CALIFORNIA STATE UNIVERSITY
FULLERTON

COLLEGE OF ENGINEERING
AND COMPUTER SCIENCE

Program Performance Review

Master of Science in Software Engineering (MSE)

Online Program

Self-Study Report

March 30, 2019

Submitted By:

The MSE Program Committee

Table of Contents

I. DEPARTMENT/PROGRAM MISSION, GOALS AND ENVIRONMENT

1.1 Program Mission and Goals

Software Engineering is the application of a systematic, disciplined, and quantifiable approach to the development, operation and maintenance of software; that is, the application of engineering to software. Demand for high-quality software engineers is significantly increasing, as almost all systems need software to run on them, and as software systems become larger and more complex.

The *Master of Science in Software Engineering (MSE)* is an online degree program hosted by the Computer Science Department at California State University at Fullerton (CSUF). The program provides high quality graduate education in the essential discipline of software engineering. The MSE online program maintains the same class quality as found in the on-campus classes.

The *mission* of the MSE program is "to provide students with the basic body of knowledge in the application of a systematic, disciplined, quantifiable approach to software development; to the operation and maintenance of quality software in the areas of software process, software requirements, software architecture and design, software construction, software testing, software maintenance, software project management, software measurement, software process assessment and improvement, and software engineers' ethics". The *goals* of the MSE program are "to prepare individuals for careers as software engineers and managers in the industry and government agencies, as well as for advanced study as researchers in the software engineering areas".

This is in accordance with the Computer Science Department Mission and Goals:
"The mission of the Computer Science Department is to provide students with a strong fundamental knowledge of Computer Science and the practical skills to adapt as technology changes."

The mission and goals for the MSE program are also consistent with the mission of the Engineering and Computer Science (ECS) College:
"The mission of the College of Engineering and Computer Science (ECS) is simple. We want to educate engineers and computer scientists who will graduate with state-of-the art knowledge in their chosen field and are ready to embark on careers in industry and government, or proceed to acquire advanced degrees in their own or related fields."

The MSE program is designed also to implement the University's Mission, Goals, and Strategies:
"Learning is preeminent at California State University, Fullerton. We aspire to combine the best qualities of teaching and research universities where actively engaged students, faculty and staff work in close collaboration to expand knowledge.

Our affordable undergraduate and graduate programs provide students the best of current practice, theory and research, and integrate professional studies with preparation in the arts and sciences. Through experiences in and out of the classroom, students develop the habit of intellectual inquiry, prepare for challenging professions, strengthen relationships to their communities and contribute productively to society.

We are a comprehensive, regional university with a global outlook, located in Orange County, a technologically rich and culturally vibrant area of metropolitan Los Angeles. Our expertise and diversity serve as a distinctive resource and catalyst for partnerships with public and private organizations. We strive to be a center of activity essential to the intellectual, cultural and economic development of our region."

1.2 Changes and Trends in the Discipline and the Program's Response

In the early stages of computer science, computer scientists focused on problem solving and implementing solutions. As computer technologies have advanced rapidly and the use of computers have variously diversified, the focus of computer scientists has shifted to the methodologies in the development of large scale, high quality software. Furthermore, with increasing global market competitiveness, the software industry has encountered several challenges, including how to adapt to technological and market changes, how to deal with increased software complexity and scale, how to meet customer demand for quality software and fast delivery of the product, how to outsource for software cost reduction, and how to face market risks in the global economy.

In order to deal with these challenges, the software industry demands quality software engineers who have, not only traditional technical-oriented knowledge but also, process-oriented knowledge and skills; who understand quality, cost, and time constraints in producing high quality software; who understand that, to be competitive in the fast changing global market, high quality software needs to be produced at minimized costs and within deadlines.

In response to these industry and environmental changes, the MSE program was designed for working professionals in the field of computer science who wish to further their skills and pursue graduate-level applied and theoretical knowledge of process-oriented software engineering. The program emphasizes a comprehensive and thorough process-oriented approach to software development that is fundamentally grounded in software engineering research, theory, principles and practices. The emphasis on software with a process-oriented curriculum makes this program unique among many other master's degree programs in software engineering.

The program successfully meets industry and student demand for substantial curricular changes. There have also been further changes in the program curriculum since the program started, namely the adoption of new trends of software development methods, such as agile processes.

1.3 Future Program Priorities

The MSE program committee consists of full-time faculty members who have taught MSE courses. The MSE committee meets regularly to discuss the curriculum, course contents, course schedule, recruitments, improvement in student learning, teaching effectiveness, student concerns, and possible curriculum changes and enhancement. The MSE program also has the Industrial Advisory Board (IAB), which consists of industry professionals in the field, including alumni. The MSE committee often meets IAB members to discuss current market trends and to gain feedback to ensure that the program curriculum is up to date. The most recent IAB meeting was in December 2018. We obtained feedback on our curriculum and course contents, and our instructors are working to incorporate their suggestions.

We believe that continuous improvement of the quality of the program is the key to providing students with the best education. Accordingly, we will improve the program quality and student learning by developing better lecture materials and methodologies, building a student network, providing better service to students, and establishing a process for smoother program operation. Another important priority for the program is to increase enrollment; or, at the very least, to maintain a stable enrollment, depending on the availability of resources. So far, the enrollment for the program has been successful in maintaining at least two sections per course (see Table 5 and Table 6 in Appendix II); and we will actively seek to increase enrollment in the future.

Our current faculty members in the program have excellent theoretical and practical knowledge on the subject areas they are teaching. To improve the teaching quality, each faculty member will have to stay current in the areas of their expertise by researching, receiving industry related training, attending technical conferences, and putting more effort in developing their teaching materials.

To improve the student learning, it is important for faculty members to utilize current technologies, employ a more hands-on approach, perform quantitative measures and analysis of student learning, and provide students with online and/or offline seminars.

A student network is particularly important for classmates and alumni of online programs, as they have fewer opportunities to meet each other than do students of offline programs.

Providing proactive student service is necessary for resolving student issues and concerns in a timely manner. According to our experience, without proactive service, students have a tendency to panic when they are in trouble.

For smoother program operation, establishing a process with clearer procedures and policies is important, especially when there are changes to the program coordinator or faculty members who are teaching in the program.

Increasing enrollment through more effective marketing and promotion for the program is also very important in sustaining the program.

In addition, we need to monitor market trends to act timely and make any necessary curriculum changes based on the feedback from the program advisory board. The MSE program has an advisory board that consists both of internal and external program committees. The internal committee consists of the department chair, program coordinator and faculty members who teach in the program. They meet several times throughout the year to discuss the course schedule, recruitment, improvements to student learning and teaching effectiveness, student concerns, and possible curriculum changes and enhancement. The external committee consists of professionals from the industry and graduated students from the program (Industrial Advisory Board (IAB)). The board meets when necessary to discuss current market trends and gain feedback to ensure that the program curriculum is up to date.

1.4 How the Special Session Self-Support Mode Is Included in the Mission, Goals, and Priorities of the Program

Not applicable.

Although we have a very similar program, the Accelerated Master of Science in Software Engineering (AMSE), it is run, scheduled and administrated separately by Extension and International Programs (EIP, formally University Extended Education (UEE)). We, MSE, are administered by the College of Engineering and Computer Science (ECS), and keep data separately.

II. DEPARTMENT/PROGRAM DESCRIPTION AND ANALYSIS

2.1 Program Description

The MSE program emphasizes process-oriented software engineering. The program offers a 30-unit coursework (10 courses) to be completed over a 22-month long, year-round program, in which students take two courses per semester, including one summer session. The program is offered in the cohort format, meaning that students in the same cohort follow the same course schedule throughout the program. The courses offered in the program are as follows:

Core Courses:

- CPSC 541 Systems and Software Standards and Requirements
- CPSC 542 Software Verification and Validation
- CPSC 543 Software Maintenance
- CPSC 544 Advanced Software Process
- CPSC 545 Software Design and Architecture
- CPSC 546 Modern Software Management
- CPSC 547 Software Measurement
- CPSC 548 Professional, Ethical and Legal Issues for Software Engineers

Capstone Projects:

- CPSC 597 I Graduate Project in Computer Science
- CPSC 597 II Graduate Project in Computer Science

Software engineering emphasizes “4 P”s – Project, Product, Process and People.
The goal of software engineering is to produce a software product under a project, using a process by people. The following table [Table 1] shows how our curriculum covers the 4 Ps.

Table 1: Curriculum Coverage for Software Engineering Elements

	Project	Product	Process	People
CPSC 541		x	o	
CPSC 542		o	x	
CPSC 543		o	x	
CPSC 544		x	o	
CPSC 545		o	x	
CPSC 546	o		x	x
CPSC 547	x		o	
CPSC 548	x			o
CPSC 597 (I,II)	o	x	x	x

[Legend] o: direct coverage, x: indirect coverage

All of these courses are also offered in the regular computer science program, except CPSC 548. Thus, students who miss a course(s) in their cohort year have the opportunity to take the regular course(s) if they stay close to the campus and transfer the credits to their study plan.

The culminating activity of this program will be the capstone project (CPSC 597 I and CPSC 597 II) where each student applies his/her knowledge and skills to a medium or large scale software project, using a supportive theoretical and research framework. Students who complete the program will have a well-rounded understanding of the theory, research, implementation, and evaluation of software process as it applies to a variety of applications.

In addition to the course requirements, new students are required to complete a virtual orientation session prior to the beginning of the program and are strongly recommended to attend the in-person orientation. Current students are strongly recommended to participate in the midpoint workshop or a virtual midpoint session, in order to assess their progress, to discuss their concerns or issues with courses or the program, and to provide overall feedback on the program. The year-round class schedule over a 22-month period is as follows:

Year 1:
New student orientation

Fall semester:	CPSC 541 and CPSC 544
Spring semester:	CPSC 546 and CPSC 545
Summer semester:	CPSC 548 and CPSC 597 I

Year 2:
Midpoint workshop

Fall semester:	CPSC 542 and CPSC 547
Spring Semester:	CPSC 543 and CPSC 597 II

2.2 Substantial Curricular Changes

Since the beginning of the program, we made a few changes in the course contents and course-offering schedule. Besides these, there have been no substantial curricular changes; no programs have been discontinued.

The title of Computer Science 544 was changed from Software Process to Advanced Software Process, to prevent confusion with the undergraduate course Computer Science 466 (Software Process). The course contents were expanded to include much more advanced and current topics, including agile processes, Scrum and XP.

The title of Computer Science 546 was changed from Software Project Management to Modern Software Management in order to better reflect what is covered in the course. Traditionally, this course had only covered subjects within the topic of software project management. Recently, this area has been expanded to include additional subjects, such as infrastructure, quality, process, external factors, etc. To meet writing requirements, this course requires project reports, which involve substantial writing.

The schedules for Computer Science 542 (Software Verification and Validation) and Computer Science 597 I (Graduate Project in Computer Science) were switched. Computer Science 542 (Software Verification and Validation) had been offered during the summer and Computer Science 597 I (Graduate Project in Computer Science) had been offered in the fall semester. Computer Science 542 is now offered in the fall and Computer Science 597 I is now offered during the summer session. The MSE committee made this change to provide students with enough time to digest rigorous subjects on software verification and validation and to allow more time for them to think and focus on their graduate projects before they take on the second part of their capstone project, Computer Science 597 II, in the spring semester.

2.3 Student Demand for the Program

Everyone needs software nowadays. The number of software development jobs is increasing every year. According to the Bureau of Labor Statistics, the number of such jobs in 2016 was 1,256,200. The projected percent increase in software developers' employment from 2016 to 2026 is 24%, much greater than the 7% average growth rate for all occupations. The number of jobs for computer and information systems managers in 2016 was 367,600. The projected percent increase in the number of such jobs from 2016 to 2026 is 12% (also greater than average). [Source: https://www.bls.gov/ooh/computer-and-information-technology/software-developers.htm]

According to the STEM State-Level Analysis, done by the Center on Education and the Workforce at Georgetown University, 17% of all MA jobs in California will be in a STEM field by 2018; and 49% of STEM jobs in California will be in Computer Occupations (computer technicians, programmers, and scientists) by 2018. [Source: https://cew.georgetown.edu/wp-content/uploads/2014/11/stem-states-complete-update2.pdf]

According to “Engineering by the Numbers,” authored by Brian L. Yoder, the number of master's degrees awarded has increased from prior years. In 2017, 64,602 master's degrees were awarded, up from 62,596 in 2016. Master's degrees in Computer Science (inside Engineering) increased from 4,924 in 2008 to 10,602 in 2017. [Source:

https://www.asee.org/documents/papers-and-publications/publications/college-profiles/2017-Engineering-by-Numbers-Engineering-Statistics.pdf]

We have seen increased student demand for the program in the past years. We currently maintain two sections for each course with class sizes between 25 - 35 students. Demand for the program is still high, and our active and effective marketing and promotion for the program will attract even more students. Appendix VII shows more details on student demand. The feedback we received through multiple surveys from current students and graduates of the program indicate a high student satisfaction rate for the relevance of knowledge and skills learned from this program, the faculty quality and the program overall (for more details, see Appendix VII).

2.4 Enrollment Trends for the Program

Enrollment for the program has been consistent and stable (see Table 5 in Appendix II). Demand for the program is strong, and the possibility of increasing enrollment will depend on active and effective marketing and promotion for the program as well as on our resources for increasing capacity. More information related to enrollment can be found in Table 5, Table 6-A, and Table 6-B in Appendix II.

2.5 Plans for Curricular Changes

At this point, we have no plan to make substantial changes in the curriculum. We will continually monitor and evaluate the program quality and market trends. We are open to adapting to any necessary changes in the future. Individual course contents are updated by the instructors, based on the change and advancement of specific areas, to stay current in the field. We also have a good relationship with the Software Engineering Institute (SEI) at Carnegie Mellon University. Our faculty members frequently attend special programs offered by the SEI to update their knowledge and techniques, to keep pace with the industry’s needs and direction (in addition to their own research).

One of the most important tasks for improving program quality and student learning will be to build a proper infrastructure for adopting more hands-on activities in each course; for supporting students in hands-on activities in various realistic scenarios. Building this infrastructure will take a significant amount of time, resources and effort.

2.6 Special Sessions

Not applicable.

III. DOCUMENTATION OF STUDENT ACADEMIC ACHIEVEMENT AND ASSESSMENT OF STUDENTS LEARNING OUTCOMES

3.1 Program Assessment Plan

Organization of Assessment

The assessment team (a.k.a. the assessment committee) of the Master of Science in Software Engineering (MSE) program consists of at least three faculty members, selected among the faculty who have been teaching MSE courses. MSE faculty involvement in almost all program activities, including assessment activities, by convention, is done voluntarily.

Assessment activities in the MSE program are planned and implemented by the MSE assessment team, approved by the MSE faculty/committee, and executed continuously by all participating faculty members.

We seek consensus by the whole team (agreement based on enough discussions) before making a decision; majority vote is used only if consensus is not achieved.

Assessment data consists of Direct Evidence (evidence collected from components in participating courses, such as exams, homework, projects, etc.) and Indirect Evidence (evidence collected to support the validity of Direct Evidence, such as course exit survey, program exit survey, alumni survey, new student survey, etc.).

Assessment data (information involved in all assessment activities, including data collection, data analysis, action plans, action plan implementation, etc.) are stored in the MSE asset library (data storage), which is managed by the MSE Coordinator and the MSE office. The MSE Coordinator facilitates configuration management for assessment and analysis data, so that the assessment team can perform their functions properly.

Currently we use Google Drive for assessment data collection, and use Dropbox to save backup data. Authentication and authorization of different access levels are used to keep confidentiality of data. Faculty members are given Read/Write access rights (for granting privileges to perform necessary tasks) depending on their roles in the program. Compliant Assist is used to report assessment results to the university every year.

MSE assessment process, assessment cycle and assessment plan are shown in the following sections.

Assessment Process

MSE assessment activities are done systematically based on the preplanned process. The following diagram [Figure 1] shows the MSE assessment process.

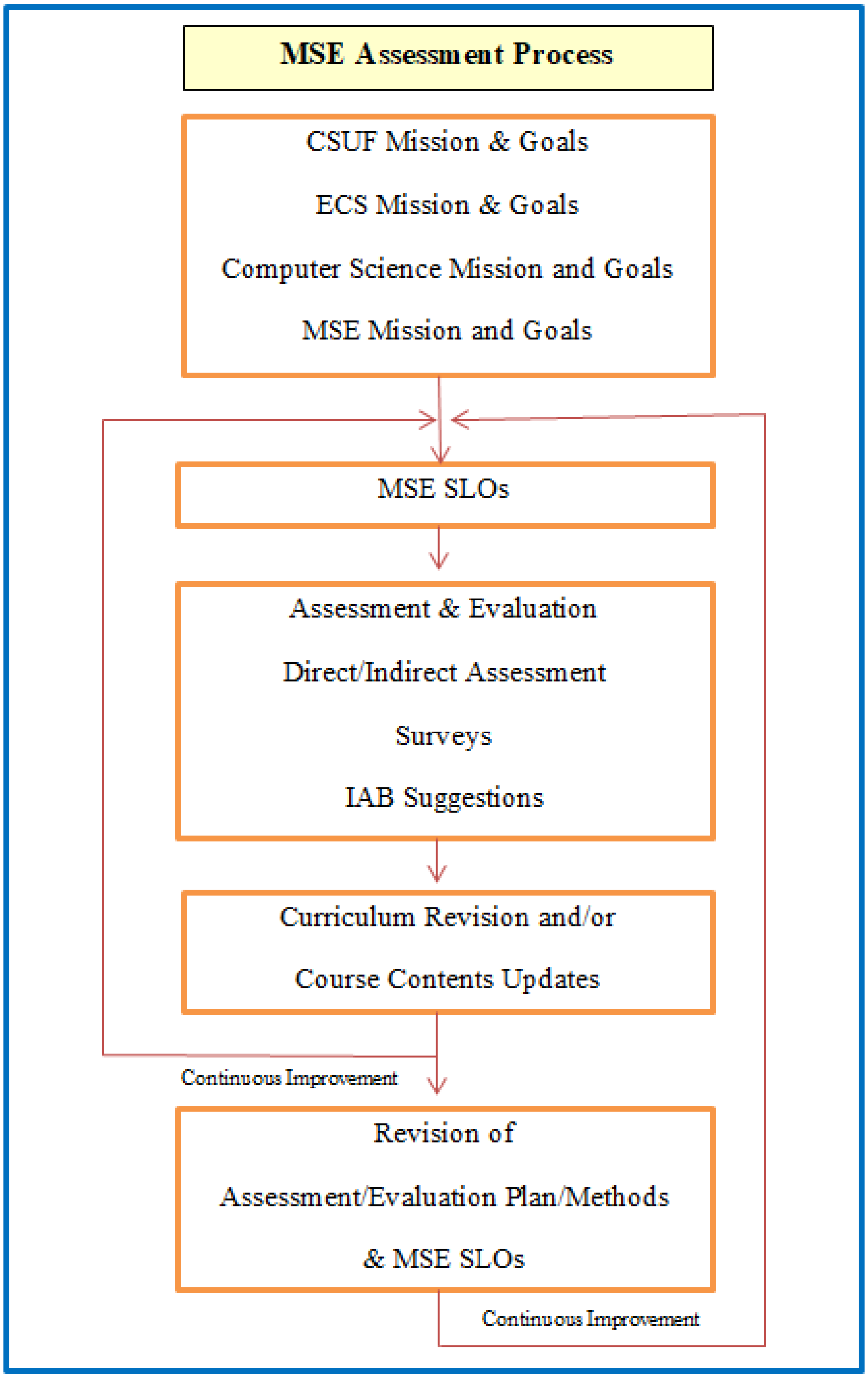


Figure 1: MSE Assessment Process

Assessment Cycle

The MSE assessment is done every academic year (as planned in the next section). Every year, before the end of each semester, the MSE Coordinator facilitates assessment data collection methods and data storage, and informs all participating faculty members. Every year, assessment data is collected by faculty members assigned corresponding courses (for Direct/Indirect Evidence) and by the MSE Coordinator (for some Indirect Evidence).

