Beranek and Newman Report 3453, BBN Inc., Cambridge, Massachusetts, Dec. 1983.

Charniak, Eugene., 1985. A Single-Semantic-Process Theory of Parsing, Tech. Report, Department of Computer Science, Brown University, 1985.

Fikes, R.E. and Henderson, D.A., 1980. On supporting the use of procedures in office work, Proceedings of the 1st Annual AAAI Conference, American Association for Artificial Intelligence, Aug. 1980.

Fillmore, C., 1968. The Case for Case, in Bach and Harms (eds.), Universals in Linguistic Theory, Holt Rhinehart and Winston, New York, 1968.

Goldstein, I.P. and Roberts, R.B., 1977. NUDGE: A knowledge-based scheduling program, Proceedings of the 5th International Joint Conference on Artificial Intelligence, 1977.

Hammer, M. and Zisman, M., 1979. Design and implementation of office information systems, Proceedings of the NYU Symposium on Automated Office Systems, New York University, NY. 1979.

Hayes, P.J. and Carbonell, J.G., 1981. Multi-strategy parsing and its role in robust man-machine communication, Tech. Rep CMU-CS-81-118, Carnegie-Mellon University, Computer Science Dept., May 1981.

Hayes, P.J. and Carbonell, J.G., 1983. A tutorial on techniques and applications for Natural Language Processing, Tech. Rep. CMU-CS-83-150, Carnegie-Mellon University, Computer Science Dept., Oct. 1983.

Hendrix, G.G., 1977. Human engineering for applied Natural Language Processing, Proceedings of the 5th International Joint Conference on Artificial Intelligence, 1977.

Ladd, I., and Tsichritzis., 1980 An Office Form Flow Model, Proceedings of the National Computer Conference, 1980.

Schank, R.C., 1975. Conceptual Information Processing, North-Holland, 1975.

Tsichritzis, D., 1979. A Form Manipulation System, in Proceedings of the NYU Symposium on Automated Office Systems, New York University, New York, 1979.

Tsichritzis, D., 1982. Form Management, Communications of the ACM, Vol. 25, No. 7, July 1982.

Woods, W.A., 1970. Transition Network Grammars for Natural Language analysis, Communications of the ACM, Vol. 13, No. 10, Oct. 1970.

Yao, S.B., Herner, A.R., Shi, Z. and Luo, D., 1984. FORMANAGER: An Office Forms Management System, ACM Transactions on Office Information Systems, Vol. 2, No. 3, July 1984.

Zloof, M.M., 1982. Office-by-example: A business language that unifies data and word processing and electronic mail, IBM Systems Journal, Vol. 21, No. 3, 1982.