Department of Computer Science Program Review
August, 2015

1. Introduction

1.1 Purpose of this Report
1.2 What are you most proud of?
1.3 Department Description

2. Detailed Description

2.1 Programs
  2.1.1. Purpose of each program
  2.1.2. Program assessment methods, outcomes, use of findings
2.2 Faculty (please see Annual Reports for more information)
  2.2.1 Profile
  2.2.2 Aggregate of research and scholarly activity
  2.2.3 Contributions to academic programs
2.3 Students
  2.3.1 Profile – admission criteria, number, gender, degrees
  2.3.2. Graduate Student productivity
  2.3.3 Graduates: number, degrees, placement

3. Self- Appraisal

3.1 Appraisal in program mission, disciplinary and professional, clients, student needs
  3.1.1 What do you do?
  3.1.2 Why do you do it?
  3.1.3 How well do you do it?
  3.1.4 How well does your program reflect your mission?
3.2 Appraisal based on your department strengths and weaknesses
  3.2.1 What are your strengths?
  3.2.2 What are your weaknesses?
  3.2.3 What are the constraints that limit you?
  3.2.4 What opportunities are potentially available to your department?
  3.2.5 What are possible threats to your department's mission?

4. Conclusion

4.1 The department's most promising future direction
4.2 Action plan activities based on this self-study

Appendices

Annual Report – past eight years
Faculty Vitae – past two years
Previous Program Review Report 2008
Department Facilities – room assignments, floor plan 2002 and 2015
Capstone Program (past two years), feedback
SROI (two years)
Assessment Guide

1. Introduction

At the time of our most recent Program Review, in 2008, the Department of Computer Science was in a transitional phase with a new department head and a host of junior faculty. Indeed, the department was noticeably bifurcated with roughly half the 14 full-time faculty being very senior, with many years of combined experience, and the other half being Assistant Professors on the tenure track.

At about that time, five main ideas about how to most effectively move forward were developed:

1. the department needed to be more research active in publishing, the pursuit of external funding, the management of the graduate program, and even decisions about new courses in the curriculum;
2. the department needed to concentrate on identifiable focus areas of research and instruction: software engineering and bioinformatics/data mining were chosen;
3. the department needed to take a more organized approach to advising as rising enrollments created time and space pressure;
4. the department needed to be more entrepreneurial, especially in raising revenue through online (Distance and Continuing Education) courses;
5. the department needed an improved climate regarding women faculty and women students.

The intervening years have seen continuous growth in enrollment and success as a department. We are proud of our progress but a little apprehensive about the future. We are managing right now, but what does the future hold?

1.1 Purpose of this report, including the period covered by the report. What was the date of the last report?

Program Review takes place every seven years. This report covers the period from 2008 through 2015. The date of the last report was 2008.

1.2 What are you most proud of?

There are so many things to choose from.
Paraphrasing from our own newsletter: "[we are in a] "Golden Age" of Computer Science at North Dakota State University. We are proud to preside in the presence of a University Distinguished Professor (William Perrizo), a Meier Junior Professor (Anne Denton), TWO NSF Career Award winners (Hyunsook Do and Wei Jin), and a Jefferson Science Fellow (Ken Nygard). This last is a highly prestigious appointment to a year in Washington, DC at the U.S. State Department."

However, if forced to choose just one, we are most proud of the highly coveted NDSU Advance/FORWARD Department Equity Award. This came with a cash prize that we used to upgrade the departmental conference room with a big-screen TV and video conference capabilities.

1.3 Department Description: Mission Statement

The Department of Computer Science and Operations Research at North Dakota State University strives to provide the highest possible quality programs for the citizens of North Dakota, the region, the United States, and the world. We expect to maintain our leadership within the state and region at both the undergraduate and graduate levels. We will strive for increased recognition and success in research. We will work to ensure that the citizens and businesses of North Dakota remain competitive with those from other states.

See Appendices for facility description.

2. Detailed Description

The mainstay of the undergraduate program is the Bachelor of Science degree, which traditionally holds about 90% of the students, with the other 10% enrolled in the Bachelor of Arts program. There is a long-standing 'double major' in Mathematics and Computer Science, that holds a handful of students each year, and two new joint programs with Physics and Statistics that are just getting started.

Graduate programs in both Computer Science and Software Engineering are listed with the date of first degree granted in parenthesis.

2.1 Programs

2.1.1

Bachelor of Arts

In Computer Science: less quantitative program that concentrates on web development, but still provides a very good foundation for working in any area of information technology.

Bachelor of Science

In Computer Science: comprehensive program providing a foundation for working in any area of information technology or going to graduate school.

In Computer Science and Physics: joint degree with Physics for students interested in the very active boundary between these two fields.

In Computer Science and Statistics: joint degree with Statistics for students interested in data analytics and related very active areas.

In Computer Science and Mathematics: joint degree with mathematics for students with a more analytical focus.

Master of Science

In Computer Science *(1990)*
Master of Science in Computer Science: comprehensive program to provide excellent opportunities for success in information technology jobs or pursuit of a Ph.D.

In Software Engineering *(1990)*

Master of Science in Software Engineering: comprehensive program to provide excellent opportunities in software development jobs or pursuit of a Ph.D.

Master of Software Engineering – M.S.E. *(2012)*
Master of Software Engineering: coursework-only program for software professionals.

Certificate of Software Engineering *(2009)*
Graduate Certificate in Software Engineering: four course introduction to software engineering for those without much background.

Doctor of Philosophy
In Computer Science *(1990)*
Ph.D. In Computer Science: comprehensive program to provide excellent opportunities for research jobs or higher education jobs.

In Software Engineering *(2002)*
Ph.D. In Software Engineering: same as Ph.D. in Computer Science.

2.1.2 Program assessment methods, outcomes, use of findings

All these programs are assessed in similar ways. Since none of the programs have a fixed set of courses or course ordering, the Department divided the courses into three groups: those which are almost always taken at the beginning of a student's program, those that may be taken at an intermediate stage, and those which are taken at the end of a program (capstones for undergraduate programs, and final disquisition defense for most graduate degrees). We use the beginning courses to establish a baseline of student mastery of program objectives. We use the intermediate courses to assess student improvement on program objectives as they progress through a program. The ending activities are used to assess student mastery of program objectives as they complete a program.

For each course or ending activity, we select randomly ten assignments or examinations. These are provided anonymously to the assessor, who is not the course instructor. The assessor uses one or more rubrics to assign a degree of mastery of selected objectives based on the sampled materials.

Additional details may be found in our Assessment Guide, which is an Appendix to this document.

2.2 Faculty

2.2.1 Current Personnel (Fourteen full time tenure track faculty)

Professors:

Slator, Brian – Department Head
Dr. Slator has taught courses in artificial intelligence (AI), multimedia educational systems, computer science problem solving, and comparative languages. His research interests revolve around active environments for learning, including the use of software agents, case-based reasoning, knowledge representation, multimedia systems, distance education, synthetic environments, and multi-user educational games. Dr. Slator is a recipient of the Ernest L.

Boyer International Award for Excellence in Teaching, Learning and Technology. Since fall, 2007, he has been Department Head.

Magel, Kenneth – Associate Head
Dr. Magel teaches a wide variety of courses, including software engineering, programming languages, and social implications of computing. His software engineering research activities explore what makes programming difficult and programs complex. Dr. Magel conducts seminars and courses in XML, C# and .net technologies. He coordinates the graduate programs in software engineering. Beginning July 1, 2007 he became Associate Head for the Department.

Perrizo, William – University Distinguished Professor
Dr. Perrizo teaches courses in database systems, data mining, bioinformatics, and networks. His research interests include database and information systems, data mining, data warehousing, distributed database systems, bioinformatics, precision agriculture, and remotely sensed data management and visualization. His research has been funded by many federal and private sources. Dr. Perrizo is a co-founder of the worldwide Virtual Conference on Bioinformatics. Dr. Perrizo has served in leadership roles for many conferences and on many boards and has a strong international reputation in research. In fall, 2007, he became one of the first seven University Distinguished Professors at NDSU, and in spring, 2008, was named Fargo-Moorhead Chamber of Commerce Professor

Nygard, Kendall – US State Department Jefferson Science Fellow
Dr. Nygard teaches courses in simulation, social implications of computing, mathematical modeling, network optimization, systems analysis and design, and software testing and maintenance. His research interests include software systems for military mission planning for cooperative control of autonomous aircraft systems, software agents, and geographic information systems (GIS) for school transportation. Primary sponsors of his research are the Air Force and Navy. Dr. Nygard received the Jefferson Science Fellowship for 2013-14. A highly prestigious appointment to a year in Washington, DC at the U.S. State Department – only a handful of scientists are chosen each year.

Ubhaya, Vasant
Dr. Ubhaya teaches courses in Discrete Mathematics, Algorithm Analysis, Performance Evaluation, Mathematical Programming, and Dynamic Programming. He does research in Algorithms, Optimization and Approximation, and publishes his results regularly in journals. He is often invited by professional societies to organize and chair sessions, and give talks at their meetings. His research has been supported by the National Science Foundation and EPSCOR.

Associate Professors:

Denton, Anne – James A. Meier Junior Professor (2011-2014)
Dr. Denton teaches courses in database management, bioinformatics, problem solving and foundations of computer science. Her research interests include data mining, bioinformatics, course management systems for distance education, and computational physics. Anne received the Meier Junior Professor award; she has also accepted the Graduate Coordinator roll starting 2013.

Do, Hyunsook – NSF Career Award Recipient *2012*
Dr. Do teaches courses in networks, network security, and software engineering. Her research program concerns software engineering, particularly software testing, maintenance, and empirical methodologies. Hyunsook received the NSF Career Award in 2012. Dr. Do moved to North Texas University starting Fall 2015.

Jin, Wei – NSF Career Award Recipient *2015*
Dr. Wei Jin teaches courses in comparative languages and information retrieval. Her research interests focus on Text Mining, Information Retrieval and social Network analysis and Bioinformatics. Dr. Jin received the NSF Career Award in 2015. The second time the computer science department faculty received this award. Dr. Jin was also awarded tenure effective July 2015.

Knudson, Dean – half time
Dr. Knudson is coordinator of the capstone program for Bachelor of Science students in CS and MIS. In this role he develops external sponsors for projects and mentors the student teams in project management. He teaches CSCI 445, Capstone: Software Projects. Dr. Knudson has extensive experience working as a development executive for Microsoft and several other companies. He is a half-time Associate Professor.

Kong, Jun
Dr. Kong is interested in visual modeling languages, model driven development and web-data interoperation. He teaches courses in operating systems and human computer interaction. Jun was promoted to Associate Professor with tenure effective spring, 2012.

Li, Juan
Dr. Juan Li teaches courses in artificial intelligence, parallel and distributed simulations. Her research interests are in networking and distributed systems. Dr. Li was promoted in summer of 2014.

Ludwig, Simone
Dr. Simone Ludwig joined the faculty in the fall of 2010. She teaches courses in assembly programming and artificial Intelligence. Her research interest combines distributed computing with artificial intelligence. Simone received tenure for July 2015.

Salem, Saeed
Dr. Salem teaches courses in Bioinformatics and Data Mining. His research interests are in bioinformatics, biological networks, data mining and machine learning. Saeed received tenure July 2015.

Walia, Gursimran
Dr. Walia teaches courses in Software Project Planning and Empirical Software Engineering. His research interests are empirical software engineering, psychology in software engineering, software quality, information assurance and software engineering for computer security. Dr. Walia was awarded the College Excellence in Teaching Award. Dr. Walia also received tenure July 2015.

Yan, Changhui
Dr. Yan teaches courses in Bioinformatics. His research interests include developing computational methods and tools to assist biologists to investigate problems in complex biological systems. Dr. Yan was promoted to Associate Professor in summer 2014.

