ABSTRACT
A working expert system for assessing a student's course requirements for graduation in various departments of a university is presented. The system draws on AI techniques of expert systems and natural language processing to facilitate interaction with both intermediate- and end-users. Expertise is expressed in rules that take the form of English sentences. The flexibility of the AI approach allowed us to benefit substantially from an earlier related system for admission of transfer students. A key feature with respect to the workplace is the system's transparency or non-disruptiveness, achieved by using an existing database and providing a special editor for output that looks like a filled-in form of the type that has traditionally been prepared by hand. Transparency and ease of use have, together, enhanced acceptability of the system to the people in the registrar's office, who have collaborated enthusiastically on design and testing. Acceptability should also make possible decentralization to the departments, the university's analog to the government field office.

INTRODUCTION
A significant and labor-intensive task at universities has been the evaluation of student transcripts for graduation requirements. To deal with this task we have developed OTTO, the "Ottomatic" Transcript Translation Organizer, an expert system whose rule representation is in a stylized but recognizable English, patterned after the ROSIE system [7]. The expert system is "loosely coupled" to a database in that it downloads the data it needs and thereafter operates independently of the database. The system is untroubled by the deficiencies of loose coupling noted by Lirov and Ravikumar [6], since the relevant time intervals are long and the selection of data is straightforward. The system downloads all the relevant information associated with a particular student from an existing centralized database, transforms it into a Lisp expression and reorganizes the information according to the requirements of a designated major or program. This capability permits final graduation degree-checking and also supports advising and planning along the way, including "what-if" questions about changing major or specialization. By decentralizing the evaluation process so that it can run on individual personal computers in the departments, accessibility to a large student population is increased dramatically, saving a lot of student and staff time. For information on a mainframe system with related objectives see [1] and [2].

Our experience and results are relevant to AISIIG because the university, whether or not it is formally a government body, as in our case and that of other state universities, resembles a government agency more than it does a commercial enterprise, in that we satisfy regulations rather than pursue profits. More specifically, our system deals with a situation involving a hierarchy of regulations, at the campus-wide, school-wide and departmental levels, which may be compared to levels of government. Furthermore, our service delivery is decentralized, at the department level, which compares to a field office, whereas the repository of official information lies centrally, with the registrar. These characteristics of the environment have had an important influence on the design and development of the system.

Our primary objective during the development of the system was to provide a very easy method for entering requirements for a new degree program and updating the requirements of existing degree programs. Basing the system on microcomputers permits good screen characteristics and potential use in all departments, without sacrificing the power and speed of mainframe systems. OTTO has two general parts, corresponding to two categories of users, neither requiring any knowledge of the internal workings of the system. Users in the first category are students, departmental secretaries or academic advisors who need to gauge the student's progress in meeting requirements. In this environment, the user is given the various options that are available and help is provided on all data that should be entered. In the second category of users are the domain experts such as the registrar's staff, who update the requirements of degree programs. One prime consideration during the development of this second environment was to keep the interaction as simple as possible, without sacrificing the flexibility and the power required to enter the requirements of a variety of degree programs across the academic spectrum.

OTTO uses if-then rules to encode expertise [3]. The only input required is the student's data, comprising name, social security number, address, the courses taken, and the grades obtained in those courses. For our case at George Mason University, this data was obtained from a Hewlett-Packard mainframe. Once this data is on a floppy or hard disk, OTTO converts it into a Lisp expression and can then evaluate it against any degree program.

CURRICULUM CUSTOMIZER AND END USERS
OTTO keys academic requirements by major and catalog year. The requirements and the rules that govern the satisfaction of these requirements are specified by a member of the regis-
such as "humanities" or "literature" courses. Another is used to indicate any course name changes, so that a course taken major, and for customizing the output format so that it clearly reflects those requirements. It is easy to make the output look like the old forms, traditionally filled in by hand, thereby easing the transition and promoting ready acceptance of the new system. The rule editor allows for the creation of the necessary rule base. Accurate assessment of a student's accomplishment is achieved by providing an English-like rule language that is flexible and powerful enough to handle all aspects of the application. A typical degree program requires about thirty rules. It is important that some rules be applied before others, specifically, completed courses should be used to satisfy requirements before electives. The curriculum customizer can achieve this effect by specifying the order of the application of rule sets. This ordering is independent of the layout of the printout form.

OTTO provides the curriculum customizer with some special purpose facilities. One of the facilities makes it easy to specify or update the list of courses that fulfill special definitions, such as "humanities" or "literature" courses. Another is used to indicate any course name changes, so that a course taken under either its old or new name can be properly applied. A third special facility gives the curriculum customizer the ability to copy and rename either the printout form or the rules, so that once the requirements for one major have been defined, the academic programs of similar majors can be specified with only a limited amount of editing. This facility minimizes the work needed to do year-to-year updating.

