Experience with INTELLECT: Artificial Intelligence Technology Transfer

Larry R. Harris

Artificial Intelligence Corporation
100 Fifth Avenue
Waltham, MA 02254

Editor's Note: At the 1983 National Conference, there was a special session on Technology Transfer. Six speakers from different industrial organizations presented their personal views on the process of turning the results of AI research and development into commercial practice. Unfortunately there was not time to produce and distribute a written record of the Symposium for the attendees. To correct that problem, we will publish edited versions of the six oral presentations, at the rate of one or two per issue.

The following two articles, by Larry Harris of Artificial Intelligence Corporation and S. Jerrold Kaplan of Teknowledge, inaugurate the series Other speakers at the Symposium were Reid Smith (Schlumberger), Stephen Polit (DEC), Ed Taylor (TRW) and Earl Sacerdoti (formerly Machine Intelligence Corporation, now with Teknowledge). — *Robert S. Engelmore*

ARTIFICIAL INTELLIGENCE TECHNOLOGY TRANSFER is the diffusion of AI research techniques into commercial products. I have been involved in this process since 1975, when the Artificial Intelligence Corporation began to develop ROBOT, the prototype of INTELLECT, a commercially viable natural language interface to data base systems which has been on the market since 1981. In this article, I will discuss AI technology transfer with particular reference to my experiences with the commercialization of INTELLECT.

I will begin with the historical perspective of where the field of AI came from, where it is now, and where it is going. Next, I will describe my interpretation of the present market structure for AI products and some specific marketing perspectives. I will then briefly describe the product INTELLECT and its capabilities as an example of a state-of-the-art commercial system. Next, I will describe some of the experiences, which I think are typical, that my company has encountered in commercializing INTELLECT Finally, I will summarize my main points and give some advice to AI researchers who are planning to commercialize their systems.

The Historical Perspective

When the first artificial intelligence research was done, the technical challenges were so great that it was impossible to work on real commercial problems. Instead, researchers addressed the basic issues of the field through work on simple, "toy" problems. In the very early days, researchers hoped to discover some basic underlying techniques for solving a whole class of problems. Research was focused on very general approaches, such as the General Problem Solver and heuristic search techniques. As time went on, the hope of finding underlying techniques diminished, and people began focusing on very specific systems. The outgrowth of that approach is today's expert systems technology, which exhibits a very narrow band of capability but is highly functional within that band.

At this time, artificial intelligence technology has reached the point where it can be and is being applied to some real-

world problems. This is a positive step for the field, because as we address these commercial problems, researchers are discovering some fundamental issues and some unresolved important problems of AI that remain to be solved. These discoveries provide feedback and direction for future research.

The AI Software Market Structure

There is a long distance between designing a system which successfully addresses a toy problem and finally designing and marketing a commercial system. When moving from the research environment into the commercial sphere, researchers must learn to define the kinds of marketable products needed, to determine their attributes, and to outline the market structure for these products. The first type of AI systems are usually custom systems, which solve one problem for one user. A number of companies exist that build such systems under contract for specific organizations. There are also generic systems having many applications that can be sold over and over to a variety of organizations. These are sold by companies that have a very strong product orientation.

Companies having the latter type of product orientation can choose to design vertical or horizontal application products. A vertical application product has an industry-specific capability. For example, a system that configures minicomputers could be sold to many different computer manufacturers. Another vertical application product would be a system having some natural language capability and considerable expertise in personnel for use with a personnel data base. This could be sold to a much larger group of companies than the first product, but it would still have only one application.

There are also horizontal applications products, or what I call "general systems software". An example of this type of product is a generic expert systems tool. This type of product addresses a much broader market than a vertical application; it can be sold to all industries. However, it also puts a great deal more pressure on the underlying technology, since it must be possible to configure such a system for a variety of applications.

Horizontal application systems must be designed so that they can be applied by personnel with no knowledge of artificial intelligence. A system that requires a computational linguist, for example, to design each new application will be unsuccessful because there are very few computational linguists available. It is important for companies selling general systems software to provide customer support. Even if a system is designed to be applied by people without AI training, they will make mistakes and ask the AI company to help solve their problems. The company must be able to respond to those problems and to meet the needs of the customer base. Designing an AI-based technology that can be used and even supported by non-AI personnel is a challenge that must be met by companies developing a horizontal product-oriented capability.

