# Three Systems Analysis and Development Courses are Needed in the Information Systems Curriculum: Course Content Described

**Jack Russell, Ph.D, CCP**
Professor of CIS
Northwestern State University
Nachitoches, Lousiana 71497
E-mail: russeL1@tarleton.edu

## INTRODUCTION:

Most disciplines in computer information systems, information systems, computer science, and information sciences teach a systems analysis and design course. Anyone who has taught a systems analysis and design course would agree that there are many content issues, as well as philosophical differences, among educators as to which methods should be emphasized. Today, this is especially an issue with new methods (i.e., Object Oriented Analysis (OOA), Object Modeling Technique (OMT), Unified Modeling Language (UML)) making their entrance. The most noteworthy is Object Oriented Analysis that makes use of OMT or UML. As faculty, do we follow an OOA approach or do we follow a more traditional methodology such as Information Engineering (IE), Prototyping, Rapid Application Design (RAD) or Structured Analysis?

According to most texts on the subject, the purpose of the first course in systems analysis and design is to provide the student with a broad range of knowledge about different modeling techniques that survey both the OOA arena and the Information Engineering/Structured Analysis/RAD arenas as well(1). In this first systems analysis (S/A) course, students must learn a myriad of concepts related to each of these techniques and methods. As a result, the amount of coverage time for each methodology is reduced to accommodate new techniques and methods. The result is students know less and less about each method. Our college graduates' reduced knowledge level of process and data modeling comes as a shock to many corporate analysts(2). This is especially true when these skills are at a premium within the development world today.

### Systems Analyst -- Number One Occupation in America

According to the U.S. Labor Department (January 1999) the number one occupation in America is the business systems analyst. The business systems analyst was rated higher than any other computer occupation as well. The engineer came in a close second.

Companies are searching for qualified systems analysts at an unprecedented rate. Industry admits that universities cannot provide graduates quickly enough who possess the right skills in analysis and design to satisfy industry's needs today. Specifically, two factors are contributing to the shortage of systems analysts. One, there is a tremendous shortage of computer specialists regardless of the area of expertise. Anyone who has kept up with the news already know that there continues to be a shortage of network analysts, application computer programmers, database analysts, internet specialists and

many others in related computer career opportunities.

The irony lies in the fact that this disparity between supply and demand for analysts is that many universities continue to teach only one systems analysis and design course. Many faculty are acutely cognizant of course limitations related to the guidelines of various business school accrediting institutions. Typically these guidelines allow for no more than thirty-six hours in the field. Often this course limitation prohibits the addition of new courses.

Typically a curriculum consists of courses that have been available for a long time, and it will also include more state-of-the-art courses like JAVA, Small Talk, COOL JEX and the like. In other words, as time as marched by and based on new demands, the curriculums have expanded. This evolving curriculum will often occur as a result of too much bias and not enough will be based on a thorough analysis of departmental vision and purpose.

**The IS '97 Model Curriculum Emerges**

Based on a horrific need for curriculum reform and guidelines, the various professional arms which include the Association of Computing Machinery (ACM), Association of Information Systems (AIS), and the Association of Information Technology Professionals (AITP) endorsed the *IS '97 Model Curriculum and Guidelines for Undergraduate Degree Programs in Information Systems.* The IS '97 was engineered by Dr. Gordon B. Davis, Dr. John Gorgone, Dr. J. Daniel Cougar, Dr. David Feinstein and Dr. Herbert Longenecker, Jr. along with a host of contributors from around the country. This model curriculum is the first collaborative curriculum effort of the ACM, AIS and AITP societies and is supported by other interested organizations. The curriculum was reviewed at eleven national and international meetings involving over one thousand experts from both academia and business. The major rationale for the curriculum is centered on the need to remain current in this rapidly changing field. Frequent curriculum update is needed if it is to remain effective.

The importance of the IS '97 curriculum effort is based on the continuing strong demand for graduates in this field. This strong demand is forecast by the U.S. Bureau of Labor Statistics and is expected to rapidly grow until the year 2005. A 110 percent increase for systems analysts is expected for the period 1992 through 2005 averaging over eight (8) percent annually.

Considering all occupations, according to this U.S. Bureau, the systems analyst career is projected to have one of the highest growth rates. The *IS '97 Curriculum* supports a strong balance of courses that in turn support competencies in program design, programming, advanced programming languages, productivity software, database management systems (DBMS), hardware and software technologies and operating systems, data and file structures, networks and telecommunications, decision support theory and practice, systems analysis, systems design and implementation, and project management.

