EJEE Special Issue
Computing as an Engineering Tool

European Journal of Engineering Education
vol. 25, pp. 45-56

Title of the Submission: A Skills Development Framework For Learning Computing Tools in the Context of Engineering Practice
Word Count: 4908 (excluding references and appendices)
Topic Keywords: Informatics, Ethics, Engineering Computing, Engineering Practice

Contact Author:
Dr David Lowe
Affiliation: Faculty of Engineering
University of Technology, Sydney
Address: P.O. Box 123 Broadway
City: Sydney
State/Province: NSW
Postal/Zip Code: 2007
Country: Australia
Telephone: +61 2 9514 2526
Fax: +61 2 9514 2435
E-mail: dbl@eng.uts.edu.au

Additional Authors:
Dr. Craig Scott
Mr. Ravin Bagia
Affiliation: Faculty of Engineering
University of Technology, Sydney
Address: P.O. Box 123 Broadway
City: Sydney
State/Province: NSW
Postal/Zip Code: 2007
Country: Australia
Telephone: +61 2 9514 2397
Fax: +61 2 9514 2435
E-mail: cascott, ravin@eng.uts.edu.au
A SKILLS DEVELOPMENT FRAMEWORK FOR LEARNING COMPUTING TOOLS IN THE CONTEXT OF ENGINEERING PRACTICE

Dr. D. B. Lowe, Dr. C. A. Scott and Mr. Ravin Bagia
Faculty of Engineering
University of Technology, Sydney
PO Box 123 Broadway NSW 2007
Ph: +61 2 9514 2526 Fax: +61 2 9514 2435
e-mail: {dbl, cascott, ravin}@eng.uts.edu.au

SUMMARY

Informatics is a first year subject in the new Bachelor of Engineering/Graduate Diploma in Engineering Practice degree at the University of Technology, Sydney. All undergraduate engineering students must do this subject irrespective of their intended engineering discipline (Civil, Mechanical, Environmental Systems, Electrical, Telecommunications, Computer Systems). The focus of this subject is to introduce students to computational tools (such as spreadsheets and programming languages) and information retrieval tools (such as the WWW), which support engineering, as well as looking at the effective and professional use of these tools. Informatics has three core elements: the development of specific skills which will assist students in both their studies and their professional practice; the development of an ability to continue to develop further skills independently, and consideration of a wide variety of issues related to the computational tools that the students are using. Examples of the issues covered include: privacy; accuracy of content; fraud; security; the professional and ethical responsibility of engineers for the results of their calculations; selecting the best tool for the job; and limitations of computational tools. This paper will review the activities and challenges the students faced, describe the approaches that we adopted and the various issues we raised, the expected and actual outcomes arising from the activities, and our plans for the future of the subject.

INTRODUCTION

In 1997, the Faculty of Engineering of the University of Technology, Sydney (UTS) undertook a major overhaul of its undergraduate Engineering programs. This included redeveloping the structure, content, and focus of all undergraduate subjects. The essence of this redevelopment is a set of core subjects common to all undergraduate engineering degrees, with the students focusing their degree through the choice of a field-of-practice major (discipline) and electives. Informatics is one of the core subjects and hence it is undertaken by all students irrespective of their field of practice be it Mechanical, Civil, Environmental Systems, Electrical, Telecommunications, Software, or Computer Systems. The aim of Informatics is to introduce 1st year students to the use of computing tools in an engineering context.

Although the use of computing tools is regularly taught in the context of engineering problems, it is often not taught in the context of engineering practice. For example, software development or programming subjects will often teach programming in a way which shows how it can be used to address specific problems; such as how to write software for engineering analysis. Such subjects will, however, rarely consider issues such as the impact of errors or inaccuracies in programs, the professional responsibilities of the students with respect to the developed software, or how students develop an ability to continually adopt changing tool sets. In developing Informatics we explicitly attempted to address these considerations.

