# A Business-Rules Approach for Departmental Advising

**Lisa Burnell, Ryan Figg, Minh Phan, Justin Smith, Akihiro Sugihara**
**Computer Science Department, Texas Christian University, Fort Worth, Texas 76129**
**L.Burnell@tcu.edu, Phone (817) 257-7166; Fax (817) 2567-6188**

**John Priest**
**Industrial and Manufacturing Systems Engineering**
**University of Texas, Arlington TX 76019**
**jpriest@uta.edu, Phone (817) 272-3168, Fax: (817) 272-3406**

**John R. Durrett**
**Area of Information Systems & Quantitative Sciences, MS 2101,Texas Tech University, Lubbock, TX, 79409 USA,**
**john@durrett.org, Phone (806) 742-2994, Fax: (806) 742-3193**

**Abstract**

The advising process plays an essential role in the success of a student's academic life, and should include mentoring and counseling offered by faculty advisors. Unfortunately, the limited advising time available is often spent on routine tasks such as course advising and approvals. In addition, faculty advisors spend a large proportion of advising time answering the same questions for multiple students. Non-traditional students and distance education make it more difficult for student and faculty advisor to find an agreeable time to meet. While universities have implemented web-based advisor systems for routine tasks, most of these systems make static, non-personalized materials such as catalogs available and offer email communication between student and advisor. Moving beyond this approach, we seek to provide a means by which a student may engage in interactive dialogues with a "Virtual Advisor" that possesses current, reliable knowledge and offers personalized advice. VAT (Virtual Advisor Technology), is an intelligent knowledge-based system being developed as a multi-university, multi-departmental team design project. The intent of this online advising system is to assist students in planning semester schedules and degree plans. VAT uses rule-based reasoning to generate degree plans, check scheduling conflicts, and recommend personalized advice based on student preferences.

## Introduction

One of the most effective techniques for improving student learning and quality of educational experience is high-quality personal academic advising by faculty. Students with good mentoring achieve higher grades, are less likely to drop out, develop more ethical reasoning skills, and have more positive attitudes towards learning and to their institutions (Kramer 1995). However this personal advising entails a substantial time commitment and responsibility on the part of faculty. Too much time is spent on routine and mundane tasks rather than the mentoring and strategic planning tasks that could better help the student.

Under the traditional faculty-based advising system, the student also has responsibilities. Students should make an effort to get to know their advisor. They have to maintain their academic advising and career planning files. Every student is expected to know the degree requirements and other relevant academic policies and procedures. From this information they want to develop and evaluate different degree plans and schedules given the student's personal and unique preferences and requirements.

Advising season is an extremely busy part of the semester for faculty advisors, as well as students. Not to mention the flood of e-mail and phone calls from clueless students, just try to imagine the trouble an advisor has to go through to fit in his or her schedule more than 30 assigned advising sessions. One solution is to have an intelligent online advising system, where the students can complete the advising process and have the system generate a valid degree plan and schedule. This would save faculty advisors time, and it would save the students several trips to the faculty advisor 's office.

Currently several colleges and universities have made an effort to have an online advising system. A majority of these online advising systems employ a straightforward strategy of converting static paper-based advising materials to web-based electronic form. The term "virtual advisor" is applied to any online resource that aids a student in selecting classes or degree plans. While this is undoubtedly useful, it does not substitute for the advising process because it cannot account for student's preferences or give personalized advice. We seek to provide a means by which a student may engage in interactive dialogues with a "Virtual Advisor". Students want to quickly get answers to specific questions such as what courses are available next semester, what courses he or she is required to take, or when courses should be taken.

## Departmental Degree Planning

In this section, we outline some of the major challenges inherent in the advising domain and outline specific requirements that led us to the design decision to use a business-rules approach to representing dynamic and complex policies. Consider a typical student shown in the scenario below.