The Marketing Perspective

What are the implications of these choices? What does it mean to choose one marketing perspective over another? One important factor in this choice is leverage, namely, how many times a company can resell the work that it has put into a product. At the one extreme is contract development consulting, which provides the least amount of leverage. A great deal can be charged for custom systems, but they can be sold only once. At the other extreme, a totally product-based orientation allows a company to sell each product many times. This approach provides the most leverage.

The second factor to consider is market size. General-purpose generic software can be sold to a much broader market than application-specific software. It is usually harder to sell, because people find it difficult to understand how to apply general-purpose capabilities, whereas they can immediately understand and use application-specific software. However, application-specific software has a limited size market, and prospective developers should determine its market size first.

The third important factor is orientation. Should the process of bringing artificial intelligence technology into the marketplace be technology-driven or market-driven? The technology-driven or "technology push" approach is bringing the technology that you know and have developed into the market before examining the needs of that market. Since all AI researchers are basically technologists, this approach is a tempting one. However, it may prove costly; a company using the technology-driven approach may develop a product only to find that there is a very small market for it. The best approach is a market-driven approach, which researches the needs of the market and develops the appropriate AI technology to meet those needs.

The fourth factor to consider is integration. A system can be sold either in a stand-alone mode or interfaced with software manufactured by other companies. An expert system for general ledger or financial applications must somehow interface with the existing financial data bases in the client corporations. If such a system were designed on a stand-alone basis, it would have to replicate all the functions of the clients' existing software, and the clients would have to be willing to convert their existing structure and orientation to a new mode of operation. When designing a natural language interface to data base systems, you must decide whether to build a data base system or not. Since IBM just invested 350 person-years of effort in developing its new release of the data base system SQL, it would be difficult to compete with that technology even if your system has a better natural language capability. Almost every area of AI must somehow merge with the existing commercial structure.

INTELLECT: Natural Language Information Center Software

INTELLECT is a natural language product that works in

the Information Center environment doing natural language query against data bases and interfacing with other software tools. The system is based on a non-deterministic augmented transition network parser. Since our orientation is market-driven, we simply chose a technology that would achieve the desired results and continued to use it even after newer, but less proven, techniques became available. In the final analysis, the underlying technology plays a very small role in the overall scheme of things.

INTELLECT also employs domain-independent semantics. Since we wanted to build a generic tool that could be applied by people with no knowledge of AI across a wide variety of application domains, the semantics could not be embedded deeply in the system. Designers building generic prototypes must make an initial commitment to keeping semantics as domain-independent or as loosely coupled as possible. They must resist the temptation to make commitments to specific applications in order to solve specific problems. Otherwise, they may later have to redesign the system from a generic-tool perspective.

Our orientation is product-based; we want to sell the same product repeatedly. We want it to be general purpose so that it can be used in a wide variety of application domains and we want to remove as much of the AI mystique as possible from the process of using it. In terms of market positioning we have made the commitment to be market-driven, to find out what the real needs of the marketplace are in terms of the problem we are trying to solve, and to choose the appropriate technology to solve that problem. We also made the commitment to interface to existing software and to work within the common commercial data processing structure, but, at the same time, not to try to reproduce the existing data base technology, graphics technology, and so forth.

INTELLECT was designed to address the spectrum of natural language applications which we feel are commercially viable today. These include data base query and integrated access to a variety of managerial software tools which can be used for more complex analytical requests. The latter applications were introduced about two years ago; they appeal to a different market sector than pure data base query.

