GRADUATE SCHOOL –
PLAN OF STUDY
TECHNIQUE

CONCEPTION, DESIGN, IMPLEMENTATION AND
USAGE

SANKAR G. VYAKARANAM
MASTER’S PROJECT
DEPARTMENT OF COMPUTER SCIENCE AND SOFTWARE ENGINEERING
MAY 2001
GS-POS\textit{t}
\textbf{(Graduate School - Plan Of Study \textit{Technique})}

\textbf{Abstract}

Presently, graduate students at Auburn University have the ability to fill out their Plans of Study on the worldwide web by choosing appropriate forms from the Graduate School Homepage. However, after filling the form, the students are required to take a printout of a ‘pretty-format’ page and walk it to the different professors and the Graduate School for the approval process. GS-POS\textit{t} is a project funded by the Graduate School of Auburn University, which addresses this aspect of graduate study. The goal of GS-POS\textit{t} is to make both the filling out and filing of Plans of Study online and hence, paperless. The project was started in November 2000. It was quickly identified to be a standard 3-Tier Web-Application with certain elaborate and complex requirements specifications. The backend is an RDBMS database (SQL Server 7.0), the frontend is the Plan of Study Forms residing on the Web-browser (HTML & Javascript) and the middle tier, which implements most of the business logic is chosen to be Active Server Pages technology.
A software methodology was used to help organize and implement the project. Incremental Model with replacement increments is being used. The frontend, the backend, the middle-tier, and the Database were separately designed after elaborately discussing the desired functionalities of GS-POS\textit{t}.
The first increment is the backbone structure of the web application with just the basic functions. It included designing the database, setting up forms to walk through the entire process of filling out a Plan of Study and the database connection that enables inserting, updating and deleting of student records. ER modeling was done to design the database, to improve normalization of data and reduce redundancies and other anomalies. An OLE-DB connection mechanism was used to connect the SQL Server and the \texttt{.asp} pages, for fast, optimal connection. UML diagrams (use case diagrams, more specifically) were also constructed to identify ‘typical usage scenarios’ of the application, using a free version of Rational Rose 2000. A mechanism to generate a unique Plan of Study ID for every student in a given department and with a given degree is also included in this increment.

Another increment deals with Forms Design and User Input Validation. The forms are intuitively designed and the main Plan of Study form is expandable to accommodate an unusually large number of courses, if required. The data being entered is checked against ‘valid data’ and any discrepancies are alerted back to the user.

The third increment essentially deals with the flow of control and the access restrictions of the Plan of Study document. Complex Business Logic was implemented as SQL Stored Procedures and appropriate Componentization. Elaborate flow charts were drawn and the design was discussed numerous times and optimized to maximize performance, usability and back-end and database efficiency.

The project was developed keeping in mind scalability, extensibility and browser compatibility issues. The overall design, coupled with \texttt{ASP} and \texttt{SQL Server} technologies, facilitates future modifications as requirements change.
ACKNOWLEDGEMENTS

I would like to express my gratitude to the following people, who have helped me in finishing this project.

- Dr. Stephen MacFarland
- Dr. Homer Carlisle
- Jayasree Jaganathan
- Donna Tyson
- Amol Shambharkar
- Sree Balaji
- Srinivasan Iyer
- Muneer Yachb
GS-POSt

(Graduate School - Plan Of Study Technique)

INTRODUCTION

GS-POSt is a 3-tier web application. The basic architecture is as follows:

**GS-POSt Architecture**

*(A 3-Tier Web Application)*

![Diagram showing the architecture of GS-POSt]

The goal of GS-POSt is to make both the filling out and filing of Plans of Study of
Graduate Students at Auburn University online, and hence paperless. The project started in November of 2000 in the form of discussions that identified the elaborate requirements specifications. The starting point was a documented account of the salient features of the web-application. The information that needs to be collected and the order in which the information shall be collected were identified next. Then, the process of approval (or otherwise) and the process of revision were broken into a series of sub-steps. Access restrictions and permissions upon proper authentication were then laid out.

SQL Server 7.0, an enterprise-level database, was consciously chosen over MSAccess for scalability reasons. Scalability was also considered while coding the HTML forms. VBScript [Ref. 1] and Javascript [Ref. 2,3] Arrays were extensively used to give flexibility for future changes/modifications.

**Organization of this Project Report:**

- Preliminary modeling using the Unified Modeling Language is explained.
- The design of the Database is explained. Discussion includes important tables created and stored procedures.
- The design of the Middle-Tier is explained. Important background daemon programs constitute the bulk of this tier.
- The Frontend design is explained. Intuitiveness and robustness of the forms are discussed.
- Other important aspects of the project such as Scalability, Extensibility, Usability and security are discussed here.
Finally, the concluding paragraph talks about future developments that could be pursued.

Appendices include Database diagrams, UML diagrams, and flowcharts that illustrate the design of Frontend and Middle-tier logic.