Evidence data is analyzed and evaluated by the assessment team. Based on assessment evaluation results, if necessary, improvement action plans are built by corresponding faculty members, in consultation with the assessment team, and action plans are implemented by faculty members assigned to the corresponding courses in the following years. The MSE coordinator facilitates proper transferring of necessary information about participating courses to their faculty members.

If necessary, revision of curriculum and revision of course contents are done based on the assessment evaluation results every year. Revision and adjustment of MSE assessment plan and methods, SLOs, and curriculum mapping are done every six years.

The following table [Table 2-A] shows current/future assessment activities.

Table 2-A: MSE Program Assessment Activities

Year	**Assessment Activities**
AY 2015 - 2016	Assessment of SLO, Analyze Data, Build Action Plans, Implement Action Plans
AY 2016 - 2017	Assessment of SLO, Analyze Data, Build Action Plans, Implement Action Plans
AY 2017 - 2018	Assessment of SLO, Analyze Data, Build Action Plans, Implement Action Plans
AY 2018 - 2019	Assessment of SLO, Analyze Data, Build Action Plans, Implement Action Plans
AY 2019 - 2020	Assessment of SLO, Analyze Data, Build Action Plans, Implement Action Plans
AY 2020 - 2021	Assessment of SLO, Analyze Data, Build Action Plans, Implement Action Plans Evaluation and Adjustment of Assessment Plan/Methods, SLOs, Curriculum Mapping

Assessment Plan

The MSE assessment is performed as follows [Table 2-B].

Table 2-B: MSE Program Assessment Plan

SLOs	When to Assess	What Evidence to Collect (Measures & Strategies)	Who Will Collect	How Assessed	How “Closing the Loop” Decision Made	How Assessment Results Will Be Used /Acting on Assessment
SLO-1 Professional Ethics	Yearly when corresponding course offered	Exams, homework, projects	Instructor	At least 2/3 (66%) satisfactory & developing results	Review by assessment committee and program coordinator	Enforce weak contents
SLO-2 Comprehension and Communication	Yearly when corresponding course offered	Exams, homework, projects	Instructor	At least 2/3 (66%) satisfactory & developing results	Review by assessment committee and program coordinator	Enforce weak contents
SLO-3 Practices, Process, Assessment and Improvement	Yearly when corresponding course offered	Exams, homework, projects	Instructor	At least 2/3 (66%) satisfactory & developing results	Review by assessment committee and program coordinator	Enforce weak contents
SLO4 Collaborative Work	Yearly when corresponding course offered	Exams, homework, projects	Instructor	At least 2/3 (66%) satisfactory & developing results	Review by assessment committee and program coordinator	Enforce weak contents
Course Survey	Yearly when corresponding course offered	Survey	Instructor	At least 2/3 (66%) satisfactory & developing results	Review by assessment committee and program coordinator	Course contents updates
New Student Survey, Exit Survey, or Alumni Survey	Orientation, midterm workshop, on graduation, or after graduation	Survey	Coordinator	At least 2/3 (66%) satisfactory & developing results	Review by assessment committee and program coordinator	Revision of curriculum, SLO, assessment plan/methods

3.2 Student Learning Outcomes (SLOs)

Student Learning Outcomes (SLOs)

The MSE program sets the following four Student Learning Outcomes (SLOs). It is well aligned with the missions and goals of the Department of Computer Science, the College of Engineering and Computer Science (ECS), and CSUF.

SLO-1: Professional, legal and ethical issues in software engineering
Be aware of professional, legal, and ethical issues in the software engineering area, and social impact. Be a responsible engineer to healthy society.

SLO-2: Software requirements comprehension and communication
Understand requirements clearly, and communicate with stakeholders in an appropriate manner. Be able to communicate properly through oral presentation and in a written form.

SLO-3: Software practices, process, assessment, and improvement
Be able to identify and use the best practices in various parts of software development. Be able to assess the current work process, and to improve continuously.

SLO-4: Collaborative work
Be able to work collaboratively with others. Be a good citizen in a team involved.

Performance Indicators (PIs)

The Student Learning Outcomes (SLOs) are assessed by measuring the following seven Performance Indicators (PIs).

COMM: Able to communicate clearly. Write a clear document with an appropriate tone. Deliver a clear presentation with an appropriate tone.
Satisfactory: Able to write and/or present very clearly
Developing: Able to write and/or present somehow
Unsatisfactory: Not able to write and/or present clearly

COOP: Cooperate effectively on a group project.
Satisfactory: Able to work cooperatively
Developing: Able to work with others somehow
Unsatisfactory: Not able to work with others well

PRAC: Exercise the best practices for design, coding, testing, maintenance, measurement, and management.
Satisfactory: Able to identify and use the known best practices
Developing: Able to identify the best practices and use somehow
Unsatisfactory: Not able to use the best practices

ETH: Demonstrate an understanding of professional ethics appropriate to the use or development of software engineering artifacts.
Satisfactory: Understand most of the ethical issues
Developing: Understand some
Unsatisfactory: Does not understand well

PROC: Demonstrate knowledge of a software engineering process (e.g. sequential, iterative processes).
Satisfactory: Able to understand and apply processes
Developing: Able to apply somewhat
Unsatisfactory: Not able to apply properly

SPEC: Ability to understand and specify the requirements properly
Satisfactory: Able to write a correct specification of the requirements in a proper manner
Developing: Able to write somehow
Unsatisfactory: Not able to write specification correctly

STD: Understand and use standards (e.g., ISO, IEEE, CMMI)
Satisfactory: Able to identify and use standards in appropriate areas
Developing: Able to use standards somehow
Unsatisfactory: Not able to use standard properly

How to Assess

The program coordinator will thoroughly assess students at the point of their application by considering their GPA, personal statement, resume, TOEFL scores (for non-English speaking foreign students), and, if necessary, their technology skills, based on an interview. Once students start the program, the common assessment strategies and methods used for assessing student learning and the program’s learning goals are as follows:

- Homework, quiz, chapter summary, examinations
- Online discussions
- Team project report and presentation
- Individual research/project/survey/case study paper or report and presentation
- Prototype demo
- Software tool evaluation
- Software project management practicum

- Software process or engineering workshop/seminar
- Final graduate project report/thesis
- Course survey, Exit survey, Alumni survey

Each learning goal is met by one or more courses and is assessed by one or more assessment strategies and methods. The following assessment matrix [Table 3] illustrates how each Student Learning Outcome (SLO) is met and assessed.

Table 3: Assessment Matrix for Student Learning Outcomes (SLOs)

Student Learning Outcomes (SLOs)	**Relevant courses**	**Assessment Methods**
SLO-1: Professional, legal and ethical issues in software engineering ETH	CPSC 548 Course Survey Exit Survey Alumni Survey	Homework, chapter summary, examinations, case study report, online discussions, team project report, presentation, surveys
SLO-2: Software requirements comprehension and communication COMM, SPEC, STD	CPSC 541 CPSC 543 CPSC 545 CPSC 546 CPSC 547 CPSC 548 CPSC 597 Course Survey Exit Survey Alumni Survey	Homework, quiz, examinations, online discussions, individual/team project report and presentation, case study report, prototype demo, surveys
SLO-3: Software practices, process, assessment, and improvement PRAC, PROC	CPSC 542 CPSC 544 CPSC 545 CPSC 546 CPSC 547 CPSC 597 Course Survey Exit Survey Alumni Survey	Homework, quiz, examinations, online discussions, individual/team project report and presentation, software tool evaluation, surveys
SLO-4: Collaborative work COOP	CPSC 542 CPSC 543 CPSC 544 Course Survey Exit Survey Alumni Survey	Online discussions, case study report, team projects, team research/survey paper, presentation, surveys

Each MSE faculty is responsible for teaching the course within the defined guideline of the course contents and learning goals. Within the defined guideline, the MSE faculty is free to develop his/her course materials and methodology to deliver the course contents and to assess student learning. Each course uses a multi-modal assessment methodology. The following section shows mapping of Student Learning Outcomes (SLOs) and Performance Indicators (PIs).

Mapping of Student Learning Outcomes (SLOs) and Performance Indicators (PIs)

The following shows what Performance Indicators (PIs) are used to assess which Student Learning Outcomes (SLOs).

SLO-1: Professional, legal and ethical issues in software engineering
ETH

SLO-2: Software requirements comprehension and communication
COMM, SPEC, STD

SLO-3: Software practices, process, assessment, and improvement
PRAC, PROC

SLO-4: Collaborative work
COOP

Student Learning Outcomes (SLOs) and Performance Indicators (PIs) are mapped to MSE courses [Table 4].

Table 4: MSE Curriculum Map

Curriculum Map (Mapping of Courses to SLOs and PIs)

MSE Courses	**Student Learning Outcomes (SLOs)**			
	SLO 1	**SLO 2**	**SLO 3**	**SLO 4**
541		SPEC, STD		
542			PRAC	COOP
543		COMM		COOP
544			PRAC, PROC	COOP
545		SPEC	PRAC	
546		COMM	PRAC	
547		COMM	PRAC	
548	ETH	COMM		
597		SPEC	PROC	
Course Survey (Indirect Evidence Supporting each course)	PIs assigned to the course	PIs assigned to the course	PIs assigned to the course	PIs assigned to the course
Exit Survey, or Alumni Survey (Indirect Evidence Supporting the program)	ETH	COMM, SPEC, STD	PRAC, PROC	COOP

3.3 How Assessment Results Have Been Used to Improve Teaching and Learning Practices, and Overall Program Effectiveness

When this program started, most instructors relied on traditional assessment strategies such as exams, assignments and projects. Instructors later realized the importance of other methods that are not typically used in a traditional classroom setting. Some of the methods that were developed later include online discussion, online presentation, reading assignments to encourage self-study, writing that requires critical thinking and problem solving, peer learning, technology-based collaboration, and communication.

According to our experiences so far during the past five years of the program, these assessment measures work well in evaluating many of the program learning goals. We will continue to develop new assessment strategies and methods. One possible method under development is to evaluate students' application skills in various real-world situations. For this method to work well, we need to build a proper infrastructure that simulates realistic projects in different types of organizations. This infrastructure could be used throughout the program for students' hands-on activities.

3.4 Other Quality Indicators Identified by the Program

According to the survey results collected from current students and graduates of the program, the overall student satisfaction for the program in terms of student learning and program quality are extremely high (for details, see figures in Appendix VII). The survey shows almost all (almost 100%) students are satisfied with this program. There are also many reports from students who mentioned that they got promoted or found new jobs mainly due to their MSE degrees.

Our program is also highly ranked by many ranking organizations (for details, see Appendix VII). We are 5th – 26th nation-wide in 2018-2019 rankings. Furthermore, all the schools that are ranked above us are only fully-funded Ph.D. granting research schools. Among non-Ph.D. granting universities, we are the 1st. In California, we are the 3rd, following USC and UCLA (Appendix VII).

After taking into account feedback from students and the industrial advisory board, we currently have no plans to make significant modification of the program. That said, we will continually seek more effective methods and techniques to deliver course content, for better student learning and assessment. One technique to improve student learning is to create lecture materials with visual presentation. A second is to establish a clear communication channel between student and instructor, to avoid any student confusion.

The MSE Program Committee also oversees the faculty in the program, in addition to the industrial advisory board.

3.5 How Student Learning Is Assessed in Online Formats

As discussed in the previous sections, the common assessment methods used for assessing student learning mostly include traditional assessment methods such as homework, quizzes, examinations, team projects, presentations, research papers, case studies and prototype demos; and non-traditional methods that are proper for online programs such as online discussion, readings, chapter summaries, preparing effective materials for online presentation using various technologies, peer learning, technology-based collaboration and communication, and workshops or seminars.

IV. FACULTY

4.1 Changes in the Program FTEF since the Last Program Review

The MSE program currently has five tenured faculty members (for more details, see Appendix IV for faculty data and Appendix VI for faculty CV's). All of them joined the program from the beginning and since then the program has been very stable. We maintain five Full-Time-Equivalent-Faculty (FTEF), and 56.0 – 66.6 Academic-Year Full-Time-Equivalent-Students (AYFTES) (see Table 9 in Appendix III).

The program generally has two cohorts, taking two classes in each semester including summer. The program offers a total of eight classes each semester that cover major areas of software engineering. These eight classes are regularly taught by the five MSE faculty members.

We heavily promoted faculty training in various areas outside the faculty's original expertise. Since each faculty member is trained in more than one area of the program, even if some members leave for their sabbatical or difference-in-pay, there will be no negative impact on student learning or on the sustainability of the program.

4.2 Priorities for Additional Faculty Hires

The Computer Science Department, which is the main base of the MSE program, has enough instructors to teach all the courses that are offered in the MSE program as well as the courses that are offered in the regular computer science degree programs (both for graduate and undergraduate). Moreover, the faculty of the program cover all areas of expertise; and each faculty member in the program has more than one area of expertise. Thus, even if a faculty member cannot teach the course due to leave or retirement, other faculty members can cover the course. We also have a strong pool of qualified alumni instructors. They have good work-experience and are in high positions in their own organizations, which enables them to serve as role-models and mentors to students.

If the program is expanded to offer more than three sections for each of the MSE courses, we may need to hire additional faculty members. Hiring additional faculty will depend on successful future recruitment and not on a change of the program.

Currently we offer ten courses for MSE, and all of them are required; none are elective. If in the future we begin offering elective courses (to meet students' preferences), we will need more faculty members. Elective courses can be implemented immediately and easily, if we have enough students to sustain the course offerings. Note, however, that elective courses may cause over or low enrollment.

4.3 Faculty Teaching in the Program

The MSE Program has five regular faculty members who have both a strong academic background and professional expertise. All the courses in the MSE program are mostly taught by these five full-time tenured faculty members. One fourth of an office staff member supports various office-work, including graduation check. One student assistant helps with office-work and recruitment activities.

A brief biography, including education, experience and the areas of expertise for each faculty is listed below (See more detailed CV's in Appendix III Faculty):

Dr. Ning Chen, Professor, Computer Science

Education: Ph.D., Computer Engineering. Colorado State University.

Related experience: Supervised numerous software projects, provided consulting to several companies, published numerous research papers related to software engineering.

Expert in the following areas: Software Testing, Verification and Validation, Software Architecture and Design, Embedded Systems, Enterprise Software.

Dr. Bin Cong, Professor, Computer Science

Education: Ph.D., Computer Science, University of Texas at Dallas.

Related experience: Supervised and managed numerous software projects; provided consulting to several companies; served as Chief Technology Officer, Angel Engineers, Sunnyvale, California; published research papers related to software engineering; served as a Software Engineering Institute (SEI) Certified CMMI High Maturity Lead Appraiser; and as a SEI CMMI Instructor.

Expert in the following areas:
Software Process Engineering, CMM and CMMI, Parallel and Distributed Processing, Computer Network, Network Security, Graph Embedding, Design and Analysis of Sequential and Parallel Algorithms, Neural Network, Heuristic Search.

Dr. James Choi, Professor, Computer Science

Education: Ph.D., Computer Science, University of Southern California.

Related Experience: Served as research associate, University of Southern California; conducted research in Software Engineering; supervised a number of software projects; developed VLS software in conjunction with Hughes Aircraft, Northrop, and Lockheed Martin; published numerous research papers related to software engineering.

Expert in the following areas: Reverse Software Engineering, Configuration Management, Software Maintenance, Software Ethics, UML And UML Development Methodology, SLC Model and SLC Development (Classical and OOM).

Dr. Chang-Hyun Jo, MSE Coordinator and Professor, Computer Science

Education: Ph.D., Computer Science, Oklahoma State University.

Related experience: Served as a Software Engineering Institute (SEI) Authorized CMMI Instructor; holds SEI Certificates on CMMI, Software Architecture Professional, Certificates in ATAM Evaluator, Certified Scrum Master; conducted numerous funded projects in software engineering for eighteen years, provided training and consulting for numerous organizations, participated as the Korean Delegate in the ITU-T SG10 international standardization for seven years funded by ETRI, Samsung Electronics, and Korea Telecommunications, served on the ACM SAC conference program committee for ten years, and serves on the IEEE Educational Activities Board (Professional Certificate Program for Body of Knowledge Software Engineering (SWEBOK)).

Expert in the following areas: Software Process (Definition, Assessment, Improvement), Software Analysis and Design, Software Architecture, Programming Language Design and Programming Environments (Compilers, Interpreters, Debuggers, Editors, CASE tools), CMMI, SCAMPI, ATAM, ISO 12207, ISO 15504 (SPICE), Agile methods such as Scrum and XP.

Dr. Christopher T. Ryu, Department Chair and Professor, Computer Science

Education: Ph.D., Computer Science, University of Houston, Houston, TX.

Related Experience: Supervised and managed numerous industry and research projects; received more than $700,000 in grants to support his research projects; published numerous articles in the related areas; has over seven years of industry experience with various companies and organizations, including Volt group, EDS; provided consulting to many local companies including LA Metro and Union Station in building software systems, finding business solutions, and IT infrastructure.

Expert in the following areas: Software Management, Software Design and Architecture, Artificial Intelligence, Machine Learning, Data Science, Computational Finance.

4.4 Information on Instructor Participation in Special Sessions Self-Support Programs Offered by the Program

Not applicable.

V. STUDENT SUPPORT AND ADVISING

5.1 Student Advising

The Computer Science Department requires all undergraduate students to receive mandatory advising each year on their study plan. The department faculty provides advisement on research and career opportunities.

The department first provides graduate students with advisement during the student orientation. Students are advised again when they create a study plan. After this, students can receive advisement when they need it.

