Rubric for Assessment of CSE Junior Project Courses

This rubric is very much under construction!

Background: This rubric is intended to help assess key aspects of some of the CSE program outcomes that the junior project courses (CSE 3901, 3902; CSE 3903 is not offered regularly) contribute to. The instructor of any section of CSE 3901 or 3902 may use this rubric to provide (formative) feedback to students in the project teams in the course about the levels of their achievement of the abilities represented by the various dimensions below or for grading purposes or for both. In any case, once each year, the coordinator of each course should provide, to the Undergrad Studies Committee, the rubric results for all the teams (with a code name for each team) in one section of the course; this means, given that there are typically 30 students in a section and each team consists of 5 students, that each year the committee will get 6 completed rubrics from each of 3901 and 3902. That will allow the committee to discuss the extent to which students in the program are achieving these important outcomes, identify any weaknesses, and, if necessary, come up with ideas for possible changes, possibly in other parts of the curriculum, to address them.

It is worth mentioning that while the main goals of CSE 3901 and 3902 are very similar to each other, there are also some important differences between the two courses. One main difference is that 3901 includes "technology teams" (which are separate from the project teams) with each team researching a relevant system or tool and making a presentation to the class about its findings whereas 3902 does not include such a component. The rubric below is designed with this difference in mind. (Another difference between the two courses is that while student teams in 3901 work on a series of small projects that are somewhat independent of each other (with the final project being somewhat more comprehensive), 3902 teams work on a single large project which is broken up into several small pieces. This difference does not have a direct effect on the dimensions in this rubric. Of course, another key difference is that 3901 concerns webapps whereas 3902 deals with interactive computer games; the dimensions in this rubric are phrased in such a way that they are equally appropriate for both courses as well as 3903, which deals with system-level software, if and when we offer that course again.)

The rubric includes six dimensions, specified in the first column of each of the six rows in the table below. Each dimension is assigned a score of 1 through 4, these values representing increasing degrees of achievement as described in the next four columns of each row. The instructor should assign, in the last column, a value between 1 and 4 --fractional values okay-- for each dimension. Additional comments may be noted at the bottom.

Each of the six dimensions below is related to one or, in some cases, several of the *student outcomes*, (a) through (n), of the CSE program. The numbers corresponding to the particular outcomes related to each dimension are listed in the first column along with the respective dimension.

**
For convenience, the BS-CSE outcomes are listed below; this will be removed from this rubric once the rubric is finalized.
BS-CSE student outcomes: Students in the BS-CSE program will attain:

a. an ability to apply knowledge of computing, mathematics including discrete mathematics as well as probability and statistics, science, and engineering;
b. an ability to design and conduct experiments, as well as to analyze and interpret data;
c. an ability to design, implement, and evaluate a software or a software/hardware system, component, or process to meet desired needs within realistic constraints such as memory, runtime efficiency, as well as appropriate constraints related to economic, environmental, social, political, ethical, health and safety, manufacturability, and sustainability considerations;
d. an ability to function on multi-disciplinary teams;
e. an ability to identify, formulate, and solve engineering problems;
f. an understanding of professional, ethical, legal, security and social issues and responsibilities;
g. an ability to communicate effectively with a range of audiences;
h. an ability to analyze the local and global impact of computing on individuals, organizations, and society;
i. a recognition of the need for, and an ability to engage in life-long learning and continuing professional development;
j. a knowledge of contemporary issues;
k. an ability to use the techniques, skills, and modern engineering tools necessary for practice as a CSE professional;
l. an ability to analyze a problem, and identify and define the computing requirements appropriate to its solution;
m. an ability to apply mathematical foundations, algorithmic principles, and computer science theory in the modeling and design of computer-based systems in a way that demonstrates comprehension of the tradeoffs involved in design choices;
n. an ability to apply design and development principles in the construction of software systems of varying complexity.