Most computing courses develop specific skills in various tools but they fail to recognise that the range of tools which students will encounter in engineering practice goes well beyond what can be even partially addressed within a formal degree program. Indeed, many tools (and the technologies which underlie them) are changing so rapidly that there will be little similarity between those available at the start of a student’s degree and those at the end of the degree. This means that the students will regularly encounter tools with which they have no experience or familiarity. Two needs thus arise for engineering students: the ability to learn new tools and the ability to critically evaluate tools.

Students often lack the ability (or confidence) to continually develop their own skills. Yet as part of their professional engineering practice they will be constantly required to develop not only their knowledge, but also their skills in utilising the tools which support them in their practice. Within a relatively short time after
graduation they are also required to mentor other colleagues in the use of these tools. This means that the students not only need to learn specific tools, but need also to “learn how to learn” so that they may continually refresh their existing tool skills and acquire new ones without the support that a teaching and learning environment provides.

Throughout most of their engineering education, students are rarely explicitly encouraged to develop a sense of the professional implications of the use of tools. This includes aspects such as an understanding of the accuracy of results obtained, the strengths and limitations of the models that underpin the tools, and how to select the best tool for the job. These factors are all critical in ensuring that tools are used in an appropriate fashion.

Given the above observations, Informatics has three core purposes. The first is to develop specific skills in using information and computational tools (such as spreadsheets, the internet, and programming languages). The second is to develop an ability to continue to develop skills in a broad range of tools. The third, and most unusual, is to develop an understanding of the professional issues which surround the use of computational tools - the types of tools and resources which allow engineers to manage the information, models, complex calculations, and processes associated with carrying out engineering activities. We shall begin this paper by looking at the subject’s educational objectives in a little more detail. We will then look at the subject structure and content which we have developed to address these objectives, and some of the early outcomes that we have observed.

SUBJECT PURPOSE AND EDUCATIONAL OBJECTIVES

The development of Informatics was undertaken using a very methodical approach. We began by considering how the subject could contribute to the overall objectives of the engineering degree program at UTS, and then refined these contributions into a set of specific educational objectives for the subject. Let us begin by considering the overall objectives of the degree program.

In redeveloping the undergraduate B.E. / DipEngPractice degree programs at UTS a set of desirable graduate attributes was defined [7]. These graduate attributes, which are in accordance with the reports and competency standards of the Institution of Engineers, Australia [2, 3, 4], provide a definitive description of the characteristics that we believe are important for our graduates. The attributes also act as threads providing coupling between subjects and cohesion for the overall course. In developing the structure and content for Informatics we began by looking at these graduate attributes within the context of the overall focus of the subject and determining the attributes to which the subject could best contribute. Appendix 1 includes a partial extract from the graduate attributes and the ways in which Informatics aims to contribute to these attributes.

These contributions were analysed to determine the specific educational objectives of Informatics. One significant observation is that the contributions apparently work at three levels. First is the targeted development of understanding and skills related to specific areas (such as skills in using a particular programming language or a particular type of spreadsheet). The subject aims to develop specific skills for several reasons. Developing these skills will be important in developing the students’ confidence with respect to the use and role of tools. They will also provide the students with the skills required to support their learning in other areas (such as using modelling tools in systems design and spreadsheets for analysis of laboratory results). Finally, it will help the students understand the broader issues discussed elsewhere in the subject.

Second is the development of general understanding and/or knowledge related to the area of informatics tools and resources (such as understanding the different classes of computational tools that are available). We explicitly do not want to only develop skills in using particular tools. Rather we wish to develop the students ability to understand general classes of tools and to have the ability to confidently approach the development of their own skills in using new tools (both different variants of known tools, and new classes of tools). A key aspect of this is not tying the skills modules to a specific product, for example the spreadsheet module can be completed using any spreadsheet product. An added advantage of this is that students are free to use the software packages they have installed at home and not have to purchase new software simply to undertake the subject.