In the MSE program, all students first receive advisement during the new student orientation. Unlike students in the regular computer science graduate program, students in the MSE program do not need to meet with their advisor or the program coordinator to create a study plan since there is already a set study plan for all MSE students. Although students can contact their advisor to discuss their issues, research opportunities, or seek career related advisement, the program coordinator is generally in charge of advising MSE students with the support of the staff members in the computer science department.

5.2 Student Participation in Honors Programs and Research with Faculty

The MSE program in the Computer Science Department does not offer any Honor programs; there is, though, a nation-wide honors club, called UPE (Upsilon Pi Epsilon).

The department encourages both undergraduate and graduate students to work with faculty by participating in research projects. Although very few undergraduate students participate in research with faculty, it is common for graduate students to collaborate with faculty by participating in research projects and then publishing their papers before they graduate. Though it is more challenging for MSE students to collaborate with faculty for research, some students and faculty work together on research projects and publish their papers.

The following list of papers exhibit some of the outcomes of their collaborative research during the past year:

N. Chen, E. Chen and I. Chen, “Integrating Software Testing Standard ISO/IEC/IEEE 29119 to Agile Development,” Int’l Conf on Software Engineering Research & Practice (SERP2018), Las Vegas, USA.

C. Lo and N. Chen, “IEEE 42010 and Agile Process—Create Architecture Description through Agile Architecture Framework,” Int'l Conf on Software Engineering Research & Practice (SERP2017), Las Vegas, USA.

R. Subramania, N. Chen and T. Zhu, “Behavior Driven Test Automation Framework,” Int'l Conf on Software Engineering Research & Practice (SERP2017), Las Vegas, USA.

K. Bhalerao and N. Chen, “Principles of Continuous Integration (CI) in Practice,” Int'l Conf on Software Engineering Research & Practice (SERP2016), Las Vegas, USA.

VI. RESOURCES AND FACILITIES

6.1 State and Non-state Support and Resources Received

The MSE program is state supported. The program initially received funding for the development of the degree proposal and course materials for all ten courses. At that time, each faculty member in the program received 3-units of release time and $5,000 in grant money.

The state funding for faculty positions in the College of Engineering and Computer Science corresponds to an FTES target, based on an SFR of an approximately thirty to one ratio. Except for the first few years, the MSE program has had an average of thirty students in each class for eight sections of four courses, meeting the SFR target.

In addition to the funding of faculty positions, the MSE program also collects distance-learning fees that are $33 per unit ($99 per course for a student). The distance-learning fee significantly helps support various program activities, faculty training and conference attendance, workshop and seminar, and the purchase of necessary equipment and software to support faculty research as well as the operation of the program. Such budget allocated to the MSE program is shown in Table 10 (in Appendix IV).

6.2 Special Equipment Used by the Program

The MSE program has purchased various equipment including necessary hardware and software to support faculty instruction; computers, audio and video recorders, servers, backup tools, and other tools necessary to support instructional needs; and faculty research that is related to the areas covered in the program.

The MSE program has used the online learning tools (Blackboard, Moodle (Titanium)) that are maintained by the IT Department in the university.

The Computer Science Department is a member of multiple academic alliances with several companies including Microsoft, IBM, Oracle, Adobe and Dropbox. Students and faculty can freely use major software from these companies for research and instructional purpose. In some cases, the program may need to purchase other tools and software for individual faculty member’s research and instructional purpose. In addition, the program may also need to purchase several servers, software systems, and tools to build an infrastructure to support hands-on activities for students.

6.3 Library Resources

The MSE faculty and students use electronic journals as well as books. The CSUF library subscribes to the database that is provided by major computer science professional organizations, such as Association for Computing Machinery (ACM) and Institute of Electrical and Electronics Engineers (IEEE) Computer Society, which provide journal articles, conference proceedings, standards, and books. The resources that are provided by the CSUF library and the professional organizations cover student and faculty needs.

VII. LONG-TERM PLANS

7.1 Summary of Long-term Plan

The MSE program's long-term plans include continuous assessing and improvement of program quality and student learning, increasing enrollment, establishing a process for program operation and transition, implementing better measurements.

To continuously improve the program quality and student learning, we will improve the lecture quality by utilizing various methods and technologies; staying current in the technologies by attending special training, workshops, or conference and being active in research; and keeping close relationships with industry. In addition, for a better student learning experience, we plan to build a proper infrastructure that simulates real-world environment for hands-on activities that are related to subjects throughout the program.

To maintain or increase enrollment, we plan to promote the program by utilizing search engines (e.g., Google) and SNS (e.g., Facebook, LinkedIn, etc.); visiting local companies (e.g., Raytheon); utilizing student and alumni network; hosting training and certification programs on specific software engineering subjects; hosting seminars and conferences on special topics in software engineering; and partnering with computer science departments in other countries. The MSE program regularly sponsors seminars on campus, invites prominent speakers, and recruits students.

We have established many procedures and policies to standardize MSE operation and we will maintain these protocols to ensure smooth program operation. To better measure the program quality and student learning, we have defined and will refine proper standards that reflect the learning goals, we will continue to collect the data through expanded questions in surveys and possibly from the final exams of each course, and to analyze the data to assess how well we are achieving the learning goals of individual courses and the goals of the overall program.

7.2 Long-term Plan for Implementation of the University's, Department's and Program's Mission and Goals

The long-term plans for the MSE program, as described in Section 7.1, are mainly to improve the quality of program and student learning. The implementation of these plans will also support the University's Mission as well as the Computer Science Department's Mission, as described in Section 1.1.

7.3 Potential Evidence in Support of Goals

Evidence that we are successfully implementing our plan for improving program quality and student learning includes: high student and industry satisfaction with the program, high student quality, student excellence in academic achievement, and high graduation rates. Compared to other departments, we have maintained relatively higher rates of graduation in 2-4 years (72.6% - 90.4%) (see Table 7 in Appendix II). We will continue delivering high rates.

Consistent enrollment is good evidence of program quality and, as a result of continuous recruitment effort and curriculum improvement, we have maintained and will continue to maintain consistent enrollment (see Table 5 and Table 6 in Appendix II).

Evidence of successful implementation of a student network includes the size of the network and effective use of the network for student announcements and announcing possible programs.

Evidence of successful implementation of program operation and transition process can be found in written documents about procedures, policies, and other necessary materials. A more easeful transition provides additional evidence.

Evidence of successfully measuring program quality and student learning can be found in clearly written operational definitions on measuring and collecting the assessment data; analysis of the measured data from extensive questions in surveys or exams; appropriate interpretation of the analyzed data that indicate the level of program quality and student learning; and performance of continuous improvement based on the measured and analyzed data. With effective assessment, we will continuously improve the program.

7.4 Long-term Budget Plan

Table 12 (in Appendix IV) shows the program's budget plan over the next five years, based on expected, consistent enrollment (Table 11 in Appendix IV). The enrollment numbers are the total number of students from two cohorts, taking two classes, each of which consists of two or three sections. The large portion of funds, other than from the state, is from distance-learning fees. Expenses include the two courses of release time, paid to the program coordinator, food and material expenses for various program events, costs for program promotion, equipment, technology, faculty training and research support, and a student assistant. The slight increase in expense is applied to reflect enrollment and inflation. Salaries for faculty who teach in the program are covered through the generated FTES.

VIII. APPENDICES Connected to the Self-Study (Required Data)

[Required Appendices]
Appendix I. Undergraduate Degree Programs (Not Applicable)
Appendix II. Graduate Degree Programs
Appendix III. Faculty
Appendix IV. Resources
Appendix V. Long-term Planning
Appendix VI. Curriculum Vitae of Faculty

[Supplementary Appendices]
Appendix VII. Survey Results for Program Assessment
Appendix VIII. Course Descriptions
Appendix IX. Glossary and Formulas Used in the Report

Appendix I. Undergraduate Degree Programs

This part is not applicable to this program performance review (i.e. Table 1 – Table 4 are not applicable).

Appendix II. Graduate Degree Programs

This is to document academic achievement of the MSE program. This is based on the data provided by the Office of Assessment and Institutional Effectiveness [March 2019]

TABLE 5. Graduate Program Applications, Admissions, and Enrollments

Table 5 shows the number of student applications, number of students admitted, the percentage of students admitted, the number of new enrollments, and the percentage of new enrollments. Percentage of students admitted is equal to the number of students admitted divided by the number of students who applied. Percentage of students enrolled is equal to the number of students enrolled divided by the number of students admitted.

TABLE 5. GRADUATE PROGRAM APPLICATIONS, ADMISSIONS, AND ENROLLMENTS

Academic Year (AY)	Applied	Admitted	Enrolled
2013-2014	133	119	62
2014-2015	72	62	48
2015-2016	78	66	50
2016-2017	95	78	61
2017-2018	102	89	69

TABLE 6. Graduate Program Enrollment in FTES

Table 6-A and 6-B show student enrollment data for the past five years.

TABLE 6-A. GRADUATE PROGRAM ENROLLMENT IN FTES

Academic Year	Enrollment in FTES
2013-2014	56.0
2014-2015	50.4
2015-2016	56.8
2016-2017	66.6
2017-2018	77.6

TABLE 6-B. GRADUATE PROGRAM ENROLLMENT IN HEADCOUNT

Academic Year	Headcount	FTES per HC
2013-2014	123	0.5
2014-2015	107	0.5
2015-2016	95	0.6
2016-2017	111	0.6
2017-2018	128	0.6

TABLE 7. Graduate Student Graduation Rates

Table 7 shows the graduation rate for master's-seeking students.

TABLE 7. GRADUATION RATES FOR MASTER'S-SEEKING STUDENTS

Beginning year in MSE program	Headcount	% graduated in 2 years	% graduated in 3 years	% graduated in 4 years
2012	52	88.5	90.4	90.4
2013	62	72.6	77.4	79.0
2014	48	75.0	75.0	75.0
2015	50	84.0	90.0	90.0
2016	61	77.0	NA	NA

TABLE 8. Master's Degrees Awarded

Table 8 shows the number of Master's degrees awarded by the program.

TABLE 8. MASTER'S DEGREES AWARDED

College Year	Degrees Awarded
2012-2013	40
2013-2014	58
2014-2015	52
2015-2016	39
2016-2017	48

Appendix III. Faculty

TABLE 9. Full-Time Instructional Faculty

For the six most recent academic years, Table 9 shows the number of tenured faculty, faculty on tenure track, faculty on sabbatical, faculty in FERP, lecturers, full-time faculty equivalents (FTEF), and academic year full-time student equivalents (AY- FTES).

TABLE 9. FACULTY COMPOSITION

AY	Tenured	Tenure Track	Sabbatical	FERP	Lecturers	FTEF	AY-FTES
2013-2014	4	0	1	0	0	5	56.0
2014-2015	4	0	1	0	0	5	50.1
2015-2016	5	0	0	0	0	5	56.9
2016-2017	5	0	0	0	0	5	66.6
2017-2018	5	0	0	0	0	5	61.5

Appendix IV. Resources

TABLE 10. Resources

Table 10 shows the funding that the program received for the past five years, including from the state-supported budget and supplemental student's fee.

TABLE 10. RESOURCES

Academic Year	State-Support Funds + Supplemental Student's Fee Allocated to MSE Program
2018-2019	$44,740
2017-2018	$30,000
2016-2017	$5,443
2015-2016	$45,270
2014-2015	Data not available

- This is a budget only for the MSE program's advertisement, faculty training, travel, hardware and software purchase.
- Faculty's salaries are not included.
- Salaries for a ¼ staff member and a student assistant are included.

Appendix V. Long-term Planning

In addition to the long-term planning statement in Chapter VII, here we describe how to assess student learning objectives, expected enrollment and budget, and the process of program operation.

Assessment of Student Learning and Program Quality

To measure and assess the student learning objectives, we will use various assessment methods (specified in Chapter III) and regular surveys. Learning objectives include: professional, legal and ethical issues in software engineering, software requirements comprehension and communication, software practices, process, assessment, and improvement, and collaborative work.

To improvement the assessment of program quality, we will expand the current survey that students take during the new student orientation and the midpoint workshop, with more questions about additional aspects of the program. We will collect survey results and analyze them. We will manage assessment data properly in order to keep confidentiality and to improve the program continuously.

Enrollment Stability and Increase

Table 11 shows the expected enrollment for the next five years, starting 2020.

TABLE 11. ENROLLMENT EXPECTATION

Cohort Year	*State Support Enrollment*	*Total FTES Generated*
2020	130	46.0
2021	130	46.0
2022	150	52.5
2023	150	52.5
2024	150	52.5

Table 12 shows the expected budget for these years.

TABLE 12. LONG-TERM BUDGET PLAN

Cohort Year	*State Support Enrollment*	*Total FTES Generated*	*MSE Budget*
2020	130	46.0	$45,000
2021	130	46.0	$45,000
2022	150	52.5	$50,000
2023	150	52.5	$50,000
2024	150	52.5	$50,000

To achieve this goal, we will analyze and track the effectiveness of promotion methods, and develop new ones.

The Process of Program Operation

We will create documents that provide all necessary procedural and policy information about the program as well as instructions for operating the program, such as a list of tasks to be performed, events to schedule, promotional methods, etc. We will record and store various kinds of meeting minutes, such as the program committee's meeting minutes, the program office's meeting minutes, and the industrial advisory board's meeting minutes.

Appendix VI. Curriculum Vitae of Faculty

CURRICULUM VITAE

Revised: March, 2019

NING CHEN

Professor
Department of Computer Science
California State University, Fullerton
Fullerton, CA 92834

PROFESSIONAL INTERESTS

Software Testing (SQA, V&V, Testing), Software Architecture, Enterprise Computing (Odoo, Python, Django), Embedded Systems (AWS IoT, freeRTOS), Formulation, Integration and Supervision of Multi-disciplinary Research and Development (Microgrid).

EDUCATION

Ph.D., Electrical Engineering, Colorado State University,Fort Collins, Colorado, 1986.
M.S., Electrical Engineering, Colorado State University,Fort Collins, Colorado, 1984.
B.S., Hydraulics Engineering, National Cheng KungUniversity, Tainan, Taiwan, ROC, 1978.

PROFESSIONAL AND RESEARCH EXPERIENCE

1999 - Present	Professor, Department of Computer Science, California State University, Fullerton, California.
2014- 2017	Founding Program Coordinator, Accelerated Master in Software Engineering program, California State University, Fullerton, California.
2011 - 2014	Program Coordinator, Online Master in Software Engineering program, California State University, Fullerton, California.
2000 - 2003	Chair, Department of Computer Science, California State University, Fullerton, California.

RECENT PUBLICATIONS (Papers/Patents / Reports)

N. Chen, E. Chen and I Chen, “Integrating Software Testing Standard ISO/IEC/IEEE 29119 to Agile Development,” Int'l Conf on Software Engineering Research & Practice (SERP2018), Las Vegas, USA

S. C. Lo and N. Chen, “IEEE 42010 and Agile Process- Create Architecture Description through Agile Architecture Framework,” ,” Int'l Conf on Software Engineering Research & Practice (SERP2017), Las Vegas, USA

R. Subramania, N. Chen and T. Zhu, “Behavior Driven Test Automation Framework,” Int'l Conf on Software Engineering Research & Practice (SERP2017), Las Vegas, USA

K. Bhalerao and N. Chen, “Principles of Continuous Integration (CI) in Practice,” Int'l Conf on Software Engineering Research & Practice (SERP2016), Las Vegas, USA

N. Chen, “IEEE std 829-2008 and Agile Process - Can they work together”, Int'l Conf on Software Engineering Research & Practice (SERP2013), Las Vegas, USA

T.W. Calwell, and N. Chen, “Grip Pressure Sensor,” US Patent U.S. Patent# 8,033,916 issue date: October 11, 2011

N. Chen and W. L. Cheung, “Bow-to-string pressure training device for bowed string music instruments,” US Patent # 7605317, issue date: 10/20/2009

INDUSTRY ACTIVITIES

Consultant

Microgrid (Blockchain, IoT) , Smart Power Grid Sensing (Grafana Time Series) in the Power Electric Industry

HONORS

Excellent in Teaching Award, UPCEA, Leaders in Professional, Continuing and Online Education, 2017
Faculty Marshal, College of Engineering and Computer Science, 2013

CERTIFICATIONS AND PROFESSIONAL MEMBERSHIP

Software Architecture: Principles and Practices, Software Engineering Institute, Carnegie Mellon University
Member, Chinese-American Computer Association

SERVICE

Faculty Selection Committee 2017-18
Faculty Selection Committee (Chair) 2016-17
Graduate Committee 2016-17
Undergraduate Committee 2018-19
Department Chair (2000-2003)

RECENT CONTRACTS, GRANTS

Intramural Grant, 2014-15, $3000
University Missions and Goals Initiative $14,000, 2002.
N. Chen, “Cyber Lab,” 1998-99 University Missions and Goals Initiatives. Award size: $10,365.00.
N. Chen, “ECS Faculty Project – Defining a 300 or 400 level course in Introduction to Systems Engineering,” Funded by Rockwell Endowment income 1998-99.
"Firmware Programming and Testing of Foreman's DNC," funded by AMADA Engineering and Service Inc., $5000.00,
"Microprocessor Based Tape Drive Emulator" funded by AMADA Engineering and Service Inc., $35,000.00
Seed Grant of $1,500.00 from School of Engineering and Computer Science, CSUF, 1991
"Micro-mouse Competition Program for Junior Colleges" approved by California Lottery Fund, 1990, $14,000.00

Curriculum Vitae

Song-James Choi
Tel: 657-2787257   Email: jchoi@fullerton.edu

POSITION HISTORY

2018 -	Vice Chair, Computer Science Dept. California State University: Fullerton, CA
2014 - 2018	Program Coordinator for Masters of Software Engineering California State University: Fullerton, CA
2009 -	Professor, Computer Science Dept. California State University: Fullerton, CA
2006 - 2009	Chair, Computer Science Dept. California State University: Fullerton, CA
2004 – 2005	Program Coordinator for Masters of Software Engineering California State University: Fullerton, CA
2002 – 2009	Associate Professor, Computer Science Dept. California State University: Fullerton, CA
1996 – 2002	Assistant Professor, Computer Science Dept. California State University: Fullerton, CA
1993 -1996	Visiting Research Scientist, Computer Science Dept. University of Southern California: Los Angeles, CA
1990 – 1993	Research Associate, Computer Science Dept. University of Southern California: Los Angeles, CA
1988 – 1989	Assistant Professor, Computer Science Dept. Calif. State University in Los Angeles: Los Angeles, CA
1985 – 1988	Research Assistant, Computer Science Dept. University of Southern California: Los Angeles, CA

EDUCATION

- University of Southern California: Los Angeles, CA
  Ph.D. - Computer Science
  Major in Software Engineering

