Jenny Carter · Michael O'Grady
Clive Rosen *Editors*

Higher Education Computer Science

A Manual of Practical Approaches

Springer

Higher Education Computer Science

Jenny Carter · Michael O'Grady · Clive Rosen
Editors

Higher Education Computer Science

A Manual of Practical Approaches

Springer

Editors
Jenny Carter
Department of Computer Science
University of Huddersfield
Huddersfield, UK

Clive Rosen
Passerelle Systems
Newcastle-under-Lyme, UK

Michael O'Grady
Department of Computer Science
University of Huddersfield
Huddersfield, UK

ISBN 978-3-319-98589-3 ISBN 978-3-319-98590-9 (eBook)
https://doi.org/10.1007/978-3-319-98590-9

Library of Congress Control Number: 2018951212

Preface

It is cliché to say that higher education is changing. There has been continuous change at least since the 1970s. Nevertheless, the changes that are occurring at the moment seem to be more profound and more widespread than ever. All institutions, however prestigious or uncelebrated, are being affected. For some of the most prestigious, the shock of the change has been greatest, and this is new. Universities that have prided themselves on their research records are being asked to reconsider their teaching capabilities. They are being challenged on their records on student diversity. They are being forced to examine their utility to the economy.

There are other forces at play. Increased competition for students between universities (both globally and within country) and the requirement to be self-financing are driving universities to justify their value to prospective students, in terms of both the financial investment and the long-term skills students will need to prosper in a rapidly changing employment market. Technological change in the form of access to information, about both the universities themselves and the subjects they teach, is powering a trend towards consumerism amongst students. People are asking the question "what are universities for?" Teaching materials in the form of MOOCs are freely available and of high quality; and if not there, there is always Wikipedia. Social media has just about extinguished the last vestiges of deference.

In the light of all these social changes, government has weighed in (and waded in) to insist on accountability. Initially (in the UK), this was to justify expenditure on research. More recently, the "Teaching Excellence Framework" has sought to establish measures of teaching quality. The validity, and even the reliability of these measures, has been questioned, but whatever their academic credibility, the truism that "whatever we measure we change" has already affected the university sector.

Computer science and its related disciplines have been more exposed to these forces than most subject areas. The industry-oriented nature of the subject has resulted in high volatility in student application numbers as market trends affect demand for graduates. Technical change within the subject area has caused curricula instability. Waves from almost mystical adoration of computers and computing to commodified dismissal (what's the difference between a computer and a washing machine?) combined with perceptions of subject complexity and gender stereotyping exacerbate the cyclical trends in subject popularity.

These generic factors together with subject characteristics such as its basic intangibility and intellectual complexity have made CS and its allied subject areas inherently difficult to teach. Large student numbers lead to diverse student populations. Poor coverage of the subject area at the pre-university level results in bipolar distributions of subject knowledge amongst university entrants. The gap between physical constructs and the subject's virtual concepts creates an intellectual schism students must navigate to make progress.

Teachers have been aware of these problems for many years and have tried various approaches to address the issues. Yet the increased pressure generated by the recent, intense scrutiny has meant that the urgency to find solutions has intensified further. This book gathers together a range of approaches that individual instructors have found helpful in addressing these common problems. These are practical applications that experienced practitioners have adopted to meet the needs of their students. The combined experience of contributors to this book is approaching 500 years. We cannot claim to have found solutions; that is unlikely to ever happen. But, by bringing together this community of practice in one volume, we hope to stimulate your own ideas, vitalise your teaching and enhance your practice.

The book is divided into three parts: "Approaches to Learning", "Teaching: Examples of Practice" and "Employability and Group Work". The "Approaches to Learning" part, whilst based on personal experience as is the whole book, offers some ideas about how we can move away from didactic delivery stage front. The "Teaching: Examples of Practice" part addresses some specific problem areas in teaching CS: programming, information systems management and design as well as some ideas about delivery to diverse student cohorts and automatic marking of programming work. Finally, the "Employability and Group Work" part does what it says on the tin, providing some novel ways of approaching employability.

Liz Coulter-Smith's opening chapter on student "multitasking" in the classroom, in some ways, does not quite fit with the rest of the book as it is not strictly focussed on computer science students. It is included here because (a) computing students are amongst the most likely groups of students to engage in multitasking, and (b) by looking at the changed culture and experience this current generation students have grown up with, it sets the scene so well for the chapters that follow. Diane Kitchin's chapter on active learning offers one of the ways in which we can respond to these changes and address our student's needs and expectations. The "flipped" classroom, Michael O'Grady's chapter, offers an alternative approach. Jenny Carter and Francisco Chiclana's chapter on distance learning and Thomas Lancaster's chapter on academic integrity address two issues that have arisen as a consequence of technological change affecting the classroom environment. All the chapters in this part share a common orientation; they are student-centric rather than lecturer-centric. This attitudinal shift is one that might not sit comfortably on the shoulders of some staff, but we consider it to be essential if we are to engage with, and maintain the engagement of our millennial students. Furthermore, if we can enhance the quality and quantity of engagement, we have a better chance of satisfying the expectations of other stakeholders as well as the students.

The second part, focussed on teaching, arises out of the knowledge and experience of the contributors to this book, of the teaching of computing. It addresses how we can best overcome some of the specific difficulties computer science students face in this most abstract yet practical of subjects. Carlton McDonald's chapter on the teaching of programming analyses the difficulties many novice students face when learning to programme, whilst David Collins's chapter offers an approach using graphics to overcome these difficulties with a smile. Steve Wade's chapter is similar but focusses on information systems management, and Carlo Fabricatore and Maria Ximena López look at systems design. Arjab Singh Khuman's chapter takes a broader perspective on student engagement by looking at style rather than content (though he covers both). Finally in this part, Luke Attwood and Jenny Carter offer some relief for marking programming assignments.

The final part of the book on employability and group work offers some guidance on how to embrace the employability agenda without compromising academic standards. The Enterprise Showcase outlined by Gary Allen and Mike Mavromihales offers one solution, whilst Clive Rosen's chapter on group projects provides a framework for decision-making regarding the running of group projects as well as some practical suggestions. Chris Procter and Vicki Harvey suggest that satisfying employers' expectations compliments rather than compromises the learning process. Finally, Sue Beckingham rounds of the book with her exposition on provisioning students with the soft skills demanded of our students today.

The philosophy underpinning this book is that the relationship between student and instructor is fundamental to the success of the student. It needs to be built on mutual respect and regard. We aim to maximise the achieved potential of students. The approach is facilitative rather than didactic, supportive rather than patriarchal. This may not suit all pedagogic styles or all students, but we believe that a transition from the traditional master/pupil approach is essential to meet the current and future demands of the educational environment.

One word of caution: students have not necessarily, and may not, buy into the contract of having to commit their own time and intellectual effort in order to be successful. This can be a source of conflict between student and staff. However, one of the implicit terms of the contract is that staff must commit to seeking the best approaches to support the learning of their students. This cannot be abrogated even if students don't keep their part of the bargain. We hope that this book will be of aid to teachers seeking to meet their obligations. There are many of us out there!

Two final points:

1. We know of the semantic controversy between use of the terms pedagogy and andragogy, but we do not wish to intervene. In this volume, both terms are used interchangeably.
2. Similarly, the terms “teaching” and “lecturing”, and nouns “instructor” and “facilitator” are all used in the spirit described above, to support student learning.

We hope you find this book helpful, informative and, dare we say it, enjoyable.

Newcastle-under-Lyme, UK Clive Rosen
Huddersfield, UK Jenny Carter
Huddersfield, UK Michael O’Grady

Contents

Part I Approaches to Learning

1 **Changing Minds: Multitasking During Lectures** 3
Liz Coulter-Smith

2 **Active Learning in Large Lectures** 17
Diane Kitchin

3 **The Flipped Classroom** 29
Michael O'Grady

4 **Distance Learning: Lessons Learned from a UK Masters Programme** 51
Jenny Carter and Francisco Chiclana

5 **Academic Integrity for Computer Science Instructors** 59
Thomas Lancaster

Part II Teaching: Examples of Practice

6 **Why Is Teaching Programming Difficult?** 75
Carlton McDonald

7 **Using Graphics to Inspire Failing Students** 95
David Collins

8 **Best Practices for Teaching Information Systems Modelling** 111
Steve Wade

9 **Promoting Design Thinking Through Knowledge Maps: A Case Study in Computer Games Design and Development Education** 127
Carlo Fabricatore and Maria Ximena López

10 Fostering Inclusivity Through Dynamic Teaching Practices 147
Arjab Singh Khuman

11 Semi-automating the Marking of a Java Programming Portfolio Assessment: A Case Study from a UK Undergraduate Programme 161
Luke Attwood and Jenny Carter

Part III Employability and Group Work

12 The Enterprise Showcase Experience 173
Gary Allen and Mike Mavromihales

13 Task Versus Process: A Taxonomy for Group Projects 187
Clive Rosen

14 Realising the Threshold of Employability in Higher Education 203
Chris Procter and Vicki Harvey

15 Baseline Skills—Scaffolding Soft Skills Development Within the Curriculum 221
Sue Beckingham

Index 243

Part I
Approaches to Learning

Chapter 1
Changing Minds: Multitasking During Lectures

Liz Coulter-Smith

Abstract This chapter takes a multidisciplinary approach to multitasking. Media multitasking has, consequently, become a frequent topic amongst academics yet some remarkable new research reveals we may not be taking into full account the changes to our students' ability to learn given the changes to their brains. The risks of multitasking to student achievement has been well researched yet many of the positive related developments in the neurosciences are less well known. This chapter reviews some of this research bringing together information foraging theory, cognitive control and confirmation bias as they relate to the multitasking Generation Z student in higher education. Some significant research findings are discussed including using laptops and similar devices in the classroom. A small survey underpins these discussions at the end of the chapter highlighting student perspectives on multitasking during lectures.

Keywords Multitasking · Cognition · Information foraging · Academic performance

1.1 Introduction

It is in our nature to do more than one thing at a time: To multitask. Multitasking feels good. Dopamine is released every time we turn to a new task (Strayer and Watson 2012). Our motivation to multitask is a natural human urge–we are foragers, and more recently in our technological history, information foragers (Pirolli and Card 1995).

Multitasking is defined as using two or more media concurrently. It is slightly different from task switching where one switches attention between two tasks. They are closely related, but for our purposes, we will define multitasking, sometimes referred to as media-multitasking, as involving at least one device that coincides

L. Coulter-Smith (✉)
University of Northampton, Northampton, UK
e-mail: liz.coulter-smith@northampton.ac.uk

J. Carter et al. (eds.), *Higher Education Computer Science*,
https://doi.org/10.1007/978-3-319-98590-9_1

with the "performance of two or more functionally independent tasks with each of the tasks having unique goals involving distinct stimuli (or stimulus attributes), mental transformation, and response outputs" (Sanbonmatsu et al. 2013).

Multitasking with various devices is also commonplace in university classrooms (Junco 2012). Three out of four students believe technology improves their educational experience and since 2015, 90% of students have both laptops and smartphones[1] (Statista 2017). Media device dependency, especially among 18–20-year-olds, shows 44% are compelled to access a device at least once every ten minutes (VitalSource 2015). These factors are profoundly impacting student focus, attention, distraction and consequently academic performance. Nonetheless, these factors are complex yet offer possible solutions that may require substantial shifts in thinking, both on the part of the student and the lecturer.

This chapter discusses why students are compelled to multitask particularly around information-intensive tasks. The focus is on multitasking in the classroom of first-year university students but also attempts to understand the current multitasking debates involved in attention and distraction in the context of teaching computer science in higher education. This discussion then delves into a few of the recent studies in neuroscience to better understand the complex relationships that underpin multitasking. To summarise, this chapter seeks to expand the discussion on multitasking through the lens of a multidisciplinary approach to the topic. Through a small pilot survey at the end of the chapter, we gather data drawn from a group of first-year computer science students as first-hand evidence of the state of the debate.

1.2 Information Foraging Theory and Multitasking

We have to ask why humans have a compulsion to multitask? What is driving this urge? One theory stands out and helps make sense of this innate drive to multitask where we are in pursuit of information-intensive tasks. Understanding this problem from a behavioural standpoint is vital given the context of teaching and learning in the classroom and given the increasingly sophisticated social and technological tools at the students' disposal. Information foraging theory (IFT) was developed at the Palo Alto Research Center (PARC), to develop project models for the User Interface Research Area, this theory provided 'novel' information visualisation for searching and browsing (Pirolli and Card 1995, p. 50). IFT goes some way to explaining our drive as humans to accumulate information. This theory is particularly important due to the level of information available to students and their drive to multitask and task switch. The IFT research team at PARC primarily used participants from the areas of business intelligence and an MBA. The team quickly realised the depth and variety of phenomena that needed to be dealt with when handling massive volumes of information, deadline constraints and complex search decisions in the context of uncertainty. Early on they realised they were dealing with something different from

[1]Between 2011 and 2017 smartphone use doubled from 21.6 to 44.9 million in the United Kingdom.

the standard human-computer interaction tasks originating from cognitive engineering models of the 1990s. Comparatively, they recognised the behaviours of people seeking information was largely determined or shaped by the architecture of that content, referred to as an *information environment*. It was clear that the participants' behaviour was only minimally shaped by their knowledge of the user interface. What is interesting here is how this early model maps onto the classroom and the context of learning since Pirolli also found behaviour tended to be dominated by uncertainty and continual evaluation–a common attribute when learning a new skill or concept. IFT was theoretically developed from optimal foraging theory (OFT) (Krebs 1977). OFT is largely a theory developed from predictive models of decision rules used by predators and originating from the theory of natural selection focussed on maximising food intake during foraging (MacArthur and Pianka 1966). Generally, IFT theory asserts we have evolved to use, information to solve problems that can pose a threat to us in our environment. Rather than forage for food, we have evolved and adapted to forage for information. IFT theory goes on to explain that we have adapted cognitive solutions for survival. The technological need for survival forms a basis for human interaction with information technologies as demonstrated by the World Wide Web (Pirolli and Card 1995, p. 51). The earliest discussions about multitasking borrowed heavily from the biological sciences in Pirolli and Card's paper. The book Pirolli wrote followed twelve years later, 'Information Foraging Theory' (Pirolli 2007) and it is a singularly foundational work. More recently and no less importantly is the book 'The Distracted Mind' (Gazzaley and Rosen 2016) which further develops information foraging theory from a neuroscience perspective. These two works bring together information foraging and neuroscience placing a plausible bridge for researchers attempting to explain, at least in part, the phenomenon of the human drive to multitask. If we consider these two major works as a partial framework or model for further exploration, then there is a more positive perspective on multitasking than has previously been published since one can then view it as part of our natural evolution and adaptive ability to gather and make sense of increasingly large volumes of information and data in this era.

1.3 Multitasking Is Multidisciplinary

It became apparent that there was a need to expand this chapter beyond the issues of education and to consider the advances in neurosciences and cognitive psychology. It was also clear that media multitasking and its effects have been investigated exhaustively in many ways. "The problem of how the brain undertakes multiple tasks concurrently is one of the oldest in psychology and neuroscience" (Verghese et al. 2016).

In 2009 a summit at Stanford University's Center for Advanced Study in Behavioural Sciences (CASBS) considered the impact of multitasking on learning and development. The purpose was to pull together a multidisciplinary, coherent and scholarly research agenda. Participants came from the fields of neuroscience, child development, cognitive science, communication, education, and business policy. Terms were agreed, including using the word multitasking itself, and that multitasking had become a universal problem needing urgent attention. Solutions were being demanded by parents, educators, employers, workers, and marketers. Clifford Nass, a professor of communication at Stanford noted, “If you mention multitasking, people go insane—it’s all they want to talk about”. He described the problem of multitasking as “a challenge to human cognition” (Ophir et al. 2009).

1.3.1 Multitasking and the Brain

To better understand how distraction relates to multitasking we will explore a few aspects of neuroscience and our mechanical sensory capacities. To interpret multitasking, we need to consider the brain’s attention networks underlying our ability to switch tasks (Rothbart and Posner 2015, p. 3). Neuroimaging has recently revealed that even subtle shifts in tasks activate neural areas. The cerebellum has two areas of operation one that uses sensory signals and the other motor signals. In effect, the cerebellum is our *motor* for learning–particularly when it comes to learning new motor skills (Hatten and Lisberger 2013, p. 2). The cerebellum is capable of plasticity[2] allowing neurons to communicate with one another (this is a simplified explanation) in dynamic ways. Generally, the mechanical and sensory portions of the brain operate together as long as only one task is involved. However, introduce more than one task and communication between these parts begins to break down resulting in the grave consequences as demonstrated by driving and texting (Kramer et al. 2007). Similarly, most of us have experienced ‘going on autopilot’ and driving from one destination to another without being able to fully recall the trip. This phenomenon is experienced since we were likely thinking about something else during the mechanical process of driving–the learned mechanical process of driving has been saved to memory. However, introduce another mechanical process, say picking up a mobile, or a third–using ones’ fingers to text, and even a fourth composing a text, and you have a recipe for disaster–the entire efficiency of the process is significantly diminished. Evidence of this can be seen in the United States where nearly half a million people were injured or killed in accidents involving this combination of texting and driving (Highway Traffic Safety Administration and Department of Transportation 2016).

1.3.2 Action-Based Learning

The learning environment of the classroom, has both sensory and mechanical parallels as described above and where the brain is concerned, but with much less catastrophic multitasking consequences. Impaired listening or attention are often obvious

[2]For a further explanation of adult cognitive plasticity (see Lövdén et al. 2010).

to those trying to convey information to students who may be generally unaware that they are missing much of what is being said. Recent research in neuroplasticity and learning suggests that a simple physical movement may activate the hippocampus in ways not previously understood (Cassilhas et al. 2016, p. 168). This discovery is significant concerning Action-based learning (ABL) approaches since movement supports how the brain connects to preparing itself to learn. ABL is a process or pedagogy of brain activated learning linked to the action of motor skills. This approach fits in well with the learner requiring greater stimulus yet it has been observed that ABL is rarely discussed as a potential solution or even partial solution to the problem of distraction or inclination to excessively multitask in the classroom. (An omission that is addressed in this volume in Diane Kitchin's chapter on active learning.)

The problem of how we help students manage or break the cycle of multitasking in class may be diverted or rewired using methods like ABL. ABL requires substantial changes to the way lectures are planned and executed. The current state of most lecturing methods, where a long talk is involved is yet another reason why lectures are becoming less able to facilitate learning and why ABL has come to the fore as one potentially rich approach.

1.3.3 Gen Z and Boredom

This year we will see our first Generation of students born between 2000 and the present. Generation Z (Gen Zers or Gen Z) has arrived in higher education. This generation was born into an Internet-connected world, has grown up with the smartphones, and may have spent the past decade using many social networks. The Gen Zers are a generation that prefers communicating through social media over direct communication. For the Gen Zers,[3] waiting is not much of an option and they are conditioned to pick up their smartphone or device to find a rapid release from boredom. Since the arrival of the smartphone waiting in lines at the store or for a train have become less of a problem. We can fill that time perusing the news, checking our social networks and email. Gen X and Zers use technology to 'personalise everything', they are technologically skilful and prefer Web applications and email (Reisenwitz and Iyer 2009, p. 91).

It seems logical that if students are physically active and working towards a goal or a solution to a problem they will be less likely to stop, pick up their phone and check Facebook–they will be less likely to want to interrupt their processes due to boredom.[4] This generation gets bored fast and the antidote to a nice hit of dopamine is to check into social media. It activates them, and physiologically this generation

[3]Our survey found 55% of students multitasked due to boredom. 62% identified lecturers reading from slides as another cause for multitasking during formal lectures.

[4]Our survey found 55% of students multitasked due to boredom. 62% identified lecturers reading from slides as another cause for multitasking during formal lectures.

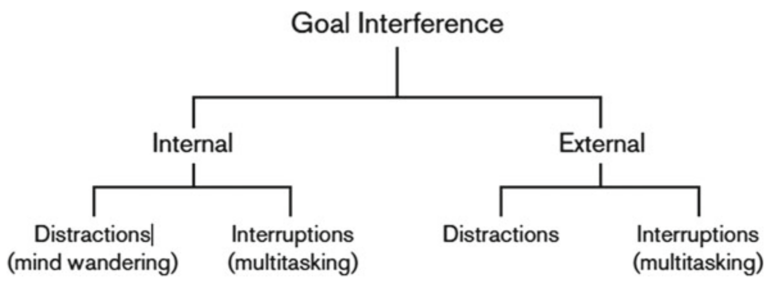


Fig. 1.1 Gazzaley's conceptual framework for goal interference (Mishra et al. 2013) and adapted from Clapp and Gazzaley (2012)

has become accustomed to multitasking in this way in the same way that we would probably receive a similar hit from eating something satisfying.

1.3.4 Cognitive Systems and Control

Cognitive control and its functions are central to the concept of multitasking. Although we cannot pursue this in depth in this chapter, some basic concepts are considered. Gazzaley breaks this down into internal and external factors about interference [see Fig. 1.1 (Gazzaley and Rosen 2016)]. Interference represents those things that distract and interrupt us whether of our internal making or externally driven. The brain is a complex information processing system. As a system, it is structured and optimised for performance. Again, in Fig. 1.1, Gazzaley shows how goal interference competes with internal and external factors as we try to achieve our aims.

Students, however distracted, are just trying their best to achieve their aims with often incomplete information about how to manage themselves. Perhaps there is a need to help them understand how they can optimise their work through understanding some of the concepts around multitasking. As discussed earlier, it could be argued that they are living in a more distracted environment than existed a decade ago. Of course, experience and management of goal interference will likely swing widely between the individual depending on countless variations. However, several areas can be supported in the classroom by adjusting our teaching methods, by considering recent research, and by embracing rather than negating technological changes.

1.3.5 Confirmation Bias and Supertaskers

Another issue brought up in our survey and anecdotally with first-year students was the role confirmation bias plays alongside multitasking. Over the past two years, there has been a higher proportion of students who believe they are supertaskers capable of rapid attention shifting with devices in what they regularly believe is efficiency. It is often talked about as a sought-after skill. It is true the way students often interact with a keyboard and respond to screen-based information is–fast. Many studies have tested the supertasker phenomenon (Watson and Strayer 2010; Nicholas 2010). However, current laboratory research still asserts that simultaneous task performance suffers during multitasking (Dux et al. 2009; Garner and Dux 2015). The problem is that speed and fluidity do not necessarily translate into the ability to apply and learn new skills. Even more problematic, how do we help students to understand this when they believe what they have in a sense become indoctrinated into–a cult of speed and freedom of unfettered access. Furthermore, studies of the brain have shown (Watson and Strayer 2010; Strayer and Watson 2012) there are only at most 2% of individuals capable of multitasking or able do more than one thing at a time efficiently. However employers also seem to believe multitasking is a sought-after skill and regularly advertise for it in programming jobs.[5] Also, students see other students with similar behaviours in class and come to believe that doing more than one thing at once is either expected, normal or both, further exacerbating this problematic issue.

1.3.6 Academic Writing: A Bridge Too Far

Writing is a higher order learning skill. It is also an area where academics have seen significant and growing difficulties for students. It is possible that the rise of the essay mills may well be related to the problems students are facing having to write an extended academic paper. If, as mentioned earlier many students are experiencing a reduced depth of processing, increasing stress levels including anxiety due to multitasking, then their ability to invoke creative problem solving will ultimately be hampered (Firat 2013). We are finding that fewer students are often only capable of shallow focus work (Loh and Kanai 2015; Nicholas 2010) leaving them unable to tackle harder work requiring greater cognitive power and focus. So it is not only focus, since academic writing is a difficult task that requires deeper thinking and higher cognitive skills than what current students spend most of the time doing both inside and outside the higher education environment. These problems become most visible towards the end of their course when they are asked to develop a dissertation, a large piece of writing requiring work over an extended period. Students seem less prepared for this challenge, and we need to do more to assist them constructively. It is likely that a spectrum of variables are at work here from brain and neurological functioning to insufficient awareness and ability to manage distractions effectively. Throw on top of this the inability to manage and focus attention in the sea of the increased use of social media all these factors are contributing to the problem.

[5] Searching the word "multitasking" site: indeed.com and "multitasking" site: indeed.co.uk show a difference of 73,300 US compared to 6760 UK. This may suggest a difference in educational and employment emphasis. It could also be just a reflection of population differences.

1.4 Debating the Banning of Laptops During Lectures

Moving now from our increased understanding of why students multitask and how the brain functions we can further explore the impact these conditions are having in the classroom. One of the standout factors aligned to multitasking in the classroom is social media usage. In 2005 Facebook and MySpace were launched, closely followed by Twitter and YouTube in 2006. The exponential shift happened a few years after the launch of smartphones, specifically the iPhone in 2007, and the 2008 launch of the Android mobile operating system. It then took only a few years for mobile computing to appear in classrooms where more than half the class were in possession of a mobile. In 2010 only a few students had smartphones, but by about 2013 the increase had become pronounced and new problems around attention and distraction were commonplace. By 2015 virtually every student had a smartphone in the classroom and often they had more than one device. This fast pace has put stresses on the higher education system and our ability as teachers to adapt our methods at pace with these changes. Combine this with the continuing exponential growth of social media usage, and one has a perfect storm.

By the fall of 2016, social media usage amongst university freshman in the United States averaged over six hours a week, an increase of over 40.9 or 27.2% greater than in 2011 and 2014 (Eagan et al. 2017, p. 20). There were over 10 million participants in this survey. Being an election year in the United States may have had some impact on this data. However, if social media continues to increase at a similar rate, what changes are likely in the classroom? Will increased usage of social network sites (SNSs) amongst students increase distraction and attention levels in the classroom and if so how will we adapt our methods? It is essential to develop strategies to improve engagement in this changing environment as well as considering both cognitive and information systems models as part of that development. As professors and lecturers across the globe experiment with various approaches to control these relatively recent changes, we see both extreme and light touch reactions. One wing demonstrates only a modest understanding of the collision of human-to-human and human-computer interactions at play. For example, Seth Godin[6] taking an oppositional stance towards Susan Dynarski, a professor at the University of Michigan. Dynarski published an op-ed in the New York Times stating that she has forbidden students from using laptops in her lectures (Dynarski 2017). Godin believes Dynarski has missed the point altogether. According to Godin, Dynarski is laying the blame in the wrong place by asking students to slow down their clock speed and listen attentively in addition to notetaking—all at the same rate. He argues this is unreasonable to expect this given the technological changes in recent years and lays some blame on universities not adapting quickly enough. Godin states "the solution isn't to ban the laptop from the lecture it's time to ban the lecture from the classroom" (Godin 2017). He also believes the lecture should be digitally recorded so students can review it, as and when they choose to. However, the problem may not require institutions to do away with the

[6]Seth Godin is a well known entrepreneur, bestselling author, writer and marketing and leadership blogger.

lecture hall, and it is worth considering the possibility of something in between these two somewhat extreme ends of the spectrum. Shorter lectures formed of no more than five to seven minutes followed by activities to discover information closely related to the presentation may be more motivational and engaging. The traditional 45–60 minute lecture is still currently the norm but is unsustainable given the changing environment. There are a number of arguments against banning laptops, not least of whether such a ban would be compatible with an ethos of open education and how such a ban might be enforced. There is the question of potential discrimination against students with disabilities, or if some students were allowed laptops to support their disability, discrimination against students without disabilities. Furthermore for "Zers" a laptop or smartphone may be the most efficient way to take notes and to instantly look up additional information. Some research suggests that students who multitask using their laptop during lectures perform less well compared to those that do not (Sana et al. 2013). However, one must ask—if students had more advice on *how* to take notes optimally, would this study still be valid? The early days of email usage in the mid-nineties had a pretty steep learning curve and compared to numbers of technologies and applications we have now it seems an almost silly comparison, yet we all struggled with learning how to manage email. Academic staff misused and overused the medium while simultaneously bemoaning the extra workload. We may have to consider students similarly don't know *how* to manage their devices optimally to improve their performance. Sana's study above was only investigated with forty participants. A limited sample suggests a need for a more comprehensive study that also considers using an intervention method as a control group and then comparing the data similarly to a study undertaken at Ryerson University (Tassone et al. 2017, p. 1).