The third level of contribution is to students’ understanding related to the professional application of their skills. A major goal of the subject is to ensure that the students develop an understanding of the implication of their professional activities and their responsibilities with respect to these activities. For example, if the students develop software that controls a robot that subsequently malfunctions and injures or kills someone, are they responsible?
Based on the above contributions to the development of graduate attributes we identified a set of subject educational objectives. Again, these objectives are too numerous to include here. Appendix B contains an extract from the detailed subject objectives.

SUBJECT CONSTRAINTS

Before considering the structure of the subject that was developed to address the learning objectives, it is worthwhile considering a number of constraints that affected the subject design. The first major element that was important to take into account was the huge variation in the level of understanding and skills in the commencing students. Although it is normal to expect variations in the prior learning of students, especially in first year subjects, the variation is more pronounced in subjects which focus at least partly on technical and computing skills. We have found that it is common to have students with little or no exposure to computers (though this is, admittedly, becoming more rare) through to students who have very extensive (albeit rather narrow) experience. The structure of the subject needs to take into account the Recognition of Prior Learning (RPL) in such a way that all students in the class have an effective learning experience. The RPL issue is further complicated, however, with students often incorrectly perceiving the ability to use a computer package as a true understanding of the purpose and uses of that tool.

Along with large variations in prior learning, there was also a large variation in the expectations in outcomes, and attitudes of various students. Students across all engineering disciplines were required to complete this subject, and their relative motivations and expectations varied considerably. Students undertaking mechanical and civil engineering majors had a greater tendency to view the subject (at least initially) as unnecessary, or as one of the hurdles that had to be jumped in order to get a degree. Students in Telecommunications and Computer Systems majors were more likely to be motivated, having a higher interest in computing for its own sake.

The subject structure also needs to take into account the limited resources available. The subject will typically have up to 320 students enrolled every semester. To accommodate this many students in laboratories at one time is impractical. Similarly, anything that can be done to ease the burden on staff or facilities (such as computer rooms and tutorial rooms) needed to be considered.

SUBJECT STRUCTURE

Based on the above observations, a flexible structure was designed for the subject. This structure is based around a series of subject modules. These modules focus on different aspects of the subject, and are broken into two main streams: skills and issues. The issues modules cover the development of understanding of classes of informatics tools, uses and limitations of these tools, the relationships between these tools, meta-skills required to independently learn new tools, and ethical and professional obligations in using these tools. The skills modules cover specific skills in different tools, including spreadsheets, the Internet, operating systems, programming, and a number of other tools.

Students are required to complete a set of compulsory modules, and one or more elective modules. To accommodate the huge variability in the prior learning and experience of the students, and the variations in student expectations, the students are given the option of negotiating to replace compulsory modules (in which they can demonstrate appropriate prior understanding) with an alternative selected from the elective modules. In this way we are encouraging students to utilise the opportunity to extend themselves and maximise their learning as much as possible, as well as providing them with the opportunity to focus their learning in areas most appropriate to their needs. The consequences of this are that student expectations and motivations are positively influenced concomitant with positive changes to study strategies and learning outcomes [8].

The various modules includes large classes (lectures involving presentation and explanation of material), small classes (tutorials and workshops), individual study (tutor-supported study involving one-on-one interaction between a tutor and a student), and independent study (study undertaken solely by the student) supported by self-learning modules. Each of these modes of learning/teaching has different resource implications, but overall provide a balanced approach to the subject.

To provide a sense of cohesion to the subject (especially in light of its module-based nature), students are allocated to a specific tutor for the duration of the subject. This tutor takes the students for all small class sessions, including both issues and skills modules. This tutor becomes responsible for the students overall subject program (including aspects such as negotiating with them about exemptions from compulsory modules
and maintaining their marks for the subject) and will normally be the students first point of call in the case of questions or problems related to subject content and logistics.

**ASSESSMENT**

The assessment in the subject is designed to provide a balance between formative assessment and summative assessment. With the described subject structure it is impractical for every module undertaken by a student to have individual summative assessment tasks, as this would result in an excessive load for staff and the potential for students to become too focussed on marks and not on learning. Nevertheless it is important that the extent of the students understanding of both skills and issues be effectively evaluated. To achieve this the assessment in the subject is split into two components: Mastery assessment and Advanced assessment.