Assistant Professor of Practice (Non Tenure Track Faculty)

Abufardeh, Sameer
Dr. Abufardeh teaches courses in Java. His research interest has been in the area of requirements engineering. He received his Ph.D. from the Department in fall, 2008.
Sameer was promoted to Professor of Practice effective spring, 2012. Dr. Abufardeh moved to the U. of Minnesota – Crookston starting Fall 2015.

Myronovych, Oksana
Dr. Myronovych teaches courses in Java, C#.NET, PHP, Software Testing, Modern Software Development, and the advanced Visual Basic .NET courses. Her research interest has been in the area of Requirements engineering and Software testing. She received a Ph.D. in 2009. She is currently the Faculty Advisor for the student chapter of UPE, the Computer Science Honor Society. Dr. Myronovych was promoted to Professor of Practice in December 2010.

Senior Lecturer

Latimer, Joseph
Mr. Latimer teaches courses in sections of UNIX, Java, and Web-scripting.

Kotala, Pratap
Mr. Kotala teaches courses in systems analysis and design and foundations of programming for MIS majors. He also teaches the distance and continuing education sections of business use of computers.

Lecturer

Fleming, Janet
Ms. Fleming teaches courses in Microsoft office suite, Visual Basic and COBOL

Radermacher, Alex
Mr. Radermacher teaches the beginning undergraduate courses in Java.

Krush, Joan – Student Advisor
Ms. Krush is a halftime lecturer and advisor. Joan has a MA in student development in Postsecondary Education. She assists our students with their advising needs, leads student recruiting, and teaches sections of University 189. Joan received the University Outstanding Academic Advisor award for 2013-14.

Systems Administrator

Olson, Nathan

Nathan Olson joined us July 2014 as the department systems administrator and does all the maintenance and purchasing of all the computer equipment. He also does the configurations for the departmental instructional laboratories.

Programmer Analysts – Research Technicians

Hokanson, Guy – half time

Mr. Guy Hokanson began his Programmer Analyst position August 2007. His duties include educational games, research and software development.

Borchert, Otto – half time

Mr. Otto Borchert began his Programmer Analyst position August 2007. His duties include educational games, research and software development. Mr. Borchert moved to Gonzaga University starting Fall 2015

Staff

Huber, Carole – Administrative Assistant

Ms. Huber coordinates the administrative functions of the department. This includes managing research and appropriated funds, purchasing and accounts payable. She is the contact person for all student employment applications, time-slips, and tuition waivers. She coordinates all Teaching/Research/Grading positions for the department.

Annette Sprague – Administrative Secretary

Annette Sprague joined the department May 2015. She carries out office support functions, including data development, reporting, survey work, and assisting students and faculty. She is the assistant to the Graduate Coordinators.

Opheim, Betty –Administrative Secretary, half time

Ms. Betty Opheim carries out inventory, equipment and software support functions and assists in back-up office support. She is half-time.

Dickerson, Jane – Office Assistant, half time

Ms. Dickerson is half time office support for the department. She assists students, faculty, and staff.

2.3 Students

2.3.1 –Graduate Student Profile (admission criteria, number, gender)

	2007/2008	2008/2009	2009/2010	2010/2011	2011/2012	2012/2013	2013/2014	2014/2015
MS -- CS								
Male	76	90	84	82	68	66	52	39
Female	21	31	25	31	21	25	22	22
Total	97	121	109	113	89	91	74	61
MS -- SE								
Male	24	30	34	36	36	38	32	35
Female	7	9	10	12	12	13	11	8
Total	31	39	44	48	48	51	43	43
MSE								
Male	n/a	n/a	n/a	4	3	24	29	27
Female	n/a	n/a	n/a	0	1	4	9	15
Total				4	4	28	38	42
CERT -- SE								
Male	n/a	n/a	n/a	1	3	0	4	5
Female	n/a	n/a	n/a	0	0	0	1	1
Total				1	3	0	5	6
PhD -- CS								
Male	14	17	25	28	29	21	17	23
Female	0	1	1	2	2	6	5	3
Total	14	18	26	30	31	27	22	26
PhD -- SE								
Male	10	10	10	11	13	15	14	18
Female	1	1	2	3	3	3	5	2
Total	11	11	12	14	16	18	19	20

2.3.2 Total Graduate Teaching Assistants for undergraduate courses (CSCI 114 – CSCI 116 – CSCI 122 – CSCI 159 – CSCI 227)

Fall 2007	19	Spring 2008	12
Fall 2008	15	Spring 2009	12
Fall 2009	15	Spring 2010	14
Fall 2010	18	Spring 2011	15
Fall 2011	15	Spring 2012	12
Fall 2012	14	Spring 2013	12
Fall 2013	13	Spring 2014	10
Fall 2014	11	Spring 2015	10

2.3.3 Graduates/Undergraduates Profile (number, time of degree)

	Enrollment Fall 2014						Total Degrees Fall 14/Spring 2015		
AY	1st FR	2nd SO	3rd JR	4th SR	Total UG	Total Grad	BS/ BA	MS Comp Sc. Software	PhD Comp Sc. Software
2014 - 2015	79	89	76	100	344	199	28/7	11/11	1/3
2013 - 2014	85	77	79	80	321	201	41/4	25/18	5/3
2012 - 2013	82	65	54	95	296	209	76/7	30/5	4/0
2011 - 2012	75	47	59	95	276	217	47/5	30/5	4/0
2010- 2011	68	63	55	74	267	210	38/2	26/6	0/1
2009 – 2010	71	55	48	79	253	197	38/2	20/6	1/1
2008 - 2009	66	54	44	84	248	187	40/3	17/4	1/1

3. Self-appraisal

3.1 Appraisal in regards to program mission, disciplinary and professional context, clients, or student needs

3.1.1 What do you do?

We fulfill our Mission Statement by offering a variety of undergraduate and graduate programs to meet the needs of local companies including Microsoft, John Deere, Eide Bailey, and many others, large and small. We pursue several research initiatives in which both funding agencies and regional companies have significant interest, including big data analysis, bioinformatics, mobile and cloud computing, and computer networks.

3.1.2 Why do you do it?

We try to meet the needs of students whether they choose graduate school or the workforce. We try to be responsive to local and regional employers as they suggest curricular modification. Our graduates are highly sought and are prepared for successful careers.

We try to provide best-practices programs which serve as models for colleges and universities in North Dakota, South Dakota, and much of Minnesota. As one of many recent examples, Valley City State University consulted actively with our faculty as they created their undergraduate

major in Software Engineering. The U. of Minnesota – Crookston bases their new software engineering program upon our design.

We have supplied many Computer Science, Information Technology, Software Engineering, and Management Information Systems faculty for universities in the tri-state area. We are the largest source of faculty in these areas for this region.

3.1.3 How well do you do it?
Every indication is that we do very well. Nearly all our graduates from any program have successful, full time careers in Computer Science or a closely related field.

In recent years, two of our faculties have received the very coveted NSF Career Award for junior faculty. One senior faculty was awarded a Jefferson Fellowship and spent a year working for the U.S. Department of State in Washington, D.C. More than half our faculties have received nationally competitive grants from external funding sources.

The Department is the regional leader in working to attract underrepresented groups to Computer Science and fostering a positive climate. In September, 2013, the department received the NDSU Advance Forward Department award. Several department faculty participated in national meetings on attracting underrepresented groups every summer.

3.1.4 How well does your program reflect your mission?
The programs we offer, the research we do, and the service we provide to the region, the nation, and the profession is driven by our Mission Statement. Periodically, we review our activities to ensure they continue to contribute to our mission.

3.2 Appraisal based on your department's strengths and weaknesses

3.2.1. What are your strengths?
Our biggest strength is our people. The faculty, staff, and administrators strive to use our limited resources to serve our students and the citizens of North Dakota. All our people respect each other. We do not have departmental factions.

Another major strength is our students, undergraduate and graduate. There are two student organizations. The student chapter of the ACM is very active, engaging many students in activities, organizing events, inviting speakers, hosting visitors. They are very welcoming and inclusive and present a friendly face to incoming students. Both undergraduates and graduates may join.

The Computer Science Honor Society (Upsilon Pi Epsilon) is a newer organization. They recognize top students and host the bi-annual employers breakfast the department sponsors in the morning before the career fair in the Fargodome each fall and spring.

Another major strength is our relationships with local industry. Our capstone course for undergraduates has an astonishing record of supporting student teams to meet real needs of area businesses. See http://csprojects.cs.ndsu.nodak.edu/capstone/PreviousProjects/.

The repeat business is what tells the story. Companies line up to participate, and many years there is a waiting list. Most recently an international component has been developed where joint projects with Universities in Germany and Sweden have been undertaken. Future international opportunities are planned for Australia.

Many local companies recognize our variety of graduate programs as significant assets for their success. Many graduate students work at area companies while pursuing their degree. We have developed a small Industry-University Consortium that has provided a few faculty with research opportunities and several students with stipends for research work.

We are well-equipped and well-appointed. Largely through entrepreneurial efforts we have outfitted faculty, staff, and students with modern equipment and office furnishings. We have developed an infrastructure for departmental virtualization in support of research, and an integrated system for data backups, security and protection.

Using entrepreneurial income from various sources we have managed our space better than anyone thought possible. A hallway was turned into offices. Another hallway became laboratory space. A lab has been divided into two offices. An open area near a receptionist area was converted to another office. All these innovations and renovations were driven by growth and ambition.

3.2.2 What are your weaknesses?

Our main weaknesses arise from the difficulties associated with managing our large enrollments with our available resources, and available space. We are currently searching for a tenure-track faculty member to replace a recent resignation. When we fill that position we will be at capacity, without room to add anyone, short of sharing offices. We are coping at the present, but there is no room for additional growth.

Our faculty share very small research laboratories. We have only one teaching laboratory. As the diagrams in the Department Facility Appendix show, we have gone as far as we can in accommodating growth by replacing hallways and other space with offices and laboratories. These modifications were mainly for adding instructors to meet the increasing teaching pressure, even (reluctantly) converting a faculty line in one case.

3.2.3 What are the constraints that limit you?

We are constrained by the realities.
Seven years ago in our Program Review we described a need for more space and more resources. Now, seven years later, we are wiser and more experienced.
We do not request more space. We know there is none to be had.
We do not want any more faculty lines. We do not have the space to house them.
We would like to stop requesting ‘extra section’ money to accommodate our wait lists and teach our over-capacity classrooms. We would like to stop hiring temporary instructors to teach extra sections.

3.2.4 What opportunities are potentially available to your department?

In terms of new program development, with existing faculty resources, a joint undergraduate and graduate program with the Department of Statistics in the very active, nationally vital area of big

data analysis would fit our mission and our faculty expertise. Expanded efforts in digital security and virtual plus augmented reality are exciting possibilities. Perhaps, we could exploit growing opportunities in the Internet of Things, if we can find the personnel and resources.

We may during the next five years be able to use digital technology to support a consortium of area schools to leverage our individual resources and expertise in joint research and curriculum efforts. Computer Science and Software Engineering are vital areas for our state and nation's future. The opportunities for expansion in Computer Science are greater than in almost any other field.

3.2.5 What are possible threats to your department's mission?
We have successfully pursued our mission for many years, and we have attained the status of regional leader in many respects. We have relied on growth for many years, and now we are approaching our sustainable capacity.

Another threat is the possibility of losing faculty. We have had a relatively long period of stability but this summer two important department members in Software Engineering left for other academic positions. The shortage of faculty in that area makes it difficult for us to offer enough SE graduate courses, especially online courses. Enrollment increases create a certain amount of personal and professional pressure.

4. Conclusion

After years of continuous growth we see ourselves approaching capacity. We are research leaders, with the aim of increasing research productivity, but we are finding it increasingly difficult to meet continuously growing teaching commitments.
We do not want to continue repeatedly requesting extra section money. We do not want increasing reliance on an ever-swelling cadre of adjunct faculty to keep pace with demand.
We think we see a way to make things better. We propose to cap enrollment.