1.4.1 Note-Taking

The research on note-taking goes back to the 1960s where there was considerable debate about how and when to listen and take notes (Eisner and Rohde 1959). It is worth having a brief look at how note-taking fits into the multitasking debate. Many researchers believe that taking notes on a laptop will impair performance compared to those who take notes longhand (Mueller and Oppenheimer 2014, p. 1; Bellur et al. 2015, p. 65; Fried 2008, p. 47). The problem is not the technology or mandating rules to comply with it. The problem is more precisely that students need assistance managing the interplay of these issues. Generally, most studies tend to support a rule or discipline-based solution in the classroom more or less finding fault with the student, the technology or the social media networks and default towards asserting that students must follow "proper rules […] and abide by these rules" (Anshari et al. 2017). This approach mainly describes the problem but misses the importance of considering a model sensitive to context, changing cognitive conditions and human-systems design persistently shaping behaviour and influencing human evolution.

1.5 Smartphone Dependencies

Dependency on smartphones and academic performance form another area aligned with variables contributing to multitasking. The plethora of research over the past decade on this topic (Samaha and Hawi 2016; Junco 2012, pp. 505–514) is well documented. Students are often in a state of discomfort having to turn off or look away from monitors or devices during the formal part of a lecture. There are many issues at work here. Firstly, students have become used to large amounts of visual activity and stimulus with the average 19-year-old checking their phone every ten minutes. Secondly, most students have had a smartphone for at least five years or more and lived in a context where these technologies have been an inseparable part of their daily lives. The smartphone has become an object of instant gratification, a quick fix for boredom and has neurologically altered their brains and consequent behaviours. Often this is leading to a form of addiction (Terry et al. 2016, p. 245). We can now confirm this has changed our students' brains having grown-up in tandem with smartphones and mobile computing (Loh and Kanai 2015, pp. 2–3). If we can accept this, then much of what has been discussed in this chapter should begin to make sense. With this in mind try to imagine what a student would be experiencing in the average university classroom. Imagine how frustrating it would be to sit for extended periods while the lecturer reads from slides. This approach still occurs in many lecture halls in both the United States and the UK. The lecture format will likely not keep students engaged unless it is short (5–7 min), targeted and has a specific outcome followed up quickly by an information consolidation activity. So, we currently have a problem, and it is not with the student—we are missing opportunities to create engagement in the classroom.

1.6 The Survey

A survey on multitasking was carried out between March 21st–31st 2017 on a cohort of 60 undergraduate students taking a first-year, core, web development module. The students were asked to describe their multitasking habits during formal lectures. The study aimed to discover perceptions about multitasking behaviour.

A Likert scale was used for 22 questions. A 23rd question asked if they would like to share their thoughts. The Likert scale was especially useful for establishing some evidence of a possible correlation between high percentages of neutral answers and whether questions were either too broad or vaguely stated. (The detail of these results has not been included.) The highest neutral score was 42% for the question: *I believe multitasking during lectures is a smart thing to do.*

1.6.1 *Intrinsic Questions*

Four questions were similar for a reason. These were questions about whether participants would change their minds about multitasking. 60% were willing to change their minds if multitasking proved to them it could: lower or improve their grades (66%), harm their learning (60%) or improve their learning (48%). 55% believed they could get more done with 43% thinking it made them more efficient.

1.6.2 *Extrinsic Questions*

Just 58% of the students said they were using one or more devices to multitask during their formal lectures. This result is generally in line with other studies. The reason for this appeared to be that they felt they could get more done 55%, while 62% said they multitasked because lecturers were reading from slides, while 55% said their multitasking was due to boredom during the lecture. In some ways, this is encouraging as a change in teaching approach may result in more active or participatory learning. No students felt any pressure to multitask by their lecturers (0%).

1.6.3 *Employability*

In 2012 at the CASBS summit, Clifford Nass stated: "companies now create policies that force their employees to multitask". In our study, just 11.7% thought multitasking would make them more employable. This result demonstrates an opportunity to raise awareness amongst students for employability purposes. Oddly, 40% said they believed multitasking to be an essential skill. There has been an increasing frequency 'multitasking' appearing in job posts for software developers. This response is interesting despite evidence multitasking skills are often sought by employers. However, there is a difference in emphasis between the United States, and the United Kingdom in this regard. Oddly respondents did not consider multitasking to be an employability factor as highlighted in some research (Burak 2012; Crenshaw 2008).

The survey shows some evidence that computer science students in the UK have varied views on whether multitasking during class lectures is positive or negative. Though one comment did not see the point of the survey or why their views about it would be interesting. This response suggests students need more information about this for their continuous and focused information-seeking behaviours about multitasking. Similarly, lecturers may want to alter teaching methods to reflect the changed cohort as mentioned earlier. Students also appear to want the facts about multitasking as there seemed to be some slippage between what they believe and what may help them in their studies and professional life.

1.7 Conclusion

Early on in this research project, it became apparent that the study needed to expand beyond issues of education and therefore consider the recent advances in the cognitive neurosciences and cognitive psychology. It also became clear that media multitasking and its effects have been investigated exhaustively in many ways. "The problem of how the brain undertakes multiple tasks concurrently is one of the oldest in psychology and neuroscience" (Verghese et al. 2016). What has been offered in this chapter is the breadth and depth of the challenge ahead and to some extent behind us as mediators in the classroom. Further advancements and changing frontiers in the sciences are still being discovered and how much Gen Zers brains have been altered is becoming apparent. However, as Susan Greenfield asserts "the brain is exquisitely adaptable" (Greenfield 2015) and further research will likely bring enhancements possible for our ongoing adaptation concerning information foraging. It is also possible that with these advancements there will be more 'supertaskers' among us (Strayer and Watson 2012). Video games are an indication of this and have been shown to be highly beneficial to multitasking particularly with older participants (Mishra et al. 2016). These developments indicate not all aspects of multitasking mean poor performance as some researchers assert (Bellur et al. 2015, p. 65). Changes are underway that will continue to test us as educators though, and students will require specific and targeted guidance about the risks and benefits of multitasking as they manage their courses, careers and lives. However, I would suggest that there is one conclusion we can certainly draw. Multitasking is prevalent, and it is here to stay. We can either choose to rile against it, or adapt our methods to accommodate it. Accommodation would seem to be the more productive approach. It might well be worth considering how best to incorporate this changing learning environment into our teaching.

References

Anshari M, Almunawar MN, Shahrill M, Wicaksono DK, Huda M (2017) Smartphones usage in the classrooms: learning aid or interference? Educ Inf Technol 22(6):3063–3079

Bellur S, Nowak KL, Hull KS (2015) Make it our time: in class multitaskers have lower academic performance. Comput Hum Behav 53(2015):63–70

Burak LJ (2012) Multitasking in the university classroom. Int J Sch Teach Learn 6(2):1–13

Cassilhas RC, Tufik S, de Mello MT (2016) Physical exercise, neuroplasticity, spatial learning and memory. Cell Mol Life Sci: CMLS 73(5):975–983

Clapp WC, Gazzaley A (2012) Distinct mechanisms for the impact of distraction and interruption on working memory in aging. Neurobiol Aging 33(1):134–148

Crenshaw D (2008) The myth of multitasking: how "doing it all" gets nothing done. Wiley
Dux PE, Tombu MN, Harrison S, Rogers BP, Tong F, Marois R (2009) Training improves multitasking performance by increasing the speed of information processing in human prefrontal cortex. Neuron 63(1):127–138
Dynarski S (2017) Laptops are great. But not during a lecture or a meeting. The New York Times. Accessed 23 June 2018
Eagan K, Stolzenberg EB, Ramirez JJ, Aragon MC, Suchard MR, Hurtado S (2017) The American freshman: national norms fall 2016. Los Angeles, Higher Education Research Institute, UCLA
Eisner S, Rohde K (1959) Note taking during or after the lecture. J Educ Psychol 50(6):301–304
Firat M (2013) Continuous partial attention as a problematic technology use: a case of educators. J Educators Online 10(2):1–20
Fried CB (2008) In-class laptop use and its effects on student learning. Comput Educ 50(3):906–914
Garner KG, Dux PE (2015) Training conquers multitasking costs by dividing task representations in the frontoparietal-subcortical system. Proc Nat Acad Sci 112(46):14372–14377
Gazzaley A, Rosen L (2016) The distracted mind: ancient brains in a high-tech world. MIT Press
Godin S (2017) No laptops in the lecture hall. Medium. Available at https://medium.com/@thisissethsblog/no-laptops-in-the-lecture-hall-1847b6d3315. Accessed 23 June 2018
Greenfield S (2015) Mind change: how digital technologies are leaving their mark on our brains. Random House Incorporated
Hatten ME, Lisberger SG (2013) Neuroscience: Multitasking on the run. ELife 2:e00641
Junco R (2012) In-class multitasking and academic performance. Comput Hum Behav 28(6):2236–2243
Kramer A, Wiegmann D, Kirlik A (2007) Attention: from theory to practice, vol 4. Oxford University Press
Krebs J (1977) Optimal foraging: theory and experiment. Nature 268:583–584
Loh KK, Kanai R (2015) How has the Internet reshaped human cognition? The Neuroscientis 22(5):506–520
Lövdén M, Bäckman L, Lindenberger U, Schaefer S, Schmiedek F (2010) A theoretical framework for the study of adult cognitive plasticity. Psychol Bull 136(4):659
MacArthur RH, Pianka ER (1966) On optimal use of a patchy environment. Am Nat 100(916):603–609
Mishra J, Anguera JA, Gazzaley A (2016) Video Games for neuro-cognitive optimization. Neuron 90(2):214–218
Mishra J, Anguera JA, Ziegler DA, Gazzaley A (2013) A cognitive framework for understanding and improving interference resolution in the brain. In Progress in brain research, vol. 207, pp. 351–377. Elsevier.
Mueller PA, Oppenheimer DM (2014) The pen is mightier than the keyboard: advantages of longhand over laptop note taking. Psychol Sci 25(6):1159–1168
Nicholas C (2010) The shallows: what the Internet is doing to our brains. Inc., WW Norton & Company
Ophir E, Nass C, Wagner AD (2009) Cognitive control in media multitaskers. Proc Natl Acad Sci 106(37):15583–15587
Pirolli P (2007) Information foraging theory: adaptive interaction with information. Oxford University Press
Pirolli P, Card S (1995) Information foraging in information access environments. In: Proceedings of conference on human factors in computing systems, pp 51–58. ACM Press/Addison-Wesley Publishing Co
Reisenwitz TH, Iyer R (2009) Differences in generation X and generation Y: implications for the organization and marketers. Mark Manage J 19(2)
Rothbart MK, Posner MI (2015) The developing brain in a multitasking world. Dev Rev 35:42–63
Samaha M, Hawi NS (2016) Relationships among smartphone addiction, stress, academic performance, and satisfaction with life. Comput Hum Behav 57:321–325

Sana F, Weston T, Cepeda NJ (2013) Laptop multitasking hinders classroom learning for both users and nearby peers. Comput Educ 62:24–31

Sanbonmatsu DM, Strayer DL, Medeiros-Ward N, Watson JM (2013) Who multi-tasks and why? Multi-tasking ability, perceived multi-tasking ability, impulsivity, and sensation seeking. PloS One 8(1):e54402

Statista (2017) Forcast of smartphone user numbers in the United Kingdom (UK) from 2015–2022. Available at https://www.statista.com/statistics/270821/smartphone-user-in-the-united-kingdom-uk/

Strayer DL, Watson JM (2012) Supertaskers and the multitasking brain. Sci Am Mind 23(1):22–29

Tassone A, Liu JJ, Reed MJ, Vickers K (2017) Multitasking in the classroom: testing an educational intervention as a method of reducing multitasking. Active Learn High Educ. https://doi.org/10.1177/1469787417740772

Terry CA, Mishra P, Roseth CJ (2016) Preference for multitasking, technological dependency, student metacognition, & pervasive technology use: an experimental intervention. Comput Hum Behav 65:241–251

Highway Traffic Safety Administration N, Department of Transportation U (2016) Traffic safety facts driver electronic device use in 2016 (June 2017), pp.1–8. Available at https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/812426

Verghese A, Garner KG, Mattingley JB, Dux PE (2016) Prefrontal cortex structure predicts training-induced improvements in multitasking performance. J Neurosci 36(9):2638–2645

VitalSource (2015) Students Want More from Classroom Tech survey reveals. Available at https://www.vitalsource.com/press/fifth-annual-vitalsource-wakefield-survey-finds-college-students-want-more-and-better-classroom-technology. Accessed 2 April 2017

Watson JM, Strayer DL (2010) Supertaskers: profiles in extraordinary multitasking ability. Psychon Bull Rev 17(4):479–485

Chapter 2
Active Learning in Large Lectures

Diane Kitchin

Abstract An increasingly diverse student body combined with pressures to demonstrate excellence in teaching and improve results presents challenges for lecturers. Active learning techniques have attracted interest and discussion amongst educationalists. This chapter investigates the challenges and gives a practical guide to techniques that have been used effectively in a large lecture situation with first year students.

Keywords Active learning · Large lectures · Computing education
Constructivism

2.1 Introduction

Changes in higher education over recent years have seen a rise in the number of students going to University and consequently a more diverse student body. This has led to a need to help students make a smooth transition between school and University and to adjust to different environments, different delivery styles and different expectations. With increased competition for students amongst Universities, and the new demands of the Teaching Excellence Framework (TEF), comes increased expectations on lecturers—we must ensure our students pass our modules, and moreover pass them with good marks. Excellent and effective teaching is seen as a key factor by Universities in attracting students to their courses. The University of Huddersfield mentions the quality of its teaching in four out of six of the factors that demonstrate its excellence on its 'About us' page for new and prospective students. (University of Huddersfield 2018). Active learning pedagogies was one of the themes identified by the Higher Education Academy (HEA) in its review of the written submissions Universities included as part of their TEF documentation, in support of the assessment criteria for Teaching quality, Learning environment and Student and outcomes learning

D. Kitchin (✉)
University of Huddersfield, Huddersfield, UK
e-mail: d.kitchin@hud.ac.uk

J. Carter et al. (eds.), *Higher Education Computer Science*,
https://doi.org/10.1007/978-3-319-98590-9_2

gain. (HEA 2017). In the HEA report on TEF 2 the authors state that "Course design comes out as a prevalent aspect for providers upgraded to a Gold award …. Features mentioned in some statements included: … use of active learning…" (Moore et al. 2017).

Additional challenges to providing an active learning experience are that group sizes are often large, particularly for first year teaching, where lectures may be delivered to groups of 150 students or more. At Huddersfield with this diversity in the student body we have seen an increase in the number and level of student support initiatives. We have also seen a shift in the type of learning activities available, with more studio work, project work and group work. Research in approaches to learning advocates a more learner-focused approach in teaching, with active learning being much-discussed over recent years.

In this chapter we focus on the practical techniques adopted to overcome these challenges and foster an active learning environment, particularly in a large lecture situation. The remainder of this chapter is structured as follows: Sect. 2.2 discusses the challenges and motivation for this work in more detail; Sect. 2.3 describes active learning and reviews some of the literature; Sect. 2.4 describes the specific, practical techniques used to deliver active learning in a large Computing Science and Mathematics lecture; Sect. 2.5 reflects on outcomes and the use of these techniques and looks at possible further development and Sect. 2.6 gives a summary and presents our conclusions.

2.2 Challenges and Motivation

The specific context for this discussion is the delivery of a Computing Science and Mathematics module to first year students. This is a long-running module, which has existed in some form or another for many years, and which has been delivered by me for almost 10 years. It covers topics such as set theory, graphs and trees, propositional logic, sorting algorithms, Finite state automata, grammars and languages, regular expressions, binary search trees and tree traversal algorithms. It is typically taken by a very large group of around 100–170 students. Invariably, there will be a very wide range of abilities in each group. Some students won't have done any Maths since GCSE two or more years previously and may only have a grade C. Others will be very able students who have studied A-level Maths or Computing or both, and so they may have covered some of the topics already. There will also be a small number of mature students, who could have been out of formal education for a number of years, and who may be nervous about the subject. There will also be a number of international students, who may have had very different educational experiences. Again some of them will have covered similar material before, while others may not. A large mixed-background, mixed-ability group presents many challenges. How can we keep the attention of and give new challenges to students who are familiar with a topic, whilst not overwhelming and alienating students who view the material as difficult and possibly scary?

There are a number of barriers to understanding in such a group. The cultural backgrounds and life experiences of students can vary widely, so I always try to bear in mind that not all examples will be appropriate for such a mixed audience. For example, once when teaching Java classes and inheritance to a mixed group of students, using what seemed to be a simple Bank Account example, I was puzzled as to why one of the international students was struggling to understand the material. It turned out to be because he was unfamiliar with the meaning of terms like Current Account, Savings account etc., rather than the programming language concepts. Some students may have been taught just to accept and learn what they are told, and will not have been encouraged to ask questions. Also many students just assume that maths is difficult or that they can't do it, and often don't recognise the difference between the general maths they studied at school and the discrete maths relevant to computing. With such a large and diverse group, it is important to keep all the students interested and engaged, and to make sure that the teaching materials don't provide an unnecessary barrier to learning or exclude any students. Even small things can make a difference—such as not choosing exclusively male or white British names in examples. So a simple set theory example of a set of students = {Bob, Tom, Harry} could make a student feel excluded if the examples all follow a similar pattern.

2.3 Active Learning

In the past, traditional lectures were often viewed merely as a vehicle to convey a large amount of information to a lot of students at one time. Students were expected to be largely passive, just listening and taking notes, and perhaps occasionally asking or answering questions. Research has shown that audience attention in lectures begins to wane every 10–20 min. This traditional approach has been questioned by educationalists and researchers, with theories such as Constructivism (Bruner 1960, Piaget 1950, and others) advocating active learning.

Constructivist theories see learning as an active process, whereby students build on and adapt existing knowledge or knowledge structures in the mind known as 'schemata', so developing and deepening their understanding. Knowledge isn't something that exists in isolation, it requires a context and connections to what we already know, so when we learn we are not just adding new information, we also have to make sense of it and give it meaning in relation to existing knowledge. "Learning is thus an active process of individual transformation and changes in understanding" (Fry et al. 2015, p. 65). Acceptance of this theory entails a change in the way teachers think about teaching and how they do it. Constructivists argue that the teachers must encourage a deep approach to learning, with context and expectations being set by the teacher. In addition, some researchers emphasise the importance of learning outcomes, arguing that the curriculum should be constructively aligned (Biggs and Tang 2009) with the teaching environment and assessment, and that students should be made aware of and understand these learning outcomes.

The question now arises “what exactly is active learning?”. There are many different definitions in the literature. The term active learning originates in the work of Revans (1971). Prince (2004) offers a simple and useful explanation, defining “the core elements of active learning to be introducing activities into the traditional lecture and promoting student engagement”.

Bonwell and Eison (1991) define active learning as “instructional activities involving students in doing things and thinking about what they are doing.” According to Weltman (2007) the activities can range from the very active, such as physically making something to doing something like playing a game or to the less-involved such as watching a video, and then applying what has been learned from this in some way. A commonly used lower-level example of active learning could just be allowing time in a lecture for students to think about or discuss the material presented. The argument is that since students can only concentrate effectively for about 15 min, introducing pauses two or three times in a lecture, or having some change of activity or focus is beneficial and a simple way of making the learning experience more active. Other commonly mentioned methods include problem based learning, and cooperative and collaborative learning.

We should also ask, before adopting an active learning approach, what evidence there is for its effectiveness. Weltman (ibid.) cites various studies (Raelin and Coghlan 2006; Sarason and Banbury 2004; Sutherland and Bonwell 1996; Umble and Umble 2004) which have found supporting evidence for the effectiveness of active learning techniques. According to a review by Prince (2004) there is evidence in favour of active learning. He cites Bonwell and Eison (1991) who “conclude that it leads to better student attitudes and improvements in students’ thinking and writing”. His review includes the work of Felder et al. (2002) who recommend active learning as “one of the teaching methods that work”. However, Prince also points out that the support for active learning from the literature he reviews is not conclusive and the improvements may only be small. Prince also reviews empirical support for active learning and concludes that “introducing activity into lectures can significantly improve recall of information while extensive evidence supports the benefits of student engagement” (Prince 2004, p. 4).

2.4 Techniques and Practices

As previously mentioned, the techniques, practices and examples given in this section are all taken from a first year Computing Science and Mathematics module. The material used has been built up over a number of years by colleagues who taught the module previously (Ron Simpson, Barbara Smith, Lee McCluskey and John Turner) and myself. A lot of the material is theoretical, technical and factual, with little room for individual interpretation. Nevertheless, students have to build up their own understanding and mental models. So the aim is to make the lectures a mixture of factual delivery combined with various active learning techniques, to encourage engagement and facilitate learning.

The active learning techniques which are used are:

a. Motivation with real applications
b. Worked examples on a Visualiser
c. Students work on further examples in the lecture
d. Animations and applets
e. Short videos
f. Active participation and enactment.

Examples of the use of all of these techniques will be given in the remainder of this section.

Often in large lectures students are largely passive—just listening and perhaps taking notes. From the start of the term students are encouraged to be active learners in these lectures, participating in exercises and even activities. They should not think of themselves as empty vessels passively waiting to be filled with information and knowledge, but as active participants in a process. So it is made clear from the first week that they will be expected to take an active part in lectures, to bring pen and paper each week and to ask and answer questions.

Some students will see Maths as difficult, or a topic that they have previously found challenging, and some won't have done any computing science at all. They often don't initially see the relevance of theoretical material to their studies—"what has maths got to do with computing?" they may ask. It is important to build their confidence and to give them some motivation for learning what they may perceive to be hard and unnecessary. Thus a good starting point for every lecture is an example of a practical application within the real world or from computing of the particular topic to be studied that week. It also helps give some context for what may be completely new material. This, in terms of constructivist theory, helps them to begin to build or adapt a knowledge structure in their minds. For example, few students will have studied graph theory and will not link it with computing. Some concrete examples of the application of graph theory that could be given would be for modelling a network configuration or modelling dynamic processes, while trees are commonly used data structures they will meet in computing. For Finite state machines (FSM) a computer game would make a good example—where the states of the machine represent the states of a games, something almost all student will be familiar with. The states represented could be "explore", "attack", "evade" and so on with transitions between states labelled "player nearby", "no player in sight", "player attacks", "player idle" etc.

Once the context, motivation and applications are clear, these examples can then be followed by some definitions of the terminology or concepts being introduced. Wherever possible each concept or item should be accompanied by an example or diagram—e.g. vertex, edge, graph, tree. As with all written examples the Visualiser is used for this. In logic a simple example of a proposition is given—maybe something intriguing or amusing—e.g. an example of a simple proposition could be: "My real name is Trillian" (from Hitchhiker's Guide to the Galaxy), or perhaps something more current, but the idea is to choose something that is familiar to them from the everyday world. The students can then be asked to think of and write down their own example, perhaps something relating to them, and either share it with the person next to them or tell the whole group, when asked.

The general pattern followed is exposition, example, activity, so, as soon as possible the students are given a small example or problem to work out. For example, when teaching set theory, after the concept has been explained and an everyday example given, they can then be asked to raise their hand if they are in the set of students with brown hair. Then the set of students with blue eyes, then the intersection (i.e. if they are in both sets) and so on. Another everyday example used is something like the "Who am I game" where students are shown a collection of pictures of men and women, with and without hats, glasses and beards and asked to determine the members of various sets, and the results of applying simple set operators. The idea is that the theory is more likely to be retained if they've been physically involved by putting their hand up, or writing a simple sentence etc.

As the lecture progresses the examples and exercises become increasingly technical. After introduction of all basic definitions and concepts I do a simple worked example with the students. I prefer to use a Visualiser for this rather than writing on a white board, because it is clearer for the students to see, as it is projected on to the main screen, and also I can still look at and talk to them while doing the example, rather than turning my back to them.

Two or three problems of increasing difficulty may be given for the students to work on. For example, three sentences in English to be translated into propositional logic, or using inference rules to prove logical arguments. I always do at least 1 or 2 problems with them first and then let them have a go on their own. Alternatively, if the exercise is more difficult or time-consuming, such as a more complex FSM, then I would talk through it, but involving the students in its construction, asking them what to do next at each step—e.g. which edge or label should be added next, where it should go etc., so continually encouraging involvement and engagement, and discussing why a particular suggestion might or might not work. I might ask "would adding a particular transition as a loop allow the generation of invalid strings, which would mean we should add a new state instead"? It is important to keep checking for understanding—encouraging a culture of asking questions (I say I don't expect them to understand everything immediately and so I'm worried if they're not asking questions).

For the more able students, hopefully the lecture serves as a reminder or refresher if they have covered the material before, and also a way to help them reinforce or add to what they already know. More challenging questions can be included in the tutorial exercises. These students often are more confident in answering questions or volunteering information in the lecture, which can help weaker students who might be reluctant to speak up.

A lecture on recursion will now be used as an example which incorporates many of the active learning techniques mentioned at the start of this section. The learning outcome is that students should develop an understanding of the working of a recursive algorithm—which then leads on in following weeks to understanding different recursive algorithms used in sorting or tree traversal. It begins with an everyday example that should build on existing knowledge. The context is them widened with examples from computing and maths to help motivate further learning. Eventually there is a physical enactment of the algorithm by the some of the student followed by a more formal worked example.

The example from everyday life to introduce the idea of recursion is walking across a moor in the fog. If we just try to find our way across the foggy moor by attempting to walk in a straight line in one direction, we can easily get lost and end up walking round in circles. However, if we use our compass to identify a small feature or landmark about 100 m ahead in the right direction, such as a rock or hillock, which we then walk to, and keep following this strategy, we have broken our problem down into a smaller one which we can solve easily (walking 100 m). We can keep applying this method to the smaller problem that remains, as we are now 100 m close to our destination. We can write this as an algorithm for crossing the rest of the moor:

- If you can see your destination, walk to it;
- If not,
  - choose a landmark in the right direction 100 m further on and walk to it;
  - cross the rest of the moor.

This is a recursive algorithm; we have defined crossing the rest of the moor in terms of… crossing the rest of the moor.

A useful way to change the focus of student attention is to use short animations or applets within a lecture to illustrate a particular concept. This lecture next moves on to another example where recursion can be used—the Tower of Hanoi. There is a dynamic application showing the Tower of Hanoi (Dynamic Drive). This puzzle has three pegs and a number of discs of graduated sizes, each with a hole in the middle so that they can be placed on a peg. The aim is to move the tower of discs from one peg to another. Although we could physically move them in one go, this is not allowed by the rules of the puzzle:

- We can only move one disc at a time.
- We can only put a disc on top of a larger disc, or on the base board.

This puzzle according to legend, originates in a monastery in Hanoi (Vietnam), where the monks are engaged in moving a tower of 64 discs from one peg to another; when they have finished, the legend says that the world will end.

The application allows the user to select the number of starting discs, and then to move them one at a time from one peg to another. It also shows the minimum number of moves and the number of moves made by the user. I usually run this starting with three discs, moving them from the start peg to the finish peg, and then increase the number of discs by one on each attempt, trying to keep to the minimum number of moves. This can be quite stressful for the lecturer, once the number of discs increases, but you may be rewarded with a round of applause if you can do a sufficiently impressive number of discs without making a mistake—six is a good number to aim for! The students can also have a go at this themselves after the lecture or in one of the tutorials, to help increase engagement with the topic.

The demonstration can then be followed by showing the pseudocode for a recursive algorithm to solve the problem.

So far we have moved from a simple everyday example, followed by an outline recursive algorithm, to a demonstration, followed by a more detailed recursive algorithm. The next step is to ask students to take part in animating a more mathematical example of a recursive algorithm. The problem to be solved is calculating the factorial of an integer—8!

I explain that I ask a student (A) to work out 8! for me. He's not keen on multiplying, so this seems like a lot of work, but he can see that 8! is the same as 8 * 7! He thinks that multiplying something by 8 would be manageable, so he asks a friend to work out 7! for him. The friend (B) feels the same way about multiplying as A, so she asks C to work out 6! for her, and she knows that all she has to do is multiply C's answer by 7. They carry on in this way: C asks D for 5!, D asks E for 4!, E asks F for 3!, F asks G for 2!, and G asks H for 1!