**UML DESIGN**

UML [Ref. 4, 5] is a de-facto industry standard for modeling Software Artifacts/projects. An evaluation copy of Rational Rose Enterprise Edition 2000 was downloaded from the Rational web site [www.rational.com] and the UML Modeler of the Rational Rose Software was used to do some basic UML modeling of GS-POS1. GS-POS1 is a web-application as opposed to Object-Oriented Software Application. Its implementation is object-based, and not object-oriented. Hence Class Diagrams and subsequent Class Interaction diagrams were deemed not directly relevant to this project. However, Use-Case Diagrams [Ref. 4, 5] are extremely useful in further identifying the requirement specifications of GS-POS1 and modeling the design of the project. Furthermore, use-case diagrams identify the typical user usage scenarios of GS-POST and hence, can be used for functional testing of the final implementation. Appendix 2 contains certain important Use-Case Diagrams generated for this project.
Use-case Modeling

System behavior is documented in a use case model that illustrates the system’s intended functions (use cases), its surroundings (actors), and the relationships between the use cases and actors (use case diagrams).

Use cases drive the whole development process. They describe the typical interaction that a user has with the system in order to achieve some goal. Each 'use case' should indicate a function that the user can understand and that has value for that user. The use case model starts in the Inception Phase with the identification of actors and the principal use cases for the system. The model is then matured in the Elaboration Phase.

Use cases represent the functionality provided by the system. They model a dialogue between the actor and the system.

Each use case contains a flow of events, which is a description of the events needed to accomplish the use case functionality. The flow of events is written in terms of what the system should do, not how the system does it. A use case diagram is a graphical representation of some or all of its actors, use cases, and their interactions for a system.

Use Cases Identified

❖ Saving a Plan of Study: The student has a place to save his/her Plan of Study on the server. This Use Case depicts that functionality of the system.

❖ Submitting a Plan of Study: Submitting the Plan of Study by the Student will trigger the act of passing the Plan of Study to the Committee members for
inspection.

- **Revising a Plan of Study**: The student's plan of study is not approved for whatever reason after he/she submits it. Then he/she executes this Use Case scenario, and resubmits the modified Plan of Study.

- **Graduate Advisor Approval**: The graduate advisor (Major Professor) logs in and acts on a plan of study by clicking the approval button, before possibly editing it.

- **Graduate Advisor Removal**: The graduate advisor (Major Professor) logs in and acts on a plan of study by clicking the reject button.

- **Graduate Program Officer Approval**: The Graduate Program Officer has the permissions to edit a Plan of Study before approving it. This use case depicts that functionality.

- **Graduate Program Officer Rejection**: Likewise, the Graduate Program Officer may choose to reject a Plan of Study.

- **Committee Approval**: A Faculty Member participates in this Use Case wherein he/she inspects the Plan of Study and approves it. A Committee Member may not edit a Plan of Study.

- **Committee Rejection**: As with others, this use case is self-explanatory. Basically, the professor has some reservations about a student's plan of study and does not approve it. The student is informed and has to make appropriate changes.

- **Department Head Approval**: The department head also may not edit a Plan of Study. This use-case represents the approval process of the Plan of Study by the department head.

- **Department Head Rejection**: The department head clicks on the reject button of
the Plan of Study he/she is viewing.

- **Graduate School Approval:** Graduate school is the last authority to act on a Plan of Study. If this Use Case is successfully executed, then the Students Plan of Study is 'on file'.

- **Graduate School Rejection:** For whatever reason, the Graduate School might not approve the Student's Plan of Study. The student is informed and has to make appropriate changes.

**Actors Modeling**

Actors are not part of the system. They represent anyone or anything that must interact with the system under development. An Actor is a role that a user plays. An actor can be a person or an external system.

**Actors Identified**

- **Student:** A student interacts with the GS-POSt system by entering his Plan of Study and either saving it or submitting it. The student cannot edit the Plan of Study after submitting it, until any final action has been taken on it. The student can however consult his/her Chair in order to make changes to the Plan of study while its being reviewed. This interaction between the student and the graduate advisor is in fact encouraged.

- **Chair of the Student’s Committee:** The chair of the Student’s committee (Major
Professor) of the student plays a very important role. This authority has the ability to edit the Plan of Study once it reaches him/her. The chair contacts the student for making changes to the Plan of Study or informs the student after any changes to the Plan of Study have been made.

❖ **Graduate Program Officer**: Every department at Auburn University has a faculty member under this role. A Graduate Program Officer has access and editing permissions on all Students' Plans of Study under his/her department.

❖ **Committee Members**: These authorities do not have the facility to edit a Plan of Study that reaches them. They can only either approve it or reject it or they can always consult the student it they want to impose any conditions on their approval process. The Plan of Study reaches these authorities after it passes through the Chair of the Student’s Committee (Graduate Advisor or Major Professor) and the Graduate Program Officer.

❖ **Department Head**: The authority to which a Plan of Study would go to for review before it is sent to the Graduate School, is the Department Head. Department Head’s signature is required on every Plan of Study of that respective department. However, a department head may not edit the Plan of Study.

❖ **Graduate School Authority**: Graduate School is where the Plan of Study finally reaches after approval from all the Committee Members. The Graduate School can again either approve or reject the Plan of Study for whatever reasons and let the department know about it. The Graduate School always has the authority to edit/make changes to a Plan of Study at any given point of time.
The System in Action

A static model describes the elements of the system and their relationships, whereas a dynamic model describes the behavior of the system over time. The use case model describes the interaction of the system with the surroundings where as the static model describes the internal elements that make up the system. Hence, Plan of Study is a system component that, with respect to the static model, describes the document as a whole that is required for a student to submit to the graduate school, which files it upon the completion of the approval process.