All students are required to complete the Mastery assessment component, which solely determines the students’ pass/fail grade but not their final mark. If the student passes the Mastery component then they are guaranteed a pass result in the subject even if they do not attempt the advanced assessment components. If they fail any component of the Mastery assessment then they fail the subject. This approach removes the common problem of “mark-accumulation” and focuses the student on developing an appropriate level of understanding. To satisfy the Mastery requirements, the students must satisfactorily complete all the subject modules (including any negotiated elective modules), and pass the final oral and coding exams. Satisfactory completion of each module is based on evaluating, on a Pass/Fail basis, the student achieving an acceptable level of understanding. This is determined through the module assessment tasks and through the examination of a Journal that the students maintain throughout the subject. The Journal is an important element of the assessment and the students are encouraged to record their reflections as they engage with the subject matter.

Those students who wish to earn a grade beyond a Pass can attempt the advanced assessment component. The Advanced assessment components involve a major programming assignment, and a learning contract which requires the student to negotiate a task which demonstrates understanding of two or more of the non-programming skills modules and at least one major aspect of the issues modules.

The Mastery/Advanced approach to assessment has been used extensively in similar subjects (notably software development subjects) over the last few years and has proved to be particularly effective [6]. This structure encourages students to focus on their learning, as distinct from focusing on mark accumulation, yet provides the scope for more ambitious students to obtain a grade which reflects the extra learning and understanding they have achieved. This approach also allows tutors to concentrate on constructive feedback rather than focussing on allocating marks in situations where they matter little. The negotiated Learning Contracts which form the Advanced assessment allow students the freedom to explore topics and problems which are of particular interest to them. The diversity of Learning Contracts negotiated and submitted is testament to the motivational value of this approach. A compulsory component of any Learning Contract is a personal reflection examining the degree to which the student met their learning objectives, which approaches to the learning goal(s) were effective and why, and also those approaches which were less effective.

**EXAMPLES**

**Problem Solving Module (Introduction to Programming and Algorithm Design)**

*Informatics* is different to many introductory programming subjects in that it offers students a choice of programming language. To avoid duplication of content across the various programming modules and to start the process of developing generic programming skills, a generic “Problem Solving” module was developed. The focus of this module was not programming but rather approaches to solving problems – including aspects such as algorithm design and generic programming concepts. Experiences gained from other programming subjects have indicated that those students who struggle often have weak problem solving skills. In Informatics, programming is discussed as a mechanism for implementing a solution to a problem rather than allowing students to work under the misconception that sitting at a terminal and typing code will eventually solve problems. The lecture component of the module introduced the problem solving strategies, the strengths and weaknesses of computer programs, and mechanisms for dealing with complexity.

The key to the module was a set of tutorial/workshops involving hands-on problem solving. The aim is to first learn to solve problems, then to formalise the algorithm, to recognise that a given algorithm may be optimal for one problem and not necessarily optimal for a similar problem, and finally recognise which types of algorithms are suitable for implementation using a computer program. Some of the key benefits gained by students, as
indicated by their journal reflections, are the acknowledgment that there is more than one solution, and that, individually, they will not be able to identify all possible solutions to a problem and a team approach may be more appropriate. As an example, students are asked to bring a deck of playing cards to these workshops, which are then used for various exercises. These include finding a given card in the deck, sorting a deck of cards, and scoring a Blackjack hand. These examples also introduce the uncertainty in user specifications with students raising questions such as “[When sorting] is the Ace below the 2 or above the King?” Student feedback from these lectures and workshops was very positive.

