TO : Professor Z.G. Gong. Head of DCIS - FST
FROM : Kam H. Vat, Department of CIS – FST
SUBJECT : Feedback on Program Outcomes and Program Criteria
DATE : October 8, 2010
CC : Professor C.L. Philip Chen, Dean of FST; Professor E.H. Wu, Associate Dean of FST

Thanks for holding the 2010SEP22-departmental meeting, requesting for feedback and suggestions for the DCIS Program Outcomes and Program Criteria in the occasion of our Faculty's (FST) initiative to apply for Program Accreditation for our various engineering programs first from HKIE and then from ABET starting with the year 2010/2011. On reflecting on the responses from our colleagues concerning the various changes rendered in the context of our DCIS's preparation for Program Accreditation, I have organized my thinking and feedback for your reference as follows:

Concerning the Program Outcomes (Old Curriculum – Software Engineering):

It is my observation that our Department of CIS, has earlier, through the efforts of Professor Gong and Dr. Derek Wong, defined the following program outcomes with the information obtained from our Dean, Professor Philip Chen. These outcomes are explicitly defined for all graduates of the Bachelor of Science in Computer Science program:

- (a) an ability to apply knowledge of computing, mathematics, science, and engineering;
- (b) an ability to design and conduct experiments, as well as to analyze and interpret data;
- (c) an ability to design, implement, and evaluate a computer-based system, process, component, or program to meet desired needs within realistic constraints such as economic, environmental, social, political, ethical, health and safety, manufacturability, and sustainability;
- (d) an ability to function effectively on multi-disciplinary teams;
- (e) an ability to analyze a problem, and identify, formulate and use the appropriate application requirements for obtaining its computing solution;
- (f) an understanding of professional, ethical, legal, security and social issues and responsibilities;
- (g) an ability to communicate effectively;
- (h) the broad education necessary to understand the impact of computing solutions in a global, economic, environmental, and societal context;
- (i) a recognition of the need for, and an ability to engage in life-long learning;
- (j) a knowledge of contemporary issues;
- (k) an ability to use the techniques, skills, and modern computer tools necessary for engineering practice;
- (l) an ability to use the computer/IT tools relevant to the disciplines along with an understanding of their processes and limitations.

Yet, since we are still officially running our Software Engineering program (considered as the old curriculum), I am wondering if the following set of program outcomes could better serve our purposes (SE2004, pp.15-16: http://sites.computer.org/ccse/SE2004Volume.pdf):

1) *Show mastery of the software engineering knowledge and skills, and professional issues necessary to begin practice as a software engineer.*

Students, through regular reinforcement and practice, need to gain confidence in their abilities as they progress through a software engineering program of study. In most instances, knowledge, as well as skills, is acquired through a staged approach with different levels being achieved as each academic term progresses. In addition, graduates need to gain an understanding and appreciation of professional issues related to ethics and professional conduct, economics, and the societal needs.

2) *Work as an individual and as part of a team to develop and deliver quality software artifacts.*

Students need to complete tasks that involve work as an individual, but also many other tasks that entail working with a group of individuals. For group work, students ought to be informed of the nature of groups and of group activities and/or roles as explicitly as possible. This must include an emphasis on the importance of such matters as a disciplined approach, the need to adhere to deadlines, communication, and individual as well as team performance evaluations.

3) *Recognize conflicting project objectives, finding acceptable compromises within limitations of cost, time, knowledge, existing systems, and organizations.*

Students should engage in exercises that expose them to conflicting, and even changing requirements. There should be a strong element of the real world present in such cases to ensure that the experience is realistic. Curriculum units should address these issues, with the aim of ensuring high quality requirements and a feasible software design.

4) *Design appropriate solutions in one or more application domains using software engineering approaches that integrate ethical, social, legal, and economic concerns.*

Throughout their study, students need to be exposed to a variety of appropriate approaches to engineering design in the general sense, and to specific problem solving in various kinds of applications domains for software. They need to be able to understand the strengths and the weaknesses of the various options available and the implications of the selection of appropriate approaches for a given situation. Their proposed design solutions must be made within the context of ethical, social, legal, security, and economic concerns.