A process view is viewing the model from a different axes where in a particular sequence of events (process elements) are modeled to indicate a likely or desired process flow. For instance, over a period of time, the Plan of Study can be thought of as undergoing a process transformation, which takes place in a well-defined order.

A Process View:

- Student *edits* the Plan of Study.
- Student *saves* the Plan of Study.
- Student *submits* the Plan of Study.
- The Plan of Study gets reviewed by the Graduate Advisor (Major Professor) and is either *edited* or *approved* or *rejected*.
- The Plan of Study gets reviewed by the Graduate Program Officer of that particular department and is either *edited* or *approved* or *rejected*. 
The Plan of Study gets reviewed by the rest of the Committee and is either approved or rejected.

The Plan of Study gets reviewed by the department head and is either approved or rejected.

The Plan of Study gets reviewed by the Graduate School Authority and is either edited or approved or rejected.

The Plan of Study is filed at the Graduate School and is good, and changing its state is only possible by following all of the enlisted steps (process).

The above process view can also be depicted as a State Diagram because it shows how a single object behaves across many use cases. A flow sheet depicting the process view (State Diagram) is illustrated in Appendix 3.

UML is an effective design tool to model software projects and processes. Because of a confluence of ideas, techniques, personalities, and politics, UML promises to become a standard notation for representing the structure of data in the object-oriented community.

DATABASE DESIGN

The database was designed using ER modeling. A student can have more than one plan of study, even though it is rare that a student would simultaneously enroll in two different programs. Hence, every plan of study is unique and has a unique identifier in the
database called POSCode. The main Database diagram that illustrates the relationships between the most important tables of GS-POST is attached as Appendix 1.

The database is appropriately named as GS-POST. It presently consists of 7 tables of which four are initially populated with critical information about the courses at Auburn University. Referential Integrity constraints have been laid out, to avoid foreign key discrepancies. Authentication information is obtained from data that is replicated into the SQL Server database from a master datasource that is periodically updated.

SQL Server 7.0 [Ref. 6] is the database server used. The choice of the database is ideal as it is extensible, high-performance and compatible (with respect to software configuration) with the rest of the software and the platform used. ADO-DB connection was used to connect the SQL Server database to the ASP pages. The connection is known to be faster and more efficient than the ODBC connection [Ref. 7]. Database queries and other operations were written in T-SQL. Transact-SQL is the proprietary language of Microsoft’s SQL Server.

A certain amount of redundancy (de-normalization) is deliberately added when the StudentId, SocialSecurity and StudentLastname values are added to the PlanOfStudy table. This is because it simplifies the logic for certain very frequent database queries. Stored Procedures are extensively used in GS-POST. They are called from within the .asp pages to perform certain repetitive, frequently needed tasks in an efficient manner. After
identifying a list of posCodes, by executing relevant queries, each posCode can be sent as a parameter to appropriate Stored Procedures to extract appropriate information.

Some of the important tables created are briefly described below:

- **PlanOfStudy**: This is the most important table of the database. It contains information about each individual plan of study. PosCode is the name of the column that is the primary key of this table. It is generated as an Identity column in SQL Server 7.0 with a seed of 1 and an increment of 1. Each Plan of Study has a unique identifier. This way, a given Auburn Student can have more than one Plans of Study filed. The data for this table is entered by Student, Major Professor and/or Graduate Program Officer.

- **StudentCourses**: This is the main table that contains all the courses and the details of each course for a given Plan of Study. The data for this table is entered by Student, Major Professor and/or Graduate Program Officer.

- **Committee**: This table holds the information about the Committee for a particular Plan of Study. The data for this table is entered by Student, Major Professor and/or Graduate Program Officer.

- **Degrees**: This table contains information about the degrees that each department at Auburn University offers. The data for this table is entered by the Graduate School Authority.

- **Departments**: This table contains information about what are all the departments
and their codes presently in Auburn University. The data for this table is entered by the Graduate School Authority.

- **RequiredCourses**: For a given department, a given degree might have some required courses as stipulated by Auburn University. This table contains that information. The data for this table is entered by the Graduate School Authority.

- **SubjectCodes**: This is a utility table containing only one column containing the list of all subject codes for all the courses offered by Auburn University. The data for this table is entered by the Graduate School Authority.

- **GID Table**: This is the main User Information table, which contains information about all the students, faculty members and staff of Auburn University along with their names and other details. This table is generated by a different campus organization and is replicated regularly into the SQL Server database that GS-POSSt uses.

T-SQL [Ref. 6] is the proprietary database language of SQL Server. T-SQL Stored Procedures [Ref. 6] are extensively used in this project. In addition to flexible information retrieval, stored procedures offer several other advantages as follows:

- Stored procedures are only parsed once, and an execution plan is created once when the procedure is first run. This can speed operation significantly. Note that you can force a stored procedure to rework the execution plan each time the procedure is run, in case a procedure requires different execution plans based upon the parameters passed in.
- Users can be granted different permissions on stored procedures than the permissions on the underlying tables. Thus, fields in a table that should not be read or modified by most users are not exposed.

