Software Engineering Team Projects
people.eecs.ku.edu/~saiedian/teaching

Professor Hossein Saiedian

1 Group Projects

Our software engineering class will be divided into groups of 5-6 students. Large groups are chosen partly to challenge students in issues related to project management. (Smaller groups teach fewer lessons about software development in groups.) Students are strongly encouraged to form their own groups of compatible people. Each team should elect a team leader (or administrator, or manager) who will be responsible for coordinating the activities of the other team members as well as communicating with the instructor. Teams are encouraged to assign other roles to the members, such as ‘Project Administrator,’ ‘Deputy Project Administrator,’ ‘Configuration Manager,’ ‘Quality Assurance Manager,’ ‘Maintenance Engineer,’ etc. The team leader (administrator) would also be responsible for final technical decisions as well as making sure everyone comes to meetings and does their share of work. The final implementation should be in Java, C, or C++. One of the UNIX-based systems on campus (e.g., eecs.ku.edu) is acceptable but PC-based systems are also OK. You must strive for operating systems compatibility. The projects will take most of the semester with major write-ups at approximately 2/3-week intervals.

2 Why Group Projects?

The purpose of this course is to give students a broad background in software engineering issues as well as provide them with a “real world” experience in the design and development of a software system. With these goals in mind, we feel that a team project is necessary. There are several advantages to team projects:

- Due to time constraints, most systems developed by industry are not designed and implemented by a single person. Students need to develop the coordinating and communication skills required in real-world environments. Thus, it is felt that students should have the experience of developing a system with team approach.
• The students must do their part of the project using another student’s specification. This generates a substantive amount of discussion between students.

• Because of the nature of developing a software system in a team environment, other issues emerge. Within this framework, students come to realize that effective communication is a must if groups and project teams are to be successful.

• Students learn that they “can’t do it all” and that they have to depend on each other. They learn to recognize and use their individual strengths and expertise. They also learn to recognize other team member's strengths and weaknesses and adjust accordingly.

• Group projects tend to have “hidden” requirements that only emerge under diligent probing during team discussions.

• Students are exposed to the challenges of integrating other individuals’ work.

All of these benefits definitively outweigh problems involved in developing and administrating a team project. Please note the following general notes about the teams:

• It is best to identify not only a team or project ‘Administrator’ but also a ‘Deputy Administrator’ who will take over the project administration (leadership) when the primary administrator is absent.

• Each team must meet a minimum of three (3) hours per week outside the class session.

• Only the team leader (or administrator, or manager) should serve as the team's interface with the client and/or the instructor.

• The grading of the project is team-oriented and thus each member of the team will receive the same grade for the project. Thus the team quality assurance person must make sure that proper review of the presentation is done to correct any deficiency that may exist.

• If there is any problem regarding a member of the team, the problem must be documented and signed by the members of the team at least two weeks prior to the date when the final project is due.

3 Professionalism is Important

You will be graded on the quality of the work you produce, not on how many hours a week you spend on it. Use your energy and time wisely. However, you are requested to create professionally-looking documents, not
only for “clients,” but also for communication among yourselves. Portfolios, labeled theme binders, etc., are recommended. Choose a name for your group and always write team members' names on the project assignments you turn in. Each part of project worths approximately 50 points and is graded based on accuracy, consistency, and completeness of its contents as well as its organization (e.g., appropriate title, section and paragraph names) and appearance (e.g., consistent page numbers). For each document to be turned in, include a title that identifies the document (e.g., “Requirements Definition for the ... System”), the name of the team, names of the team members, course title, instructor’s name, and date.

Even though it is expected that some members of your group will be better writers, some better programmers, and so on, you are not to divide the labor on these grounds. This class is to be a learning experience, and each of you should get a substantial amount of practice in all of these areas, not just the ones you already good at. You will not be graded directly on your writing style, but good writing often conveys ideas more clearly than poor writing. Thus, it is to your advantage to organize your thoughts and write well. Note that document length is not equivalent to quality. Also, it is a good idea to use the same editor and document preparation system for all documents to make modifications and elaboration into a more detailed document easier.

It is very important that project assignments be completed on time to allow you to complete the entire project on schedule.

4 Keep a Log of Your Activities

It is a good idea to maintain a log of the time you spent on the project and a description of what you did during that time. The logs will help you see how you spend your time and help you make better predictions of the time needed by the different phases of a software project. It is also a good idea to record the reason for a computer run, the amount of CPU time, the changes since the last time, the result of run, and so forth. It would always be interesting to look back and see how much time you really spent on the project in front of a terminal and how much you spent away from it.