4.1. The department's most promising future direction lies in planning for the future.

We are nearing overextension with barely enough faculty to effectively deal with current student enrollment. We fear this will someday be unsustainable. We have worked over the last few years on increasing the quality of our graduate students, while decreasing the number seeking a Master Degree, and decreasing the number of our overall graduate student numbers. We have introduced the Master of Software Engineering and an Option C in the Master of Science in Computer Science as coursework-only programs to meet the needs of regional industry while reducing the need for faculty-supervised Masters-level graduate student research**.**
We propose to develop a corollary plan for our undergraduate programs.
One long-range goal would be to reduce the undergraduate program to a smaller and more elite group, and use it to specifically aim students towards graduate studies here. This would raise the quality of both programs and free resources to begin once again growing the graduate program.

4.2. Action plan activities based on this self-study

We do not have a fully developed plan at this time and seek input from Program Review. What positive, constructive steps are being taken across campus to manage enrollment while bolstering quality?
We are discussing entrance standards for acceptance into our undergraduate programs. For example, we could propose that any undergraduate accepted in Computer Science must rank in the top 25% of their high school graduating class and have completed College Algebra or the equivalent with a B or better grade.
We hope to find the winning formula that will allow us to effectively manage our resources while increasing our quality and reputation as a top regional Computer Science program.

Appendices

Annual Report – past 8 years
See https://www.ndsu.edu/cs/policies_and_information/annual_report/

Faculty Vitae – past two years
See https://www.ndsu.edu/cs/about_us/faculty_and_staff/

Previous Program Review Report 2008
V. Recommendations from the Previous Program Review (in bold)

1. The department is working hard to buck the national trend towards lower undergraduate enrollments in computer science, and is currently succeeding. The department is taking a leadership position on the NDSU campus in the development of "twinning" and other programs to assure continued growth of their undergraduate program. The Program Review Committee commends the department for these efforts, and recommends that these efforts continue to be recognized and supported by the college and by the university.
Undergraduate enrollments have continued to grow over the last seven years. They now are a serious problem for the department. We do not have sufficient staff to offer required and elective courses as often as students reasonably expect. More importantly, class sizes in these courses have grown beyond reasonable sizes, often reaching more than forty students in a single section and in some cases (eg. CSci222, CSci366, CSci489), going as high as 60+.

2. Resources given to the department for the 100-level "service courses" seem inadequate. Even with availability of online versions of these classes, the sections of these classes are always full. It is recommended that additional support be given to the department for teaching these 100-level courses, especially if student enrollments continue to rise. Additional funding for teaching assistants and additional office space for teaching assistants are needed.
We have added several sections to our service courses; however, the sections remain larger than optimal, routinely reaching sixty students or more. We fear the impending elimination of distance education as an option for service courses will place a significant additional load on our service courses, especially CSci 114 and CSci 116.

3. The department has critical space needs. Several new faculty positions have been approved for hire, and there is no office space to house them. The department has only one dedicated teaching lab, and professors are sharing research labs. It is recommended that the space needs of the department be reviewed by their college and the university.
Unfortunately, no new space has been given to the department during the last seven years. As explained elsewhere in this document, we have sacrificed hallways and other amenities to try to build additional offices for instructors.

4. Faculty turnover is constant issue the department faces. The loss of 1-2 faculty per year is common, usually for higher pay elsewhere. It is recommended that the department and college identify the reasons for this turnover, and that adequate resources be given to the department to hire and retain a quality faculty.

Faculty turnover has lessened. We lost only four faculty during the last five years. Unfortunately, two left this summer for other academic positions. However, we have a looming problem in that several faculty are above or nearly seventy years of age. At least some of these faculty are likely to retire in the next few years.

5. Involvement in research, publishing, and the obtaining of grants is uneven across the faculty. It is recommended that the chair of the department continue to encourage all tenure-track and tenured faculty to be active in research and obtaining grants.
This has been a priority. More faculty regularly publish thirteen out of fourteen) and try for externally competitive research grants (twelve out of fourteen).

6. The percentage of undergraduate students who finish their B.S. degrees appears to be lower in computer science than for their college as a whole. The current plan to hire a dedicated advisor should help with student retention. It is recommended that the department study the problem, and find ways to improve undergraduate graduation rates.
Two factors are significant which are unusual in our college: (1) students can get good jobs and careers before completing the bachelor's degree; and (2) most students know little about the computer science requirements and expectations before entering the program.

7. The last program review (1999) recommended that "The department should limit the number of new graduate students to ensure the quality of each student's experience." This problem is as large now as it was in 1999. It is recommended that the department place a high priority on improving the completion rate of graduate students, even if it requires a reduction in the numbers of graduate students accepted. The department has stated its own goals of reducing the number of graduate students, to about 120 graduate students. This would be a good start, but is still probably more students than can be adequately directed and advised, especially if the department's goal of increasing the percentage of Ph.D. students is realized. A dedicated course work advisor for graduate students would be very helpful, not only for the graduate students, but for those faculty members with very heavy advising loads.
We have improved the quality of our graduate population by increasing standards and eliminating most conditional admittances. We have worked on increasing the Ph.D. population and reduced the faculty load by introducing non-research degrees (the M.S.E. and Option C in the M.S. in C.S.).

8. It is recommended that the department work towards reducing the percentage of graduate students on quarter-time assistantships, increasing the percentage of graduate students on half-time assistantships, and reducing the extreme differences in pay among those with half-time assistantships. By having fewer, better-paid graduate students, the completion rate for their graduate students should improve.
We have increased the percentage of assistantships which are half time, but not eliminated the quarter time assistantship.

Department Facility – room assignments

Computer Science Building	Room #	Occupant Name	Occupant Type	Purpose of Room (complete only if Single-Use)
Quentin Burdick Building	108	Perrizo, William	Faculty	Research Lab
	110	Walia, Gursimran	Faculty	Research Lab
	110A	Walia, Gursimran	Faculty	Research Lab
	110B	Walia, Gursimran	Faculty	Research Lab
	112	Salem, Saeed	Faculty	Research Lab
	158	Denton, Anne	Faculty	Research Lab
	160	Teaching Assistants	Student	Research Lab
	162	Student Chapter	Student	Research Lab
	258	General Office	Staff	General Office
	A1	Conference Room	Other	Conference Room
	A2	Jin Wei / Juan Li	Faculty	Research Lab
	A3	Slator, Brian	Faculty	Research Lab
	A4	Slator, Brian	Faculty	Research Lab
	A5	Kong, Jun	Faculty	Research Lab
	A6	Ludwig, Simone	Faculty	Research Lab
	A7	NDSU Utility	Other	Utility Room
	A8	Jin Wei	Faculty	General Office
	A9	Ubhaya, Vasant	Faculty	General Office
	A10	Fleming, Janet	Faculty	General Office
	A11	Latimer, Joe	Faculty	General Office
	A12	Li, Juan	Faculty	General Office
	A13	Ludwig, Simone	Faculty	General Office
	A14	Kong, Jun	Faculty	General Office
	A15	Radermacher, Alex	Faculty	General Office
	A16	Perrizo, William	Faculty	General Office
	A17	Walia, Gursimran	Faculty	Research Lab
	A18	Yan, Changhui	Faculty	Research Lab
	A19	Nygard / Ubhaya	Faculty	General Office
	A20	Olson, Nate	Staff	General Office
	A21	Knudson, Dean	Faculty	General Office
	A22	Magel, Ken	Faculty	General Office
	A23	Do, Hyunsook	Faculty	General Office
	A24	Nygard, Ken	Faculty	General Office
	A25	Server/Utility	Other	Utility Room
	A26	Slator, Brian	Faculty	General Office
	A27	Yan, Changhui	Faculty	General Office
	A28	Denton, Anne	Faculty	General Office
	B1	Huber, Carole	Staff	General Office
	B2	Printer / Kitchen	Other	Kitchen
	B3	Myronovych, O	Faculty	General Office
	B4	Opheim, Betty	Staff	General Office
	B5	Kotala, Pratap	Faculty	General Office
	B6	Abufardeh, Sameer	Faculty	General Office
	B7	Krush, Joan	Faculty	General Office
	B8	Salem, Saeed	Faculty	General Office
	B9	Conference Room	Other	Conference Room
	244	Teaching Lab	Student	Teaching Lab
	244A	Utility /Storage	Other	Utility Room

Department facility 2002-2015

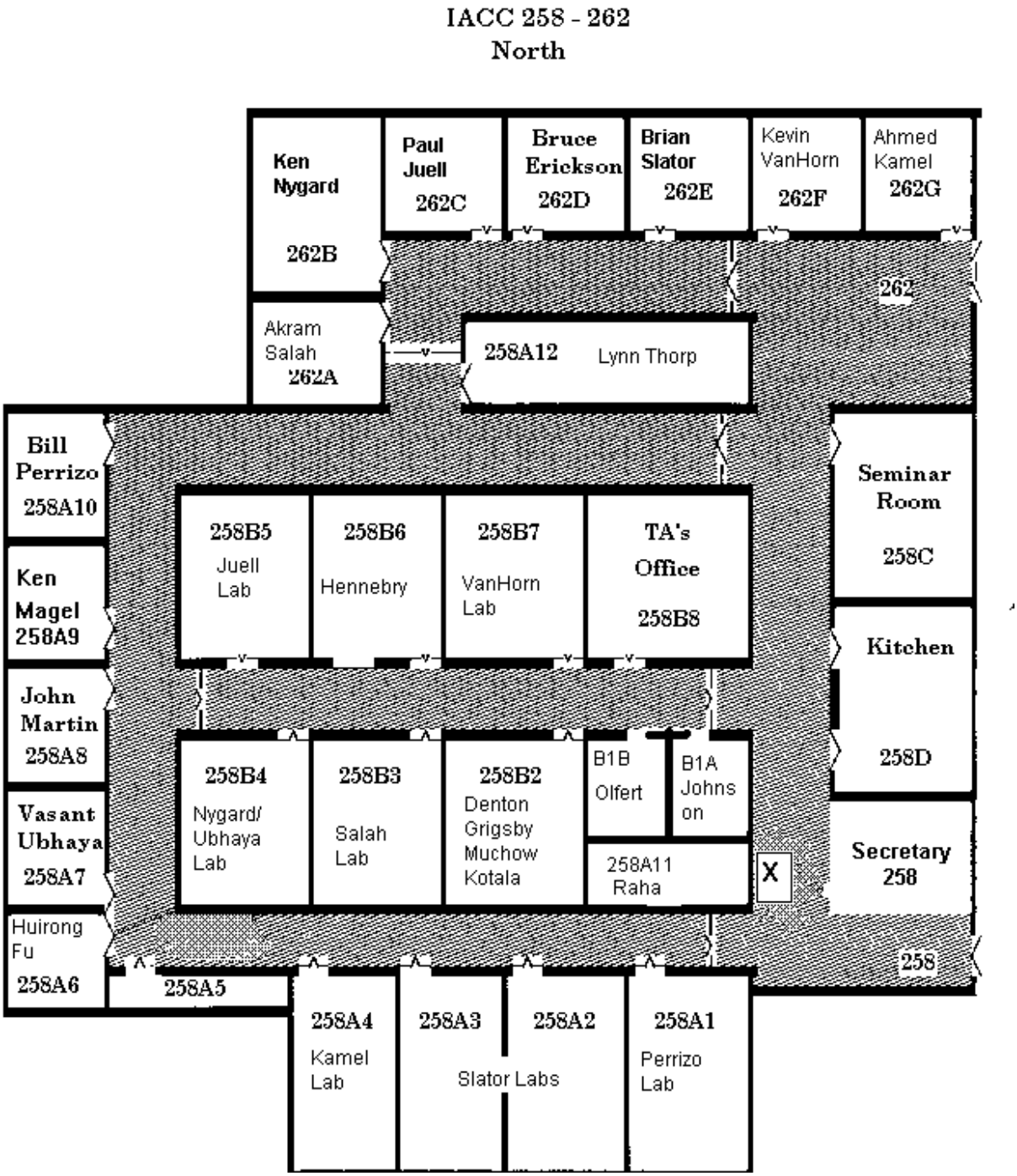

Figure 1: the Department of Computer Science floor plan in 2002 shows 12 faculty offices, 9 research labs, 1 systems office and equipment room, 1 TA Office, 3 instructor offices, 1 seminar room and a kitchen. Connecting hallways made communication and navigation simple and straightforward.