**

Course number, semester: ____

Project Team's Code: ________________________________

	1	2	3	4	**Points assigned**
Ability to design, implement, and evaluate a software system to meet desired needs within relevant constraints. Outcomes: (c,e,l,m,n)	Poor design focused on minimally meeting the functional requirements; Implementation seems buggy; Little or no attention paid to questions related to memory usage, response time, etc.	Acceptable design that meets most functional requirements; Implementation mostly bug-free; Takes some account of some key constraints; Design/implementation seems brittle and not to built to evolve.	Satisfactory, flexible design meeting all functional requirements; Good bug-free implementation; Accounts for several important constraints.	Excellent design and superb implementation; Meets all functional requirements; flexible design can accomodate potential future changes; Takes careful account of all key constraints.	
Ability to design and conduct experiments to test software systems and interpret results to debug and improve system. Outcome: (b)	No systematic testing of software to ensure reasonable coverage of possible cases; No testing of system performance.	Team's testing approach provides basic essential coverage of possible cases; Team has also tested key aspects of system performance to a limited extent.	Carefully designed set of test cases to cover a suitable range of situations; Careful testing of system performance with respect to key factors such as network traffic as well as user response in different conditions.	Well-designed, systematic test suite providing excellent coverage assessing system performance for both typical and extreme cases; The suite is designed to test system performance with respect to all important factors, including network traffic, memory usage, with the results being used to tune portions of the system.	

Ability to use modern techniques and tools, including version control systems, communication tools, standard documentation and testing practices, etc., necessary for success as a CSE professional.
Outcomes: (k,i)
	Minimal/irregular use of important professional tools;
	Trivial/no version control;
	Ad-hoc/haphazard testing;
	Sketchy/poor documentation.
		Moderate use of important professional tools; Commits, branches, tags, etc add some useful versioning information; Adoption of standard conventions for documentation and testing, but inconsistent application.	Effective use of standard professional tools;
			Many features of version control used to support development efforts; Robust integration of team communication tools; Standard conventions respected for documentation and testing.
				Excellent use of professional tools and systems; Fully leveraging all features of version control, testing frameworks, documentation systems, etc;
				Seamless use of many tools within the development toolchain, eg. continuous integration.	

Ability to account for relevant social and ethical considerations in the design of software systems. Outcomes: (f,h,j)	No attention paid to social/ethical considerations. Team apparently did not even consider the question of potential social and ethical implications of the system.	Minimal consideration of social and ethical implications of system that might arise in extreme cases/situations.	Reasonable attention paid to social and ethical implications of system with all typical use case scenarios for the system being accounted for; Some consideration of potential harm that may be caused in extreme cases.	Excellent analysis of general ethical issues related to system and its impact on society as well as analysis of the system with respect to the requirements of the ACM/IEEE Code.	

Work effectively in software development teams. Outcome: (d)	Dysfunctional team; Members blamed each other for problems in project; Team spirit completely lacking.	Team functioned at minimal level of effectiveness; Members concentrated on distinct parts of system without concern for impact on other members' work.	Generally effective team; Members interested in presenting a positive picture of the team's work; Team members had a broad idea of other members' work on project.	Very effective team; Team members not only worked as a cohesive unit during design and development of the system but also went out of the way to assign appropriate credit to each member for his/her contributions to various aspects of the project.	
Ability to engage in effective written communication. Outcome: (g)	Documentation consisted of little more than (poorly commented) system code.	Documentation partly effective at conveying the technical aspects of system; Rationale for design choices, testing approach, etc., unclear; Skimpy user manual; Information future teams may need to evolve system lacking.	System documentation clearly presented all important aspects of project: design and implementation details, details of test scripts etc. Well-written user manual.	Excellent documentation of all aspects of the system including design and implementation choices, relevant code details, processes and tools used, and test scripts, all described in a structured and integrated manner; Information to enable future designers to evolve system included; Well-designed user manual; Illustrations, graphics, and layout executed to excellent effect.	
(Only for CSE 3901) Ability to engage in effective oral communication. Outcome: (g)	Presentation not effective; Even the problem being addressed by the technology/ service was not clear; Responses to relatively simple questions were often unclear.	Presentation adequate at providing a basic explanation of the problem being addressed and essential details of the technology/service being presented; Audience questions were generally handled in an acceptable manner.	Presentation was effective, if somewhat uninspiring, at explaining the problem being addressed and important details of the technology/service being presented as well as some other tools/services that addressed the same problem; Responses to questions were reasonable although some went into too much technical detail.	The presentation was polished, informative and engaging; Both the importance of the problem being addressed as well as the essential details of the technology/service were effectively explained; In answering questions, the team provided the right level and type of detail.	
Comments:					