Once the registrar's staff has made up the output forms and corresponding rule sets of each major and year, the files can be given to the respective departments, or academic advisors on a floppy disk, along with the data of the students in that department. This is the concept behind decentralization of the advising and evaluation process. Every department can have the OTTO system, the data of the students in the department, and the degree requirements on the hard disk of a microcomputer. The student (or a departmental secretary, for security) would enter his or her social security number, the degree program and the catalog year and get a printed form listing the courses required for that degree program, along with the courses still needed for graduation. This decentralization relieves the registrar's office of the drudgery of evaluating each student individually. OTTO allows students to test the effects of graduating under a different catalog year or under a different major by providing facilities that allow for the running of a single transcript against various programs and catalog years. The system allows the registrar to run multiple transcripts under a single academic program and year, so that degree candidates that have deficiencies will be identified easily.

**SYSTEM MODULES**

The OTTO system [4] consists of four major modules: the inference engine, the rule editor, the form editor and the interface. We now describe the role of each module and the human-computer interface issues.

**THE INFERENCE ENGINE**

OTTO rules run on GenlSys (Generic Information System), a substantially revised version of GESBT, a forward-chaining rule-based inference engine built by SAIC as part of a collaborative project with the authors. GenlSys views a knowledge base as a collection of objects and rules that, once activated, continues inferring new knowledge based on the rules and objects present, until nothing new can be inferred. For OTTO, the rules of the knowledge base are those specified by the curriculum customizer. After some preprocessing, to eliminate repeated courses and to mark courses that have undergone name changes, the courses listed on the student's transcript become the objects of the knowledge base. As the inference engine cycles through the rules, only those rules in which the courses taken by the student satisfy the requirements in the rule antecedent fire. The desired consequent, in most cases, involves decreasing the hours still needed, increasing the hours fulfilled and destroying the object, so that the course it represents cannot be used to satisfy other requirements.

In evaluating a student transcript, completed courses must be used to satisfy the most restrictive requirements before being used to meet more general ones. For example, English 111 satisfies both the English 111 requirement and that for a non-science elective. GenlSys fires rules in the order in which they appear in the activated knowledge base. It also allows for more than one knowledge base to exist, though only one is running at a time. This latter facility was used to permit the curriculum customizer group the rules into rule sets, and to specify the order in which these rule sets are activated. By ordering rule sets from specific to general, and not allowing courses to satisfy the antecedent of more than one rule in any one rule set, the curriculum customizer no longer need be concerned with the order of rules within each set. When evaluation begins, the activated knowledge base consists of the rule set consisting of the most restrictive rules, and the courses taken, represented as objects. When this knowledge base terminates, a new knowledge base is created, consisting of the next most specific rule set, and the objects (courses) still remaining in the first knowledge base. Processing continues in this manner until all rule sets have been activated.

**THE RULES AND THE RULE EDITOR**

The curriculum customizer uses the rule editor in order to create the requirements for a degree program, in the form of a rule base. This module, then converts the rules, or revisions to them, in the form of a list, for storage. The editor presents a menu from which a choice can be made about creating, viewing, editing, deleting, inserting or printing the rules. Within these options there are sub-menus to facilitate further processing.

The rule language, shown in figure 1, is a stylized subset of English. This language was developed with the active help of the staff from the registrar's office, in order to use familiar terms. Each rule consists of an antecedent and a consequent. To specify the desired processing sequence for courses that satisfy different kinds of requirements, (ex. specific courses in the major department, then elective in the major, then related
ANTECEDENTS

if dept is [Span]
  one of [Span, Engl, Fren]
  not one of [Span, Engl, Fren]
  not [Span]

If a [lit] course
  and numb is
    one of [101, 103]
    not [302]
    > [117]
    > [302]
  and numbers are
    both of 101, 102
  and grade is [ABC]
  and [Span-ctr] is > [12]
  < [0]
  = [6]
  and decide okmaj
  and need of [ ] is > [ ]

CONSEQUENCES

then
  when [okmaj] okfor [Iic].
  okfor [IIIb].
  and increase [Span-ctr] by [3].
  and decrease [Fren-ctr] by [3].

then forget.

Figure 1: The Rule Language

EXAMPLE 1:
IF DEPT IS CS
  AND NUMB IS >= 300
  AND NUMB IS < 600
  AND DECIDE OKMAJ
 THEN OKFOR IVG.

EXAMPLE 2:
IF DEPT IS ENGL
  AND NUMB IS ONE OF 102, 302
 THEN OKFOR IA.

EXAMPLE 3:
IF NOT ACTIVITY COURSE
 THEN OKFOR ELECTIVES.