- University of Southern California: Los Angeles, CA
  M.S. - Computer Science
  Major in Software Engineering

- Technical University of Karlsruhe: Karlsruhe, Germany
  M.S. – Physics

- Technical University of Karlsruhe: Karlsruhe, Germany
  B.S. – Physics

- Salem High School: Salem, Germany

PUBLICATIONS

- C. Jo, G. Chen and J. Choi, "A Framework for BDI Agent-Based Software Engineering", *Studia Informatica Universalis (Int'l Journal)*, 2005

- Choi and Scacchi, Formal Analysis of Structural Correctness of Software Descriptions, International Journal of Computers and Applications, 2003

- Choi and Scacchi, Modeling and Simulating Software Acquisition Process Architecture, Journal of Systems and Software, 2001

- Choi, E3SD: AN Environment Supporting Structural Correctness of SLC Descriptions, IASTED Software Engineering Conference, 2000

- Choi and Scacchi, Modeling and Simulating Software Acquisition Process Architecture, ProSim2000, 2000

- Scacchi and Choi, Experience with Software Architecture and Configured Software Descriptions, WESAS 2000, May 2000

- Choi, Normalization and Tools Supporting the Structural Correctness of SLC Descriptions, Proc. in IASTED International Conference on Software Engineering, 1998

- Choi, Software Maintenance through Reverse engineering, Proc. in 8th KSEA Technology Conference, 1998

- Choi and Scacchi, Softman: An Environment for Forward and Reverse Computer Aided Software Engineering, Information and Software Technology, 33(9):664-674, November 1991

- Choi and Scacchi, Extracting and Restructuring the Design of Large Systems, IEEE Software (7):66-73, January 1990

- Choi and Scacchi, Assuring the Correctness of Configured Software Descriptions, Proc. 2nd Int. Workshop on Software Configuration Management, ACM Software Engineering, 17(7) 67-76, 1989

- Choi, Softman: An Environment for Forward and Reverse Engineering, PHD Thesis, 1988

RESEARCH GRANTS

- 2000, Lockheed Martin, for Research in Identification and Qualification of Reuse of Legacy Software Systems

- 1999, Junior/Senior/general Faculty Research Award, for investigation of Reengineering Software Designs Using Restructuring

- 1998 SH Corp, for investigation of Reverse Engineering for maintaining and Improving Software Systems

- 1998 Lockheed Martin, for Research in Reuse of Software Using Orthogonal Threads

- 1990 - 1993 Northrop Corporation, for Research in Problem and Opportunities for very large software engineering and software process engineering

- 1990-1992 Naval Ocean System Center, for Research in the System Factory Approach to Large Scale Software Engineering Environments

- 1989-1992 Pacific Bell, for Research in the USC System Factory Project

- 1988-1990 AT&T Laboratories, for Research into flexible software manufacturing systems

- 1985 TRW Defense Systems Group, for Research in Computer Science and Engineering

PROFESSIONAL ACTIVITIES

- **Program Chair**, Workshop on Bridging the Gap, CSUF, 2007
- **Program Committee Member**, *IASTED International Conference on Software Engineering*, Innsbruck, Austria, 2007 (SE2007)
- **Program Committee Member**, *IASTED International Conference on Software Engineering*,
  Innsbruck, Austria, 2006 (SE2006)
- **Program Committee Member**, *IASTED International Conference on Software Engineering*, Innsbruck, Austria, 2005 (SE 2005)
- **Program Committee Member**, *IASTED International Conference on Software Engineering*, Innsbruck, Austria, 2004 (SE2004)
- **Program Committee Member**, *7th IASTED International Conference on Software Engineering and Applications*, Marina Del Rey, USA, 2003 (SEA2003)
- **Program Committee Member**, *6th IASTED International Conference on Software Engineering and Applications*, 2002 (SEA2002)
- **Program Committee Member**, 2nd ACIS Annual International Conference on Computer and Information Science (ICIS) '02), Seoul, Korea, 2002
- **Program Committee Member** , SNPD '02 - 3rd ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing, Madrid, Spain, 2002

SERVICE ACTIVITIES

- **Department Chair (2006 -2009)**
- **Coordinator for Masters of Software Engineering (MSE) Program (2004 – 2005, 2014 - 2018)**
- **Masters of Engineering Management Program Committee Chair**
- **Dept. Executive Committee**
- **Dept. Personnel Committee**
- **Dept. Selection Committee**
- **Dept. IRC (Instructional Resource Committee)**
- **Dept. UPE Advisor**
- **Dept. Graduate Committee**
- **Dept. Undergraduate Committee**
- **Dept. Library Committee**
- **College Curriculum Committee**
- **University Ad Hoc Global Competency Committee**
- **University Research Committee**
- **University Library Committee**
- **University Professional Leave Committee**
- **University Presidents Scholar Committee**

SCHOLARY/CREATIVE ACTIVTIES

The following list shows some of the projects and I have supervised or reviewed last several years.

1. Pravin Singi, “Network Monitoring System – Implementation of Support Vector Machine for Ipv4 Categorization”
2. Chetan Chinthanippu, “Online Marketing Survey”
3. Asana Javidfar, “P2P Car Sharing Web Application”
4. Amit Anand, “Business Intelligence in Cloud”
5. Omar, Alruhaily, “Volunteer web Application”
6. Sanket Dable, “iHealth CSUF – An iOS Application”
7. Moses Odjoji, “Workout Companion System”
8. Swati Shende, “Predictive Analytics (S2014)”
9. Khalid Alahmari, “Step + I Phone Application”
10. Aparna Asha, “Recommder System based on MapReduce”
11. Anuj Patel, AppLcker – Stay Protected Always
12. Tanmayee Yerakala, “Enabling Secure and Efficient Ranked Multi-Keyword Search over Outsourced Cloud Data”
13. Nitin Agnihotri (S2013),” CSUF’s Post Baccalaureate Pre-health Website”
14. Ukaj Bakhade, “ Stick Market Prediction for Short term Investment”
15. Karthika Jayaseelan, “Data Leakage Detection System Using Watermarking”
16. Jyothi Duggimpudi, “Continuous Neighbor Discovery in Asynchronous Sensor Network”
17. Kiran Kaithe, “Integrated Software Configuration Management Tool”
18. Nam Ngo, “Developing a Management Tool for a Financial Advising Company using VMC Architecture”
19. Lavanya Palamakula, “Movie Recommendation Based on Naïve Bayes Classifier and Collaborative Filtering”
20. Vidhi Bhagdev, “Dashboard & Reporting for Patrol”
21. Mangla Vasu, Call Management system
22. Ryan Nance, A continuous Tetsing Plug-in for MS Visual Studio 2010 and Nunit 2.5.9
23. Charles Wang, “Improving Open Source Involvement through Social Coding”
24. Sasan Soroushi, “Configuration Management – Roles and Responsibilities”
25. Karen Bhonesale, “WebOs”
26. Lakshimi Mattru, “Intelligent On-line Shopping”
27. Trong Tat, “Book Search Application”
28. Laxmi Motwani, “Provider’s Self Services”
29. Seema Kale, “Reimplementation of the ECS website using Content Management System”
30. Baldeep Singh, “Human Resource Management System”
31. Pankaj Jaiwal, “Dissemination Based Information System”
32. Magn Conkle, “Jitterburg”
33. Gary Lew, “Extending Firefox”
34. Alexander Senf, “Software Architecture for Helicopter Autopilot”

35. Jaydhar Vyas, “ XDC Convert – AN Application Converting XML and Relational Data”
36. Nadeem Ahmed, “Web services and Grid Computing”
37. Fracncis, Cho, “Web based Small Business Financial Analyzer”
38. Shaikth Rahman, “Mobile RSS Reader”
39. Richard Hyunh, “Wedding Planner”
40. Quy Troung, “Prescription Processing System”
41. Sung Hyun Lee, “Implementation of Alignment Algorithm for Patterns”
42. Chang-Ching Chen, “Massive Email System Using Software Engineering Methodology”
43. Gulay Turkoglu, “Design of Colored Petri Nets Models for Traffic Signals”
44. Liyu Zhang, “Web based car rental system”
45. Tianzhi Zheng, “Creating a Modeling Language for BDI Agent Software Development Process”
46. BingBing Xu, “A Web based Class scheduling”
47. Melissa Mikios, “A Computer Science Tutor”
48. Roopa Chinnakotla, “A Compiler to Translate SVG Format to Java”
49. Sravani Dhanireddy, “Developing CASE Tool for Agent Based Modeling Technique”
50. Dae Young Cho, “Regression Test for BDI Agent based Software”
51. Himabindu Chekuri, “Software Requirements Specification for Electronic Medical Link Client Module”
52. Satish Pasala, “Software Specification for Electronic Medical Link Server Side”
53. Sridevi Nelluri, “Software Specification for Website Developer”
54. Jian Yang, “A DMS-based online Training System”
55. Yun Quan, “Developing VoiceXML Web Services Based on Accounting System
56. Yu Ma, “Web Application Development Using NET Technologies”
57. Quan Li, “A Quoting Management System For CJ Interior”
58. Hung_hsun Wang, “Lightweight Encryption in Multimedia”
59. Ling Sun, “Web Application System for a World Communication, Inc”
60. Alfredo Torres, “Executing VB NET Applications with Mono on Linux”
61. Pratyusha Banerjee, “Electronic Health Records System Using Web services”
62. Prannita Kale, “Web based data management system for hunting and fishing license application”
63. Phillip Reese, “Video Game Development with Life Cycle”
64. Erick Abasto, “Educational Search Engine”
65. Shaikh A. Rahman, “Mobil RSS Reader”
66. Vinay Shah, “Inventory and Financial Tracking System”
67. Husam Azma, “ WebCRM”
68. Ankit Shah, “Stock Value Trend Protection Using Binary String Comparison”
69. Eugene Chow, “Time-Series forecasting with Support of Trend Analysis”
70. Sunwind Le, “The Social Aspect of Agent based Programming Language”
71. Pei-Chin Wong, “Design and Implement an E-commerce Web-site for TP Motor sport”

BIN CONG
Department of Computer Science
California State University at Fullerton
Email: bcong@fullerton.edu

Education:
Ph.D. Computer Science, University of Texas at Dallas, August 1991
B.S. Computer Science, Nanjing University, China, August 1982

Academic Experience:
August 2004 - Present: Professor, California State University at Fullerton
Aug. 2004 – Aug 2007: Coordinator of the MSE (Master of Software Engineering – On-line) at California State University.
August 2000 – August 2004: Associated Professor, California State University at Fullerton
August 1998 – August 2000: Assistant Professor, California State University at Fullerton
September 1997 – August 1998: Associate Professor, Cal Poly at San Luis Obispo
May 1996 – August 1997: Associate Professor, South Dakota State University
August 1991 - May 1996: Assistant Professor, South Dakota State University

Non-Academic Experience
Aug. 2000 – Aug. 2001: (Professional Leave.) Chief Technology Officer, AE Inc. at Silicon Valley

Certifications
May 2012: Certified Scrum Product Owner
Feb. 2012: Certified Scrum Master
September 2002 - Present: Certified CMMI High Maturity Lead, and Instructor

Current membership in professional organizations

ACM，IEEE, and CMMI Institute

Honors and awards
WHO'S WHO among America's Teachers 1996
International WHO'S WHO of Information Technology 1997
Outstanding Faculty for Research and Creative Activities, CSUF, 2002-2005
Outstanding Faculty Recognition Award for Service, CSUF, April 2003-2006
Distinguished Oversea Chinese Scholar, Oct 2008

Distinguished Faculty Member, College of ECS，May 2011

Chinese IT Standard Committee Member，2015 – 2018

Service activities

Served as Chair for following Department Committees in past five years:

Department Personnel Committee, Department Selection Committee

Served as Coordinator for the following programs in Cal State Univ. at Fullerton:

MSE （Master of Science in Software Engineering), AMSE (Accelerated Master of Science in Software Engineering), Graduate Program in Computer Science

Served as the reviewer for CMMI 2.0 model, 2017 – 2019

Served as CMMI 2.0 panel in 2018’s CMMI conference

Invited to give presentations in CMMI Conference and China’s TiD conference

Publications in past 5 years

1. “Making CMMI a Safety Net for Agile Development”, Capability Counts 2016, Reston, VA.
2. “Spin and re-spin a web to catch all possible bugs: a new way to build and continuously refine a process performance model”, Capability Counts 2017, Reston, VA.
3. “Unity of learning and doing: value driven Agile and Lean software development”, a book published by Post & Telecom Press (A top Chinese Publisher in IT), Oct 1, 2017

Recent professional development activities

Received training of CMMI 2.0 in 2017 and 2018

Curriculum Vitae

Chang-Hyun Jo
Professor, Ph.D.
Department of Computer Science
California State University at Fullerton
Fullerton, CA 92834-6870, USA
657-278-7255
(E-mail) cjo@fullerton.edu
(WWW) http://jo.ecs.fullerton.edu

EDUCATION:

Ph.D. in Computer Science (May 1991), Oklahoma State University, Stillwater, OK, USA.
https://go.okstate.edu/
M.S. in Computer Science (July 1988), Oklahoma State University, Stillwater, OK, USA.
https://go.okstate.edu/
B.Economics in Statistics (Feb. 1984), SungKyunKwan University, Seoul, Korea.
https://www.skku.edu/eng/

CERTIFICATES:

Certified Scrum Master
Certified Product Owner
SEI Certified CMMI Instructor
SEI Certificate in CMMI
SEI ATAM Evaluator Certificate
SEI Software Architecture Professional Certificate
IEEE Computer Society Education Board (IEEE-Certified Software Development Professional (CSDP), SWEBOK)

EXPERIENCE:

August 2002 – Present:
Professor (Tenured) and the Coordinator of Master of Science in Software Engineering (MSE) Program, Department of Computer Science, California State University at Fullerton, CA, USA

- Teaching: Software Engineering, Compilers, Programming Languages, Seminar in Computer Sciences
- Research: Numerous funded research and consulting, Publication of 70+ technical papers since 1988
- Service: ACM SAC Programming Languages Track Chair (1998-2010), Graduate Advisor, Vice Chair, ACM Student Advisor, Graduate Committee, Undergraduate Committee, MSE Committee, AMSE Committee, Resource Committee, Executive Committee, etc.
- Student Advising: More than 100 M.S. students in the area of software engineering, CMMI, SCAMPI, software design, software architecture, ISO 12207, ISO 15504 (SPICE), agent-oriented software engineering, agent-based programming languages and compilers, Agile processes, Scrum, XP.

Oct. 1998 – July 2002:
Associate Professor (Tenured), Department of Computer Science, University of North Dakota, Grand Forks, ND, USA

- Teaching: Organization of Programming Languages, Software Engineering, Principles of Translation,
    - Computer Science II (Ada), Compiler Design (Java Bytecode), Advanced Software Engineering (OOA/OOD),
    - Programming Languages and Paradigms (Distributed Computing with Java/CORBA – Programming for Internet/Web Server/Collaborative Tools).
- Student Advising: 7 M.S. Students and Undergraduate Student Projects/Thesis
    - Service: Faculty Search Committee Chair, Undergraduate Committee Chair (curriculum, assessment, accreditation), and several committee members.
- Research: Several research grant, publication of 7 technical papers in conference proceedings.

Sept. 1991 – Aug. 1998:
Associate Professor (Tenured), Computer Science Department, Kyonggi University, Suwon, Kyonggi-Do & Seoul, Korea.

- Teaching: Programming Languages, Compiler Writing, Computer Science, Artificial Intelligence, Programming Methodology and Practices, Multimedia Programming, etc. (Undergraduate/MS/Ph.D. Courses)
- Student Advising: 4 M.S. graduate student projects
- Serving many committees including the University Information Committee Chair
- Research: Numerous research grants, publications of almost 40 technical papers in journals, conferences and international standards (between 1988 and 1998).

Department Chair (March 1992 - Feb. 1994), Computer Science Department, Kyonggi University, Korea.

- Academic Administration, Fund Managing, Lab Renovation

Part-time Research Staff (Invited Joint-Research Position) (Sept. 1992 - July 1993), Electronics and Telecommunications Research Institute (ETRI), Korea.

- Joint Research: Object-Oriented CHILL Design and Compiler Construction, Debugger, Software Testing etc.
- Delegate to International Standardization of ITU-T Z.200
  (Sponsored by Korea Telecommunications Tech. Association (TTA), ETRI, Samsung)

Aug. 87 - May 1991:
Graduate Teaching Assistant, Computer Science Department, Oklahoma State University, USA.

- Class Teaching: Compiler Writing, PL/I Programming Language
- Grading: Discrete Mathematics, Operating Systems, Computer Science, Computer Architecture
- 7 Publications including technical papers in ACM/IEEE/ISMM conferences, MS thesis, and Ph.D. Dissertation.

Dec. 83 - April 85:
Software Engineer, Electronics Research Lab. Hyo-Sung Co., Korea.

- Constructing Chinese/Japanese Dictionary Utility Program, Testing and maintaining Hitachi COBOL Compiler, Writing Technical Reports and Lab Standards.

PROFESSIONAL AFFILIATION AND ACTIVITIES:
Member of Scrum Alliance
Member of ACM, IEEE Computer Society
Programming Language Track Chair of ACM Symposium on Applied Computing (ACM SAC) (1998 - 2008)
Editor of ACM Applied Computing Review (1997-2008)
Asia Membership Liaison of ACM SIGAPP (1996-1998)
Secretary of ACM SIGAPP (2005-2007)
ITU-T SG10 Korean Delegate (1992-1998: Involved in International Standardization for CHILL)
Editor and Vice Chair of Korea Information Processing Society (1996-1998)
Co-organizer of Special Interest Group on Object-Oriented Technology in Korea (1997-1998)

AREA OF INTEREST:
Software Process Improvement, Software Architecture, ATAM, CMMI, SCAMPI, Agent-Based Programming Language and Software Engineering (APL, AMT, etc.), Object-Oriented S/W Engineering (Modeling, Analysis, Design, Testing, Tools, Formal Specification), Concurrent Object-Oriented Programming Language Design and Compiler Construction (Parallel-C++, CHILL-96), Integrated Programming Environments (Debugger, Compiler/Interpreter), Computer Supported Cooperative Work (CSCW)/Collaborative tools/Groupware on the Internet/Web, Ubiquitous Computing, SaaS (Software as a Service)

PUBLICATIONS:

1. Jo, Chang-Hyun. *Abstraction and Specification of Local Area Networks*. **M.S. Thesis**. (Donald D. Fisher, Advisor), Department of Computing and Information Sciences. Oklahoma State University. (July 1988).
https://shareok.org/handle/11244/15620

2. Jo, Chang-Hyun, Fisher, Donald D. and George, K.M. Abstraction and Specification of Local Area Networks. Proceedings of the Eighth Annual International Phoenix Conference on Computers and Communications, Scottsdale, Arizona, (March 22-24, 1989), IEEE Computer Society Press, (1989), 337-342.