***Advisor Scenario***

*Elton wants to graduate as quickly as possible, subject to the realities of his specific situation. He can only take on-line or night classes, since he works full time during the day. In the summer, he could take a*

*morning class, but he'd prefer not to. He figures that if the classes aren't too hard, he could take 15 hours a semester. If he has one or more tough or really time-consuming classes, he better stick to 9 or 12. He wants to get a COSC degree, but if he can graduate much sooner, he'd like to consider a CISC degree. Before making a decision to switch majors, he'd like to better understand what the differences between the two degrees are — specifically, what kinds of jobs he can get, where those jobs are, and how much they pay.*

A primary design requirement is to cope with dynamic, complex rules that vary widely across departments and universities and over time. It must be easy to adapt to specific and evolving requirements within and across universities. For example, it should be straightforward for a large MIS department faculty advisor to customize the system to fit their requirements, even though the system is being initially designed for small computer science department requirements. Some rules, data and processes are common across departments, colleges or universities, while others are unique and dynamic (rapidly changing). The system needs to be designed to account for this reality. Some challenging domain knowledge is shown below.

- Course numbers that change (e.g. CSE 1320 was known as 141 2 years ago)
- Catalog changes, including changes to general education courses required for all majors.
- Degree requirements that have "one of" and "two from" course requirements: for example, fine arts electives (from Music, Art, etc.), or 2 electives selected from a list of EE, CSE, Math or Physics courses.
- Co-requisites, for example, a student can take Calculus I and Physics I together, since Calculus I is a co-requisite for Physics I. Consider that a student signs up for these courses, then drops Calculus I and now wants to take Physics II (that has Physics I as a prerequisite). What's the right advice? Does it depend on the department or the student?
- Variation in defining what constitutes a "passing grade". At one college, "lower division" courses must be passed with a C or better, while other courses just need a D or better. Some departments, or several within a college or school, have many such rules. What is even worse is that the rules and sometimes even the terminology change!
- Departments that have policies regarding admission to a professional program, in which students must complete certain lower-level courses and have a certain grades (calculated 3 different ways) before being admitted to the upper division (and allowed to take those courses). In one department, students were allowed in 3000 level courses before they were admitted to upper division, but generally denied requests for 4000 level courses until they were admitted to upper division. So it was common practice to override stated policy.

User requirements for the advisor include the following selectable tasks by the student

- Preferred graduation date – user selected
- Absolute graduation date – user selected
- System suggested graduation date based on user selected number of hours, schedule and other limitations
- Quickest or shortest path of the fewest number of semesters to graduate
- Critical path of courses to meet graduation goal (i.e. because of prerequisites and schedule these courses must be taken in this order or the graduation date will be postponed)
- Absolute schedule limitations by semester/all (no exceptions) – max. number of hours, evening, day, web-based, no class before 9:30, no class after 5:30, etc.
- Preferred schedule limitations by semester/all (exceptions allowed to meet other requirements such as graduation date, exceptions are presented to user)
- Able to average course difficulty rating over the remaining semesters
- Set a maximum difficulty rating to not exceed
- Instructor preferences or list of instructors to avoid
- Select certain classes at certain times/semester and system then develops the plan around these selections

## Why not use a DBMS solution?

It is easy enough to represent that EE 1301 is a prerequisite for EE 1302, but as shown earlier, the policies for who can take a course become much more complex. A relational database is a good repository for basic data downloaded from other systems – student transcripts, course catalog descriptions, etc. Our goal is to capture in the DB that portion of the student advising ontology that is static– schema definitions should be generally stable.

Triggers and other stored procedures are good solutions if the knowledge encoded by the procedures is static. Student advising tasks are not. We believe this is one of the reasons why a colleague of one the authors described the task attempted by this system as "impossible". Past experience at two of the authors' schools have shown that attempting to build departmental advising support systems that rely on capturing advising knowledge in program code and relational databases are impractical to maintain. Over time, the systems become less useful, because they encode old rules and no one has time to try to maintain the system. In one large (400+ student) department, attempts have been made to redesign the advising system from time to time. In this system, there is no centralized way to make changes, and the method is

unorganized and tedious. With VAT, changes will be centralized and systematic.

## VAT

VAT (Virtual Advisor Technology) is an expert system designed to be a complete online departmental advising solution that can fit within a web-based knowledge portal. VAT is intended to deliver to students such interactive services as degree planning, course approval, transfer analysis, and providing general information regarding types of degree plans and market outlook that could help students have a better understanding and be more interested in their major. An agent based approach was chosen that currently uses the Jess rule based expert system shell, Java, and Oracle. The key architectural feature is the clean separation of domain knowledge from processing and display functions, a feature argued for in the business rules community (e.g., Grosof, et al. 2000 and Seiler 1999). Additional architectural decisions are driven by the CT approach to object and agent oriented design (Durrett, Burnell and Priest 2001).