- Stored procedures can encapsulate business rules. Thus, if a stored procedure for selecting records for inclusion in some other process changes, only the stored procedure needs to change to select the correct records. This is especially important for systems that use more than a single client application.

- Stored procedures can reduce network use, especially in a Wide Area Network (WAN). Rather than passing a long, ad hoc query string, only the name of the procedure and the parameters needs to be passed over the wire to the server.

Stored procedures are intuitively named in GS-POST. For example getStudentInfo is a stored procedure which takes one input parameter, which is studentId, and returns single record set by fetching information about the student from the StudentInfo, PlanOfStudy, Departments and Degrees table.

**MIDDLE-TIER DESIGN**

The complexities of the requirements specifications are effectively converted to business logic through the implementation of the Middle-Tier using Active Server Pages
Technology [Ref. 8]. The main variable that determines the current status of a particular Plan of Study is called posStatus. posStatus is an integer, and its value is described in yet another table called Status. This variable is incremented to signify the progress of a Plan of Study in its lifecycle (see below). The access levels, such as editing capabilities or view capabilities are determined by the value of posStatus also.

RequestFlag and onFile are two other variables that together with posStatus keep track of the creation and revision process of a Plan of Study. RequestFlag has a scope of a row in the database, where as onFile flag is set on the entire Plan of Study. A requestFlag value of 1 indicates a delete on that row (course or committee member) and a value of 2 indicates that that particular entry is to be added to the current Plan of Study.

Different Stages of a Plan of Study's Lifecycle

- **Stage 5**: The posStatus is given a value of 5 when the student opens a plan of study and is in the process of editing it and saving it.
- **Stage 10**: The posStatus is given a value of 10 when the student submits the plan of study to start the review process on it.
- **Stage 15**: The posStatus is given a value of 15 when the email is sent to the Major professor by a daemon program that runs at periodic intervals of time.
- **Stage 17**: The posStatus is given a value of 17 when a reminder email has to be sent to the major professor because the Major Professor took no action for a certain (long)
period of time – 21 days.

- **Stage 19**: Major Professor rejects the Plan of Study.

- **Stage 20**: The posStatus is given a value of 20 when the Major Professor approves the plan of study, possibly after editing it.

- **Stage 25**: Email is sent to the Graduate Program Officer of that particular department informing him/her that the Plan of Study has passed the Graduate Advisor (Major Professor).

- **Stage 27**: Similar to Stage 17, an email reminding the GPO is sent and the posStatus is set to this value. This happens after every 21 days.

- **Stage 29**: GPO rejects the Plan of Study.

- **Stage 30**: The Graduate Program Officer acts on the Plan of Study and approves it, possibly after editing it.

- **Stage 35**: The posStatus is given a value of 35 when emails are sent to the Committee Members informing that the Major Professor and the Graduate Program Officer have both looked at it and have approved it.

- **Stage 37**: Reminder emails are again sent to the particular Committee members who have not responded to the emails by taking any action on the plan of study, after every 21 days.

- **Stage 39**: Committee Member rejects the Plan of Study.

- **Stage 40**: This stage happens when all the committee members have finished approving the Plan of Study document.

- **Stage 45**: The department head receives an email about the Plan of Study after the GPO approves it.
- **Stage 47**: After every 21 days, a reminder email is sent to the department head.

- **Stage 49**: The department head rejects the Plan of Study.

- **Stage 50**: The department head acts on the Plan of Study with an Approval. At this stage, the Plan of Study becomes visible to the Graduate School.

- **Stage 54**: The Graduate School Authority rejects the Plan of Study.

- **Stage 55**: Graduate School Approves the Plan of Study. Simultaneously when the PosStatus is being changed to 55, the onFile flag in the PlanOfStudy table is also set.

No email is sent to the Graduate School after stage 50. It would be required of the Graduate School Authority (GSA) to log on to GS-POSt and there will be a feature that will provide the GSA with all the Plans of Study that have crossed the reviewing stage of the department head (Status 50). Also, extensive search facilities are provided to the Graduate School Authority to view Plans of Study based on Student ID, Social Security, Student Last Name, Department, posStatus value etc. These search capabilities enable the GSA to track the progress of any Plan of Study in GS-POSt.

The numbers chosen for the values of posStatus do not appear on the front-end for users to see. Also, they don’t have to correspond to an ascending order as a Plan of Study progresses along its lifecycle.

Once a Plan of Study is ‘on File’, the student can yet again revise it. During this revision process, the Plan of Study passes on from the Student to the Major Professor and then to the Graduate School Authority directly. It bypasses the GPO, Committee Members and the Department Head.
Background Daemon Programs:

A daemon is a background program. A daemon is usually entrusted with the responsibility of executing certain repetitive but important tasks at scheduled times. Six daemons have been written in VBScript that do the important tasks of checking periodically the posStatus values and updating them after acting on them.

- **studentActs.vbs**: This daemon changes the PosStatus value from 10 to 15 after emailing the Major Professor about the action taken by the Student on the Plan of Study.

- **majorProfessorActs.vbs**: This daemon changes the PosStatus value from 20 to 25 after emailing the GPO about the action taken by the Major Professor on the Plan of Study. An email is also sent to the student. This daemon also acts on the occasion where in the Major Professor decides to reject the Plan of Study. The posStatus value would then have been set to 19 and this program then emails student about the rejection and resets the PosStatus to 5.