5) *Demonstrate an understanding of and apply current theories, models, and techniques that provide a basis for problem identification and analysis, software design, development, implementation, verification, and documentation.*

The presence of the Capstone project, an important final activity at the end of a software engineering program of study, is of considerable importance in this regard. It offers students the opportunity to tackle a major project and demonstrate their ability to bring together topics from a variety of courses and apply them effectively. This mechanism allows students to demonstrate their appreciation of the broad range of software engineering topics and their ability to apply their skills to genuine effect. This should also include the ability to offer reflections on their achievements.

6) *Demonstrate an understanding and appreciation for the importance of negotiation, effective work habits, leadership, and good communication with stakeholders in a typical software development environment.*

   It is important to have within a program of study at least one major activity that involves having to produce a solution for a client. Software engineers must take the view that they have to produce software that is of genuine utility. Where possible, we should integrate within the program a period of industrial experience, as well as invited lectures from practicing software engineers, and even involvement in such matters as external software competitions. All this provides a richer experience and helps to create an environment that is supportive of the production of high quality software engineering graduates.

7) *Learn new models, techniques, and technologies as they emerge and appreciate the necessity of such continuing professional development.*

   By the time they come to the end of their program of study, students should be showing evidence of being a self-motivated life-long learner. Such a situation is achieved through a series of stages inserted at various places of a program of study. In later academic years, such as at the capstone stage, students should be ready and willing to learn new ideas. But again, students need to be exposed to best practice in this regard at earlier stages.

In fact, my understanding is that we are yet to establish justifiable rationales to switch our program name from the existing Software Engineering to the proposed Computer Science. It is understood that our original program in Software Engineering had been created in 1989 as if it looked very much like a Computer Science curriculum. Yet, it is our responsibility to continually adapt this program according to the needs of our society and those of our students, even though we have not done much in this important area in the past. Surely, we also need to cast our vote to determine if we really need to switch the program. My understanding is that we were somehow informed that we need to change the name of our undergraduate program earlier this year, without given a chance to cast our vote to determine whether or not we should proceed with the name change, given another chance to debate why to do the change. I believe that is an issue yet to be handled properly, because we are altogether held accountable for what decisions to make, by the University Council as well as by the local community as a whole.

Concerning the Program Criteria (Old Curriculum – Software Engineering):

It is observed that the program criteria document demonstrated in our 2010SEP22 departmental meeting include the following areas of interest, which could be considered as the body of disciplinary knowledge, suggested by Professor Gong and Dr. Derek Wong, on behalf of our Department of Computer and Information Science:

- Discrete Mathematics (DM)
- Probability and Statistics (PS)
- Linear Algebra (LA)
- Symbolic Logic (SL)
- Algorithmic Principles (AP)
- Computer Science (CS)
- Design and Development Principles (DD)
- Programming Language (PL)

Since we are expected to map our individual course content into related areas of interest using this set of program criteria, kindly allow me to wonder if it could be so much better, based on our current curriculum of Software Engineering, to adopt the following recommended program criteria from SE2004 (pp.20-35), referred to as the Software Engineering Education Knowledge (SEEK) areas in the international arena of Software Engineering education.

- Computing Essentials (CMP)
- Mathematical and Engineering Fundamentals (FND)
- Professional Practice (PRF)
- Software Modeling & Analysis (MAA)
- Software Design (DES)
- Software Verification & Validation (VAV)
- Software Evolution (EVL)
- Software Process (PRO)
- Software Quality (QUA)
- Software Management (MGT)

It should be noted that based on the SEEK organization, for each knowledge area we are informed of the specifics of the knowledge units, and topics in that area. For each knowledge unit, recommended contact hours are designated. For each topic, a Bloom taxonomy level (indicating what capability a graduate should possess) and the topic's relevance (indicating whether the topics is essential, desirable, or optional to the core) are designated. In other words, this is very valuable information for DCIS to tailor our program to meet ABET (or HKIE) accreditation requirements. So, kindly consult this important document of SE2004, for a detailed analysis of what we could benefit from it. I believe this is a responsibility we at DCIS must not save so as not to become remissful.