The student is the primary user of VAT. The student will be able to select classes based on their individual needs, both for the coming semester and in the long term. The student's record and personal information will be accessible by the Virtual Advisor, allowing the system to recommend classes that will best fit into the student's plans. Other options available to the student include review of courses taken and GPA calculations and schedule reorganization based on time constraints. All of these options are available to existing students as well as incoming students and transfers.

VAT is designed to ease the process of the traditional faculty advising system, but not to completely replace it. Faculty advisors still have a major role to play in the VAT system. The department chair will be responsible for entering class sections into the system, approving certain student requests, and maintaining the graduation requirements for each degree plan. Local databases store the data needed by the Virtual Advisor system. Students, faculty advisors, and the department chair (or his appointed system administrator) will be able to interact with the system without having to be knowledgeable about the database structure.

## Design of the Business Rules for the Advisor

Domain knowledge was obtained from published material and interviews with faculty advisors. The knowledge was first separated by source, i.e. a university, college or departmental policy. This was then analyzed to identify policies regarding the same topic to determine a conflict resolution strategy. Our analysis indicates that a simple policy of checking from smallest unit (department) to largest (university) is adequate. Other resolution strategies are possible.

The VAT architecture (Figure 1) is primarily composed of three major modules: The user interface module, the degree planner module, and the degree plan modifier module. In addition a Connection Module handles how the VAT system connects to different types of databases supplied by the university registration database system.

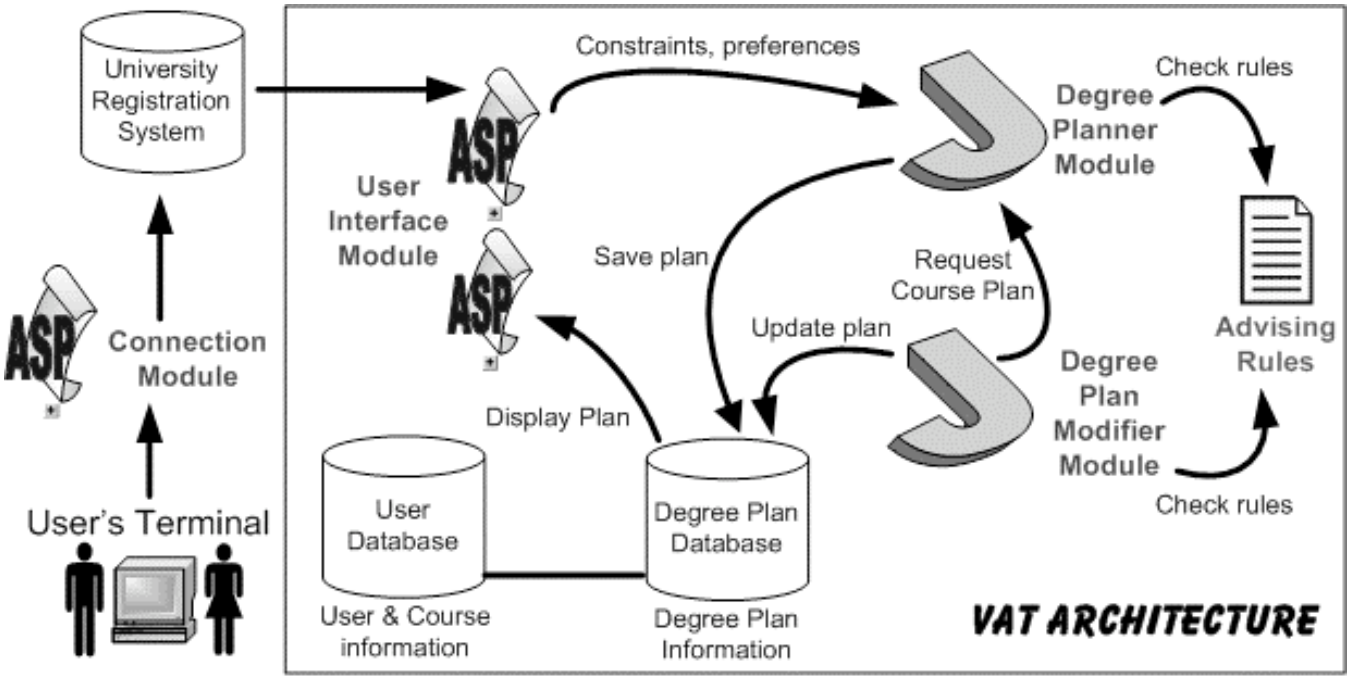


**Figure 1: VAT Architecture:** The diagram of the VAT architecture shows the data flow between the primary modules. The program input is supplied by the user. Not shown is the planned connection to external systems, such as University-wide registration systems.