The IS 97 curriculum consists of ten courses (30 hours). The last three of the courses include the following: IS '97.7 (Analysis and Logical Design), IS '97.8 (Physical Design and Implementation with DBMS) and IS '97.9 (Physical Design and Implementation with a Programming Environment).

The analyst must be able to analyze at the conceptual or logical level. The IS '97.7 course addresses this level. It is often

the traditional systems analysis course, but emphasizes a balance of data and process modeling concepts with object-oriented analysis concepts as well. It stresses interviewing and presentation skills. The course includes joint application design (JAD), prototyping and database design.

The analyst must be able to also physically design and implement the conceptual model; therefore, the IS '97.8 focuses on data modeling and process modeling tools: Information Engineering and object design approaches such as Object Modeling Technique (OMT) or Unified Modeling Languages (UML). The course emphasizes the use of CASE (Computer-Aided Systems Engineering) tools and repositories along with Windows/GUI design and coding.

The analyst must be able to build robust, client-server and web solutions from the physical model. The IS '97.9 course purports to teach the student how to do this. Emphasis is placed on software construction of event-driven, windows-based applications that utilize client-server technology. Industry demand is high for specialists in this area.

**Three Systems Analysis Courses are Needed within the IS Curriculum**

The main purpose of this article is to encourage the teaching of all three of the systems courses proposed by the IS '97 curriculum but with emphasis in the use of model-based analysis and design. In other words, more emphasis should be placed on rapid application design (RAD) and CASE technology. To achieve this goal, an outline is presented of three essential courses that should be taught pertaining to systems analysis and design. It is the second course within the series that is highlighted and discussed in greater detail. The author has designed a third course as well and it will be the subject of a future paper. The narrative that follows briefly describes the three courses. The first course is titled “Systems Analysis and Design" which is essentially the IS 97.7 course. The second course is titled “Advanced Systems Development I (Model-Based Development) which is essentially the IS '97.8 course with a special proclivity toward model-based development. The third course is titled “Advanced Systems Development II (Client-server Design and Development). The third course is essentially the IS '97.9 with special attention to client-server development within a full-life cycle environment.

**The First Course (Systems Analysis and Design)**

The traditional systems analysis course can continue to provide a broad knowledge base pertaining to both the data and process-modeling worlds in addition to the upcoming object oriented environment. Students will learn a gamut of modeling techniques which will include Information Engineering, RAD, Joint Application Design (JAD), Prototyping, Object Oriented Analysis and Object Modeling Technique. The learning outcomes of this course will ensure that the student will have a basic grasp of the terminology for all techniques. The course includes interviewing, presentation skills, group dynamics, some JAD and prototyping with a hypothetical client.

**The Second Course (Advanced Systems Development I (A Model-Based Approach)**

Educators and corporate professionals seem to agree that students need a comprehensive understanding of both methodological approaches to systems analysis and design. The learning objectives of the proposed second course in systems

analysis is centered on the belief that strong data and process modeling techniques are essential to the success of the college graduate who plans to enter systems analysis and systems development. Moreover, to develop an understanding of the object oriented concepts students need to have a firm understanding of both process and data modeling. Students must drill through a plurality of entity relationship diagrams before they are proficient. Students must work within a team environment on a non-trivial business problem that requires both data and process modeling talent. Having a strong knowledge of IE, Structured Analysis, RAD and Prototyping provides the future student of Object Oriented Analysis the advantage of being able to see the affinity between the various methods as well as their differences. Finally, the student, with a strong proficiency in using full life cycle development software will have a tremendous edge as they enter the profession. Students will use a model-based approach to the use of repositories and code generation through the use of a full-life cycle, model-based, software development product.

**The Third Course: (Advanced Systems Development II)**

College graduates must have a thorough grasp of object-oriented analysis and design skills. They must have a strong grasp of Object Modeling Technique and Unified Modeling Language. In addition to understanding the basic concepts of OOA the student must be able to apply this knowledge in a team setting and solve "real world" problems. The OOA concepts must be reviewed and students will be required to apply this knowledge to a case study. At least twenty five percent (25%) of the course would be devoted to OOA concepts. Approximately twenty five percent (25%) of the class time will be devoted to the application of OOA techniques (OMT, UML). Fifty percent (50%) of the class would be devoted to interviewing techniques, structured interviewing, proposal writing, payback analysis, client-server and web enabling concepts and tools. As the reader knows the demand for graduates with web enabling and client-server skills are at an all time high in the job market. The teacher of the third course can be creative as to which object-oriented avenue to take. The author has postponed OOA to the third course. This deviates a little from the model of the IS '97 where it is introduced in the second systems course (IS '98). The teacher should focus on allowing students to analyze, design and implement a complete business system using an object-oriented approach. This business application should involve web enabling and/or client-server functionality. If time does not permit emphasizing both technologies then it is recommended to emphasize a web-enabled feature somewhere within the business system.