- **gpoActs.vbs**: This daemon changes the PosStatus value and notifies the Committee Members and the student appropriately based on the two possible actions that could be taken by the Graduate Program Officer – approval, rejection of the Plan of Study. PosStatus is changed from 30 to 35 if major professor has approved it, 29 to 5 if GPO has rejected it.
committeeActs.vbs: This program periodically checks and acts when all the Committee Members have approved a given Plan of Study. It also checks to see if any member rejected the Plan of Study, in which case the student is notified. The feature of remainder emails is also implemented here as explained before.

depthHeadActs.vbs: The department head could approve or reject a Plan of Study once, and this daemon appropriately identifies this action (based on the PosStatus value) and sends out email to the student. PosStatus is changed from 45 to 50 if department head approves it, 44 to 5, if he/she rejects the Plan of Study.

graduateSchoolActs.vbs: The graduate school authority gets to act on a Plan of Study for the first time only after it passes the department head. This authority can then make changes and approve the Plan of Study (PosStatus = 55 and onFile = 1) or unconditionally reject the Plan of Study (email sent to Student and PosStatus reset to 5). This daemon implements this logic.

Reminders.vbs: This daemon is responsible for querying the database and sending out reminder emails to the faculty member (Major Professor, GPO, Committee Members, or Department head) that no action has been taken on a particular Plan of Study for a particular period of time.

FRONTEND DESIGN

The main login page of GS-POSt will be available as a link from the Graduate School
Homepage of Auburn University. The database of GS-POSt contains information about the students and the faculty members. Hence, the login page just has userId and Password text fields and upon proper authentication and identification, the GS-POSt user is appropriately directed to a subsequent page. In the userId field, users are required to enter the Windows Domain Global ID that every student and faculty member of Auburn is provided with.

**Form Layout**

The layout of forms to fill out a plan of study is designed keeping user intuitiveness in mind. The information that needs to be collected for a Plan of Study is split into sections and sequentially collected from different forms by prompting the student at various times. The student can submit the Plan of Study only at one page and once the student submits it successfully, the Plan of Study is sent to the Major Professor for review. The student, on the other hand, can review and revise the plan of study and save it, which basically saves the changes made to the Plan of Study to the database for submittal at a later date.

Extensive User-Input Validation code is written in javascript to reduce the possibilities of errors unintentional or otherwise by students. For example, the social security, year and term a particular course was taken are validated against 'legal' values, and prompted for correction in case of illegal value entries. Some of the important validations on a particular Plan of Study before its submission by the student are as follows:
• The number of minimum required credit hours for a particular degree as stipulated by the graduate school is present in the database in the Degrees table. At the time of submission, the total number of hours in the Plan of Study is checked against the number in the database.

• Every Plan of Study should have a faculty member as a Graduate Advisor and a faculty member as a Graduate Program Officer. These two faculty members could be the same.

• For the students picking the Committee Members, a double drop-down menu is provided for them. The first drop down menu lets the student pick the department. The second drop down menu then lists all the faculty members of the department that the student has picked. The student can then make the choice from this second list. This feature is facilitated by means of directly accessing the GID table that was discussed in the Database Design section.

As can be seen from above, forms are designed to minimize errors of data entry. As the student is filling out the forms, there is certain information that is being entered into the database that the student is not directly entering, but is inferred. A couple of examples of this important use of Javascript are:

• The total number of Committee Members is recorded.

• The total number of student courses and the total number of credit hours of all the courses combined in a Plan of Study are recorded.
OTHER ASPECTS OF GS-POSt

Scalability

The project is developed keeping in mind the issues of scalability. Some of the programming practices and decisions followed in order to have a scalable solution are:

- The HTML forms are scalable as the use of VBScript and Javascript arrays has been done instead of hard-coding any fixed number of form elements.
- Limited use of Session Variables is made as there is a lot of memory and processing overhead on the Web Server as the total number of session variables increases.
- SQL Server 7.0 is an industrial strength database. The database’s scalability features are exploited fully by careful design and judicious use of Stored Procedures.

Extensibility

Issues of Extensibility are also kept in mind in the design and development of GS-POSt. Some of the programming practices and design decisions taken in order to have an
extensible solution are:

- Flags and other numeric indicators are used to tag a particular status or snapshot of a Plan of Study.
- Object-Oriented designing has been made, modularizing the project appropriately.
- Stored Procedures with specific tasks have been identified which eases development and debugging process and facilitates future modifications.

Browser compatibility is also kept in mind and hence DHTML and other proprietary technologies are avoided.

**Security**

As with any other web application, security of the web site and the database is of paramount importance. Some of the notable features of GS-POSt that aid security are:

- **Authentication**: Only after proper verification of the person’s GID and password, does the user get access into the GS-POSt web pages.
- **Browser Sessions**: All the ASP code of each web page is bracketed between tags which verify that a session variable called ValidUser is set to true. If not, a message saying Invalid User is displayed and the user is redirected to the login page. This also means that after a certain period of inactivity (20 minutes is the default session
timeout time), the user would be forcefully logged out and has to re-identity himself/herself to access GS-POSt data.

- **Data Integrity:** Proper User Validation and foreign key references in the database tables ensure that no illegal data corrupts the database. A stored procedure is only executed after validating all the parameters that it takes in.