Concerning the Program Outcomes (New Curriculum – Computer Science):

If, after some careful study, there is a justifiable reason and need (after extensive consultation with students and staff, and industry or society) to set up another program in Computer Science, besides the existing Software Engineering program, we should do our best to make it a rigorous and competitive one, according to the guidelines of some world-famous bodies, such as IEEE/ACM's Computing Curricula 2001 (http://www.acm.org/education/curric_vols/cc2001.pdf) and Computer Science 2008 (http://www.acm.org/education/curricula/ComputerScience2008.pdf). It is a very healthy sign of growth and maturity in our department to resort to such international guidance, to craft our program outcomes as follows (CS2008, pp.20-22):

- Cognitive capabilities and skills relating to computer science
  - *Knowledge and understanding*: Demonstrate knowledge and understanding of essential facts, concepts, principles, and theories relating to computer science and software applications.
  - *Modeling*: Use such knowledge and understanding in the modeling and design of computer-based systems in a way that demonstrates comprehension of the tradeoff involved in design choices.
  - *Requirements*: Identify and analyze criteria and specifications appropriate to specific problems, and plan strategies for their solution.

- *Understanding the elements of computational thinking*: This includes recognizing its broad relevance in everyday life as well as its applicability within other domains, and being able to apply it in appropriate circumstances.
- *Critical evaluation and testing*: Analyze the extent to which a computer-based system meets the criteria defined for its current use and future development.
- *Methods and tools*: Deploy appropriate theory, practices, and tools for the specification, design, implementation, and maintenance as well as the evaluation of computer-based systems.
- *Professional responsibility*: Recognize and be guided by the social, professional, legal and ethical as well as cultural issues involved in the use of computer technology. Increasingly cultural issues are also relevant.

- Practical capabilities and skills relating to computer science
  - *Design and implementation*: Specify, design, and implement computer-based systems.
  - *Evaluation*: Evaluate systems in terms of general quality attributes and possible tradeoffs presented within the given problem.
  - *Information management*: Apply the principles of effective information management, information organization, and information-retrieval skills to information of various kinds, including text, images, sound, and video. This must include managing any security issues.
  - *Human-computer interaction*: Apply the principles of human-computer interaction to the evaluation and construction of a wide range of materials including user interfaces, web pages, multimedia systems and mobile systems.
  - *Risk assessment*: Identify any risks (and this includes any safety or security aspects) that may be involved in the operation of computing equipment within a given context.
  - *Tools*: Deploy effectively the tools used for the construction and documentation of software, with particular emphasis on understanding the whole process involved in using computers to solve practical problems. This should include tools for software control including version control and configuration management.
  - *Software reuse*: Be aware of the existence of publicly available software (such as APIs or open source materials) and engage effectively in open-source projects.
  - *Operation*: Operate computing equipment and software systems effectively.

- Additional transferable skills
  - *Communication*: Make succinct presentations to a range of audiences about technical problems and their solutions. This may involve face-to-face, written communication or electronic communication.
  - *Teamwork*: Be able to work effectively as a member of a development team.
  - *Numeracy*: Understand and explain the quantitative dimensions of a problem.
  - *Self-management*: Manage one's own learning and development, including time management and organizational skills.
  - *Professional development*: Keep abreast of current developments in the discipline to continue one's own professional development.
  - *Software reuse, open source issues*: Separate compilation.

Concerning the Program Criteria (New Curriculum – Computer Science):

It is observed that the program criteria document demonstrated in our 2010SEP22 departmental meeting include the following areas of interest, which could be considered as the body of

disciplinary knowledge, suggested by Professor Gong and Dr. Derek Wong, on behalf of our Department of Computer and Information Science:

- Discrete Mathematics (DM)
- Probability and Statistics (PS)
- Linear Algebra (LA)
- Symbolic Logic (SL)
- Algorithmic Principles (AP)
- Computer Science (CS)
- Design and Development Principles (DD)
- Programming Language (PL)

Since we are expected to map our individual course content into related areas of interest using this set of program criteria, kindly allow me to suggest that this mapping could better be done, for our new curriculum in Computer Science, based on the following knowledge areas (CS2008, pp.15-20), referred to as the Body of Knowledge for CS areas in the international arena of Computer Science education:

- Discrete Structures (DS)
- Programming Fundamentals (PF)
- Algorithms and Complexity (AL)
- Architecture and Organization (AR)
- Operating Systems (OS)
- Net-Centric Computing (NC)
- Programming Languages (PL)
- Human-Computer Interaction (HC)
- Graphics and Visual Computing (GV)
- Intelligent Systems (IS)
- Information Management (IM)
- Social and Professional Issues (SP)
- Software Engineering (SE)
- Computational Science (CN)\

Concerning the accreditation of our Program:

1. In order to accomplish the goal of accreditation, I believe a sincere and genuine ongoing attempt to review the curricular details of our CIS program is indispensable. This must also be done collaboratively among our faculty. We cannot afford to renovate only the façade of our program, naively believing that everything is good enough. Currently, our undergraduate program, leading to the degree of Bachelor of Science in Software Engineering, is not in par with the substances expected in a Software Engineering curriculum as recommended by the SE2004 guideline (http://sites.computer.org/ccse/). Instead, although it looks much like a Computer Science (CS) curriculum, it is largely short of a unifying philosophy hooking up modern CS elements as recommended by the CS2008 Curriculum Update (http://www.acm.org/education/curricula-recommendations). It is especially weak in the context of evidence-based assessment in student learning, a critical element in the accreditation requirement from such international body as ABET (Accreditation Board for Engineering and Technology).

2. I sincerely hope that accreditation of our program is not regarded as a make-believe trivial process which is important as far as the goal is yet to be reached. Surely, an engineering instructor's life was simpler in the old days, especially before 1990 in the US. All they have to do was lecture on the topics in the syllabus, give the assignments and tests, and assign grades. If the syllabus was covered, the course was considered successful, regardless of whether or not students learned anything. Conversations in the faculty lounge were about research budgets and words like "learning outcomes," and "Bloom's taxonomy" never entered the picture. And the term "assessment" was hardly heard, too. Skills such as critical thinking, creative problem solving, interpersonal skills and entrepreneurship, were hardly taught nor assessed nor valued in engineering curricula. Indeed, people in industry have been complaining about those deficiencies in engineering graduates for long, but it is only not long ago that growing numbers of engineering school administrators and faculty members around the world, especially, in the US and Europe, initiated curriculum reforms designed to equip students with a broad array of new skills. In the US, NSF's move to pour serious money into promoting those reforms in the 1990s, has been followed by ABET's dramatic switch (to the surprises of many) to an outcomes-based program evaluation and accreditation system, starting in 1996, and finalizing as a universal move in 2001. Namely, if we are to receive accreditation from ABET, we should better get started with outcomes-based learning, teaching, and assessment practices as soon as possible. It surely takes us from four to eight years time, as a conservative estimate, before any concrete evidences in student learning could be built up ready for external evaluation, given the current fuzzy state of our program.

3. Under the new ABET accreditation rules of game, a program was no longer evaluated based on what the faculty was teaching – how many credits of engineering XXX – but rather on what the students were learning. Engineering programs now had to define in concrete terms the knowledge, skills, and values they wanted their graduates to have and then prove that the graduates in fact were getting them. To get the proof, the program instructors had to come up with ways to measure those attributes – which is to say, they now had to get used to the business of assessing learning outcomes. Most of us on engineering faculties were uncomfortable with this situation. Some of the ILOs (intended learning outcomes) were familiar, such as mathematical problem solving ability, skills in computer applications, and they believed they knew how to teach and assess them. Other ILOs were problematic, since they included such attributes as awareness of professional and ethical responsibilities, communication skills, ability to work in multi-disciplinary teams, and understanding of global and societal impact of engineering decisions. Since the 1990s, many engineering professors have invented solutions in efforts to prepare for their next ABET visit, a sort of real-life scenarios to live with. If we are pursuing the path of ABET accreditation, we will definitely have to face similar situation.