Department facility 2002-2015

Figure 2: the Department of Computer Science floor plan in 2015 shows we are up to 15 faculty offices, down to 7 research labs, the systems office and equipment room separated into 2, no TA Offices, up to 8 instructor offices, 2 seminar rooms, a kitchen, and an office for the department administrator. Connecting hallways have all been eliminated to accommodate growth.

Spring 2015

2015 Projects	Opinion of Capstone Program*	Willing to sponsor next year	Final Grade from Sponsor	Comments - mine	Comments - Sponsor/Mentor
Adventium	Very Good	For Sure	A	First time with this company and everything went very well.	"This is an excellent opportunity for both the students and the industry sponsors." "It was a pleasure working with this team." "Each of them will be great contributors on software/hardware development teams in industry."
ATC	Very Good	**Likely**	A	Good project, team did not have a lot of networking background but picked up on it quickly and did a nice job	"It's always a lot of fun"
Bolder Thinking	Very Good	For Sure	A	Great job done by 3 of the 4 that started the project. One was pulled off and given an F for not contributing. Very nice UI on their application.	"The three remaining team members did a great job"
Capturis (NISC)	Very Good and Good	**For sure and Marginal**	D	Major communication issues - sponsor thought team was telling him things were great when they were not getting tasks completely finished	"I hate to do this (give a D), because I have had such a positive experience with NDSU's Capstone project in the past, but we are unable to leverage a single line of code that the team produced" "Dean has done a fantastic job of developing this program over the years. It is always a joy to work with him."
CNE	Good	For Sure	A-	Picked up from where some consortium work had been done earlier.	This "front end" work is getting close - "back end" work being done by consortium needs lots more work
Ericsson -(Sweden)	Very Good	**For Sure**	A-	Good job on a cool project - needed to learn lots of new things (communications, mobile development, etc.)	"I really enjoy these projects." "Love to do it again if line organization agrees and you'll have me!"
IBM	Good	For Sure	A	Somewhat different kind of project that was more finding and stringing together the right set of tools in the right way. (Mentors are old NDSU grads who worked on an IBM capstone team then were hired by IBM a few years ago.)	"Lack of Linux experience is a concern from multiple mentors that I see."
Inwerken (Germany)	Good	**Likely**	B-	OK project, team didn't get quite as far as I would have liked	"Considering the framework complexity they did a good job - overall."
John Deere	Very Good	**Depends on legal issues**	A	Very nice technical work, good teamwork	"Based on the final report-outs, I have been impressed with the projects and their execution."
Microsoft	Very Good	**For Sure (need to work through legal issues)**	A	One of the best capstone projects in the 10 years I have been doing them. Great team!	"Excellent work!"
NICTA (Australia)	Very Good	For sure	A+	First project with Australia - things went very well	"I really enjoyed the process and working with the team."
Rockwell	Very Good	**For Sure**	A-	Another good project with a longtime sponsor	"thank you for having us again this year and I enjoy participating!"
UGPTI	Very Good	**For Sure**	A+	Probably the best project yet for UGPTI after about ten others.	"It is a great program for both the students and customers!"
Valley Express	Good	**Likely**	A	Good project, team made significant progress	"Everyone was great to work with and I can't think of anything they could have done better."
	10 Very Good 5 Good	10 For Sure 3 likely 1 marginal 1 depends on legal			

Spring 2014

2014 Projects	Opinion of Capstone Program*	Willing to sponsor next year	Final Grade from Sponsor	Comments - mine	Comments - Sponsor/Mentor
Bolder Thinking	Very Good	For Sure	A	There were some major internal personnel issues within the student team. Most members worked through this very well. Two team members were asked to stay on working on the project.	"great work and easy to work with", "there was a clear issue with one of the group members, dispite this the other group members work together smashingly"
CNE	blank	For Sure	B	New sponsor - ties to consortium	"I would like to be part o fthis process again. I think with the lessons that I have personally learned I would be better at helping a group get a better final product."
FBS	Very Good	Likely	A	The project definition changed fairly early on in the semester from writing a content management system to integrating an exiting one into the FBS system.	"They met all the objectives of the project" "Excellent communication throughout the project" "The program is run very well."
IBM	Very Good	Likely	A	Complex open software project that also required learning the IBM OpenStack	"Mentoring for this class has become a high light of my job responsibilities.", "I think what they have done will be extremely useful for many community members and I'm excited to see how it progresses."
Inwerken	Very Good	For Sure	A	This company is located in Hannover, Germany. This team had a problem student that was removed from the team part way through the semester.	"We have been really happy with the group. They performed well and it was great to work with them." "we were worried a little bit about the performance but after (person X) left the group the team bulding moved forward and also the overall team performance."
John Deere	Good	Likely	A	A very good project at the end, however, there was another problem student on this team that had to be removed from the team part way through the semester.	"They produced a better product than we anticipated from them." "Our requirements and definition were somewhat loose, students responded well to changes and members limited scope creep when possible."
Microsoft	Very Good	For Sure	A	Continuation of last year's project	Project may become commercial next year (Bison Tracker) so will have a new project
NISC	Very Good	For Sure	A	Another good project	"The finished project was exactly what David (the NISC Director) was envisioning and he was very pleased with the end result."
Rockwell Collins	Very Good	For Sure	A-	Very tough technically challenging project - complex airplane simulator, communications software, tablet application were all part of this project	"This project was particularly challenging as the team was both integrating with a tool they hadn't used or seen before but also creating new code of their own to interface with it."
UGPTI	Very Good	Likely	D	Students worked hard but had a major miscommunication that never got fully resolved	"huge miscommunication developed", "we were unaware that they misunderstood that the data was to be imported into a database and then repting would be done within the database", "they were dedicated"
Valley Express	Very Good	For Sure	A	New sponsor - ties to consortium	"good program with good people top to bottom", "everyone involved has been so dedicated and cooperative in providing such a viable solution"
	9 Very Good 1 Good 1 Blank	7 For Sure 4 Likely			

The capstone course has been a continued success. Many companies repeat. Waiting lists are not uncommon.

	Students Opinion of Program Value				
	Very Good	Good	Marginal	Poor	No Comment
2005	8	11	2	1	
2006	13	5	1	1	
2007	20	6			
2008	30	6			
2009	25	6	1		
2010	33	18	4	1	1
2011	27	10			
2012	33	17	3		
2013	22	8	1	1	
2014	24	20	5		1
2015	22	10	2	1	

	Sponsors Opinion of Capstone Program				
	Very Good	Good	Marginal	Poor	No Comment
2005	13	6	1	1	2
2006	8	3			
2007	8	3			
2008	9	5			
2009	5	5			
2010	7	5			
2011	11	2			
2012	13	3			
2013	10				1
2014	9	1			1
2015	10	5			

*Student Rating of Instruction (past two years)

FALL, 2014 and SPRING 2015

Questions	VG	G	IB	P	VP	OMIT	DEPARTMENT LEVEL Mean	S.D.	#R
100 TO 200 LEVEL									
1. Your satisfaction with the instruction in this course.	36.7	41.5	14.8	5.6	1.3	0.3	4.150	0.900	1999
2. The instructor as a teacher.	39.1	38.8	14.7	5.7	1.3	0.5	4.208	0.901	1997
3. The ability of the instructor to communicate effectively	36.9	36.4	16.5	7.6	2.3	0.3	4.093	0.966	2000
4. The quality of this course	34.1	40.1	18.3	5.3	1.6	0.5	4.098	0.908	1997
5. The fairness of procedures for grading this course.	52.2	37.1	8.2	1.5	0.6	0.5	4.414	0.753	1996
6. Your understanding of the course content.	34.5	43.9	16.1	4.1	0.8	0.5	4.135	0.818	1995
Questions	**SA**	**A**	**N**	**D**	**SD**	**OMIT**	**DEPARTMENT LEVEL Mean**	**S.D.**	**#R**
7. Instructor created an atmosphere conducive to learning.	36.1	44.5	15.0	3.2	0.8	0.3	4.210	0.794	1998
8. Instructor provided well-defined course objective.	38.1	46.2	12.0	2.9	0.5	0.3	4.231	0.790	1996
9. Instructor provided content/materials clear/organized	38.4	43.4	12.7	4.1	0.9	0.4	4.177	0.868	1995
10. I understood how grades were assigned in this course.	46.1	39.2	9.5	3.8	1.0	0.4	4.312	0.818	1995
11. I met/exceeded the course objectives for this course	38.1	45.6	12.9	2.7	0.4	0.4	4.216	0.757	1994
12. Instructor was available to assist students outside class	44.4	38.3	13.1	3.4	0.5	0.3	4.264	0.811	1996
13. Instructor provided feedback in a timely manner.	42.4	40.2	11.1	4.2	1.6	0.4	4.214	0.875	1997
14. Instructor provided relevant feedback that helped me learn	37.9	40.1	15.8	4.8	1.2	0.4	4.139	0.881	1995
15. Instructor set + maintained high standards for students	35.7	44.8	15.9	2.1	1.0	0.4	4.151	0.799	1997
16. Physical environment was conducive to learning	38.4	44.0	12.9	2.8	1.5	0.4	4.181	0.814	1995

Questions	VG	G	IB	P	VP	OMIT	DEPARTMENT LEVEL Mean	S.D.	#R
300 TO 400 LEVEL									
1. Your satisfaction with the instruction in this course.	39.9	43.3	12.2	2.7	1.9	0.0	4.150	0.900	1999
2. The instructor as a teacher.	47.6	36.9	11.0	3.2	1.2	0.1	4.208	0.901	1997
3. The ability of the instructor to communicate effectively	39.4	40.5	15.3	3.3	1.4	0.0	4.093	1.011	2000
4. The quality of this course	37.9	44.3	12.7	3.6	1.4	0.0	4.098	0.908	1997
5. The fairness of procedures for grading this course.	52.4	39.5	5.8	0.9	1.3	0.1	4.414	0.753	1996
6. Your understanding of the course content.	34.3	49.5	13.3	2.2	0.6	0.1	4.135	0.818	1995
Questions	**SA**	**A**	**N**	**D**	**SD**	**OMIT**	**DEPARTMENT LEVEL Mean**	**S.D.**	**#R**
7. Instructor created an atmosphere	37.2	50.7	9.4	1.9	0.4	0.4	4.210	0.794	1998

conducive to learning.									
8. Instructor provided well-defined course objective.	38.1	48.3	9.4	2.6	0.9	0.7	4.231	0.790	1996
9. Instructor provided content/materials clear/organized	37.6	44.4	12.3	3.2	1.7	0.7	4.177	0.868	1995
10. I understood how grades were assigned in this course.	46.5	43.6	6.9	0.9	1.6	0.6	4.312	0.849	1995
11. I met/exceeded the course objectives for this course	34.9	51.4	11.7	1.2	0.3	0.6	4.216	0.757	1994
12. Instructor was available to assist students outside class	41.2	44.3	11.9	1.3	0.7	0.6	4.264	0.811	1996
13. Instructor provided feedback in a timely manner.	37.5	49.5	8.5	2.2	1.9	0.4	4.214	0.875	1997
14. Instructor provided relevant feedback that helped me learn	34.0	49.3	11.9	2.6	1.6	0.6	4.139	0.881	1995
15. Instructor set + maintained high standards for students	30.8	52.4	14.2	1.6	0.6	0.4	4.151	0.799	1997
16. Physical environment was conducive to learning	33.0	49.8	14.3	1.7	0.7	0.4	4.181	0.814	1995