But finally, 1! doesn't need any multiplying, so H knows that the answer is just 1, and passes this back to G. Now G has to multiply that answer by 2, and passes the answer (2) to F, who multiplies that by 3, and passes her answer (6) to E. E multiplies 6 by 4 and passes 24 to D; D multiplies 24 by 5 and passes 120 to C, C multiplies 120 by 6 and passes 720 to B; B multiplies 720 by 7 and passes 5040 to A, who finally realises that multiplying by 8 isn't necessarily that easy but does it anyway and gets 40,320 to pass back to me.

I have pre-prepared A4 cards with the relevant numbers and calculations written on them. i.e. "Calculate 8!", "Calculate 7! and so on. I ask for 8 student volunteers and ask them to stand at the front of the lecture theatre. These cards are given to the students and we begin the animation of the algorithm, with each student passing the relevant Calculate card to the next in turn. When we reach the base case of 1!, the last student then hands back a pre-prepared card with the answer—i.e. $1! = 1$. The next student in line takes this, and then hands on a card saying $2! = 2 * 1 = 2$, and so on until finally card with the answer is passed back to me.

Following this change of focus and the physical activity, the students are then shown and talked through some Java code to calculate the factorial of an integer. After explaining the code, an example (5!) is drawn using the Visualiser, showing each recursive call until the base case is reached, and then the unwinding of the recursion until the answer is returned. All worked examples done on the Visualiser are made available after the lecture via the University VLE, along with the lecture notes and tutorial exercises. Lecture capture software also enables students to watch all or selected parts of the lecture again, facilitating further consolidation and reflection.

There are many animations and videos available; some useful examples follow. When teaching sorting algorithms there is an interesting animation from Toptal (n.d.) that gives a visual representation of several different algorithms sorting a number of lines of different lengths in real time, so that students can easily see the differences in performance, given different initial conditions. Also the Hungarian folk dancing representation of Quick sort on YouTube can give a useful break and change in focus and activity, and inject some humour into the lecture.

One of the key techniques is allowing students time in the lectures to do short exercises to reinforce what has been taught. For example, in propositional logic, after explaining how to translate from natural language sentences into propositional calculus, they will be given three small problems to solve. The first one I will do with them on the Visualiser, and I will then ask them to do the second problem themselves on paper. Students can either work alone or in pairs, allowing an opportunity for reflection and collaborative work. After a few minutes we will talk through the solution, and then the process is repeated for the third problem. These simple exercises allow the students to reflect on and apply the material presented to them. It can also highlight any areas they have not understood, so they can be encouraged to ask further questions as a result of attempting the exercises.

All lectures have an associated tutorial, where students can reinforce their learning and apply it to graded exercises. There is also the opportunity for collaborative work if they choose. The aim is for this knowledge to be useful to and used by the student in the rest of their University course and in their working life.

2.5 Outcomes and Reflection

How do we assess whether these methods work? Prince (2004, p. 2) suggests that the outcomes that should be considered include “measures of factual knowledge, relevant skills and student attitudes, and …student retention”.

We can look at the outcomes of the assessment, which for this module is a 2-hour examination. We can look at Lecture capture star ratings and module evaluations. We can listen to feedback in student panels. From my own work I have noted that those lectures made available via lecture capture that get a higher star rating are the ones I’ve discussed here—that is propositional logic, recursion and sorting algorithms, which have the highest level of active learning content. Anecdotally, students on the whole are happy with the teaching they receive in this module as evidenced by their comments in our Student Panel meetings.

While the outcomes mentioned have all been good, they don’t tell us which particular aspects of the teaching are the most effective. They don’t tell us what may in fact work better, and they don’t tell us which concepts students find the most challenging.

From my own experience, it seems that some students respond better than others to the active learning techniques discussed. For example, when asked to work on examples in the lecture, some students will work enthusiastically on the problem, discussing it and sharing solutions with their neighbours. Others are less willing to join in—perhaps finding the problem daunting or difficult. Doing a couple of exercises with them first can help, and I also try to give them an easy way to start—so, for example, when drawing a FSM, I suggest they think of the shortest possible legal input, and draw a diagram for that first. This may be as far as they get, but hopefully they get some sense of achievement from this. The worked solution then shows them how to build on this.

Videos and animations are generally well-received, and seem to hold everyone's attention. Obviously they have to be carefully chosen to be both interesting and informative. For the student recursion exercise it is useful to mention it to a few students beforehand, in one of the tutorials perhaps and check they will be willing to volunteer on the day, otherwise it can take a little while to encourage students to stand at the front. However, once everything is in place, it engages the audience.

We might want to consider which students benefit the most. All students should benefit to some extent from being more engaged with what is going on—active learning should be more interesting than just sitting there. Students who don't benefit tend to be those who haven't engaged with the course or the module, or those who have pre-conceived ideas about theory and maths, and believe that they can't do it. Additionally, students who have been absent and not caught up or are finding the module difficult may find the in-lecture exercises too hard. Extra optional support sessions are provided on a weekly drop-in basis to help with this, but it is difficult to ensure that all the students who need them attend. There will always be some students whose reaction to something being difficult is to ignore it and stop attending. Some very able students may find the exercises too easy, but can be given additional advanced exercises to do in the tutorials or in their own time. Ideally for all students their mental knowledge structures should continue to develop as they build on what they have learned in the rest of their degree course. The literature provides some support for the effectiveness of active learning, although many reviewers have identified a need for further study. The conclusions given by Prince (2004, p. 7) are that there is "support for all forms of active learning examined …students will remember more content if brief activities are introduced to the lecture".

What pitfalls are there in adopting an active learning approach? There is a risk that if the exercises are too difficult then the students will just use it as an opportunity to chat amongst themselves. The amount of time given to work on an exercise should be limited, and the lecturer should intervene and go through the example if it seems that students are struggling or not working effectively. It can be difficult to gauge what is the correct amount of time to allow to complete an exercise, as not all students work at the same pace. But by keeping a close eye on who is doing what, who is writing, and general noise levels, it can be done.

It is also important to set the right initial tone and atmosphere as far as possible. So normally students are expected to listen and not chat or disturb other students—but when doing an exercise, they can discuss things amongst themselves. However, once the lecturer beings going through the answer, then their full attention should be expected. Thus changes in activity are accompanied by changes in pace, tone, atmosphere and noise levels. Technology can also cause problems—so any links to videos or animations should be checked before the lecture, in case they no longer work.

It might be asked if content has to be reduced to incorporate active learning. Over the years that this module has been delivered, some topics have been removed through natural wastage, for example, because they no longer feed into specific second year modules—so the removal of a second year Formal Methods module resulted in material on relations and functions being omitted. Such changes in content allow for an increase in the amount of active learning that can be included. Material can also be restructured and reworked to allow for the inclusion of active learning material—perhaps replacing a wordy explanation with a shorter one accompanied by a video or animation, for example.

Finally, if we want to extend the use of active learning techniques, then further development could include the use of a Student Response system for students to give answers to questions to assess their level of understanding. This might help make it clearer to the lecturer which specific aspects or topics students are struggling with. A system developed at Huddersfield enables teachers to initiate questions, and students to respond using their own mobile devices such as phones, laptops or tablets. It allows data to be collected and automatically stored for future retrieval (Meng and Lu 2011). The system can be used in activity-based or problem-based educational settings regardless of the size, age or knowledge background of the learning group. This could provide useful information as to where additional explanation or examples would be beneficial, and help better assess the usefulness of the active components, as well as actually providing an additional active component in the lecture.

2.6 Summary and Conclusion

This chapter has discussed the challenges of teaching large, diverse groups of students, it has described active learning, looked at the motivations for its use and reviewed some of the literature. We have presented the specific, practical techniques and examples used to deliver active learning in a large Computing Science and Mathematics lecture and then considered the outcomes and potential pitfalls of using these techniques along with possible further developments.

References

Biggs J, Tang C (2009) Teaching for quality learning at university, 3rd edn. Open University Press, Maidenhead

Bonwell CC, Eison JA (1991) Active learning: creating excitement in the classroom, ASHEERIC Higher Education Report No. 1, George Washington University, Washington DC

Bruner J (1960) The process of education Cambridge. Harvard University Press, MA

Dynamic Drive (no date) The Tower of Hanoi. Retrieved from http://www.dynamicdrive.com/dynamicindex12/towerhanoi.htm

Felder R, Brent R, Stice J (2002) National effective teaching institute: workshop materials. American society for engineering education annual conference, Montreal, Quebec, Canada

Fry H, Ketteridge S, Marshall S (2015) A handbook for teaching and learning in higher education: enhancing academic practice. Milton Park, Abingdon, Oxon, Routledge

HEA (2017) TEF2 Infographic. Retrieved from https://www.heacademy.ac.uk/knowledge-hub/evidencing-teaching-excellence

Meng Z, Lu J (2011) Opportunities of interactive learning systems with evolutions in mobile devices: a case study. In: Proceedings of the 2011 international conference on internet computing ICOMP 2011. CSREA Press, pp 238–244. ISBN 1601321864

Moore J, Higham L, J Sanders J (ARC Network) in partnership with Jones S, Candarli D, Mountford-Zimdars A (2017) Evidencing teaching excellence analysis of the teaching excellence framework (TEF2) provider submissions. Retrieved from https://www.heacademy.ac.uk/knowledge-hub/evidencing-teaching-excellence

Piaget J (1950) The psychology of intelligence. Routledge and Kegan Paul, London

Prince M (2004) Does active learning work a review of the research. J Eng Educ 93(3):223–231

Raelin J, Coghlan D (2006) Developing managers as learners and researchers: using action learning and action research. J Manage Educ 30(5):670–689

Revans RW (1971) Developing effective managers; a new approach to business education. Praeger Publishers

Sarason Y, Banbury C (2004) Active learning facilitated by using a game-show format or who doesn't want to be a millionaire? J Manag Educ 28(4):509–518

Sutherland TE, Bonwell CC (1996) Using active learning in college classes: a range of options for Faculty in Teaching and Learning in Higher Education. (2008). Acad Manag Learn Educ 7(1):139–142. Retrieved from http://www.jstor.org/stable/40214510

Toptal (no date) Sorting algorithm animations. Retrieved from https://www.toptal.com/developers/sorting-algorithms

Umble M, Umble EJ (2004) Using active learning to transform the monte hall problem into an invaluable classroom exercise. Decis Sci J Innovative Educ 2(2):213–217

University of Huddersfield (2018) About us. Retrieved from https://www.hud.ac.uk/about/

Weltman D (2007) A comparison of traditional and active learning methods: an empirical investigation utilizing a linear mixed model, Ph.D. thesis, The University of Texas at Arlington

YouTube Quick-sort with Hungarian (Küküllőmenti legényes) folk dance. Retrieved from https://www.youtube.com/watch?feature=player_embedded&v=ywWBy6J5gz8

Chapter 3
The Flipped Classroom

Michael O'Grady

Abstract A first year introductory internet and digital media module, taught as an in- and out-service across two Schools and covering a wide range of music courses, resulted in attendance and completion problems on a yearly basis. Students ranged from technical/programmers through to musicians and performers. Keeping them all engaged, motivated and appreciative of the learning materials was a challenge. The Flipped Classroom approach of front-loading weekly learning materials on to the VLE and providing more time for structured tutorial and computer lab work provided a distinct improvement in student marks and satisfaction scores. This Chapter assesses student marks against attendance and VLE engagement from this approach over one academic year. The learning materials provided a broad richness of weekly-themed resources including: screencasts; text; PowerPoint slides; screencasts of slideshows; scholarly articles; blog posts; web articles; and embedded YouTube videos. Students were encouraged to read broadly and assimilate as much as possible from the varying formats of presentation whilst having weekly structured tutorial content. The weekly content creation/linkages were designed to provide additional materials for those on the technical spectrum and additional content for the creatives. Student marks are compared against attendance and also against VLE engagement over the year. Whilst both indicate a benefit with increasing attendance and VLE engagement, there is a greater parity with the latter—attendance in class is less important than perusing the weekly resources. Students were able to perform to first class standards whilst engaging fully with the VLE and having a less-than 50% attendance in class. A broad section of students engaged with materials before the weekly lab sessions and some engaged with hardly any tutorial work at all. VLE access data also shows many students engaged to the early hours of the morning. The Flipped Classroom approach resulted in better module results as well as an improved classroom experience: greater focus on the learning materials and tutorial exercises; much less time surfing and social networking in class; and more time in class for one-to-one assistance.

M. O'Grady (✉)
Department of Computer Science, University of Huddersfield, Huddersfield, UK
e-mail: m.ogrady@hud.ac.uk

J. Carter et al. (eds.), *Higher Education Computer Science*,
https://doi.org/10.1007/978-3-319-98590-9_3

Keywords Flipped classroom · VLE · Attendance · Engagement · Learning resources

3.1 Introduction

This Chapter discusses the introduction of a Flipped Classroom approach to teaching first year digital and internet technology content to music students during the academic year 2014/15. The module, entitled 'Introduction to Digital Media and the Internet' provides broad coverage to a wide range of music students across two Schools (Computing and Engineering as well as Music, Humanities and Media, C&E and MHM respectively). Some courses had this module as core and others as optional. Around 70 students took the module in 14/15, spread over four groups of 1.5 hour sessions per week based in computer labs.

Student backgrounds and courses ranged from the computer-programming BSc Music Technology students who wanted technical detail and coding through to musical artists and performers on BA Popular Music production. The latter were interested in setting up websites, content, promotion, video for marketing and visually expressing their own or their band's music persona, whilst the former were unmotivated to anything 'arty' and performance related, wanting instead to learn code and technically manipulate audio.

The module, based in C&E, was an in-house service module to Engineering colleagues and an external service module for students studying music within MHM. The module was intended, by their various Course Leaders, to be 'all things to all students' in the area of digital/multimedia. However, the reality hadn't live up to the billing, mainly because of the diversity and expectations of students and in part from a demotion in its importance by having the two hours contact time reduced to 90 min. The lost 30 min being assigned to a more difficult and core module on each course. As a result, one of the recurring complaints was that students didn't have enough time in class for one-on-one support to understand and complete their weekly tasks.

This module had been running for over 10 years. The different module leaders taking their own slant on what was taught within the specification. Different versions of the module covered the following:

- technical website production (coding from scratch)—good for techies, bad for performers;
- marketing, visuals, video and DVD production (Photoshop and branding from scratch, lots of visual assets produced)—bad for techies, better for performers;
- making content managed websites to a professional standard, publishing of image and blog content and video creation—the author's approach, good for some, bad for others.

Having taught the module for several years with relatively low levels of student satisfaction (see below), the flipped classroom approach seemed intuitively like a good direction to take. The basic premise was to:

- provide a weekly structured diet of rich and broad content available for pre-reading before the class session;
- some content would overtly appeal to the creative/performance music students and some to the more technically-minded;
- provide an adaptive-release set of weekly tutorial tasks to be completed before or in the class. Initially invisible, students gain access by working through the weekly pre-reading and click the *Reviewed* adaptive-release button at the bottom; and
- minimise the taught/instructional aspects of the 90 min weekly contact and maximise tutorial one-on-one assistance.

The flipped classroom approach was to provide an increased amount of content for each side of the technical/art divide. It required much in-class support and verbal direction (individually and class-wide), with constant prompting and reminders on approaches to take with assignment work. In addition, each student was provided an opportunity to extend their learning (within the direction of their course) with experimental work. Each of the two termly and equally-weighted assignments allowed up to 20% of marks to be assigned towards experimentation—areas of work and learning defined as anything above/beyond the specific course content.

The result of this work was a doubling of student satisfaction from below 50% to around 90% from using this approach. Additionally, the author had access to a relatively large amount of corroborative data from different student-monitoring systems. Although the biggest disappointment was data available via UniLearn, our branded version of the Blackboard^{TM} VLE.

The results (see later) show some interesting findings regarding student behaviour and performance by comparing data from:

- the two equally weighted assignments and their average;
- attendance monitoring of students in timetabled sessions; and
- click-through engagement with different elements of the VLE.

Further data, not covered in this Chapter, related to engagement with the module-specific reading list (MyReading) resources; formal module evaluation questionnaire; and a custom questionnaire.

This Chapter therefore presents an overview of how the module was structured in the VLE, how it was taught, what the students thought, author's conclusions and backed up with a broad range of data.

3.2 The Module

The module provides a broad-brush approach to dealing with the internet, digital content and multimedia as well as providing an introduction to marketing and promotion of individuals/personas/bands/groups via websites, video and social media. The content was tailored specifically to students in the music industry.

The diverse student backgrounds and reduced content time mentioned above provided a challenging learning environment.

Approximately 85–90% of the cohort come from the more technically-orientated BSc-type courses within our School and 10–15% from MHM. The service arrangements were managed by Engineering colleagues who also managed MHM student attachment to the module. The decision to reduce weekly contact hours from 2 to 1.5 h was made so that other core modules (mainly programming for the technical students) could receive an additional 30 min weekly.

The main challenge was to keep all the students on all six courses challenged and engaged, however the courses ranged from BSc Music Technology and Software Development through to BA Music Technology and Popular Music. The latter group including students who were studying music and leading on instruments such as piano, brass, wind, harp etc.

Unfortunately, the module has been polarising student satisfaction for many years; it was not enjoyed by all students, seen as being unchallenging for some and too difficult and technical for others. For many, they developed digital life-skills to help promote a music career that wasn't yet fledged and most enjoyed the video element in term 2. So the challenge was to take roughly the same taught content (agreed by the various course leaders) and make it more engaging and useful to all the students.

3.3 Motivation and Justification of the Flipped Classroom Approach

During 2010–2013, the author was engaged in providing screencast lecture content and summative feedback for students and wanted more exposure to technologies that utilised video and voice recording, not least to gain enhanced broad digital literacy skills. As part of this interest, the author attended an internal Facilitating Online Learning course (FOL) in 2013. It was designed for those wanting to undertake distance learning courses and gain exposure to cutting-edge webinar technology. The desire to attend was primarily to gain exposure to the webinar software (Adobe Connect) and to become more comfortable in front of a webcam for live, recorded virtual meetings and screencasts.

Whilst the webinar element was very empowering, the presentation of materials via a bespoke VLE module was itself presented using the flipped classroom approach and demonstrated an engaging learning environment. This approach entailed:

- Pre-reading of set materials **before** entering the virtual webinar meeting, including occasional completion of tasks (Wikis, Discussions etc.);
- Pre-reading included academic papers, blog articles and embedded video resources;
- Each week's learning materials centred around the practical use and engagement of the VLE and webinar software;
- All class sessions were hosted remotely on the webinar software;

- A set of webinar facilities was demonstrated: chat, polls, live voice and camera; and
- The facilitators also presented applications from their desktop through the webinar software (Word, PowerPoint etc.).

Over 20 years ago, King (1993) referred to the traditional taught lecture as the 'Sage on the stage' and commented that whilst some subjects work well with this approach, there was merit in learning stemming from more of a facilitation process, referred to as the 'Guide on the side'. The flipped classroom approach identifies with the latter and is an instructional approach, combining a greater mix of delivery methodologies of learning resources, including student discussions, group working, collaboration using online tools and encouragement to reflect more. In effect, potentially covering all of Bloom's Taxonomy (Gilboy et al. 2015).

In their work with HE students in Public Health, Galway et al. (2014) found that students generally liked the flipped approach when learning new content. Over 80% of students preferred the flipped classroom with blended online learning materials and problem-solving exercises in a classroom environment. Some students commented that they were more aware of subject-related news articles, more reflective about what the new material meant, and how it related to learning in other areas/modules. Some of their students also commented that they felt obligated to do the pre-reading before arriving in the session because it is part of the course set-up, otherwise they tended not to engage with reading. Some students can initially have difficulty and/or resent what they perceive as too much reading/learning in the early weeks of such a course and this can impact negatively on student satisfaction (Critz and Knight 2013; Missildine et al. 2013), although this impact tends to lessen as they become used to the pre-loading of weekly work over the course of the academic year.

Significantly improved student satisfaction is also a key factor in the flipped classroom for High School students (Sergis et al. 2018), compared to traditional taught environments. Students gained a greater sense of accomplishment and incentives to participate in the learning process. Lower-performing students gained the highest levels of satisfaction and also benefitting from being more able to engage in deeply collaborative work, probably because of the wider window for improvement.

There were some personal challenges in following the FOL course, many of which would be mirrored for the students:

- Time management of having to do the work before class. The system is less effective if students don't do the learning;
- More reading and learning over short periods than the author was used to—it was quite a culture shock, something that would be more challenging to students;
- The possibility of having progress and non-progress monitored; and
- Increased contact with colleagues from other Schools—usually infrequent and unplanned but always pleasurable and advantageous.

In short, the flipped classroom requires much time and effort up-front, doesn't necessarily work well if students don't engage and is totally different to the way most learning takes place (…during and after the session).

3.4 Establishing the Flipped Classroom Approach

All content was placed on our Virtual Learning Environment (VLE). The University has a formal structure for the menu items in all modules and are as follows:

- Announcements (allowing *sticky* messages to remain at the top);
- Module Information (the module handbook);
- Staff Information;
- Learning Resources (the main weekly learning and tutorial area);
- Assessment; and
- Reading List (branded as *MyReading* internally).

There is now an additional menu item covering lecture Screen Capture which was introduced in 2015 and is not covered in this Chapter.

The Learning Resources area is arranged at the discretion of the module leader and is the main point of call in the teaching week. For this module, it was arranged weekly (teaching weeks rather than admin weeks) and released for viewing the previous week.

For each week created, the folder/container is placed at the top of the page and numbered with the teaching week (1–12 and 13–24, for terms 1 and 2 respectively) and titled appropriately. Students should always be aware of what week they are studying in and what weeks, if any, they have missed—the current week is always at the top.

Each week has three sections:

- An Introductory item with text and possibly containing an embedded descriptive screencast. An introductory screencast was created for several weeks at the beginning and soon dropped due to workload and propensity for it to become outdated the following year;
- A folder with an introduction to the learning content. The folder contains between 4 and 8 separate sequential bite-size items of content. The last item contains a button (*Reviewed*) that, once clicked, forces the Weekly Tutorial work to become visible and available to the individual student; and
- Weekly Tutorial folder—a sequential series of tasks for students to undertake ideally before or during (and after) the weekly session.

All elements (folders, items and adaptive-release buttons) were activated for data recording to determine if a student clicked that particular link. This data was then used to assess VLE engagement and individual performance across the assignment work. Below is a closer look at the two main content folders.

Weekly Learning Content Folder

Each weekly content folder has a set of instructions and guidance prior to students entering. The weekly objectives, learning milestones, linkage with previous weeks' content and advice for assignments can all appear in this introduction text.

Items populating the folder are intended to be viewed sequentially from top to bottom by each student. Learning materials include:

- embedded graphics to demonstrate what type of item resource they were looking at (text, web, video);
- text within the item;
- text and links to uploaded files (Word, pdf, spreadsheet, slides);
- hyperlinks to one or more web resources;
- embedded video—either via the VLE video insert widget feature, or more reliably, via a direct YouTube (or similar) iFrame HTML *embed* code so that the video both displayed and played directly from the page; and
- Screencasts (generally uploaded to YouTube by the author and set to *Unlisted*) explaining some concepts, working through a PowerPoint slideshow or showing how a tutorial exercise is completed.

Some older slideshows (typically no more than four years old) were included in the resources as additional reference items with specific guidance to currency, applicability and comparisons with current screencasts, as appropriate.

Weekly Tutorial Tasks Folder

Initially entitled 'The Learning Module' (a direct copy of the FOL course format), the title was soon changed to Weekly Tutorial Tasks. This folder takes students through the weekly tutorial work in a step-by-step way with indexed pages presenting small elements of work; a sort of 'do this, do that' approach. There was no new learning content in this section although students were exposed to new techniques with software applications. Also, any screencasts in this folder were more specific about using tools and applications in relation to the weekly tasks and assignments.

After the introductory weeks of a new topic and associated software application(s), students sometimes had difficulty performing the weekly tasks. There is a cognitive disconnect between watching a live demo on the projection screen in class, then remembering the steps and doing the same for themselves on a PC. The Flipped Classroom approach was chosen in part to provide more time in class for this purpose, but non-attendees, less-able and less-motivated students can struggle to grasp the essentials by the end of a session. So, screencasts of the author working through the task solutions were created and made available a week or two later. Students would be reminded at the start of each relevant class that a previous week's solution was available as a screencast at a certain location within the VLE. It is good to avoid this being a predictable resource to ensure students don't rely on it as a means to avoid engagement and learning.

Using special 'Catch-up' or 'Problems Solved' screencasts were also created approximately once per term to cover common issues. These would appear as items in the weekly chronology of the Learning Resources page. The screencasts might show answers to common questions from class sessions, email questions and always bring the subject around to implications for the coming assignment.

Once activated, the initially invisible Weekly Tutorial Tasks folder is similar in appearance to the learning folder in that the student is expected to work their way down the stated activities. These are very pragmatic instructions such as Open an Adobe Application with the sample file, then make some additions, alterations and experimentation, save and view the work. The author has developed a shorthand for sequential instructions where the symbol '>' is used to concatenate follow-on instructions. An example for making a hyperlink in MS Word is shown below:

1. *…previous instructions…*;
2. Open a web page in a browser window that will be linked from your document;
3. Select the **URL > Ctrl + C**;
4. In Word, highlight the words(s) providing the **link > Ctrl + K > Ctrl + V > OK > Ctrl + S**;
5. *…continue…*

There was a very focused chronology of weekly tasks expected of the students, particularly because the finished results of work can vary greatly if the order of work undertaken changes in some interfaces. Sometimes the tasks comprise quickly-produced bullet points with a fuller explanation provided in class or by embedded screencasts.

Some students would have undertaken the necessary learning and attempted the tutorial work in one or more sessions. At the other extreme, some who would not have activated the weekly tutorial folder would be unlikely to have read the planned materials before class.

There was a short learning period of around three weeks before everyone was fully aware of the expectations placed on them regarding pre-reading and the practical work in sessions. One challenge was that this was the only 20 credit module in a 120 credit diet using this approach.

3.5 What the Results Show—The Data

There three main data sources accessed for this analysis:

- Overall module and individual assignment marks—two assignments, equal in weighting and duration plus the resulting averaged module mark;
- Attendance database—each student is required to swipe into each timetabled session; and
- VLE engagement reports (the click data on items, folders and *Reviewed* buttons).

Module Marks

For several years, the module has been split into two separate equally-weighted assignments spread across two terms of 12 weeks each. A term comprises 11 teaching weeks and a centrally placed Guidance Week (similar to a Reading Week). Term 1 dealt with the construction of a content management system (CMS) website to a semi-professional standard including customised layout menu, content, imagery, security, backups etc. Students could do as much technical or creative work as their course and

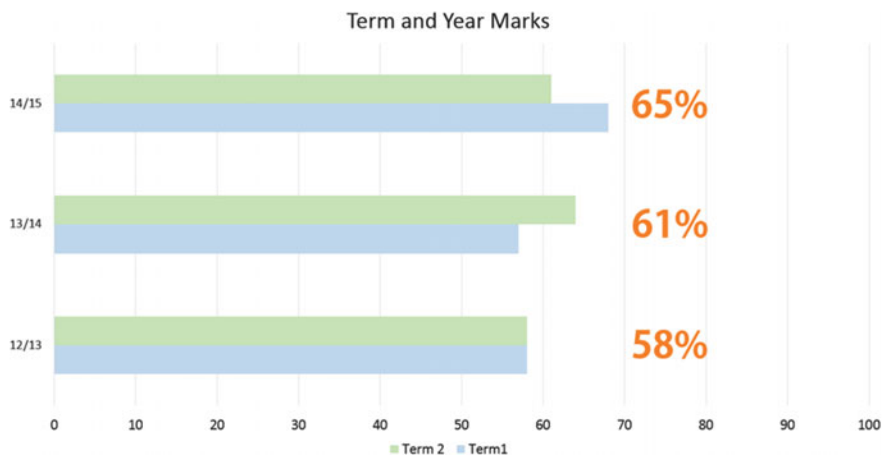


Fig. 3.1 Averages for the module for three years

interest dictated. The theme was at the student's discretion but they were encouraged to create something that served a module or their current- or future-self (performer or band website). This work is very chronological in the build-up of necessary skills for good website creation. The weekly learning is quite intensive, with much to learn, highly technically orientated and there are opportunities for technical students to engage in learning code.

