WASHINGTON STATE UNIVERSITY

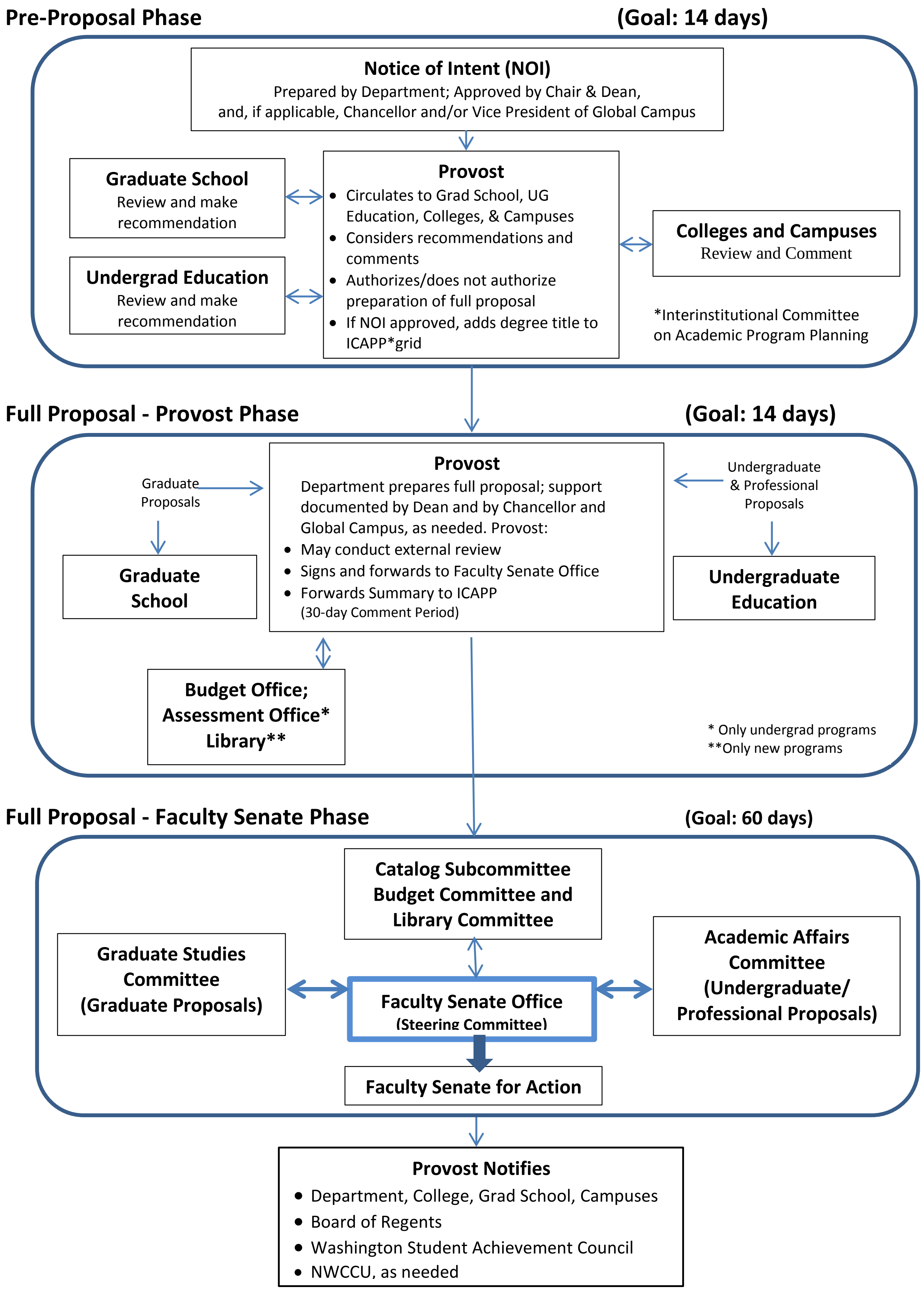
Approval Process, Instructions, and Templates

for Proposing

A New Degree Program
(not currently offered by WSU at any location)

Table of Contents

Approval Process for Creating, Extending, Moving, Consolidating, Renaming, or Eliminating a Degree Program (7/25/14)


Pre-Proposal Phase
(Goal: 14 days)
Notice of Intent (NOI)
Prepared by Department; Approved by Chair & Dean, and, if applicable, Chancellor and/or Vice President of Global Campus
Graduate School
Review and make recommendation
Undergrad Education
Review and make recommendation
Provost
• Circulates to Grad School, UG Education, Colleges, & Campuses
• Considers recommendations and comments
• Authorizes/does not authorize preparation of full proposal
• If NOI approved, adds degree title to ICAPP*grid
Colleges and Campuses
Review and Comment
*Interinstitutional Committee on Academic Program Planning
Full Proposal - Provost Phase
(Goal: 14 days)
Graduate Proposals
Graduate School
Provost
Department prepares full proposal; support documented by Dean and by Chancellor and Global Campus, as needed. Provost:
• May conduct external review
• Signs and forwards to Faculty Senate Office
• Forwards Summary to ICAPP
(30-day Comment Period)
Undergraduate & Professional Proposals
Undergraduate Education
Budget Office; Assessment Office* Library**
* Only undergrad programs
**Only new programs
Full Proposal - Faculty Senate Phase
(Goal: 60 days)
Catalog Subcommittee Budget Committee and Library Committee
Graduate Studies Committee (Graduate Proposals)
Faculty Senate Office
(Steering Committee)
Academic Affairs Committee (Undergraduate/ Professional Proposals)
Faculty Senate for Action
Provost Notifies
• Department, College, Grad School, Campuses
• Board of Regents
• Washington Student Achievement Council
• NWCCU, as needed

Process Rationale

Pre-Proposal Phase (Notice of Intent) The purpose of the Pre-proposal Phase is to inform academic units across the University of potential changes at the degree program level. This enables the identification of any conflicts and possible collaborations with existing or planned programs before significant time is invested in the development of a full proposal. Additionally, the review of pre-proposals within the Provost's Office enables the alignment of proposed degree programs with University goals to be assessed prior to developing a full proposal.

Full Proposal – Provost Phase The purpose of the Provost Phase of full proposals is to assure that financial resources and personnel, learning outcome assessment, and diversity provisions are sufficient. Additionally, the examination enables the identification and resolution of any issues before proposals are forwarded to the Faculty Senate and, when deemed necessary, inclusion of external reviewers, thus facilitating these reviews.

ICAPP Review – Sharing early information and, subsequently, a summary of the full proposal with the other public colleges and universities insures that issues of minimizing unnecessary duplication and enhancing collaboration can be appropriately addressed among the institutions involved.

Washington Student Achievement Council Notification – The WSAC maintains the database of all degree programs offered in the state of Washington, as well as of the programs approved for students using veterans' benefits, and they are responsible for identifying underserved regions and populations, as well as fields of study where there is unmet state need. So it is important that their inventory of programs is kept up to date.

Instructions

1. Complete and submit a **Notice of Intent** (NOI) for the proposed new program to your dean and, if applicable, chancellor and/or vice president of the Global Campus.

2. *Once the NOI has been approved by the Provost Office*, please electronically complete Workbook 1, Analyzing Library Capacity and Workbook 2, Analyzing Demand and Cost using as much space as necessary for each item – do not be constrained by the space between questions on the template. Please work through the Workbooks before completing the Proposal Template and forward them along with the Proposal Template itself. (Details in the Workbooks will be used internally, but will not be forwarded to external entities or reviewers.)

3. Complete and forward the Proposal Template (along with the Workbooks) to your college dean(s) and, if applicable, chancellor and/or vice president of the Global Campus. The dean will forward the proposal electronically to the Provost's Office (donnac@wsu.edu).

4. If the new degree will include new courses or new program requirements, submit the required change forms (https://www.ronet.wsu.edu/ROPubs/Apps/HomePage.ASP) directly to the Registrar's Office at the same time the proposal itself is sent to the Provost's Office. *The Faculty Senate Curriculum Subcommittee will not consider new program proposals until new courses and requirements for that program have been approved.*

5. The completed Proposal Template will be forwarded to the Faculty Senate Office for review by the appropriate committees and the Faculty Senate. A summary of the proposal will also

be forwarded to the Interinstitutional Committee on Academic Program Planning (ICAPP) for posting on the Council of Presidents website for 30-day statewide comment (see diagram) and, as appropriate, to the Northwest Commission on Colleges and Universities.

6. In addition to your department and college, the following resources can answer questions:

a.	Graduate School	335-3535
b.	Provost's Office	335-5581
c.	Budget Office	335-7783

Templates

Notice of Intent: Complete and submit the template at XXXXX to your dean and, if applicable, chancellor and/or vice president of the Global Campus (see diagram).

Workbooks Complete the following two workbooks: one to assess the adequacy of library holdings and services, and a second to assess student demand and costs. The workbooks should be completed prior to the Proposal Template.

Workbook 1 – Analyzing Library Capacity

The Faculty Senate Library Committee reviews all proposals for new degree programs for adequacy of library holdings and services. To assist the committee in its deliberations, please address the topics below in your proposal in collaboration with the librarian(s) responsible for collection development in your discipline(s). The names of appropriate librarians are available from the Director of Libraries at 335-4558 or from your dean's office.

1. In specific terms, describe the adequacy of existing capacity:

Questions to ask:

- How adequate are the existing library collections for the proposed program?
- How adequate is the existing library equipment for the proposed program?
- How adequate are the existing personnel and services for the proposed program?
- How will this program contribute to the funding of existing serials, given their ever increasing costs?

The current library collections, equipment, and personnel and services are adequate to meet the needs of the proposed program.

2. What is the need for new library collections:

Areas to consider:

a. Serials (e.g., journals or indexes in print, electronic format, microform, etc.):
1) List new serials titles (and costs) that will be needed.
2) What funds have been designated for these titles and for the ongoing serials subscriptions?
3) Can any of your current serials subscriptions be cancelled to purchase the new titles?
4) What additional library equipment will be needed and how will it be funded (e.g., terminals, CD-ROM readers, etc.)?

b. Monographs (e.g., books in print, electronic format, etc.):
1) Will monographs need to be purchased?
2) Have continuing funds been designated for these and future purchases?
3) What additional library equipment will be needed and how will it be funded (e.g., terminals, CD-ROM readers, etc.)?

c. Media (e.g., films, videotapes, sound recordings, etc.):
1) Are media materials needed?
2) Have funds been designated?
3) What additional multimedia equipment will be needed and how will it be funded?

No new library collections are needed for the proposed program at this time.

3. What new library personnel will be needed?

Questions to ask:

- Will specialized expertise be required to serve your new program?
- Will additional library staff or faculty need to be hired?
- If so, how will the position(s) be funded?

No new library personnel will be needed for the proposed program at this time.

4. What additional library services will be needed?

Questions to ask:

- To what extent will additional interlibrary loan services be required?
- On-line network access?
- References services?
- Library user education?
- If so, have funds been designated for this purpose?

No new additional library services are needed at this time.

5. For programs offered away from the Pullman campus: To what extent will collections and services be provided from Pullman and to what extent by other campus or local libraries?

The majority, if not all, library services will be online and can be provided from Pullman.

6. Are there any other library resource considerations *(e.g., additional space):*

No.

Workbook 2 – Analyzing Demand and Cost

Situational Analysis: The purpose of this section is to identify the strengths and weaknesses of the department(s) as they relate to competition.

Strengths: Why is your department/school able to provide the proposed new degree better than other WSU departments/schools or other universities?

New faculty expertise in Software Engineering and existing faculty expertise in complementary areas of Computer Science and Computer Engineering create a unique and advantageous opportunity for the School of EECS to offer the proposed new online MSSE degree.

Weaknesses: What characteristics of your department or school disadvantage it in offering the proposed new program relative to other WSU departments/schools or competitor universities? Why might other WSU departments/schools or universities be equally or better able to offer the proposed new degree?

None. No other department/school at WSU is positioned to offer the proposed degree at the proposed minimal costs.

Opportunities: Opportunities, as related to this degree program, are developed from your department's/school's strengths or positive circumstances.

Questions to ask:

- What is happening in the state/nation/higher education now that we can take advantage of?
- How can we best take advantage of it?
- How long will this "window of opportunity" be available?

The demand for software engineers, especially in state of Washington, is at an all-time high. WA state government and legislator as well as WA industries are making huge investments in education and training of computer scientists and software engineers as demonstrated by the recent state's engineering expansion investments. With faculty expertise in computer science and engineering and adding new expertise in software engineering, the School of EECS is well positioned to establish the proposed program and address one of the state's highest workforce needs. The demand for software engineers is expected to continue to grow till 2020 and beyond.

Threats: A threat is a problem. Relative to the proposed degree, is there anything that appears to endanger your current situation or future opportunities?

Questions to ask yourself:

- What uncontrollable factors can influence our success?
- What is the worst that is likely to happen?
- For how long is the threat likely to continue?
- How can we eliminate or minimize its effects?

No major threats are expected.

Competitive Analysis: The competitive environment includes other WSU departments/schools as well as competitor colleges and universities, both public and private.

Determine who your top competitors are. Examine other institutions providing a similar program. Be aware that the "competitor" may not look like Washington State University and may not provide education in the same manner that you are proposing. For example, the new online MIS program might compete for the same students not just with other MIS providers but also with some technical training and computer science programs. Don't think too narrowly in this area. Choose competitors whom you believe are actively seeking the students you would like to attract. Competitors may include similar programs at WSU.

Select a strongest, geographically nearest, and lowest price competitor that are accessible to the same pool of students, and describe each of them as completely as possible using the following characteristics:

Name of program and credit hours – indicate the program that is currently being offered. Theirs may not be exactly the same as yours, but should be similar enough to be considered a competitor.

Total Enrollment - number of existing students enrolled in this certificate and/or program.

Total Cost for Certificate and/or Program and Cost per credit hour

Access – what medium is used to communicate with the students?

Faculty to student ratio

Support Services – Other than the instructor, what staff and/or services are provided for the student? How does the student gain access to these support services?

How long has this certificate and/or program been offered? – If not currently offered, what is expected timing of entry into the market?

What is each program's weakness? – Think in terms of areas that may work to your advantage.

What is each program's advantage? - What specific characteristic makes each institution "stand out"? Why would someone choose the other program over yours? This is also called a differential advantage – the trait that makes you "different" and puts you at an advantage. This should help you in determining what marketing strategy you will take. For example, if you know that one of the others is “cheaper”, you can then decide if you want to lower your prices to compete head-to-head, or take the “quality” approach in marketing your program.