3. Jo, Chang-Hyun and George, K.M. Distributed object-oriented programming with dynamic objects. OSU-CS-TR-89-14, Department of Computer Science, Oklahoma State University (1989).

4. Jo, Chang-Hyun. Parallel-C++: An object-based parallel programming language. (Abstract/Presentation) Centennial Graduate Student Research Symposium, Oklahoma State University, (Feb. 27, 1990).

5. Jo, Chang-Hyun and George, K.M. Language concepts using dynamic and distributed objects. Proceeding of the ACM 1991 Computer Science Conference (ACM/CSC '91), San Antonio, Texas, (March 5-7, 1991), ACM Press (1991), 211-220.
   http://dl.acm.org/citation.cfm?id=327243

6. Jo, Chang-Hyun, George, K.M. and Teague, K.A. Parallelizing translator for an object-oriented parallel programming language. Proceedings of the Tenth Annual International Phoenix Conference on Computers and Communications (IPCCC '91), Scottsdale, Arizona, (March 27-30, 1991), IEEE Computer Society Press (1991), 265-271.

7. Jo, Chang-Hyun. *The Design and Implementation of an Object-Oriented Parallel Programming Language*. **Ph.D. Dissertation**, (K. M. George, Advisor), Department of Computer Science, Oklahoma State University (May 1991).
   https://shareok.org/handle/11244/20669

8. Jo, Chang-Hyun and George, K.M. Storage management for dynamic objects in a distributed memory system. Proceedings of the Fourth ISMM International Conference on Parallel and Distributed Computing and Systems. Washington, DC, ACTA Press, (ISBN: 0-88986-159-5), (October 8-11, 1991), 112-116.

9. Lee, J. K., ***Jo, Chang-Hyun**, et. al. A Survey of Object-Oriented Programming Concepts. ETRI, Korea, TT/E-TM92-05, (1992).

10. Jo, Chang-Hyun. Investment Consultant Expert System. Proc., Basic Science Research Institute, Kyonggi University, Vol.6, No.1, pp.73-84, (1992).

11. Jo, Chang-Hyun. Natural Language Processing using Prolog. Kyonggi University Journal, No.31, 329-342, (1992).

12. Lee, J.-K., ***Jo, Chang-Hyun**, Lee, D.-G., Choi, W., Song, Y.-K., and Kim, Y.-S., A Survey: Object-Oriented Programming Concepts and Paradigm, Electronics and Telecommunications Trend Analysis, ETRI, Korea, 45-68, (April 1993).

13. Lee, J.-K., ***Jo, Chang-Hyun**, Lee, D.-G., Choi, W., Song, Y.-K., and Kim, Y.-S., A Study of Type Extension for OOCHILL, Korea Information Science Society 1993 Spring Conference, (1993).

14. Lee, J.-K. ***Jo, Chang-Hyun**, Lee, D.-G., Choi, W., Choi, G.-B., and Lee, C.-K. An efficient implementation of type-test and type-guard for an object-oriented switching system. Proc. of the International Conference on Globalization of Computer & Communication (INFOCOM'93), Tata McGraw-Hill, New Delhi, 148-155, (1993).

15. Jo, Chang-Hyun, A Study on Concurrent Object-Oriented Programming, The Statistical Review, V.2, Research Institute of Applied Statistics, 61-76, (May 1994).

16. Jo, Chang-Hyun and Choi, Wan. A Proposal for Concurrent Objects in CHILL. ITU-T SG10 Contribution, 10/D/0025, Geneva, Swiss, (1994. 10.).

17. Jo, Chang-Hyun and Lee, Sang-Hong. Analysis for the Syntactic Problems in Object-Oriented CHILL. ITU-T SG10 Contribution, 10/TD/0055, Geneva, Swiss, (1994. 10.).

18. Kim S., ***Jo, Chang-Hyun**. An Implementation Scheme for a Concurrent Object-Oriented Language on a Parallel System, Korea Information Science Society Conference (Fall), 117-120, (1994).

19. Kim, S., ***Jo, Chang-Hyun**. A Design and Implementation of Concurrent Object-Based C Preprocessor, Korea Information Processing Society Conference (Fall), 75-78, (1994).

20. Baek, I., ***Jo, Chang-Hyun**. An Implementation Scheme for Concurrent Object-Oriented CHILL on SPARC, Korea Information Science Society Conference (Fall), 113-116, (1994).

21. Jo, Chang-Hyun. Guarded Methods in Object-Oriented CHILL. 1995 CHILL Expert Meeting Contribution, WD.004, SIEMENS, Taipei, (1995. 3.).

22. Jo, Chang-Hyun et al. Design of a Concurrent Object-Oriented Programming Language for Telecommunications, Korea Information Science Society Conference (Spring), 337-340, (1995. 4.).

23. Jo, Chang-Hyun, Guarded Methods for Concurrent Objects. Korea Information Processing Society Conference (Spring), 459-462, (1995. 5.)

24. Jo, Chang-Hyun, et al. Encoding for Debugging Information of Object-Oriented Programs, Korea Information Science Society Conference (Choong-Chung Regional), 253-256, (1995. 11.24 - 25).

25. Jo, Chang-Hyun. An Experiment on a Concurrent Object-Oriented Programming Language. Proc. of the ACM Symposium on Applied Computing (SAC'96), Philadelphia, USA, 98-104, (1996. 2. 18-20).
https://dl.acm.org/citation.cfm?id=331157

26. Kim, P., Lee, C., Lee, J., ***Jo, Chang-Hyun**. A Development of a SPARC Back-End, Korea Information Processing Society Conference (Spring), 138-141, (1996. 4. 12-13).

27. Im, H. , Kim, P., ***Jo, Chang-Hyun**, et al. A Design of an Object-Oriented CHILL Debugger, Korea Information Processing Society Conference (Spring), 142-145, (1996. 4. 12-13).

28. Lee, C., Kim, J., ***Jo, Chang-Hyun**. Design and Prototyping of a C++ Compiler Front-End, Korea Information Science Society Conference (Fall), 819-822, (1996. 10. 25-26).

29. Kim, P., Im, H., ***Jo, Chang-Hyun**, et. al. A Model Test of Object-Oriented CHILL Debugger, Korea Information Science Society Conference (Fall), 827-830, (1996. 10. 25-26).

30. Jo, Chang-Hyun, et al. A Design and Prototyping of an Object-Oriented Program Debugger, Proc. of the ACM Symposium on Applied Computing (SAC'97), San Jose, USA, 45-51, (Feb.28, 1997).
    https://dl.acm.org/citation.cfm?id=331708&dl=ACM&coll=DL

31. Im, H., ***Jo, Chang-Hyun**, et al. A Debugging Scheme for Object-Oriented Program with Visibility, Korea Information Processing Society Conference (Spring), 866-869, (1997. 4. 12.).

32. Lee, C., Son, J, and ***Jo, Chang-Hyun**. An Implementation of a Concurrent Object-Oriented Language: Parallel-C++, Korea Information Science Society Conference (Fall), 277-280, (1997. 10).

33. Jo, Chang-Hyun and Kang, S. An Experiment on Semantic Prototyping, Korea Information Science Society Conference (Fall), 331-334, (1997. 10.).

34. Jo, Chang-Hyun. A Proposal for an Overloading Mechanism in Object-Oriented CHILL, Journal of Basic Science, 10(2), 393-401, Kyonggi University, (1997. 12.).

35. Jo, Chang-Hyun, A Formal Definition of Module Mode in CHILL, Journal of Basic Science, 19(2), 403-410, Kyonggi University, (1997. 12.).

36. Jo, Chang-Hyun. A Dynamic Binding and a Dynamic Type Checking for Polymorphism, Journal of Kyonggi University, 41(2), 289-304, (1997. 12.).

37. Jo, Chang-Hyun. A Design of Generic Constructs on an Object-Oriented Programming Language, Journal of Kyonggi University, 41(2), 305-319, (1997. 12.).

38. Jo, Chang-Hyun, et al. A Realization of Concurrent Object-Oriented Programming, Proc. of the ACM Symposium on Applied Computing (SAC'98), Atlanta, Georgia, USA, 558-563, (Feb. 27 – Mar 1, 1998).
    https://dl.acm.org/citation.cfm?doid=330560.330928

39. Jo, Chang-Hyun et al. Implementation of a Code Generator for SPARC, Journal of Korea Information Science Society, 4(1), 178-189, (Feb. 1998).

40. Jo, Chang-Hyun, Jea Gi Son, Younwoo Kang, and Phill Soo Lim, “The Distributed Programming Environment on the Internet”, ACM 1999 13th Annual Symposium on Applied Computing (ACM SAC’99), San Antonio, Texas, 85-90, (Feb. 28 – March 2, 1999). https://dl.acm.org/citation.cfm?doid=298151.298206

41. Denehy, Timothy E. and **Jo, Chang-Hyun**. “Parallel-C++ for the Java Virtual Machine”, ND-EPSCoR Poster Session, North Dakota State University, July 28, 1999.

42. Arnold, Allen J. and **Jo, Chang-Hyun**, "Distributed Programming Environment to Facilitate Geographically Separate Collaboration and Education", ND-SD Joint EPSCoR Conference Poster Session, September 10, 1999.

43. Denehy, Timothy E. and **Jo, Chang-Hyun**. "Parallel-C++ for the Java Virtual Machine", ACM 2000 14th Annual Symposium on Applied Computing (ACM SAC'00), 843-848, Como, Italy, (March 2000). https://dl.acm.org/citation.cfm?doid=338407.338576

44. Chang-Hyun Jo, Introduction to the Programming Languages Track, ACM Annual Symposium on Applied Computing (SAC), 722-724, Villa Olmo, Como, Italy, 2000. http://www.acm.org/conferences/sac/sac2000/

45. Jo, Chang-Hyun. A Seamless Approach to the Agent Development, ACM 2001 15th Annual Symposium on Applied Computing (ACM SAC'01), Las Vegas, 641-647, (March 2001). https://dl.acm.org/citation.cfm?id=372791

46. Chang-Hyun Jo, Introduction to the Programming Languages Track, ACM Annual Symposium on Applied Computing (SAC), 2001. http://www.sigapp.org/sac/

47. Jo, Chang-Hyun and Allen J. Arnold, "Agent-based Programming Language: APL", ACM 2002 16th Annual Symposium on Applied Computing (ACM SAC'02), Madrid, Spain, 27-31, (March 2002). https://dl.acm.org/citation.cfm?id=508799

48. Jo, Chang-Hyun and Rajeev R. Raje, "Editorial Message: Special Track on Programming Languages and Object Technologies", Proceedings of ACM Symposium on Applied Computing (SAC), 925-926, Universidad Carlos III De Madrid, Spain, March 11-14, 2002. http://www.sigapp.org/sac/

49. Feng, Xin and **Jo, Chang-Hyun**, "Agent-Based Stock Trader", The 18th International Conference on Computers and Their Applications, The International Society for Computers and Their Applications, (ISCA CATA-2003), 275-278, Honolulu, Hawaii, March 26-28, 2003. (ISBN 1-880843-46-3)
(A draft found at) http://www.ecs.fullerton.edu/~jo/research/Feng2003/Jo-Feng-isca2003.pdf

50. Lin, Dongqing, Wiggen, Thomas P. and **Jo, Chang-Hyun**, "A Restaurant Finder using Belief-Desire-Intention Agent Model and Java Technology", The 18th International Conference on Computers and Their Applications, The International Society for Computers and Their Applications, (ISCA CATA-2003), 404-407, Honolulu, Hawaii, March 26-28, 2003. (ISBN 1-880843-46-3)

51. Zhao, Wei and **Jo, Chang-Hyun**. “A Compiler Design for the Agent-Based Programming Language”, The 18th International Conference on Computers and Their Applications, The International Society for Computers and Their Applications, (ISCA CATA-2003), 393-396, Honolulu, Hawaii, March 26-28, 2003. (ISBN 1-880843-46-3)
http://dblp.uni-trier.de/db/conf/cata/cata2003.html#ZhaoJ03

52. Jo, Chang-Hyun and Rajeev R. Raje, “Editorial Message: Special Track on Programming Languages and Object Technologies”, Proceedings of ACM Symposium on Applied Computing (SAC), 1046-1047, Melbourne, Florida, March 9-12, 2003.
https://dl.acm.org/citation.cfm?doid=952532.952737

53. Jo, Chang-Hyun. “A New Way of Discovery of Belief, Desire and Intention in the BDI Agent-Based Software Modeling”, The International Conference and Exhibition on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management (HNICEM 2003), Manila, Pavillon Hotel, March 27-30, 2003. (ISBN# 971-92723-0-9)

54. Jo, Chang-Hyun and Allen J. Arnold, “A Portable and Collaborative Distributed Programming Environment”, The 2003 International Multi-Conference in Computer Science and Computer Engineering – The International Conference on Software Engineering, (IMCCSCE – SERP’03), 198-203, Las Vegas, Nevada, June 23-26, 2003.

55. Jo, Chang-Hyun and Einhorn, Jeffery M., “A Process for BDI Agent-Based Software Construction”, The 2003 International Multi-Conference in Computer Science and Computer Engineering – The International Conference on Software Engineering, (IMCCSCE – SERP’03), 204-209, Las Vegas, Nevada, June 23-26, 2003.

56. Einhorn, Jeffery M. and **Jo, Chang-Hyun**. “A Use-Case Based BDI Agent Software Development Process”, Proceedings of the 2nd International Workshop on Agent-Oriented Methodologies, 7-20, ACM *OOPSLA-2003*, Anaheim, October 26-30, 2003. also at http://www.open.org.au/Conferences/oopsla2003/proc.htm , Oct. 26, 2003.

57. Jo, Chang-Hyun, Zhao, Wei and Cong, Bin., “A Design and Implementation of the Belief-Desire-Intention Agent-based Programming Language”, ***Information: An International Interdisciplinary Journal***, ISSN 1343-4500 (print), 1344-8994 (electronic), 7(1), International Information Institute, (January 2004), 137-153.

58. Jo, Chang-Hyun. “A New Way of Discovery of Belief, Desire and Intention in the BDI Agent-Based Software Modeling”, ***the International Journal of Advanced Computational Intelligence & Intelligent Informatics (JACIII)***, ISSN 1343-0130, 8(1), 2-6, Jan. 2004.
https://www.fujipress.jp/jaciii/jc/ (the most downloaded in January 2015, Oct. 2014, Sept. 2014) https://www.fujipress.jp/jaciii/jc/jacii000800010002/?highlight=chang-hyun%20jo

59. Jo, Chang-Hyun, Guobin Chen and James Choi, “A New Approach to the BDI Agent-Based Modeling”, ACM SAC 2004, 1541-1545, Nicosia, Cyprus, March 14-17, 2004.
https://dl.acm.org/citation.cfm?id=967900.968208

60. Chang-Hyun Jo, Marjan Mernik, and Barrett R. Bryant, Editorial Message: Special Track on the Programming Languages, ACM Symposium on Applied Computing (SAC), 1453-1454, Nicosia, Cyprus, March 14-17, 2004.
https://dl.acm.org/citation.cfm?doid=967900.968191

61. Jo, Chang-Hyun, Guobin Chen and James Choi, "A Framework for BDI Agent-Based Software Engineering", *Studia Informatica Universalis (International Journal)*, Vol.3, No.3, Editions SUGER, 15 rue Catulienne, F-93200 Saint-Denis, France; 41 rue G. Lussac, 75005 Paris, France, (ISBN 2-912590-30-2), (ISSN 1621-7545), 285-314, 2004.
https://dblp.org/db/journals/siu/siu3.html

62. Jo, Chang-Hyun, Won-Young Kim, Jeong-Min Shim, and Wan Choi. Agent-based Framework for Software On-Demand. Proceedings of the IEEE 7th International Conference on Advanced Communication Technology (ICACT), Phoenix Park, Korea, (Vol.2), 730-735, Feb. 21-23., 2005. (IEEE Catalog # 05EX1046) (ISBN: 89-5519-123-5)
https://ieeexplore.ieee.org/document/1462879

63. Chang-Hyun Jo, Marjan Mernik, and Barrett R. Bryant, Editorial: Programming Languages Track, ACM Symposium on Applied Computing (SAC), 1383-1384, 2005.
http://www.acm.org/conferences/sac/sac2005/

64. Park, Seungjin, Seong-Moo Yoo, M. Al-Shurman, B. V. Voost, **Chang-Hyun Jo**, "ARM: Anticipated Route Maintenance Scheme in Location-Aided Mobile Ad Hoc Networks", *Journal of Communications and Networks*, Vol.7, No.3, 325-336, September 2005.

65. Jeong-Min Shim, Won-Young Kim, Wan Choi and **Chang-Hyun Jo**. Agent-Based Framework for the Software On-Demand Service. Information: An International Journal, International Information Institute, A Special Issue for the Software Engineering and Software Security, Tokyo, Japan, Vol.8, No.5, 725-738, September, 2005, (ISSN 1343-4500)
http://www.information-iii.org/abs_e2.html#No5-2005

66. Jo, Chang-Hyun and Einhorn, Jeffery M., "A BDI Agent-Based Software Process", *Journal of Object Technology (JOT)*, Vol.4, No.9, 101-121, November – December 2005.
http://www.jot.fm/issues/issue_2005_11/article3/
http://www.jot.fm/contents/issue_2005_11/article3.html

67. Chang-Hyun Jo, Marjan Mernik, and Barrett R. Bryant, Editorial Message: Programming Languages Track, ACM Symposium on Applied Computing (SAC), 1601-1602, Dijon, France, April 23-27, 2006.
http://www.acm.org/conferences/sac/sac2006/
http://oldwww.acm.org/conferences/sac/sac2006/pages/Final-Program-SAC06.pdf

68. Chang-Hyun Jo, Marjan Mernik, and Barrett R. Bryant, Editorial Message: Programming Languages Track, ACM Symposium on Applied Computing (SAC), Seoul, Korea, March 11-15, 2007.
http://jo.ecs.fullerton.edu/sac/sac2007/

69. Chang-Hyun Jo, Marjan Mernik, and Barrett R. Bryant, Special Track on Programming Languages: Editorial Message, ACM Symposium on Applied Computing (SAC), 189-190, Fortaleza, Ceara, Brazil, March 16-20, 2008.
    https://dl.acm.org/citation.cfm?doid=1363686.1363738

70. Marjan Mernik, Barrett R. Bryant, and **Chang-Hyun Jo**, Editorial, Special issue on the Programming Languages Track at the 23rd ACM Symposium on Applied Computing, Science of Computing Programming (SCP), Elsevier, 75, 919-290, 2010.
    https://www.journals.elsevier.com/science-of-computer-programming

71. Chang-Hyun Jo, Software Architecture: 5W-1H, Software and System Process Improvement Network (SPIN) Meeting, Northrop Grumman, E2 Presentation Center (formally TRW), Redondo Beach, CA, 9am-12noon, Feb.12, 2010.