In the second term students learn about planning for video creation, filming, screencasting and editing digital videos. Again, this serves to aid their future-self, to aid placement job searches in year two or, as many discovered, a chance to have lots of fun. Many of the technical students were encouraged to produce videos along the lines of: 'A day in the life of a music producer' or 'How to engineer a [Hip-Hop] track'. This work was more creative and thoughtful for students than the website and for many, very enjoyable.

Figure 3.1 shows a bar chart of termly assessment and class average marks for the three years up to and including 2014/15. We have recently adopted an institutional target of 75% of students achieving a minimum 60% module marks (the lower boundary of a 2:1 Degree classification) in relation to TEF metrics and a steady growth can be seen over the three academic years from 58, 61 to 65% for the year in question. Additionally, the lower bar in a pair represents the technical website-build work (term 1) whilst the upper bar represents term 2 video work. There seems to be a significant increase in term 1 marks from around 57/58 in the preceding years to 68%. This is attributed to the increased exposure to materials for both the technical and creatively/visually-minded students of the class. It is also partly reflected in the 20% of marks awarded for experimentation and pursuing of specialist knowledge within the assignment, as befits their course and interest.

The yearly cohorts are particularly high achieving (evidenced by UCAS entry points) and passionate group of students, some having the same subject back-

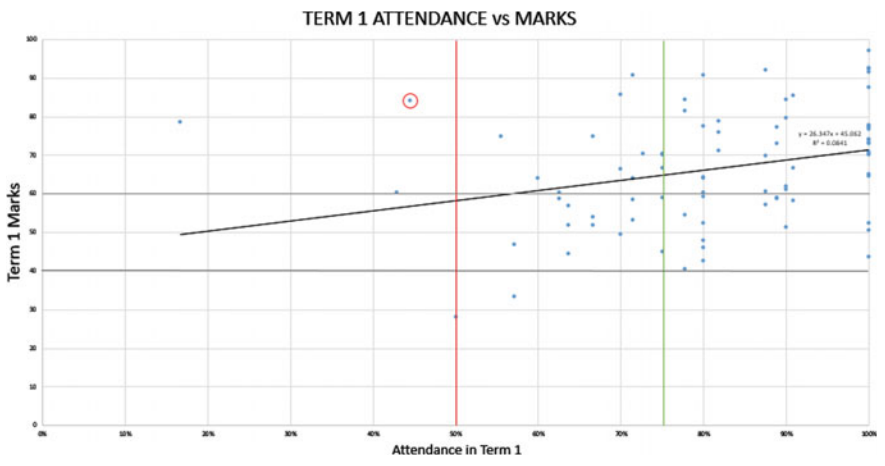


Fig. 3.2 Attendance versus marks for assignment 1

ground as our Web and Multimedia students. They can produce significantly better work than the latter group. However, whilst much of the work handed in is of a high quality, there is a persistent yearly element of students with low final marks due to missing elements of work; they do great work but don't always complete all the sections. Consequently, excellent students can scrape by with averages marks close to the 40% pass.

Attendance Versus Attainment

As an institution, we have had attendance monitoring in all timetabled sessions for several years and it is used to inform student risk algorithms for *support and guidance* interventions. Two key triggers are maintained: (a) students should ideally have an overall attendance record of 75% and (b) if attendance falls below 50% then the student is requested to attend a formal meeting. In reality, there are escalating interventions as attendance falls towards the lower value (invites to attend with support staff for example).

Attendance was very good in term 1 as seen in Fig. 3.2. This shows Attendance in sessions against Assignment Marks for individual students. The upper horizontal line lies at the desired minimum of 60% marks and the lower line lies at the 40% pass.

Figure 3.2 shows clear grouping in the top right corner with approximately a third of students falling in the 50–75% attendance band. Interestingly, there is a 50/50 split in the 50–75% attendance band of those above and below 60% marks indicating that attendance may not be that important when students have access to materials remotely, although it is clear that high attendance and high marks go together in the 75–100% attendance band. Here, 70% of students attain 60% marks or more. Additionally, there are 15 students with 100% attendance, all passing and covering the full range of marks between 40 and 100%. The prescriptive/list nature of all the

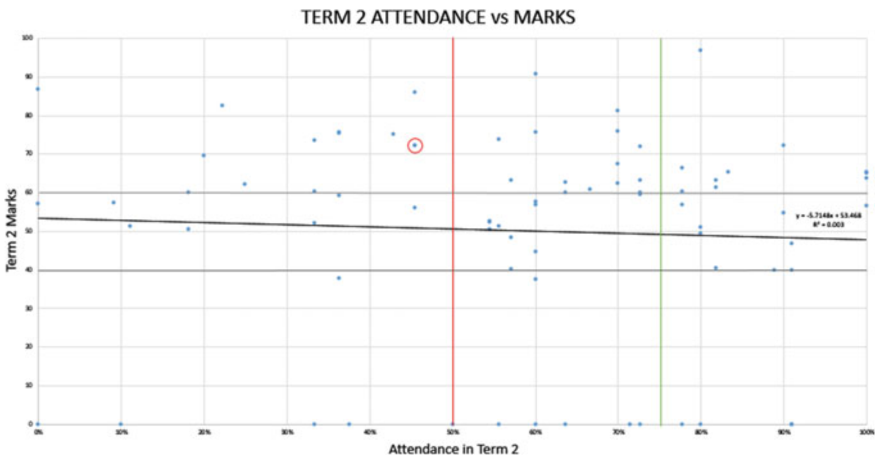


Fig. 3.3 Attendance versus marks for assignment 2

weekly learning items provides a useful checklist for assignment work, especially when students undertake additional experimentation and it is easy for students to ensure they work in 'one of each' learning points into their assignments.

The student result marked with a circle is one who will be referred to several times in this Chapter. This student was a particularly consistent high-achiever over both terms but who was a borderline case for a disciplinary attendance interview with the Dean. They managed their time exceptionally well and had high engagement with the VLE, as we'll see later.

Term 2 (Fig. 3.3) shows a different picture for the cohort, partly because of: the less intensive learning requirements; a more creative assignment including design and documentation; a need to 'get out of the building' and film their footage; sometimes the need for managing actors; and the willingness of some students to 'play the system' (students who reduce both attendance and VLE engagement then cram their assignment work into the last few weeks).

Figure 3.3 shows a broad spread of marks across the attendance range, notably with 10 having marks higher than 60% and attendance worse than 50%. There are 13 students who, having withdrawn or suspended, have zero marks. Interestingly, the 50/50 weighting allows students to have less than a 40% pass on one assignment element if the other mark compensates. The student with the ring is still attaining just enough to get a First having dropped their attendance to 45%. Clearly, the negative slope defies perceived wisdom, mainly because of the high-attending non-submitters.

Figure 3.3 shows a good set of attainment results, mainly due to the creative and enjoyed nature of this assignment, all but two fully-submitting achieving a pass grade or better.

Keeping all the non-submitters and averaging over the whole year, Fig. 3.4 shows a broad scatter around a relatively shallow incline, showing that grades improve with

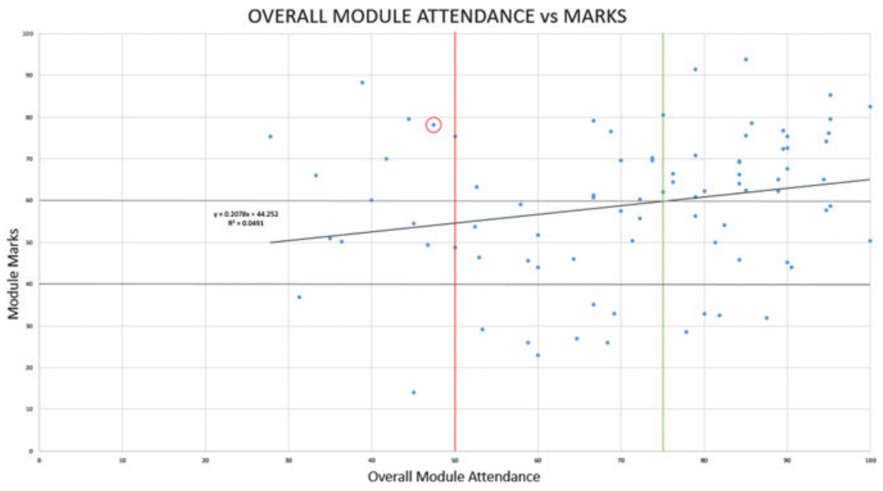


Fig. 3.4 Attendance versus final marks for the module

attendance. There are still four students who have good attendance and fail to make a 40% pass (bottom right corner).

The trend line indicates a 44.3% pass without attendance and only 65% with full attendance though the R^2 value is relatively small at 0.0491. The red-circle student continues to provide an exception to this result.

In conclusion, it is clear that in many cases, good attendance results in good marks, though not always. Weekly engagement with structured learning materials and progressive tutorial content would seem to indicate that good attainment marks result from good attendance and that there are clear exceptions, especially when materials are made available for flexible access and students are good at time management. The term 2 creative work was less successful at engaging students to attend, but again, continued flexible access to resources has resulted in good results in well-motivated students.

VLE Engagement

Some useful high-level information came out of UniLearn (our branded *Blackboard*^{TM} VLE) in relation to overall student engagement. However, pursuit of this data revealed that the dataset for each module only lasted six months before it was overwritten and was not easily obtainable without firstly knowing that such data would be needed at the beginning of a term (which it wasn't). Thus each term's VLE engagement information came from two data sets; one taken after Christmas (term 1) and one in May (term 2) with much work done accessing rolled-back stored data.

Summary engagement data is available for daily click activity, hourly engagement, the number of *Reviewed* buttons clicked and the overall student engagement (total number of clicks). In all, over 220 elements were embedded into the VLE module, most of which were set to record data (one or more student clicks).

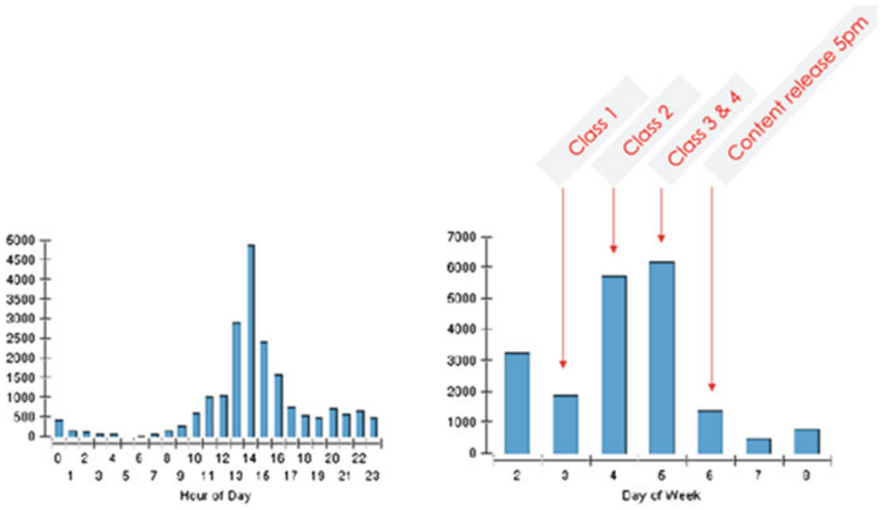


Fig. 3.5 Hourly and daily VLE engagement during term 1

The hourly and daily summative data for term 1 is shown in Fig. 3.5. Hourly engagement (left) shows peak engagement occurs in class times between 11:00 and 17:00 and that around 8% of the peak (around 500 out of 6000) uses occur in the evenings in each hour up to about 01:00. The daily engagement (Fig. 3.5, right) shows high use mid-week with around a quarter of peak daily use coming on Fridays when the materials are released for the following week. Weekend use is low but there is around 50% peak use on Mondays when there were no classes. Clearly students are engaging flexibly with the VLE content.

Term 2 data for hourly and daily engagement summaries show a similar profile with much reduced values, with peaks of around 50% of term 1 engagement. Interestingly, whilst the Thursday engagement reduces from around 6000 to 3500, (class time), the Monday values drop little, from 3200 to 2600, indicating that there is less class engagement (attendance) but students are still using Mondays to work on the module (Fig. 3.6).

The sequential daily engagement in term 1 (Fig. 3.7), where technical content is high, shows a relatively good engagement with a dip to around 45% of peak mid-term and then relatively stable usage at around 60% of peak towards the end-term and assignment hand in. The assignment is handed out in teaching week 2 with 900 click peaks in the weeks either side. There is a non-teaching Guidance Week in the middle of term then a resurgence back up to 600 at the end.

The term 2 graph shows a more relaxed attitude to VLE engagement, running in parallel with the attendance (Fig. 3.8); peak engagement in the first four weeks then a rapid fall-off to the three week Easter holidays and a low resurgence in the last two weeks before assignment hand in.

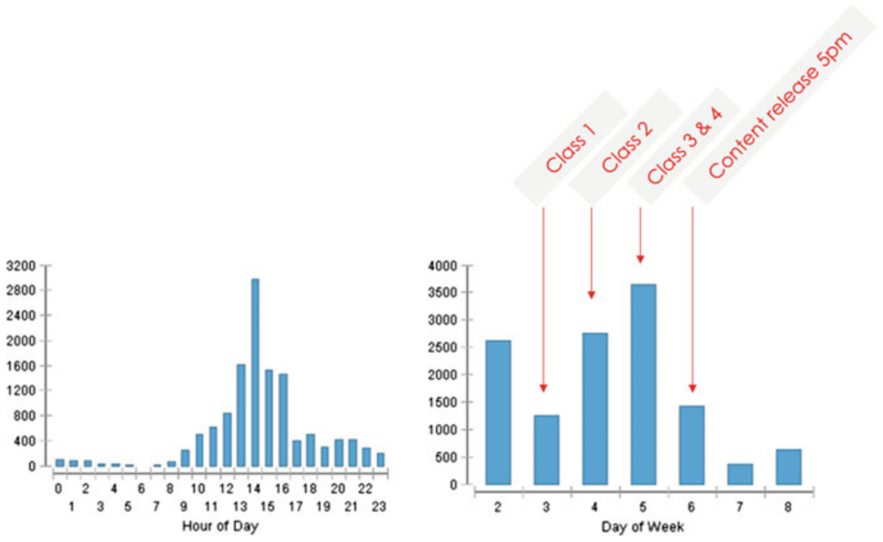


Fig. 3.6 Hourly and daily VLE engagement during term 2

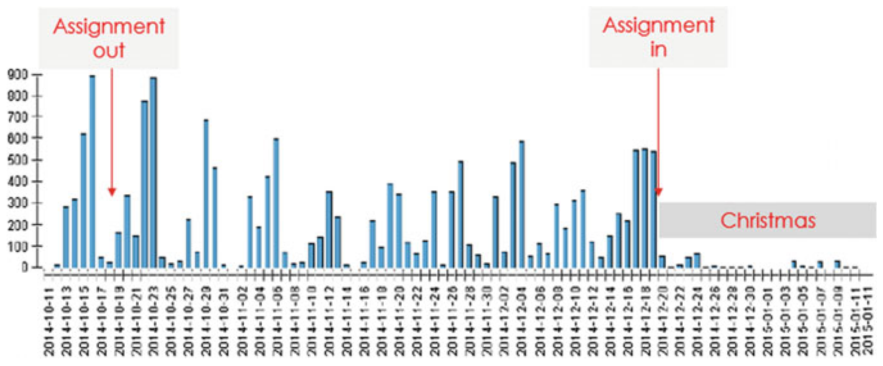


Fig. 3.7 VLE engagement during term 1

VLE Engagement Versus Marks Data

Due to the difficulty obtaining data from Blackboard^{TM}, the results are quite crude, although quite revealing. The results below look at the resulting total *Reviewed* button clicks vs attainment marks for each student (Fig. 3.9) followed by the total number of engagement clicks vs attainment marks (Fig. 3.10). The differentiation is that the former highlights only the exposure of the Weekly Tutorial Tasks (by clicking the *Reviewed* button) and the latter addresses all clicks within the VLS for each student.

Figure 3.9 clearly shows that whilst there is broad spread across the chart, there is a correlation between the students' journey into the weekly tutorial folder and their marks. With 26 *Reviewed* buttons available, our circled student (poor attendance, high

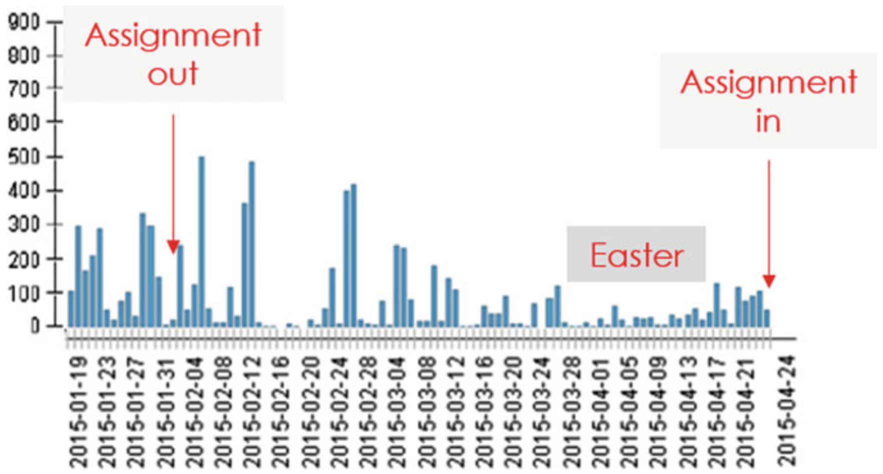


Fig. 3.8 VLE engagement during term 2

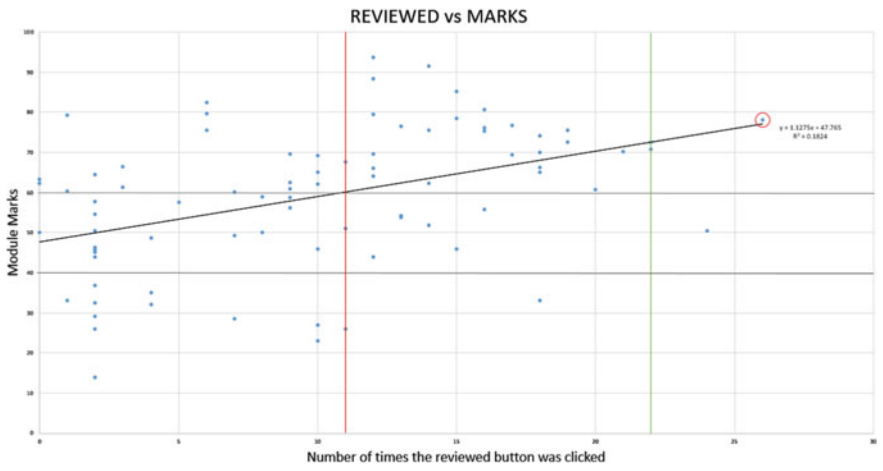


Fig. 3.9 Total reviewed button clicks versus module marks

performance) can be seen as having the highest engagement in this area, seemingly, having looked at all the resources. This tends to support the idea that some students are working strategically through the resources in their own time whilst minimising travel and contact time in class. This is an increasingly common occurrence with a larger proportion of the Huddersfield intake commuting from home due to spiralling costs.

Since the *Reviewed* button does not indicate whether a student worked through the resulting exposed tutorial materials, it does shows that some attained reasonable

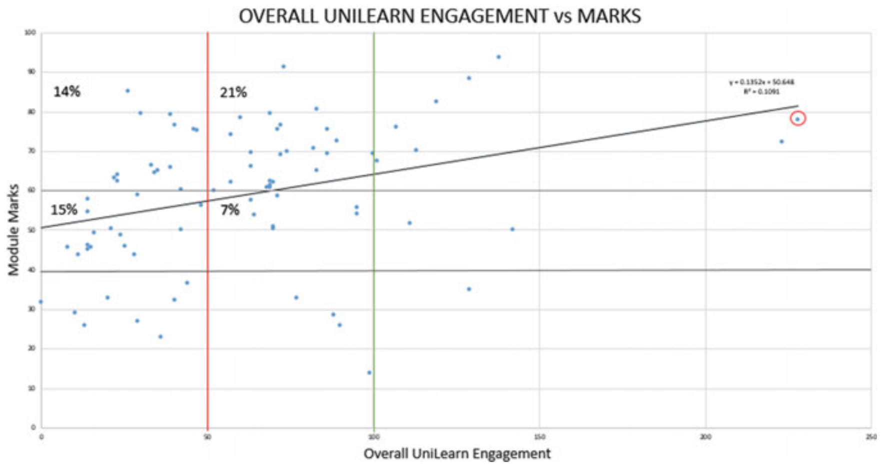


Fig. 3.10 Total engagement clicks versus module marks

marks whilst having entered no more than five weekly tutorial areas (seven students with greater than 60% marks and nine with marks between 40 and 59%).

A vertical line in Fig. 3.9 is placed at 22 Reviewed clicks, representing one click for each teaching week and nominally encompassing the bulk of clicks. A second line is placed nominally at a mid-point of 11 clicks. Analysis of the grades to the left of this line show that there are roughly equal numbers achieving 60% or more (18) with those between 40 and 60% (17) and a relatively large proportion not passing (12). Percentage results are 18, 17 and 12% respectively. However, between 11 and 22 clicks, the resulting values are 27, 6 and 1% showing a dramatic rise in success of those engaging with the content.

Whilst the R^2 Regression value of 0.1824 is high, the line and trend is persuasive, the intersect still resulting in a mark of almost 48%. The trendline slope is 1.1275 which is approximately five times steeper than that of overall attendance slope of 0.2078 (Fig. 3.4). Hence is can be argued that VLE engagement, particularly into areas that are actively opened by students, is of greater significance to student marks than simply attending classes.

Figure 3.10 shows the total engagement clicks within the VLE module per student against their module mark. Approximately 230 clicks were possible and once again, our circled student is leading the way on engagement.

Two vertical lines have been drawn nominally to act as a datum and were chosen partly to reflect the data grouping and be positioned on significant engagement numbers (50 and 100). Without the two data points at around 230 clicks, there seems to be a very broad scatter, which is not helped by the non-passing marks, typically from students who had not submitted assignment 2 (those who had suspended, withdrawn or repeating the module in the following year).

Figure 3.10 also shows that there are roughly equal numbers of students below the 50 click engagement line with marks 40 to 59% (15%) with those attaining 60% and above (14%). Although not investigated, the author's view is that this group included those with previous skills and experience of the learning materials who didn't need to access the tutorial materials.

Interestingly, the increased VLE engagement between 50 and 100 clicks sees three times more students attaining 60% or more to those between 40 and 59% (21% of students compared to 7% respectively). This is a good indicator that those who have a reasonable engagement with the learning materials attain better marks, not least by having a greater knowledge of what the assignment requires of them. Only 10% of the students (nine) engaged between 100 and 150 clicks with all but one passing and just over half (five) achieving marks of 70% or more.

It is from the above that overall engagement with the VLE is a more significant factor in increasing student performance than attendance in class and that flexible access to materials is a predominant factor in increasing student engagement and attainment.

3.6 Benefits of the Flipped Classroom Approach

There were some early gains with this approach, the first being the engagement of students in class. In previous years, the ratio of taught content to tutorial and support work was roughly 2:1 due to the reduced time in class. There was a tendency for students to become distracted and noisy unless firmly controlled and guided. The flipped classroom approach provided a sea-change; the technical and information-rich nature of term 1 materials resulted in students being eager and engaged with their learning and tasks, to a point where there would be utter silence for long periods. Some students would not want to be disturbed by enquiries to see if they needed any help.

The need to be very organised in advance of the day of materials release (typically Friday of the week before) is almost a reason to put this item in the *Issues* section (following). However, it provided excellent time management practice and was a benefit simply because the teaching week generally remained free of problems—all the hard work was completed the previous week.

Students like having a regimented presentation of materials which is in a predictable format and is consistently available. It's one less thing for them to worry over. Also, providing a one-page weekly timetable across all weeks and the consistent numbering, labelling and naming of the weekly learning items allowed quick access backwards at assignment time.

Students appreciated having the current week's material located at the top of the page. Scrolling down the page to find older materials is an intuitive process—recent learning is near the top and content from many weeks ago is further down the page.

Many students commented that they liked having the materials available in advance of the session, particularly over the weekend so they could take advantage of the 1.5 hour session to do practical work. It would be advantageous to some of the cohort to have materials available for at least a week prior to the class session.

The Learning Resources area provided access to a wide range of material formats; students enjoyed the mix of web articles, digitally-accessed academic papers, blog posts, podcasts, videos and screencasts. The use of a variety of different *voices* (literally and metaphorically) helps students accept new and challenging theories, concepts and techniques These resources do need careful choice regarding academic level, the potential for mixed messages and relevance. The change of *presenter voice* is important because different experts can not only tell the same story differently, but they approach the content with different backgrounds and motivations, often introducing associated and interesting asides. These are a good opportunity for increasing the engagement and extending wider reading around the subject.

The weekly tutorial work has historically comprised pdf instructions with graphics and screengrabs to allow students to build/create digital assets such as elements of a website, manipulate a digital image, to edit elements of a digital video etc. These are typically sequential step-by-step guides. They were updated yearly to reflect software versions and interface updates, however, with the increased workload arising from the flipped classroom approach, the author decided early on to reference previous year's tutorials as legacy materials, with additional time taken in class to point out the relevance and changes exposed in the older material.

3.7 Issues with Flipped Classroom Approach

There is a need to upload a lot of relevant, appropriate and variable content into the weekly areas in the VLE in advance. It's a lot of work! This can seem relentless at first until the methodology of finding and presenting resources becomes familiar. There is a need to formalise the content structure and to make weekly work tasks *approachable* before the session and also to provide additional challenge within (and beyond) the session. The weekly content needs to be valuable to the students rather than a hasty collection of web links and a mix of format types helps break up the learning process.

Use of Adaptive Release for the weekly tutorial tasks (use of the *Reviewed* button) was not as important a facility with students as it might be with others (say staff on an internal training course). Students were not informed that it would be possible to analyse data on whether they click each *Reviewed* button or not. Telling them this information can often work as an implied threat and can be useful in driving up usage and exposure to the materials. However, students can *play the game* and scroll down the learning resources, click the *Reviewed* button and click on the exposed tutorial tasks simply for the purposes of achieving a better engagement score.

Whilst the benefits of using the flipped classroom are clear, the connection between engagement with the materials and summative marks is also clear, so anything that can be done to drive students into the content is well worth the effort. Ideally, there needs to be some incentive for students to investigate all the materials; such as a weekly class discussion item, contribution to a Wiki or Discussion Board based on *deeper* materials, creating a logbook that forms part of the assessment strategy etc. This additional work was not done due to workload.

Students have no way of being reminded of what they have read or what sections they have worked through or simply visited. Data access via our version of Blackboard provided very rudimentary data that was difficult to use. Other VLEs may treat these issue better.

The system of presenting learning materials, work tasks and tutorial solutions on a VLE could easily be circumvented by students. Good students may do so to accelerate their learning whilst poorly performing students may dip into see if it is of interest and to see the complexity of materials presented, or not dip in at all. There ideally needs to be a mechanism that encourages or requires students to open and engage with as much of the content as possible; this was a failing in this work and the obvious next-step.

The basic approach of expecting students to engage in pre-reading materials before class provided an additional challenge, which soon became apparent in the first weeks of this methodology. It was the only module (20 credits in 120 for the year) that required students to undertake pre-reading before the class. For many, this was a culture-shock and presented them with time management problems which could be easily avoided if desired. Pre-reading became a polarising activity with some relishing the access to deep and rich materials and others learning the predictability of the system to do the minimum required to attain their desired grade. The flipped classroom approach would be more successful if a greater number of modules were taught this way.

3.8 Personal Reflections

Moving from weekly PowerPoint presentations and tutorial sheets to the flipped classroom was a big workload commitment because of the many new resources needing to be found, evaluated and presented for student dissemination. Existing teaching materials became the focus around the generation of new weekly content but soon became relegated in importance for both myself and students, mainly because they aren't presented in class time. Students may or may not want to look through a series of slides, and it may be possible to collect that snippet of data if the VLE allows. However, one of my colleagues maintains that his students would prefer to watch a video of his presentation rather than watch him present live, so it would be worth testing this hypothesis by converting some slideshows to screencasts and embedding on a content page, then testing.