What is each competitor’s market share? - What percent of the total market for this type of program belongs to each institution?

Example: **Market:** all students enrolled in 4-year public colleges in state (WA)
Market size: 89,200 students (source: internet sites for all 6 colleges)
WSU enrollment: ~ 28,000 (all locations) (source: www.wsu.edu)
WSU’s Market share: approximately 31%

Software engineering graduates are rare and easily demand 6-figure salaries upon graduation. While other states such as California, Massachusetts, Pennsylvania, Illinois, and Indiana have committed to building software engineering programs and producing SE graduates, the State of Washington is lagging in this area. To the best of our knowledge, currently there is no online Master of Science in Software Engineering degree offered at any of the 4-year institutions of higher education in the Sate. University of Washington has a face-to-face, evening professional Master’s degree in Computer Science and Engineering and UW-Bothell offers a face-to-face Master of Science program in Computer Science and Software Engineering. Western Governors University offers competency-based online MS degrees in Information Technology, but not in Software Engineering.

Therefore, the proposed new online MSSE program at WSU Global Campus will be unique and will help the State and its computing/IT industries meet their needs by producing highly skilled and trained graduates in software engineering.

Washington State University:

Additionally, are there any barriers that might inhibit WSU from entering this market? These might include required economies of scale, brand identity, accreditation standards, known plans of competitors, access to distribution, switching costs and government policy.

No barriers are identified that might inhibit WSU from entering this online market.

Competitor 1 ____________________________

Competitor 2 ____________________________

Competitor 3 ____________________________

Demand Analysis:

Employer Demand:

Employer demand is defined as the number of program graduates needed to fill current and anticipated job openings. Please include information from professional societies and their publications, industry advisory groups and advocacy groups, internal studies, department of education, department of labor, or employment security department statistics, letters of support, and other sources to estimate current employer demand for graduates of the program.

Questions to ask:

- What is the national employer demand for graduates in this program area?
- Is national employer demand trending upward or downward?
- What is the regional and local employer demand for graduates?
- Is regional and local employer demand trending upward or downward?

Access to baccalaureate and graduate degrees continues to be a major issue curbing overall educational attainment and nationwide economic competitiveness in Washington. The problem is most acute in the areas of computer science (CS) and software engineering (SE). A 2011 joint report prepared by the Washington State Higher Education Board, State Board for Community and Technical Colleges, and Workforce Training and Education Coordinating Board indicates that during the 2014-2019 period there will be a ratio of nearly 2 to 1 between demand and supply in the State for graduates with BS degrees in CS or SE.

A recent comprehensive market study by Global Campus shows that software engineering skills are highly sought after in the workplace. Employment in the area of software engineering is growing rapidly with jobs such as Software Development Engineer, Software Developer, and Software Engineer growing about 87%, 56% and 74%, respectively, in Washington alone in 2012-2013. Employer demand is expected to continue this trajectory through 2020.

Data from the second half of 2013 show that 4,483 regional jobs posted requiring Software Engineering skills at graduate level and 11,881 regional jobs posted requiring Software Engineering skills at Bachelor's level.

Student Demand:

Student demand is the number of qualified students desiring to participate in your program. Student demand is determined by several factors including:

Market – the geographic area from which the program will attract students.

Questions to ask:

- Where are potential students physically located? (e.g., international, national, state-wide, regional, local, etc.)
- Would potential students be required to relocate or can they remain at home via distance-learning?

Since this is a distance-learning (online) program, there are no geographic limitations for student participation. We expect the majority of the initial online MSSE students to come from the state of Washington, esp. from the west side of the state. However, over the long-term, we anticipate global participation with no geographical boundaries.

Market size – the number of potential students in the market area

Questions to ask:

- What is the current number of students in existing programs in the proposed market area in this field?
- What is the potential number of students forecasted?

In recent years, due to industry demand and high-paying career opportunities, enrollment in computer science and related fields has rapidly grown nationally, within the state of Washington, and at WSU. Our computer science program has grown by an average of 12.5% per year in the past 5 years, almost doubling its enrollment during that time span. All studies indicate that the increase in demand for computer science and related fields, including software engineering, will continue to grow in the next decade due to the strong industry demand for a highly educated, qualified workforce in this area.

Market Segment – the characteristics of students that you intend to serve

Questions to ask:

- What are the characteristics of students currently in the department's programs (age, location, employment, goals, etc.)?
- Why do they choose WSU?
- What kind of students choose to go elsewhere for programs like this? Why?

The characteristics of the majority of the students participating in this program will be that of industry professionals with baccalaureate degrees who are interested in gaining advanced, graduate degrees in software engineering. All participating students strive to have professional careers in computer or programming industries. Many choose this program and WSU due to the high quality of our programs or due to family traditions and ties to WSU.

Market capacity – the upper boundary of a market. This would represent and include every potential student interested in the program within the market area. If all of the needs are served and there is an excess of supply over demand, then the market is considered saturated

Referring to "Market Size," "Employer Demand," and "Competitive Analysis" sections, we believe the proposed program to be unique in the state with plenty of student demand that is not being fulfilled by the state's 4-year higher education institutions.

Growth rate – the rate at which demand is increasing in the target market (geographic area of interest). What is the expected growth rate of student and employer demand?

Questions to ask:

- What are long-term population trends, especially in the target age group?
- Are competitor-institutions planning to introduce similar programs/expand existing ones?
- Is long-term employer demand expected to grow, remain stable, or decline?

Refer to "Market Size," "Employer Demand," and "Competitive Analysis" sections.

Target Market –This is the group of people whose needs you will focus on fulfilling better than anyone else.

Questions to ask:

- Who are they?
- What is their need?
- How will we serve it?

The target market for this program consists of place-bound professionals working in computing or software industry who desire to attain advanced, graduate degrees in Software Engineering.

Estimate the number of individuals you expect to enroll from your target market for the 1st, 2nd and 3rd years. This market segment can be based on demographics -- e.g., the number of students who complete community college in WA each year with an AA degree with a business emphasis, or (for a graduate program) the number of students who graduate with an undergraduate degree in this field in the Northwest. This will help you identify potential trends and your target market.

Your **target market** is usually the segment that has the largest numbers of individuals in it. However, if that segment's needs are already being taken care of by one of your competitors, you may wish to target another group or go for the specialty "niche," or secondary market. Note that it may be better to target 50% of a smaller group rather than 2% of a global market.

	1st year	**2nd year**	**3rd year**
Target	___20__	___30___	___40___

To whom will your marketing efforts be directed? What are the key characteristics of that segment to which you will appeal?

TARGET MARKET:	**Characteristics:**
State Computing and Software Industries	*All professionals or individuals with a BS degree in computer science or related field who may be interested in attaining an advanced, MS degree in Software Engineering.*

Recruitment Plan:

1. How and where are students going to find out about this program?

Through websites and brochures, utilizing the market operations of the WSU Global Campus.

2. Who will represent this department in its promotion activities?

One of the senior software engineering faculty hires will take on the program coordination role and will represent the department in its promotion activities.

3. What specific venues can you use to promote an awareness of this new program?

Marketing and recruitment plans will be developed and executed by WSU Global Campus.

4. What means will be used to access and educate businesses, industry, agencies, and/or institutions about this offering?

Through our existing industry advisory board, current connections and links with the industry, websites, and brochures.

Financial Analysis:

A major factor in determining whether or not a proposed program is viable is financial feasibility. The following Excel spreadsheet contains five tables, which can be pasted into the proposal template document after it is completed. Both your college's Finance Officer and the University Budget Office are available to assist you with this spreadsheet.

New
Program-Location Cos

Enrollment Projections – Table 3
Enrollment objectives are established to provide a measurement of the cost of the program. They are based on expected enrollment trends and the capacity of your unit to realize those opportunities by meeting student needs. Use Table 3 of the spreadsheet to report enrollment projections.

When considering a new program, the focus will be on its cost per student FTE (full time equivalent). There are guidelines available in the Budget office to help the University assess which programs are high or low cost, compared to other programs in the same discipline, as well as the overall cost mix for all University programs.

Faculty Participation – Table 4
Use Table 4 of the spreadsheet to list the faculty resources required by the new program.

Cost Projections – Table 5
Many of the expenses involved in creating new programs can be absorbed into the existing structure. However, a new program can add fixed and variable costs that significantly impact the financial analysis.

- **Fixed** costs are independent of the number of students in the classes – they will not change (not considering inflation) as you move from Year 1 to Year N when you reach what you consider to be "Full Enrollment."

- **Variable costs** are those costs that vary depending upon the number of students. These costs will grow from Year 1 to Year N. Some costs exhibit a step function pattern; that is, they are fixed for X number of students, but increase for X+1 students and again for 2X students. For purposes of this worksheet, assume your enrollment goals will be met.

If you are using similar kinds and sizes of courses and similar methods of delivery as an existing program, you may be able to project the costs of the new program fairly closely by determining the cost of the existing program. If this site will use different delivery methods, start with fewer students, or otherwise differ from the existing one, this may not be the case. Check with your college's Finance Officer or the Budget Office for assistance.

Direct and indirect expenses must be considered in the financial analysis. **Direct expenses** are specifically tied to the proposed program and include:

- Instructor salaries and benefits
- Administrator salaries and benefits
- Clerical Support salaries and benefits
- Graduate Assistant salaries and benefits
- Equipment costs
- Travel costs
- Goods and Services – phones, copying, etc.
- Classroom materials costs
- Other

Indirect expenses are costs that are often associated with existing or additional support services that increase incrementally because of the addition of the program. These costs should not be confused with the Facilities and Administrative (F&A) costs that are applied to grants and contracts. The indirect costs related to new programs are the facilities, academic support, administrative support and student services costs that are in place to support the delivery of the University's academic programs. The Budget Office tracks the overall cost of these services, and the appropriate rate is included in the template.

Note: if you are developing a new program that will be delivered via the Global Campus, you should reduce the indirect percentage to 0.32 in the template. Please call the budget office if you have questions).

Opportunity costs are the costs of not doing something else. They are not included in Table 5, but should be kept in mind. For example, if an instructor or other existing resources are "re-allocated" to this proposal, what area will be affected and what is the value of these resources? Every time a new program or site is proposed, we should carefully consider that it is subtracting resources from other programs or sites. If a new program or site is not taking resources from other programs, it may imply that we have underutilized resources. How does your proposal address this?

Additional Information for Completing Table 5

- The Internal Reallocation column indicates that the costs within the column will be covered by reallocation resources from other programs within the department or college.

- New State Funds should only be shown as a source when a program is being developed at one of the branches or if implementation will await the availability of new funds (e.g., state-funded High Demand FTEs). Note that listing items in the New State Funds column does not imply or guarantee that these funds will be available when needed.

- Other Sources of funding (e.g., be matching funds for equipment or in-kind resources available to the program).

- Complete the template using your best estimates of the costs to deliver this new program, both in the first year of delivery and in the year that you expect it to reach full capacity (Year N). It is often true that the first year of a program has higher costs per student FTE, as the enrollments in early years are lower than expected full capacity. Over time, as the number of FTE increase, the costs per FTE will decrease.

- The spreadsheet will calculate both the indirect and total costs, as well as the cost per student FTE.

Administrative/Support Staff Participation – Table 7
Use Table 7 of the spreadsheet to list the administrative/support staff FTE resources required by the new program.

Salary Cost Detail – Table 8
Use Table 8 to provide aggregated salary data for the personnel in Table 4 and Table 7 for Years 1 and N. Do not include faculty and staff names in Table 8. Provide only the aggregate salary data for each category in the table.

NOW *SUMMARIZE* THE INFORMATION AND ANALYSIS FROM THE ABOVE WORKBOOKS IN COMPLETING THE PROPOSAL TEMPLATE

New Degree Program Template

The Proposal *Template* leads you to answer the array of questions about your proposed program that are important to your department, your college, the Faculty Senate, the State, and, in some cases, external reviewers.

By placing all proposals in a similar format, this template provides a common standard for comparison, ensuring that all potential programs can be evaluated in an equitable fashion. It can be used to determine whether or not a program is feasible within the university's academic and financial situation, and if it will have the resources to further the University's objective of providing high quality education and scholarship.

Finally, this template can become a framework to think about the viability of your ideas. It can thus be a tool for strengthening both your proposal and the resulting program itself, since a program that is starved for either students or resources from its inception is not likely to become a high quality program.

Here are some of the things you will be asking as you complete the template:

> What are the aspirations for the reputation of this program – local, regional, national? What will it take to make that a reality?
>
> Who are we trying to attract with this new program? Will it bring new students to the university, better meet the needs of current students in the department, or draw students away from other departments?
>
> How strong is the demand for education of this kind, and in what specific careers will someone who receives such an education will find meaningful employment?
>
> How many students do we need to attract to break even, and can both the market and our capacity support this number?

Providing good answers to hard questions maximizes the likelihood that a new program will not just win faculty senate and administration acceptance, but ultimately will be successful in attracting students and placing graduates.

WASHINGTON STATE UNIVERSITY

Proposal to Offer a New Degree Program

I. Overview

Program Title: Master of Science in Software Engineering (MSSE)

Degree (level) Master of (type) Science

In (major or field) Software Engineering

CIP Code (consult registrar): Computer Software Engineering: 14.0903
(**C**lassification of **I**nstructional **P**rograms)

Department(s): School of Electrical Engineering and Computer Science (EECS)

College(s): Voiland College of Engineering and Architecture

Departmental Contact:

Name: Behrooz Shirazi	Title: Professor and Director
Phone: 509-335-8148	e-mail: shirazi@wsu.edu

Campus of Origin: Pullman, WSU Global Campus

Starting Date: Fall 2016

<u>Method of course delivery: (check all that apply)</u>

- ☐ Classroom (including hybrid)
  - ☐ Pullman
  - ❍ Vancouver
  - ❍ Tri-Cities
  - ❍ Spokane
  - ❍ WSU Research, Learning, or Extension Center(s) at:
  - ☐ Other Location(s) at:
- ☐ AMS or Video-Conferencing System
- ☑ The Global Campus
- ☐ Other (please describe)

II. Mission Statement

Washington State University

Vision
Washington State University will be recognized as one of the nation's leading land-grant research universities.

Mission
Washington State University is a public research university committed to its land-grant heritage and tradition of service to society. Our mission is threefold:

- To advance knowledge through creative research and scholarship across a wide range of academic disciplines.
- To extend knowledge through innovative educational programs in which emerging scholars are mentored to realize their highest potential and assume roles of leadership, responsibility, and service to society.
- To apply knowledge through local and global engagement that will improve the quality of life and enhance the economy of the state, nation, and world.