## The Degree Planning Module

This module uses business rules and user specified constraints to generate a valid degree plan from the current semester until the student graduates. The Degree Planning Module primarily handles data from various sources and feeds it to the other modules. Core requirements, course lists, and university policies are required to generate a degree plan the student desires. The student can alter the degree plan and submit it for requirement compliance and practicality. This module could be ported to other advisor systems in case the VAT interface does not meet the requirements of the university where the system is being implemented. Likewise, the department specific knowledge, encoded as rules, is localized within this module, easing the task of updating rules or moving the system to other departments. An simplified example rule in Jess is shown below

```
;; Generate a list of all the PASSED courses in a student's
transcript
;; Passing grades are department selectable
(defrule passed-course-list ""
   (goal-is-to (action get-passed-courses) (studentID ?ID))
   (courses (studentID ?ID) (course-list $?passed))
   (transcript ?ID ?course ?semester ?year ?grade)
   (test (member$ ?grade $?passing-grades))
   (test (not (member$ ?course $?passed)))
=> ;; add valid course to list
   (assert (courses  (studentID ?ID)
        (course-list $?newpassedlist))))
```

## The Interface Module

The VAT Interface Module (Figure 2) is responsible for getting user inputs necessary for generating a valid degree plan or schedule. Such inputs include user preferences and constraints such as time of class, type of class, number of hours willing to take per semester, which semester to start, etc. All of the user's inputs are currently passed to the Java servlet Degree Planner Module. The Interface Module displays requested information such as degree plans, semester schedule, class history, and GPA computation. The interface module also makes paper-based advising materials available as a reference for students. Another task of the Interface module is to handle the security of the system. The user is required to log in before he or she can access any personal information.

## The Degree Plan Modifier Module

The Degree Plan Modifier Module is an equally important part of the VAT system. Since this module pulls information from the Degree Planner Module, it is only accessible to the student when there is a degree plan for that student in the system. The degree plan modifier module allows the student to edit the placement of courses. It requests the available course plan information from the Degree Planner and does its own background processing to check for the validity of the plan before saving it to the system.