REFERENCES & CITATIONS

Researchers in Agent-Based Models, 2010.
http://www.agent-based-models.com/blog/researchers/j/

The E-print Network, U.S. Department of Energy (DoE), Energy, science, and technology for the research community, 2014 (last updated).
http://www.osti.gov/eprints/topicpages/documents/category/07/j/index.html (outdated)

Gary Palosaari, The Boeing Company, How to Establish a Process Architecture and Use it for Process Improvement, CMMI Technology Conference & User Group, Nov. 15-18, 2010.
https://ndiastorage.blob.core.usgovcloudapi.net/ndia/2010/CMMI/ThursdayTrack4_10942Palosaari.pdf

Nora Houari and Behrouz Homayoun Far, A Novel Approach for Developing Autonomous and Collaborative Agents, Knowledge-Based Intelligent Information and Engineering Systems, 9th International Conference, KES 2005, Melbourne, Australia, September 2005. Proceedings, Part III, Springer.

Luciana Parisi, Contagious Architecture: Computation, Aesthetics, and Space, MIT Press, 2013.

Paul Antony Martin, Distributed Processing, Patent, WO1998013758A1, 1998.
https://patents.google.com/patent/WO1998013758A1/en

International Technical Program Committee, 6th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management, 2013.

RESEARCH GRANTS

Research Grants (2002 to Present)

Research Title	**Funding**	**Amount**	**Period**
A Research on the Framework and Programming Environment for On-Demand Software	ETRI, Korea	$25,000	May 1, 2004 - November 30, 2004
A Research on the Programming Environment for the Evergreen Technique and Software on Demand	ETRI, Korea	$26,000	June 2005 – November 2005
A Research on the Programming Environment for the Evergreen Technique and Software on Demand (Renewed project)	ETRI, Korea	$30,000	July 2006 – November 2006
Notes: ETRI: Electronic and Telecommunications Research Institute (ETRI), Korea			

Research Grants (1999 to 2001)

Research Title	**Funding**	**Amount**	**Period**
Development of the Distributed Programming Environment	UND ORPC(FRCAC)	US$ 1,700	1999.1-'99.12
Development of the Distributed Programming Environment	ND EPSCoR TRIC	US$ 3,500	1999.5-'99.11
Development of the New Course Material	UND SIDP	US$ 2,700	2000.7-'00.8
Component-Based Framework (1st yr)	ND EPSCoR IIP (1st)	US$ 7,500	2000.5-'01.4
Component-Based Framework (2nd yr)	ND EPSCoR IIP (2nd)	US$ 7,500	2001.5-'02.4
Notes: • UND ORPD: University of North Dakota, Office of Research & Program Development, Faculty Research and Creative Activity Committee • ND: North Dakota State • EPSCoR: Experimental Program to Stimulate Competitive Research • TRIC: Technology Transfer Into Commercialization • UND SIDP: University of North Dakota, Summer Instructional Development Professorship • IIP: Seed Grant			

Small Awards (1999 to 2001)

Research Title	Funding	Amount	Period
Travel to the ACM SAC'99, San Antonio	ND EPSCoR	US$ 250	1999.1.- '99.5.
Travel to the ACM SAC'99, San Antonio	UND ORPD	US$ 428	1999.1.- '99.5.
Travel to the NASA	ND EPSCoR	US$ 1,000	1999.2.- '99.3.
Travel to the Oak Ridge National Lab.	ND EPSCoR	US$ 500	1999.8.
Travel to the OOPSLA'99, Denver	UND FIDC	US$ 660	1999.8.17.
OOPSLA'00 Educational Symposium Scholarship	ACM SIGPLAN	US$ 1,950	2000.10.15-19.

Notes

- ND EPSCoR: North Dakota St., Experimental Program to Stimulate Competitive Research
- UND: University of North Dakota, Office of Research and Program Development (ORPD), Faculty Research and Creative Activity Committee

Research Grants (1991-1998)

Research Title	Funding	Amount	Period
The Next Gen. CHILL Compiler	ETRI, Korea	US$17,000	1993.7-'94.6
ATM CHILL Compiler	ETRI, Korea	US$ 6,700	1994.7-'95.6
CHILL/SPARC Back-End	Samsung Electronics	US$28,000	1995.1-'95.12
Object-Oriented CHILL Debugger Research	ETRI, Korea	US$17,000	1995.3-'95.12
Object-Oriented CHILL Manual	ETRI, Korea	US$25,000	1995.9-'96.4
Compiler Front-End	Kyonggi Univ., Korea	US$ 2,800	1995.11-96.10
Object-Oriented CHILL Debugger Development	ETRI, Korea	US$22,500	1996.3-'96.12
Semantic Prototyping	Kyonggi, Univ., Korea	US$ 3,400	1997.1-'97.6
OO S/W Testing Language	ETRI, Korea	US$22,500	1997.1-'97.12
Audit Guidelines for IS Develop.	National Comp. Agency, Korea	US$20,000	1998.8-'98.10

Notes

- ETRI, Korea: Electronics and Telecommunications Research Institute
- National Comp. Agency: Korea National Computerization Agency

Curriculum Vitae

Christopher Ryu, Ph.D.
Chair and Professor
Department of Computer Science
California State University, Fullerton
Email: tryu@fullerton.edu Phone: (657) 278-7231

Education

Ph.D., Computer Science, University of Houston, 1998.
B.S., Computer Science, Inha University, 1986.

Professional Experience

- Assistant Professor, Associate Professor, and Professor, Department of Computer Science, California State University, Fullerton, 1999 – present.
- Industry project experience as project manager, architect, and software engineer between 1986 – 2015: development of intelligent video surveillance management system for LA metro and Union station; Business process, technology infrastructure, and management system for local companies such as Francis-Mustoe & Co., Russo, Fleck & Associates, Bee2buy; Technology advisor of forecasting and portfolio optimization modules for a hedge fund company; Development of an oil and gas utility management system, maintained IBM legacy systems at EDS; Development of Material Requirements Planning (MRP) system, Handok Co.

Research Interests

Artificial Intelligence (AI), Machine learning, Data Science, Software Design and Architecture, Software Management, Computational Finance

Current Projects

- Predictive modeling for the gas quality and consumption using smart sensors
- Song composition and voice modeling using AI
- Intelligent video surveillance management system for digital guards
- Time-series forecasting, risk modeling, portfolio optimization
- Moving objects databases and sensor networks

Grants and Past Projects

- C. Ryu (PI); $292,159; Demonstration of Smart Combustion Technology using Natural Gas Fuel Quality Sensor, sponsored by California Energy Commissions, UCR, and ECS, 2018 – 2021.
- C. Ryu (PI); $7,500; Fuel Characterization and Property Prediction Model for Variable-blend Natural Gas Vehicle Technology; Research, Scholarship, and Creative Incentive Grant, CSUF, 2017.
- C. Ryu (PI); A Web-based Telecommunications Link and Orbital Analysis, Simulation, and Operation; $200,863; NASA-Jet Propulsion Laboratory; 2002 - 2009.

- C. Ryu (co-PI) with M. Tolmasky, K. Kantardjeff, A. Cohen, D. Eernisse; Decontamination of affected areas after a bio-terrorist attack; $24,000; University Mission & Goals; 2003.
- C. Ryu (Consultant) with Joyce K. Ono, Judith Kandel, Merri Lynn Casem, and William J. Hoese; Development of Faculty Collaboratives to Assess Achievement of Student Learning Outcomes in Critical Thinking in Biology Core Courses; $444,005; NSF; 2002 - 2005.
- C. Ryu (co-PI) with H. Chung; Developing Performance Model and Adaptive Control Components for Brushless DC Motor based on Computer Simulation and Intelligent Learning Approaches; $139,111; UMC; 2001 - 2002.
- C. Ryu (PI); A Web-based Information Management System; $19,240; FMC; 2001 - 2002.
- C. Ryu (PI); Data Model and User Interface for ASSIST Scheduling Module Engine; $9,800. Lockheed Martin Missiles & Space; 2000.
- C. Ryu (PI); Natural-Language Understanding Database Query Interface System; $5,000; CSU Program for Research, Scholarship, and Creative Activity, 2000.

Selected peer-reviewed publications within the last 8 years

- P. Roy, C. Ryu, S. Dong, C. Park, “Development of a Natural Gas Methane Number Prediction Model”, Journal of Fuel, Vol 246, Elsevier, 2019.
- P. Roy, C. Ryu, and C. Park, “Predicting Wobbe Index and methane number of a renewable natural gas by the measurement of simple physical properties”, Journal of Fuel, Vol 224, Elsevier, 2018.
- C. Ryu and R. Rivas, “Writing a New Song using AI”, The US-Korea Conference on Science, Technology, and Entrepreneurship, Washington D.C., 20017.
- J.S. Chan and C. Ryu, “A Monte Carlo-based Approach to Improve Classification Accuracy”, International Conference on Smart Media and Applications, Kota Kinabalu, Malaysia, 2013.
- P. Danaee and C. Ryu, “Regression Analysis by Incorporating Sector Dynamics for Financial Time Series Data”, The International Conference on Data Mining, Las Vegas, Nevada, 2013.
- Y. Park, D. Seo, J. Yun, C. Ryu, J. Kim, J. Yoo, “An Efficient data-centric storage method using time parameter for sensor networks”, Information Sciences, International Journal, Elsevier 18, p. 4806-4817, 2010.

In addition to the above list, Christopher Ryu published over 40 articles.

Awards

- Contribution award towards the globalization of the campus from California State University, Fullerton, 2010
- Outstanding teacher (collaborative teaching), 2008
- Outstanding teacher and scholar award from California State University, Fullerton, 2007
- Outstanding teacher and scholars from California State University, Fullerton, 2005

Professional Activities

- A program committee member, International IEEE Conference on Tools with Artificial Intelligence, 2017.

- A reviewer for the journal of cluster computing, 2017.
- A reviewer for International Journal on Artificial Intelligence Tools, 2016.
- A reviewer for UC-Riverside federal research grant proposals, National Center for Sustainable Transportation, 2016.
- An editorial board member for KSII Transactions on Internet and Information Systems, 2013 - present.
- A guest editor for Special Issue on Advanced IT Convergence, the Scientific World Journal, 2014.
- A keynote speaker for the second International Conference on Smart Media and Applications, Kota Kinabalu, Malaysia, 2013
- A reviewer for Journal of Cluster Computing, 2013
- A reviewer for PLOS ONE, Open Access Journal, 2013
- A reviewer for International Journals of Distributed Sensor Networks, 2013
- A program committee member for RFD database conference, 2012
- A reviewer for Biomedcentral, 2012
- A reviewer for IEEE Systems, Man, and Cybernetics, 2012
- A reviewer for the IEEE transactions on Education, 2010
- A reviewer for the journal of Data and Knowledge Engineering, 2007
- A reviewer for the journal of supercomputing, 2006
- A reviewer for the journal of Data and Knowledge Engineering, 2006
- A reviewer for the IEEE transaction on education 2005 - 2006
- A reviewer of the CSUPERB grant, 2005
- A reviewer for the IEEE transactions on Systems, Man, and Cybernetics - Part B, 2005
- A reviewer for the International Journal of Data & Knowledge Engineering, 2005
- A reviewer for the Bioinformatics Journal, 2004
- A reviewer of the journal, “Bioinformatics”, International Society for Computational Biology (ISCB), 2004
- A reviewer for several computer science books 2003~2006
- A program committee member for the International Conference on Artificial Intelligence, 2001 ~ 2003
- A reviewer for 3rd International Conference on Advances in Web-Age Information Management, 2002
- A reviewer for the international conference Smart Engineering System Design: Neural networks, Fuzzy logic, Evolutionary programming, Complex systems, and Data mining, 2001

Department Services

- The chair of the computer science department since Fall 2018
- The chair of the search committee, 2017
- The chair of the search committee, 2014
- ABET coordinator, 2012 – 2014
- The program coordinator for the Master’s Degree in Software Engineering (MSE), 2008 - 2011.
- The graduate coordinator, 2007 - 2008

- Restructured the undergraduate curriculum, 2004 - 2006
- The undergraduate coordinator, 2003 - 2005
- Established the International Honor Society for the Computer Science (UPE) at CSUF and the first advisor, 2005
- The contact person for the Oracle Academic Initiative program, 2004
- Introduced the Rational Rose's Software Engineering for Education Development (SEED) program from IBM at CSUF, 2002
- Introduced the Microsoft Academic Alliance Program into the Computer Science Department, 2000

College Services

- A member of DPC for the computer engineering department, 2015 – 2016
- A member of department personnel committee for the mechanical engineering (ME) department (elected by the ME faculty members), 2014
- A member of curriculum committee, 2005, and 2008 - 2011
- Presented the data integration and information retrieval at the ECS technology breakfast, 2005
- A Computer Science, Engineering Technology, Math (CSEM) scholarship committee member, 2002 ~ 2004
- A judge for the 2002, 2003, and 2004 scholarship committee

University Services

- A member of outstanding professor committee, 2016 – 2017
- A member of the Faculty Personnel Committee (FPC), 2011 - 2013
- A member of the University strategic planning member, 2010
- A review member for the ISDS's online IT program, the college of business, 2009
- A member for the academic senate IT committee, 2005
- One of the key members for the interdisciplinary project, "Genomic Data Warehouse and Automatic Sequence Pattern Discovery and Management", 2001 - 2006
- One of four planning team members for the Center for the Applied BioComputing, BioEngineering, and BioPharmaceutical Studies (CABS) at CSUF, 2003
- A panel member for "Writing Your Professional Development Plan (PDP)" Workshop sponsored by FDC, 2002
- The Chair of Academic Senate Extended Education Committee, 2001 - 2002

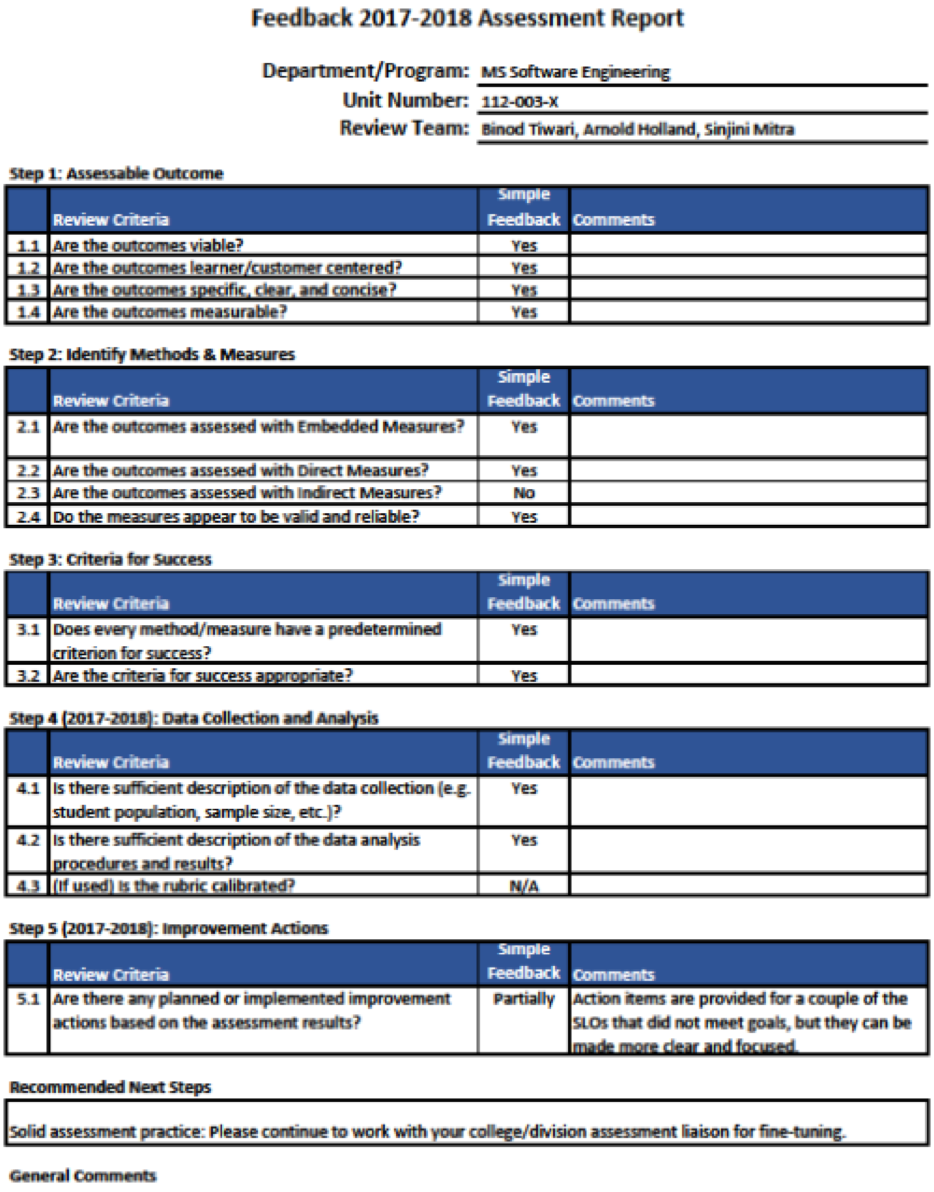
Community Services

- Southern California Chapter president for the Korean-American Scientists and Engineers Association (KSEA), 2006
- A news article about the future technology appeared on Titan Daily, Fall 2000 and another article about the NASA/JPL project appeared on Orange County Register Sunday edition, Spring 2005
- An assistant coach for a girls' soccer club organized by Junior United Soccer Association (JUSA), Placentia, CA, 2001

Appendix VII. Survey Results for Program Assessment

Assessment Report

Following is the Feedback 2017-2018 Assessment Report we made on Compliance Assist. The report shows that our assessment activities are operating smoothly and according to plan, with one recommendation, viz., the use of some indirect evidence as well.