What is the Mission statement of your Department(s)?

The School of Electrical Engineering and Computer Science (EECS) education, research, and outreach missions are as follows.

Education mission:
- Educate graduates for professional leadership, civic influence, and lifelong learning
- Provide an education based on a theoretical, experimental, and ethical foundation and enhanced by opportunities for participation in research, internships, international studies, interdisciplinary programs, or programs in entrepreneurship

Research mission:
- Conduct research and develop technology to address present and future societal problems
- Advance the state-of-the-art in areas incorporating technical disciplines from electrical engineering and computer science
- Collaborate with researchers from other disciplines to address societal grand challenge problems

Outreach mission:
- Serve the community and the profession by participating in activities designed to improve and preserve the body of knowledge in engineering and computing
- Participate in service that advances engineering and computing education
- Transfer research results to communities, the nation, and the world to increase economic equity, quality of life, and ecological sustainability

Your College(s)?
The Voiland College of Engineering and Architecture (VCEA) mission is to provide a comprehensive education to a diverse constituency in engineering and architecture that prepares students to contribute effectively to the profession and society, for advanced study, and for lifelong learning; to conduct research, integrated with education, in selected areas of excellence, within traditional disciplines and within interdisciplinary teams, technologically important and relevant to the region and nation; and to serve constituents through technology and design transfer partnerships and extended education programs.

Your Campus(s)?

The mission statement for the Pullman campus is already outlined above. The mission statement for the WSU Global Campus is as follows:

The Global Campus is a door that connects the world to WSU and WSU to the world. It provides access to the best of WSU for students, faculty, and anyone seeking to gain or share knowledge. The Global Campus advances WSU's mission to bring education beyond geographic boundaries. And it goes beyond education to create a virtual gathering place that offers a true campus experience.

Describe how this proposed program will complement or reflect these missions.

Questions to ask:

- Where are we? (as a department/college/campus)
- Where do we want to go (or to develop, or to be perceived)?
- How will the proposed program help us get there?

The proposed new online Master of Science degree in Software Engineering (MSSE) is in line with and reflects the university, global campus, college, and school missions in several ways, including:

- Producing highly qualified, much in demand, software engineering professionals.
- Expanding access to high-quality master's degrees in science and engineering in the state, the nation, and globally.
- Producing work-ready graduates with advanced, state-of-the-art education and training.
- Meeting the workforce needs of the state and regional industries.
- Establishing and fostering research in software engineering at WSU.
- Providing support for high priority research and educational initiatives such as data science/analytics.

III. Program Description

Questions to ask:

- What is the nature and focus of this program?
- Is it interdisciplinary in nature? If so, what are the fields of study involved, and how will multiple units work together in delivering the program? *(Document support from all units involved.)*
- Within what discipline(s) does it fall? What distinguishes it from other similar disciplines or from other branches of the same field?
- Is it a broad, general program or will it focus on one specialization? Does it offer more than one option?

The proposed online Master's degree in Software Engineering is intended to be a complementary, sister program to already existing MS Computer Science program at WSU. As sister disciplines, computer science and software engineering share the fundamentals of a computer science curriculum. Where they differ is in advanced, graduate level courses—CS focuses on topics in machine learning, data science, algorithm design, distributed and networked systems, human computer interfacing, pervasive computing, bioinformatics, and other topics of interest to the students. In contract, a MSSE program focuses on advanced courses in software design and development, software testing and validation, software maintenance, software security, and software management and integration—all specialties of high demand among the State's computing and IT industries.

In addition, the proposed online MSSE program will utilize two existing courses from the MS Engineering Management program that are related to professional ethics and project management.

It should be noted that software engineering and software design principles play fundamental and supporting roles in application development in many domains, including those served by computational and data sciences, such as health or environmental informatics, business analytics, and bioinformatics. Therefore, several of the advanced elective courses in software engineering will be of interdisciplinary nature with applications in business, biology, health, and environmental sciences.

The proposed MSSE courses and degree program will be offered entirely online (distance-learning) through the WSU Global Campus. The online, asynchronous delivery mode makes the program equally desirable for working professionals looking for part-time, slower-pace participation as well as full-time graduate students seeking a quicker-pace to completion for a professional MS degree.

The online MSSE degree will be a terminal, non-thesis, degree designed to address the needs of professionals seeking advanced degrees or those seeking employment in industry/commercial sector right after graduation.

The online MSSE program will be offered and managed by a mix of faculty as follows: A new Clinical Software Engineering faculty that has been hired using Engineering Expansion funds, several existing CS faculty, and leveraging the newly proposed BS in Software Engineering program with four new Software Engineering faculty [see the accompanying BSSE proposal to the Faculty Senate] that will be offered in both Pullman and WSU North Puget Sound at Everett campus. *Over time, as the enrollment grows, we will rely on the funds generated by the university revenue sharing policy to sustain and grow the program; i.e., make it a self-funded program.*

We believe due to the acute computing and software workforce shortage in the state, it is imperative that we develop the proposed online MSSE program as soon as possible and simultaneously with the BSSE program. This can be achieved with minimum risk by careful planning and skewing the timelines for the development of the two programs:

- First, it should be noted that the MSSE degree will only require 6 new software engineering courses, 2 of which will be shared with the BSSE program. Combing the new software engineering courses with 2 Engineering Management courses and 2 Computer Science courses will constitute the curriculum for the entire degree. The development of these courses will be spread over 4 semesters and offered as initial students progress through their degree.

- Second, the MSSE degree is slated to start simultaneously with the BSSE degree in Fall 2016. This will allow the faculty time to develop all new required software engineering courses for the start of the program while the elective courses will be developed over time. A development schedule for the courses is provided in Section VI.6 (p. 29).

IV. State Need and Student Demand for the Program

Summarize your conclusions about need and demand from the *Workbook II - Analyzing Demand and Cost* here:

Access to baccalaureate and graduate degrees continues to be a major issue curbing overall educational attainment and nationwide economic competitiveness in Washington. The problem is most acute in the areas of computer science (CS) and software engineering (SE). A 2011 joint report prepared by the Washington State Higher Education Board, State Board for Community and Technical Colleges, and Workforce Training and Education Coordinating Board indicates that during the 2014-2019 period there will be a ratio of nearly 2 to 1 between demand and supply in the State for graduates with BS degrees in CS or SE.

The land grant mission of the University is to extend access to education. Delivering the proposed MS degree online and asynchronously, provides access to qualified place-bound individuals state-wide, nationally, and internationally. Additionally, the demand for individuals with software engineering skills in Washington State and the Pacific Northwest has grown steadily and significantly over time and is expected to continue to grow rapidly in the coming decade.

A recent comprehensive market study by Global Campus shows that software engineering skills are highly sought after in the workplace. Employment in the area of software engineering is growing rapidly with jobs such as Software Development Engineer, Software Developer, and Software Engineer growing about 87%, 56% and 74%, respectively, in Washington alone in 2012-2013. It is expected to continue this trajectory through 2020.

Data from the second half of 2013 show that 4,483 regional jobs posted requiring Software Engineering skills at graduate level and 11,881 regional jobs posted requiring Software Engineering skills at Bachelor's level.

Software engineering graduates are rare and easily demand 6-figure salaries upon graduation. While other states such as California, Massachusetts, Pennsylvania, Illinois, and Indiana have committed to building software engineering programs and producing SE graduates, the State of Washington is lagging in this area. To the best of our knowledge, currently there is no online Master of Science in Software Engineering degree offered at any of the 4-year institutions of higher education in the Sate. University of Washington has a face-to-face, evening professional Master's degree in Computer Science and Engineering and UW-Bothell offers a face-to-face Master of Science program in Computer Science and Software Engineering. Western Governors University offers competency-based online MS degrees in Information Technology, but not in Software Engineering.

Therefore, the proposed new online MSSE program at WSU Global Campus will be unique and will help the State and its computing/IT industries meet their needs by producing highly skilled and trained graduates in software engineering.

Alignment with Strategic Plans:

The proposed online MSSE degree is inline with the University, VCEA, and EECS strategic plans in building strong research and educational programs in the data science or data analytics areas. A widely accepted definition[1] denotes that data science takes place at the intersection of hacking skills (software development), math and statistics knowledge, and domain expertise. Therefore, software engineering plays a central and supporting role in any strong data science program.

In today's data driven societies and economies, scientific and engineering solutions often heavily rely on acquisition, management, and careful analysis of vast datasets. This analysis requires a skill-set as broad as it is deep in that scientists must be experts not only in their own domain, but in statistics, computing, algorithm building, and software design as well. Therefore, software engineering plays a central role as WSU, VCEA, and EECS strive to build strong research and educational programs in data science/analytics. For example, many of the MS degrees in Data Science or Data Analytics at universities such as NYU, Columbia, IIT, and SMU have software engineering as part of their core curriculum.

Data science research requires interdisciplinary collaboration among computer scientists, software engineers, mathematician and statisticians, and domain experts. Since software engineering expertise is currently non-existing, or at a minimal level, in Pullman, the proposed software engineering program will provide an opportunity to fundamentally contribute towards building a strong data science research program on Campus. Furthermore, the proposed program will provide a strong supporting role in building robust, vibrant PhD programs in data science by helping in the production of PhD graduates with strong multi-disciplinary expertise (including software development).

V. Goals and Objectives, Student Learning Outcomes and Assessment

A. Goals and Objectives

Questions to ask:

- What are we trying to achieve with this program?
- How will we assess whether we are meeting our goals and objectives – i.e., how will we gather information and how will we use it?

The goal of the proposed MSSE program is to produce highly qualified software engineering professionals to meet the workforce needs of the state and regional industries.

The objectives of our MSSE program are to allow students to acquire the education and professional skills necessary to:

1. Identify and solve problems relevant to the Software Engineering discipline.
2. Compete and advance in the Software Engineering industry.

[1] http://drewconway.com/zia/2013/3/26/the-data-science-venn-diagram

To assess whether we meet the objectives we will define students learning outcomes and map the outcomes to the program objectives.

B. Student Learning Outcomes

Questions to ask:

- What will our graduates know and be able to do as a result of this program?
- Are these outcomes observable and measurable?
- Do they align with other university learning goals, such as the Seven Goals of the Baccalaureate, and/or with standards from professional or disciplinary organizations?

The MSSE program enables students to attain, by the time of graduation:

(1) [Professional Software Engineering Knowledge] mastery of software engineering knowledge and skills and of the professional standards necessary to practice as a software engineer.
(2) [Technical Software Engineering Knowledge] understanding of and ability to apply appropriate theories, models, and techniques that provide a basis for problem identification and analysis, software design, development, implementation, verification, and documentation.
(3) [Teamwork] an ability to be an effective member of a team, including teams that are geographically distributed, effectively communicate both orally and in writing, and to lead or manage projects.
(4) [Depth] mastery of one or more subdomains in Software Engineering (Capstone Experience).

Table 1 provides a correspondence between the students learning outcomes (rows) and the program objectives (columns) defined in Section V.A (p. 23).

Table 1: Correspondence between students learning outcomes and program objectives.

	1. Identify and solve problems relevant to Software Engineering.	2. Compete and advance in the Software Engineering industry.
(1)	x	x
(2)	x	x
(3)	x	x
(4)	x	x

C. Assessment of Student Learning and Student Achievement (resources and samples appended)

Questions to ask:

- How will we assure whether students are achieving the student learning outcomes?
- Does this program include a capstone class or experience, where students demonstrate mastery of the learning outcomes and assessment can readily occur?
- What resources are available to support assessment?

- If the new program will be included in an existing program's assessment plan, how will the assessment data be disaggregated to provide useful information on the new program? How will the new program participate in assessment? Are there any unique assessment needs or constraints (may include delivery mode, for example)?

The proposed assessment plan is attached as Appendix A (p. 38).

VI. Curriculum

Questions to ask:

- What courses will be required?
- What electives (if applicable) will be available?
- What courses from other departments/colleges will be used? *(Document support from those units.)*
- How do the required courses align with the student learning outcomes to ensure students can build and refine skills and knowledge over time? Has the curriculum been mapped?
- If this is an undergraduate program, how have the needs of transfer students been taken into account in planning for it? What arrangements are in place to ensure that pre-requisites are readily available and that CC students will be appropriately prepared and well advised? If this is an undergraduate program offered at a location other than Pullman, how have local community colleges and their faculties been involved in planning for it?

Please attach a curriculum map (matrix aligning courses and the program's student learning outcomes)

Section VI.1 provides a summary outline for the proposed MSSE degree. Following are a curriculum map aligning the courses and the student learning outcomes (Section VI.2), examples of full-time and part-time study plans (Sections VI.3 and VI.4, respectively), and a summary description of the courses (Section VI.5).

The syllabi for the new courses are provided in Appendix B (p.39) and major change forms are simultaneously being submitted to the Faculty Senate.

1. Summary outline of the degree

The proposed MSSE program requires 6 few Software Engineering courses, 2 of which are shared with the BSSE program and are part of the core courses. The MSSE program reuses 2 existing Computer Science courses that are elective. The MSSE program also uses 2 Engineering Management courses as part of the advanced courses. The proposed MSSE program requires 30 credits; following is an outline of the degree:

Core courses (9 credits)

Cpt S 515*: Advanced algorithms*
Cpt S 484: *Software Requirements*
Cpt S 487: *Software Design and Architecture*

Advanced courses (15 credits)

Cpt S 581: *Software Maintenance*
Cpt S 582: *Software Testing*
Cpt S 583: *Software Quality*
E M 522: *Leadership, Supervision and Management*
E M 564: *Project Management*

Elective courses (6 credits)
Cpt S 580: *Advanced Databases*
Cpt S 564: *Distributed Systems Concepts and Programming*
Or any other 5xx level course in Software Engineering, Computer Science, Computer Engineering, or Math. The courses that are not offered online through the proposed degree can be taken locally and transferred into the program upon approval by the Graduate Studies Committee.

2. Curriculum map aligning courses and the student learning outcomes:

Table 2: Map aligning courses and the student learning outcomes.

Microsoft Excel Worksheet

3. Example Full-time Study Plan (30 credit hours)

First Year

First semester

Cpt S 484: *Software Requirements*	3
Cpt S 515*: Advanced algorithms*	3
E M 522: *Leadership, Supervision and Management*	3
*Software Engineering Option Course	3
	Total credit hours: 12

Second semester

*Software Engineering Option Course	3
Cpt S 487: *Software Design and Architecture*	3
Cpt S 581: *Software Maintenance*	3
E M 564: *Project Management*	3
	Total credit hours: 12

Second Year

First semester

Cpt S 583: *Software Quality*	3
Cpt S 582: *Software Testing*	3
**Capstone Experience	0
	Total credit hours: 6

*Software Engineering Option Course can be any 5xx level course in Software Engineering, Computer Science, Computer Engineering, or Math. The courses that are not offered online through the proposed degree can be taken locally and transferred into the program upon approval by the Graduate Studies Committee. The courses listed in Section VI.4.a (p. 28) will be made available online as initial list of Software Engineering Option Courses.

