EXPLORER: A Natural Language Processing System for Oil Exploration

Wendy G. Lehnert
Department of Computer and Information Sciences
Graduate Research Center
University of Massachusetts
Amherst, Mass. 01003

Steven P. Shwartz
Cognitive Systems Inc.
234 Church Street
New Haven, Ct. 06510

EXPLORER (Lehnert and Shwartz, 1982; Shwartz 1982) is a non-fragile, "hands-on" language analysis system that allows oil explorationists with no knowledge of computers or computer programming to create customized maps. Users in Tulsa, Denver, and New Orleans currently have dial-up access to a DEC-20 where EXPLORER is implemented in TLISP. A user converses with EXPLORER about a desired map until both parties have agreed on an adequate and unambiguous set of specifications. Another phone link then carries EXPLORER's output to an IBM 3033 which runs database retrieval routines on commercial well data. When all the information has been secured from well data, a graphics system takes over to perform the actual map generation. EXPLORER is currently undergoing evaluation, and it is targetted for a 1983 installation in all regional offices of a major oil company* for restricted use by geologists and geophysicists.

Since our intended user population is naive about computers, EXPLORER's interactive design is dominated by "user-friendly" features. EXPLORER processes information retrieval requests stated in English, without imposing vocabulary limitations or syntactic restrictions on the user. Using a 7000-word dictionary, EXPLORER makes inferences about what a user is saying and initiates interactive dialogues when map specifications are not adequate or potentially ambiguous.

While "user-friendly" system designs often accomodate novice users at the expense of efficient interactions with expert users, EXPLORER naturally tunes itself to both novice and expert users. An expert can state a request very concisely, without a long interactive session, while a novice is guided by a question-and-answer dialogue. Since all users are expected to enter mispellings and typos, an intelligent spelling corrector enables the user to correct typographical errors quickly and easily.

*A proprietary agreement forbids us from identifying them.

Once a request has been analyzed by EXPLORER, the user is given an opportunity to verify EXPLORER's understanding of that request or make changes as needed.

The following example will illustrate the complexity of the map requests that EXPLORER is designed to handle. Following the input, EXPLORER displays its interpretation of the request for verification by the user.

```
EXPLORER     Version 02     9/23/81

READY

*Show me a map of all tight wells drilled
*before May 1, 1980 but since May 1, 1970
*by texaco that show oil deeper than 2000',
*are themselves deeper than 5000', are now
*operated by shell, are wildcat wells where
*the operator reported a drilling problem,
*and have mechanical logs, drill stem tests,
*and a commercial oil analysis, that were
*drilled within the area defined by latitude
*30 deg 20 min 30 sec to 31:20:30 and
*80-81.  scale 2000 fest.

By FEST, do you mean FEET  (Y or N)    *y

User requests a POSTED map
latitude  30.34167 - 31.34167
longitude  80 - 81
output medium: PAPER
output device: PHOTODOT
filters: DRILLING DEPTH > 5000 FEET
         COMPLETION DATE >= 5/1/1970
         COMPLETION DATE < 5/1/1980
         OIL ANALYSIS AVAILABLE
         DRILL STEM TESTS PERFORMED
         MECHANICAL LOG FILE WELL
         DRILLING PROBLEM
         WILDCAT WELL
         SHELL CURRENT OPERATOR
         WELL SHOW OF OIL > 2000'
         TEXACO ORIGINAL OPERATOR
         TIGHT HOLE WELL
scale of 1" = 2000.0'
```

As this example shows, EXPLORER can handle long and ungrammatical requests, make inferences, and perform complicated word sense disambiguation. For example, the word "show" has two distinct meanings in the above request. The first instance of "show" is synonymous with "display", whereas the second time "show" is used it has a technical meaning (i.e., an indicator of oil or gas in a well). The surrounding conceptual context is used to determine which sense of "show" is appropriate in each case.

We suspect expert users who utilize EXPLORER regularly will develop a feel for EXPLORER's limitations and flexibilities. If such a user wants to enter a request as efficiently as possible, EXPLORER will handle "telegraphic" English which does not have to conform to standard syntactic conventions:

```
EXPLORER     Version 02     9/23/81

READY

*tcg bibb georgia ci 100 scale 2000.

 User requests a STRUCTURE map
  output medium: PAPER
  output device: PHOTODOT
  county: BIBB, GEORGIA
  on TCG
  CI = 100'
  scale of 1" = 2000.0'
```

As long as the request is conceptually coherent and unambiguous, EXPLORER will be able to handle a wide range of stylistic constructs. This last request was sufficiently specified for EXPLORER to interpret it without further user interaction, but many requests (especially those generated by novice users) are likely to require an interactive dialog. For example, the following interaction might take place with an extremely novice user:

```
EXPLORER     Version 02     9/23/81

READY

*Map the tcg.

A map region was not specified.
Do you want the same geographic region as the
last request (Y or N)?

*n

Do you wish to specify the map region by
county or by geographic coordinates?

*c

Please input county or counties by name or
number.
*Bibb

Bibb County is in the following states:
ALABAMA GEORGIA

Please input the appropriate state or a new
county specification.

*make it new haven, ct.

A contour interval was not specified.

Would you like the CI from the previous
request? (Y or N)
Previous CI = 100'

*y

A scale was not specified.

Would you like the scale from the previous
request? (Y or N)
Previous scale = 2000.0 FT-PER-IN

*y


  User requests a STRUCTURE map
   output medium: PAPER
   output device: photodot
   county: NEW HAVEN, CONNECTICUT
   on TCG
   CI = 100'
   scale of 1" = 2000.0'
```

EXPLORER will query a user as needed to get missing information and resolve any ambiguities that may be present. Notice that EXPLORER naturally offers the user an option of inheriting many specifications from the previous map request. Explorationists often find it useful to examine a sequence of related maps, so our interface has been designed to make map sequences easy to generate.