- **Error handling and housekeeping code:** All the ADO objects [Ref. 7] are used cautiously and as soon as they are used, they are set to null. That way, there are no loose connection objects or recordset objects to the database hanging. Also, after executing each stored procedure, the number of records affected are checked in order to confirm the proper execution of the stored procedure.

**CONCLUSION**

The project has been designed carefully at every step of the process to be flexible enough to adapt to changing needs. After extensive testing and fine-tuning of the look and feel of the web-site, and after adding a section explaining GS-POSt users of its facilities and features, it would go live on the Auburn University web pages. A copy of the web project should be kept aside as the 'development copy' and that needs to be tinkereded first and only after confirming that the changes are to the desired effect, they should be mirrored on to the 'production copy' or 'live copy'. At any given time, there has to be atleast one programmer/developer to maintain, update GS-POSt.
REFERENCES

CITED REFERENCES


5. *UML Distilled*: Martin Fowler and Kendall Scott, Addison Wesley Publication


OTHER REFERENCES

1. *Using Stored Procedures in Active Server Pages*


2. *Developing Web Applications*


3. *Reference Information on Microsoft Technologies (SQL Server 7.0, IIS, Visual Interdev, Active Server Pages, etc)*

   [www.msdn.microsoft.com](http://www.msdn.microsoft.com)
LIST OF APPENDICES

1. **Database Design** - Relationships amongst all the important tables of GS-POSt

2. **Preliminary UML Diagrams** - Typical Use-case Scenarios

3. **Plan of Study Dynamics** - UML State Diagram, the main Middle-Tier Business Logic Flowchart implementation

4. **Front-end Design 1** – The main flowchart of frontend design

5. **Front-end Design 2** – Student Plan of Study Edit Mode flowchart

6. **Middle Tier Design** - Flowchart logic of an example VBScript Daemon

7. **T-SQL Stored Procedure** - Example code of a SQL Server 7.0 Stored Procedure

8. **GS-POSt at work** – Sample pages (print outs) of the working GS-POSt application.
APPENDIX 2

PRELIMINARY UML DESIGN DIAGRAMS

(Use Case Scenarios)

**ACTOR**

- **STUDENT**
  - **ACTION**: FILLS OUT, EDITS, SAVES A PLAN OF STUDY

- **STUDENT**
  - **ACTION**: SUBMITS A PLAN OF STUDY

- **MAJOR PROFESSOR**
  - **ACTION**: EDITS AND APPROVES PLAN OF STUDY

- **MAJOR PROFESSOR**
  - **ACTION**: REJECTS THE PLAN OF STUDY
APPENDIX 4

FRONT-END DESIGN (MAIN)

LOGIN PAGE

AUTHENTICATION AND REDIRECTION

STUDENT

FACULTY MEMBER

GRADUATE SCHOOL AUTHORITY

MAJOR PROFESSOR

COMMITTEE MEMBER

GRADUATE PROGRAM OFFICER

DEPARTMENT HEAD
APPENDIX 5

STUDENT PLAN-OF-STUDY EDIT MODE: FRONT-END DESIGN

STUDENT

STUDENT INFORMATION

COURSES INFORMATION

TRANSFER CREDIT INFORMATION

SUPPORTING WORK INFORMATION

COMMITTEE INFORMATION
APPENDIX 6

DAEMON *GPOACTS.VBS*

PICK OUT ALL PLANS OF STUDY FOR WHICH THE POSSTATUS IS EITHER 29 OR 30

29

SET POSSTATUS BACK TO 5. SEND AN EMAIL TO STUDENT SAYING THAT GPO HAS REJECTED THE POS.

30

SEND EMAIL TO ALL THE COMMITTEE MEMBERS OF THAT POS, SET POSSTATUS TO 35, EMAIL STUDENT.
APPENDIX 7

EXAMPLE OF A STORED PROCEDURE

(This Stored Procedure is called after a student successfully submits the first form (StudentInfo.asp) while filling out a new Plan of Study)

CREATE PROCEDURE putStudentInfo
    @studentId nvarchar(12), @ssn nvarchar(12), @FirstName nvarchar(32), @MiddleName nvarchar(32), @LastName nvarchar(32),
    @studentEmail nvarchar(50), @departmentCode nvarchar(10), @degreeCode nvarchar(10), @iden int OUTPUT

AS

begin

    insert into StudentInfo (studentId, SocialSecurity, FirstName, MiddleName, LastName) values (@studentId, @ssn, @FirstName, @MiddleName, @LastName)

    insert into PlanOfStudy(studentId, StudentEmail, studentLastName, studentSSN, departmentCode, degreeCode)
    values (@studentId, @StudentEmail, @LastName, @ssn, @departmentCode, @degreeCode)

    select @iden = @@identity

end

If @@error <> 0
    Begin
        Return -1
    End
Welcome to the Graduate School Plan of Study Transaction (GS-POST) system. GS-POST was created to allow for the more efficient creation, submission, approval, and use of plans of study. Unlike baccalaureate programs, graduate programs of study are customized for each student. GS-POST converts the plan of study process from a paper-based, static activity to an interactive, dynamic, useable product. GS-POST allows graduate students and faculty to access accurate and up-to-date plans of study from anywhere at anytime through the web.