Questions	**VG**	**G**	**IB**	**P**	**VP**	**OMIT**	**DEPARTMENT LEVEL**		
							Mean	**S.D.**	**#R**
600 TO 700 LEVEL									
1. Your satisfaction with the instruction in this course.	59.2	29.4	5.7	1.3	0.9	3.5	4.150	0.900	1999
2. The instructor as a teacher.	64.9	26.8	3.5	0.9	0.9	3.1	4.208	0.901	1997
3. The ability of the instructor to communicate effectively	61.8	25.9	7.0	1.8	0.4	3.1	4.093	0.966	2000
4. The quality of this course	53.5	35.5	6.6	0.4	0.9	3.1	4.098	0.908	1997
5. The fairness of procedures for grading this course.	61.0	29.4	3.9	0.9	1.3	3.5	4.414	0.753	1996
6. Your understanding of the course content.	45.6	43.9	5.7	0.9	0.4	3.5	4.135	0.818	1995
Questions	**SA**	**A**	**N**	**D**	**SD**	**OMIT**	**DEPARTMENT LEVEL**		
							Mean	**S.D.**	**#R**
7. Instructor created an atmosphere conducive to learning.	63.6	28.5	3.9	1.3	0.0	2.6	4.210	0.794	1998
8. Instructor provided well-defined course objective.	59.2	31.1	4.4	1.8	0.9	2.6	4.231	0.790	1996
9. Instructor provided content/materials clear/organized	57.5	31.1	4.4	3.1	1.3	2.6	4.177	0.868	1995
10. I understood how grades were assigned in this course.	58.3	30.7	7.0	0.0	0.9	3.1	4.312	0.818	1995
11. I met/exceeded the course objectives for this course	50.4	36.4	7.0	2.6	0.0	3.5	4.216	0.757	1994
12. Instructor was available to assist students outside class	57.0	32.9	5.3	0.4	1.3	3.1	4.264	0.811	1996
13. Instructor provided feedback in a timely manner.	58.3	30.3	4.8	2.6	1.3	2.6	4.214	0.875	1997
14. Instructor provided relevant feedback that helped me learn	55.7	32.0	5.7	2.2	1.3	3.1	4.139	0.881	1995
15. Instructor set + maintained high standards for students	50.4	37.3	7.0	0.9	1.8	2.6	4.151	0.799	1997
16. Physical environment was conducive to learning	53.1	36.4	6.6	0.4	0.0	3.5	4.181	0.814	1995

FALL, 2013 and SPRING 2014

Questions	VG	G	IB	P	VP	OMIT	DEPARTMENT LEVEL Mean	S.D.	#R
100 TO 200 LEVEL									
1. Your satisfaction with the instruction in this course.	29.1	44.5	17.4	6.6	2.2	0.3	4.114	0.905	1815
2. The instructor as a teacher.	32.5	40.4	17.8	6.8	2.4	0.2	4.151	0.929	1816
3. The ability of the instructor to communicate effectively	28.0	36.4	21.9	9.9	3.5	0.4	4.007	1.005	1812
4. The quality of this course	27.0	44.5	18.8	7.9	1.6	0.3	4.031	0.926	1816
5. The fairness of procedures for grading this course.	44.2	40.3	10.1	4.4	0.9	0.2	4.326	0.819	1815
6. Your understanding of the course content.	29.4	45.9	18.7	3.6	1.9	0.6	4.070	0.872	1813
Questions	**SA**	**A**	**N**	**D**	**SD**	**OMIT**	**DEPARTMENT LEVEL Mean**	**S.D.**	**#R**
7. Instructor created an atmosphere conducive to learning.	30.1	43.1	19.7	4.6	1.4	1.0	4.118	0.864	1381
8. Instructor provided well-defined course objective.	31.1	49.2	13.0	4.8	1.1	0.8	4.176	0.819	1383
9. Instructor provided content/materials clear/organized	30.6	44.8	16.6	6.0	0.9	1.0	4.118	0.884	1380
10. I understood how grades were assigned in this course.	40.0	43.6	10.9	3.5	1.1	0.9	4.118	0.884	1380
11. I met/exceeded the course objectives for this course	31.5	48.7	15.9	2.0	1.0	0.9	4.144	0.774	1379
12. Instructor was available to assist students outside class	39.2	39.4	16.6	2.9	1.0	0.8	4.223	0.813	1383
13. Instructor provided feedback in a timely manner.	33.6	43.3	16.4	4.6	1.1	0.9	4.156	0.833	1382
14. Instructor provided relevant feedback that helped me learn	30.7	41.4	19.9	5.2	1.9	0.8	4.061	0.884	1383
15. Instructor set + maintained high standards for students	27.5	46.9	20.7	3.4	0.7	0.8	4.066	0.791	1382
16. Physical environment was conducive to learning	30.5	43.3	18.8	4.9	1.6	1.0	4.047	0.877	1381

Questions	VG	G	IB	P	VP	OMIT	DEPARTMENT LEVEL Mean	S.D.	#R
300 TO 400 LEVEL									
1. Your satisfaction with the instruction in this course.	45.1	41.4	10.1	2.4	0.5	0.5	4.114	0.905	1815
2. The instructor as a teacher.	49.0	39.1	8.8	1.7	1.0	0.5	4.151	0.929	1816
3. The ability of the instructor to communicate effectively	42.4	40.6	12.6	3.0	0.5	0.8	4.007	1.005	1812
4. The quality of this course	38.0	41.8	14.8	4.2	0.8	0.3	4.031	0.926	1816
5. The fairness of procedures for grading this course.	53.2	37.0	6.9	1.9	0.5	0.5	4.326	0.819	1815
6. Your understanding of the course content.	35.9	45.3	14.1	3.0	1.3	0.3	4.037	0.892	1945
Questions	**SA**	**A**	**N**	**D**	**SD**	**OMIT**	**DEPARTMENT LEVEL Mean**	**S.D.**	**#R**
7. Instructor created an atmosphere conducive to learning.	44.2	42.4	9.6	1.8	0.5	1.5	4.118	0.864	1381
8. Instructor provided well-defined	45.7	42.1	9.1	1.5	0.0	1.5	4.176	0.819	1383

course objective.									
9. Instructor provided content/materials clear/organized	45.9	37.1	11.9	2.8	0.8	1.5	4.118	0.884	1380
10. I understood how grades were assigned in this course.	49.5	40.1	7.1	1.3	0.5	1.5	4.279	0.805	1382
11. I met/exceeded the course objectives for this course	33.8	50.3	12.2	2.3	0.0	1.5	4.144	0.774	1379
12. Instructor was available to assist students outside class	42.9	45.2	8.9	1.5	0.0	1.5	4.223	0.813	1383
13. Instructor provided feedback in a timely manner.	39.3	49.7	8.4	0.8	0.3	1.5	4.156	0.833	1382
14. Instructor provided relevant feedback that helped me learn	34.5	49.7	12.4	2.0	0.0	1.5	4.061	0.884	1383
15. Instructor set + maintained high standards for students	29.9	51.8	16.0	0.8	0.0	1.5	4.066	0.791	1382
16. Physical environment was conducive to learning	31.2	45.4	18.0	3.6	0.3	1.5	4.037	0.892	1945

Questions	**VG**	**G**	**IB**	**P**	**VP**	**OMIT**	**DEPARTMENT LEVEL** **Mean**	**S.D.**	**#R**
600 TO 700 LEVEL									
1. Your satisfaction with the instruction in this course.	60.6	32.7	5.3	0.4	0.9	0.0	4.114	0.905	1815
2. The instructor as a teacher.	67.3	27.0	4.9	0.0	0.9	0.0	4.151	0.929	1816
3. The ability of the instructor to communicate effectively	66.4	24.3	8.0	0.0	1.3	0.0	4.007	1.005	1812
4. The quality of this course	60.6	29.2	8.4	0.9	0.9	0.0	4.031	0.926	1816
5. The fairness of procedures for grading this course.	64.2	27.4	6.6	0.4	0.9	0.4	4.326	0.819	1815
6. Your understanding of the course content.	46.9	44.2	7.5	0.0	1.3	0.0	4.070	0.872	1813
Questions	**SA**	**A**	**N**	**D**	**SD**	**OMIT**	**DEPARTMENT LEVEL** **Mean**	**S.D.**	**#R**
7. Instructor created an atmosphere conducive to learning.	63.0	33.6	2.5	0.0	0.0	0.0	4.118	0.864	1381
8. Instructor provided well-defined course objective.	60.5	34.5	4.2	0.8	0.0	0.0	4.176	0.819	1383
9. Instructor provided content/materials clear/organized	66.4	24.4	7.6	0.0	0.8	0.8	4.118	0.884	1380
10. I understood how grades were assigned in this course.	61.3	36.1	1.7	0.8	0.0	0.0	4.279	0.805	1382
11. I met/exceeded the course objectives for this course	52.1	41.2	4.2	0.0	0.0	2.5	4.144	0.774	1379
12. Instructor was available to assist students outside class	60.5	35.3	4.2	0.0	0.0	0.0	4.223	0.813	1383
13. Instructor provided feedback in a timely manner.	60.5	32.8	5.0	1.7	0.0	0.0	4.156	0.833	1382
14. Instructor provided relevant feedback that helped me learn	58.8	36.1	4.2	0.0	0.8	0.0	4.061	0.884	1383
15. Instructor set + maintained high standards for students	57.1	35.3	6.7	0.0	0.0	0.8	4.066	0.791	1382
16. Physical environment was conducive to learning	59.7	38.7	1.7	0.0	0.0	0.0	4.047	0.877	1381

Near-Final Draft of
Department of Computer Science
And Operations Research
Assessment Manual
Version: August 19, 2015

Contents

1. Service Course Assessment
2. Formative Assessments
3. B.A. and B.S. Program Outcomes
4. Performance Criteria
5. Old Curriculum: Table of Courses Satisfying Objectives
6. New Curriculum: Table of Courses Satisfying Objectives
7. Rating Rubrics for Performance Criteria
8. Graduate Degrees
    a. Program Objectives
    b. Mapping Objectives to Courses
    c. Rubrics for Graduate Assessment

Service Course Assessment

The NDSU General Education Committee has published seven objectives for General Education courses. These objectives are:

1. Communicate effectively in a variety of contexts and formats;
2. Locate and use information for making appropriate personal and professional decisions;
3. Comprehend the concepts and perspectives needed to function in national and international societies;
4. Comprehend intra-personal and interpersonal dynamics;
5. Comprehend concepts and methods of inquiry in science and technology, and their applications for society;
6. Integrate knowledge and ideas in a coherent and meaningful manner;
7. Comprehend the need for lifelong learning.

Here is a mapping of our General Education courses to these objectives:

Course	Obj. 1	Obj. 2	Obj. 3	Obj. 4	Obj. 5	Obj. 6	Obj. 7
CSci 114					X	X	
CSci 116					X	X	X
CSci 122					X	X	
CSci 125					X	X	
CSci 155					X	X	
CSci 159					X	X	X

Under Outcome 5, our General Education courses concentrate on the following methods of inquiry from the General Education Committee rubric:

1. Setup a study experiment to answer a question;
2. Analyze data/evidence to answer a question or support/or not the hypothesis;
3. Use methods of inquiry to solve a problem.

For Applications for Society, we concentrate on:

2. Apply science/technology to a problem;

4. Use methods of inquiry to address a problem.

Under Outcome 6, we concentrate on this part of Integrate knowledge and ideas in a coherent manner:

1. Identify and organize information relevant to a question or issue.

For Integrate knowledge and ideas in a meaningful manner, we concentrate on:

1. Identify significant patters from information relevant to a question or issue;
4. Integrate information to gain new insights relevant to a question or issue;
5. Integrate perspectives and points of view to gain new insights relevant to a question or issue.