**Students are required to satisfy the Capstone Requirement stated in Section V.C (p. 24).

4. Example Part-time Study Plan (30 credit hours)

First Year

First semester
Cpt S 484: *Software Requirements* 3
Cpt S 515*: Advanced algorithms* 3

Total credit hours: 6

Second semester
Cpt S 487: *Software Design and Architecture* 3
E M 564: *Project Management* 3
Total credit hours: 6

Second Year

First semester
E M 522: *Leadership, Supervision and Management* 3
*Software Engineering Option Course 3

Total credit hours: 6

Second semester
Cpt S 581: *Software Maintenance* 3
*Software Engineering Option Course 3
Total credit hours: 6

Third Year

First semester
Cpt S 583: *Software Quality* 3
Cpt S 582: *Software Testing* 3
**Capstone Experience 0
Total credit hours: 6

*Software Engineering Option Course can be any 5xx level course in Software Engineering, Computer Science, Computer Engineering, or Math. The courses that are not offered online through the proposed degree can be taken locally and transferred into the program upon approval by the Graduate Studies Committee. The courses listed in Section VI.4.a (p. 28) will be made available online as initial list of Software Engineering Option Courses.

**Students are required to satisfy the Capstone Requirement stated in Section V.C (p. 24).

5. Information on Individual Courses

a. Existing Computer Science and Software Engineering courses (Cpt S):

Cpt S 564: *Distributed Systems Concepts and Programming* 4
Professor/Coordinator: Dave Bakken. Elective course for the MSSE program. Concepts of distributed systems; naming, security, networking, replication, synchronization, quality of service; programming middleware. Credit not granted for both CPT S 464 and CPT S 564. Offered at 400 and 500 level. Cooperative: Open to UI degree-seeking students.

Cpt S 580: *Advanced Databases* 4
Professor/Coordinator: Yinghui Wu. Elective course for the MSSE program. Advanced study of non-relational data models, the internals of a database management system, and database frontiers.

b. Existing Engineering and Technology Management courses (E M):

E M 522: *Leadership, Supervision and Management* 3
Professor/Coordinator: John Pricco. Required course for the MSSE program. Prerequisite: None. Strategies of supervision with practical application techniques presented to create individual and organizational motivation. Credit not granted for both E M 422 and 522. Offered at 400 and 500 level.

E M 564: *Project Management* 3
Professor/Coordinator: Bill Gray. Required course for the MSSE program. Prerequisite: None. Technical tools, Critical Path Method (CPM), Program Evaluation Review Technique (PERT), cost/schedule control systems, behavioral issues and organizational structure. Credit not granted for both E M 464 and E M 564. Offered at 400 and 500 level.

c. New Software Engineering courses (Cpt S):

Cpt S 484: *Software Requirements* 3
Professors/Coordinators: Venera Arnaoudova, Bolong Zeng, and Evan Olds. Required course for the MSSE program. Course Prerequisite: CPT S 322 with a C or better. Elicitation, analysis, specification, and validation of software requirements as well as the management of requirements during the software life cycle.

Cpt S 487: *Software Design and Architecture* 3
Professors/Coordinators: Venera Arnaoudova, Bolong Zeng, and Evan Olds. Required course for the MSSE program. Course Prerequisite: CPT S 322 with a C or better; Cpt S 321 with a C or better. Software design; design principles, patterns, and anti-patterns; design quality attributes and evaluation; architectural styles, architectural patterns and anti-patterns.

Cpt S 515*: Advanced algorithms* 3
Professor/Coordinator: Zhe Dang. Required course for the MSSE program. Advanced algorithms, randomized algorithms, Hashing, approximation algorithms.

Cpt S 581: *Software Maintenance* 3

Professors/Coordinators: Venera Arnaoudova, Bolong Zeng, and Evan Olds. Required course for the MSSE program. Course Prerequisite: Graduate standing. Software maintenance, refactoring, reengineering, reverse engineering.

Cpt S 582: *Software Testing* 3
Professors/Coordinators: Venera Arnaoudova, Bolong Zeng, and Evan Olds. Required course for the MSSE program. Course Prerequisite: Graduate standing. Software testing, testing levels, testing objectives, testing techniques, test related measures, testing tools.

Cpt S 583: *Software Quality* 3
Professors/Coordinators: Venera Arnaoudova, Bolong Zeng, and Evan Olds. Required course for the MSSE program. Course Prerequisite: Graduate standing. Software quality, quality assurance, process and product quality, software measures, quality attributes, quality management.

6. Course development schedule

Two courses will be developed every semester starting from Fall 2015. The proposed development schedule is as follows:

Fall 2015
- Cpt S 484: *Software Requirements*
- Cpt S 515*: Advanced algorithms*

Spring 2016
- Cpt S 582: *Software Testing*
- Cpt S 564: *Distributed Systems Concepts and Programming*

Summer 2015
- Cpt S 583: *Software Quality*
- Cpt S 487: *Software Design and Architecture*

Fall 2016
- Cpt S 581: *Software Maintenance*
- Cpt S 580: *Advanced Databases*

7. Course rotation plan

Four courses will be offered in the Fall and Spring semesters. A tentative plan for offering the MSSE courses follows:

Fall semester:
- Cpt S 484: *Software Requirements* or Cpt S 583: *Software Quality* (i.e., the courses will be offered on a two-year rotation starting with Cpt S 484 in Fall 2016)
- Cpt S 582: *Software Testing*
- Cpt S 515*: Advanced algorithms*
- *Software Engineering Option Course (Cpt S 564: *Distributed Systems Concepts and Programming* in Fall 2016)

Spring semester:
Cpt S 487: *Software Design and Architecture*
Cpt S 581: *Software Maintenance*
*Software Engineering Option Course (Cpt S 580: *Advanced Databases* in Spring 2017)

The courses from the Engineering Management program are offered in the Fall. E M 564 is also offered in the Spring semester.

VII. Uses of Technology

Questions to ask:

- What kinds of technology will be used in teaching this curriculum?
- Will instructors or students need any training or support using technology? If so, how will the training or support be provided?
- What technologies will the students learn to use in order to be employed in this field? To what extent do the class technologies align with technologies in the field?

The proposed MSSE program will primarily rely on computer systems and software tools as technologies for offering the curriculum. The participating distance Software Engineering students will be required to have their own laptops or computer systems for their various programming and project assignment needs. Therefore, no new physical computing facilities will be required.

Many of the software tools needed for the MSSE is already in place and in use for the Computer Science program. Additional tools will be acquired as needed.

The Computer Science and Software Engineering faculty are already experts in the use of computing technologies and software tools. The existing CS and the proposed SE curricula will prepare the students for the efficient use of these technologies and tools.

CS and SE students will use industry standard systems and tools in their course work, including but not limited to: Linux, Windows, iOS, and Android operating systems; programming tools such as Visual Studio; software revision control tools such as Git/Github; databases and search engines such as SQL and Solr; and, web and network development tools such as Apache and Microsoft .Net Framework.

VIII. Delivery methods

Questions to ask:

- Will this be an entirely site-based, face-to-face program, or will part or all of it be delivered off-campus and/or electronically?
- If the latter, what parts and by what media?
- If site-based and face-to-face, when will the program be offered (day/evening/weekend)?
- Will students or instructors need any training or support in using the delivery methods? If so, how will that training or support be provided?

The proposed MSSE courses and degree program will be offered entirely online (distance-learning) through the WSU Global Campus. The online, asynchronous delivery mode makes the program equally desirable for working professionals looking for part-time, slower-pace participation as well as full-time graduate students seeking a quicker-pace to completion of a professional MS degree.

IX. Students

A. How many students to you expect to serve with this program?
(If you expect a combination of part time and full time students, please use the FTE Calculator, at **Table 2** of http://www.budget.wsu.edu/Cost_template.xls , before completing this table.)

Table 3: Expected program growth for the MSSE online program.

Number of Students	**Year 1**	**Year 2**	**Year 3**	**Year _4_***
Headcount	20	30	40	50
FTE**	7	11	15	19

* Enter year number in which program anticipates reaching full enrollment

B. Admission Requirements

Questions to ask:

- What are the certification requirements into this major (for undergraduates), or the departmental process and admission requirements (for graduate programs)?

The admission requirements for the online MSSE degree will be similar to the existing MS Computer Science program, as follows:

The School of EECS evaluates applicants for admission to its graduate programs based on college transcripts, GPA, the score on the general GRE, (3) letters of recommendation, a statement of purpose, and TOEFL score, if applicable for international students.

Students whose undergraduate studies did not include material equivalent to that covered in the following WSU courses will be asked to take course work to resolve their undergraduate deficiencies: CptS 121, 122, 223, 260, 317, 355, 360, 350, Phil 201, and Math 216. All or most of these courses should be completed before the student is eligible for admission into the MS program in Software Engineering. The admissions committee may require the student to correct other undergraduate deficiencies as well, including undergraduate prerequisite courses to graduate courses.

In addition to the MS Computer Science program requirement, students are expected to have passed an equivalent of an introductory course in Software Engineering (Cpt S 322) and possess practical experience in software engineering or software development.

C. Expected time for Program Completion

Questions to ask:

- Will most students be full time or part time?
- How long will it take each type of student?
- If this is an undergraduate program, can it be completed in four years (if so, please outline a 4-year course of study; if not, please explain), and
- How can transfer students articulate smoothly into the program and complete it with approximately the same number of total credits as students who enter WSU as freshmen?*

The online MSSE degree is designed to be completed in 1.5 to 2 years with a full-time course workload. However, the online, asynchronous delivery mode makes the program equally desirable for working professionals looking for part-time, slower-pace participation as well as full-time graduate students seeking a quicker-pace to completion of a professional MS degree.

D. Advising

Questions to ask:

- Who will provide academic advising for the students?
- How will advisors be assigned?

The existing School of EECS Graduate Programs Coordinator will provide academic advising for the online MSSE students.

Each participating student will choose a faculty mentor and supervisor from among the computer science and software engineering faculty for guidance throughout their graduate studies.

E. Diversity

- Please describe specific efforts planned to recruit and retain students who are persons of color, disabled, or whose gender is underrepresented in this discipline.

Women are underrepresented in Computer Science, Software Engineering, and some engineering disciplines such as Computer and Electrical Engineering. Therefore, the School of EECS is developing several initiatives to increase participation of women in its programs, including the proposed MSSE program. These initiatives include:

- Development of a marketing and promotional video featuring one of our female computer science athletes
- Supporting student clubs that are organized and run by women students with the goal of recruiting more women students to CS, CE, SE, and EE disciplines
- Becoming a member of and participating in programs offered by National Center for Women & Information Technology (NCWIT) Academic Alliance

- Partnering with American Association of University Women (AAUW) to set up a summer Tech Trek Camp for 150 high school girls to come to WSU each year and participate in a week-long math and science camp working with experienced professors on creative and engaging hands-on programming projects

X. Faculty and Administration

The online MSSE program will be offered and managed by a mix of faculty as follows: A new Clinial Software Engineering faculty, several existing CS faculty, and leveraging the newly proposed BS in Software Engineering program (BSSE) that will be offered in Pullman and at WSU North Puget Sound at Everett campus [see the attached BSSE proposal].

We are in the process of hiring a Software Engineering faculty with complementary expertise in Data science. This position will be supported by the existing Engineering Expansion funds. In 2014 we requested new state funding for establishing the BSSE degree.

The BSSE program will include the development of several dual-purpose, conjoint advanced electives that are simultaneously used in both the MSSE and BSSE programs. The development of these courses does not represent additional costs/faculty time for the MSSE since those factors (time/cost) are already accounted for in the BSSE program. No new staff support is required, as the existing EECS Graduate Programs Coordinator will manage the MSSE program as well.

In summary, the new faculty and administration resources needed for the MSSE online program consists of one faculty as shown below:

Table 4: Program Faculty.

Name or Position Identifier*	Rank/Title**	Status***	FTE****	% Effort in Program	FTE in Program
Venera Arnaoudova	Assistant Professor	Tenure track	1	100.0%	1.00
To be recruited	Assoc/Full	Tenure track or clinical	1	100.0%	1.00
Evan Olds	Assistant Professor	Clinical Teaching	1	100.0%	1.00
Bolong Zeng	Assistant Professor	Clinical Teachnig	1	100.0%	1.00
To be recruited	Assistant/Assoc	Clinical	1	100.0%	1.00
Total Faculty FTE					5.00

*E.g., Jones, current faculty, new hire, etc
**E.g., professor, associate professor, assistant professor, instructor, teaching assistant
***Tenure-track, clinical, adjunct, etc
****E.g., full-time appointment is 1.0 FTE, half-time is 0.5 FTE, etc.

XI. Facilities

Questions to ask:

- Will this program require new teaching laboratories?
- Will this program require new research facilities?
- Will this program require specialized equipment?
- If so, what resources are available for this purpose?

The proposed MSSE program will primarily rely on computer systems and software tools as technologies for offering the curriculum. The participating distance Software Engineering students will be required to have their own laptops or computer systems for their various programming and project assignment needs. Therefore, no new physical computing facilities will be required.

Many of the software tools needed for the MSSE is already in place and in use for the Computer Science program. Additional tools will be acquired as needed.

The tenure-track software engineering faculty hires (hired through the BSSE program) will be able to acquire their research computing needs through their start-up packages.

WSU Online (Global Campus) provides support to faculty in the development and delivery of the online course:

- An eLearning Consultant, with expertise in instructional design of online courses will work 1:1 with faculty members developing online courses to ensure that best practices and pedagogical recommendations for successful online learning are understood.
- The WSU Online media team will work with faculty to create appropriate media and interactive activities to promote learning and enhance engagement.
- The same eLearning Consultant will continue to support the faculty member during delivery as issues unique to the online learning environment arise.
- WSU Online provides face to face orientation and trainings and online tutorials to support online instructors.
- Managing proctored exams for the course, if needed.
- 24/7 technical support.
- Ongoing maintenance or updating of courses, each semester of offering is provided by WSU Online.