The process of finding interesting and relevant materials never stopped being fun; mainly because of the updating of my own personal knowledge and also because of the interest and difference it made to the students' learning experience. One of the challenges with working in the fast-moving technology area is the need to update materials which can quickly become outdated and this is something that needs careful attention each year. It may be useful to have a Current Trends section which resides above the weekly content and has current-interest links and information. Putting new content at the top of this folder provides an archive of previous years' content if it is sectionalised by academic years, making a virtue from additional updating.

All of the weekly preparation work was completed in the week before it was taught, so contact time was far more relaxed for both myself and students in that week. This allowed more time in class for signposting towards upcoming content and assignment work as well as reference previous weeks' content as questions arose. There was also more time to spend dealing with student support as well as suggesting and demonstrating improvements they may consider making towards assignment work. And finally (significantly), interactions were more fun and creative in class and it became easier and quicker to remember student names.

Additional evidence regarding module evaluation scores and comments, use of a data-aware reading list, additional survey information and the use of screencasting to provide assignment feedback and marks justification are not covered in this Chapter but provide additional support of the above findings. Whilst there were significant improvements in student learning in this approach, it is anticipated that additional improvements could be gained by including more-engaging and interactive activities into the weekly tutorial sessions creating a better blended learning experience (gamification, use of Wikis, Discussion Boards, polls etc.).

3.9 Conclusions

Whilst only providing a vehicle for presenting learning materials and weekly tutorial sheets, this flipped classroom approach has proved beneficial for students in: motivating them to engage with the materials; being studious and focussed in lab sessions; and raising metrics such as module marks. The approach in providing such materials was consistent over both terms, the first (technical) half produced high levels of attendance and VLE engagement whilst the second half (creative) suffered from reduced attendance and VLE engagement. Average marks in term 1 increased by about 10% whilst term 2 marks reduced by approximately 3% compared to previous years. It is concluded that creative content needs a greater level of activity planned in the weekly sessions to drive attendance and learning.

Students benefit from having a logical weekly presentation of materials in their VLE. Using a combination of teaching-week numbers and appropriate titles allowed students easy searchable access to a large bank of materials at any time in the year.

Around a quarter of the cohort accessed the weekly materials in the preceding week and many spread their engagement across the weekend and into the early hours of mornings.

VLE engagement seems to be a better indicator of student performance than attendance in class for this cohort. If rules and penalties allow students not to attend sessions, then some will take advantage and manage their time accordingly and others will decide not to engage with the learning.

The need to actively open the weekly tutorial materials by clicking the *Reviewed* button is easily done without having to undertake the learning. The fact that many did not take this shortcut indicated a distinct lack of engagement by some.

The provision of learning materials provides a rich tapestry to all types of learners with excellent opportunities for asides and extra reading for those interested in the topic. For staff, a great deal of learning and sifting, especially of the myriad web-based opinions, can provide an enjoyable and nourishing activity, not least to keep abreast of current thinking.

References

Critz CM, Knight D (2013) Using the flipped classroom in graduate nursing education. Nurse Educ 38(5):210–213. https://doi.org/10.1097/nne.0b013e3182a0e56a

Galway LP, Corbett KK, Takaro TK, Tairyan K, Frank E (2014) A novel integration of online and flipped classroom instructional models in public health higher education. BMC Med Educ 14(1). (08/2014)

Gilboy MB, Heinerichs S, Pazzaglia (2015) Enhancing student engagement using the flipped classroom. J Nut Educ Behav 47(1):109–114

King A (1993) From sage on the stage to guide on the side. CollTeach 41(1993):30–35

Missildine K, Fountain R, Summers L, Gosselin K (2013) Flipping the classroom to improve student performance and satisfaction. J Nurs Educ 52(10):597–599

Sergis, S, Sampson, DG, Pelliccione L (2018) Investigating the impact of flipped classroom on students' learning experiences: a self-determination theory approach. Comput Human Behav 78. (01/2018)

Chapter 4
Distance Learning: Lessons Learned from a UK Masters Programme

Jenny Carter and Francisco Chiclana

Abstract The MSc Intelligent Systems (IS) and the MSc Intelligent Systems and Robotics (ISR) programmes at De Montfort University are Masters courses that are delivered both on-site and by distance learning. The courses have been running successfully on-site for over 10 years. Designing and delivering courses as distance learning presents a challenge, particularly where the content includes technical and practical elements. For this work we look back at some of the techniques that have been adopted and consider their success or otherwise at enabling us to overcome these challenges. We reflect on some previous studies that have been undertaken over the years of running the courses. Finally, the lessons learned from these successful programmes are considered with a view to generalising the approach and more specifically how we can apply our experiences to the development of new distance courses in Data Analytics and in Artificial Intelligence at Huddersfield University.

Keywords Distance learning · Intelligent systems · Robotics · Postgraduate

4.1 Introduction

The MSc Intelligent Systems (IS) and the MSc Intelligent Systems and Robotics (ISR) programmes at De Montfort University (DMU) are Masters level courses that are delivered both on-site and by distance learning (DL). The courses are delivered mainly by the members of the Centre for Computational Intelligence (CCI) at DMU. Their development enabled the teaching team to capitalise on the research taking place within the CCI and therefore on the strengths of the staff delivering the modules.

J. Carter (✉)
University of Huddersfield, Huddersfield, UK
e-mail: j.carter@hud.ac.uk

F. Chiclana
De Montfort University, Leicester, UK
e-mail: chiclana@dmu.ac.uk

J. Carter et al. (eds.), *Higher Education Computer Science*,
https://doi.org/10.1007/978-3-319-98590-9_4

Pre-programme	Induction Unit
Semester 1 (year 1)	Research Methods (RM)
	Fuzzy Logic (FL)
Semester 2 (year 1)	Artificial Neural Networks (ANN)
	Computational Intelligence Optimisation (CIO)
Semester 1 (year 2)	Artificial Intelligence Programming (AIP)
	Mobile Robots (MR)
Semester 2 (year 2)	Applied Computational Intelligence (ACI)
	Intelligent Mobile Robots (MSc ISR) or
	Data Mining (MSc IS)
Year 3 (typically in year 3 or sometimes on successful completion of 4 taught modules)	MSc Independent Project

Fig. 4.1 The course structure

Usually there are around 60 students enrolled on the courses at any time and at various stages in their programme of study.

Each MSc consists of 8 taught modules and an independent project which is equivalent to 4 modules. Each module is worth 15 credits. The MSc ISR includes two mobile robots modules whilst MSc IS replaces one of these with a Data Mining module as an alternative application area for those less interested in pursuing mobile robotics work. In semester one, a Research Methods module is delivered to ensure that students develop the necessary skills to carry out literature searches, write project proposals and reports. The Applied Computational Intelligence module (ACI) gives students the opportunity to select a relevant area of interest to them and to explore it in greater depth. In this paper we discuss the approaches to delivering the courses to both distance and on-site students. Figure 4.1 shows the pattern of delivery of the modules for distance students; for full time on site students, the pattern is 4 modules in semester 1, 4 in semester 2 and the project over the summer. We also investigate the motivation of the students for embarking on such a programme with a particular focus on how they consider it could affect their employability.

The remainder of the paper is structured as follows: Sect. 4.2 discusses approaches to learning on the MSc programmes and how this fits with recognised approaches from the associated literature; Sect. 4.3 considers the lessons learned from ten years of running such a course; Sect. 4.4 considers how we might generalise from the experience of running the course, thus enabling further courses to be developed in a similar way; Sect. 4.5 summarises the work.

4.2 Approaches to Learning

This was the first distance learning (DL) programme to be developed within the faculty that made use of the virtual learning environment (VLE). It was therefore necessary to find guidance from organisations such as the QAA and from pedagogic literature in order to develop good quality teaching materials. This has been reported in more detail in (Carter et al. 2015; Carter and Pettit 2014) but is summarised here to provide background. Many of the modules employ experiential learning as described by Kolb in (1984), this is evident in particular through the assessment styles where students are encouraged to explore ideas and try out different approaches to problem solving. This is also supported by JISC (2010). All module materials are placed on the VLE which at the time of writing is Blackboard and the course leader ensures a consistent appearance on all modules.

The Quality Assurance Agency (QAA) for Higher Education (QAA Quality Code 2014a, b) publishes codes of practice for the developers of HE courses. These support developers at both undergraduate and postgraduate levels. They also provide specific guidance on flexible and distributed learning. These are updated as technology and ideas develop and are the best place to start when working on a new course. For the MScs, we drew heavily on these to guide the development. One of the important pieces of advice that was more specifically for distance students is to make extensive use of formative feedback.

A particularly effective way of providing formative feedback has been the use of the discussion board facility within Blackboard. The way that this is employed is that on all modules except for Applied Computation Intelligence, there is a discussion board activity worth 10%. The marks are awarded for relevant and timely contributions. It is organised in such a way that most weeks a short activity is provided and the students are expected to post something in response. These responses can be posted in the same week or within a week or two. The idea is that this is something comparable to looking over the shoulders of on-site students during a laboratory session or a tutorial activity session and giving some feedback, particularly where there is misunderstanding evident. For example, on the Fuzzy Logic module, the activity for the first week is to post a brief explanation of why a fuzzy membership grade is not to be confused with a probability (a common misunderstanding) and to give an illustrative example. The tutor would look through all of the posts—spending only a short time, probably no more than an hour a week for all posts—and if all is well they may simply write 'good' or 'fine' on each. However, if there is a misunderstanding they can intercept and give more detailed feedback. What we have found is that the marks for the discussion board activity (awarded at the end of the module) provide the incentive for engagement and that the students then get used to it and start to post responses to each others' posts and hence develop working relationships. It also serves as a way of staff becoming familiar with how the students write and carry out their work and it can help to flag other potential issues.

We also believe it is important to provide constructive and timely feedback on any assessed summative work (the university has a 3 weeks turn around policy). Marking through the use of electronic means made this much quicker than previously requiring hard copies and is now the norm on all of the modules as is the case in almost all courses now in many universities. Making sure that feedback is of good quality is another important consideration as advised in Clark (2008), Nicol and Milligan (2006), Wiggins (2001) and we aim to achieve this by ensuring that feedback is explanatory and where possible provided at the place in the piece of assessed work where the errors took place. Research has shown (Clark 2008) that this improves learning and it is an easy approach to implement. We previously reported on this in Carter et al. (2011). We take on board further advice from the QAA and try to ensure that we do not over-assess our students.

In Goodyear et al. (2008) we meet the idea of networked learning. This is learning in which communications technology is used to promote connections between learners and their tutors and between learners and their peers. Our discussion board described earlier encourages this for each module and some module leaders have other mechanisms built into their modules (for example the neural networks module uses some peer assessment). Others use blogs, wikis and the course has a Facebook group. Blogs are often used to monitor progress with project type work (e.g. in the ACI module and for some when completing the Independent Project). Wikis are used for organisational purposes e.g. booking time slots for presentations or viva voces but also as a way of sharing findings. For example posters or presentation slides (sometimes with sound) that have been developed by students can be uploaded to a Wiki for all students to see. We leave decisions about the choice of communications technology to staff, which results in a variety of methods. Many use Skype for supervision meetings and for presentations and viva voce sessions though others choose to use the facilities within the VLE.

In more recent years, DMU has adopted an approach across the university known as Universal Design for Learning (UDL) (www.cast.org). UDL aims to promote the development of learning materials that suits all learning types and was introduced partly as a response to the Government withdrawal of the Disabled Support Allowance for students with special educational needs; one such example would be those students who may need a note-taker due to dyslexia. UDL draws on the idea that natural learner variability is the rule rather than the exception. The introduction of this required all modules to undertake an initial audit to check key basic requirements of learning materials and assessments met UDL criteria. Due to the careful development of the DL materials for the MSc courses, almost all modules met these criteria completely at the first pass through this audit. A drive such as the promotion of UDL compliance, as well as serving learner variability, also promotes quality and appropriate provision of DL learning materials.

The next section provides a discussion of the lessons learned from designing and delivering such a course.

4.3 Lessons Learned

In order to find out more about the students' perspective on the course, a study (Carter et al. 2015) was carried out where the perceptions of the students were gathered and analysed. The findings of this previous work showed that most of the DL students did not feel isolated in their studies which was pleasing as this has been an important aim with the design of the course delivery. Another interesting observation from the work was that many of the students already in employment (those studying by DL) were doing the course either to enhance their careers in terms of new potential jobs but many were studying to improve the way they carried out the job they were already doing. A noticeable proportion of students were studying with a view to doing further academic research as a result of completing the course.

From a technical perspective, some modules posed particular practical problems when designing the delivery and these evolved over the years. For example, the Mobile Robots module initially involved physically posting a lego robot to all enrolled students. This became impractical as the numbers grew and the locations of the students became more widespread. We also found that when we simplified this by sending directly to each student from an Amazon order, that Amazon does not deliver to all countries, so this was only a temporary approach. More recently, simulation software has been used for both robotics modules (previously simulation software was only used for the Intelligent Mobile Robots module), the software being open source making it much more accessible for all students. Advantages are that the students can develop more sophisticated robot simulations that they could not necessarily implement with a lego robot. The approach to the robotics module is discussed further in Coupland (2010). Some students who decide to take robotics further, e.g. for the Applied Computational Intelligence module or for their project, purchased their own robot and applied the techniques practically. The delivery of the robotics module has previously been reported in Carter and Coupland (2014).

Most modules are now able to use open source software though some still use specialist software that requires licenses for the students. In these cases the Faculty pays for licenses for students and they are able to download and install the software on their own computers.

Another module that has proved to be particularly successful is Applied Computational Intelligence. This is a 15 credit module that takes place prior to the 60 credit project module. The assessment of ACI requires the students to investigate an area of interest to them and to develop an AI solution drawing on the techniques that they have learned in the other modules. The module is effectively a mini-project and it enables the students to practise for the 60 credit project module that they do towards the end of their course. The result is that some students extend the work they did for ACI into their project work. In some cases the ideas have been extended further into work for a Ph.D. The submission of assignments for ACI required the students to write up their work as a report using an d IEEE conference template. Where students produced very good work and where there was an element of novelty, these were then worked into papers for publication at conferences. This has been very popular with the students as it gives them confidence and validity in their work. It also gives them first hand training at becoming a researcher and presenting at conferences, networking with other presenters and so on.

In order to find interesting ways of encouraging both DL and on-site students to meet, we built in optional activities. Firstly, all DL students were invited to visit whenever they wanted and were welcome to attend time-tabled sessions if they chose to come to Leicester. Quite a few students did this from time to time and one of our most frequent attendees lived in Canada but came to visit DMU when his business venture brought him to Europe. We organised two other visits most years and these were partially funded by the university. These were a robotics competition known as the Robot Challenge. Originally this was always held in Vienna and we usually took around 6 students—combination of DL and on-site each year. The Robot Challenge has grown and is now in a different location each year in the same way as a conference. We allowed students taking the ACI module to work as a team on an entry to the Robot Challenge should they wish to. As the assignment for ACI was individual, they took responsibility for different parts of the project and wrote up individual reports for submission. This worked well and there was usually at least one group, sometimes more, that opted to do this.

The other activity that we did regularly was to attend the BCS Specialist Group on Artificial Intelligence (SGAI) conference held each year in December at Peterhouse College Cambridge. This was also a popular experience for both DL and on-site students. Occasionally one of the students would submit and present a conference paper; otherwise they attended, joined in the workshops and benefitted from the networking possibilities.

4.4 Recommendations and Further Developments

The lessons learned from the development and running of this course can be summarised as a set of guidelines that can be applied to new courses. In this section we provide the guidelines and consider how these can be used to develop a new course at The University of Huddersfield: the MSc in Data Analytics and MSc Artificial Intelligence.

All Universities have their own QA processes which are generally very similar so it is assumed that the guidelines will be taken in this context. These usually include input from marketing teams about the predicted popularity of the course.

1. Content relevant to industry and research—at the forefront of the discipline
2. Committed team with subject expertise and interest
3. Thorough materials available and prepared well in advance, making use of such valuable resources and advice as provided by the QAA and JISC (QAA Quality Code 2014a, b; JISC 2010)
4. Support for flexibility of style of delivery—can help with motivation/ commitment from staff

5. Support for flexibility of communication means for students—keep up to date—good advice in https://www.jisc.ac.uk/guides/technology-and-tools-for-online-learning
6. Support flexibility of pace for study (e.g. number of modules per year)
7. Discussion board—helps to motivate and connect including DL and on-site
8. Share materials and delivery pattern for both on-site and DL as they can support each other
9. Don't have commitment to attend but create optional opportunities e.g. conference, competitions
10. Encourage publications
11. To keep good students interested allow for creativity in the design of assignments—many like to relate to their employment or interests or aspirations.
12. Find interesting, subject related and inexpensive ways of enabling students to meet.

At Huddersfield a new Institute for Data Science is about to open and this provides the perfect staff base (as did the CCI at DMU) to develop a course that is in this same mould. We therefore plan to develop a Data Analytics MSc that will be delivered from September 2019 to both on-site and DL students. Similarly, the existing research group, PARK (Planning, Autonomy and Representation of Knowledge) will provide the knowledge and research basis for a new MSc in Artificial Intelligence.

4.5 Conclusions

In this paper we have described the MSc in Intelligent Systems and MSc Intelligent Systems and Robotics. As courses that run both on-site and by DL they are often used as an example in our own institution, resulting in the development of further taught PG courses delivered using the same pattern.

Delivering courses at a distance is a topical area. With the many available mechanisms for interacting with learners electronically there are a number of choices to be made regarding the approach to take. In this paper we have described some of the approaches taken to the delivery of the learning materials and our approaches to assessment and feedback. These approaches have been seen as valuable assets that support the development of e-learning both in terms of trying and testing appropriate technologies but also addressing pedagogic issues that arise when delivering distance and e-materials.

Embracing UDL as a university has reinforced the approaches taken in the development of the learning materials for the MScs. Embracing UDL has driven the development of good practice in teaching and benefits approaches to the DL provisions, ensuring that good quality materials to suit a wide audience are developed.

The course at DMU is successful and sustainable. It has changed in the mechanisms for delivery as technology has evolved and the content is reviewed frequently to maintain the currency of the content. The lessons learned over the years have provided a template for other course at DMU and we hope to pass these on to other academics in other institutions as we believe the ideas are worthy of reuse.

We have provided a checklist for others who wish to develop courses that are delivered in a similar way and specifically intend to apply the approach to the development of the MSc Data Analytics at the University of Huddersfield.

References

Carter J, Pettit I (2014) E-learning for employability: a case study from a UK master's programme. In: Motta G, Wu B (eds) Software engineering education for a global e-service economy. Progress in IS. Springer, Cham

Carter J, Chiclana F, Al-Omari, M (2015) E-Learning for distance students: a case study from a UK masters programme. In: HEA 2015, the European conference on technology in the classroom 2015

Carter J, Coupland S (2010) Teaching robotics at the postgraduate level: delivering for on site and distance learning. In: Proceedings of the international conference on robotics in education (RIE2010)

Carter J, Coupland S (2014) Robotics for distance learning: a case study from a UK masters programme. In: HEA 2014, STEM annual conference 2014

Carter J, Matthews S, Coupland S (2011) Teaching robotics at the postgraduate level: assessment & feedback for on site and distance learning. In: Proceedings of the international conference on robotics in education (RIE2011)

Clark R, Meyer RE (2008) E-learning and the science of instruction. Wiley

Goodyear, P, Banks S, Hodgson V, McConnell D (2008) Advances in research on networked learning. Springer, pp 1–10

http://www.cast.org

https://www.jisc.ac.uk/guides/technology-and-tools-for-online-learning

JISC (2010) Designing interactive assessments to promote independent learning. http://www.jisc.ac.uk/media/documents/programmes/elearning/digiassess_interactiveassessments.pdf

Kolb D (1984) Experiential learning: experience as the source of learning and development. Prentice-Hall, New Jersey

Nicol DJ, Milligan C (2006) Rethinking technology-supported assessment in terms of the seven principles of good feedback practice. In: Bryan C, Clegg K (eds) Innovative assessment in higher education. Taylor and Francis Group Ltd, London

QAA Quality Code (2014a) Part B, Assuring & Enhancing Academic Quality, Chapter B3. http://www.qaa.ac.uk/assuring-standards-and-quality/the-quality-code/quality-code-part-b

QAA Quality Code (2014b) Part B, Assuring & Enhancing Academic Quality, Chapter B6. http://www.qaa.ac.uk/assuring-standards-and-quality/the-quality-code/quality-code-part-b

Wiggins G (2001) Educative assessment. Jossey-Bass, San Francisco

Chapter 5
Academic Integrity for Computer Science Instructors

Thomas Lancaster

Abstract For Computer Science instructors, upholding academic integrity requires approaching teaching and assessment in a way that communicates progressive principles such as honesty, trust, fairness, respect and responsibility to students, At the same time, instructors also have to take steps to make it untenable for students to commit academic misconduct. This means that instructors need to be aware of unacceptable conduct by students, covering behaviours such as plagiarism, collusion, contract cheating, examination cheating and research fraud. Instructors also need to put measures into place to design out opportunities for students to engage in such unacceptable behaviours. This chapter explores academic integrity from the perspective of the knowledge needed by a Computer Science instructor. This is a changeable feast, as new methods to subvert academic integrity are always emerging, particularly in Computer Science where many students have the skills needed to develop the technology that aids new ways of cheating. As such, the chapter recommends that instructors deliver their curriculum with a pro-active focus on academic integrity from the outset. This includes leading by example and developing assessments that remove easy opportunities for students to cheat. This also means putting methods of detecting academic misconduct in place, even if detecting misconduct is only really intended as a measure designed to disincentivise students from cheating since they may get caught.

Keywords Academic integrity · Academic misconduct · Plagiarism · Contract cheating

T. Lancaster (✉)
Department of Computing, Imperial College London, London, UK
e-mail: thomas@thomaslancaster.co.uk

J. Carter et al. (eds.), *Higher Education Computer Science*,
https://doi.org/10.1007/978-3-319-98590-9_5

5.1 Introduction

The term academic integrity describes the positive action where a student strives to take full advantage of the learning opportunities offered to them, complete assessment to the best of their ability and engage productively with the university community. It is part of the framework of ethical understanding that students are expected to gain during their journey to become a professional. Academic integrity is often presented alongside a parallel, yet less positive message that academic integrity means that students must not cheat. As such, the research and teaching practice on academic integrity lies closely intertwined with that related to student plagiarism.

For the Computer Science instructor, communicating and teaching the principles behind academic integrity offers challenge. Industry practices require students to gain experience collaborating and working in teams. Software development often involves the reuse of existing code bases. Students need to demonstrate that they are skilled at using online resources to find solutions to the problems they come across during the programming process. Students therefore need to understand the difference between acceptable and unacceptable conduct in a variety of educational and work-like situations.

The Computer Science instructor too must operate within the same academic integrity constraints as their students. This means that they must lead by example, taking every opportunity to design learning resources that remove opportunities for students to accidentally breach academic integrity norms. This also means that they should ensure that, where students do commit plagiarism or fail to follow acceptable academic integrity practice, suitable action is taken.

This chapter addresses academic integrity for the Computer Science discipline by providing practical ideas and considerations. After a brief introduction to academic integrity terminology and challenges, the chapter focuses on three specific problems: (1) introducing the importance of academic integrity to students; (2) engaging students with assessment and reducing the temptation for them to cheat; and (3) detecting when academic integrity has been breached. Detection, although somewhat counter to presenting academic integrity as a set of positive virtues, remains a necessary evil since it is not fair for some students to gain qualifications they do not deserve at the expense of other students who are putting the expected effort into their studies.

This chapter is intended to be of use to all Computer Science instructors looking to improve how they ensure academic integrity is upheld within their teaching. It is aimed at instructors of all experience levels. It is also hoped that the chapter will be of use beyond the Computer Science discipline, particularly for other practical and industry led subjects.

No attempt is made in this chapter to provide a source for every idea. It includes, for example, many now standard observations and recommendations on assessment design. These have been widely mentioned in the literature, including in other papers by the author and it would be difficult to credit these to a single definitive source. Instead, it is intended to collate the relevant information in a succinct form suitable as a primer for the modern Computer Science instructor.

5.2 Academic Integrity in Computer Science

The Exemplary Academic Integrity Project (2013) defines academic integrity as:

> acting with the values of honesty, trust, fairness, respect and responsibility in learning, teaching and research. It is important for students, teachers, researchers and professional staff to act in an honest way, be responsible for their actions, and show fairness in every part of their work. All students and staff should be an example to others of how to act with integrity in their study and work. Academic integrity is important for an individual's and a school's reputation.

The definition is useful as it reverses the traditional way in which academic integrity is presented, which is simply by telling a student what they may not do. The definitions provides a set of values for students to follow during their academic life. It further notes that these values apply equally to everyone involved in education, which also includes instructors.

Despite an attempt to focus on the positive, any practical chapter on academic integrity would be incomplete without indicating the unacceptable behaviours that students have been observed using. An overarching term for such behaviours is academic misconduct. These behaviours are also often referred to simply as methods of student cheating.

Several of the methods through which academic integrity can be breached and which are most relevant for Computer Science are summarised in Table 5.1.

The definitions in Table 5.1 are not intended to offer a complete list of the ways in which academic misconduct can be breached. Such a list can never be complete so long as new methods of teaching and assessment are developed and these are accompanied by revised methods of cheating. Academic misconduct can include any activity intended to give a student an unfair advantage over other people. Other examples include students using social engineering techniques to gain access to early copies of exam papers. They can include students hacking into computer systems to change their marks. There are also areas where instructor opinions of what constitutes academic misconduct may vary, such as where a student uses chemical enhancements, such as study drugs, designed to allow them to concentrate for longer and hence gain a higher mark than their peers.

There are also areas where academic integrity can be breached, but this may not be deliberate. For example, a student could be judged to have plagiarised as a result of poor study skills. They could have committed research fraud through misunderstanding research methodologies or incorrectly processing data. This is why education and support for students is important. Addressing this once an academic integrity breach has taken place is too late. This is because it is not always possible to tell if this was intentional or accidental.

Table 5.1 Types of academic integrity breaches of interest to the Computer Science instructor

Academic integrity breach	Description
Plagiarism	The process where a student uses the words or ideas of another without acknowledgement. A particular issue in Computer Science is *source code plagiarism*, where program code is copied or reused in an unacceptable manner. A related area is *essay spinning*, where students use automated software, such as that intended to translate between languages, to disguise plagiarised work and avoid detection (Lancaster and Clarke 2009; Jones and Sheridan 2015)
Collusion	Where two or more students have worked closely together during the production of assessed work, going beyond acceptable levels of collaboration. When collusion is taken to the level where the submitted work of two students is identical or very similar, this also represents a form of plagiarism
Contract cheating	The behaviour where a student uses a third party to complete assessed work for them, or attempts to do use a third party in this manner (Clarke and Lancaster 2006). This involves an exchange of benefits for the parties involved, often in the form of a fee. Contract cheating was originally observed primarily relating to the outsourcing of technical Computer Science assignments, including programming, but also covers written work, such as reports. Some contract cheating is facilitated through *essay mills*, companies that are primarily online and which offer to produce original bespoke assessments for students
Exam cheating	Where a student attempts to gain an unfair advantage in an examination, for instance by referring to concealed notes or through secret communication with an individual outside an exam hall. This behaviour includes *impersonation*, where a third party attempts to switch places with an individual who is meant to be there so that they can sit the exam for them. This can be possible for exams taken face-to-face or online. One enabling mechanism for impersonation includes having a test taker disguised to look like the person meant to be taking the exam. A second enabler involves student identification cards being doctored so that the third party does not raise suspicion
Research fraud	The process where research results and conclusions are returned that are not backed up by verifiable evidence. This can include deliberate errors in the research methodology, such as mistreating or ignoring data. This can also include not acting within a defined ethical framework. A specific example is *data fabrication*, where reliable data is not collected but instead produced in a way that benefits the student in achieving the result or conclusion they would like to see. In Computer Science, this may perhaps be an issue during the Project stage of a student's course

A suitable understanding of what plagiarism is and why this is considered a problem will help instructors trying to extend their knowledge to other forms for academic misconduct. Many commentators have linked a growth in academic misconduct with the prevalence of Internet aided plagiarism (Austin and Brown 1999). Students entering higher education have grown up surrounded with the Internet for all of their life and so their concept of the value and ownership of information may be different to that of their instructors.