**More Reasons to Support the Teaching of Traditional Methods**

New college graduates with web-enabled and client-server skills, who also possess a strong understanding of both data and process modeling, are at the pinnacle of demand. Conference proceedings often reveal that students are expected to understand traditional process and data modeling once they get to the job. The following reasons support the teaching of an advanced course, which emphasizes process and data modeling using Integrated CASE with full code generation.

While some companies are beginning to show initial evidence of using object-oriented databases the movement remains slow and erratic. Latest statistics

indicate that still over 80 percent of American companies continue to follow Information Engineering, Prototyping, JAD, or RAD rather than Object Oriented Analysis. The reason lies in the myriad of data and process models currently within use today(3). The author believes that there is a strong movement in the direction of OOA and object oriented tools, but at the present time we should not swiftly abandon the more traditional analysis and design methods. The corporate world would be disappointed if we did abandon IE and prototyping with only a cursory coverage given to it; therefore, a strong course in data and process modeling is essential in providing future graduates with the right skills at the right time. Once a strong knowledge base of data and process modeling is built using the second course (Advanced Systems Development I), the third advanced systems development course should have a strong learning objective that will include object modeling and Object Oriented-Analysis. The author believes that OOA must be taught within the IS curriculum. The sequencing of these courses will become the real issue. The concepts of objects, attributes, behaviors, encapsulation, classes, inheritance, supertypes, subtypes, object class relationships, messaging, polymorphism, "use case" modeling and use "case dependency diagrams" are certainly better understood with a strong grasp of both data and processing modeling. The systems analysis textbooks such as Whitten, Bentley, McGraw Hill, 1997 encourage teachers to teach the chapters related to traditional data modeling and process modeling prior to teaching object modeling (4). Many agree that the basis of OOA is to reduce the complexity associated with data modeling and processing modeling. This requires a strong understanding of Information Engineering (IE) and Rapid Application Development (RAD). These subjects require more time than can be presented in a single course.

Corporate executives and representatives continue to recruit college information systems graduates that have relational modeling skills and who can effectively use and manipulate relational databases. The demand for these relational skills has not diminished. In fact, the demand is increasing. Companies who invested heavily in the 70's and early 80's in hierarchical and network databases are just now trying to upgrade these legacy systems to the relational environment using DB2, DB2-2, Oracle, SQL Server, Sybase and others. The need for new graduates that possess these relational database skills is still on the climb (5).

**THE PROBLEM**:

**Students Lack Specific Skills in Data and Processing Modeling**

Today, as previously mentioned, a common criticism from the corporate representative is that the typical college graduate with a degree in an information systems discipline has a poor grasp of the systems development process regardless of the methodology being followed. Often these students are quite competent in various programming languages, data structures, operating systems, network and database management systems; however, when they are confronted with rudimentary tasks related to process and data modeling, the new hire is often perplexed and unable to perform adequately. Students who have basic knowledge of Object-Oriented Analysis and Design are certainly praised for their knowledge, but often have great difficulty functioning within the typical development environment which continues to use relational databases, COBOL, C++,

DBMS, SQL and integrated CASE products (6).

New information systems graduates who possess data and process modeling skills are often made team leaders prior to other peers, as well as their superiors in some cases. New graduates who understand IE and Structured Analysis are able to grasp OOA concepts quickly.

**Students Need Team Skills**

A small percentage of these newly hired graduates have had to work as a team member on an academic project from the planning phase all the way through implementation. Many students have worked together as a team on the analysis phase of a project but have not performed significant design and implementation in a full life-cycle development environment. The typical beginning analysis and design course simply is too overloaded with theories, concepts and methods to allow students much time to immerse themselves into real-life industry experiences. The limited time available also prohibits students from becoming proficient with the use of an Integrated Computer-Aided Systems Engineering product. Proficiency with CASE is becoming a must in systems analysis and development today. This is especially true when team participants build non-trivial and intricate models that usually require each person to work with precision.