Below is an example of one type of system INTELLECT is competing with, a state-of-the-art formal query analysis system.

```
USER:
PRINT NAME, 82-JUL-ACT-SALES, 82-JUL-EST-SALES,
   82-JUL-ACT-SALES - 82-JUL-EST-SALES,
   (82-JUL-ACT-SALES - 82-JUL-EST-SALES)
/ 82-JUL-ACT-SALES, IF REGION='NEW ENGLAND'
   AND 82-YTD-ACT-SALES UNDER 82-QUOTA
```

Even though the question is reasonably simple, most people do not immediately understand it. There is a conceptual gap between the expression of the formal request and the way people think. The whole purpose of a natural language system, however, is to allow people to express the same question in a variety of ways, each of which is immediately understandable. Below is one expression to the INTELLECT system of the same request

USER: I WONDER HOW ACTUAL SALES FOR LAST MONTH COMPARES TO THE FORECAST FOR PEOPLE UNDER QUOTA IN NEW ENGLAND

INTELLECT: PRINT A COMPARISON OF LAST NAME, 82 JUL ACT SALES AND 82 JUL EST SALES OF ALL SALES PEOPLE WITH REGION = NORTHEAST & 82 YTD ACT % QUOTA UNDER 100 00

THE NUMBER OF RECORDS TO BE SEARCHED IS 40

LAST NAME	1982 JULY SALES	1982 JULY EST SALES	CHANGE	% CHANGE
SMITH	$54,474	$52,868	$1,606	2 95
ALEXANDER	$54,833	$52,936	$1,897	3 46
ADKINS	$76,072	$75,631	$441	0 58
ASIN	$42,144	$38,214	$3,930	9 33
BRADY	$40,530	$39,569	$961	2 37
COOKE	$41,318	$40,406	$912	2 21
GIRTON	$40,423	$40,393	$30	0 07
GNANDT	$64,213	$63,878	$335	0 52
GOLDSTEIN	$37,977	$37,942	$35	0 09
GOODWYN	$63,779	$63,906	$127-	0 20-
GOULD	$77,161	$75,870	$1,291	1 67
HILTON	$79,412	$75,388	$4,024	5 07
JOCHEM	$66,103	$66,316	$213-	0 32-
JOCHEM	$103,455	$100,808	$2,647	2 56
KENWORTH	$76,879	$77,633	$754-	0 98-

NEXT REQUEST

The difference between the state of the art in formal systems technology and the contribution that AI makes is immediately identifiable, something that people relate to and understand. People will clearly react to the technology very positively when they have the choice of using it. The real problem for AI companies is delivering on that promise, because people will look at examples like this, make extrapolations, and build expectations that the system must live up to. We must either scale down their expectations or live up to them.

Below is an example of another question presented through INTELLECT to a data base that a sales manager might maintain on his selling force

USER: GIVE ME THE NAMES OF THE WOMEN IN THE WESTERN REGION WHO ARE OVER QUOTA

INTELLECT: PRINT THE LAST NAME AND 82 YTD ACT % OF QUOTA OF ALL SALES PEOPLE WITH SEX = FEMALE & REGION = WEST & 82 YTD ACT % QUOTA OVER 100 00

THE NUMBER OF RECORDS TO BE SEARCHED IS 24

LAST NAME	1982 YTD % OF QUOTA	LAST NAME	1982 YTD % OF QUOTA
ACORD	158 20	DOLON	152 42
ARBOGAST	146 37	FREUH	150 90
CENTOR	145 55	GARDNER	135 64
CHAMPION	147 26	GASKINS	152 66
CHIANG	138 49	HERNANDEZ	168 08
COBBS	141 71	HERNANDEZ	159 07
COIRO	178 16	HEWITT	146 02
DEXTER	157 30	ISON	141 80
DIX	142 41	ISRAEL	152 75
DOLLAR	143 03		

The two lines below the question are an unambiguous, formalized restatement that INTELLECT presents to the user to show how it interpreted that request. Since there are no ambiguities to resolve, it immediately delivers the answer. The user of a natural language system can choose from an exceedingly wide variety of syntactic constructions. When the same question is asked another way, "Who are the women that are over quota out West?" the system gives roughly the same echo and answer. The only difference is that it also gives first names, since the lexicon is built to respond with both first and last names to "who" questions. When the same (reasonably complex) question is phrased in an extremely terse fashion, "Western women over quota," the system again gives roughly the same echo and answer. In short, the user can ask the question in any way he or she wants to, rather than learning the way that the formal system wants it to be asked.