Computer Science instructors also have to be aware of the types of academic misconduct that appear to be becoming more visible. For example, research into contract cheating has regularly seen assessments from across the Computer Science spectrum outsourced (Jenkins and Helmore 2006; Lancaster and Clarke 2007). Technical assignments, particularly those which do not require English language proficiency, can be contracted out to a global marketplace, often for a cost that is economically feasible for students. The range of assessments that can be outsourced contains activities contributing strongly to a student's final degree classification. This includes major work such as a final year project, assessments completed in stages and tasks where students are intended to provide their own personal reflection.

The use of examinations may remove some of the academic integrity challenges seen with coursework, but examinations themselves can be susceptible to contract cheating. Students have been observed paying or using a third party to provide them with undocumented help in an exam situation (Lancaster and Clarke 2017). There are also examples where students have used an impersonator to take an exam for them. With the move for Computer Science courses to be delivered online, it appears that such courses have particular vulnerabilities where academic integrity can be breached. That means that special attention has to be paid to the design and delivery of such online courses.

5.3 Teaching Academic Integrity Principles

Students need to be equipped with the necessary tools for success as part of their Computer Science course. This includes helping students to develop an understanding of how they can progress through their studies whilst always holding academic integrity in their mind. For the instructor, this requires supporting students to avoid accidentally breaching academic integrity principles.

It is unlikely to be sufficient for an instructor to assume that students arriving at university all share a common understanding of what academic integrity is, why it is important or how they can act with integrity. This can particularly be the case where students arrive from different cultures, educational upbringings and backgrounds.

This means that teaching academic integrity is necessary. Such teaching needs to cover two main considerations.

First, students need to understand what academic integrity means, why this is essential to their studies and how academic integrity values relate to their everyday life.

Second, students need to be shown the practical techniques they will require during their course. These are the techniques necessary for them to avoid academic integrity breaches and to show that they respect the resources that they are working with.

As with many subjects, academic integrity is not a subject that can just be taught once and forgotten about. One model that can be used is to introduce students to the appropriate concepts early on in their course, integrating opportunities for students to receive formative feedback on how well they have engaged with the material. These concepts can then be developed and reinforced throughout the course, with students provided with information specific to the subject they are working on and their level to study.

For example, when arriving at university, students may need to be shown the practical skills of finding and referencing textual information very early on. This teaching would be supported by exploring why valuing the ownership of information is important and the possible consequences that taking information from a source without acknowledgement would lead to in their everyday life. In a career situation, this could lead to an employee being dismissed, Intellectual property breaches could also lead to litigation for the company they work for.

Discussions about other types of academic misconduct and how this can be breached can occur at appropriate points during the student journey. For instance, correct examination conduct and the skills needed to be successful with exams can be explored early on in their course before students take low risk tests. In Computer Science, presentation of the concept of examination success can also be related to the professional examinations, such as vendor certifications. These examinations are ones that students may be expected to take beyond their time at university and throughout their career.

Introducing a discussion on contract cheating can be more difficult. Some instructors may believe that all students already know this is wrong, but such a view does not account for differences in student cultural upbringing. There are cultures where students are taught that using the words of experts is not only allowable, it is expected. There are also students who already know that this is wrong and are still contract cheating. The counter view held by some students, that information belongs to them once they have paid for or acquired it, needs to be openly explored.

Challenging conversations can be framed as part of a wider discussion regarding what level of external help is permissible. This is an important distinction when attempting to build Computer Science student team working skills ready for industry, but still ensuring that students receive credit for their own individual effort and contributions.

Many academic integrity discussions also need to happen at the level of individual subjects. A student learning introductory programming skills does need to demonstrate that they have mastered basic coding concepts, even though the level of exercises set may be trivial to many experienced programmers. This means that taking a code fragment from an existing online codebase to solve a simple exercise regarding manipulating strings, for example, would not be considered appropriate at such an early stage of the student's academic journey. Once a student is required to demonstrate that they are ready for a professional programming career, the expectations on them may be different. In such a case, it may be considered both acceptable and desirable for a student to demonstrate that they can find and reuse existing code fragments.

There may be further complications when a student needs a resource to complete an assessment, but that resource itself is not a key area that is being examined. An example may be on a web development assessment, where a student wishes to improve the look of their website by using existing artwork, but it is their coding skills that are being assessed. In such a case, students need to be made aware of the instructor's expectations of how they demonstrate academic integrity.

For Computer Science, Simon et al. (2016) recommend that exactly what students are allowed to do and what they may not do is spelled out for them on the assignment brief. This may include detailing which individuals and services they can and cannot get help from, which resources they may and may not use and what aspects of the assessment they can and cannot solicit help with. It may also be worth agreeing such a list with students as part of the assessment development regime, thus also helping the student cohort to feel some ownership of the decisions made.

One message that is worth communicating to students is that academic misconduct is not a victimless crime. A student who used dishonest tactics to get ahead could end up getting better marks than a student that worked hard. A cheating student could even get a job that they don't deserve, taking this away from a more honest student. That is why it's important for instructors and students to work in close partnership and develop a shared understanding of why academic integrity is important for all parties operating within the higher education landscape.

5.4 Assessing with Academic Integrity

Both students and instructors have a role to play in ensuring that academic integrity is maintained through the assessment process. As this chapter has already discussed, students need to understand the benefits of completing their own work and commit to doing this to the best of their ability. Students also need to be equipped with the academic skills necessary for their own success.

Instructors have further responsibilities during the assessment process, an area which students often consider the most important part of their educational experience. Instructors need to develop assessments that reduce the opportunity for students to accidentally breach academic integrity. They should also try and remove any of the temptations that may mean that some students would choose to cheat if they thought that they could get away with it. This can be accomplished by rethinking and restructuring the assessment process, or by removing any benefits that attempting to cheat may bring.

Student interest in assessment can be improved by making this more engaging. Mechanisms for doing this will vary between student cohorts, so some partnership working with students is necessary here. It may, for example, be that a particular student group prefers examinations over coursework, group assessments over individual assessments or practical tasks over written ones. It may be possible to co-design assessments with students so that they feel some ownership of the assessments tasks that are set.

There may also be cases where students feel that assessments are simply hurdles that they need to jump. Working with students can help to share the message that these are measures of progress. Demonstrating to students that formative assessment is useful can also pay dividends, particularly in the areas of Computer Science where students need to build up skills gradually, such as when learning computer programming.

Assessments can be developed so that students cannot gain a passing mark by using repeatedly cheating using the same techniques. A simple way of doing this would be to set two different types of assessment within a subject, such as both a coursework and a test. In such a case, the students would be required to demonstrate their competency in both items. In this particular circumstance, a student who had plagiarised their coursework and not been caught would still have to find a different way to cheat during the supervised test element. As well as engaging the students using multiple assessment modalities, this also reduces the benefit of cheating, since the required learning would still need to be completed for the second element of assessment.

A supervised component of coursework assessment can also be useful, particularly where instructors are worried about contract cheating (Lancaster and Clarke 2016). This can include both written tests and practical tests, but also supervised coursework sessions, presentations, spoken viva voce examinations, industry style coding interviews and product demonstrations. There are many opportunities to innovate here.

Employability initiatives are also worthy of consideration. For instance, more authentic forms of teaching and assessment can be used, such as a simulated work environment where students work on assigned projects during office hours to develop and document a solution to a problem posed by an external client. Such assessments can also be of value for students when they looking to present a professional portfolio and to showcase on their Curriculum Vitae that they have acquired a wide range of new skills during their course.

Instructors do need to be wary about the need to balance several trade-offs when developing assessments. They have to balance encouraging academic integrity, reducing the value of academic misconduct, meeting industry requirements and their own ability to manage the assessment process within their wider workload commitments. As an example, an individual spoken examination, or viva voce, may be a very good method of authenticating that a student knows and understand a subject. Despite the advantage the time commitment necessary to do this fairly for all students in a large Computer Science class may mean that a spoken exam is simply not practical. Even if the time required is not the issue, there are still issues of fairness to consider, such as how the instructor ensures that students who have their viva early in the process are not at a disadvantage over students scheduled later on. The latter group could be said to have both more time to prepare and the potential to quiz earlier students over the questions that they are likely to be asked. There can also be challenges posed to the consistency of processes when a large team of assessors is used.

Compromises are possible and recommended. It may be that instructors work together to ensure a varied diet of assessments across the whole year of a course, with extra academic integrity measures put into place during those assessments that underpin later learning. For instance, the crucial assessments in the first year of a traditional Computer Science course might be considered to be those for the core programming and mathematics subjects, an understanding of which is needed for success in later years. It is also possible to set an end-of-year viva, bringing together the skills from across a variety of subjects and helping to check that students have an understanding of how these distinct areas all integrate together. Such a viva can also be resourced using multiple instructors to ensure that this process is manageable. Major modules, such as a final year student project that often represents the culmination of a degree, can also have extra academic integrity checkpoints put into place.

Another option for assessment that combines several recommendations of good practice is for an instructor to structure this using multiple linked components. Here is a simple example regarding how this could be used for a computer programming module, but similar approaches are also possible for other subjects. First, the student completes a coursework assessment as usual, where they develop a solution to a programming problem. They submit their code for that coursework. At a scheduled time, they attend a practical examination where they are required to make changes to the code they already submitted. This approach requires students to be familiar with their code, reducing the usefulness of them having outsourced its production or used other cheating methods. Where students chose not to engage with academic integrity during the coursework process they would still have to master the programming skills required for the practical exam. The student should realise that there is little benefit to them not putting in the effort during the coursework phase, hence encouraging them to concentrate properly on the assessment.

Most of the examples in this part of the chapter have focused on coursework assessment, but examinations are also susceptible to cheating. Even an activity like a viva voce examination can be cheated on if a student has advance access to a set of standard questions, or if someone outside the exam room can feed the student answers through a concealed earpiece. Some examination practices used by instructors are themselves questionable, for instance where the questions have been taken from previous papers that students have access to, or match those provided in a sample paper. This would make the exam merely test of memory skills and not ability. Hence, instructors need to be continually aware of opportunities where their own practice can lead to the manifestation of academic misconduct.

Academic integrity in examinations can be further preserved by carefully developed examination processes. This includes specifying the equipment that students are allowed to use, supplying equipment they cannot have tampered with such as pens and calculators, monitoring and recording screen activity during practical examinations, and ensuring that all examination questions are original and go through a robust quality checking and moderation process. If nothing else, all examinations need to be invigilated and students alert to the fact that their actions are being watched, thus removing the benefit from any opportunistic attempts by students to cheat.

5.5 Detecting Breaches of Academic Integrity

Instructors need to be aware that detection tools and punitive measures are not a single solution to academic integrity challenges, but they can be used to support other approaches. Tools can also provide instructors with some assurance where they believe that academic integrity has been breached. Many software tools already exist for the detection of breaches of academic integrity and the choice of the most suitable tool has to be closely matched to different situations. But software cannot be considered a complete or infallible solution. It may always possible for students to cheat in ways that tools do not detect. Many web pages, videos and social media posts exist where students share the latest ideas to defeat detection technology.

Tools for detecting traditional forms of plagiarism are available and this is a field that is well-established. These tools are available to identify similarity in written text, within source code submissions in many programming languages and for specific technical environments such as spreadsheet assignments or database assignments. These tools flag student submissions that warrant further manual investigation by an instructor. For instance, this may include a written student submission where sections of the work match Internet sources. The results from the tools do not directly indicate plagiarism, so human judgement is required. There are many acceptable reasons why a tool may give such a false hit, for instance two textual documents may appear to match each other, but the cause may turn out to be the use of a correctly cited quotation that was common to both documents.

Although plagiarism detection tools are well-established, tools designed to detect contract cheating are not so readily available. Indications are that students are turning to third party writers, programmers and other contractors precisely because the original work they supply is unlikely to be detected using current plagiarism detection technology.

The best advice available at present is for instructors to approach assessing coursework with a keen eye and questioning mind, with the view that that someone other than the student may have produced this item of coursework. For an instructor who knows a student, their abilities or their writing style, such simple checks as looking at the properties of the document they submitted can be useful. These may show suspicious file creation dates or author names.

Several techniques that could be used to support the automated detection of contract cheating have also shown promise. Checks that the writing style of a student remains consistent from one assessment to the next are possible. Some work has been made to automate such checks (Juola 2017). Similar techniques could be used to decide if student programming style remains consistent from one exercise to the next. It has been suggested that contextual information could be extracted from student submissions to help develop artificial intelligence systems to detect contract cheating (Lancaster and Clarke 2014). Student mark profiles can also be analysed to see where student performance shows inconsistencies of the type that may suggest that the student has received external help (Clare et al. 2017).

The use of tools to detect academic misconduct is an important part of any complete academic integrity process. Using tools protects the value of the results and overall qualifications awarded to the students who are working with academic integrity. Their use shows that academic integrity is being treated seriously. The existence of such tools and evidence of their use also provides a deterrent effect to students who may otherwise have considered cheating, but have changed that view due to the risk that they may be caught.

If an investigation suggests that a student has breached academic integrity regulations, it is important that a fair, transparent and consistent approach is followed. This means that all students in this situation are considered using an identical process and treated in the same way. Doing this ensures that the wider higher education setting also complies with the underlying principles of academic integrity. Where a fair process does identify that a student has breached academic integrity, this also has to lead to a suitable outcome, such as a fairly and consistently applied penalty for the student. There may also be the need for arrangements to support the student in future to be put into place.

The same technology used to detect plagiarism and potentially penalise students can also be used in a positive manner. For example, plagiarism detection software can be introduced in a formative way to allow students to find out if they are accidentally demonstrating poor academic practice (Halgamuge 2017). As this is formative, the process can then be used to put support into place for students, rather than attempting to penalise them. For instance, where a student draft report is run through plagiarism detection software and found to contain copied text, the student could be provided with support on how they should correctly acknowledge and reference information from an external source. Such support is useful for students to ensure that they do not accidentally breach academic processes when in a summative situation.

5.6 Conclusions

An understanding of academic integrity, including how academic integrity can be promoted to students and how academic integrity should be continually maintained, is important knowledge needed by all current Computer Science instructors. This chapter has motivated the need for academic integrity in Computer Science education and has provided an overview of current thinking and ideas within the field.

When students cheat, many will have considered that they are taking a risk in return for a potential reward. For such a student, cheating has to offer them some sort of benefit, such as passing an assessment point or obtaining a better mark than they otherwise would have done. That benefit has to outweigh the chance of them being caught and the likely knowledge of what will happen to them if they are found to have breached academic integrity.

Some students cheat because they do not know any better. That risk can be mitigated against by ensuring that a programme of academic integrity tuition and support is a core component of study for all students.

Other students cheat because they fear failure and believe that their academic misconduct is unlikely to be detected. That risk to integrity can be reduced through careful instructional design and considered assessment processes.

There are students who cheat because they do not see the value in what they are being taught or the assessments they are being asked to do. This risk can also be mitigated against by developing engaging teaching and incorporating real world examples.

In Computer Science, there are also students who have the ability to complete assessment for themselves, but just want to show that they can beat the system. This is an example of the long-standing hacker mentality already evident in the Computer Science population. There is no single solution to prevent this, but here simply ensuring that students are kept engaged and interested may offer the key.

Academic integrity is not presented as just a problem to be solved. It may be impossible to completely ensure it, particularly when students are technically sophisticated and the tools they have available to help them to take shortcuts to succeed are always changing. After all, students are using methods of cheating now that were unheard of a decade ago. But there are also great opportunities for the Computer Science academic. Those academics are well-equipped to be at the forefront of future academic integrity research. Such academics are more likely than most to have the technical skills necessary to conduct research into ongoing topics as plagiarism detection or emerging topics such as the detection of contract cheating or essay spinning.

Academic integrity within the Computer Science discipline can also be supported by using a wide variety of assessment types, including work simulations. Not every career path has the same view of integrity, even though documents such as professional body authorised codes of conduct offer some general principles. Within Computer Science education, there are areas where the best solution towards ensuring academic integrity is not clear, such as where the border between unacceptable collusion and expected workplace collaboration is concerned.

The opportunities for Computer Science academics to develop processes, definitions and solutions for the preservation of academic honesty also offers opportunities for these academics. Many academic integrity challenges, such as those linking employability and collaboration, are likely to be more prevalent in Computer Science than in other academic disciplines. This also means that the prospect exists for the Computer Science specialist to take a leadership role in the academic integrity field.

Above all, academic integrity needs to be more than just an add-on to Computer Science. Many of the same principles used to design good assessments also reduce the opportunities for academic misconduct. They are already designed to engage students and to help them to succeed. An instructor working in this manner is therefore operating with many of the most important principles of academic integrity at heart.

References

Austin M, Brown L (1999) Internet plagiarism: developing strategies to curb student academic dishonesty. The Internet and Higher Educ 2(1):21–33

Clare J, Walker S, Hobson J (2017) Can we detect contract cheating using existing assessment data? Applying crime prevention theory to an academic integrity issue. Int J Educ Integrity 13(1)

Clarke R, Lancaster T (2006) Eliminating the successor to plagiarism? Identifying the usage of contract cheating sites. In: 2nd plagiarism: prevention, practice and policy conference 2006, Newcastle, United Kingdom

Exemplary Academic Integrity Project (2013) Resources on academic integrity. https://lo.unisa.edu.au/course/view.php?id=6751&section=6. Accessed 18 Apr 2018

Halgamuge, M., 2017. The use and analysis of anti-plagiarism software: Turnitin tool for formative assessment and feedback. Computer Applications in Engineering Education

Jenkins T, Helmore S (2006) Coursework for cash: the threat from online plagiarism. In: Proceedings of 7th annual higher education academy conference in information and computer sciences, Dublin, Ireland

Jones M, Sheridan L (2015) Back translation: an emerging sophisticated cyber strategy to subvert advances in ‘digital age’ plagiarism detection and prevention. Assess Eval High Educ 40(5):712–724

Juola (2017) Detecting contract cheating via stylometric methods, plagiarism across Europe and beyond 2017. Czech Republic, Brno

Lancaster T, Clarke R (2007) Assessing contract cheating through auction sites—a computing perspective. In: 8th annual higher education academy conference in information and computer sciences, Southampton, United Kingdom

Lancaster T, Clarke R (2009) Automated essay spinning—an initial investigation. In: 10th annual higher education academy conference in information and computer sciences, Kent, United Kingdom

Lancaster T, Clarke R (2014) An initial analysis of the contextual information available within auction posts on contract cheating agency websites. In: Proceedings of International workshop on informatics for intelligent context-aware enterprise systems (ICAES 2014), University of Victoria, Victoria, Canada

Lancaster T, Clarke R (2016) Contract cheating—the outsourcing of assessed student work. In: Bretag T (ed) Handbook of academic integrity, SpringerReference

Lancaster T, Clarke R (2017) Rethinking assessment by examination in the age of contract cheating, plagiarism across Europe and beyond 2017. Czech Republic, Brno

Simon, Sheard J, Morgan M, Petersen A, Settle A, Sinclair J, Cross G, Riedesel C (2016) Negotiating the maze of academic integrity in computing education. Proceedings of the 2016 ITiCSE working group reports, pp 57–80

Part II
Teaching: Examples of Practice

Chapter 6
Why Is Teaching Programming Difficult?

Carlton McDonald

Abstract Teaching 1st year programming is a major challenge at all universities. It doesn't seem to matter what programming language is used, how much support is provided to the students, or how the students are assessed, or at which university the teaching and learning takes place. Learning to program is hard enough as it is (Pine, in Learn to program. The Pragmatic Programmers, 2009). Given that an A level in Computer Studies is still not a prerequisite for admission to a course teaching computer programming although A level Mathematics often is. We look at a number of issues around the difficulties of learning to program, the vast change in the nature of programming language concepts, libraries and application areas, and ask the question, are expectations of beginner programmers realistic given the short amount of time given to learning to program?

Keywords Continuous assessment · Formative feedback · Student motivation Portfolios · Learning to program · Teaching programming

6.1 Introduction

Teaching beginners programming is riddled with challenges:

- What's the point of programming?
- Motivating Students
- Why can't I dive straight into programming?
- Language Learning
- Cooperative learning
- Weekly, consistent learning

C. McDonald (✉)
University of Derby, Kedleston Road, Derby DE22 1GB, UK
e-mail: c.g.mcdonald@derby.ac.uk

- Incremental Learning
- Language Problems
- Assessment versus Learning.

This chapter looks at these challenges and how there is a good case to review the expectations of students particularly on degree courses, given the changes of programming languages, increased language complexity, and what can be learnt and applied in the contact time given to students new to programming. Programming skills are applied to web pages, web servers, mobile, distributed, cloud and systems programming to name just a few. However, many of the new application contexts need a new programming language in a browser context. JavaScript has a unique document object to become familiar with. It is neither obvious or intuitively understood, Programming the document object needs to be taught/learned. We look at the issues at each stage of the learning programming process up to undergraduate programming and some of the reasons that learning to programme is difficult.

6.2 What's the Point of Programming?

Imagine teaching origami. One of the first things you want to do is to show your students the things that can be produced with origami. Seeing these creative objects inspired some pupils to want to learn how to do it. As a seven or eight year old at primary school, the author remembers being excited by Origami and the different animals and objects that could be made. The discipline of folding the edges as precisely as possible, following written instructions, is effectively being a computer executing written instructions. The process of reflecting on origami exercises enables pupils to see the importance, not just of the quality of the instructions, but also of the accuracy of the execution. Many of the classmates struggled, at times, to follow the 2D paper based instructions attempting to illustrate folding movements, however the more one follows instructions, as difficult as they are at first, the easier they become to understand. Those that invest the time become experts at understanding the language in which the instructions are written. This is only half of the skill required to produce origami artefacts.

In addition to being able to understand the program, the robot following the program instructions needs to be able to meticulously and precisely follow the instructions. Those children that fold a square in half in order to make a fish or a bird, need to produce an accurate square to start with, and a precise diagonal fold in half, in order to have a reasonable looking and flapping bird at the end of the process of paper folding. These observations enabled the author to recognise the importance of following instructions carefully. The experience of following origami steps from books also occasionally meant that the occasional program didn't work because the instructions describing what tasks needed to be completed were unclear.

Traditional *sequential* programming is akin to writing the origami procedure in a step by step manner for a robot (the computer) to follow. The average programmer can manage to produce a program with less than 20 language constructs (if—then—else, while, for, arrays, procedures, classes, constructors, procedure and function calls etc.). This is much more than the Basic programming language (variable declaration, types, if—then—else, while, for, arrays, and procedures) that could be used to solve any computer problem, although not in the most efficient way. This relatively tiny set of instructions meant that programs took tens of thousands of lines of code, in the same way that a human language (like Tok Pisin, the *lingua franca* of Papua New Guinea) requires many more words to describe concepts than a language like English with its 100,000+ words.

The purpose of programming is to provide instructions to a machine to carry out a task or series of tasks. For example, book an airline flight, play a game, make a bank transaction, and billions of other tasks.

6.3 Motivating Students

The approach used by the author to get students excited about programming is to introduce the first lesson as an opportunity to program their own robot. This piques their attention. Students that have already started to nod off, wake up; those that are slumped, sit up excited with anticipation to see their robot. In preparation for programming their robot, they must first learn the simplest set of instructions that the robot will understand:

Forward *n* steps
Turn left
Turn right

The robot is to be navigated from any one corner of the classroom to the diagonally opposite corner of the classroom using a sequence of the above three instructions. It is surprising how these three instructions can be used to fulfil all navigation tasks in a single storey building, or even a city on foot.

6.3.1 Introducing the Robots

In order to see how good they are at programming these instructions need to be provided to a robot. Students are then paired off, if there is anyone that is not paired off they work with the tutor. Once they have settled down, they introduce themselves to each other and are notified that their partner is their "robot". They have to take it in turns to be programmed in an interpretive manner, i.e. receive instructions, one at a time, until they are not able to complete the instruction, or the program has completed.

The results of testing their robots indicate a number of issues. Firstly, is the need for initialisation instructions, e.g. stand up, face the front. These aren't in our very restricted language but all programs require initialisation statements. The second issue is how dumb robots are! Why can't they see that the desk they are about to walk into is going to cause them damage? Robots, computers, automatons do exactly as they are told by their programmers. If something goes wrong, it is always the programmers fault. This requires the programmer to revisit the program and edit the instructions. Programming robots to perform physical tangible tasks is comparatively easy in contrast to algorithmic intangible tasks where formulae and variable manipulation is required (see Variables later in this chapter).

6.3.2 Testing

How important is it to test the program? Having constructed the program, we cannot just assume that it will work. Testing, getting their robot to carry out the tasks called out to them is how the programmer is able to see if the program is ready to be distributed to all robots.

6.3.3 Single Solution

When asked, what are the limitations of their program, it is identified that the program only works in the room we are currently in. It is not a program that can be generalised to navigate any room. It is at this point that the purpose of decision making in programs is required. In addition to decision points it is also pointed out that we also need to be able to repeat tasks. We then go through a couple of iterations of modifying the program to add to the instruction set:

If condition is true **then** do task1
If condition is true **then** do task2 **else** do task3
Repeat task4 **while** condition is true

Programs are then modified, robots deployed, crashed, reviewed and students leave understanding programming languages, instructions, initialisation, testing and incremental development having had a fun first class.

6.3.4 Learning Attitudes

Most programmers are extremely confident and having written the program assume that it is correct after the first attempt. Many students are lazy. When asked to write their program, it is surprising how many of them have to be told to get out their pen and paper, tablet or laptop and put away their mobile phone! It seems that many of the students educated in the UK are just waiting for the tutor to provide the solution. This is a major problem.

Tell me, and I will forget, show me and I will remember, let me do it and I will understand.

This spoon fed generation just want the answer. Don't make me think! It is at this stage of the first class that those that are going to do well become apparent to the tutor. Programming is a practical activity, those that immediately set about writing the instructions are those that will be natural programmers. They are prepared to have a go, confident *they* can do it. They come into the class enthusiastic and are eager to start. When the program fails, they are completely undeterred, even more determined they can do it.

Other students, even though the task is straightforward and not a task that can only be completed by degree students, immediately start to look around to see what others are doing. It is as if they are not sure what they are supposed to be doing. After discussion, it emerges that what they want to know is there no instructions that say: "stand up", "turn away from the desk". They don't know how to initialise their robot. When told they can indicate some initialisation steps, this second group of students still continue to look around, typing/writing nothing until told to write something. It is surprising how tentative and uncertain these students are, as if they daren't get anything wrong lest they be expelled or imprisoned should anything go wrong. The third group of students are those that are in class but not involved in class. They seem reluctant to speak, reluctant to answer questions, reluctant to get into pairs and seem embarrassed as they pace out the instructions when it is their turn to be the "robot". Once in a while one or two of these students work well on their own and feel a little awkward in public. This third group of students are a challenge to motivate and will be the group that ask, "what do I need to do to pass"? "What do I need to do to get a first"? More interested in the qualification than the learning associated with the course.

6.3.5 Variables

One of the most difficult concepts to teach a complete beginner is the programming concept of a variable. Having attempted to teach 16 year olds programming in Papua New Guinea, the author spent a two hour lesson trying convey the idea of a variable. Why do we need a box to put things in? Was a question raised in trying to convey the concept of a storage place for things we want to remember in the program. Why does the robot need a pocket to store a single item? why doesn't it just remember the value? The computer has memory doesn't it? The questions are a mass of confusion and are akin to the difficulties encountered by children attempting to learn algebra: "what are *x* and *y* for?", I asked my parents and they said they have never used simultaneous equations in their life, why do I need to understand it?

Unfortunately, it is true that not everyone who registers for a Computer Programming course, having never attempted programming before, will get past this stage. If they have never grasped algebra (even the word sounds alien, they think to themselves), it is an extremely challenging hurdle to overcome, because it involves abstraction: not having a physical or concrete entity. This is perhaps the reason that the course with the highest drop out rate of all UK degrees is Computer Science (Pine 2009). Admission to a Computer Science course therefore requires GCSE mathematics, not because programming is a mathematical activity but because GCSE requires the ability to think in an abstract manner, manipulating concepts in the mind, passing mentally through each instruction of the program with the aid of variable values on paper in order to determine where we are in the program and what is happening at each point in the program.