For Outcome 7, we concentrate on:

1. Identify and explain the need for lifelong learning;

2. Analyze the need for lifelong learning from the perspective of rapidly changing knowledge;

3. Examine the need for lifelong learning in order to understand how contemporary knowledge can be translated into new contexts, or how it may not be valid in the future;

4. Evaluate the perspective of rapidly changing knowledge to understand the need for lifelong learning.

For our assessment of our General Education courses, we select a subset of Objective aspects and examine samples of student work. We assess each sample of student work using the following rubric:

Score 0	Score 1	Score 2	Score 3	Score 4	Score *5*
The work shows no use of the Objective aspect	The objective aspect is used incorrectly	The objective aspect use is incomplete	The objective aspect use mimics uses taught	The work uses the objective aspect in their own way	The work shows significant insight into the objective

We expect students to score 2 to 4 on our assessment. Over time, we hope to see fewer 2's and more 4's and 5's. Several scores of 0 or 1 indicate that we might need to make significant changes to the course.

Formative Assessments

Computer Science Department

Version: February, 2013

All sections of all courses are expected to perform at least one formative assessment during each semester. This assessment is to assist you in adjusting your course to improve student learning. The assessment should not have any impact on student grades, but you might wish to give a small amount of extra credit to ensure that students take it seriously.

Most formative assessments should be done during class time. They should be limited to no more than five to ten minutes. Usually, you should discuss the results in summary form and what you intend to change as a result with your students at the next class meeting.

The Department suggests that you perform one formative assessment during approximately the fifth week of the semester and another during approximately the tenth week. However, if you notice student difficulties, you may wish to perform these assessments earlier in the semester.

The results are for you. You do not need to share them with anyone else, but you should pay attention to them. If done properly, these assessments should be valuable in helping you improve your teaching and in improving student learning. If you are teaching a service course, your supervisor should discuss your formative assessment results with you soon after the assessment is done.

One excellent source of techniques that can be used for formative assessment is http://pages.uoregon.edu/tep/resources/newteach/fifty_cats.pdf . Numbers 5, 6, and 7 are particularly recommended for your use.

B.A. and B.S. Degrees

Program Outcomes

From Spring, 2014

In s series of meetings during the spring semester, 2014, the Department decided to adopt the CULE outcomes for our B.A. and B.S. degrees. As explained in the May, 2014 Department Assessment Report, we carefully determined which course would fit under each CULE outcome. Those assignments are presented in the following table:

Outcome	List of Relevant Undergraduate Courses
Communication	CSci 160, 161, 213, 313, 366, 371, 413, 415, 445, 477, 488, 489
Critical Thinking	CSci 160, 161, 213, 222, 312, 313, 336, 366, 372, 374, 415, 418, 426, 428, 467, 477 488, 489
Understanding and Applying Technology	CSci 213, 312, 313, 336, 366, 372, 374, 426, 428, 445, 453, 454, 474, 477, 479, 488
Understanding the Natural and Physical Worlds	CSci 415, 418, 426, 428, 458, 459, 469, 479, 488
Understanding Modern Societies	CSci 366, 371, 413, 458, 488, 489
Personal and Social Responsibility	CSci 160, 161, 213, 313, 413, 428, 445, 469, 473, 489

Although we have a completely new set of program outcomes, we did not feel we should replace our well established assessment procedures unless it was necessary. Therefore, in fall, 2014, we mapped the previous set of program outcomes to the new set. This mapping is provided in the following table:

New (CULE) Program Outcome	Corresponding Previous Program Outcomes
Communications	4 (Work in Teams), 6 (Communicate Effectively)
Critical Thinking	2 (Analyze Problems), 3 (Develop Solutions), 10 (Understand Tradeoffs
Understanding and Applying Technology	1 (Apply Knowledge), 9 (Use Current Methods and Tools), 11 (Build Software Systems)
Understanding the Natural and Physical Worlds	2 (analyze Problems, 10 (Understand Tradeoffs)
Understanding Modern Societies	5 (Understand Issues), 7 (Analyze Impact)
Personal and Social, Responsibility	5 (Understand Issues), 7 (Analyze Impact), 8 (Continue Development)

All of the previous program outcomes are presented on the next page with additional explanations.

We plan to use the rubrics for the previous program outcomes to assess the new program outcomes. This procedure will be tried during 2015-18 and evaluated for usefulness and effectiveness over that three year trial.

B.A. and B.S. Degrees
Program Outcomes
From Fall, 2005 through Fall, 2013
Adopted from ABET

1. Apply Knowledge

An ability to apply knowledge of computing and mathematics appropriate to the discipline;

2. Analyze Problems

An ability to analyze a problem, and identify and define the computing requirements appropriate to its solution;

3. Develop Solutions

An ability to design, implement, and evaluate a computer-based system, process, component, or program to meet desired needs;

4. Work in Teams

An ability to function effectively on teams to accomplish a common goal;

5. Understand Issues

An understanding of professional, ethical, legal, security, and social issues and responsibilities;

6. Communicate Effectively

An ability to communicate effectively with a range of audiences;

7. Analyze Impact

An ability to analyze the local and global impact of computing on individuals, organizations, and society;

8. Continue Development

Recognition of the need for, and an ability to engage in, continuing professional development;

9. Use Current Methods and Tools

An ability to use current techniques, skills, and tools necessary for computing practices;

10. Understand Tradeoffs

An ability to apply mathematical foundations, algorithmic principles, and computer science theory in the modeling and design of computer-based systems in a way that demonstrates comprehension of the tradeoffs involved in design choices;

11. Build Software Systems

An ability to apply design and development principles in the construction of software systems of varying complexity.

Performance Criteria

(Activities to Exercise the Program Objectives)

December, 2008

Program Outcome	Performance Criterion
A. Apply Knowledge	1. Solve problems using knowledge presented in the course; 2. Use course knowledge to solve test questions; 3. Improve a presented solution to handle situations more effectively; 4. Extend a presented solution to handle additional situations.
B. Analyze Problems	1. Practice Requirements elicitation; 2. Given a description of a problem, determine the requirements of a solution; 3. Explain why a given problem should or should not be solved with software; 4. Practice Requirements Analysis.
C. Develop Solutions	1. Given a problem, design and implement a solution; 2. Critique a given design for software; 3. Carefully test and debug a computer application; 4. Use metrics to evaluate a software implementation.
D. Work in Teams	1. Discuss a problem or question in small groups during a class period and prepare a report of conclusions; 2. Divide a software development project among a team of students and complete it; 3. Work as a class to solve a problem cooperatively; 4. Work in small teams to solve a set of problems during a week or longer; 5. Work in small teams to explore and evaluate alternative approaches to a situation.
E. Understand Issues	1. Discuss the issues involved in an ethical situation; 2. Prepare and present a short play or debate that explores the issues and viewpoints involved in an ethical situation;

	3. Evaluate and improve a provided paper discussing the different viewpoints in a situation; 4. Prepare a paper justifying a specific position on an ethical dilemma.
F. Communicate Effectively	1. Give an effective Powerpoint or other presentation in a class; 2. Write a user manual for an application; 3. Write a test plan for an application; 4. Participate in a group discussion; 5. Prepare and conduct structured interviews to develop software requirements.
G. Analyze Impact	1. Discuss the impact of a category of software on its potential users and others; 2. Prepare a report on how a specific category of users are or will be effected by a particular type of computer use; 3. Prepare a report on how a specific type of software should be configured to best serve a specific community of potential users; 4. Evaluate which of two or more types of software would best serve society and more specific stakeholders.
H. Continue Professional Development	1. Answer a set of questions concerning the work described in a journal or professional publication; 2. Keep a list of professional publications and/or web sites consulted during the semester. Require the list to have a minimum size and diversity; 3. Give students a topic and some exercises to solve in that topic. Require students to learn about the topic on their own; 4. Require students to learn and use a software tool or language on their own; 5. Have each student prepare a plan of what they expect to learn (knowledge and skills) on their own during the first five years after they graduate and how.
I. Use Current Methods and Tools	1. Give students exercises using current tools; 2. Ask students to discuss or report on the advantages of a specific modern tool or

	method; 3. Ask students to demonstrate how to use a new tool or method to other students; 4. Have groups of students research and evaluate a new tool or method.
J. Understand Trade-offs	1. Have students develop multiple solutions to a problem and compare those solutions to each other using algorithm analysis; 2. Give students two or more solutions to a particular problem and ask them to justify their selection of one of those solutions using algorithm analysis; 3. Have students critique a given design or solution with respect to tradeoffs using appropriate theory and analysis; 4. Have students critique UML models using appropriate theory.
K. Build Software Systems	1. Either singly or in teams develop a software application using design and development principles; 2. Have students critique a software design using best principles; 3. Have students critique the source code of an application using best principles; 4. Have students improve a software design or source code using best principles.

Old Curriculum

Table of Courses Satisfying Objectives

December, 2008 Version

The old curriculum was updated for students entering in Fall, 2011. The old curriculum is still available for students who have not yet completed their programs and entered before Fall, 2011. The old curriculum will no longer be available from Fall, 2015 on.

Codes Used: B (baseline competency established); P (practice opportunities); E (exit competency evaluated).

Required Courses:

Course	Oa	Ob	Oc	Od	Oe	Of	Og	Oh	Oi	Oj	Ok
160		B	B		B	B					
161		P	P					B	B		
222		P	P					P		P	
335	P							P	P		
336	P							P	P		
366			P	B					P	B	P
372	B					P	B		P		
373		P	P								
374		P			P		P			P	
445	E			E		E			E		E
467	P	P				P			P		
474	P		P						P	E	
475	P		E	P		P					
468	P	E	E	P			P				P
489					E	E	E	E	E		

Elective Courses (3 must be taken):

Course	Oa	Ob	Oc	Od	Oe	Of	Og	Oh	Oi	Oj	Ok
413		P	P	P		P			P		
418	P	P					p			P	
426	P	P	P		P			P			
453		P						P		P	P
454		P						P	P		P
458	P		P			P			P		
459				P				P		P	P
469					P		P	P	P		
476				P		P	P			P	
477					P	P			P		P
479			P		P	P		P			
488	P		P	P		P					P

New Curriculum

Table of Courses Satisfying Objectives
April, 2013 Version

This curriculum is required of students entering North Dakota State University in Fall, 2011 or later. The old curriculum is still available for students who entered earlier, but will cease in Fall, 2014.

Codes Used: B (baseline competency established); P (practice opportunities); E (exit competency evaluated).

Required Courses:

Course	Oa	Ob	Oc	Od	Oe	Of	Og	Oh	Oi	Oj	Ok
160		B	B		B	B					
161		P	P					B	B		
213	B	P	P			P			P	B	B
222		P	P					P		P	
313		P	P	B		P				P	P
336	P							P	P		
366			P	P					P	P	P
372	P					P	B		P		
374		P			P		P			P	
415		P	P		P	P	P			P	
445	E			E		E			E		E
467	P	P				P			P		
474	P		P						P	E	
468	P	E	E	P			P				P
489					E	E	E	E	E		

Elective Courses (3 must be taken):

Course	Oa	Ob	Oc	Od	Oe	Of	Og	Oh	Oi	Oj	Ok
371	P		P	P				P		P	P
413		P	P	P		P			P		
418	P	P					p			P	
426	P	P	P		P			P			
436	P	P	P		P		P	P	P	P	
453		P						P		P	P
454		P						P	P		P
458	P		P			P			P		
459				P				P		P	P
469					P		P	P	P		
473	P	P	P	P	P	P	P	P			
476				P		P	P			P	
477					P	P			P		P
479			P		P	P		P			
488	P		P	P		P					P

Rating Rubrics

For Performance Criteria

February, 2009

The performance criteria are more fully described in another document.