Natural language is ambiguous. The system must be able to detect and clarify ambiguities as it goes along Below is a simple demonstration of this ability.

USER: LIST THE SALESMEN IN NEW YORK WHO ARE UNDER QUOTA

INTELLECT: YOUR REQUEST IS AMBIGUOUS TO ME. DO YOU WANT
1) CITY = NEW YORK
2) STATE = NEW YORK
PLEASE ENTER THE NUMBER OF THE INTERPRETATION YOU INTENDED

USER: 2

INTELLECT: PRINT THE LAST NAME AND 82 YTD ACT % QUOTA OF ALL SALES PEOPLE WITH STATE = NEW YORK &82 YTD ACT % QUOTA UNDER 100 00

THE NUMBER OF RECORDS TO BE SEARCHED IS 60

LAST NAME	1982 YTD % OF QUOTA	LAST NAME	1982 YTD % OF QUOTA
BAHN	96 24	HOLMAN	98 15
DYKES	96 10	JONES	94 34
ELEY	96 60	KERO	95 88
GRAYSON	97 42	KEYSER	96 08

NEXT REQUEST

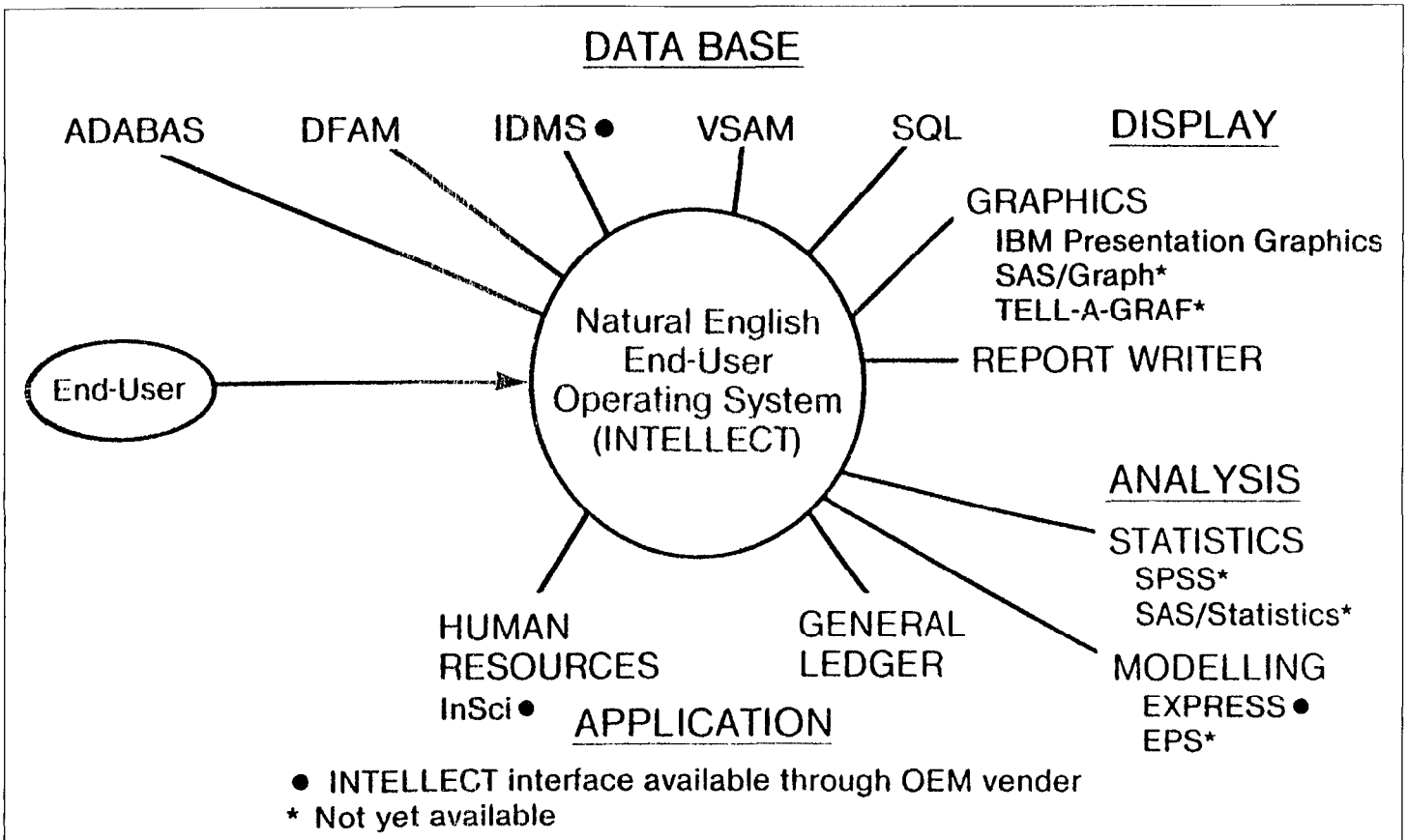
Since it is not clear whether New York City or New York State is meant, INTELLECT provides a menu of two choices that the user can make to resolve the ambiguity. The system is committed to resolving all ambiguities before answering the question. It is much more important to give no answer or to ask for clarification than to randomly guess at the intended meaning in order to give an answer. In other cases, the system can infer what the user's meaning is. If the user asks, "Which of the New York employees live in Buffalo?" the system determines that Buffalo is a city and that therefore New York State makes more sense. The system must decide when to ask the user for clarification and when to try to infer the user's meaning.