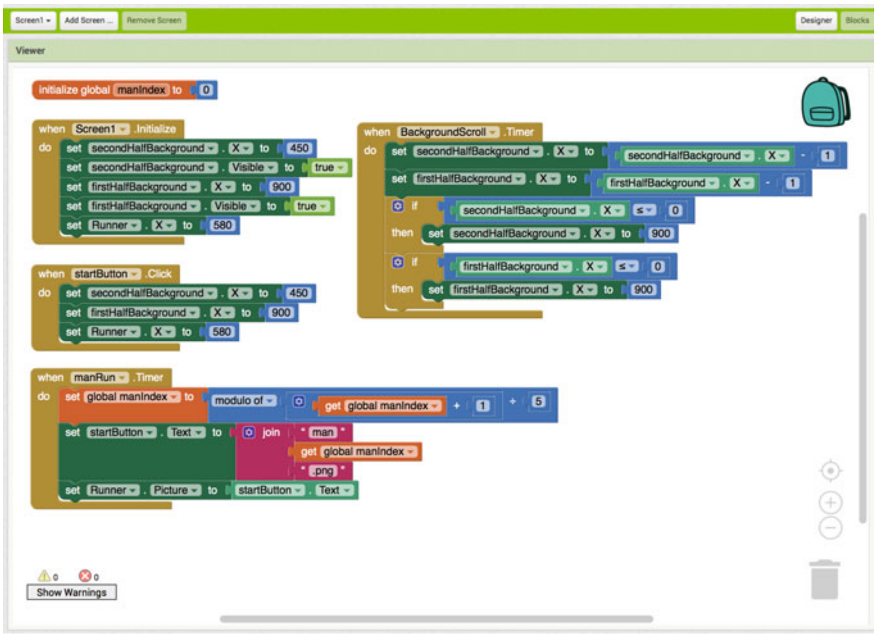
6.3.6 First Class Reflection

Programming robots has always proved a fun activity to get the class experiencing and reflecting on programming processes and concepts. Unfortunately, when the rest of the course is focussed on programming a machine, the kinaesthetic learning is perceived by some as simply typing on a keyboard. That's barely kinaesthetic, otherwise typing is the way in which kinaesthetic learners can learn anything, nevertheless writing programs in a computer laboratory is referred to as a practical! And that is what it is… to some students! Students build programs, which interact with users and produce results displayed on a screen. Kinaesthetic learners are able to make that association fairly easily. They find completing programming tasks quite fulfilling, whereas others just see text and pictures on a screen and it doesn't evoke positive feelings other than: "I have the marks required of this week's practical, and can still pass the module". For them programming is not a labour of love, just labour.

In the same way that with any practical skill, the more time that is spent doing the activity the better one becomes. Consider origami, playing badminton, driving a car… The real challenge for the programming lecture is to motivate those students that aren't natural problem solvers, that took a while to get into the robot programming and find that the keyboard programming is very unfulfilling. It is for this reason that it is the author's view that students are far more likely to spend time producing programs if those programs enable them to create programs for themselves or their family and friends to use. A tiny percentage of the world's population use DOS (whatever that is), Windows or programs that are executed on the command line. It is time that such introductory programming courses were commanded to be put on a line and executed. Everyone interacts with a browser, so although not a rich instruction set, well structured, and limited Object style programming is available, JavaScript is the most important language in the world. Write JavaScript programs and everyone in the world is able to use your program. This inspires many student, whereas a C# program in 2018 seems hardly relevant to hand held environments from which people interact with software. Even server side programming requires a good grasp of JavaScript concepts.

In a similar vein, the web is a quarter of a century old, and mobile devices are ubiquitous, permitting access to a browser but if students are able to develop apps that they can show to family and friends this too might motivate them much more than DOS programs.

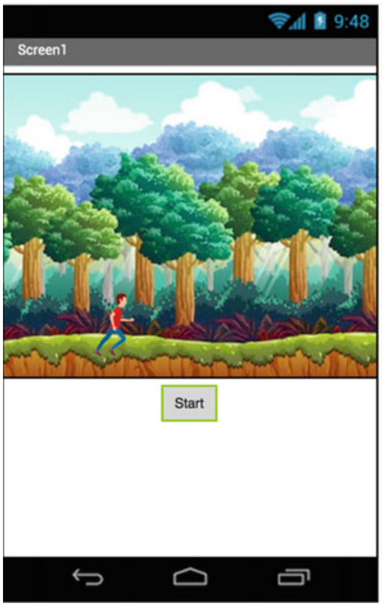
In the 1980's Computer Science purists were adamant: the only way to teach programming is to teach an assembly language. It is a while since such nonsense has been vocalised in a university but the Computer Science purists are now insisting that the only way to learn to program is with C++, C# and Java. Such arguments are way out of date and have some basis in the fact that many lecturers believe that the way in which they learnt is the best, the languages they are experts in, are the best for all time. It is amazing that the most profitable computing company in the world ditched one of these antiquated tools relatively recently. Apple devices were programmed in Objective-C, a language way ahead of its time when it was first created at the Stepstone company in the early 1980s by Brad Cox and Tom Love. It was licensed by Steve Jobs' NeXT Computer Inc. in the late 1980s but way behind the times when finally superseded by Swift in 2013. Yes, speaking to the die hard purists, an expert Objective-C programmer can make the program far more efficient than the best optimising compiler. However, the best Objective-C program could cause the greatest harm when memory access is to an unintended area. We need the best high level programmers, not low level programmers. When was the last time a modern developer wrote an assembler program? Only a very small percentage need learn this interesting programming paradigm. Nowadays we need the best web programmers and mobile programmers. As a result, the first environment in which introductory programming, in the module Principles of Programming, is taught is MIT's AppInventor. AppInventor was a Google Project, which they passed to MIT in 2012, and made open source (The Verge 2018) (Figs. 6.1 and 6.2).



```
initialize global manIndex to 0

when Screen1.Initialize
do  set secondHalfBackground.X to 450
    set secondHalfBackground.Visible to true
    set firstHalfBackground.X to 900
    set firstHalfBackground.Visible to true
    set Runner.X to 580

when startButton.Click
do  set secondHalfBackground.X to 450
    set firstHalfBackground.X to 900
    set Runner.X to 580

when manRun.Timer
do  set global manIndex to modulo of (get global manIndex + 1) ÷ 5
    set startButton.Text to join "man"
                                 get global manIndex
                                 ".png"
    set Runner.Picture to startButton.Text

when BackgroundScroll.Timer
do  set secondHalfBackground.X to secondHalfBackground.X - 1
    set firstHalfBackground.X to firstHalfBackground.X - 1
    if   secondHalfBackground.X ≤ 0
    then set secondHalfBackground.X to 900
    if   firstHalfBackground.X ≤ 0
    then set firstHalfBackground.X to 900
```

Fig. 6.1 A complete AppInventor program that animates a runner in front of a scrolling background

Fig. 6.2 Visual representation of the animated runner and scrolling background



AppInventor is a colourful jigsaw puzzle style programming environment for Android devices only. It is so easy to program Android devices that schools around the world are using AppInventor to teach mobile programming to very young children and upwards. AppInventor uses pieces to construct programs by dragging visual components and dropping them onto other visual pieces called "blocks" in a blocks editor.

At the time of writing, AppInventor is being ported by a couple of the original AppInventor creators Arun Saigal and WeiHua Li, to iOS devices through a platform Thunkable (Thunkable Thoughts—The How and Why You Would Make Your Own Beautiful App 2018a). The interface and programming constructs currently work, however, sensors, GPS, Accelerometer etc do not currently work in an iOS environment. Read about the exciting Thunkable platform developments here: https://blog.thunkable.com/ (Thunkable Thoughts—The How and Why You Would Make Your Own Beautiful App 2018b).

One of the best things about AppInventor is that there is a really easy method of designing User Interfaces or the visual representation of the app. It is the visual design that determines a high percentage of the blocks that are needed to construct the program.

Students require no real technical skills (other than drag and drop) in order to put the interface together and so in a short space of time they can have all screens designed before attempting to write the code for the app.

The feedback from the students is much more complimentary with regards to learning to program. Students still find it difficult to solve problems in the blocks editor where the programming is performed. With the logic of if-then-else and loops being the main hurdles. This is particularly difficult when these are "nested" within other if-then-else or loop statements.

AppInventor 2 (AI2) is not as good for learning to program as the original AppInventor Classic because AI2 does not allow the program to be debugged (going through the program step by step and examining values). This inability to debug, by manually stepping through the code, one instruction at a time, makes it impossible for students to learn how debug a program. Stepping through a program, step by step, statement by statement, is an excellent way of understanding the flow through a program. A beginner is able to go through the program step by step, over and over again, until they are able to predict what happens next. Debugging also allows the debugger to examine the values of variables, UI component states, values and sensors at each point in the program.

This is the only downside of using AppInventor. This lack of debugging was a serious factor as to whether to continue to teach AppInventor. However, the reason it was continued, is that students feel empowered to develop programmes at a high level and quickly. This makes AppInventor the ideal platform for Agile Development Methodologies. It is also perfect for scrums and sprints. Take a break, watch Agile in 5 min http://www.youtube.com (Krishna 2018), Watch Scrums and Sprints in 7 min: http://www.youtube.com (Youtube 2018).

In the early 1980s learning to program on mainframe computers was difficult because students could only practice at their place of learning. In the early 1990s introductory programming was inevitably producing command line programs despite being in a Windows environment. This meant that students were producing programs that were not up to date as far as the beginner was concerned. Most universities used Windows Operating Systems, but Windows programming was not for the faint hearted, so students would often finish a degree and not have written a single Windows program. In the late 1990s the browser was the platform of choice for many users, browsing (or surfing as it used to be called) really took off. Nevertheless, few universities taught Applets or JavaScript as the introductory programming language so that the beginner could develop software for themselves, their family and friends. We were still producing command line programs.

AppInventor has changed all of that, students are able to design software for the ubiquitous Android platform. Some of them show what they have done to their family and friends and invest time, outside of class, practicing their skills. Others, too many others, do nothing from one class to the next.

Motivating students so that they want to come to the practical is most of the battle to learn programming. Ease of development of something meaningful in a few minutes rather than "hello world!" is much more of an inspiration to get out of bed. Has anyone noticed how many students struggle to get to class at 9:00 am? AppInventor, enables students to produce apps that can solve all sorts of problems that students can feel a real sense of accomplishment. A tiny number of absolute beginners, each year, really appreciate and express how much they love programming in AppInventor.

6.4 Why Can't I Dive Straight into Programming?

Students often dive straight into writing their program without thinking what they actually want to accomplish. This is like an engineer putting a big plank wood across a river and saying that he has built a bridge as he walks across the plank successfully. When he rides his motorbike across and ends up in the river below it shows the wisdom of "thinking before you program"! When learning to speak a new language, no one would attempt to construct sentences and paragraphs in the new language without first thinking what they would say, in their native tongue, and then translating that into the new language. Even if the new language is understood it is better to write a series of instructions to get to the supermarket for your Spanish neighbour. In programming terms this would mean writing in one's native tongue, in addition to drawing what one wants to happen in the program and then converting the written natural language instructions to programming language instructions. This early thinking, design stage, helps one to attempt to tackle each of the events that the user might generate, or environmental events (location, altitude, device orientation etc).

Students want to do things as efficiently as possible in terms of the time required by them to complete a task. They feel that trial and error is a good way of solving problems… after all it has served them well so far (with the exception of the rubic's cube which they went on to YouTube after the first unsuccessful attempt to see how it was done). This approach results in the idea that it is better use of time to produce the program straight away rather than spending time designing it on paper because whatever goes on paper still needs to be transcribed into a computer program, this, for many beginners, is a waste of time, why not just write the program using a language that is unfamiliar, for a machine that is unfamiliar.

Even expert programmers put some ideas down on paper, helping to organise the difficult parts of the program in their head rather than producing code that may need to be reorganised. It is better, always better, to structure the program on paper. It is similar to planning the construction of building, any builder who can save you money by not "wasting time" with drawings, calculations and designs is someone you will find has lots of dissatisfied customers living in houses that still aren't finished, or with the bathroom opening out into the dining room. It works as a house but it clearly was not thought through the way in which the components of the house would fit together.

6.5 Language Learning

Learning a human language requires a number of stages unless one is born and hears the language from birth. The first stage is to hear the language over and over and over again. One firstly, begins to recognise a few words, followed by recognising the context in which they are used. Learning the grammar of a language is as important as learning the words making up the language. In programming, the syntax (grammar) of languages has continued to grow, with modern languages perhaps reaching 60 or 70 words and symbols, whereas early programming languages had perhaps as little as 20 words and symbols. Almost any program can be written with the following 3 components: sequence, selection and iteration. Sequence is the way in which one statement is followed by a second statement. Selection are the decision points in a program. All selection points in a program are effectively a combination of a number of if-then-else statements. However even the else is unnecessary if every statement is preceded by its antecedent:

```
if age > 18 then canVote = true
if age <= 18 then canVote = false
```

It is more efficient to use an else:

```
if age > 18 then canVote = true
else
    if age <= 18 then canVote = false
```

The reason using an "else" is more efficient, is that the second 'if' will only be performed if the first 'if' statement fails. We actually use the expression "executed" rather than "performed", this is frightening, subconsciously, to the beginner programmer, it slightly adds to the anxiety, particularly that of people that grew up when a personal computer cost £3000, and one daren't break something that cost as much as a used car.

A reason that some students don't make the transition to programming is whereas everyone learns the language of communication from birth, not everyone learns the language of logic or problem solving. Those students that enjoy solving problems find it easier to learn programming than those that never play strategy games, or solve puzzles. The skills can be learnt but many cannot see the point, they much prefer the answer to be provided to them than have the joy of solving the problem themselves. Most university computer science courses have a separate module covering mathematics, very few have a separate problem solving skills module.

Programming may be applied in numerous contexts: desktop, browser, mobile, server-side, distributed, cloud, server and system are the main areas expected of most Computer Science graduates in 2018. Eight different types of programming, few of which the students can be experts in given the time constraints of undergraduate degrees. However, there are other languages for AI and natural language processing, robotics, more libraries than one could learn in a lifetime, so the problem is not just learning a language as it was 40 years ago. The diversity of all of these application areas mean that the base from which the students need to build needs to be much wider than that of 30 or 40 years ago. There are many more concepts expected to be understood by today's programmers and the academic revalidation cycle means that concepts are changing far quicker than the curriculum is changing. There are so many programming libraries that students have to have an idea of where to find things. However, many computer science courses seem to be little different to the approach of 40 years ago. The author did his introduction to programming in 1981. Pascal programming culminated with arrays and pointers at the end of a year, we wrote our own procedures and there were no objects. The ability to apply today's programming skills to, say, a browser environment manipulating components as well as the document is non-trivial, and a different language is used for programs in a browser environment, yet another for mobiles, and still another for server-side programming…

6.5.1 Language Problems

It depends which lecturer one speaks to as to what their preferred approach is. What's more, there is bound to be an adverse comment about the language, tool or platform:

"Java is the best language", says the module leader, Professor Indonesia, "it is the most widely used in industry".

"No, it isn't", says Dr. Australia, taking the practical on Tuesday, "Python is the language to use, it is functional", Wednesday's tutor, James Gosling, insists on C# because it is the most widely used in industry.

"I thought Java was" says the student, "that's what Professor Indonesia told us".

"He's a bit out of date", says Gosling, "C# is sort of Java with reliability, productivity and security deleted" (Joy 2002), he continues, "it has few restrictions and is therefore more flexible".

The tutor for Thursday's group, is the most popular with the students but they don't understand anything he says. "Ruby is a dynamic, interpreted, reflective, object-oriented, general purpose programming language“. says Yukihiro Matsumoto. “Ruby was influenced by Perl, Smalltalk, Eiffel, Ada and Lisp” (En.wikipedia.org 2018).

“We don’t know who any of those people are”, say the class. On Friday, the lecturer nearing retirement, Carlton McDonald, says the best Programming language for complete beginners is Prolog, Programming in logic. Research has shown that if you have never programmed before Prolog is easy to learn because you say what you want, rather than how you want it done. This is declarative programming.

Such a diversity of opinions further adds to the confusion amongst students, but the lack of an agreed best first programming language and development environment is just the beginning of the language problems. The amount of terminology used in programming alone is enough to make every medical student respect the drugs that the computer programming students have to have a working knowledge of:

Protocol (sounds like cortisol)
Inheritance
Binding
Scope
Closure
Override
Exceptions
Polymorphism
Parameter
Public
Private
Protected
And very many more…

Medical students spend years learning their drugs but are able to practice with a working knowledge of a subset of significant drugs within a few years. It is equally hard for the programmer because many of these terms are abstract, refer to relations between components (organs, vessels and bones) and systems. A program has a complicated structure, and there are patterns that can be used to put the body together. Programs can contain hundreds of thousands of lines, and are very complicated systems yet we expect students to learn programming in one or two 20 credit modules in the first year of their degree. This is not realistic.

A medical student must take and pass A levels Biology, Chemistry and Maths with the highest grades. This is not the case for computer science courses. Perhaps its time to review the effort required to learn how to program. There are an awful lot of students that manage to pass the programming modules, sometimes by referral, but never want to be programmers after graduation.

Programming languages preferences, and the sniping from one camp to another, confuses students in the friendly fire. The programming concepts that are expected of modern high level programming languages have not been estimated in terms of the time taken to learn these concepts. If one is struggling to understand the languages and concepts it is impossible to explain the difficulty one is having at a particular time. If the student can't express the difficulty, how can the tutor help the student? Repeating what was said five times, only louder isn't going to help. Many of the concepts need practical examples, followed by practical applications in order to fully grasp them. 12 hours per week is nowhere near enough for learning programming. Two or three hours of supervised practicals are nowhere near enough for beginners, especially when 180 minutes in a class of 20 is only 9 min of one to one conversation on average per student per week. Over 12 weeks this is a little over an hour. What chance do the first year students have?

Are the expectations of the tutors of students too ambitious given the enormous changes in computer programming concepts, languages, libraries, application areas and paradigms? This review is overdue and may lead to a reduction in the 10% of students that fail Computer Science courses in the UK. However, it needs to happen at a high level, perhaps coordinated by HEA?

6.6 Cooperative Learning

The author has for decades insisted on individual assignments for an introduction to programming to be certain that students understand programming rather than understanding written programs. However, in 2017 the Programming Principles module had two cohorts: one from the department of Computing and Mathematics and a second from the Business School doing an IT for management as part of a business course. It was here, 32 years after first teaching programming, that it was observed that the students that work together on solving their problems were those that achieved the best grades. There is no need to insist that students work solely alone. Learning actually takes place more when students are learning in groups. Students prefer the safety of asking their friends more than asking the tutor in front of the rest of the class. As long as students understand what they are producing at the end, there should be no hesitation in permitting cooperative learning.

Those that explain to their friends however, sometimes have a program that works but not a program that works well. It is at this point that we have to address efficiency. We can write a statement seven times or, have a loop that goes from 1 to 7 performing the same statement each time around the loop. "It works", says the student, yes but what if you need to do the same statement 100 times, or 100,000 times? We don't just want something to work, on a mobile device there is not much worse than using an app that consumes excessive amounts of battery because it works inefficiently, or that has run out of memory because the programmer couldn't find an efficient was of writing the program to be compact and also to perform the tasks with as few statements as possible. Once this has been explained to the small group, or the entire class, if many of them are making the same mistake, or had the same hiatus, they will have learnt something. We often learn from our own mistakes; a wise person will learn from the mistakes of others.

6.7 Weekly, Consistent Learning

Nine times out of ten, the portfolio approach to 100% coursework requires the students to demonstrate all of the work in the final week or just after the end of the module and all in one go. During these demonstrations many of the students have no idea how the little program they wrote ten weeks earlier, or in some cases, a few days earlier, works.

There is no doubt that if students are assessed at the end of the session at which they are instructed they will remember more if:

- They have had to apply the knowledge through practical activity
- The more they have had to apply it, the more they will remember what they did (repetition deepens impression)
- They have not had time to forget what they were taught, if the knowledge was delivered 60 seconds earlier students will remember more than 60 min earlier, and far more than if the knowledge was give 60 days earlier.

It is for this reason, it is better to not allow students to wait until the end of the module before they attempt to apply the knowledge and practical skills they learnt at the start of the module. Assess them as early as possible by means of a few questions at the end of each session. These can be discussion questions, multiple choice questions, short written answer questions. The weekly assessment for programming could be: can you solve the following problems… extend the program in the following ways? Etc.

Weekly exercises require a lot of thought and preparation. A concern raised by many is that we are over assessing the students. The student concern is: will I get any marks for this? This failure to see the importance of answering questions is quite concerning. It seems that Computing students, and men in particular, are less articulate than their humanities peers. Actually, that is not entirely true. In the 1990s there was a discussion of software development research papers in week 10, of a 12 week course on Advanced Programming Methodologies. At the time there were approximately a third of the class from Austria. The Austrian students were far more articulate than the UK students (there were no females in the Advanced Programming Methodologies course in the 1990s). There is a body of research that should be done to determine the difference in the Austrian culture versus the UK culture, and why the Austrians significantly outperformed their UK counterparts.

When teaching programming it is not unusual to ask questions, and not receive responses. Students need to be encouraged to engage in class discussions and reminded that if they don't know the answer to a question, or don't understand something then half of the class doesn't know or understand either.

The more programs students write, the better they are able to understand programming challenges. However, if there is no one around to guide them as they attempt to program they are less inclined to invest that time, because one can easily spend hours or even days trying to get something working, without guidance. The approach that an increasing number of students are taking is to search for a complete solution to the programming task. This is easier than trying to build it oneself and spending days getting nowhere. Once a student is demoralised it is difficult to get them back on track.

6.8 Incremental Programming, Incremental Learning

"**It's not that I'm so smart, it's just that I stay with problems longer**."—Albert Einstein (1879–1955)

If you want to try and solve anything, don't try and solve everything.

All programming problems are more easily solved if they are broken into smaller pieces. 40 years ago, Jackson Structured Programming structured programs around the structure of the data. To a small extent we have come full circle: When designing a mobile app, a series of storyboards, or screen designs, will identify all of the inputs and outputs for a screen, i.e. all the data the app has to deal with. Incremental development says start with a screen and generate the outputs required by utilising the user inputs and device sensor values required for each output. Do one at a time. This approach means that the screen designs are created first, then the transitions between screens, finally each output on each screen is implemented.

The result should be that one always has a program that is working, the only feature that may not work should be the one being worked on. The other features have not been started yet.

Programming is not unique in terms of production projects, in that products with numerous components have component tests and also dependencies. One is able to have a program that is incomplete but can undergo component tests. If one is building a car incrementally, the seats have to be complete before the steering wheel and the brakes must be working before a test drive. In a program there are data dependencies. All the data must be read before the calculations on all the data can be performed. Which parts of the app should one develop first? Agile methodologies say, work on the functions that one feels comfortable and confident to develop first. The delayed gratification personality says work on the hardest functions first. The entrepreneur, or consultant, works on the parts that will impress the client the most. Object Oriented suggests the one should work on the least connected objects, then there are bottom up approaches and top down approaches. There isn't a natural place to start.

This poses a problem for the beginner, where do I start? What should I do first? Just tell me what to do. This flexibility is a source of consternation initially because on the one hand programming appears to be a linear sequence of steps, in the event handling approach we don't know which event is going to happen first. It is not at all linear. On the other hand, component development of classes and functions, seems to be completely random.

As it happens, at the time of writing, the author's son completed a degree in Biochemistry and has been offered a job to become a software developer. The first three months are unpaid because he is receiving training. 12 weeks of training. 480 hours of development training is worth far more than 3 month's salary (although it doesn't feel that way to a 22 year old) and opens up a lifetime of development opportunities. It will be interesting to see what approach they take, and how much he is expected to know, and what he is expected to be able to do at the end of the three months.

6.9 Assessment Versus Learning

There is an unfortunate dilemma that students face. They don't have enough time to do all the exercises required to become fully conversant with the programming techniques that build on each other, each week. Many of them have lost sight of the reason they are at university: to learn. This is partly because the external pressure on UK universities to achieve and maintain TEF gold. The relationships between staff and students are becoming entirely formalised because there is a perception that academic staff aren't working hard enough. It is the student experience that suffers. If lecturers are overworked and stressed out they don't have time to be available for students.

Students also have time constraints, not entirely due to course expectations. The biggest impact appears to be in the form of part-time work. A few years ago, at the author's institution, a report came out from the Students Union indicating that the average student was working 16 hours per week. If the average student is working 16 h per week, and not every student works, there are students that are working 20, 25 h, even as much as full-time night shifts, in some cases. This is possible because the emphasis on student-centred learning means that the bulk of student effort is inevitably away from the classroom. They feel that this allows them to study at times when they are not at work or university.

It is financially poor students that are affected most by the need to work but it is not just poor students. This academic year, the author has three children at university, two at undergraduate level and one at post graduate level. As the parental income of both parents as educators is quite high, the children have reduced maintenance loans, and need to work. Their loans do not cover their accommodation costs. The Student Academic Experience Survey Report 2018 indicates that students at post-92 institutions have relatively high volumes of [course related] work outside the university (Neves and Hillman 2018).

Surveys have shown students are increasingly sleep deprived. There is a generation of teens growing up chronically sleep-deprived. According to a 2006 National Sleep Foundation poll, the organization's most recent survey of teen sleep, more than 87% of high school students in the United States get far less than the recommended eight to 10 hours, and the amount of time they sleep is decreasing—a serious threat to their health, safety and academic success (News Center 2015).

As a result of the changing culture and the high demands of the subject, students manage their time by doing the activities that will not only be assessed but actually accumulate marks in order to pass the module. It is for that reason students are increasingly asking: "What do I need to do to pass?" or "What do I need to do to get a 1st"? This selective learning is from those students that are struggling to manage the demands of modern computer science degrees with those of being able to keep their debt low.

It is therefore understandable that students don't have time to complete all the tasks we prepare for them. It's the main reason students want video recordings of lectures rather than documents to read. It takes effort, a lot of effort, to read unless it is one's preferred learning style. As students are sleep deprived reading sends many of them to sleep, so they find themselves drugged up with caffeine or reading the same paragraph 3 or 4 times between snoozes.

6.10 Summary

Computer programming languages along with computer technologies are unrecognisable in comparison to 40 years ago. The computer was the size of a classroom, the programming language BASIC had a small set of instructions with which to assemble programs. Today the computer in our pocket or handbag is able to do far more than several computers could do in several rooms. The programming languages and programming concepts are significantly more complicated, and if one doesn't know fully how to utilise an ArrayList for example, it is impossible to be able to pass an array as a parameter, or to process an array of elements polymorphically. Two or three hours of supervised practicals per week are nowhere near enough for beginners. The changing culture and the high demands of the course are the reason the drop-out rate for Computer Science is the highest of all degrees in the UK. It is time to conduct a review of the expectations of the challenges of teaching and learning to program.

References

En.wikipedia.org (2018) Ruby (programming language). [online] Available at: https://en.wikipedia.org/wiki/Ruby_(programming_language). Accessed 17 June 2018

Joy B (2002) Microsoft's blind spot. cnet.com. Retrieved 12 Jan 2010

Krishna R (2018) Intro to agile under 5 minutes. [online] YouTube. Available at: https://www.youtube.com/watch?v=N2hDKpgzdIE. Accessed 17 June 2018

Nevevs J, Hillman N (2018) The student academic experience survey report. [online] Higher Education Academy. Available at: https://www.heacademy.ac.uk/knowledge-hub/student-academic-experience-survey-report-2018. Accessed 21 June 2018

News Center (2015) Among teens, sleep deprivation an epidemic. [online] Available at: https://med.stanford.edu/news/all-news/2015/10/among-teens-sleep-deprivation-an-epidemic.html. Accessed 17 June 2018

Pine C (2009) Learn to program, 2nd edn. The Pragmatic Programmers, ISBN 978-1-93435-636-4

The Verge (2018) MIT brings Google App Inventor back from the dead as open-source project. [online] Available at: https://www.theverge.com/2012/1/21/2723656/google-app-inventor-open-souce-code-released-by-mit. Accessed 16 June 2018

Thunkable Thoughts—The How and Why You Would Make Your Own Beautiful App (2018a) Thunkable Thoughts—The How and Why You Would Make Your Own Beautiful App. [online] Available at: https://blog.thunkable.com. Accessed 16 June 2018

Thunkable Thoughts—The How and Why You Would Make Your Own Beautiful App (2018b) Mark Friedman, Cofounder of App Inventor, Joins Thunkable!. [online] Available at: https://blog.thunkable.com/mark-friedman-founder-of-app-inventor-joins-thunkable-ee3f130a2835. Accessed 17 June 2018

YouTube (2018) Introduction to scrum—7 minutes. [online] Available at: https://www.youtube.com/watch?v=9TycLR0TqFA. Accessed 17 June 2018

Chapter 7
Using Graphics to Inspire Failing Students

David Collins

Abstract The chapter summarises recent challenges faced by teachers of first year undergraduate programming and their causes. It proceeds to describe a pragmatic means of addressing such problems through the provision of parallel second programming module provision. The approach provides a means of motivating weaker students, introducing remedial prerequisite knowledge whilst avoiding the sacrifice of employability and other perceived goals of early *high-flyers* in the subject.