**Students Need More Model-Based Development Skills**

The author believes it is essential that senior level information systems students gain a strong degree of proficiency at using both prototyping tools and full-life cycle design and development tools, i.e. I-CASE or Model-Based Development. This environment is essential for students to work as a team and complete specific project deliverables on time and within budgetary constraints. This belief may go in the face of those who believe that we should not teach specific technology but instead should teach only theories and concepts. My philosophy of teaching attempts to integrate theories and concepts and the application of both. The author believes that theories and concepts must be brought alive with actual applications for the students to work on with analysis and design software products. Students should especially gain proficiency in the use of these tools that promote skills in client-server development and the web enabling of system applications. Besides, students with these specific software skills are in great demand in industry and provide an additional job outlet for your new graduates. The author believes that while we continue to be a clearinghouse of fresh and new knowledge our goal should also promote the training of our graduates to be able to immediately become productive within the systems analysis, design and development arena. This is exactly what the author has done at his university. Students with specific client-server or web enabling software skills are hired before those who do not. Just as an example, in the area of systems analysis, one can simply surf the Internet to see the great demand for graduates with specific Oracle Developer 2000 and Sterling Products skills in Cool Gen and Cool Jex. These are highly sought-after skills by various consulting companies within the United States (7).

**Reasons Why the First Course in Systems Analysis is not enough:**

In this first course of systems analysis and design, students are often taught what a modern system analyst does and the role of management and users in the systems problem solving cycle. Students are often made aware of modern business trends

and the associated implications for the systems analyst. Students will likely review a number of information systems fundamentals related to transaction processing, interactive processing, MIS, decision support systems, expert systems, data warehousing and Integrated Computer-Aided Systems Engineering.

The first systems analysis course often will include knowledge about the different methodologies that are usually followed in the analysis and design process. These methodologies typically include the following: Structured Analysis, Information Engineering (IE), Joint Application Design (JAD), Prototyping, Business Process Redesign (BPR), Object-Oriented Analysis (OOA) and others (8). Students will often learn the basic differences of some of these methods, but time constraints prohibit students from delving deeply into this material.

Teachers frequently skip several of the chapters in the typical systems analysis text simply because of time limitations for each section of material. Most teachers of systems analysis will agree that time does not permit complete coverage of chapters on data modeling techniques, process modeling techniques and various user interface styles. Often, teachers of systems analysis and design lack blocks of time to adequately teach these topics that include JAD, RAD and Prototyping methods in great enough detail for the material to become meaningful and relevant to the student.

Often a cursory coverage is provided pertaining to the use of a CASE software product. This type of coverage is often inadequate in providing the student with proficiency in its use. The better texts (author's opinion) on the subject of Systems Analysis and Design typically provide some exposure to the use of CASE. For example, *Systems Analysis and Design Methods* by Whitten and Bentley (McGraw Hill) uses a product called Systems Architect by Popkin Software(9). A text by Course Technologies titled *Systems Analysis and Design*, Third Edition, by Shelly, Cashman and Rosenblatt uses a CASE product called Visible Analyst (10). The Visible Analyst tool can be bundled with the textbook to allow the student a personal copy. These are typical scenarios with the various textbooks. Students will often create a few data models and process models with these CASE products; hence some student learning will take place. The problem is that students rarely get far enough to cause much advanced (industrial grade) learning to take place. When questioned toward the end of this course students will often admit that the material was a "blur", and there was simply no time to drill back through the material nor time to delve into examples that might challenge the motivated students.

**A Rationale for a Second Course in Systems Analysis and Design:**

The rationale for these capstone experiences in analysis and design is based upon the following:

1. Students will learn how to apply the various concepts learned from their first S/A course.
2. Students will gain an advanced understanding of data modeling by using an integrated CASE environment. Students will design relational data models that resemble real world models.
3. Students will gain an advanced understanding of process modeling by using an integrated CASE environment. Students will build robust, real world, non-trivial process models.
4. Students will learn how to apply the various relational business models to the design and implementation of a real

business system that includes Windows/GUI coding and client-server planning, testing and installation.
5. Students will learn to interact with other students on a team and find ways to solve non-trivial business problems that are both process and data intensive.
6. Students will become proficient in using an integrated software design and implementation software package like Sterling's Cool Gen, Oracle's Developer 2000 or one of the other full-life cycle development tools.

**A Course Titled: Advanced Systems Development I (Model-Based Development)**

A model capstone course that addresses the assumptions listed above will include the learning outcomes listed below. The components are built around the Information Engineering, Prototyping and Structured Analysis methodologies. The course will focus on business area analysis where the students will learn how to draw entity relationship diagrams (ERD) through proper normalization processes.

First Normal Form (1NF), Second Normal Form (2NF) and Third Normal Form (3NF) will be discussed in great detail. Students will design many ERDs from scratch. Many examples of the inappropriate design of entity relationship diagrams will also be described for student evaluation. Students will learn to examine these poorly designed entity relationship diagrams and determine what new entity types need inclusion to remove repeating fields (1NF), remove partial dependencies (2NF) and to remove transitive dependencies (3NF).

Students will be able to develop a robust data model from concepts learned from their database class and from this course as well. The author stresses the term "robust" data model. Any analysis class will expose students to some level of data modeling, but will often fall short in providing an advanced coverage of data modeling.