EXPLORER has been undergoing an initial test phase since July 13, 1982. During this time a variety of oil company employees have dialed up the program and entered map requests. While we do not yet have enough test requests for a comprehensive evaluation of the system, we have analyzed EXPLORER's performance over the three-week period from 7/13/82 to 8/6/82 in an effort to assess its strengths and weaknesses. During this time 39 requests were successfully transmitted to EXPLORER by 8 different individuals. In order to evaluate EXPLORER's performance, we will consider the following categories of performance:

(A1) original input is interpreted correctly on the first try - perfect performance.

TABLE - 1

REQUEST TYPE	NO. OF REQUESTS	SURFACE	INTERACTIVE	CONCEPTUAL
A1	4 (10%)	19(15-25)	3(3-3)	9(9-10)
A2	26 (67%)	22(1-87)	7(3-14)	11(9-22)
A3	9 (23%)	37(9-57)	8(5-12)	12(10-14)
total	39 (100%)	25(1-87)	7(3-14)	11(9-22)

(A2) original input is interpreted correctly after one or more clarifying interactions. These interactions may be due to typing errors, spelling errors, missing information, or system errors.

(A3) original input is never interpreted correctly due to a system failure of some sort.

If a request can be categorized as an A1 or A2 request, EXPLORER is fully functional even though it may make a mistake at some point in its processing. For example, if EXPLORER does not recognize a word, it will query the user for synonyms. If one of the synonyms is recognized, EXLPORER recovers from its own recognition error, and the request will be categorized as an A2 request. When a system error is fatal in the sense that the user does not or cannot recover from it, we categorize the request as an A3 request: an A3 request should not result in map generation. We have omitted from this analysis any requests that were aborted due to transmission errors or user-initiated interrupts.

In addition to our three performance categories, we will characterize the general complexity of a request in three ways:

[1] Surface Complexity:
The number of words in the original input request.

[2] Interactive Complexity:
The number of complete interactions between the user and EXPLORER during a single request dialog.

[3] Conceptual Complexity:
The number of lines generated in the target query language.

We realize that some users will try to maximize efficient communication by minimizing the number of complete interactions. At the same time, still other users will find it easier to enter a minimal request and let the system ask for more information as needed. So while there is an apparent trade-off between the length of the initial request (surface complexity) and the number of interactions needed to fully interpret that request (interactive complexity), we cannot evaluate EXPLORER's effectiveness by trying to minimize one or the other.

We must also note that conceptual complexity as it is defined here can only give a very rough idea of the conceptual content and information processing involved. It might be tempting to look for conceptual complexity as a function of surface complexity and interactive complexity, but any simple decomposition along these lines will be misleading. If a user changes the scale of a map 10 times, we will see a large interactive complexity with no change in conceptual complexity. A more sensitive set of complexity measures will have to be designed before we can expect to see correlations across the various measures.

The results of our trial test period are summarized in Table 1. We see that the average surface complexity of all requests is 25 words, with requests ranging from 1 to 87 words in length. Each request averaged 7 complete interactions, with some taking as few as 3 and others requiring as many as 14 user-interactions. The target query language requests averaged 11 lines of code, with a range between 9 and 22 lines.

In terms of performance categories, fully 67% of all requests were A2 requests. Only 10% qualified as A1 requests, with the remaining 23% falling into the A3 category. A3 requests tended to be slightly more complicated on average than A2 requests, but it is important to note that the most complex requests in terms of all three measures were nevertheless A2 requests. The relatively small precentage of A1 requests may not be significant given the size of our sample, but it is likely that the failed A3 requests would have been A2 requests had they been processed successfully. As the system's hit rate improves, we expect to see the A2 rate rise while the A1 rate remains stable. It is interesting to note that the average surface complexity of the A1 requests is very close to the average surface complexity of the A2 requests.

Almost all of the errors underlying our A3 requests were programmer errors due to an imperfect understanding of user vocabulary or the target query language. This was expected and can only be rectified with continued testing by qualified users. We are extremely pleased to have a 77% success rate at this initial stage of program test-development: EXPLORER's error rate should decrease over time as changes are made to correct the errors we uncover.

Our experience with EXPLORER suggests that it is impossible to complete a system of this complexity without some such testing phase for feedback purposes. A high degree of cooperation between program designers and intended users is therefore critical in these final stages of system development.

Our next step is to continue testing revised versions of EXPLORER, expanding our user population as the system becomes more competent. At the current rate of user feedback, we project a 3-6 month period of system revisions before we freeze the implementation for a final evaluation.

REFERENCES

Lehnert, W. and Shwartz, S. (1982). Natural Language Data Base Access with Pearl. Proceedings of the Ninth International Conference on Computational Linguistics. Prague, Czechoslovakia.

Shwartz, S. (1982). Problems with domain-independent natural language database access systems. Proceedings of the Association for Computational Linguistics. Toronto, Canada.