If your class is large, assessing a randomly selected sample of ten student submissions is fine. All student submissions must be graded, however. The grades must be sufficiently important in the course grade to motivate students to do their best work.

In most cases, a 3 should indicate reasonable performance. A 5 or 0 should be unusual (less than 10% of scores). When a performance criterion is used to introduce a skill, however, scores of 0 to 3 should be expected with an occasional 4 or 5.

Performance Criterion	0 Score	1 Score	2 Score	3	4	5
A. 1.Solve problem	Student does not seem to have understood the problem	Student understands the problem, but has made little progress towards a solution	Student has applied suitable methods, but has made significant errors	Student has applied suitable methods with minor mistakes	Student has successfully solved the problems using suitable methods.	Student has solved the problems and gone beyond the solution to describe its significance or limitations.
2. Solve test question(s)	Student has addressed the wrong question(s) or no question at all.	Student understands what is being asked, but has made little progress towards a solution.	Student has started the solution correctly, but either has made significant errors or has not gotten very far.	Student has made minor mistakes or has not finished the solution.	Student has correctly answered the question completely.	Student has gone beyond the correct answer to place that answer in some relevant context.
3. Improve solution	Student does not understand the problem and/or the presented solution	Student seems to understand the provided materials, but is unable to make progress on improvement	Student makes only minor progress on improvements.	Student makes significant progress on improvements, but is unable to complete.	Student completes the requested improvements.	Student makes the requested improvements and places the solution in some relevant context.
4. Extend solution	Student does not understand presented materials	Student understands presented materials, but is unable to progress on extensions	Student makes only minor progress on requested extensions	Student makes substantial progress, but does not complete the extensions	Student completes the extensions.	Student completes the extensions and places the solution in a relevant context.
B. 1. Elicitation	Student does not understand how to do elicitation	Student consults only himself or herself	Student consults only one or two potential users	Student uses only one elicitation method, but does a good job with that method	Student uses more than one method, but makes minor mistakes.	Student uses more than one method and gathers a good set of requirements
2. Determine requirements	Student does not seem to understand the problem	Student makes significant mistakes	Student makes minor mistakes	Student determines the correct requirements, but does not organize them	Student determines correct requirements and organizes them well.	Student goes beyond determining requirements to determine some of their

				well.		implications.
3. Solve with software	Student does not seem to understand the problem.	Student makes only preliminary progress towards a solution.	Student makes significant progress towards a solution, but does not use best practices.	Student makes significant progress and uses best practices.	Student produces an effective solution.	Student produces an effective solution using best practices.
4. Practice Analysis	Student does not seem to understand the problem.	Analysis is partial and trivial.	Analysis is incomplete, but shows substantial progress.	Analysis is effective, but poorly presented or does not use best practices.	Analysis is effective and uses best practices.	Analysis goes beyond what is expected to provide unexpected insight.
C. 1. Design and implement	Student is unable to make any progress on design or implementation	Student makes minor progress on either the design or the implementation	Student makes significant progress on the design and the implementation.	Design and implementation are largely complete, but the design does not match the implementation	Design and implementation are complete and match.	Some analysis is done on either the complete design or the complete implementation.
2. Critique design	Student misunderstand the design	Critique leaves most of the design unaddressed.	Critique addresses most of the design, but is superficial and/or inaccurate.	Good critique of the design with only minor errors or omissions.	Complete critique of the design according to best practices.	Complete critique together with useful suggestions for design improvements.
3. Test and debug	Student is unable to get any tests to work.	Student does one test case.	Student does several test cases, but does not debug correctly.	Student does several test cases according to a reasonable methodology and debugs successfully.	Student adequately tests and debugs program according to best practices.	Student adequately tests and debugs program plus provides suggestions of development practices that could have reduced errors.
4. Use metrics	Student does not seem to understand use of metrics.	Metrics not used appropriately.	Metrics used correctly, but no conclusions or wrong conclusions made.	Metrics used correctly and some correct conclusions made.	Metrics used correctly and appropriate conclusions drawn.	Metrics used correctly, appropriate conclusions drawn, and some analysis provided of why the calculated results occurred.
D. 1. Discuss	Students do not work together.	Students work together, but do not accomplish very much.	Student team has accomplishments, but is dominated by one or two students.	Student team has full participation, but does not consider all the aspects expected.	Student team has full participation and completes the task assigned well, but the report is disorganized.	Student team has full participation and produces a well-organized report.
2. Divide and complete	Students do not work together.	Students work together, but do not accomplish very much.	Student team has accomplishments primarily due to one or two members.	Team has full participation, but does not complete the project.	Team has full participation and completes the project successfully.	Team has full participation, completes the project successfully, and provides a useful analysis of their own teamwork.
3. Work cooperatively	Students seem unable to cooperate.	Students work together, but do not accomplish very much.	One or two team members dominate the accomplishments.	Team has full participation, but does not solve the entire problem or makes undiscovered mistakes.	Team has full participation and solves the problem.	Team has full participation, solves the problem correctly, and provides useful analysis of their own teamwork.
4.Work over time	Students do	Students work	One or two	Team has full	Team has full	Team has full

	not work together.	together, but do not accomplish very much.	members dominate the team's accomplishments.	participation, but does not complete all the problems correctly.	participation and completes all the problems correctly.	participation, completes all the problems correctly, and provides useful analysis of their own teamwork.
5. Evaluate alternatives	Students do not work together.	Students work together, but accomplish little.	One or two members dominate.	Team has full participation, but does not effectively consider all alternatives.	Team has full participation and reasonably evaluates all alternatives.	Team completes assignment with full participation and goes beyond the assignment in some useful way.
E. 1. Discuss	Student does not seem to understand the situation	Student can come up with only one issue for this situation and cannot discuss important ramifications of that issue.	Student can come up with one issue and discuss its implications.	Student can come up with more than one issue, but provides only superficial discussion.	Student can come up with more than one issue and discusses them well.	Student can come up with more than one issue, discuss them well, and provide a reasonable procedure for resolving the issues.
2. Play or debate	Student(s) does not seem to understand the situation.	Student(s) uses only one issue.	Important issues are presented, but important implications are not.	Important issues and important implications are presented, but not fully explored.	Situation is fully explored.	Situation is fully explored and a reasonable procedure for resolving the issues is given.
3. Evaluate and improve paper	Student does not seem to understand the important points of the paper at all.	Student understands many, but not all of the important points.	Student fully understands the paper.	Student understands the paper, but provides superficial or incorrect improvements.	Student provides useful improvements.	Student provides useful improvements and compares them to what is already in the paper.
4. Prepare a paper	Student does not seem to understand the ethical dilemma.	Student understands the dilemma, but does not present a position.	Student presents a position with some support for that position.	Student presents a position and evaluates that position with respect to other positions, but is not convincing or the paper is poorly organized.	Student does an excellent job of presenting a position and evaluating it with respect to other positions.	Student presents a position and justifies it well. Student also generalizes the ethical dilemma and describes how their position and other positions would be evaluated in the more general dilemma.
F. 1. Give presentation	Presentation is not on topic.	Presentation is not close to complete.	Presentation is complete, but poorly organized or presented.	Presentation is complete, well-organized and presented, but gives no analysis or evaluation.	Presentation is complete, well-organized and provides well-known analysis or evaluation.	Presentation is complete, well-organized, and provides innovative analysis or evaluation.
2. Write user manual	Manual explains nothing.	Manual explains some use, but leaves out major functionality.	Manual is not complete, but poorly organized or written.	Manual is complete, but poorly organized or written.	Manual is complete and well-written.	Manual is complete, well-written, and includes useful materials for a variety of potential users.
3. Write test plan	Plan contains no tests.	Plan contains a few tests, but coverage is	Plan contains a reasonable set of tests according to	Plan contains a reasonable set of tests and is	Plan organizes a reasonable set of tests according to	Plan provides indications of the errors that

		inconsistent and incomplete.	some methodology, but is poorly written or organized.	well-written and organized.	priority based on some reasonable criterion.	can be found by each test.
4. Group discussion	Group does not stay on topic for very long.	Group deals with the topic at a very superficial level.	Group has a reasonable discussion of the superficial aspects that does not have full participation.	Group goes beyond the superficial, but is dominated by only a few members.	Group has a good discussion with nearly full participation.	Group has a good discussion that leads to useful conclusions with full participation.
5. Structured interviews	Student does not understand how to conduct an interview.	Only one or two interviews are conducted and they are not well-structured.	More than two interviews are conducted, but they are not well-structured.	More than two well-structured interviews are conducted, but the student does not formulate a reasonable set of requirements.	More than two well-structured interviews of a variety of potential stakeholders are conducted, but the resulting requirements are not effective or well-organized.	Nearly all important potential stakeholders are represented in the interviews and the resulting requirements are well-organized and effective.
G. 1. Discuss impact	Student's comments indicate a lack of understanding	Student does not contribute to the discussion, but does follow it.	Student's contributions are limited to agreeing with previous speakers	Student makes small contributions	Student makes significant contributions to the discussion.	Student provides analysis showing significant insight.
2. Prepare report	Student does not appear to understand important characteristics of the users and/or the software	Student describes the users and the software correctly, but not the impact.	Student misses many important impacts, but does correctly describe some impacts.	Student describes all the important impacts, but provides no analysis.	Student provides only minimal analysis.	Student provides significant analysis.
3. Configuration report	Student does not understand the assignment.	Student describes the configuration features of the software, but not how it should be configured for this community.	Student describes how the software should be configured, but makes significant mistakes in this description.	Student correctly describes how the software should be configured, but does not explain why.	Student provides some justifications for how the software should be configured.	Student provides justifications which demonstrate significant insight and analysis.
4. Evaluate alternatives	Report is limited to describing the alternatives, and makes mistakes doing so.	Report correctly describes the alternatives, but does not address the service to society or specific stakeholders.	Report asserts how each alternative would serve society and/or specific stakeholders, but does not justify the assertions.	Report provides incorrect or incomplete justifications.	Report provides adequate justifications and analysis.	Report provides excellent justifications and analysis.
H. 1. Answer questions	Response shows lack of understanding of the paper and/or the questions.	Response addresses all questions, but merely quotes the paper.	Response shows some understanding of some aspects of the paper.	Response shows understanding of all important aspects of the paper.	Response includes explanation, examples, or justifications beyond those in the paper for several of the questions.	Response provides significant insight into the work reported in the paper which is not already contained in the paper.
2. Keep list	List is not available.	List is too small.	List is not sufficiently diverse.	List meets minimum requirements for size and diversity.	List goes well beyond the minimum requirements for size and diversity.	List shows the student has searched for relevant other articles on at least some of the topics of articles