INTELLECT can analyze data as well as retrieve it from the data base. The market requires the ability to analyze data at several levels. Below is an example of the simplest step up of doing subtotaling along a specific dimension.

USER: SHOW THE TOTAL SALES BY STATE
INTELLECT: PRINT THE TOTAL 82 ACT YTD SALES IN EACH STATE OF ALL SALES PEOPLE
THE NUMBER OF RECORDS TO RETRIEVE IS 300

STATE	1982 YTD SALES
ARIZONA	$3,960,221
CALIFORNIA	$28,073,666
COLORADO	$4,155,372
FLORIDA	$5,444,594
GEORGIA	$2,253,687
ILLINOIS	$11,020,176
KANSAS	$5,071,215
MASSACHUSETTS	$10,690,839
MISSOURI	$7,735,974
NEBRASKA	$2,514,975
NEW HAMPSHIRE	$3,952,765
NEWMEXICO	$1,742,743
NEWYORK	$29,446,794
OREGON	$2,933,139
PENNSYLVANIA	$13,067,064
TENNESSEE	$2,161,561
VERMONT	$2,281,777
WASHINGTON	$5,724,593
OVERALL SUM	$142,231,155

Here the user is not requesting specific records in the data base, but the building of a report showing the sales on a state-by-state basis. The user's next request is, "Show percentages." The user conceptualizes the addition of a column showing the percentages of the totals, so as to see more clearly the relative performance on a state-by-state basis. The system has to perform a good deal of calculation to generate this column. It goes from a one-pass to a two-pass process because it must compute the grand total before it computes the percentages Many formal query systems write out these subtotals to a temporary file and spool them back in. It is a great deal simpler for the user to question the system at the level at which he or she thinks of a problem — "Show me the percentages"— and to obtain the information



INTELLECT: The Information Center.

Figure 1.