WSU Online provides support to students:

- Acquiring required resources, such as texts and media
- Arranging for proctored exams.
- Academic Consultants provide advising for WSU Online degree seeking students.
- 24/7 technical support

XII. Finances

As explained in Section X (p. 33), the only new resources needed for the proposed program is the addition of a new Software Engineering faculty. *Over time, as the enrollment grows, we will rely on the funds generated by the university revenue sharing policy to sustain and grow the program; i.e., make it a self-funded program*. However, since the program leverages the proposed BSSE degree at Everett, the offering of the degree may be delayed if the Everett program is not funded.

Table 5: Summary of Program Costs.

This template will calculate the direct, indirect and total cost as well as the cost per student FTE.						
Enter the name of the Degree program here	Date	Internal Reallocation	New State Funds	Other Sources	Year 1 Total	Year N Total
Administrative Salaries, including benefits		-		-	-	-
Faculty Salaries, including benefits	7/1/15	200,000		-	200,000	200,000
TA/RA Salaries including benefits		-		-	-	-
Clerical Salaries, including benefits		-	-	-	-	-
Other Salaries including benefits (Adjunct Faculty)		-		-		
Contract Services		-	-	-	-	-
Goods and Services		-	-	-	-	-
Travel		-	-	-	-	-
Equipment		-	-	-	-	-
Other costs-Faculty Start-ups	7/1/15	250,000	-	-	250,000	-
Library		-	-	-	-	-
Direct Cost		**450,000**	**-**	**-**	**450,000**	**200,000**
Indirect Cost		**264,286**	**-**	**-**	**264,286**	**117,460**
Total Cost		**714,286**	**-**	**-**	**714,286**	**317,460**
FTE Students					12	30
Cost Per FTE					**59,524**	**10,582**

XIII. External Reviews

If this program is new to the Washington State University system, please provide the names and addresses of 3 – 4 external experts from similar institutions who could be contacted to provide reviews of this program.

1. Ali R. Hurson
Department of Computer Science
Missouri University of Science and Technology
hurson@mst.edu
573-341-6201

2. H.J. Siegel
Department of Electrical and Computer Engineering
Colorado State University
H.J.Siegel@colostate.edu
970-491-7982

3. Lynn Peterson
Department of Computer Science and Engineering
University of Texas at Arlington
peterson@uta.edu
817-272-5503

4. Samee Khan
Department of Electrical and Computer Engineering
North Dakota State University
samee.khan@ndsu.edu
701-231-7615

APPENDICES

A. Assessment Plan for the Student Learning Outcomes

Table 6 provides details on the plan for assessing the student learning outcomes. We will collect student work samples on a two-year rotation. In addition, we will evaluate outcome 9 annually through a Capstone Experience.

Capstone Requirement: The requirement for the Capstone Experience are that students must pass the following courses with a grade of B or higher in each:

- Cpt S 581: Software Maintenance
- Cpt S 583: Software Quality
- Cpt S 582: Software Testing

The Examination Committee will evaluate the grades and ballot on the results of the grades in the Capstone courses.

Table 6: Assessment process for the students learning outcomes.

Learning outcome	Source for outcome assessment in odd years (e.g., 2013-2014)	Source for outcome assessment in even years (e.g., 2014-2015)
(1)		Cpt S 581, Cpt S 582
(2)		Cpt S 581, Cpt S 582
(3)	Cpt S 484, Cpt S 583	
(4)	Cpt S 484, Cpt S 583	

Data analysis:

The program director is responsible for collecting the following information annually:

1. Course evaluations
2. Course grade distributions
3. Averages on specific exam/course work assignment or project questions tailored to student learning outcomes.

The MSSE program director will provide an initial evaluation of the data outlined above to determine whether the expected learning outcomes have been met. He/she will then organize an annual meeting of the Software Engineering faculty and some of the Computer Science faculty to discuss the results. The data will be summarized and potential specific improvements will be suggested in a written form and will be recorded in an archival document. The effect of the changes will be discussed and summarized in the report of the following year.

B. Syllabi for the new Software Engineering Courses

Major change forms for the newly proposed courses are simultaneously being submitted to the Faculty Senate.

Software Requirements

Course Name: Software Requirements
Course Number: Cpt S 484
Credits: 3
Lecture Hours: 3
Schedule: Offered online (asynchronously) via Global Campus
Prerequisites: Cpt S 322
Course required/elective: required.
Professors/Coordinators: Venera Arnaoudova, Bolong Zeng, and Evan Olds.

Textbook(s):
[1] I. Sommerville, Software Engineering, 9th ed., Addison-Wesley, 2011.[2]
[2] K.E. Wiegers and J. Beatty, Software Requirements, 3rd ed., Microsoft Press, 2013.[3]
[3] J.M. Wing, A Specifier's Introduction to Formal Methods, Computer, vol. 23, no. 9, 1990, pp. 8, 10–23.[4]

Course description: Elicitation, analysis, specification, and validation of software requirements as well as the management of requirements during the software life cycle.

Overview and Course Goals: This course teaches the fundamentals of Software Requirements. Students will learn how to elicit and analyze stakeholders' needs to document the desired system behavior along with relevant constraints and assumptions. They will also learn how to analyze requirements for potential conflicts and validate that the final software product satisfies the requirements.

Course topics and the corresponding program learning outcomes[5]:
- Software Requirements Fundamentals [a,c,e,f,g,h,i,k,l,m,o,p] [1,2,4]
- Requirements Process [a,e,f,g,k,l,m] [1,2,4]
- Requirements Elicitation [a,e,f,g,k,l,m,q] [1,2,3,4]
- Requirements Analysis [a,c,e,k,l,m,o,p,q] [1,2,3,4]
- Requirements Specification [a,c,e,k,l,m,o,p,q] [1,2,3,4]
- Formal Methods [a,c,e,k,l,m,o,p,q] [1,2,3,4]
- Requirements Validation [a,c,e,f,g,h,k,l,m,o,p,q] [1,2,3,4]
- Practical considerations such as requirement management [a,c,e,f,g,h,i,k,l,m,o,p,q] [1,2,3,4]
- Software Requirements Tools [a,c,e,i,k,l,m,o,p,q] [1,2,3,4]

Learning outcomes and evaluation:

At the end of the course students will be able to:

[2] Available on Amazon: ISBN-13: 978-0137035151.
[3] Available on Amazon: ISBN-13: 978-0735679665.
[4] Available on the IEEE Xplore Digital Library; access provided by WSU Libraries.
[5] The student learning outcomes for the BSSE program are labeled from 'a' to 'q'. The student learning outcomes for the MSSE program are labeled from '1' to '4'.

1. Explain the role of requirements engineering and its process.
2. Determine stakeholder requirements using multiple standard techniques.
3. Produce a specification with functional and non-functional requirements based on the elicited requirements.
4. Decide scope and priorities by negotiating with the client and other stakeholders.
5. Manage requirements.
6. Apply standard quality assurance techniques to ensure that requirements are: verifiable, traceable, measurable, testable, accurate, unambiguous, consistent, and complete.
7. Produce test cases, plans, and procedures that can be used to verify that they have defined, designed and implemented a system that meets the needs of the intended users.

Mapping student learning outcomes, course topics, and evaluations:

Student Learning Outcomes	Course topics/dates	Evaluation of Outcome
1	Fundamentals of software requirements and requirements process (weeks 1 and 2)	Mid-term 1
2	Requirements elicitation (week 3)	Project, mid-term 1
3	Requirements specification (weeks 8 and 9)	Project
4	Analysis of requirements (weeks 4, 5, 6, and 7)	Project, mid-term 2
5	Requirements management and traceability (weeks 12 and 14)	Project
6	Requirements validation (week 10, 11, 14 and 15)	Final exam
7	Acceptance testing (week 11)	Final exam, project

Week-by-week schedule:

Week	Topics	Evaluation
1	Introduction to Software Requirements Engineering. Definitions and fundamentals.	
2	Requirements process: models, actors, management, and improvement.	
3	Requirements elicitation: sources and	

	techniques.	
4	Requirements analysis: classification, modelling, and allocation.	Project deliverable 1
5	Tools for modeling requirements.	Mid-term 1
6	Introduction to formal methods.	
7	Requirements formal analysis.	
8	System Definition Document and System Requirements Specification.	
9	Software Requirement Specification.	
10	Validating requirements: reviews and prototyping.	Mid-term 2
11	Validating requirements: model validation and acceptance testing.	Project deliverable 2
12	Managing changing requirements. Tools.	
13	Requirements attributes.	
14	Tracing requirements.	
15	Measuring requirements.	
16		Final exam, project deliverable 3

Grading framework: Course grades are based on 3 exams (two mid-terms and one final) totaling 50% of the final grade and a project totaling 50% of the final grade.

The project takes students through the main activities of software requirements engineering. In the first deliverable, students will be asked to apply techniques for requirements elicitation and to document the client needs. For the second deliverable students will define the system and will produce a software requirement specification document. They will analyze the requirements and will design appropriate acceptance tests. For the third deliverable students will design and implement the system and will produce a traceability matrix for all requirements.

Final grades will be awarded on the following scale:

Interval	Grade
[90,100]	A
[87,90)	A-
[83,87)	B+
[80,83)	B
[77,80)	B-
[73,77)	C+

[70,73)	C
[67,70)	C-
[63,67)	D+
[60,63)	D
[0,60)	F

Course rules:

You must take exam during the assigned test period. Failure to do so will result in a score of zero. However, in extraordinary circumstances and at the discretion of the instructor, a make-up exam may be offered. An advanced notice must be given to the instructor beforehand.

Unless posted otherwise, assignment documents shall be submitted electronically.

Late penalty is a flat 10% deduction per day. Late assignments may be turned up to one week after the original due date, and an advanced notice must be given to the instructor beforehand for the late submission. No homework will be accepted after its due day without advanced notice or special permission from the instructor.

Bonus points will be added to your total class score for attendance as follows: 0 absence = 5% of the final grade, 1 absence = 4 %, 2 absences = 3% , and 3 or more absences = 0% bonus.

Reasonable Accommodation:

Reasonable accommodations are available for students with a documented disability. If you have a disability and need accommodations to fully participate in this class, please either visit or call the Access Center (Washington Building 217; 509-335-3417) to schedule an appointment with an Access Advisor. All accommodations MUST be approved through the Access Center.

Academic Integrity:

I encourage you to work with classmates on assignments. However, each student must turn in original work. No copying will be accepted. Students who violate WSU's Standards of Conduct for Students will receive an F as a final grade in this course, will not have the option to withdraw from the course and will be reported to the Office Student Conduct. Cheating is defined in the Standards for Student Conduct WAC 504-26-010 (3). It is strongly suggested that you read and understand these definitions. (Read more: http://apps.leg.wa.gov/wac/default.aspx?cite=504-26-010)

Safety:

Washington State University is committed to maintaining a safe environment for its faculty, staff, and students. Safety is the responsibility of every member of the campus community and individuals should know the appropriate actions to take when an emergency arises. In support of our commitment to the safety of the campus community the University has developed a Campus Safety Plan, http://safetyplan.wsu.edu. It is highly recommended that you visit this web site as well as the University emergency management web site at http://oem.wsu.edu/ to become familiar with the information provided.

Software Design and Architecture

Course Name: Software Design and Architecture
Course Number: Cpt S 487
Credits: 3
Lecture Hours: 3
Schedule: Offered online (asynchronously) via Global Campus
Prerequisites: Cpt S 322, Cpt S 321
Course required/elective: required.
Professors/Coordinators: Venera Arnaoudova, Bolong Zeng, and Evan Olds.

Textbook(s):
[1] D. Budgen, Software Design, 2nd ed., Addison-Wesley, 2003.[6]
[2] M. Page-Jones, Fundamentals of Object-Oriented Design in UML, 1st ed., Addison- Wesley, 1999.[7]
[3] W. J. Brown, R. C. Malveau, H. W. M. III, and T. J. Mowbray. AntiPatterns: Refactoring Software, Architectures, and Projects in Crisis. John Wiley & Sons, Inc., 1998.[8]
[4] I. Sommerville, Software Engineering, 9th ed., Addison-Wesley, 2011.[9]
[5] P. Clements et al., Documenting Software Architectures: Views and Beyond, 2nd ed., Pearson Education, 2010.[10]

Course description: Software design; design principles, patterns, and anti-patterns; design quality attributes and evaluation; architectural styles, architectural patterns and anti-patterns.

Overview and Course Goals:
This course teaches students about fundamental software design principles and design methodologies. Students will learn how to build and document the static and dynamic aspects of a software design. Students will also learn when and how to apply common solutions to recurring design problems and how to recognize common poor design solutions as well as how to improve them. The course will also teach techniques to evaluate software quality. The course will also introduce students to different architectural styles. The course will also view good and poor architectural practices. Students will learn about frameworks allowing dynamic components loading, Service-Oriented Architecture (SOA), and distributed systems architectures.

Course topics and the corresponding program learning outcomes[11]:
- Design notations: structural and behavioral descriptions [a,c,e,g,i,j,k,l,m,o,p,q] [1,3,4]
- Software design principles [a,c,e,g,i,j,k,l,m,o,p,q] [1,2,3,4]
- Issues during design: concurrency, events handling, data persistence, etc. [a,c,e,i,j,k,m,o,q] [1,2,3,4]
- Design patterns and anti-patterns [a,c,e,g,i,j,k,m,o,q] [1,2,3,4]
- Quality of a software design [a,c,e,g,i,j,k,l,m,o,p,q] [1,2,3,4]

[6] Available on Amazon: ISBN-13: 978-0201722192.
[7] Available on Amazon: ISBN-13: 078-5342699463.
[8] Available on Amazon: ISBN-13: 978-0471197133.
[9] Available on Amazon: ISBN-13: 978-0137035151.
[10] Available on Amazon: ISBN-13: 978-0321552686.
[11] The student learning outcomes for the BSSE program are labeled from ‘a’ to ‘q’. The student learning outcomes for the MSSE program are labeled from ‘1’ to ‘4’.