Should you wish to create a paper plan of study, please go here.

To enter GS-POST, please click the "ENTER" button below. GS-POST will ask you to login. This authentication is based on the Auburn University-assigned global ID (GID) and the Windows Domain password (OIT user ID). This generally is your Mallard ID and password.

To synchronize your passwords, Click Here
For help in understanding the various Auburn passwords, Click Here

Created by Stephen L. McFarland and Sankar G. Vyakaranam.

http://frontpage.duc.auburn.edu/gradschl/gspost/ 5/8/01
TEST SITE (UNDER CONSTRUCTION) - DO NOT USE!!!

GS-POST

UserID: 
Password: 

Sign On  Cancel

Created by Stephen L. McFarland and Sankar G. Vyakaranam.

http://frontpage.duc.auburn.edu/gradschl/gsposl/login.asp

5/6/01
Hello Shanky Gnana Vyakaranam,

Here are the Plan(s) of Study for you to edit:

Building Science-Master of Building Science :

Here are the Plan(s) of Study that you have submitted and are in the approval process:

Computer Science and Software Engineering-Master of Software Engineering (non-thesis):
Computer Science and Software Engineering-Doctor of Philosophy:
Computer Science and Software Engineering-Master of Science (thesis):

Fill out a New Plan of Study:

New Plan of Study

Exit GS-POST

http://frontpage.duc.auburn.edu/gradschl/gspost/mainMenu.asp 5/6/01
MAIN MENU

Student Information

Course Information

Transfer Credit Information

Supporting Work Information

Committee Information

SUBMIT the Plan of Study

Back to Main Menu

Exit GS-POSt
<table>
<thead>
<tr>
<th>Gender:</th>
<th>☐ Female ☑ Male</th>
</tr>
</thead>
<tbody>
<tr>
<td>First Name:</td>
<td>Shanky</td>
</tr>
<tr>
<td>Middle Name:</td>
<td>Gnana</td>
</tr>
<tr>
<td>Last Name:</td>
<td>Vyakaranam</td>
</tr>
<tr>
<td>SSN #:</td>
<td>422-45-3454</td>
</tr>
<tr>
<td>E-Mail:</td>
<td><a href="mailto:vgsankar@hotmail.com">vgsankar@hotmail.com</a></td>
</tr>
<tr>
<td>Department:</td>
<td>Computer Science and Software Engineering</td>
</tr>
<tr>
<td>Degree:</td>
<td>Master of Software Engineering (non-thesis)</td>
</tr>
</tbody>
</table>
GS-POST

For courses already completed, indicate beside each the Semester and Year in which the course was taken.
For courses not taken yet, indicate beside each, the Semester and Year that you intend to take the course.

The following course(s) are required for this graduate program:

3 hours of CAPSTONE PROJECT (BSCI 7980)

Other Required Courses are already inserted into the Plan of Study.

AUBURN UNIVERSITY GRADUATE COURSES

<table>
<thead>
<tr>
<th>Subject Code</th>
<th>Course Number</th>
<th>Name of the Course</th>
<th>Term</th>
<th>Year (yyyy)</th>
<th>Credit Hours</th>
<th>Delete Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>BSCI</td>
<td>7020</td>
<td>ADVANCED CONSTRUCTION SCHEDULING</td>
<td>Winter</td>
<td>2001</td>
<td>3</td>
<td>CANNOT DELETE</td>
</tr>
<tr>
<td>BSCI</td>
<td>7040</td>
<td>PROJECT DELIVERY SYSTEMS</td>
<td>Spring</td>
<td>2001</td>
<td>3</td>
<td>CANNOT DELETE</td>
</tr>
<tr>
<td>BSCI</td>
<td>7010</td>
<td>CONSTRUCTION LABOR &amp; PRODUCTIVITY</td>
<td>Summer</td>
<td>2001</td>
<td>3</td>
<td>CANNOT DELETE</td>
</tr>
<tr>
<td>BSCI</td>
<td>7030</td>
<td>CONSTRUCTION INFORMATION MNGT</td>
<td>Spring</td>
<td>2002</td>
<td>3</td>
<td>CANNOT DELETE</td>
</tr>
<tr>
<td>BSCI</td>
<td>7950</td>
<td>SEMINAR</td>
<td>Fall</td>
<td>2002</td>
<td>2</td>
<td>CANNOT DELETE</td>
</tr>
<tr>
<td>ADED</td>
<td>6000</td>
<td>Research</td>
<td>Fall</td>
<td>2001</td>
<td>30</td>
<td></td>
</tr>
</tbody>
</table>

None

None

None

None

None

None
See Graduate Bulletin for regulations concerning transfer of Graduate Credit. No credit will be approved without an official transcript.

**TRANSFER CREDIT**

<table>
<thead>
<tr>
<th>Institution</th>
<th>Course Name</th>
<th>Course Number</th>
<th>Year (yyyy)</th>
<th>Term</th>
<th>Credit Hours</th>
<th>Quarter or Semester</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>None ❌</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>None ❌</td>
</tr>
</tbody>
</table>

Add More Courses  cancel  save
See Graduate Bulletin for regulations concerning Supporting Work
No credit will be approved without an official transcript.