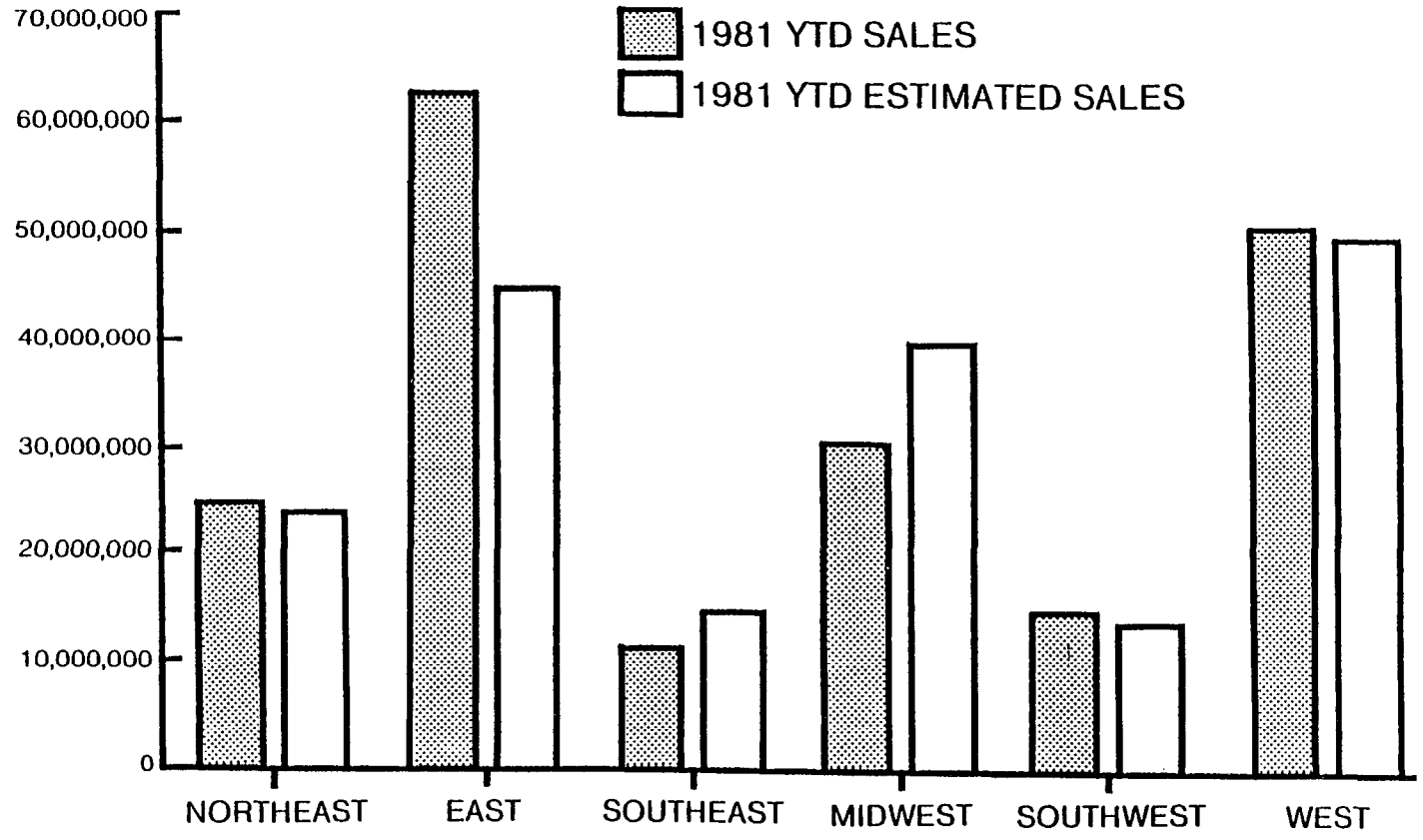
without needing to be concerned with the computational procedure the system requires. One step beyond the last query is, "Rank them." The system carries out what to the user is a simple analytical process. However, it would be very difficult to specify the required third-pass to sort the percentages, which must be done by the users of a formal query facility.

We recently began to address the problem of using INTELLECT to coordinate the specialized software tools in the Information Center. The Information Center is one of IBM's most successful marketing concepts; it was developed about ten years ago. The Information Center enables users to solve their own problems rather than going through the data processing software development cycle, where specific applications might take two or three years to build The procedure is to give the users their own machines and some software tools and allow them to solve their own problems. The basic orientation is towards end-user computing.

The typical tools available to users in this environment are a data base system, a graphics system, a modelling system, and a statistical system. INTELLECT is being marketed, not as just another tool within the Information Center, but as the supervisory tool. A natural language system frees users from the necessity of learning a different formal language for each tool they work with and from writing programs to package data and send it from one application to another. Instead, the users can use INTELLECT to analyze a request, partition the work among the software tools, and orchestrate the passage of data from one subsystem to another to carry out the overall request. The data base and graphics systems, for example, are enormously powerful tools the user can now use without having to be a specialist in each one. Figure 1 shows where INTELLECT is in terms of achieving that goal. We have interfaces to a variety of data base, graphics, and analysis systems. We will continue to enhance INTELLECT by offering interfaces to a variety of other tools.[1]