Figure 2: Examples of Rules. The first rule determines whether a course is "OKFOR" (can be counted toward) requirement IV-G on the department's standard form. If it is a CS course numbered 300 up to 600, the system must "decide" if the grade is "OKFOR" the standards of the "major". The three examples show three different ways to specify sets of courses.
electives, etc.) rules pertaining to each kind of requirement have to be placed in different rule sets. These divisions are made by making a list of courses that have to be taken under each of these categories, depending upon the requirements of degree program. The making of these divisions was not at all difficult for our curriculum customizer, and in fact, during testing, she would spot a wrong grouping right-away. Each of these major divisions is labeled as a roman numeral, and the courses under that division are labelled alphabetically. The rules then express the courses that meet the requirements of each division.

The rule language uses phrases which are short and meaningful to the curriculum customizer, like those of the form "decide X", which can appear as antecedents. One of these permits the curriculum customizer to specify how the system should "decide" if a certain D grade should count for a certain requirement. At execution time, the following steps occur: the course is applied if the grade is A, B or C; if the grade is a D, a check is made to determine if the number of unused allowable D grades in the major is greater than zero; so, the course is permitted to fulfill the given requirement, and the D grade counter for the major is decremented; otherwise, the course remains in the database for application elsewhere (figure 2).
OTTO DEGREE CHECKER

Do you want to:

Create / Edit / View Printout Form
Create / Edit / View Rule Base
Copy Printout Form under a different Major/Emphasis/Year
Copy Rule Base under a different Major/Emphasis/Year
Run OTTO DEGREE CHECK
Return to DOS

Use arrow keys to highlight desired function
Press ENTER to execute

Figure 4: Main Menu of OTTO Degree Checker

Since the rule language is very small, it was proposed that the rule be written with the help of a menu providing the language words. This menu would reduce the typing of rules and prevent illegal rules due to spelling mistakes. But it was found that typing out the entire rule was faster, and was much more comfortable to our curriculum customizers and so the idea of using menus was abandoned.

With this rule language, we were able to express the requirements of the several departments we examined at George Mason University and Northern Virginia Community College. We therefore believe it would be possible to express various requirements of other universities with few or no changes. The curriculum customizer cannot change the rule language.

THE FORM EDITOR

The form editor allows the curriculum customizer to express the number of credits required in each division, such as major requirements, electives, etc., for every degree program. Using this full-screen editor, the courses can be listed into a very elegant, easy-to-read form (figure 3). It is this form that a student sees as the output of the system, listing the credits he is yet to take, and any other comments or notes. Even here, a menu with a very comprehensive set of choices is provided, such as create a form, edit a form, print a form, etc, with subsequent menus for each choice.

THE INTERFACE

Almost all the interaction with the system is through menus, and so typing is kept to a minimum, except when inserting rules. Each of the menus has a line at the top which tells the path that led to that menu. There is a line at the bottom, explaining the choices (figure 4). From every menu, one of the choices always leads to the previous menu, in case the user decides not to do anything. In addition to the interfaces of the rule editor and the form editor, the system provides other menus for copying one form into another, or copying one rule-base into another, or changing course names or numbers. The use of menus, and appropriate error messages makes the system very easy to use by a naive user. It also makes the system very robust, and not subject to crashing. Even during rule editing, if a spelling error is made, the system advises about the error and the user is allowed to correct it. Also, uniformity among the menus is maintained by using the same keys for the same operation.

FLEXIBILITY

OTTO permits retention of the powerful existing mainframe system, with regard to the variations in the degree programs that it can handle, but at the same time, it gives the convenience of operation in a microcomputer environment. There is no doubt that the system is slower than a mainframe-based system, but by using efficient memory management methods
and faster disk-drives, the difference in speed is hardly noticeable. Decentralizing the advising and the evaluation of a student’s requirements saves a lot of student time and many students can use the system at the same time. Some of the capabilities of the system in handling the variations in various degree programs are:

a) It can handle all degree programs, irrespective of the catalog year.
b) It can handle changes in course names or numbers over the years, so that credit for the course under previous name can be counted towards the current name.
c) GPA requirements for individual majors, and the number of D grades allowed by the University or a department can be easily checked for.
d) Each course is allotted to the requirement where it can do the "most good".
e) The system can be run in batch mode so that the department can send an evaluation sheet to every student at the end of the semester.
f) The system allows grouping of a whole set of courses under a particular name (such as Humanities) so that any course from that set would count for a particular requirement.
g) The system permits checking to see if both the course and a lab attached to it are taken together.

Usage of non-standard lisp functions [5] is kept to a minimum, and hence the system can be transferred to another computer system with little or no changes. The student data required has to be in an ascii file which can be generated by any computer system.

CONCLUSION

OTTO is a complete expert system, written in GC Lisp, installed on a 20 megabyte hard disk IBM-PC, in a beta-test phase at the registrar’s office at George Mason University. Employees in the registrar’s office who are not computer programmers, quickly learned the role of curriculum customizer using the system to build rules and output forms. Students who had never used a computer system before were able to go through the system and get their evaluations done without explanation. Future work under consideration is to provide more help through pop-up menus. Also, hooking up to a mainframe, to get the student data or the rule bases is under consideration.

REFERENCES