**Learning Outcomes:**

Upon completion of this course students will be able to:
1. Compare and contrast Information Engineering, Structured Analysis, Prototyping, JAD, RAD and OOA.
2. Define Integrated Computer-Aided Systems Engineering.
3. Discriminate and apply Information Engineering to specific kinds of projects.
4. Discriminate and apply Structured Analysis to specific kinds of projects.
5. List the characteristics of Information Engineering.
6. Identify key terms used in relational database design (i.e., data model, ERD, relationship, relationship membership, cardinality, optionality, referential integrity and data integrity).
7. Draw entity relationship diagrams. Students will draw fourteen (14) diagrams. Two of the diagrams will build on the previous one. This insures that students understand normalization. The teacher asks students to improve the existing diagram by finding 1NF, 2NF and 3NF violations, and then redraw the diagram using the Cool Gen software.
8. Define terms such as functions, processes and elementary processes (or primitive processes).
9. Draw activity hierarchy diagrams that show the hierarchical relationships between the functions and processes. This diagram will require the decomposition of processes to their lowest levels that continue to leave the business in a consistent state. In other words, students will learn how to

decompose processes into ones that are loosely coupled and that contains tightly cohesive statements. Students will learn the practical application of this concept using the Cool Gen Analysis workstation. Students are presented problems in activity analysis and are expected to analyze and decompose functions and processes.

10. Draw activity dependency diagrams that will show how one process depends on the successful execution of another. Students will learn how to construct this diagram so that it is clear to the reader the physical order of the processes. In other words, each process has a precondition that must be specified before the process is executed. For example, a determination that a "customer must exist" prior to the process of "taking a customer order" is a part of this analysis. The basic purpose of having students design this diagram is to validate the activity hierarchy diagram (or decomposition diagram).
11. Analyze the various processes within the activity hierarchy and determine which of these processes should be included within a specific business system. It is the business system that will be tied to a specific project that will require specific resources. Students must learn to analyze and build appropriate business systems that translate into projects. This is a skill that industry seeks from its candidates (11).
12. Design the navigation path or dialog path between various procedures within the system. A procedure represents the user interface (window or screen) and its associated procedure action diagram (high level statements that can later be constructed into C++, COBOL, ADA, JAVA, etc.) Students will develop an understanding of the window structure of a business system by drawing either the dialog flow diagram or the navigation diagram. Both of these diagrams represent the same navigational flow between the various windows or screens, but in a different graphical format. Students will comprehend various mechanisms or conditions (i.e., exit states) that would cause transfer of control from one window to another. [12] Regardless of the development software being used it is critical for students to understand window or screen navigation. Students can learn this from Visual Basic, Visual COBOL, Visual C++, Oracle Developer 2000, Sterling Cool Gen.
13. Understand different window styles and techniques of GUI window design.
14. Build window and dialog boxes. Students will learn how to build useful main menus as a primary window. They will learn how to design primary windows that contain "pull downs", list boxes, check boxes, radio buttons, vertical scroll and horizontal scroll and bit maps.
15. Build dialog boxes that are invoked from a primary window. The dialog box is often invoked as a result of an event such as clicking a menu item on the primary window. It is the dialog box that allows data capture and update. Event processing concepts are followed here. An example: A user clicks "ADD CUSTOMER" from the menu item. This event triggers the opening of a dialog box that will enable the user to capture the attributes for a customer. The student will often develop the dialog box with a push button or a series of pushbuttons. In this example, a student would include a pushbutton called "CREATE" to cause the customer to be physically added to the database. This

would require the student to know how to create an "event action" and its associated code for this special event.

16. Build the associated Procedure Action Diagrams (PrADs) and Process Action Diagrams (PADs) for the various windows designs. The Procedure Action Diagrams represent the user interface statements behind the window and dialog box actions. The Process Action Diagrams (PADs) represent the high level SQL statements behind the elementary processes. There is a PAD for each elementary process. For example: The elementary process called CREATE CUSTOMER has a Process Action Diagram that has the logic in it that will cause a row of data to be added to the customer table.
17. Establish the technical environment for the business system. In other words, select the particular database environment, establish a database name and so forth. Students will select the appropriate operating system platform (Windows 95, Windows 98, Windows NT, OS2, MVS, etc.) for this business system.
18. Package the various procedures into an appropriate load module or separate load modules.
19. Generate the business model into C, COBOL, JAVA or ADA.
20. Install (or Compile) the source language code into machine language for testing.
21. Test the load module(s) for proper functionality and accuracy.
22. Complete the *Tutorial Series for Cool: Gen* by Jack Russell. This is a comprehensive but simple to use tutorial that carries the student in a click by click fashion through analysis, design and construction of a simple customer maintenance and sales representative maintenance model. Various universities within the Sterling University Program are using this tutorial set(12).
23. Work in a small group to solve a business problem related to a specific business area within the organization. This is a small contrived business case that students can normally solve in 12 to 15 clock hours.