5 Term Projects

There are two choices in selecting a software engineering project: (1) a realistic project from the local business community or (2) a typical “academic” project (e.g., a simplified airline reservation system, a student registration system, a library system, etc.). There are advantages in and disadvantages with either of these two approaches.

1In fact, many industrial organizations require this sort of time-keeping. I may also require that individual team members turn in their log sheet.
Real Projects for Real Clients. A realistic project from a real customer may be chosen. Care must be put into selecting a project of an appropriate size to ensure that the project can be completed within a semester and that the development process does not get out of control.

A real project will provide the students the experience of tackling a real problem and of formulating their understanding of the requirements (instead of being given artificial and/or trivial statements of the requirements). Students will have to deal with a real, external client, and will realize how important it is to have good communication skills to interact with the client (and users) who are not sophisticated in the use of computers and who are often ambiguous and unclear as to what they really want. Thus the students will gain important real-world experience while interacting with their client and will realize that no amount of lecturing is as effective as real life episodes. Furthermore, the presence of an external client may provide additional incentive for the students to do a professional job and to develop their own self-confidence in building a system for real use. At the same time, the students will improve their understanding of the software development process and will develop a sense of accomplishment.

No-Real Client Projects. As an alternative to a real project with a real client, the instructor may choose to provide the students with a software problem definition, describing the desired functionality of a system (typical examples may include: a registrar system, an airline reservation system, and so forth). The project description would require students to consider developing a system that would address the needs of some “imaginary” client (although the instructor may serve as the system procurer). The requirements are rather artificial.

The primary advantage of such projects is that the students will no longer have to make off-campus trips and would no longer have to face issues discussed earlier. However, the students very often may need to make assumptions about the problem, expected operations, and those requirements of the “imaginary” enterprise that are not specified in the project description.

6 Project Demonstration and Presentation

The last week of class sessions are reserved for project presentations and demonstrations. Presentations should be formal and will take place in the classroom. All team members should participate and contribute to the presentations. Each group will also have about 30–60 minutes to demonstrate their system. The demonstrations will take place in a lab; you should prepare about 20–40 minutes of material that exhibit the best features of your system and allow some time for questions and feedback from the audience. If your system has been properly debugged, you can let the audience tell
you what to type to show off your error handling and user help facilities. Everyone in the group should participate. You can have each person explain their part of the system or have one person typing while another is talking. You may want to give out a short description of your project. Because the time is so short, it is important to practice what you are going to say and do. A complete, bound version of your project should be turned in during the last week of classes.

Your completed project worths about 40% of your overall grade for the course. The completed project will be graded at end of the semester for completeness, correctness, documentation, consistency and uniformity.

7 Essential Project Deliverables

The term project is organized similar to the one given by Kant and Tomayko. A combination of incremental and phased development will be used and each team is required to achieve a number major milestones at approximately 2/3-week intervals. In general, each team must produce a document (e.g., requirements specification, test plan, user's manual, etc.) for each major phase of the software life cycle, and at the end of the semester do a presentation highlighting the features of the system, and a demonstration showing the its functionality. The major development phases that will be emphasized include:

- Requirements definition/specification
- Architectural design
- Detailed design
- Implementation and testing

For a more detailed description of the development process that will be emphasized and followed, you may want to consult a related article published by Saiedian.

8 Tentative Schedule and Grades

The following is a tentative list and schedule for the project documents that need to be completed and turned in (especially when you will be using the traditional waterfall approach). Additional items (e.g., a formal software configuration management plan, a software quality assurance plan, etc) may need to be turned in, either as a separate document or as part of one of the following documents (e.g., as part of the software project management plan).

1. An informal statement of the requirements for the Term Project. Team title, members, a description of tentative roles for each member, contact information, etc., should be included.
   Due: 9/9

2. A formal statement of the requirements for the Term Project. Use a combination of semi-formal and formal notations, complemented by natural language descriptions.
   Due: 9/30

3. A software project management for the Term Project.
   Due: 10/7

4. Architectural and detailed design of the Term Project.
   Due: 11/4

5. Implementation and integration of the Term Project.
   Due: 11/25

   Due: 12/2-9

Please note that the exact project deliverables as well as the due dates may change each semester (and sometimes within a semester). It is imperative that if the class is requested to follow a different development model (e.g., the iterative model), then the project deliverables will be different.