**Issues Module: Professional Responsibilities**

Several case studies were used to encourage the students to explore what it means to be a professional. The first of these case studies utilises a hypothetical situation described in “The Case of the Killer Robot” [1]. This book describes a situation where a software engineer (Randy Samuels) is indicted for manslaughter based on his involvement in the development of software that controls a robot. The robot has malfunctioned and killed one of its operators. We required the students to read extracts relating to this hypothetical case and then reflect on various issues that are raised. The reflection guided by a series of questions such as:

1. Develop a list of criteria that you believe could be used to decide whether a programmer should be held accountable in a case such as the Killer Robot.
2. Should Randy Samuels be indicted for manslaughter?
3. If you were pressured to release a product (such as the Killer Robot) before it was safe, reliable, or effective what would you do?
4. Write your reflections on how being a member of a development team could influence your approach and your sense of professional responsibility!

Following these reflections we ran a workshop for the students in which we focused on developing the students understanding of their ethical, moral, and professional obligations (as an engineer) in the appropriate use of information and tools. This includes aspects such as safety and reliability, sustainability, and critical evaluation and effective communication of the results obtained from tools. We then gave the students another set of readings related to a true case study where a software design fault in a Therac-25 medical radiation machine resulted in the deaths of a number of people [5]. Again, the students are asked to read certain material and then document their reflections based on a set of guiding questions.

**OUTCOMES AND PLANS FOR THE FUTURE**

At the time of writing this paper the subject has just completed its second offering. Feedback on the subject has been obtained using formal evaluations and small group discussions. Based on these observations it is possible to draw a number of conclusions regarding the structure of the subject. First, the students have shown a high degree of enthusiasm for the subject. In particular the technical nature of the much of the subject gains the interest of the students, and despite their initial reservations they become quite involved in the issues aspects of the subject. A number of the subject tutors have been approached by students outside normal class times, and become involved in animated discussions regarding ethical and professional questions.

Student feedback has also indicated that the overall structure of the subject is appreciated since it offers considerable choice in both what is learnt and how it is assessed. For example, one student commented that “the electives that are offered are helpful – they allowed me to learn things that are most relevant”. Another student commented that “[the various modules] really challenged my existing knowledge”. This choice has had a quite positive influence on the student motivation.

Several negative aspects of the structure also became apparent. The first is that the students found the logistics of the subject rather complicated and that this lead to some difficulties in managing their workload. For the subject flexibility to have maximum positive effect the students need to be able to understand the structure of the subject, how the various components inter-relate and the expectations on them. The students also were generally concerned about the workload. Whilst subjects involving a programming component typically have a heavy workload, the modularised structure of the subject meant that each individual module was subject to “content bloat”. We are currently considering ways in which we can improve the synergy between modules to control the workload expectations. The simplest approach, and the one that can be implemented immediately, is to introduce one or more key problems and thread these through many or all of the modules. This will allow more direct comparison and evaluation of the various tools and issues arising from their use. This will also address some students’ interpretations that the modules are isolated and unrelated components. Students also frequently expressed concerns over the heavy workload towards the end of the semester.
While the students have the opportunity to work on many of the Mastery components throughout the semester, they invariably leave the work until the last minute. The modular nature of the subject exacerbates the student time-management problem. To this end we will be endeavouring to provide better guidelines as to how much work they should have completed at any stage in the semester and to assist them in managing their time. An activity under consideration is to have a small workshop in which time management is discussed and a key outcome is each student developing a plan for completing the work they need to do in the subject. At appropriate times during the semester, subject tutors could then review with each student their progress against their plan.

Another logistical aspect was that we did not adequately consider the implications of the Mastery / Advanced assessment structure of the subject. Whilst this certainly achieved the goals of the subject with respect to removing the student focus on mark accumulation rather then developing understanding, a number of students expressed concern over the consequences of only attempting the Mastery component. At UTS, results for a subject can be presented as ungraded (satisfactory / unsatisfactory) or a combined mark / grade (with the mark being a percentage and the grade being Fail, Pass, Credit, Distinction or High Distinction). The original intention was to provide an ungraded result for those students only attempting the Mastery component, and a graded result for those also attempting any of the Advanced components. This was not practical given that the UTS system requires subjects to adopt one scheme or the other. The result is that students attempting (and passing) only the Mastery component would receive a result of 50-Pass. Many students perceived this as implying that they were not able to achieve a higher mark, rather than reflecting their choice not to attempt the advanced component. The solution we intend to implement is to have two subject numbers; one set up as ungraded and the other graded. All students enrol in the former and will be automatically transferred to the second subject when they attempt an advanced assessment component.