**Materials:**

1. *Systems Analysis and Design Methods*, Whitten Bentley and Barlow, McGraw *Hill*
2. *Sterling Guidelines for Success, Developer's Series: Analysis,* Second Edition, Sterling Part Number 2616285-0003
3. *Sterling Guidelines for Success, Developer's Series: Design,* Second Edition,
   Sterling Part Number 2616286-003
4. *A Tutorial Series for Cool: Gen 4.11,* Jack Russell
5. *Student Tool Kit for Cool: Gen* (optional)
6. Teacher handout packet, Jack Russell

**Suggested Laboratory Requirements**

It is suggested, although not a necessity, to create a separate teaching laboratory. It is essential to have a computer available for the teacher in the front of the room with a quality presentation projection system. To provide structured hands-on sessions during the semester it is essential to have a separate facility. For large lecture demonstrations it is essential to have a teaching classroom with a high end microcomputer loaded with Cool Gen, an acceptable database package (Oracle, SQL Server, DB2, DB2-2, Sybase). The system should have Visual C++ on the hard drive as well. To perform web enabled applications JAVA will provide some advantages.

In many laboratory situations students can simply "go to the lab" to finish their assignments; however, in the business systems design assignments it is helpful to lock-step the students through dialog design, window design, procedure action diagrams and view matching.

**Teacher Training Needed:**

To enable the teacher adequate training in the use of Cool Gen, Sterling Corporation will provide workshop exposure for faculty who are members of the Sterling University Program. When seating is available teachers may attend these sessions free of charge. Sterling provides some training opportunities for faculty who attend the Sterling User Conference held in Plano, TX each year. Faculty may contact Sterling through their web site at www.sterling.com. Look for the University Program information.

**Teaching Tips:**

The author has developed a Laboratory Kit for this class, which includes:

1. Questions related to Information Engineering and Structured Analysis concepts.
2. Data modeling exercises which include fourteen (14) entity relationship diagramming exercises.
3. Data normalization exercises -- Students are provided with incomplete data models and are asked to remove repeating fields, partial dependencies, and transitive dependencies and expand the entity relationship diagram as appropriate. In class, the teacher can present an unnormalized data model and the respective entity types with attributes. Then ask the students to circle the attributes within the entity types that violate first, second and third normal form. Have the students redraw the data tables with the corrected set of attributes along with the expanded entity relationship diagram.
4. Process modeling exercises -- Students are presented with various business scenarios. The various scenarios include specific narrative that requires the student to recognize functions and distinguish them from various processes. From the narrative students will decompose the functions into high-level processes. Subsequently, from the narrative provided, the student will determine how to decompose the high-level processes into elementary processes. These are the processes that correlate with actual code. Students must develop these process models on paper and also must repeat this activity with the Activity Hierarchy Diagrammer workstation within the Sterling Cool Gen product. One tip is to provide the student with only a partial list of functions, processes and elementary processes. Have the students brainstorm what other processes should be included. Students will often learn more from this class participation session than from the other.
5. Students solve various exercises that require the validation of the process structure within the activity hierarchy. These diagrams have been discussed previously and are called activity dependency diagrams. These exercises require the student to understand how the execution of one process is dependent on a subsequent process and the various events surrounding it. One teaching tip is to provide a classroom example of a slightly erroneous activity hierarchy diagram, which includes three or four elementary processes beneath one single high-level process. Make sure that one of the elementary processes do

not belong with the others. In other words, one of the four elementary processes neither provides a post-condition nor requires a pre-condition from another. Have the students draw the dependency diagram. The attentive student will discover that this particular process is incongruent with the others and should be grouped elsewhere within the hierarchy diagram. The reason, of course, is that the process is not either dependent on a prerequisite process nor provides a post condition to another.

6. Students must solve a set of exercises related to the interaction of the various processes within the activity model with the various entity types of the data model. Students develop a matrix that is often referred to as a "CRUD" matrix. CRUD is an acronym for Create, Read, Update and Delete. These four activities are known as the expected effects that a process (i.e., TAKE CUSTOMER ORDER) can have on a specific entity type (ORDER, ORDER LINE, PRODUCT, and VENDOR). Students will analyze these matrices to determine that a sufficient amount of data is made available to a specific process; on the other hand, they will analyze to determine if a given entity type has sufficient processes to maintain and manipulate the data properly.
7. Students will then be exposed to a series of exercises and questions related to business systems design. These exercises enforce concepts related to procedures, window design and process action diagrams.
8. Students will be required to complete A *Tutorial Series using Cool Gen:*
   This tutorial will step students through the process of creating a data model, process model, and business system. Then the tutorial takes the student through the process of defining the flow or navigation between the business system's windows or screens. Students learn to prototype with the user at this stage of the design process.