Feedback 2017-2018 Assessment Report

Department/Program:	MS Software Engineering
Unit Number:	112-003-X
Review Team:	Binod Tiwari, Arnold Holland, Sinjini Mitra

Step 1: Assessable Outcome

	Review Criteria	Simple Feedback	Comments
1.1	Are the outcomes viable?	Yes	
1.2	Are the outcomes learner/customer centered?	Yes	
1.3	Are the outcomes specific, clear, and concise?	Yes	
1.4	Are the outcomes measurable?	Yes	

Step 2: Identify Methods & Measures

	Review Criteria	Simple Feedback	Comments
2.1	Are the outcomes assessed with Embedded Measures?	Yes	
2.2	Are the outcomes assessed with Direct Measures?	Yes	
2.3	Are the outcomes assessed with Indirect Measures?	No	
2.4	Do the measures appear to be valid and reliable?	Yes	

Step 3: Criteria for Success

	Review Criteria	Simple Feedback	Comments
3.1	Does every method/measure have a predetermined criterion for success?	Yes	
3.2	Are the criteria for success appropriate?	Yes	

Step 4 (2017-2018): Data Collection and Analysis

	Review Criteria	Simple Feedback	Comments
4.1	Is there sufficient description of the data collection (e.g. student population, sample size, etc.)?	Yes	
4.2	Is there sufficient description of the data analysis procedures and results?	Yes	
4.3	(If used) Is the rubric calibrated?	N/A	

Step 5 (2017-2018): Improvement Actions

	Review Criteria	Simple Feedback	Comments
5.1	Are there any planned or implemented improvement actions based on the assessment results?	Partially	Action items are provided for a couple of the SLOs that did not meet goals, but they can be made more clear and focused.

Recommended Next Steps

Solid assessment practice: Please continue to work with your college/division assessment liaison for fine-tuning.

General Comments

Figure VII.1: Feedback 2017-2018 Assessment Report

Survey Results for the MSE Program

The following figures show the results of student surveys taken for the past two years during the new student orientations and midpoint workshops. The surveys show high satisfaction with the program.

First, we asked new students about their work experience. Almost all students have work experience.

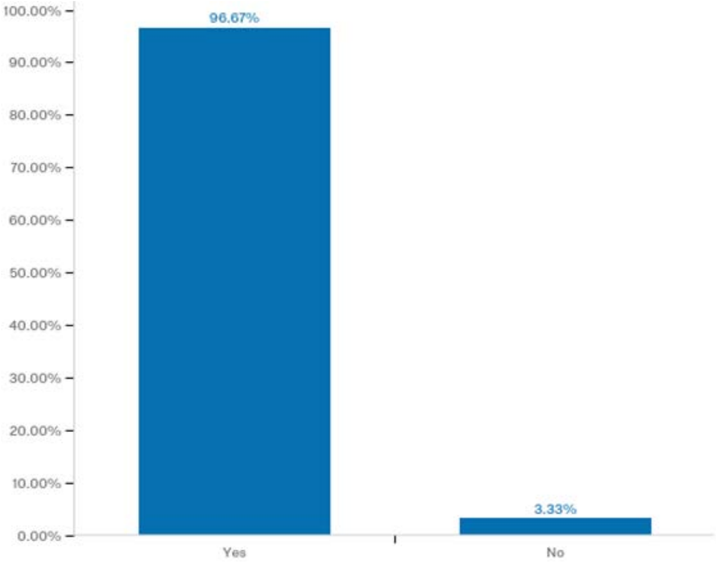


Figure VII.2: Do you have work experience? (Survey taken in Fall 2018)

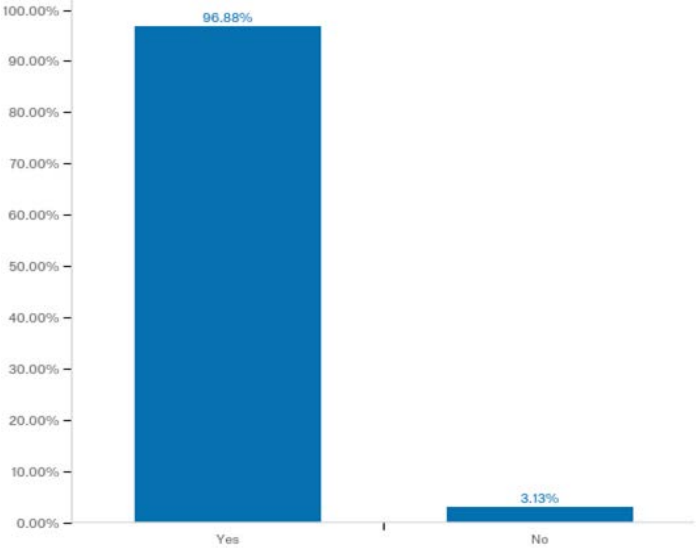


Figure VII.3: Do you have work experience? (Survey taken in Fall 2017)

Second, we asked how they found the MSE program. Most students found the program either through colleagues/friends who are alumni of the program or otherwise know about the program, or through Internet search (e.g., Google, survey organizations that have ranked our program highly).

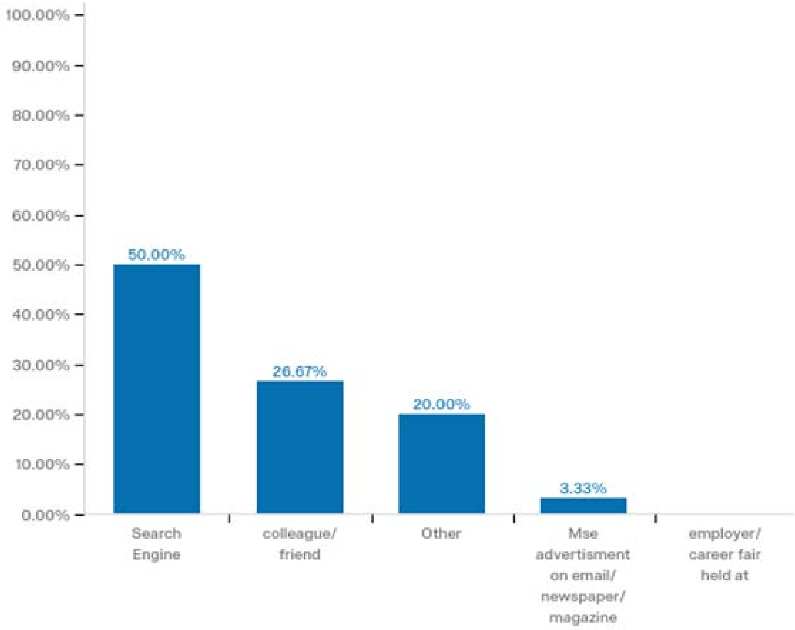


Figure VII.4: How did you find out about the program? (Survey taken in Fall 2018)

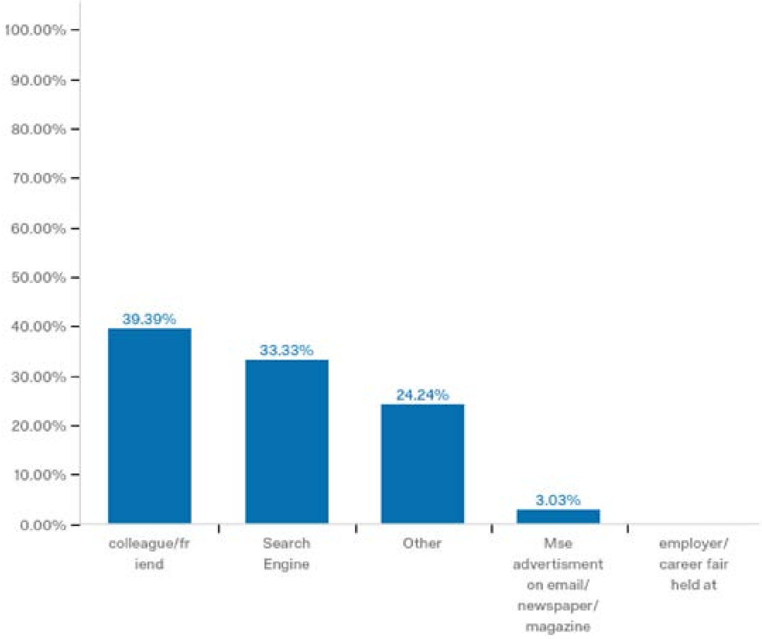


Figure VII.5: How did you find out about the program? (Survey taken in Fall 2017)

Students report that their motivation for enrolling the program is career advancement and promotion. This response is consistent across years.

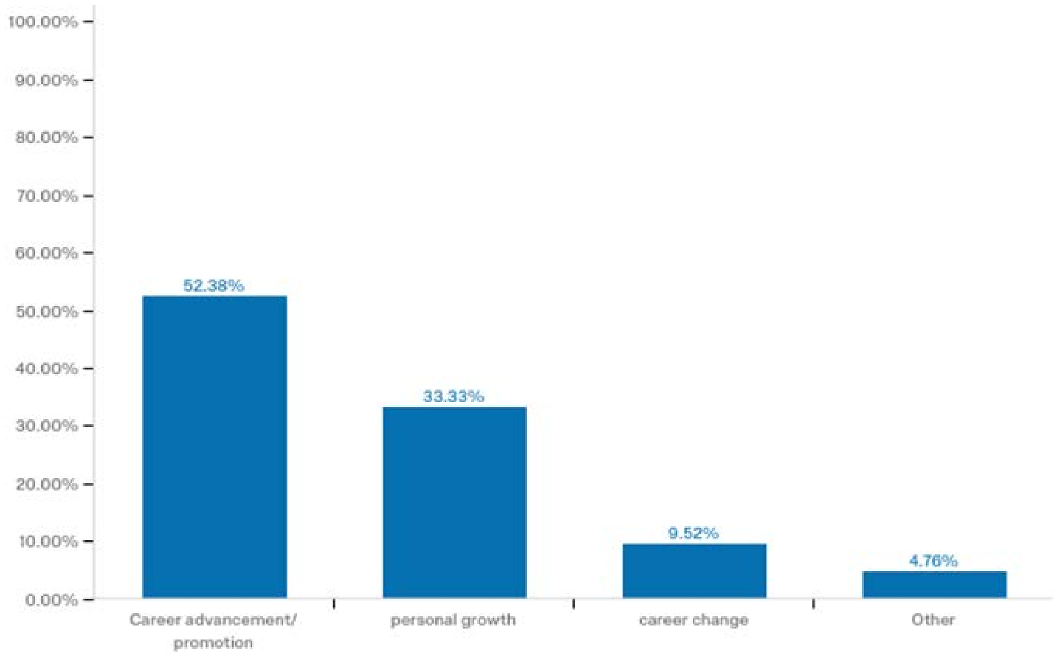


Figure VII.6: Student motivation for enrolling the MSE program (Survey taken in Fall 2018)

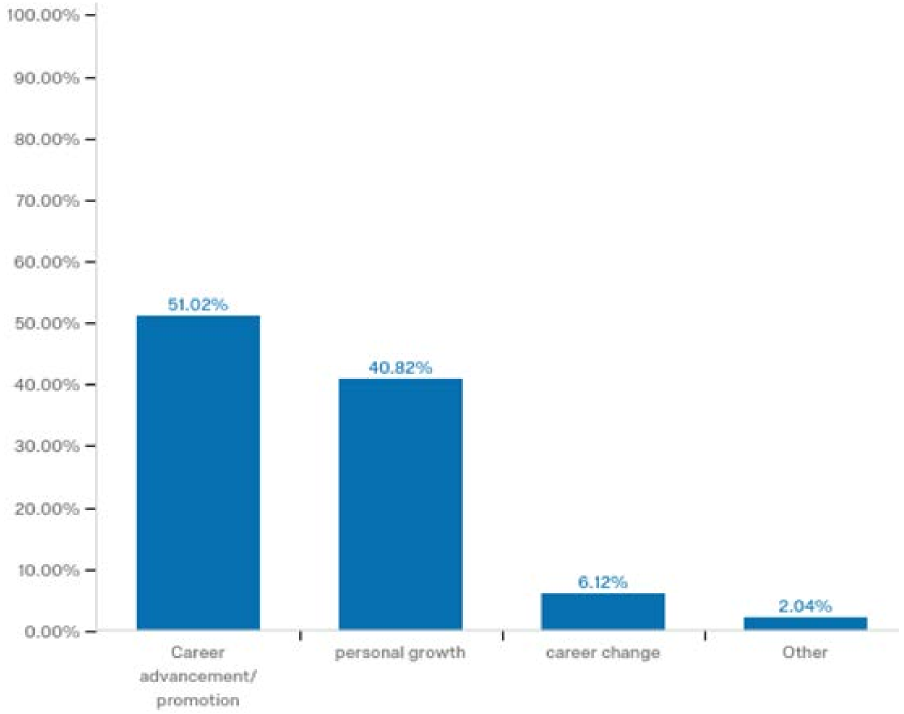


Figure VII.7: Student motivation for enrolling the MSE program (Survey taken in Fall 2017)

Almost all students are satisfied with the program. This response is consistent across years.

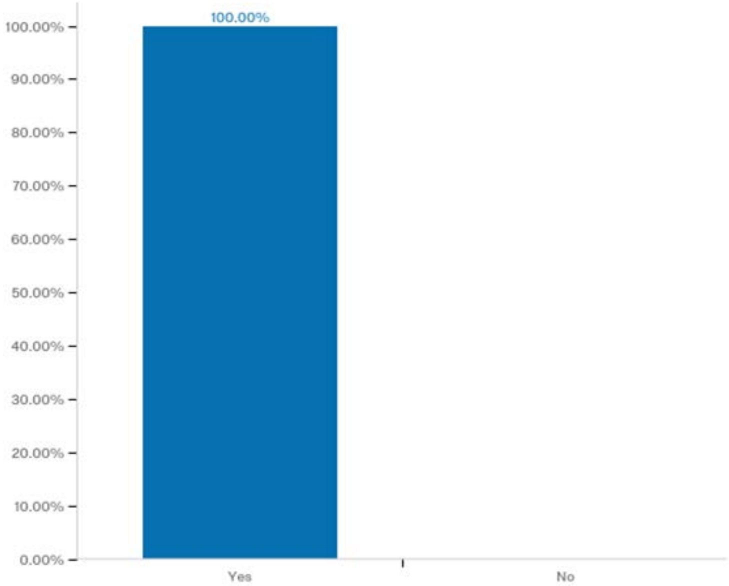


Figure VII.8: Student satisfaction with the program (Survey taken in Fall 2018)

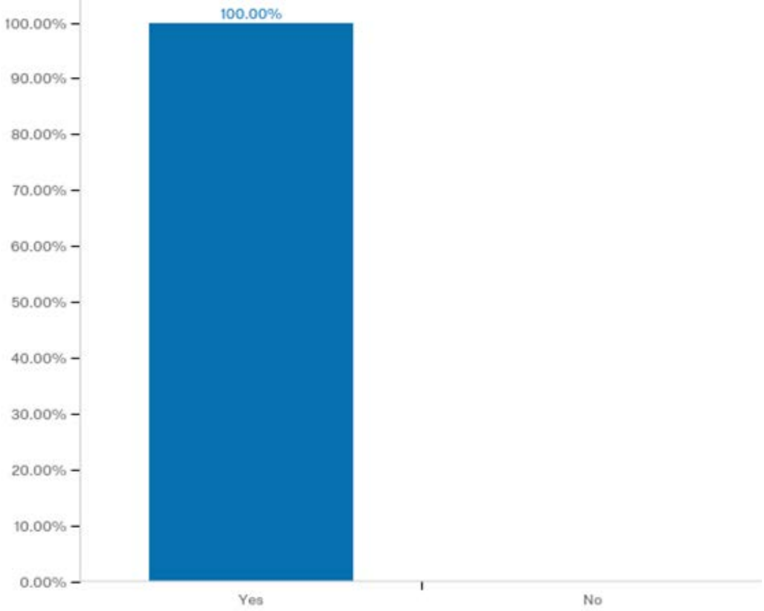


Figure VII.9: Student satisfaction with the program (Survey taken in Fall 2017)

Almost all students are satisfied with the curriculum. This response is consistent across years.

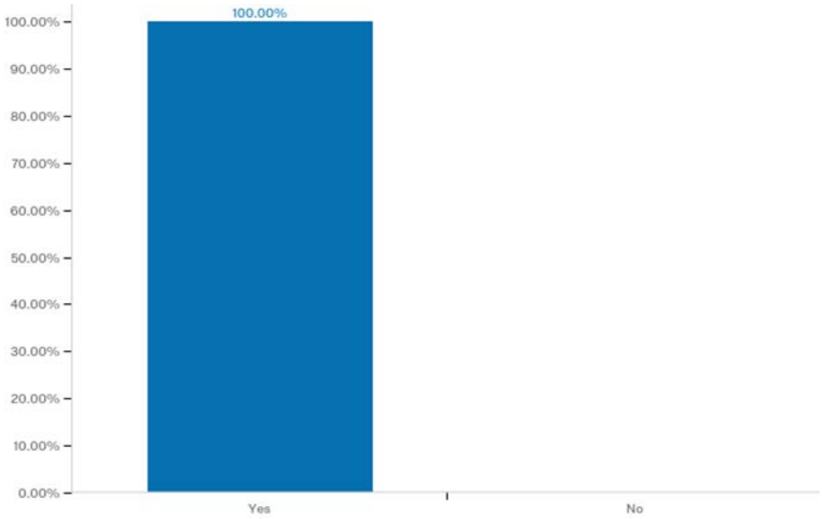


Figure VII.10: Student satisfaction with the curriculum (Survey taken in Fall 2018)

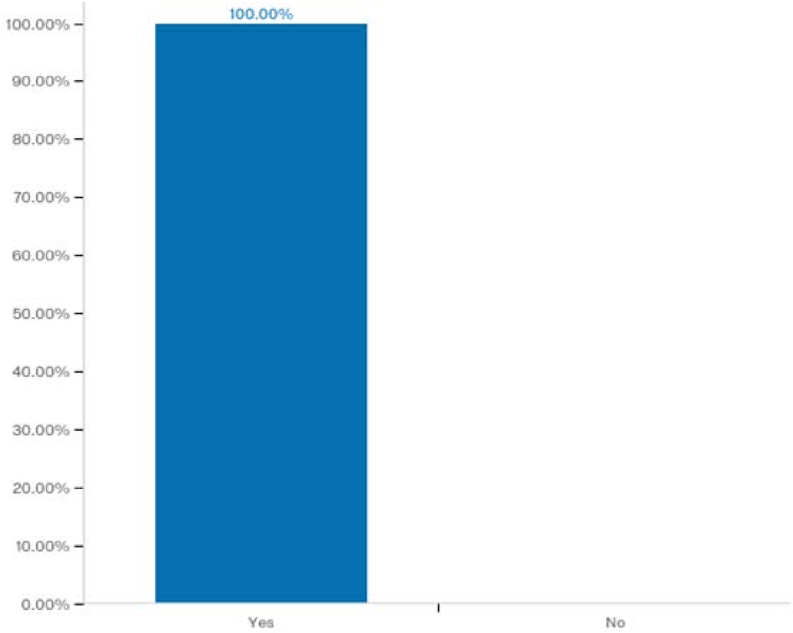


Figure VII.11: Student satisfaction with the curriculum (Survey taken in Fall 2017)

Following are the survey results from ten years ago. They show that student satisfaction with the program and curriculum are consistently very high.