Figure 2 shows the power of this concept. When the user types, "Show me a bar chart of the actual and estimated year to date sales in each region," INTELLECT requests information from the data base, evokes the summarization mode to process the data, packages it, sends it to the graphics facility, and causes this graph to be created. The system goes from raw data in the data base to a presentation-quality graph in 10 to 15 seconds of real time. This is an astounding step forward for most of the companies that have these tools

[1]We have recently extended this concept by introducing interfaces to microcomputer based software products, such as spreadsheets and graphics systems This allows the concept of the "English Environment" created by INTELLECT to link the capabilities of both mainframe and micro-computers.



Bar Chart of Actual and Estimated Year-to-Date Sales in each Region.

Figure 2.

but cannot integrate them in a way that allows this type of task to be carried out this quickly.

Experiences With Commercializing INTELLECT

I will now discuss some issues that we focused on while transferring INTELLECT from the research environment into the marketplace The first issue is density of coverage. How many ways of making the same request will be accepted by a natural language system? How many slight variations of that request will be rejected? The user may question the system at one point in time and receive an answer. Later, the user may type in what he or she believes is exactly the same request, only to have it rejected by the system. In fact, the user may have merely left out an article or transposed two words — very slight variations which are insignificant to the user but which are sufficient to cause the system to reject the sentence. This problem annoys users, and it may cause them to lose confidence in the system and their ability to use it. They may think that the system is not dependable or that it does not always work in thc same way.

To avoid this loss of confidence, the designer must make a firm commitment to achieving sufficient density of coverage, so that the system will accept all reasonable wordings of the same request. The users are not trying to deceive the system by wording their requests obscurely; they want to ask questions in ways that the system will accept. The designer's task is to allow them to do so. The designer does not need to achieve universal coverage of the English language, but must very densely cover the part of the language that the users will work in. The designer does not necessarily have to worry about some of the esoteric constructions that concern linguists, but must cover the area of the commonly asked contractions extremely well. This requires a strong commitment to detail in terms of building the grammar, maintaining the grammar, and testing the grammar. Sufficient density of coverage is essential if a natural language product is to be commercially acceptable.

The second issue is the transportability of the domain of discourse. It must be possible to transport the system from one application's domain to another The examples above showed INTELLECT applied to a sales data base. Other domains to which INTELLECT has been applied include: a human resources data base, a product data base, a financial database, inventory, and a variety of resource allocation applications.

Each application requires transporting the system from one domain of discourse to another. That process must be

carried out by people without specific AI training. The designer must make a tremendous initial commitment to building the system in such a way that this can be done When we were designing INTELLECT, I became concerned about the apparent lack of interest in the research community in developing techniques to facilitate transportability of natural language systems. Most academic systems are highly dependent on specific domain semantics. Since it is very taxing to handcraft semantics for each application domain, the use of domain-independent semantics is necessary for a commercially successful system. Successful solution of this issue is perhaps the most critical aspect of bringing a natural language based product to market.

The third issue is interfacing. As I mentioned earlier, a system can be sold either in a stand-alone mode or interfaced with available software. We chose to design INTELLECT to interface with as much as possible of the software available on IBM mainframes, which are the principal machines used by our market. INTELLECT interfaces with most of the popular data base management systems used on IBM computers, such as ADABAS, IDMS, VSAM, and SQL. In addition, INTELLECT must function within the IBM operating system and teleprocessing environment required to support a large user community. Therefore, we designed INTELLECT to support all the operating systems, virtually all the popular teleprocessing monitors, and some graphics systems. Building interfaces to these provides little challenge to the AI programmer, but it had to be done if INTELLECT is to function as a supervisory tool within the Information Center.

Another aspect of commercialization is feedback from users. The Artificial Intelligence Corporation has probably had more experience with users than any other AI company. We have sold over 200 copies of INTELLECT, and at some sites over 100 people use the system regularly. We have two main groups of customers. The first group consists of companies to which Artificial Intelligence Corporation has direct sales. Since we have the most direct contact with this group, we know their needs best. The second group consists of customers that license INTELLECT to other software vendors who embed it in their product lines.