The journal was viewed with very mixed feelings. Whilst a few students resented the need to reflect on their work, most students saw the journal as both as a means to document their learning and as a tool for venting minor frustrations through the semester. One student noted “The learning techniques taught, such as the reflective journal, have been very beneficial”. In requiring the students to maintain a journal we overestimated their ability to do this without direct guidelines and support. In future we will have the staff review journals and provide feedback on a more regular basis. We will also provide more guidelines and structured activities for the journal and structured activities to assist the students to understand and appreciate the purpose and benefits of a journal.

Our attempts to recognise prior learning were not as successful as hoped. Relatively few students elected to opt out of a mastery component with which they were already familiar and undertake an extra elective module. There are likely many reasons for this behaviour but a strong disincentive for the students is the scheduling of the electives at the end of semester when they are busy with the advanced components of the subject as well as with all their other subjects. This can be overcome relatively easily by revising the schedule and running elective modules throughout the semester and/or develop some elective modules as self-learning packages.

Finally, it is important to note that the diversity of coverage in the subject (ranging from various tools, to considerations of the students’ professional obligations in using these tools) was seen as a strong positive factor. A typical comment was “The best aspects were that we got a good introduction to many applications which we will use throughout our education/career”. Subject surveys and anecdotal evidence indicate that the students are developing, very early in the degree, a sense of the context which defines how they can (and should) approach the practice of engineering in a professional contexts.

CONCLUSIONS

At the time of writing this article, Informatics had just been offered for the second time and was being evaluated to determine what further improvements could be undertaken. Both anecdotal and formal feedback has been very positive. The students have shown that they are willing to actively engage in their learning process – despite the large variations in student prior learning, expectations and desired outcomes. The students appreciate the freedom to develop an understanding of the subject material by exploring areas that intrigue them. This in turn has had a strong impact on students’ motivation. Conversely, the students have found the complicated logistics of the subject rather difficult. This is largely a consequence of the modular nature of the subject, and an area that we are currently looking at ways to improve.

Another aspect of the modular nature is that subject is often viewed as “dumping ground” for content which doesn’t fit elsewhere. Whenever a specific skill is needed elsewhere in the course, it is assumed that Informatics can be used to develop this skill. Similarly it is easy for staff involved in the subject to add minor aspects to their
modules – but very difficult to remove content. There is a significant danger of content bloat or concept overload. This requires very careful management of the subject by staff, especially the subject coordinator.

Indeed, the staff involved in the subject is a key element in its success. The tutors in the subject provide, for the students, the cohesion between the different modules and support in negotiating appropriate learning contracts. The staff must understand the educational underpinnings of the subject in order to appreciate, and hence support, the approach that has been developed. We have also found that the staff have got to know the students much better and hence can support their educational experience more effectively.

Overall the subject has been very rewarding for both staff and students. Creative ways of structuring the subject, flexibility in the student learning approaches, and providing an engineering practice context for the development of understanding computational tools, have all resulted in a valuable and effective learning experience. Overall the subject has been logistically fragile, but educationally successful.

ACKNOWLEDGEMENTS

The authors wish to acknowledge the invaluable contributions of other staff involved in the development and implementation of Informatics: James Boswell, Martin Evans, Ron Meegoda, Austin Mack, Richard Lai, Ben Rodanski and Ali Saleh.

REFERENCES


APPENDIX 1: CONTRIBUTIONS TO GRADUATE ATTRIBUTES

The following partial excerpts are from the development documentation for Informatics. These detail how Informatics contributes to the graduate attributes of the overall B.E./DipEngPrac degree program.

Professional Formation:

Attribute 1. Values and social contexts

Critical awareness of codes of ethics, their role and limitations: Students will develop an awareness of their ethical responsibilities with respect to interpretation and utilisation of computational results.