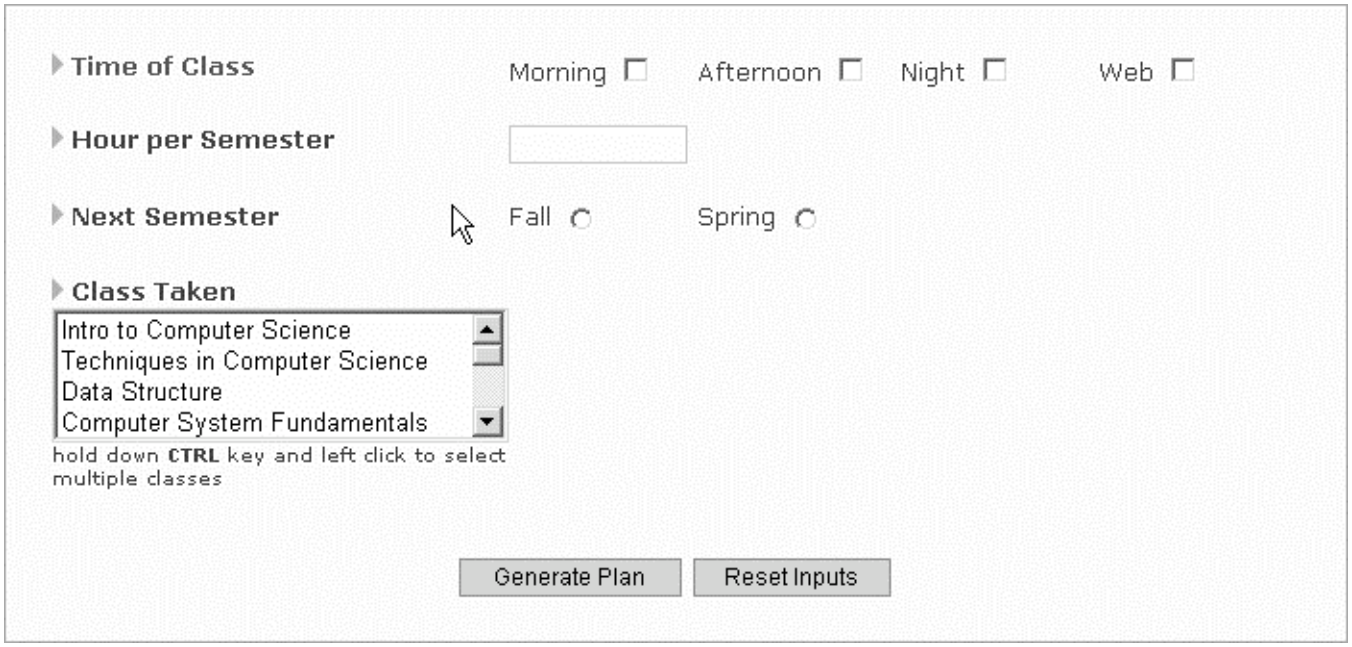


**Figure 2: VAT User Preferences:** Designed to be a utility-based agent, the VAT system takes into consideration the "happiness" of the user. Therefore, the system will try to generate a schedule that closely matches the constraints and preferences specified by the user.

## Conclusions and Future Work

The initial system was designed for three different departments (Computer Science, Management Information system and Industrial Engineering) located at three different Universities. We are making maintainability a primary goal of this system. VAT will be easy to adapt to specific and evolving requirements within and across universities. It should be straightforward for advisors from other universities to customize the system to fit their requirements by directly inputting their rules and requirements. The system will be able to handle data and processes that are specific to the department, college or university.

Testing will include a selection of potential users at the universities where VAT will initially be deployed. Selected faculty advisors will populate the database with a selection of courses and make sure the system is capable of expressing the needed knowledge, such as course times and prerequisites. Students using VAT for semester scheduling and degree planning advise will provide feedback on all parts of both processes. Department chairs will monitor the enrollment process and make sure no errors such as over enrollment or failure to meet prerequisites occur. All will be asked to give their opinions about the system by keeping a log of their experiences and how the process is better or worse than the classic advising process.

We plan to develop metrics for identifying and measuring the level of change for different rules, policies, and requirements. An example is to count the number of times an item changed over 5 years for a given department. This metric would allow us then to evaluate the ability of our initial design to adapt to change. We will then develop a contingency-based design to improve adaptability and then quantify how different items are affected by the "contingency design" (Durrett, Burnell and Priest 2002).

Detailed use cases and model analysis have provided a better understanding of how limited current online advising systems are and how to significantly enhance existing systems. VAT can help the traditional advising process by eliminating many tedious tasks and responsibilities, but it is by no means a complete replacement for the faculty advisor. Students are still highly encouraged to contact faculty advisors for their intellectual and personal growth. With VAT, there should be more time for such activities.

## References


Durrett, J. R.; Burnell L.J.; Priest, J.W. 2002. A Smart Agent-based Resource for Virtual Advising. In Proceedings of the 2002 International Resources Management Association International Conference Issues and Trends of Information Technology Management in Contemporary Organizations, Forthcoming.

Durrett, J. R.; Burnell L.J.; Priest, J.W. 2001. Organizational Theoretic Metaphors for Software Architectures, In the *10th Annual Industrial Engineering Research Conference*, Dallas, Texas. May 20-22.

Kramer, G.L. 1995. Redefining Faculty Roles for Faculty Advising. *Reaffirming the Role of Faculty in Academic Advising*, Monograph Series Number 1: 3-9

Grosof, B.; Rouvellou, I.; Degenaro, L.; Chan, H.; Rasmus, K.; Ehnebuske, D.; and McKee, B. 2000. Combining Different Business Rules Technologies, A Rationalization. In the Proceedings of the OOPSLA 2000 Workshop on Best-practices in Business Rule Design and Implementation, October 15, 2000.

Seiler, H. 1999. Managed Business Rules: A Repository-Based Approach. *PCAI* July/August, 16-19.