						on the list.
3. Learn topic	Student has not explored the topic.	Student has done only one of the exercises assigned.	Student has done some, but not all of the exercises assigned.	Student has completed all the exercises assigned.	Student has gone beyond the exercises assigned to learn more.	Student has completed all assigned exercises and has a plan to continue learning in this topic.
4. Learn tool or language	Student has not explored the tool or language.	Student cannot use the tool or language.	Student can use the tool or language, but not effectively.	Student can use the tool or language effectively.	Student understands when and how this tool or language should be used instead of other tools or languages they already knew.	Student understands why and how the limitations of this tool or language exist.
5. Development Plan	Student does not have a development plan.	Student's development plan is very vague.	Development plan has a good list of skills, but not a good idea of how they will learn those skills.	Development plan is good in some areas and not very good in others.	Development plan is very good.	Development plan includes why the student believes learning these skills will be important.
I. 1. Use current tools	Student cannot or does not use the appropriate tools.	Tools are used ineffectively.	Tools are used inefficiently.	Tools are used effectively and efficiently, but not everywhere they should be used.	Tools are used well.	Student has developed ways to combine these tools which were not intended by the tool developers, but which are effective.
2. Discuss or report on advantages	Report describes the tool, but does not list any advantages.	Report lists only one or two advantages.	Report only lists the advantages. It does not justify them.	Report lists advantages and their justifications from the Help file of the tool or tutorial on the method.	Report provides some advantages based on the student's own experience and not from the Help or tutorial.	Report provides advantages based on comparisons with other tools or methods with overlapping purposes. These advantages are not just from the tool or method documentation.
3. Demonstrate	Student is unable to demonstrate the tool.	Some parts of the demonstration do not use the tool or method correctly.	The demonstration is not well-organized.	The demonstration is well-organized, but incomplete.	The demonstration is well-organized and covers the major aspects of the tool or method.	The demonstration is well-organized and provides insight into why the tool or method works as it does.
4. Group evaluation	This student did not contribute significantly to the evaluation.	The evaluation is incomplete and has significant errors.	The evaluation is correct, but incomplete.	The evaluation is correct, nearly complete, but not very well-organized.	The evaluation is well-organized, correct, and complete.	The evaluation provides insight on why the tool or method operates as it does and/or some implications of this operation.
J. 1. Compare solutions	The student is unable to produce more than one solution.	The student produces some incorrect solutions to the problem.	The student's solutions are correct, but there is little comparison.	The comparisons are asserted without justification.	The comparisons are justified.	The comparisons provide insight into the problem itself.
2. Justify selection	Student does not make a	Student makes a choice, but does	The justification is incorrect.	The justification is	The justification is good.	The justification shows insight

	selection.	not justify it.		correct, but misses significant aspects.		into the nature of the problem and/or the solutions.
3. Critique design or solution	The student provides no critique.	The critique does not use tradeoffs correctly.	The critique does not use appropriate theory and analysis.	The critique uses appropriate theory and analysis, but incorrectly or incompletely.	The critique is accurate, complete, and uses appropriate theory and analysis correctly.	The critique justifies the use of the theory and/or analysis employed.
4. Critique UML models	The student provides no critique.	The critique does not use theory.	The critique uses theory, but incorrectly or uses the wrong theory.	The critique uses proper theory correctly, but is incomplete.	The critique is accurate, complete, and uses proper theory.	The critique explains how the critique could be addressed to reduce or eliminate the problems with the diagram.
K. 1. Develop using principles	The student or team is unable to complete the application.	The student or team completes the application, but does not document the design and/or development.	The student or team completes the application and documents, but does not use principles everywhere.	The team or student uses principles incorrectly in some places.	The team or student uses principles appropriately throughout.	The team or student justifies exceptions to principles and explains how the principles benefited the development effort.
2. Critique using principles	The student does not do a critique.	The critique is incorrect.	The critique is not based on appropriate principles.	The critique uses principles incorrectly in some places.	The principles are used appropriately throughout.	The student gives some insight into the value of using these principles.
3. Critique source code from principles	The student does not do a critique.	The critique is incorrect.	The critique is correct, but not based on best principles.	The critique uses best principles incorrectly in some places.	The best principles are used appropriately throughout.	The critique explains how the code could be improved to better satisfy the best principles.
4. Improve using principles	The student does not make any improvements.	The improvements are not correct.	The improvements are correct, but not based on best principles.	The best principles are used incorrectly in some places.	The best principles are used appropriately throughout.	The student explains how the best principles guided the improvements.

Program Objectives

Graduate Degrees

Version: February, 2013

A. Graduate Certificate in Software Engineering
    a. Students understand the present state of software development;
    b. Students understand the significant differences among informal projects and formal projects;
    c. Students can do each of the phases of software development;
    d. Students understand the advantages of an iterative development process compared to a sequential process.
B. Graduate Certificate in Electronic Commerce
    a. Students understand the impact of computers and computer networks on commerce;
    b. Students understand the present state of electronic commerce;
    c. Students can evaluate an electronic approach to commerce.
C. Master of Software Engineering (MSE)
    a. Students recognize good practices in software engineering;
    b. Students can read and understand the professional literature in software engineering;
    c. Students understand the present state of a significant area of software engineering;
    d. Students can develop useful applications using state of the art methods.
D. Master of Science in Computer Science
    a. Students understand the current status of a major area of computer science;
    b. Students can read and understand the academic literature in computer science;
    c. Students can evaluate specific research in computer science;
    d. Students can extend the state of the art in a specific area of computer science through their own research.
E. Master of Science in Software Engineering
    a. Students understand the present state of software engineering;
    b. Students can read and understand the academic literature in software engineering;
    c. Students can analyze specific research in software engineering;
    d. Students can extend the state of the art in a specific area of software engineering through their own research.
F. Ph.D. in Computer Science
    a. – d .Same first four as for M.S. in Computer Science;
    e. Students can present and defend research in a specific area of computer science to their colleagues;
G. Ph.D. in Software Engineering
    a. – d. Same first four as for M.S. in Software Engineering;

e. Students can present and defend research in software engineering to their colleagues.

Mapping Objectives to Courses
Version: March, 2013

Codes Used: B (Baseline Competency established and some practice provided); P (Practice to build competency); E (Exit from program competency level)

A. Graduate Certificate in Software Engineering

Course	Oa	Ob	Oc	Od
713	B	B	B	B
714	P	P		P
715	P	P	P	
716	P	P	P	P
717	P	P		P
718	P	P	P	P
790		E		E
Project	E		E	

B. Graduate Certificate in Electronic Commerce

Has been inactive for several years.

C. Master of Software Engineering

Course	Oa	Ob	Oc	Od
713	B	B	B	B
714	P		P	
715	P		P	P
716	P		P	P
717	P		P	P
718	P		P	P
724		E	P	P
746	P	P	P	
747	P	P	P	
765		E	P	P
793 (6 credits)	E		E	E

D. Master of Science in Computer Science

Course	Oa	Ob	Oc	Od
713	B	B	B	
724	P		P	
741	P	P	P	B
765		P	P	
790		P	P	
Other Courses (vary by student)	P	P	P	P
Comprehensive	P	P	P	
Final Examination on Thesis or Paper	E	E	E	E

E. Master of Science in Software Engineering

Course	Oa	Ob	Oc	Od
713	B	B	B	
716	P	P	P	
715 or 718	P	P	P	
765		P	P	
Other courses (vary by student)	P	P	P	P
790		P	P	
Comprehensive	P	P	P	
Final Examination on Thesis or Paper	E	E	E	E

F. Ph.D. in Computer Science

Course	Oa	Ob	Oc	Od	Oe
713	B	B	B		
724		P	P		
741	P	P	P		
765		P	P		
790	P	P	P		
Other courses (vary by student)	P	P	P	P	P
Preliminary Examination	P	P	P	P	P
Qualifier	P	P	P		

Examination					
Final Examination on Dissertation	E	E	E	E	E

G. Ph.D. in Software Engineering

H. Course	Oa	Ob	Oc	Od	Oe
713	B	B	B		
716	P	P	P		
715 or 718	P	P	P		
765		P	P		
790	P	P	P		
Other courses (vary by student)	P	P	P	P	P
Preliminary Examination	P	P	P	P	P
Qualifier Examination	P	P	P		
Final Examination on Dissertation	E	E	E	E	E

Rubrics for Graduate Assessment

Version: April. 2013

A. Course Oriented Assessment Rubrics

Baseline values are expected to be one or two, although there will be some higher scores, especially for students with previous undergraduate or graduate degrees or substantial professional experience. Practicing scores should be in the range 2 to 4 with rare, if any, 0's and a few 5's. Exit scores are expected to be 3 to 5 with a few lower scores when the student has been struggling.

Remember that which of these objectives are relevant for a particular student depends on which degree program that student is attempting. In any course, we are likely to have students from several degree programs.

Performance Criterion	Score 0	Score 1	Score 2	Score 3	Score 4	Score 5
Objective: Understand the present state	Does not associate development with a formal process	Confused about the current state	Misunderstand current development concerns	Can recite literature evaluations	Can explain current thinking in their own words	Can explain implications of current state
Objective: Differences among informal and formal projects	Thinks everything is informal	No idea when to do formal vs. informal	Can recite differences given by text and lectures	Can describe differences in their own words	Understands some implication of the differences	Can address some implications
Objective: Can do each phase of development	Does not understand the reason for phases	Skips or ignores some phases	Can recite the phases	Can follow instructions with guidance	Can follow instructions without guidance	Can innovate during some phases
Objective: Iterative versus sequential	Always sequential since it is simpler	Does not understand when each should be used	Can recite taught material	Can express in his or her own words	Understands the implication	Can address some implications
Objective: Recognize good practices	Does not understand what makes a practice good.	Cannot actually do the evaluation	Can list some good practices, but not use them	Uses good taught good practices	Can evaluate practices	Can improve a practice

Objective: Students understand the state of a CS area	Does not understand the question	Makes serious mistakes	Makes minor mistakes.	Can recite from taught material correctly	Can summarize relevant literature	Synthesizes his or her own answer
Objective: Develop useful applications	Cannot develop applications	Can do part of development, but not all	Result has significant problems	Result is good, but the process is bad	Good result and process for familiar application areas	Successfully adapts to unfamiliar application areas.

B. The Preliminary and Final Examinations

We expect the preliminary scores to range from 1 – 3 in most cases. The final scores should range from 3 – 5 in nearly all cases. For the preliminary, a 2 should indicate reasonable performance. For the final, a 4 should indicate reasonable performance. Notice, that a student might pass the preliminary or the final examination without reaching reasonable performance on these assessment measures. Further, a student can fail the preliminary or final even when their assessment is reasonable or better. There are aspects to the preliminary and final which are not captured by these assessment measures.

Performance Criterion	Score 0	Score 1	Score 2	Score 3	Score 4	Score 5
Objective: Students can present and defend research						
Student presents in a well-organized manner	No organization is apparent	Items are presented after first needed	Some needed items are not presented	Items not presented where needed	No problems with organization	Helps clarity
Student explains the research clearly	Student does not understand	Cannot be understood	No examples	Not enough examples	Clear	Superior
Student can answer questions on the research	Student does not understand the question	Cannot answer	Answer is wrong	Answer is incomplete	Answer is good	Answer provides insight beyond the work
Student proposes ways to improve or extend the research	Fails to understand the need	Incorrect means	Incomplete means	Student does not get the implications of extension	Student understands the implications	Student provides new insights
Objective: Students can read and understand the academic literature						
A student can understand what he or she reads	Student seems not to have read	Student misunderstands	Student can repeat the words	Student can summarize in own words	Student can relate this work to other work	Student has insights not present in the work

A student can use what he or she reads	Does not understand possible use	Can recite available uses	Can use in expected ways with some guidance	Can use in expected ways without guidance	Generalizes effectively	Combines with other work for effective use
A student can critique what they read	Does not understand	Can repeat critique in the work	Can critique with mistakes	Can critique successfully with guidance	Can critique without guidance	Critiques and addresses critique
A student can combine material from more than one source	Does not understand connections among works	Understands connections wrongly	Understands connections	Can make the connections with guidance	Can make connections without guidance	Can form insights from connections
Objective: Students can extend the state of the art in a specific area						
Student can apply existing research	Does not understand	Understands, but cannot apply	Can apply with guidance	Can apply without guidance	Looks for applications	Can characterize the domain
Student can apply the existing research to a new problem	Does not understand where the research is applied	Understands application incorrectly	Can apply to a new problem with guidance in problem selection and application	Can apply to a new problem with guidance in application	Can select and apply without guidance	Can modify the work to extend application
Student can extend the research to address at least one of its limitations	Does not understand the limitations	Partly understands limitations	Understands implications of limitations	Can overcome a limitation with substantial guidance	Can overcome with little guidance	Can overcome without guidance
Student can develop and use a new approach	Does not understand the need	Understands the need, but not how to deal with it	Can do with substantial guidance	Can do with minimal guidance	Can do without guidance	Can generalize the new approach or understand its limitations