Attribute 2. Management skills

Experience in applying systems thinking where appropriate and approaching complex problems: Informatics will develop an understanding of the ways in which different types of tools can aid in managing the complexity of engineering problems.

Attribute 3. Technical expertise

Recognition and application of fundamental mathematical concepts and methods: Informatics will develop an understanding of categories of tools aimed at supporting fundamental mathematical concepts and methods. In particular Informatics will develop an understanding of how tools aid in applying mathematics, the relationship between the mathematical concepts and the tools which support them, the capabilities and limitations of these tools, when it is appropriate to utilise these tools.

Recognition and application of fundamental limits and principles of physical sciences: Informatics will develop an understanding of the relationships between the fundamental limits and principles of physical sciences, and the limits and principles of tools which support the engineering utilisation of these principles. Informatics should also develop an understanding of concepts of accuracy and errors.

Personal Development:

Attribute 1. Maturity

Critical and independent thinking: Informatics will promote the ongoing development of critical and independent thinking through critical analysis of informatics tools and resources, their relevance, limitations, etc.

Awareness of the importance of self-motivation and taking responsibility of one’s own decisions: Informatics will promote the ongoing development of an understanding of the responsibility for appropriate utilisation of tools and resources in supporting decisions.

Academic Development:
**Attribute 2: Information literacy**

*Experience in seeking, accessing, retrieving, and evaluating references from a variety of sources: Informatics will promote the ongoing development of an understanding of the types of tools, and the usage of these tools, which support identification and utilisation of information from a variety of sources.*

**Attribute 3: Problem posing and solving**

*Familiarity with the process of conceptualisation and articulation of problems in terms of systems and processes: Informatics will develop the students ability to utilise informatics tools in the conceptualisation, visualisation and articulation of engineering problems.*

*Recognition of relevant principles and useful approaches for identifying and structuring a problem: Informatics will develop the students understanding of how informatics tools can be used in supporting the identification and structuring of problems.*

*Awareness of problems as constructs based on values, interests, and perceptions of needs: Informatics will develop the students awareness of the relationships between the structure of problems and how these problems are expressed within tools which support reasoning about, or manipulation of these problems.*

*Familiarity with a variety of problem solving strategies: Informatics will develop the students understanding of different classes of tools which support different strategies to problem solving (indeed, according to the brief, this is a core focus of Informatics)*

*Awareness of the need and familiarity with ways of validating solutions: Informatics will develop the students understanding of the limitations of informatics tools, and the need to validate solutions (and mechanisms for performing these validations).*

**APPENDIX 2: CONTRIBUTIONS TO GRADUATE ATTRIBUTES**

The complete set of detailed objectives for *Informatics* is long to include here. The following partial extracts provide examples of the objectives.

**Objective 1)** Informatics aims to specifically develop a deep understanding of:
- (a) the types of engineering problems which can benefit from the use of additional resources and tools, and what these benefits are, especially in terms of managing the complexity of engineering problems
- (b) categories of resources and tools which are available to support the carrying out of different aspects of engineering activities, the breadth of these resources and tools, and their strengths and limitations
- (c) which types of tools are most appropriate for given types of problems and specialisations, and the need to validate solutions from these tools
- (h) how informatics tools and resources relate to the culture of engineering
- (l) how engineering tools relate to, and support the management of, reliability and safety.

**Objective 2)** Informatics aims to develop specific understanding and skills in:
- (a) utilising the tools which assist engineers in identifying and utilising information available within libraries, on the Internet, and other suitable sources
- (e) utilising the tools which assist engineers in resource management
- (f) utilising the tools which assist engineers in risk management and decision making
- (g) utilising the tools which support different stages of the design process

**Objective 3)** Informatics aims to continue to develop and refine the students understanding and maturity with respect to:
- (a) critical and independent thinking, especially with respect to critical analysis of informatics tools and resources and their relevance and limitations.
- (b) understanding of the responsibility of engineers for appropriate and ethical utilisation of tools and resources in supporting decisions and carrying out engineering activities.
- (c) issues of sustainability.