- Design strategies and methods: function-oriented, object-oriented, aspect-oriented, etc. [a,c,e,g,i,j,k,l,m,o,p,q] [1,2,3,4]
- Architectural Styles [a,c,e,g,h,k,l,m,o,p,q] [1,2,3,4]
- Architecture design decisions and architectural tradeoffs [a,c,e,f,g,h,i,k,l,m,o,p,q] [1,2,3,4]
- Architectural patterns and anti-patterns [a,c,e,f,i,k,l,m,o,p,q] [1,2,3,4]
- Frameworks [a,c,e,f,i,k,l,m,o,p,q] [1,2,3,4]
- Service-Oriented Architecture (SOA) [a,c,e,f,i,k,l,m,o,p,q] [1,2,3,4]
- Distributed systems architectures [a,c,e,f,i,k,l,m,o,p,q] [1,2,3,4]

Learning outcomes and evaluation:

At the end of the course students will be able to:

1. Explain key concepts in software design and construct professional design documents.
2. Explain and apply software design principles.
3. Design a software system to account for key issues such as concurrency, security, and data persistence.
4. Identify opportunities to apply common design patterns and identify poor design decisions and propose alternative solutions.
5. Critique a proposed software design in terms of quality attributes; select and apply techniques to evaluate the quality of a software design.
6. Describe the main software architectural styles and select the appropriate style for a given software system.
7. Identify reusable components for the system to be developed.
8. Describe common architectural patterns and apply them when appropriate.
9. Produce architectural diagrams representing the various views of the system.

Mapping student learning outcomes, course topics, and evaluations:

Student Learning Outcomes	Course topics/dates	Evaluation of Outcome
1	Software design concepts and documentation (weeks 1, 3, and 4)	Mid-term 1; project
2	Software design principles (week 1)	Mid-term 1; project
3	Key issues during design (week 2)	Project
4	Design patterns and anti-patterns (weeks 5 and 6)	Mid-term 1; project
5	Software design quality (week 7)	Mid-term 2
6	Software architectural styles (weeks 8 and 9)	Mid-term 2; project
7	Software reuse; libraries, frameworks and components (8, 10, and 11)	Mid-term 2; project

8	Architectural patterns and anti-patterns (weeks 12 and 13)	Final exam
9	Documenting software architecture (weeks 8, 9, and 15)	Project, final exam

Week-by-week schedule:

Week	Topics	Evaluation
1	Software design fundamentals and principles (abstraction, information hiding, separation of concerns, etc.).	
2	Key issues during design: concurrency, events handling, data persistence, etc.	
3	Software design strategies and methods (e.g., top-down, bottom-up, function, data structure).	Project deliverable 1
4	Software design notations: structural and behavioral descriptions.	
5	Object-oriented design.	
6	Design patterns and anti-patterns.	Mid-term 1
7	Software design quality analysis and evaluation: quality attributes, measures, techniques, and tools.	
8	Introduction to Software architecture. Modeling architecture with UML.	Project deliverable 2
9	Architectural views. Architectural styles.	
10	Frameworks, e.g., OSGi and dynamic libraries. Technical and legal problems.	
11	Distributed systems architectures. Service-Oriented Architecture (SOA).	
12	Architectural patterns.	Mid-term 2
13	System level anti-patterns (e.g., Architecture by Implication, Analysis Paralysis).	
14	Tracing requirements in architecture.	
15	Documenting software architecture in practice.	Project deliverable 3

16		Final Exam

Grading framework: Course grades are based on 3 exams (two mid-terms and one final) totaling 50% of the final grade and a set of project deliverables totaling 50% of the final grade.

The project consists of delivering a software system with a particular stress on the design and architectural decisions and overall on the design artifacts. Students will be given the requirements and specification document for the system to be developed. For the first project deliverable, students will be asked to implement the system using an object-oriented programming language. In a second deliverable students will be asked to 1) refactor their system to improve the initial design, 2) to document the system after refactoring using structural and behavioral views as appropriate, and 3) to discuss the quality of the design overall and in comparison with the initial design. For the third deliverable students will change the architecture of the project and implement it as part of a framework.

Final grades will be awarded on the following scale:

Interval	Grade
[90,100]	A
[87,90)	A-
[83,87)	B+
[80,83)	B
[77,80)	B-
[73,77)	C+
[70,73)	C
[67,70)	C-
[63,67)	D+
[60,63)	D
[0,60)	F

Course rules:

You must take exam during the assigned test period. Failure to do so will result in a score of zero. However, in extraordinary circumstances and at the discretion of the instructor, a make-up exam may be offered. An advanced notice must be given to the instructor beforehand.

Unless posted otherwise, assignment documents shall be submitted electronically.

Late penalty is a flat 10% deduction per day. Late assignments may be turned up to one week after the original due date, and an advanced notice must be given to the instructor beforehand for the late submission. No homework will be accepted after its due day without advanced notice or special permission from the instructor.

Bonus points will be added to your total class score for attendance as follows: 0 absence = 5% of the final grade, 1 absence = 4 %, 2 absences = 3%, and 3 or more absences = 0% bonus.

Reasonable Accommodation:

Reasonable accommodations are available for students with a documented disability. If you have a disability and need accommodations to fully participate in this class, please either visit or call the Access Center (Washington Building 217; 509-335-3417) to schedule an appointment with an Access Advisor. All accommodations MUST be approved through the Access Center.

Academic Integrity:

I encourage you to work with classmates on assignments. However, each student must turn in original work. No copying will be accepted. Students who violate WSU's Standards of Conduct for Students will receive an F as a final grade in this course, will not have the option to withdraw from the course and will be reported to the Office Student Conduct. Cheating is defined in the Standards for Student Conduct WAC 504-26-010 (3). It is strongly suggested that you read and understand these definitions. (Read more: http://apps.leg.wa.gov/wac/default.aspx?cite=504-26-010)

Safety:

Washington State University is committed to maintaining a safe environment for its faculty, staff, and students. Safety is the responsibility of every member of the campus community and individuals should know the appropriate actions to take when an emergency arises. In support of our commitment to the safety of the campus community the University has developed a Campus Safety Plan, http://safetyplan.wsu.edu. It is highly recommended that you visit this web site as well as the University emergency management web site at http://oem.wsu.edu/ to become familiar with the information provided.

Software Maintenance

Course Name: Software Maintenance
Course Number: Cpt S 581
Credits: 3
Lecture Hours: 3
Schedule: Offered online (asynchronously) via Global Campus
Prerequisites: Graduate standing.
Course required/elective: required.

Professors/Coordinators: Venera Arnaoudova, Bolong Zeng, and Evan Olds.
Office: EME 127
Phone: (509) 335-1360
Email: varnaoud@eecs.wsu.edu

Textbook(s):
[1] P. Grubb and A.A. Takang, Software Maintenance: Concepts and Practice, 2nd ed., World Scientific Publishing, 2003.[12]
[2] IEEE Std. 14764-2006 (a.k.a. ISO/IEC 14764:2006) Standard for Software Engineering—Software Life Cycle Processes—Maintenance, IEEE, 2006.[13]

Additional journal/conference articles:

[3] Keith Brian Gallagher and James R. Lyle. 1991. Using Program Slicing in Software Maintenance. IEEE Transactions on Software Engineering, Vol. 17, Issue 8 (August 1991), 751–761.
[4] H.M. Sneed, "Offering Software Maintenance as an Offshore Service," Proc. IEEE International Conference on Software Maintenance (ICSM), IEEE, 2008, pp. 1–5.

Course description: Software maintenance, refactoring, reengineering, reverse engineering.

Overview and Course Goals: This course teaches students how to maintain a high quality software. Students will learn the fundamentals and key issues during software maintenance and evolution and will learn about frequently used activities, tools, and techniques.

Course topics and the corresponding program learning outcomes[14]:
- Fundamentals of Software maintenance and evolution [1,2,3,4]
- Regression testing [1,2,3,4]
- Program comprehension [1,2,3,4]
- Reengineering [1,2,3,4]
- Refactoring [1,2,3,4]
- Reverse engineering [1,2,3,4]
- Tools for software maintenance and evolution [1,2,3,4]

[12] Available on Amazon; ISBN-13: 978-9812384263.
[13] Available on the IEEE Xplore Digital Library; access provided by WSU Libraries.
[14] The student learning outcomes for the MSSE program are labeled from '1' to '4'.

Learning outcomes and evaluation:

Students that successfully complete the course will be able to:

1. Work with a project team to add/modify features of existing software systems.
2. Apply the corrective, perfective, adaptive and preventive types of software changes and maintenance types.
3. Apply regression testing techniques and verify that a change has not introduced new bugs.
4. Determine the initial and estimated impact sets of a change.
5. Apply techniques and tools to facilitate the understanding of an existing software system.
6. Apply appropriate refactoring techniques to improve the quality of the software.
7. Determine actions that need to be accomplished to perform software migration.

Mapping student learning outcomes, course topics, and evaluations:

Student Learning Outcomes	Course topics/dates	Evaluation of Outcome
1	Software maintenance activities (weeks 9, 10, 12)	Project
2	Software maintenance categories (week 1, 6)	Project
3	Regression testing (week 2)	Mid-term 1, project
4	Change impact analysis (week 3, 4)	Mid-term 1
5	Program comprehension and reverse engineering (week 10, 13)	Final exam, project
6	Quality measurement and improvement (week 7, 8, 11, 12)	Mid-term 2, project
7	Software migration (week 14)	Final exam

Week-by-week schedule:

Week	Topics	Evaluation
1	Fundamentals of software maintenance and evolution. Maintenance categories categories.	
2	Regression testing.	
3	Change impact analysis.	
4	Program slicing.	
5	Specifying, reviewing, and controlling software maintainability.	Mid-term 1

6	Management issues during software maintenance (e.g., organizational objectives, outsourcing).	
7	Maintenance cost estimation.	
8	Software maintenance measurement.	Project deliverable 1
9	Software maintenance process. Software maintenance activities.	
10	Program comprehension.	
11	Reengineering.	Mid-term 2
12	Refactoring.	
13	Reverse engineering.	
14	Migration.	
15	Retirement.	Project deliverable 2
16		Final exam

Assignments:

The assignments for this course consist of a group project in which students will work together to maintain an existing software system. Students will work with a real world open source system. For the first deliverable, they will be asked to handle a set of change requests consisting of bug fixing and addition of new features. In addition to the code implementing the change requests, students will deliver a report describing the steps, techniques, and tool that were used to implement the change request. Of critical importance here are the comprehension activities prior to the change requests, pre/post-factoring activities, bug localization, change impact analysis, unit testing and regression testing activities. Assessment criteria include the clarity of the report, the choice of techniques and justification of design decisions. For the second deliverable students will evaluate the quality of the open source system using measures that are specific to software maintenance. Those metrics may include changeability, testability, and stability. The choice and justification of the selected metrics will be part of the assessment criteria. Students will apply appropriate modernization techniques such as refactoring to improve the quality of the existing system. The choice of modernization techniques, the justification of the undertaken activities, and the history of the version control tool used by the students will be given high importance. In addition to the improved code, students will provide a report reflecting of the quality of the system before and after the modernization techniques.

All team members are expected to contribute equally to all project deliverables and to all components of each project deliverable. As part of all deliverables, the team will list the tasks assigned to each team member, the tasks completion percentages, and the tasks completion dates. The instructor will use the history of the version control system (e.g., submitted artifacts such as code and documentation, time stamps, commit messages) to verify the task completion percentages and the individual contribution of each team member. In addition, each team member will fill a confidential peer evaluation questionnaire evaluating her/his own performance and the performance of the rest of the team. The peer evaluation will be used as guidance and will help the instructor to understand the team dynamics and the contribution of the individual team members but it will not be used directly as part of the students' grades. The task assignment, the version control history, and the peer evaluation questionnaire will be used to adjust the project grade for individual team members. The team will receive the same group grade (g) for a given deliverable. The group grade will then be adjusted by an individual weight (w_i) between 0.00 and 1.00, which corresponds to the percentage of the work completed by each individual that was assigned to her/him. If the performance of an individual team member is considered unsatisfactory, i.e., she/he completes less than 80% of the work that was assigned to her/him, then to avoid penalizing the team for poor performance of individual team member, the grade of the team will be adjusted positively by discarding the tasks assigned to the team member that performed poorly.

In a team with 4 members for example, an equal task distribution will imply that each team member will complete a set of tasks that account about 25% of the work. Suppose that team members 1, 2, and 3 complete 100%, 95% and 90% of their tasks, respectively. Their individual weights will then be equal to 1, 0.95, and 0.9, respectively. Suppose also that team member 4 only completed 30% of the tasks that were assigned to her/him; his individual weight will then be 0.3. As team member 4 performed poorly, the group grade for team members 1, 2, and 3 will be adjusted by discarding the task of team member 4, i.e., by discarding 25% of the project.

The group grade will be assigned by evaluating the following components:

Deliverables	**Components**	**Percentage of the final grade**
Deliverable 1		**25%**
	Delivered software code	10%
	Software documentation	10%
	Report	5%
Deliverable 2		**25%**
	Delivered software code	10%
	Software documentation	10%
	Report	5%

Grading framework: Course grades are based on 3 exams (two mid-terms and one final) totaling 50% of the final grade and a project totaling 50% of the final grade.

Final grades will be awarded on the following scale:

Interval	Grade
>= 95%	A
>=90% and <95%	A-
>=87% and <90%	B+
>=83% and <87%	B
>=80% and <83%	B-
>=77% and <80%	C+
>=73% and <77%	C
>=70% and <73%	C-
>=65% and <70%	D+
>=60% and <65%	D
< 60%	F

Course rules:

You must take exam during the assigned test period. Failure to do so will result in a score of zero. However, in extraordinary circumstances and at the discretion of the instructor, a make-up exam may be offered. An advanced notice must be given to the instructor beforehand.

Unless posted otherwise, assignment documents shall be submitted electronically.

Late penalty is a flat 10% deduction per day. Late assignments may be turned up to one week after the original due date, and an advanced notice must be given to the instructor beforehand for the late submission. No homework will be accepted after its due day without advanced notice or special permission from the instructor.

Uncompleted peer evaluations will be returned for revisions. 10% will be deducted each day calculated from the original due day until the evaluation is completed or a total of 70% is deducted.

Reasonable Accommodation:

Reasonable accommodations are available for students with a documented disability. If you have a disability and need accommodations to fully participate in this class, please either visit or call the Access Center (Washington Building 217; 509-335-3417) to schedule an appointment with an Access Advisor. All accommodations MUST be approved through the Access Center.

Academic Integrity:

I encourage you to work with classmates on assignments. However, each student must turn in original work. No copying will be accepted. Students who violate WSU's Standards of Conduct for Students will receive an F as a final grade in this course, will not have the option to withdraw from the course and will be reported to the Office Student Conduct. Cheating is defined in the Standards for Student Conduct WAC 504-26-010 (3). It is strongly suggested that you read and understand these definitions. (Read more: http://apps.leg.wa.gov/wac/default.aspx?cite=504-26-010)

Safety:

Washington State University is committed to maintaining a safe environment for its faculty, staff, and students. Safety is the responsibility of every member of the campus community and individuals should know the appropriate actions to take when an emergency arises. In support of our commitment to the safety of the campus community the University has developed a Campus Safety Plan, http://safetyplan.wsu.edu. It is highly recommended that you visit this web site as well as the University emergency management web site at http://oem.wsu.edu/ to become familiar with the information provided.