**Supporting Work**

<table>
<thead>
<tr>
<th>Subject Code</th>
<th>Course Number</th>
<th>Name of the Course</th>
<th>Term</th>
<th>Year (yyyy)</th>
<th>Credit Hours</th>
<th>Delete Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
<td></td>
<td>None</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>None</td>
<td></td>
<td></td>
<td>None</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

[Add More Courses]  [Cancel]  [Save]
### ADVISORY COMMITTEE

<table>
<thead>
<tr>
<th>ROLE</th>
<th>FACULTY ID</th>
<th>DELETE OPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Department Head</td>
<td>crossjl</td>
<td>CANNOT DELETE</td>
</tr>
<tr>
<td>Graduate Program Officer</td>
<td>changka</td>
<td>CANNOT DELETE</td>
</tr>
<tr>
<td>Major Professor</td>
<td>carlisle</td>
<td></td>
</tr>
<tr>
<td>Committee Member 1</td>
<td>changka</td>
<td></td>
</tr>
<tr>
<td>Committee Member 2</td>
<td>hendrix</td>
<td></td>
</tr>
<tr>
<td>Committee Member 3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Committee Member 4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Committee Member 5</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Submit original and one extra copy  Auburn University  Semester Version

The Graduate School
Tenative Plan Of Study

<table>
<thead>
<tr>
<th>Last Name</th>
<th>First Name</th>
<th>Middle Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vyakaranam</td>
<td>Shanky</td>
<td>Gnanam</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Student Number</th>
<th>Date</th>
<th>Program</th>
</tr>
</thead>
<tbody>
<tr>
<td>422-45-3454</td>
<td></td>
<td>Master of Software Engineering (non-thesis)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Internship Required?</th>
<th>Foreign Language Required?</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>NO</td>
</tr>
</tbody>
</table>

Advisory Committee

<table>
<thead>
<tr>
<th>Committee Chair</th>
<th>Committee Chair Signature</th>
</tr>
</thead>
<tbody>
<tr>
<td>Homer Carlisle</td>
<td>NO DECISION YET</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Department Head</th>
<th>Department Head Signature</th>
</tr>
</thead>
<tbody>
<tr>
<td>James Cross</td>
<td>NO DECISION YET</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Graduate Program Officer</th>
<th>Graduate Program Officer Signature</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kai Chang</td>
<td>NO DECISION YET</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Member # 1</th>
<th>Member # 1 Signature:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kai Chang</td>
<td>NO DECISION YET</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Member # 2</th>
<th>Member # 2 Signature:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dean Hendrix</td>
<td>NO DECISION YET</td>
</tr>
</tbody>
</table>

Dean, Graduate School

Auburn University Course Work

<table>
<thead>
<tr>
<th>Name of Course</th>
<th>Course Code and Number</th>
<th>Semester &amp; year</th>
<th>Semester Hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Artificial Intelligence</td>
<td>COMP 560</td>
<td>Summer 1999</td>
<td>4</td>
</tr>
<tr>
<td>Object Oriented Programming</td>
<td>COMP 525</td>
<td>Summer 1999</td>
<td>3</td>
</tr>
<tr>
<td>Database Systems</td>
<td>COMP 6210</td>
<td>Winter 2000</td>
<td>3</td>
</tr>
<tr>
<td>Advanced Management Information Systems</td>
<td>MNGT 650</td>
<td>Winter 2000</td>
<td>5</td>
</tr>
<tr>
<td>MSWE Project</td>
<td>COMP 7980</td>
<td>Spring 2001</td>
<td>3</td>
</tr>
<tr>
<td>Software Quality Engineering</td>
<td>COMP 6320</td>
<td>Spring 2000</td>
<td>20</td>
</tr>
<tr>
<td>Total Semester Hours Required for Degree = 33</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(Graduate Course Hours plus Research and Dissertation Hours)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Hello Homer Washington Carlisle,

ROLE: MAJOR PROFESSOR
You have no Plans of Study to edit for which you are the Major Professor.

Here are the Plans of Study that you have approved (as a Major Professor) and are not yet in file:

Vyakaranam - MSWE  View/Print
Vyakaranam - PHD   View/Print
Vyakaranam - MS    View/Print

ROLE: COMMITTEE MEMBER

You have no Plans of Study to act upon as a Committee Member.

Exit GS-POST
Hello Kai Chang,

ROLE: MAJOR PROFESSOR
You have no Plans of Study to edit for which you are the Major Professor.

ROLE: COMMITTEE MEMBER
Here are the Plans of Study for you to act upon as a Committee Member:

Vyakaranam - MS  Approve/Disapprove

ROLE: GRADUATE PROGRAM OFFICER

You have no Plans of Study to act upon as a GPO

Here are the Plans of Study that you have approved and not yet on file:

Vyakaranam - PHD  View/Print
Vyakaranam - MS  View/Print

Exit GS-POST
MAIN MENU

Student
Information

Course
Information

Transfer Credit
Information

Supporting Work
Information

Committee
Information

Approve/Disapprove Plan of Study

Back to
Main Menu

Exit GS-POSt
GS-POST

THANK YOU FOR USING GS-POST

Go back to Login Page

Auburn University Home Page

Created by Stephen L. McFarland and Sankar G. Vyakaranam.

http://frontpage.duc.auburn.edu/gradschl/gspost/student/exit.asp 5/6/01