   Students are required to develop two primary windows and the associated set of dialog boxes for each. One primary window the student creates is a CUSTOMER MAINTENANCE window. The two associated dialog boxes are the CREATE CUSTOMER and UPDATE CUSTOMER boxes. The other primary window relates to SALESREP MAINTENANCE. The two dialog boxes relate to the creation and update activity as described previously with customer maintenance.

   Students develop the procedure action diagram logic that supports the windows and dialog box events.

   Students will develop the process action diagrams that support the various elementary processes. Some of these would include the logic for "Create Customer", "Delete Customer", "Change Customer" and "List Customer."

   Students determine the technical environments, transform the data model into a data structure list and a data store list.

   Students will generate the Data Definition Language (DDLs) and generate the database from these DDLs.

   Students generate the C++ code from the procedures, windows, procedure action diagrams and process action diagrams.

   Students install (or compile) the C++ into machine language/load modules.

   Students test their business system.

**Tutorial 1** makes sure students can build a single window called CUSTOMER MAINTENANCE and will allow the user to LIST the customer database table within a

LIST BOX. **Tutorial 2** will carry the student through building a second primary window called SALESREP MAINTENANCE and will allow the student to LIST the salesrep database table within the LIST BOX. **Tutorial 3** expands on the previous model to include an UPDATE and DELETE process for each. **Tutorial 4** will expand the model to include a REPORT capability. For example, by clicking on REPORTS menu item, the user can select a specific report that would list all customers who are serviced by a specific salesrep.

9. Once students complete the tutorial series (takes four class periods usually), the student will then be presented a small business problem. The problem presented to them will parallel many of the requirements presented within the tutorial, but the difference is that the student must solve this problem using the Sterling workstation without the help of a tutorial. Students are expected to complete this small project by working as teams. A team will consist of three students. Students are given three class periods to complete a business system that has two primary windows and requires the maintenance of at least four processes for each window/procedure. This small project intentionally resembles the tutorial to some degree so that a student who cannot remember how to perform a given task can go back into their tutorial model to get out of trouble with their project.

The author easily understands how a teacher can run out of semester before the course requirements are complete. Many times the author has had to cut back on the size of the project to accommodate the amount of time available.

The author also teaches a third course in systems analysis and design which involves more team interaction, team interviews, team proposals, and stand up presentation of a completely designed and implemented system. The kinds of design that students will perform in this third class involve both client-server considerations and also web-enabling considerations. In this course, students design both client and server procedures for a distributed client-server application. The author is currently writing a tutorial that will enable students to include presentation logic on the client and data manipulation logic on the server. This course will be the subject of a future paper. The third course will incorporate OOA and traditional model based methods.

**A Course Titled: Advanced Systems Development II (Model-Based Client-server and Web Development)**

The basic purpose of this course is enable students to build robust systems using client-server and web enablement. Students who have completed the first two systems course now know how to build both a conceptual and physical design models. Students are asked to take the physical design a step further and build code that will execute within a distributed processing environment. This is accomplished using a full-life cycle, model-based environment. The author uses the Sterling Cool Gen products for building logical and physical models that include both client and server procedures.

A brief course outline for the third course appears below. A more complete coverage of this course content will be the object of a future article or paper. The purpose here is to simply provide an overview of course content so the reader can observe how the three courses interconnect.

1. Review of business systems design concepts that include advanced data

modeling and business systems design concepts.
2. Client-server styles discussed.
3. The distributed processing client-server style emphasized.
4. The concept of client and server procedures within a model-based environment.
5. Concept of passing commands to a server procedure.
6. Concept of passing either commands or messages back to a client procedure.
7. View matching (or parameter passing) between client and server and from server back to the client.
8. The Read, Read Each, For Each, For, CASE OF, CREATE, UPDATE, DELETE, implicit versus explicit subscripting and others.
9. The concept of using relationship membership rather than using a physical combination key with associative entities.
10. The use of the ASSIGN statement in a CREATE command to avoid the creation of physical foreign keys. The use of a list box to provide a display of foreign key values from which to choose for the foreign key.
11. Midterm examination.
12. Final Project discussion and team member selection. Much emphasis is placed on the importance of team interaction. Students receive both an individual performance grade and a team performance grade.
13. Cost-benefit analysis discussed. Understanding the cost-benefit coefficient and adjusting costs and benefits to a present value.
14. Conducting data collection interviews as the first step of the project.
15. Performing a Proposal to Perform Analysis and Design and recommending a solution. The proposal includes a cost-benefit analysis with all costs and benefits adjusted to present value.
16. Design and construction of the system.
17. Testing the client-server application.
18. Final group presentation of the client-server application.
19. Final examination.