Software Testing

Course Name: Software Testing
Course Number: Cpt S 582
Credits: 3
Lecture Hours: 3
Schedule: Offered online (asynchronously) via Global Campus
Prerequisites: Graduate standing.
Course required/elective: required.

Professors/Coordinators: Venera Arnaoudova, Bolong Zeng, and Evan Olds.
Office: EME 127
Phone: (509) 335-1360
Email: varnaoud@eecs.wsu.edu

Recommended textbook(s):
[1] Paul Jorgensen, "Software Testing, A Craftman's Approach", CRC Press, 2013 (4th Edition).[15]
[2] Ilene Burnstein, "Practical Software Testing", Springer 2003.[16]
[3] Imran Bashir and Amrit Goel, "Testing Object-oriented Software: Life Cycle Solutions", Springer, 1999.[17]
[4] Robert V. Binder, Testing Object-Oriented Systems - Models, Patterns, and Tools, Addison-Wesley, 1999.[18]
[5] Ian Sommerville, "Software Engineering", Addison Wesley (9th Edition).[19]

Recommended journal/conference articles:

[7] M. J Harrold A testing Roadmap, ICSE 2000.
[8] Coverage Criteria for Logical Expressions, Paul Ammann, Jeff Offutt and Hong Huang. ISSRE '03. pages 99-107.
[9] Thomas J. Ostrand, Marc J. Balcer: The Category-Partition Method for Specifying and Generating Functional Tests. Commun. ACM 31(6): 676-686 (1988).
[10] T. Chow, Testing software design modeled by finite-state machines, IEEE TSE, May 1978.
[11] Gupta and Gupta, Data Flow Testing. The Compiler Design Handbook. Sep 2002.
[12] Harrold and McGreggor, Incremental testing of object-oriented class structures, IEEE ICSE proceedings, 1992.
[13] Tai and Daniels, Interclass test Order for Object-Oriented Software, Journal of Object-Oriented Programming, July/August 1999.
[14] Le Traon et al, Efficient Strategies for Integration and Regression Testing of OO Systems, IEEE Transactions on Reliability, March 2000, Vol. 49.

Course description: software testing, testing levels, testing objectives, testing techniques, test related measures, testing tools.

[15] Available on Amazon; ISBN-13: 978-1466560680.
[16] Available on Amazon; ISBN-13: 978-0387951317.
[17] Available on Amazon; ISBN-13: 978-0387988962.
[18] Available on Amazon; ISBN-13: 078-5342809381.
[19] Available on Amazon; ISBN-13: 978-0137035151.

Overview and Course Goals: The course teaches students the fundamentals of software testing. It teaches how to perform testing at different levels (e.g., unit testing, integration testing) and for different objectives (e.g., alpha testing, performance testing, stress testing). Students will learn to apply different testing techniques (e.g., boundary-value analysis, decision tables), how to evaluate the results of the tests as well as the quality of the tests.

Course topics and the corresponding program learning outcomes[20]**:**
- Fundamental software testing concepts [1,2,3,4]
- Different test levels (e.g., unit testing, integration testing, system testing) [1,2,3,4]
- Objectives of testing: acceptance testing, installation testing, alpha/beta testing, performance testing, etc. [1,2,3,4]
- Testing Techniques (e.g., black box, white box, mutation testing, etc.) [1,2,3,4]
- Reliability Evaluation [1,2,3,4]
- Test-related measures (e.g., fault density, mutation score) [1,2,3,4]
- Test planning and documentation [1,2,3,4]

Learning outcomes:

Students that successfully complete the course will be able to:

1. Describe different test levels and testing objectives.
2. Apply test methods for the different phases of development and life cycle of the software.
3. Identify coverage and acceptance criteria for the test based on the programming language activities, phase of development.
4. Assess the quality and reliability of the software system.
5. Plan and appropriately document for software testing activities.

Mapping student learning outcomes, course topics, and evaluations:

Student Learning Outcomes	Course topics/dates	Evaluation of Outcome
1	Software testing levels (weeks 2 and 3)	Mid-term 1
2	Software testing techniques (weeks 4-11)	Mid-term 1, mid-term 2, and project.
3	Software testing fundamentals and key issues (weeks 1 and 2)	Project
4	Quality and reliability evaluation (week 12, 13, and 14)	Final exam

[20] The student learning outcomes for the MSSE program are labeled from ‘1’ to ‘4’.

5	(Week 15)	Final exam

Week-by-week schedule:

Week	Topics	Readings	Evaluation
1	Testing Fundamentals: terminology and key issues (e.g., adequacy criteria, oracle, infeasible paths).	- Ch.1 and 2 [1]; - Ch. 1 [2] - Ch. 22 [5]	
2	Test levels: the target of the test (unit testing, integration testing, system testing).	- Ch. 6 [2]; - [7];	
3	Objectives of testing: acceptance testing, installation testing, alpha/beta testing, performance testing, etc.		
4	Testing Techniques: techniques based on intuition and experience.		Project deliverable 1
5	Black box testing techniques. Data equivalence classes (boundary partitions) and the category-partition method.	- Ch. 5, 6, and 7 [1] - Ch. 4 [2] - [8] - [9]	Mid-term 1
6	White box testing techniques. Edges, expressions, data streams.	- Ch. 9, 10, and 11 [1] - Ch5 [2] - [11]	
7	Fault-based testing techniques: mutation testing.		
8	Usage-based techniques: operational profiles.		
9	Model-based techniques: finite-state machines.	- Ch. 7 [4] - [10]	
10	Techniques based on nature of application: concurrent, web, embedded, and real-time systems.		Project deliverable 2
11	Techniques based on nature of application: object-oriented programs. E.g., Members Draw Minimal Data Usage Matrix (MADUM), Order for Inter-Class Integration Testing of Object-Oriented Software.	- Ch. 6 and 7 [3] - [12] - [13] - [14]	Mid-term 2
12	Reliability Evaluation.	- Ch. 15 [5]	
13	Processes and techniques for developing highly dependable software.	- Ch. 13 [5]	
14	Test-related measures: evaluation of the program under test and evaluation of the performed tests.		
15	Test process. Planning for testing activities and documentation.	- Ch. 8, 23, and 24 [5]	Project deliverable 3
16			Final exam

Assignments:

The assignments for this course consist of a group project in which students will work together and apply several testing techniques learned in class. In the first project deliverable students will be asked to implement a software system given a description of a client need. Students will produce a report explaining the role of each team member and all activities that took place during the development process. Assessment criteria for this deliverable include the clarity of the report, the choice and justification of followed practices and process, and the quality of the produced software. Particular importance will be paid to the functionality provided by the software and whether or not this functionality fulfils the user requirements, the history of the project activities in the version control tool, the coding standards used during development, the code readability and documentation. For the second deliverable, students will test the system using black box and white box testing techniques. Students will be assessed on the choice of techniques and the justification and on the quality of the test cases (e.g., completeness, coverage, discovered bugs). In addition to the test related artifacts, students will submit a report explaining the degree of confidence in the correctness of the software, the tools and techniques used, and they will reflect on the suitability of their choices and provide alternative solutions if the conclusion shows that a technique/tool that was used did not show to be efficient. For the third deliverable students will be given the software artifacts of another team and they will 1) evaluate the quality of the tests performed by the original developers, 2) further test the system using operational profiles and mutation testing, and 3) evaluate the program under test and the quality of the performed tests. In addition to the test related artifacts, student will produce a report describing the improvement of the test cases. The evaluation criteria for the third deliverable include the choice and justification of evaluation criteria for the existing test cases, the strategy, techniques, and tools chosen to improve the test cases, the thoroughness of the analysis comparing the existing test cases and the improved test cases from different aspects (e.g., completeness, performance).

All team members are expected to contribute equally to all project deliverables and to all components of each project deliverable. As part of all deliverables, the team will list the tasks assigned to each team member, the tasks completion percentages, and the tasks completion dates. The instructor will use the history of the version control system (e.g., submitted artifacts such as code and documentation, time stamps, commit messages) to verify the task completion percentages and the individual contribution of each team member. In addition, each team member will fill a confidential peer evaluation questionnaire evaluating her/his own performance and the performance of the rest of the team. The peer evaluation will be used as guidance and will help the instructor to understand the team dynamics and the contribution of the individual team members but it will not be used directly as part of the students' grades. The task assignment, the version control history, and the peer evaluation questionnaire will be used to adjust the project grade for individual team members. The team will receive the same group grade (g) for a given deliverable. The group grade will then be adjusted by an individual weight (w_i) between 0.00 and 1.00, which corresponds to the percentage of the work completed by each individual that was assigned to her/him. If the performance of an individual team member is considered unsatisfactory, i.e., she/he completes less than 80% of the work that was assigned to her/him, then to avoid penalizing the team for poor performance of individual team member, the grade of the team will be adjusted positively by discarding the tasks assigned to the team member that performed poorly.

In a team with 4 members for example, an equal task distribution will imply that each team member will complete a set of tasks that account about 25% of the work. Suppose that team members 1, 2, and 3 complete 100%, 95% and 90% of their tasks, respectively. Their individual weights will then be equal to 1, 0.95, and 0.9, respectively. Suppose also that team member 4 only completed 30% of the tasks that were assigned to her/him; his individual weight will then be 0.3. As team member 4 performed poorly, the group grade for team members 1, 2, and 3 will be adjusted by discarding the task of team member 4, i.e., by discarding 25% of the project.

The group grade will be assigned by evaluating the following components:

Deliverables	**Components**	**Percentage of the final grade**
Deliverable 1		**15%**
	Delivered software code	5%
	Software documentation	5%
	Report	5%
Deliverable 2		**20%**
	Quality of test cases	15%
	Report	5%
Deliverable 3		**15%**
	Evaluation of the existing test cases	5%
	Quality of the added test cases	5%
	Report	5%

Grading framework:

Course grades are based on 3 exams (two mid-terms and one final) totaling 50% of the final grade and a project totaling 50% of the final grade. The final exam is cumulative.

Final grades will be awarded on the following scale:

Interval	Grade
>= 95%	A
>=90% and <95%	A-
>=87% and <90%	B+
>=83% and <87%	B
>=80% and <83%	B-
>=77% and <80%	C+
>=73% and <77%	C
>=70% and <73%	C-
>=65% and <70%	D+
>=60% and <65%	D
< 60%	F

Course rules:

You must take exam during the assigned test period. Failure to do so will result in a score of zero. However, in extraordinary circumstances and at the discretion of the instructor, a make-up exam may be offered. An advanced notice must be given to the instructor beforehand.

Unless posted otherwise, assignment documents shall be submitted electronically.

Late penalty is a flat 10% deduction per day. Late assignments may be turned up to one week after the original due date, and an advanced notice must be given to the instructor beforehand for the late submission. No homework will be accepted after its due day without advanced notice or special permission from the instructor.

Uncompleted peer evaluations will be returned for revisions. 10% will be deducted each day calculated from the original due day until the evaluation is completed or a total of 70% is deducted.

Reasonable Accommodation:

Reasonable accommodations are available for students with a documented disability. If you have a disability and need accommodations to fully participate in this class, please either visit or call the Access Center (Washington Building 217; 509-335-3417) to schedule an appointment with an Access Advisor. All accommodations MUST be approved through the Access Center.

Academic Integrity:

I encourage you to work with classmates on assignments. However, each student must turn in original work. No copying will be accepted. Students who violate WSU's Standards of Conduct for Students will receive an F as a final grade in this course, will not have the option to withdraw from the course and will be reported to the Office Student Conduct. Cheating is defined in the Standards for Student Conduct WAC 504-26-010 (3). It is strongly suggested that you read and understand these definitions. (Read more: http://apps.leg.wa.gov/wac/default.aspx?cite=504-26-010)

Safety:

Washington State University is committed to maintaining a safe environment for its faculty, staff, and students. Safety is the responsibility of every member of the campus community and individuals should know the appropriate actions to take when an emergency arises. In support of our commitment to the safety of the campus community the University has developed a Campus Safety Plan, http://safetyplan.wsu.edu. It is highly recommended that you visit this web site as well as the University emergency management web site at http://oem.wsu.edu/ to become familiar with the information provided.

Software Quality

Course Name: Software Quality
Course Number: Cpt S 583
Credits: 3
Lecture Hours: 3
Schedule: Offered online (asynchronously) via Global Campus.
Prerequisites: Graduate standing.
Course required/elective: required.

Professors/Coordinators: Venera Arnaoudova, Bolong Zeng, and Evan Olds.
Office: EME 127
Phone: (509) 335-1360
Email: varnaoud@eecs.wsu.edu

Textbook(s):
[1] S.H. Kan, Metrics and Models in Software Quality Engineering, 2nd ed., Addison- Wesley, 2002.[21]
[2] D. Galin, Software Quality Assurance: From Theory to Implementation, Pearson Education Limited, 2004.[22]
[3] IEEE Std. 1028-2008, Software Reviews and Audits, IEEE, 2008.[23]

Course description: Software quality, quality assurance, process and product quality, software measures, quality attributes, quality management.

Overview and Course Goals: In this course students will learn about the different facets of software quality. They will also learn to define specific quality goals and develop a plan to meet these goals. The course will teach techniques that allow to manage software quality as well as metrics that can be used to assess different quality attributes that are related to the defined quality goals.

Course topics and the corresponding program learning outcomes[24]:
- Software quality models and improvement [1,2,3,4]
- Software quality assurance. [1,2,3,4]
- Software Verification and Validation (V&V) [1,2,3,4]
- Software quality measurement [1,2,3,4]
- Software quality management [1,2,3,4]

Learning outcomes and evaluation:

Students that successfully complete the course will be able to:

1. Measure the degree to which software artifacts achieve the desired quality.

[21] Available on Amazon; ISBN-13: 978-0133988086.
[22] Can be found on Amazon; ISBN-13: 978-0201709452.
[23] Available on the IEEE Xplore Digital Library; access provided by WSU Libraries.
[24] The student learning outcomes for the MSSE program are labeled from '1' to '4'.

2. Explain tradeoffs among cost, schedule, and quality.
3. Defining specific quality goals, and estimating the effort and schedule of software quality activities.
4. Determine how software quality is affected by the software development process used.
5. Select and use standards in the quality management process.
6. Explain the process of and conduct different activities to manage software quality such as reviews, audits, and walkthroughs.
7. Measure and relate the software quality to the appropriate quality attributes.