We deal directly with the Fortune 500 corporations. Since INTELLECT is a generic tool, a wide variety of these companies find it useful in their environments. We have sold it to insurance companies, oil companies, manufacturers, retailers, banks. As users, their response directs the future development of the product. Dupont's feedback was the most instrumental. In the late 1970s, when we were nearing the end of our product development phase, we felt ready to meet the demands of the commercial marketplace and shipped our product to Dupont. Not until a year after our shipment did they accept it and place it into production. Our experience with Dupont made us realize that designing a commercially viable system was much more difficult than we had expected

Natural language technology must be interfaced with other software in order to derive value. We focus on data base and graphics systems. Our second group of customers consists of original equipment manufacturers [OEMs] and joint marketing agreement companies that market their own applications systems and want a natural language technology embedded in their product to be sold within their product line. Some of these, like Cullinet, are major vendors of software for IBM mainframes. Others are human resources software vendors that sell specific, narrow-application packages which need a natural language system to query the files. When dealing through OEMs, we are dealing with another layer of indirection. Their customers must be able to apply the system in their environments, and this exacerbates the problem of designing the system so that non-AI personnel can use it. So far, we have been very successful; Cullinet has no AI people to support INTELLECT, yet they are able to sell and install the system. We would be unable to establish these OEM connections if the system were not domain-independent or could not be customized by people with no knowledge of artificial intelligence.

In June 1983, IBM announced that it had acquired non-exclusive marketing rights for INTELLECT. This was a precedent set because it was the first time that IBM has licensed business professional software for mainframes from an outside vendor This precedent recognizes the existence, credibility, and viability of artificial intelligence in the marketplace.

We have gained valuable information from user feedback. Many designers believe that a natural language system needs only the ability to retrieve data from the data base. However, users also want the system to analyze data and to bring it up to the level of detail needed to solve the problem First, this implies that the system must be able to understand the linguistic constructions that express the analysis. Second, it must be able to carry out the analysis. A significant amount of computation must take place after retrieval, and the work that the system does at this point the data is as important as the work it does before the retrieval.

We also discovered that users want the system to perform time series analysis of data For example, users of sales data bases want to see how current sales compare to the estimates that they made at the beginning of the year and how this year's sales compare to last year's. If the sales statistics are stored monthly, quarterly, year-to-date, and so on, for five years, the data base may contain over 1200 different sales figures for each product Dealing with the alternatives presented by the request, "Compare this year's sales to last year's," introduces its own linguistic and computational complexity. Considerable computation is required to decide which sales figures to retrieve from a file containing 1200 sales fields

We encountered some interesting problems when INTELLECT was installed We discovered that we had to question many of our basic assumptions about building a lexicon For instance, one site's data base had a sex field with ten unique values, rather than the expected two. At another company,

the user asked the system to list "360 sites." An obvious response to this request is to list the first 360 sites in the data base. However, the user actually wanted the system to list the sites with IBM 360 computers. INTELLECT had to be designed so that, in cases like this, non-AI users can define special terms.

Another site's personnel file contained 2200 different job titles. Originally the titles were abbreviated to eight characters. Later, they were expanded to 12 characters, and most recently to 20. Every job title in this file could have at least three different spellings—the very short, the medium short, and the not-so-short spelling. INTELLECT had to be initialized with all the spellings and meanings of each job title, even those that were obsolete.

Conclusion

My recommendations for academic researchers embarking on this voyage are as follows:

- First determine the market segment that you want to address and the requirements of the system you want to develop.
- Then choose a technology that meets those needs in the eyes of the user.
- Develop a complete marketing strategy—decide exactly how to achieve your goals and determine the implications of your decisions.
- Then develop a prototype, but refrain from announcing it as a product.
- Avoid going to the marketplace immediately claiming that the system will achieve the desired results. Instead, get some users to test it in a real-world setting on a low-key, low-profile basis.
- Expect that it will take two to five years–or even longer–to productize that prototype. It is difficult to build a product in any field of computer science; it is particularly difficult in the field of Artificial Intelligence.