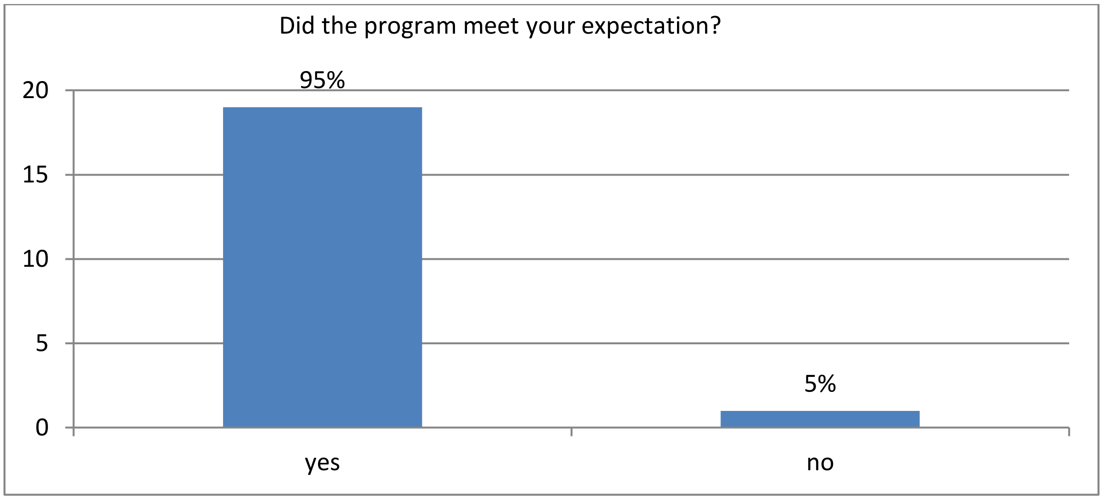


Figure VII.12: Student satisfaction with the program (Survey taken in Fall 2009)

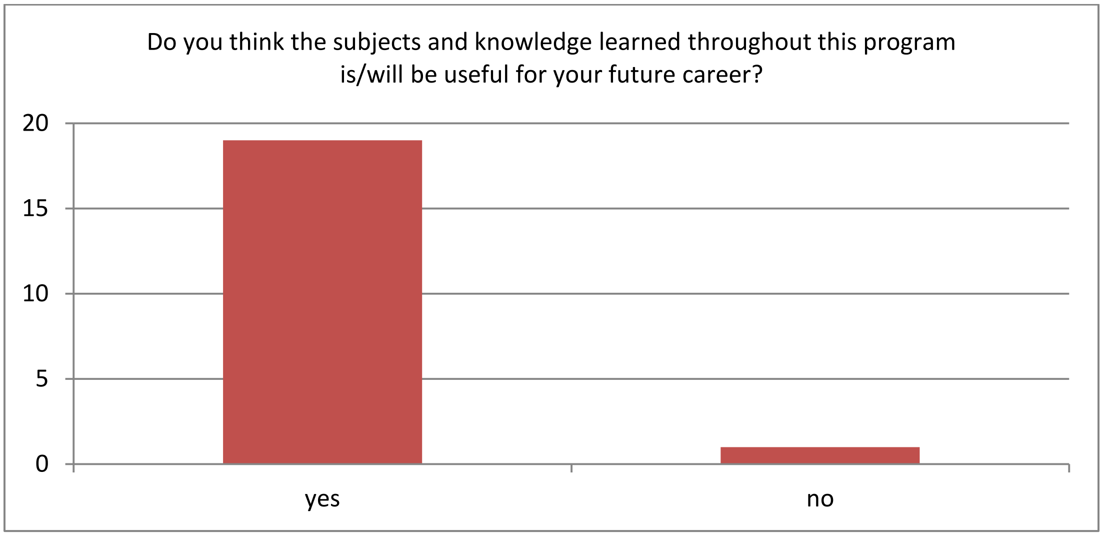


Figure VII.13: Usefulness of the subjects covered in the program (Survey taken in Fall 2009)

Half of students are sponsored by their employers. We may interpret the very high satisfaction rates for this program and curriculum (see above figures) as follows. Regarding the sponsored students, their employers believe their educations (our curriculum) are contributing to their organizations; and students themselves believe this program and curriculum help their work. The non-sponsored students also believe the program and curriculum helped their career advancement and thus that it was worth it to invest in the program. Notably, almost all MSE faculty members have received emails from students that say the program significantly helped their career advancement.

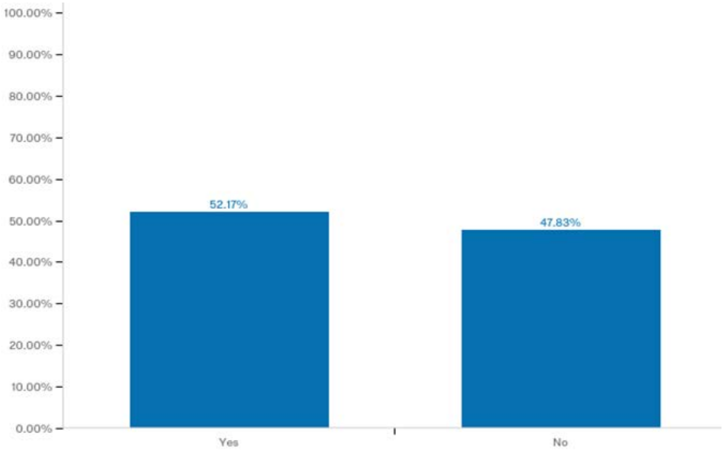


Figure VII.14: Does your employer sponsor any or all part of study? (Survey taken in Fall 2018)

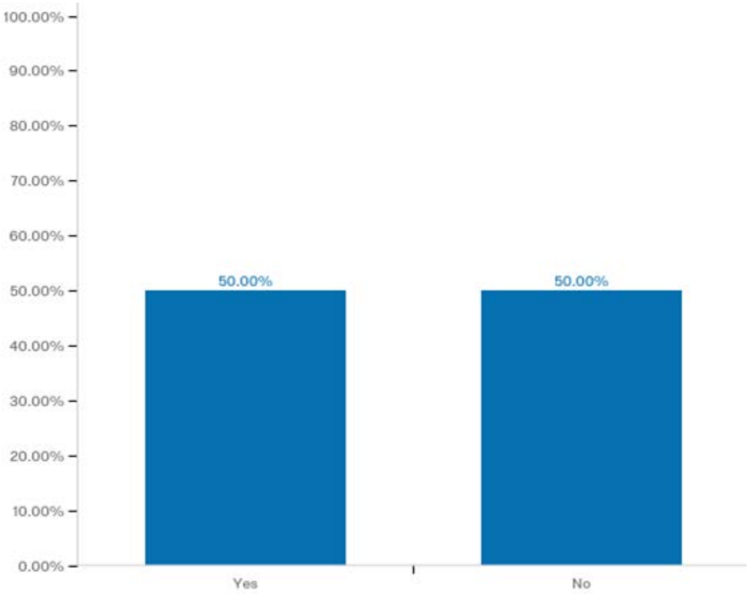


Figure VII.15: Does your employer sponsor any or all part of study? (Survey taken in Fall 2017)

MSE Program's Ranking in the Public

There are many public ranking organizations. The MSE program has achieved and maintained high rankings for many years.

For example, the US News Survey in 2019 ranks the ECS Online Program (both MS in Software Engineering and Environmental Engineering) at 26th among 96 participating schools (94 ranked and 2 un-ranked). We are the 3rd among California schools, and the 1st among non-PhD granting universities. There are also many other survey organizations that have ranked us very high (5th – 10th).

Following are summaries of the rankings with their sources:

- US News and World Report: 26th (3rd in California, 1st among non-PhD granting universities) (for both MS in Software Engineering and Environmental Engineering) [https://www.usnews.com/education/online-education/california-state-university-fullerton-OENG0016/engineering]

- Best Computer Science Schools: 5th (for the MSE program) [https://www.bestcomputerscienceschools.net/best-online-masters-in-software-engineering-degrees/]

- Best Colleges (Best Online Master's in Software Engineering Programs): 5th (for the MSE program) [https://www.bestcolleges.com/features/top-online-software-engineering-degree-graduate-programs/]

- Online Masters (Best Online Master's in Software Engineering Programs): 10th (for the MSE program) [https://www.onlinemasters.com/best-degree-programs/software-engineering/]

Appendix VIII. Course Descriptions

COMPUTER SCIENCE 541 (SYSTEMS AND SOFTWARE STANDARDS AND REQUIREMENTS)

Course description

This course covers the modern software development processes, including the differences from conventional processes; the requirements analysis, management, and specification activity of the modern process; IEEE Software Engineering Standards; establishment of standards for work product, process, requirement analysis, management, and system integration; and a process assessment framework, the Capability Maturity Model Integration (CMMI).

Learning goals

- Understand the activities; workflows; artifacts of software development; and the importance of concurrency, iteration, and risk reduction.
- Understand the CMMI, a means for assessing and improving the development process.
- Ability to analyze the requirements of a software system, and specify those requirements clearly and completely before designing and implementing that system.
- Understand and practice IEEE Software Engineering Standards.
- Ability to establish standards for work product, process, requirement analysis, management, and system integration.

Assessment strategies

Traditional examinations and homework assignments are used to assess each student’s understanding of basic concepts. A group project is used to assess each student’s ability to apply the knowledge they’ve gained in the course. Online discussion is another method used in this course.

Learning goals linked to program goals

- Software standard comprehension
- Implementation of software process
- Process assessment and appraisal
- Collaboration and team work
- Critical thinking and problem solving
- Communication

COMPUTER SCIENCE 542 (SOFTWARE VERIFICATION AND VALIDATION)

Course description

The students will learn how to ensure the development of a high quality software product. The course will cover the theory and practice of software verification and validation (V&V) such as software integrity levels, minimum V&V tasks for each software integrity level, walkthroughs, inspections and cleanroom. We will also cover important software testing topics: white-box and black-box testing, boundary value analysis, equivalence class partitioning, unit testing, functional testing and creation of test plans.

Learning goals

- Understand the theory of software testing, verification and validation
- Understand the process of software testing, verification and validation
- Ability to apply technical testing methods and use tools
- Ability to plan software testing, verification and validation
- Ability to organize and manage testing group

Assessment strategies:

- Online discussions
- Reports
- Presentation – Power Point presentation with voice required in project
- Project – one project (A Case Study of Using HP QuickTest Professional)
- Exams

Learning goals linked to program goals

- Implementation and management of software process
- Process assessment and appraisal
- Software process standards
- Collaboration/team work
- Critical thinking and problem solving
- Legal and ethical issues, and media literacy
- Research
- Communication

COMPUTER SCIENCE 543 (SOFTWARE MAINTENANCE)

Course description

Students will learn the principles of generating maintainable software. The course will also cover theory and practice of maintaining large-scale software and application of maintenance metrics. In this course, we will cover maintenance framework and maintenance process, the problems in maintaining software systems, building software in view of the maintenance problems, the activity of maintenance and some management issues in maintenance. Each student is required to work on a project in the area of software maintenance.

Learning goals

- Understand the principles of software maintenance, issues of software maintenance, generating maintainable software, and maintenance framework and process.
- Understand and ability to use software maintenance models.
- Ability to manage maintenance process.

Assessment strategies

- Exams
- Class participation through online discussions
- Chapter summaries: After student reads each chapter, the student needs to write about 1-2 pages of summary of chapter in his own words, such as, important concepts, lessons learned, applications to his own company/job etc.
- Term project

Learning goals linked to program goals

- Software process standards
- Implementation of software process
- Critical thinking and problem solving
- Research
- Communication

COMPUTER SCIENCE 544 (ADVANCED SOFTWARE PROCESS)

Course description

This course provides a practical guidance for improving the software development and maintenance process. We focus on understanding and managing the software process because this is where software organizations encounter the most serious problems.

Learning goals

- Understand the principles of software process and process improvement
- Ability to establish an effective software process for an organization
- Ability to improve the existing process

Assessment strategies

The assessment methods used in this course include homework assignments, exams, online discussions, and group projects.

Learning goals linked to program goals

- Implementation and management of software process
- Process assessment and appraisal
- Collaboration and team work
- Communication

COMPUTER SCIENCE 545 (SOFTWARE DESIGN AND ARCHITECTURE)

Course description

In this course, students learn how to architecturally design software systems. Case studies and homework are assigned as a practical component of the course.

Learning goals

- Ability to analyze and design software
- Ability to apply different architectural styles to software design

Assessment strategies

Assessment is done with the results from homework assignments, exams, and online discussions.

Learning goals linked to program goals

- Software standard comprehension
- Critical thinking and problem solving
- Collaboration and team work
- Communication

COMPUTER SCIENCE 546 (MODERN SOFTWARE MANAGEMENT)

Course description

This course will cover modern project management methodologies and techniques. Students will learn how to plan, estimate, and schedule the project; organize and direct the team; measure, monitor, and track the progress; control the project; and perform risk management. Students will also learn about quality software development process and management methodologies through the automated software defect prevention" (ADP) as well as other development process models such as Rational Unified Process, Agile, and Spiral model.

This course also meets the upper-division writing requirement for graduate students in the MSE program. Accordingly, there will be several papers to write. Part of the grades on discussion group, project and examinations will be based upon writing skills.

Learning Goals

- Ability to plan, estimate, and schedule the project; organize and direct the team; measure, monitor, and track the progress; control the project; and perform risk management
- Understand and ability to practice quality software development process and management methodologies through the automated software defect prevention as well as other development process models such as Rational Unified Process, Agile, and Spiral model.

Assessment Strategies

- Homework assignments
- Exams
- Survey or research paper writing
- Online discussions
- Team project and presentation

Learning goals linked to program goals

- Software standard comprehension
- Implementation and management of software process
- Professional, legal, and Ethical issues
- Critical thinking and problem solving
- Collaboration and team work
- Research
- Communication

COMPUTER SCIENCE 547 (SOFTWARE MEASUREMENT)

Course description

This course explores current software measurement practices. Topics include measuring: software specifications and designs, software code and implementation, and software testing and evaluation. The course stresses a practitioner-based approach to systematic measurement process model and outlines major paradigms for selecting measures.

Learning goals

- Understand the principles of software process, management, measurements, and analysis
- Ability to plan and perform software measurement process
- Ability to collect software process data and analyze process data using statistical analysis techniques including various control charts and tools
- Ability to analyze software process behavior and determine the stability and capability of the process
- Ability to identify the assignable causes in a process
- Ability to suggest the various options for process improvement

Assessment strategies

- Online discussions
- Homework assignments
- Reports and presentation
- Group project
- Exams

Learning goals linked to program goals

- Implementation and management of software process
- Process assessment and appraisal
- Collaboration/team work
- Critical thinking and problem solving
- Research
- Communication

COMPUTER SCIENCE 548 (PROFESSIONAL, ETHICAL AND LEGAL ISSUES FOR SOFTWARE ENGINEERS)

Course description

This course provides students with foundations of professional ethics and legal issues in Software Engineering and also the application of these ethics and legal issues related to computer networking, intellectual property, privacy, security, and reliability issues in the age of the information technology.

Learning goals

- Understand the foundations of professional ethics and legal issues in Software Engineering
- Ability to apply and practice the software engineering code of ethics to the areas of modern computing
- Ability to work with peers in a team setting
- Ability to communicate with peers and instructor using appropriate medium

Assessment strategies

- Exams
- Online discussions
- Assignments
- Paper writing
- Term project

Learning goals linked to program goals

- Professional, legal, and Ethical issues
- Critical thinking and problem solving
- Collaboration and team work
- Communication

COMPUTER SCIENCE 597- I (GRADUATE PROJECT IN COMPUTER SCIENCE)

Course description

A project is a significant undertaking appropriate to the fine and applied arts or to professional fields. This course is to learn how to find a project or research topic through a proper survey in the interested areas, acquire the necessary knowledge in the selected area, write a proposal, and present the proposal. After the course, students are expected to know the process of finding a project or research topic, the survey techniques, and writing a good proposal and its presentation.

Learning goals

- Understand the process of finding a project or research topic, survey techniques, and how to write a good proposal and its presentation.
- Ability to find a project or research topic through a proper survey in the interested areas and acquire the necessary knowledge in the selected area
- Write a proposal, and present the proposal

Assessment strategies

- Survey paper writing
- Online discussions
- Proposal writing
- Presentation

Learning goals linked to program goals

- Software standard comprehension
- Professional, legal, and ethical issues
- Critical thinking and problem solving
- Collaboration and team work
- Research
- Communication

COMPUTER SCIENCE 597-II (GRADUATE PROJECT IN COMPUTER SCIENCE)

Course description

This course is to learn how to conduct a development or research project proposed in the selected area, complete it, write a final project report, and present the results. After the course, students are expected to know the process of implementing a project as proposed, be able to establish a process to complete the project successfully, write a high quality project report, and present the results to an audience.

Learning goals

- Understand the process of implementing a project as proposed
- Ability to establish a process to complete the project successfully
- Ability to conduct a development or research project proposed in the selected area, complete it, write a final project report, and present the results
- Ability to apply the knowledge and techniques learned in the program to solve real-world problems

Assessment strategies

- Online discussions
- Progress report
- Final project report
- Presentation

Learning goals linked to program goals

- Software standard comprehension
- Professional, legal, and ethical issues
- Critical thinking and problem solving
- Collaboration and team work
- Research
- Communication

Appendix IX. Glossary and Formulas Used in the Report

Academic Year (AY): A time period that includes a fall semester and the subsequent spring semester. This period excludes summer. Abbreviated AY.
(e.g.)
AY 2016-2017 begins with the fall 2016 semester and ends with the spring 2017 semester.

College Year (CY): A time period that starts with a summer semester, continues through the subsequent fall semester, and ends with the following spring semester. Abbreviated CY.
(e.g.)
CY 2016-2017 includes the summer 2016, fall 2016, and spring 2017 semesters.

FERP: Faculty members who are participating in Faculty Early Retirement Program (FERP). The data shown are all based on fall semester numbers, but those who are on FERP in spring or AY are also counted as participating in FERP.

FTEF: Full Time Equivalent Faculty
(e.g.)
CSU full-time faculty workload is 15 WTUs per semester.
So FTEF is calculated by sum of WTUs/15.
When 2 full-time MSE faculty only teach two MSE courses, then FTEF = (2x15)/15 = 2.

FTES: Full Time Equivalent Students (the total student credit units being taken by master's students divided by 12).
1 FTES = 12 units
(e.g.)
If a course CPSC 545 has 30 students enrolled, FTES of the class = 3 units x 30 students / 12 units = 7.5 FTES

Headcount (HC): Number of students.

SFR: Student Faculty Ratio
SFR = FTES/FTEF or FTEF = FTES/SFR.

Units: Credit hours (3 units = 3 credit hours / course)

WTU: Weighted Teaching Units (workload)