Mapping student learning outcomes, course topics, and evaluations:

Student Learning Outcomes	Course topics/dates	Evaluation of Outcome
1	Software product quality measurement (weeks 5 and 14)	Mid-term 2, final exam, and project
2	Quality tradeoffs (weeks 1 and 2)	Mid-term 1, project
3	Quality definition (weeks 3 and 5)	Mid-term 1
4	Software process quality (week 4)	Project
5	Quality management (week 13)	Mid-term 2, project
6	Quality assurance (weeks 8, 9, and 10)	Final exam
7	Quality measurement (week 14)	Final exam, project

Week-by-week schedule:

Week	Topics	Evaluation
1	Software engineering culture and ethics.	
2	Value and costs of quality.	
3	Quality models and characteristics.	
4	Software process quality.	
5	Software product quality.	
6	Quality improvement.	Mid-term 1
7	Safety-critical systems.	Project deliverable 1
8	Software quality assurance.	
9	Software Verification and Validation (V&V).	
10	Reviews and audits: e.g., management reviews, technical reviews, systematic walk-through.	
11	Software quality requirements: influence factors, dependability, and software	Mid-term 2

	integrity levels.	
12	Defect characterization.	Project deliverable 2
13	Software quality management techniques.	
14	Software quality measurement.	
15	Software quality tools.	
16		Final exam, project deliverable 3

Assignments:

The assignments for this course consist of a group project in which students will work together to define, assure, and assess the quality of a software system. Students will be given a requirements and specification document corresponding to a client's need. For the first project deliverable students will be asked to 1) define quality goals and 2) estimate the cost of software quality based on the economic assessment of the software quality development and maintenance processes. Students will be assessed on the level of depth of the quality assurance plan, the completeness and clarity of the defined goals, the choice of quality measures and their justification (with appropriate references), the strategies that will be used to assure that the quality goals are met as well as the justification for choosing those strategies. Students will also be assessed on the clarity of the scope definition of the quality assessment plan. For the second deliverable student will design and develop the software product and show that it meets the previously defined quality goals by following the strategies defined in the quality assessment plan. Assessment criteria include the clarity of the report and the ability to summarize the results and objectively describe whether or not the quality goals are met. Moreover, students will be assessed on the improvements that they suggest to the quality assessment plat that would be expected to produce better results. Those improvements include but are not limited to clarification and—or further specification of the quality goals, the selection of alternative/additional quality measures, the selections of different techniques/tools to collect data, the selection of alternative/additional quality assurance strategies. For the third deliverable students will assess the process and product quality of a project developed by another team and suggest quality improvement. Assessment criteria are similar to the assessment criteria of the second deliverable.

All team members are expected to contribute equally to all project deliverables and to all components of each project deliverable. As part of all deliverables, the team will list the tasks assigned to each team member, the tasks completion percentages, and the tasks completion dates. The instructor will use the history of the version control system (e.g., submitted artifacts such as code and documentation, time stamps, commit messages) to verify the task completion percentages and the individual contribution of each team member. In addition, each team member will fill a confidential peer evaluation questionnaire evaluating her/his own performance and the performance of the rest of the team. The peer evaluation will be used as guidance and will help the instructor to understand the team dynamics and the contribution of the individual team members but it will not be used directly as part of the students' grades. The task assignment, the version control history, and the peer evaluation questionnaire will be used to adjust the project grade for individual team members. The team will receive the same group grade (g) for a given deliverable. The group grade will then be adjusted by an individual weight (w_i) between 0.00 and 1.00, which corresponds to the percentage of the work completed by each individual that was assigned to her/him. If the performance of an individual team member is considered unsatisfactory, i.e., she/he completes less than 80% of the work that was assigned to her/him, then to avoid penalizing the team for poor performance of individual team member, the grade of the team will be adjusted positively by discarding the tasks assigned to the team member that performed poorly.

In a team with 4 members for example, an equal task distribution will imply that each team member will complete a set of tasks that account about 25% of the work. Suppose that team members 1, 2, and 3 complete 100%, 95% and 90% of their tasks, respectively. Their individual weights will then be equal to 1, 0.95, and 0.9, respectively. Suppose also that team member 4 only completed 30% of the tasks that were assigned to her/him; his individual weight will then be 0.3. As team member 4 performed poorly, the group grade for team members 1, 2, and 3 will be adjusted by discarding the task of team member 4, i.e., by discarding 25% of the project.

The group grade will be assigned by evaluating the following components:

Deliverables	**Components**	**Percentage of the final grade**
Deliverable 1		**15%**
	Quality goals and cost estimation	5%
	Quality assurance strategies	5%
	Report	5%
Deliverable 2		**20%**
	Delivered software code	10%
	Software documentation	5%
	Report	5%
Deliverable 3		**15%**
	Evaluation of the process and product quality	5%
	Suggestions for quality improvement	5%
	Report	5%

Grading framework: Course grades are based on 3 exams (two mid-terms and one final) totaling 50% of the final grade and a project totaling 50% of the final grade.

Final grades will be awarded on the following scale:

Interval	Grade
>= 95%	A
>=90% and <95%	A-
>=87% and <90%	B+
>=83% and <87%	B
>=80% and <83%	B-
>=77% and <80%	C+
>=73% and <77%	C
>=70% and <73%	C-
>=65% and <70%	D+
>=60% and <65%	D
< 60%	F

Course rules:

You must take exam during the assigned test period. Failure to do so will result in a score of zero. However, in extraordinary circumstances and at the discretion of the instructor, a make-up exam may be offered. An advanced notice must be given to the instructor beforehand.

Unless posted otherwise, assignment documents shall be submitted electronically.

Late penalty is a flat 10% deduction per day. Late assignments may be turned up to one week after the original due date, and an advanced notice must be given to the instructor beforehand for the late submission. No homework will be accepted after its due day without advanced notice or special permission from the instructor.

Uncompleted peer evaluations will be returned for revisions. 10% will be deducted each day calculated from the original due day until the evaluation is completed or a total of 70% is deducted.

Reasonable Accommodation:

Reasonable accommodations are available for students with a documented disability. If you have a disability and need accommodations to fully participate in this class, please either visit or call the Access Center (Washington Building 217; 509-335-3417) to schedule an appointment with an Access Advisor. All accommodations MUST be approved through the Access Center.

Academic Integrity:

I encourage you to work with classmates on assignments. However, each student must turn in original work. No copying will be accepted. Students who violate WSU's Standards of Conduct for Students will receive an F as a final grade in this course, will not have the option to withdraw from the course and will be reported to the Office Student Conduct. Cheating is defined in the Standards for Student Conduct WAC 504-26-010 (3). It is strongly suggested that you read and understand these definitions. (Read more: http://apps.leg.wa.gov/wac/default.aspx?cite=504-26-010)

Safety:

Washington State University is committed to maintaining a safe environment for its faculty, staff, and students. Safety is the responsibility of every member of the campus community and individuals should know the appropriate actions to take when an emergency arises. In support of our commitment to the safety of the campus community the University has developed a Campus Safety Plan, http://safetyplan.wsu.edu. It is highly recommended that you visit this web site as well as the University emergency management web site at http://oem.wsu.edu/ to become familiar with the information provided.

Advanced algorithms

Course Name: Advanced algorithms
Course Number: Cpt S 515
Credits: 3
Lecture Hours: 3
Schedule: Offered online (asynchronously) via Global Campus
Prerequisites: Graduate standing.
Course required/elective: required.

Professors/Coordinators: Zhe Dang.
Office: EME 135
Phone: (509) 335-7238
Email: zdang@eecs.wsu.edu

Required textbook(s):
[1] Cormen, Thomas H.; Leiserson, Charles E.; Rivest, Ronald L.; Stein, Clifford (2009). Introduction to Algorithms (3rd ed.). MIT Press and McGraw-Hill.[25]

Course description: Advanced algorithms, randomized algorithms, Hashing, approximation algorithms.

Overview and Course Goals: This course will provide graduate students with a solid background on modern algorithms. It will first introduce fundamentals of algorithms (e.g., Turing model, complexity) and then explore advanced algorithms for modern computer science in areas such as big data and software engineering.

Course topics and the corresponding program learning outcomes[26]:
- Fundamentals of algorithms [2]
- Basic algorithms [2]
- Advanced algorithms [2]
- NP-completeness [2]

Learning outcomes:

Students that successfully complete the course will be able to:

1. Apply probability theory in algorithm design.
2. Apply optimization theory in algorithm design.
3. Apply traditional principles like divide and conquer and dynamic programming in algorithm design.
4. Describe the theory foundation of intractability.
5. Apply algorithms for software analysis.
6. Formulate a NP-complete proof.

[25] Available on Amazon; ISBN 0-262-03384-4.
[26] The student learning outcomes for the MSSE program are labeled from '1' to '4'.

Mapping student learning outcomes, course topics, and evaluations:

Student Learning Outcomes	Course topics/dates	Evaluation of Outcome
1	Randomized algorithms and applications (weeks 8, 9, 10, 11)	Homeworks 4 and 5, and final exam.
2	Linear programming (week 6)	Homework 3
3	Greedy algorithm, divide and conquer, and dynamic programming (week 2)	Homework 1 and 2, and mid-term exam.
4	Foundations of algorithms (weeks 1, 12, 13, and 15)	Homework 6 and final exam.
5	Algorithms for software analysis (week 7)	Homeworks 4 and 5
6	NP-complete proofs (week 13)	Homework 6 and final exam.

Week-by-week schedule:

Week	Topics	Evaluation
1	Fundamentals on algorithms: Turing model and definition of algorithms, time/space complexity, complexity analysis preliminaries: recurrence functions.	
2	Basic algorithms: greedy, divide-conquer and dynamic programming techniques. Basic algorithms and their analysis on linear structures.	Homework 1
3	Basic algorithms and their analysis on tree structures. Basic algorithms and their analysis on graphs.	
4	Advanced algorithms: Network flows.	Homework 2
5	Bipartite matching.	
6	Linear programming.	Homework 3
7	Symbolic graph search and formal verification of software.	Mid-term exam
8	Randomized algorithms: Random variables and their properties.	
9	Hashing, hash functions and universal hash functions.	Homework 4
10	Breaking MD5, Las Vegas, and Monte Carlo algorithms: examples.	
11	Google page rank and Markov chain, How to select a random test case.	Homework 5

12	P, NP, co-NP, PSPACE and #P.	
13	Many-to-one reduction and NP-completeness, NP-complete problems.	Homework 6
14	Thanksgiving break.	
15	Approximation algorithms for some NP-complete problems.	
16		Final exam

Grading framework: Course grades are based on 6 homework assignments totaling 30% of the final grade and on 2 exams (one mid-term and one final) exam totaling 70% of the final grade (30% and 40% for the mid-term and final exam, respectively).

Homework 1: Review problem for basic algorithms and their complexity. Students will be given a problem and asked to design an efficient algorithm to solve the problem. To solve the problem students should choose among the basic algorithms design principles. Assessment criteria will include the correctness of the algorithm and the time complexity of the proposed solution.

Homework 2: Problem on non linear structure and network flow. Graphs and trees are common non-linear data structures in Computer Science. Students will be given a problem and will be asked to modify and apply what they have learned in class about traditional network flow algorithms to solve the problem. Assessment criteria will include the choice of basic algorithm and the adaptation that is made to fit the specific problem.

Homework 3: Application problems on bipartite matching and linear optimization. Students will be given a problem related to the topics learned in weeks 5 and 6, where a metric of graph similarity will need to be computed in the context of program clustering. Assessment criteria will include the not only the efficiency of the algorithms but also the creativity to come up with a non-traditional approach to a traditional problem.

Homework 4: Application of symbolic encoding and hash for software verification. Student will be given a problem and will be asked to apply tools like BDD-solvers to verify the correctness of a small software system. Assessment criteria will include students' ability and efficiency to learn new software tools and new scripting languages.

Homework 5: Test case selection on a graph. Students will be given a small software system with an existing set of test cases and will be asked to apply algorithms learned in class to select a minimum set of test cases to be executed when the system is modified. Assessment criteria include the number of the test cases selected, the code coverage of those test cases, and the number of faults discovered by the test cases.

Homework 6: NP-completeness and approximation. Students will be given a set of problems and asked to prove that the problems are indeed NP-complete. Moreover, students will be asked to provide approximation algorithms to some known NP-complete problems. Assessment criteria include the clarity of the logical reasoning and the effectiveness of the approximation.

Final grades will be awarded on the following scale:

Interval	Grade
>= 95%	A
>=90% and <95%	A-
>=87% and <90%	B+

>=83% and <87%	B
>=80% and <83%	B-
>=77% and <80%	C+
>=73% and <77%	C
>=70% and <73%	C-
>=65% and <70%	D+
>=60% and <65%	D
< 60%	F

Course rules:

You must take exam during the assigned test period. Failure to do so will result in a score of zero. However, in extraordinary circumstances and at the discretion of the instructor, a make-up exam may be offered. An advanced notice must be given to the instructor beforehand.

Unless posted otherwise, assignment documents shall be submitted electronically.

Late penalty is a flat 10% deduction per day. Late assignments may be turned up to one week after the original due date, and an advanced notice must be given to the instructor beforehand for the late submission. No homework will be accepted after its due day without advanced notice or special permission from the instructor.

Bonus points will be added to your total class score for attendance as follows: 0 absence = 5% of the final grade, 1 absence = 4 %, 2 absences = 3%, and 3 or more absences = 0% bonus.

Reasonable Accommodation:

Reasonable accommodations are available for students with a documented disability. If you have a disability and need accommodations to fully participate in this class, please either visit or call the Access Center (Washington Building 217; 509-335-3417) to schedule an appointment with an Access Advisor. All accommodations MUST be approved through the Access Center.

Academic Integrity:

I encourage you to work with classmates on assignments. However, each student must turn in original work. No copying will be accepted. Students who violate WSU's Standards of Conduct for Students will receive an F as a final grade in this course, will not have the option to withdraw from the course and will be reported to the Office Student Conduct. Cheating is defined in the Standards for Student Conduct WAC 504-26-010 (3). It is strongly suggested that you read and understand these definitions. (Read more: http://apps.leg.wa.gov/wac/default.aspx?cite=504-26-010)

Safety:

Washington State University is committed to maintaining a safe environment for its faculty, staff, and students. Safety is the responsibility of every member of the campus community and individuals should know the appropriate actions to take when an emergency arises. In support of our commitment to the safety of the campus community the University has developed a Campus Safety Plan, http://safetyplan.wsu.edu. It is highly recommended that you visit this web site as well as the University emergency management web site at http://oem.wsu.edu/ to become familiar with the information provided.