**Conclusion and Recommendations:**

There are many methodologies and schools of thought related to the teaching of systems analysis and design. The author encourages information systems educators to teach a second systems analysis course that builds on the first. If your university teaches a single systems analysis course then the author encourages you and your faculty to consider expanding this critical body of knowledge within your curriculum. I realize that many curriculums are limited in the number of courses one may offer in the IS discipline. The author suggests that the IS faculty weigh the value of having future graduates who are very strong in systems analysis. Companies will be quick to hire highly skilled graduates especially in systems analysis. All we have to do is compare the starting salary for new hires in the area of systems analysis and design with starting salaries in the area of programming. In some cases there is a five to ten thousand dollar salary advantage for the entry-level systems analyst over the entry-level programmer. The author realizes that there are always exceptions to this, but when looking at the national salary surveys in general the analyst is king. Therefore, the author believes that it makes sense to provide significantly more training than is possible in a single systems analysis class.

Departments of Information Systems should provide both teaching and financial resources for an advanced course(s) in systems analysis and design. The creation of a teaching laboratory that is equipped with high-end personal computers with one or more of the integrated CASE (model-based) products such as Sterling's Cool: Gen or Oracle 2000 is essential to the success of teaching the second course in systems analysis. The author also further recommends that the IS '97.7, Analysis and Logical Design, include an alternative use of model-based development for building the conceptual data and process models. The author further recommends that the IS '97.8, Physical Design and Implementation with DBMS, include a stronger recommendation for the use of integrated CASE or model-based development tools. It would be quite feasible to have students update a database through the use of embedded SQL calls from a programming language, and then repeat the update through the use of, let's say, a Sterling Products Cool Gen model. One would essentially generate the same code from required Cool Gen diagrams that must be created. Students will now have a broader understanding of what is out there when developing robust systems.

The science and art of systems analysis is a moving target and therefore the teaching of this subject should be reevaluated frequently. To say the least, we teachers of systems analysis must be alert and willing to take both small steps at times and also quantum leaps at other times so that future information systems graduates are adequately equipped to build tomorrow's highly effective business solutions.

## References

[1] Whitten, Jeffrey and Bentley, Lonnie, *Systems Analysis and Design Methods,* Fourth Edition, Course Technology, 1997, preface.

[2] Simpson, Bill, USAA, Interview, May 1998.

[3] 1997 Proceedings of Sterling University Conference, August 1997, Plano, Texas.

[4] Baumgart, Patrick, Tier Technologies; Matile, Casey, MTW Consulting; Alborn, Al, Texas Instruments; Mansfield, Neal, EDS and Chumbley, Jon, JC Penney; Interviews, February 1998 through October 1998.

[5] Abshire, B., Lowes Companies, North Wilkesboro, NC, Interview, October 1998.

[6] Matile, Casey, MTW Consulting, Interview, September 1998.

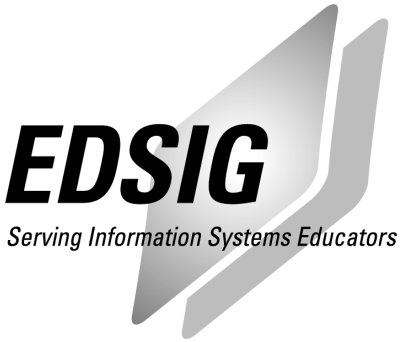
[7] Shelly, Gary; Cashman, Thomas and Rosenblatt, Harry, *Systems Analysis and Design*, Third Edition, Course Technology, 1997, Preface.

[8] Chumbley, Jon; Senior Consulting Analyst, Interview, JC Penney, Plano, Texas, November 1998.

[9] Sterling Software, *Guidelines for Success - Analysis,* Second Edition, 1997.

[10] Sterling Software, *Guidelines for Success - Design,* , Second Edition, 1997 and *A Tutorial Series for Cool:* Gen, Jack Russell


Information Systems & Computing
Academic Professionals




## STATEMENT OF PEER REVIEW INTEGRITY

All papers published in the Journal of Information Systems Education have undergone rigorous peer review. This includes an initial editor screening and double-blind refereeing by three or more expert referees.



ISSN 1055-3096