Concerning the Curriculum Development of our Program:

For almost two decades, the manner and methods of viewing education in terms of the outcomes of learning rather than the curriculum content or the actions of teachers have become increasingly the state-of-the-practice in higher learning, as exemplified by ABET's adoption of outcomes-based program evaluation and accreditation system. We have to accept that, after all, it is the learning done by learners that is the important result in educational activity. Yet, this outcomes-based work may not be smooth, as my personal experience over the past twelve years on course-based implementation (SFTW241 and SFTW300) has demonstrated, especially in an atmosphere where the concept of outcomes-based assessment has never landed officially (at UM) until recently. On a course delivery level, I have many students' hearts and minds to persuade. On a

program implementation level, we have many faculty's hearts and minds to win over for a successful implementation. California State University – Monterey Bay (CSUMB), just slightly over 15 years old, is a good example showing the first fruit of outcomes-based LTA (learning-teaching-assessing) practice. CSUMB started out as an outcomes-based LTA university, but it still takes that long to enjoy its real fruit of student achievement. The key is that assessment matters in college education. Yet, what is assessment?

According to the Middle States Commission on Higher Education in the US (Suskie, 2009), the most straightforward way to understand outcomes-based assessment is to put it into the context of a dynamic learning-teaching-assessment cycle, composed of four steps as follows:

1) Develop clearly articulated written statements of intended learning outcomes;
2) Design learning experiences that provide intentional, purposeful opportunities for students to achieve those learning outcomes;
3) Implement appropriate measures of student achievement of key learning outcomes;
4) Use assessment results to improve teaching and learning.

Indeed, the practice of outcomes-based assessment (OBA) is not a new movement, and it is related to an educational model in which curriculum and pedagogy and assessment are all focused on student learning outcomes. It is an educational process that fosters continuous attention to student learning and promotes institutional accountability based on student learning. Operationally, this approach emphasizes such important practices as: Faculty publicly articulating assessment information in advance of instruction; students being able to direct their learning efforts to clear expectations; and students' progress and acquisition of learning being determined by evidence demonstrated in achieving the learning outcomes. So, the key component in the OBA model is outcomes which inform curriculum, teaching and assessment. In this light, it is convinced that if we are truly embracing a learner-centered model of undergraduate education, outcomes-based assessment is the necessary essence not to be ignored. One of the most important conclusions about the effect of outcomes on student learning comes from the studies of John Biggs (2007). Biggs found that student achieve deep learning when they have outcomes on which to focus. If students do not know what is important to focus on in their studies, they try to cover all the information, so they skim, they cram, and they stay on the surface. If they have a priority or focus, they are able to dig, to expand, and to achieve depth of understanding.

Biggs, J. & Tang, C. (2007). *Teaching for quality learning at university, 3e.* Buckingham, Great Britain: Open University Press.

Suskie, L. (2009). *Assessing student learning: A common sense guide, 2e.* San Francisco, CA: Jossey-Bass.

Finally, it is my conviction that the curriculum (intended learning outcomes) is the heart of a student's college experience. It is a university's primary means of helping students develop in directions valued by its faculty. In today's Macau, we are being urged to assess especially carefully the quality of our curricula. We as faculty are responding to this challenge as a practical means of both attracting and retaining more students and ensuring their success and producing high-quality outcomes for Macau. Thereby, a curriculum should be based on a carefully thought-out philosophy of education and should be clearly connected to our institution's stated mission. Any curricula mission statement and written curricula goals (students' intended learning outcomes or results) should articulate curricula purpose and aims – what graduates should know and be able to do and those attitudes and values our faculty believes are appropriate to well-educated men and women in the specific program of study. These goals and their more specific objectives must be described in considerable detail and in behavioral language that will permit designing the curriculum and assessing its degree of achievement (its actual outcomes).

With that note, please kindly accept, once again, my sincere thanks for the opportunities to share with you my comments and findings in this brief memo after our first meeting in 2010 for faculty feedback in the revised study plan for our DCIS undergraduate degree programs.

All the best!

Kam Hou Vat
Department of Computer & Information Science
Faculty of Science & Technology
University of Macau, Macau
http://www.fst.umac.mo/en/staff/fstkhv.html
e-mail: fstkhv@umac.mo