Keywords First programming languages · Motivating computer science students The processing language · Failure in programming · Learning edge momentum theory · Novice programmers

7.1 Introduction

A range of factors have made the teaching of computer programming in UK universities ever more problematic in recent decades. General grade inflation at A Level (UK university entrance qualifications) has meant that the skill and experience level of entrants is difficult to predict. Changes made to both the nature and assessment of such school qualifications have led to greater selectivity within subjects making it difficult to rely upon adequate coverage of the individual topics within a discipline—for example, trigonometry or statistics within mathematics. The expansion of the university system and the concomitant competition between institutions has paradoxically often led to the direct or indirect (through the university application clearing system) reduction in entry qualifications. The same competition has led to league tables which prioritise student satisfaction and 'added value' which has resulted in strong pressures to improve retention and grant higher award levels.

Whilst the above factors have been of influence, computer science courses continue to experience problems with widespread first-year retention difficulties and low pass rates. These ultimately manifest as low numbers of upper class degree awards

D. Collins (✉)
University of Keele, Keele, UK
e-mail: d.j.collins@keele.ac.uk

J. Carter et al. (eds.), *Higher Education Computer Science*,
https://doi.org/10.1007/978-3-319-98590-9_7

(the second worst subject across UK universities) and make CS the worst subject for students either leaving with no award or with a lower award than they had originally targeted (Woodfield 2014).

In attempts to deal with these problems, many universities have experimented with different programming languages and paradigms for their first-year courses. Although the jury is probably still out regarding the success of most such endeavours, it is worthy of note that a recent comprehensive survey of first year programming language adoption in the UK revealed that such experimentation (alternatives to Java and C) is more common in lower tariff (lower entry qualification requirement) universities where the need to consider alternatives is likely to be more acute (Murphy et al. 2017). In my institution, faced with very poor first year performance levels and consequent retention problems, we also sought a remedy. In common with similarly ranked universities, at least a third of our students were not significantly benefitting from their first-year programming experience based upon assessment of their end of year performance. Some studies suggest that the figure could be even higher, given a failure of first year university assessments to accurately measure student programming knowledge and ability (Ford and Venema 2010).

Naturally, we regarded this as a serious problem and engaged in an exhaustive review incorporating internal focus group interviews and extensive analysis of student performance statistics. The overall conclusion reached was that we were not attracting the calibre of students that would benefit from the reasonably mainstream pattern of programming skill development that was embedded in our CS curriculum. That much was evident, but there were considerable differences of opinion regarding an appropriate solution to the problem. The main obstacles were a certain degree of inertia, a desire to meet expectations of more able students and the ultimate goal of maximising employability skills.

Streaming students based upon their initial aptitude for programming was a possible route forward. However, this presented several challenges: could we identify such students?; could we resource parallel streams?; could we guarantee the efficacy of an alternative approach for failing students? What we were sure of was that by the end of the first programming module (Let's call it CS1), we could identify the failing group and what the future might hold for them without some form of intervention.

Through the focus group discussions, we identified three major barriers to learning present in the failing section of the cohort: (i) Expectations and motivation, (ii) Prior knowledge and experience and (iii) The nature of the programming learning process. I will consider each of these in turn.

(i) Expectations and Motivation
For many students, the CS1 course was not what they were expecting to encounter in their CS degree. Their knowledge of the subject was heavily influenced by their experiences in school and the portrayal of the subject in popular culture. At the time, school experiences were extremely varied but most likely to be focussed on IT solution packages such as spreadsheets and 'databases'. The adoption of the new computing curriculum within schools has changed this somewhat, but the delivery of the curriculum appears to be

extremely varied at present. In popular culture, computing could be an exciting subject capable of addressing (or causing) almost any societal problem. Hacking, social networks, artificial intelligence, robotics and gaming were common themes expressed by students. In contrast, CS1 exercises tended to be mathematically focussed and rather dry. Many studies, for example (Jenkins 2001), suggest that only a minority of CS students are *intrinsically* motivated by the subject of computer programming, and it is difficult to see how the use of such exemplars would be likely to improve upon this situation.

(ii) Prior Knowledge and Experience
There was a time when most universities could rely upon a common level of skills and experience in their UG cohort. For example, a GCE O Level qualification in mathematics guaranteed a fundamental knowledge of trigonometry and calculus—but attained to varying levels. This was no longer the case and many of the exemplars used in CS1 contained concepts and terminology which were not universally shared by the cohort. The provision of a discrete mathematics module had been abandoned some time earlier when it became apparent that failing students on the CS1 module also tended to fail this compensatory module.

(iii) The nature of the Programming Learning Process
There are now a wide range of theories to account for the unusual distribution of performance produced by year one computer programming modules. Of these, the author finds the *learning edge momentum* theory to be the most compelling (Robins 2010). In this theory, success in acquiring one concept makes learning other closely linked concepts easier, and failure makes it harder. The tightly integrated nature of the concepts comprising a programming language, drives students towards extreme outcomes. Coupled with this, early failure to master concepts is de-motivational and further exacerbates the problem. Failing students typically disengage early in a programming module reporting that they were unable to grasp the introductory concepts. Failing early often means failing completely.

We needed a prompt and pragmatic solution to the above problems. The essential approach was to agree an alternative to CS2 (Programming II—Data Structures and Algorithms) for the failing section of the CS1 cohort that would attempt to both meet the broad learning objectives of CS2 and address the three barriers to learning described above. CS2 at that stage occurred in the second semester of the first year of studies. Institutional constraints required that such a module (an alternative to CS2), would also be available to some *successful* CS1 students and that completion of the new module might allow access to CS3 (Advanced Programming) in year two. CS1, CS2 and CS3 were all taught using the Java programming language. I trust that the reader appreciates the low probability of a satisfactory outcome!

The choice of language for the new module was astoundingly straightforward. We needed to build upon previous experience in Java and allow for possible progression to subsequent Java based modules (CS3). The only language that met these criteria and also offered some other pedagogic possibilities was the Processing language developed at MIT by Chris Rea and Daniel Shiffman (Reas et al. 2007). Processing is a graphics oriented language designed for non-science students to create visual art. The language comes with a simple development environment (subsequently much improved but no more complex) and a large set of example programs (termed *sketches* in the Processing terminology). The language is essentially implemented as a Java library which entails that it could be imported into a conventional Java program within a conventional Java IDE. This afforded the possibility of returning to 'mainstream' Java toward the end of the new module without requiring any major conceptual leaps. By now, I am sure that many readers are familiar with the Processing language and environment so subsequent discussion will relate to how it was used to address our specific problems. An excellent exposition of the possibilities of the language is given in Daniel Shiffman's book (Shiffman 2016).

7.2 The Learning Objectives of Our CS1 and CS2 Modules

Before progressing further, I present the aims and learning objectives of our CS1 and CS2 programming modules:

CS1
Aim: To introduce computer programming concepts using a generic (non-context specific) computer language and to develop problem-solving skills in the framework of computer programming.
And Objectives:

- Demonstrate an understanding of the basic concepts of computer programming.
- Evaluate the suitability of computer language data and control structures to achieve basic problem-solving.
- Demonstrate an understanding of the basic software engineering principles.
- Show practical experience of those basic concepts.

CS2
Aim: To develop new programming skills as part of an exploration of several important data structures and algorithms used in Computer Science.
And Objectives:

- write a program that demonstrates important features of computer programming using an object-oriented programming language;
- describe, explain and evaluate the principles and operation of several data structures that are widely used in computer science;
- use a programming language to operate, test and evaluate one or more of the widely used computer science data structures;
- select class, data and control structures for program-based problem-solving.

In common with many other universities these aims and objectives are quite broad and allow implementation using a wide range of languages and, to a lesser extent, paradigms (object-oriented has a specific mention). The nature of *Processing* is such that it affords much the same opportunities as Java but the reduction in verbosity and bureaucracy coupled with graphics and animation support allows more creative problems to be addressed and expressed early in student's encounter with the language.

The challenge became one of designing a curriculum keeping the afore-mentioned three barriers to learning in mind. We sought to (i) attempt to provide a level playing field for students in terms of prior knowledge: (ii) motivate students by selection of exemplars that would stimulate interest: (iii) ensure that students mastered concepts incrementally, thus preventing early disengagement and failure. In the pages that follow we describe some of the main aspects of the module using tables to summarise how we addressed the identified barriers. The module progresses using exemplar programs, generally incomplete, as the context in which topics are taught and explored by students. During the practical sessions associated with each topic, completion of the associated individual exercises is confirmed and recorded by post-graduate demonstrators.

7.3 The Module

7.3.1 Drawing and Graphic Transformations

In order to introduce the *Processing* language and provide some revision of basic programming principles, the first topic provides an introduction to the procedural production of graphic images using primitive drawing shapes. Students are also introduced to graphical transforms and coordinate systems. Exercises include the reproduction of example images using repetitions of translations, rotations and scaling.

Additional knowledge and skills	CS programming concepts	Motivational material
Coordinate systems	Control structures	Students are asked to produce multiple flower and plant shapes with different morphologies and at different scales
Graphical transformations	Functions and decomposition	
Colour representation	Variables and scope	
Alpha transparency	Constants	
	Formal and actual arguments	

7.3.2 Animated Clock

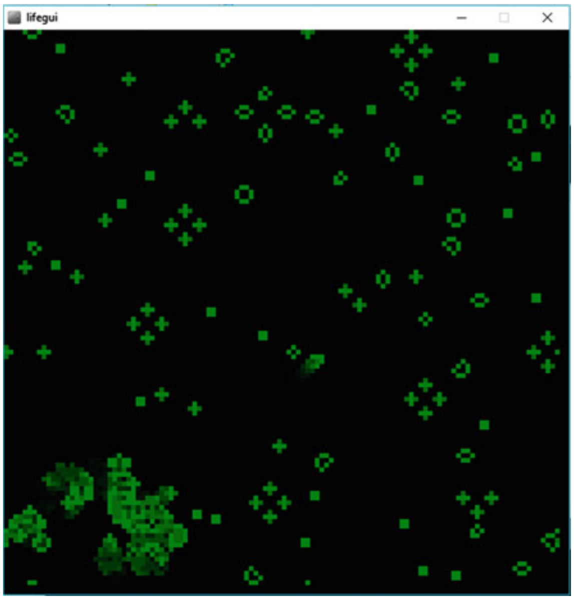
The second major exemplar involves the animation of a real-time analog clock. Students are initially provided with a program demonstrating radial motion which also interactively depicts Pythagoras's theorem, trigonometric relationships and the expression of angles in degrees and in radians (the majority of our students being ignorant of the latter upon entry to the module). Some basic initial exercises are provided which are intended to give students with the opportunity to explore these concepts. As a final exercise, students are required to EFFICIENTLY simulate the action of an analog clock.

Additional knowledge and Skills	CS programming concepts	Motivational material
Trigonometry	Library function calls	The production of a functional program with visual design selected by the student
Frame-based animation	Continuous versus discrete events	
Time zones and time representations		

7.3.3 *The Game of Life*

For this example, we provide students with a simple but visually captivating version of Conway's game of life (Gardner 1970). This provides a stimulating introduction to the concepts behind finite state machines. Exercises involve modification of survival and reproduction rules, changes to the grid size and resolution and modification of the 'neighbourhood' definition. The exercise provides the opportunity to master the use of 2/3 dimensional arrays and reasonably complex selection constructs. Students are expected to modify the code to permit different rules and neighbourhood definitions to be 'plugged in'.

Additional knowledge and skills	CS programming concepts	Motivational material
State machines	2/3 Dimensional arrays	Students are generally fascinated by the complex emergent behaviours that derive from simple rules
Turing, Von-Neumann and computing history	Cellular automata	
Modular arithmetic	Selection control constructs	
	Procedural abstraction	
	Tracing, reading and debugging code	



Conway's game of life implemented in processing with fade-out effects for dying cells

7.3.4 Card Games

Students are provided with graphics for presentation of playing cards and are asked to design algorithms for dealing and shuffling. We present some sorting algorithms and Durstenfeld's algorithm for shuffling (Durstenfeld 1964). This provides an opportunity to introduce the topic of complexity and we discuss the performance of shuffling algorithms in this context. The card decks provide an excellent medium for exploring stacks and queues and associated ADTs.

Additional knowledge and skills	CS programming concepts	Motivational material
Numeric distributions	Sorting and shuffling algorithms	Card games require a range of algorithms and data types. The relevance and purpose is immediately apparent to students. Students are invited to devise their own shuffling algorithms initially which provides a natural vehicle for exploring complexity
Statistical concepts	Algorithmic and computational complexity	
Randomness	Arraylists	
	Random number implementations	
	Abstract data types, stacks and queues	

7.3.5 Sprite Based Animation

Animation in Processing is essentially frame-based. In this section of the module we introduce the use of sprite sequences on scrolling backgrounds. Sprites may be controlled via the keyboard or with the use of a gaming input device.

Additional knowledge and skills	CS programming concepts	Motivational material
Seamless textures	File handling	Weaker students in particular are delighted to be able to produce a program that approaches the quality of a published (but retro) game
Easing in animation	Object arraylists	
Frame buffering	Asset management	
Multiple animation timelines		

7.3.6 Lunar Lander

Students are provided with a background lunar image and a transparent GIF representing the landing craft. They are also provided with an example program depicting objects falling under the influence of gravity that uses a skeleton implementation of a Vector class. They are then expected to complete the Lunar Lander Program using the keyboard to deliver 2 dimensional thrust to counter gravity and gracefully land the craft at a randomly selected location. This requires implementation of new methods for the Vector class.

Additional knowledge and skills	CS programming concepts	Motivational material
Forces and Newtonian physics	Classes and objects	A playable game with realistic control
Vectors	Keyboard polling/state detection	
Discretization	A vector ADT and its implementation	

7.3.7 Image Processing

Students receive basic lectures on Image formats, manipulation of pixel information and the construction of filters using convolution matrices.

They are provided with a scene of crime image in which a poorly illuminated suspect is seen next to a car with a number plate which is indiscernible. The final task requires the construction and application of filters (incrementally) that will clean-up the image to the extent that the suspect is identifiable and the number plate readable.

Additional knowledge and skills	CS programming concepts	Motivational material
Matrices	Convolution filters	The real-world challenge present in the exercise task is highly motivational to students and encourages them to master a topic that often produces despair when presented in a more traditional image processing context
Image representation and compression schemes	Matrix manipulation	
Bitmap versus scalar graphics		
Normalisation		

7.3.8 Collision Detection

Students are presented with a complete program (based upon the game of Asteroids) that visually represents coarse and fine collision detection and their affect upon maximum animation frame-rates. Students are required to assess the relative performance of coarse and fine approaches and derive a strategy for optimising the compromise between accuracy and performance.

Additional knowledge and skills	CS programming concepts	Motivational material
Further trigonometry	Collision detection algorithms	Students tend to be familiar with this problem and are inspired to find strategies that produce optimal game-play
	Heuristic lgorithms	

7.3.9 Boids

This is by far the most complex programming example provided to students—an object-oriented version of Craig Reynold's (Reynolds 1987) Boid simulation written in the *Processing* language. Lectures introduce the concepts involved and describe the overall design, the main classes and broad implementation details.

The exercises require students to read and follow the logic of this relatively complex piece of code. For example, they are instructed to alter the weighting applied to the three vectors that determine the speed and acceleration of Boids at each discrete decision cycle (which corresponds to the frame-rate). In order to provide more realistic behaviour, they are encouraged to experiment with allocation of weights using a Gaussian distribution. They are expected to modify the definition of neighbourhoods and, as a final exercise, to consider means of improving performance of the algorithm to provide real-time flocking utilising more sophisticated Boid animations. Specifically, they are asked to consider a means of avoiding the need to check the position of each Boid in each cycle in order to determine whether it might be considered a neighbour for purposes of calculation of the required alignment vector.

Additional knowledge and skills	CS programming concepts	Motivational material
Numeric distributions	OO design	Students are fascinated by Boid behaviour and in particular with the complexities of behaviour that can result from the application of such simple rules
	Complexity	
	Code optimisation	
	Algorithmic optimisation	

7.3.10 Recursion

Recursion can be a difficult subject for students and benefits greatly from a graphical treatment. Students are provided with a range of examples including binary trees, Sierpinski triangles, Koch snowflakes and Mandelbrot sets. A range of exercises provide experience in using linear, binary and tail recursion. The final (non-trivial) exercise requires that students add realism to the construction of a binary tree.

Additional knowledge and skills	CS programming concepts	Motivational material
Fractals	Recursion and recursion types	Recursion is explored through graphics—binary trees, Sierpinski triangles and fractals. Students are highly motivated by the task of introducing realism into the construction of a graphical binary tree
Complex numbers	Recursion performance	
	Binary trees	
	Binary search trees	

7.3.11 The Relationship Between Processing and Java (and Python, Javascript and Android)

The final topic of the module provides students with the opportunity to use *Processing* as a Java library within the NetBeans IDE. Lectures describe the language implementation and the manner in which it can be incorporated in a conventional

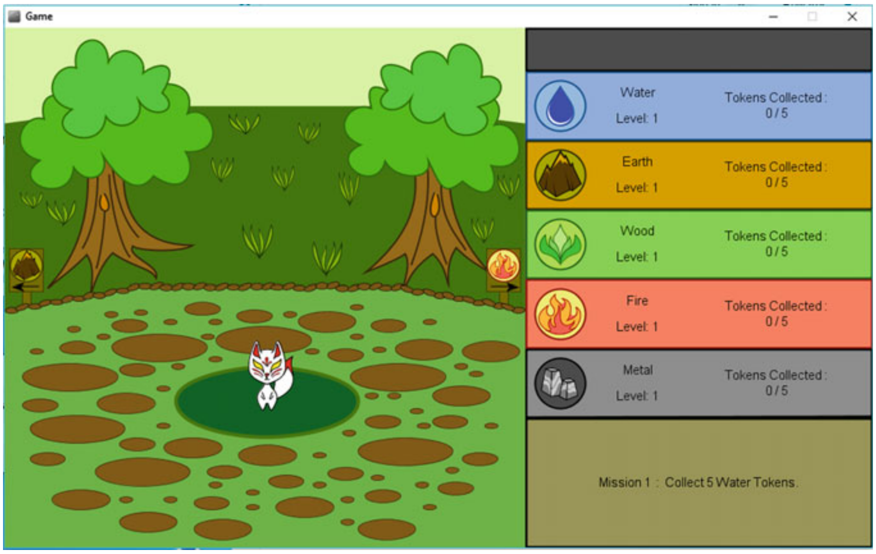
Java application. Processing implementations for Python, Javascript and Android are also described and demonstrated.

Additional knowledge and skills	CS programming concepts	Motivational material
Comparative programming languages	Programming frameworks	At this stage many students have discovered the relationship between java and processing. This topic formalises that understanding
	APIs	
	Wrapper classes	

The above list of exemplar-led topics is not exhaustive and we do change the examples used with each delivery of the module. The module has now been running for 8 years and we recognise that some of the examples are becoming stale and would benefit from new materials. We are also aware that the materials may have a gender bias that we would like to address without falling prey to obvious stereotypes.

7.4 Module Assessment

The module is assessed by examination and coursework. During the fourth week of the module students are required to submit a proposal for their assignment work (with the benefit of a list of future topics that will be covered). We point them at the more complex Processing exemplars provided with the environment and at the OpenProcessing.org website (www.openprocessing.org) for inspiration. The latter provides a virtual gallery through which students may showcase their work to the world. Each student is briefly interviewed regarding their choice of subject and is only allowed to proceed if the lecturer considers the project to be suitably challenging and capable of expressing the module's learning objectives. The progress of each student on their assignment work is monitored throughout the remaining practical sessions.



A game produced by a first year student

7.5 Discussion

There are many variables that determine the success and failure of a module. The introduction of this module was not a controlled experiment and it would not be appropriate to make comparisons or draw conclusions as if this were the case. However, we can broadly point to some of the benefits and problems.

Retention problems were dramatically reduced and weaker students were undoubtedly inspired by the new approach. Much of the student work produced in coursework assignments far exceeded our expectations and continues to do so to the present day. Stronger students had sufficient new materials as to be challenged by the approach but there were some criticisms such as the absence of access modifiers and the simplification of type casting methods in the Processing language. In practice, some of the more capable students possibly felt patronised by the introduction of what they perceived to be a simpler programming environment. Indeed, many were quick to import the *Processing* core into a Net Beans or Eclipse Java environment.

Processing has a pre-processor that wraps the processing code inside a Java class and allows access to functions without having to declare and instantiate the classes to which they belong. Combined with rich graphical features this allows the student to focus on algorithmic content. In reality, the language *is* a slightly more accessible form of Java coupled with a simple IDE and a feature rich Graphical API. Changes to the intrinsic Java are generally akin to being able to introduce **println()** rather than explain **System.out.println()**. The latter requires suspension of full understanding in weaker students and partially explains why the language creates a *learning-edge* problem with failure to understand one concept *at a specific time* leading to lack of confidence and unpreparedness to tackle subsequent related concepts.

The module has undoubtedly led many failing students from CS1 to develop an *intrinsic* motivation to further develop their programming skills. Students exhibit pride in their work and a desire to demonstrate to, and seek the opinions of their peers. The current module lecturer has persuaded a major European company to sponsor an annual award for the best student assignment work. The module's introduction was a pragmatic solution to a problem and saved us from having to meddle with the now long standing and conventional approach to teaching programming with the Java language that is embodied in our CS1-CS2-CS3 programming strand. Since that time, many UK universities have adopted *Processing* as a first programming language. Interestingly, Murphy et al's recent survey of language use concluded:

> The results in this first UK survey indicate a dominance of Java at a time when universities are still generally teaching students who are new to programming (and computer science), despite the fact that Python is perceived, by the same respondents, to be both easier to teach as well as to learn. (Murphy et al. 2017)

Of course, the likely explanation for the above is that the respondents have goals other than '*teachability*' in mind when selecting a first language — employability would undoubtedly feature highly.

From my perspective as a teacher of programming, the module undoubtedly extended my 'shelf life'. Instead of trying to avoid the weary gaze of yawning students as they tested stack implementations full of meaningless characters and numbers, I was able to witness genuine excitement in students as they retrieved cards from a discard pile or added visual leaf nodes to a binary tree. Early on, it was apparent that some students from CS1 had not really understood two-dimensional arrays. Conway's life provided both motivation and visual feedback that allowed weaker students to both perceive their relevance and master the topic. Students with little mathematical or scientific background picked up an understanding of trigonometry, Newton's laws, parabolic curves and frequency distributions pretty much as a side-effect of wanting to make a game more playable or realistic. Above all, students became far more enthusiastic and attended lectures and practical sessions with relish rather than a sense of guilt or obligation.

Even better was the fact that no compromises were being made in the teaching process. I was still teaching Java to attain the same learning objectives in the context of Computer Science and Software Engineering as were adopted by our CS2 module. Effectively I was provided with an additional teaching tool with which I could create teaching materials that were stimulating yet understandable. The overheads involved in using Processing rather than pure Java were negligible and ultimately led students to understand how they might design an effective application framework. I am indebted to the Processing team for their generous contribution to computing pedagogy.

References

Durstenfeld R (1964) Algorithm 235: random permutation. Commun ACM 7(7):420

Ford M, Venema S (2010) Assessing the success of an introductory programming course. J Info Technol Educ 9:133–145

Gardner M (1970) Mathematical games—the fantastic combinations of John Conway's new solitaire game "life". Sci Am 223:120–123. ISBN 0-89454-001-7

Jenkins T (2001) The motivation of students of programming

Murphy E, Crick T Davenport JH (2017) An analysis of introductory programming courses at UK Universities. Art Sci Eng Program 1(2), Article 18

Reas C, Fry B, Maeda J (2007) Processing: a programming handbook for visual designers and artists (1st edn). The MIT Press, p 736. ISBN 0-262-18262-9

Reynolds CW (1987) Flocks, herds, and schools: a distributed behavioral model. Comput Graphics 21(4):25–34

Robins A (2010) Learning edge momentum: a new account of outcomes in CS1. Comput Sci Educ 20(1):37–71. https://doi.org/10.1080/08993401003612167

Shiffman D (2016) Learning processing—a beginner's guide to programming images, animation, and interaction. Morgan Kaufmann. ISBN 978-0-12-394443-6

Woodfield R (2014) Undergraduate retention and attainment across the disciplines [Internet].Higher Education Academy, New York. Available from www.heacademy.ac.uk/node/10293. Accessed 17 June 2018

Chapter 8
Best Practices for Teaching Information Systems Modelling

Steve Wade

Abstract The subject of Information Systems Modelling (ISM) grew out of computer science to fill a gap created by the difficulties programmers had in understanding and solving user problems. The intention behind ISM is to facilitate communication between technologists (many of whom have no idea of the complexity of organisations) and end-users and their managers (many of whom are unable to translate their problems into feasible demands upon technology). "Best practices" in information system development might therefore be considered to be those practices which contribute in some way to improving communication between these two parties. The work described here is primarily focussed on documenting practices that address the issues associated with the seamless transition from a requirements model seamlessly to a technology based system that satisfies those requirements. This has involved reflection on lessons learned during thirty years' experience of teaching Information Systems Modelling in the context of higher education.

Keywords Information systems modelling · Requirements model
Communication · Pattern language

8.1 Background

The subject of Information Systems Modelling (ISM) grew out of computer science to fill a gap created by the difficulties programmers had in understanding and solving user problems. The intention behind ISM is to facilitate communication between technologists (many of whom have no idea of the complexity of organisations) and end-users and their managers (many of whom are unable to translate their problems into feasible demands upon technology). "Best practices" in information system development might therefore be considered to be those practices which contribute in some way to improving communication between these two parties.

S. Wade (✉)
Department of Computer Science, University of Huddersfield, Huddersfield, UK
e-mail: s.j.wade@hud.ac.uk

J. Carter et al. (eds.), *Higher Education Computer Science*,
https://doi.org/10.1007/978-3-319-98590-9_8

Before considering examples of best practice and how these may be taught we will consider the possible consequences of poor communication between technologists and end-users. Focussing on these will enable us to be clearer about the specific benefits that the practices we consider might offer. We will focus on two possible consequences of poor communication.

1. The end-user holds limited views about what they need and fails to realise that alternative superior solutions (which they have been unable to imagine) are possible.
2. The developers hold misguided opinions about what the users need because this has not been clearly explained to them.

There is good evidence to suggest that many information system failures are a result of one or both of these leading to a discrepancy between the system "as required" and the system "as delivered". If we are to avoid this discrepancy we need to deploy development methods that:

1. Provide mechanisms to make sense of and understand the details of human activities that an information system is developed to support. This involves developing some kind of information requirements model.
2. Provide a seamless transition between developing the information requirements model and the design and implementation of a technology-based system to satisfy the requirements captured in the model.

The work described here is primarily focussed on documenting practices that address the above requirements. This has involved reflection on lessons learned during thirty years' experience of teaching Information Systems Modelling in the context of higher education. Some lessons were learned in the eighties when we still made use of plastic flowchart stencils, pencils and plenty of printer paper. More came in the nineties with the rise of powerful CASE tools supporting development methods that required the designer to develop and maintain increasingly elaborate, internally-consistent collections of diagrams. The most significant lessons have been learned more recently as the evolution of powerful programming environments has encouraged a more informal approach to modelling.

In addition to learning lessons from the past it is important to prepare students for the future. Most software systems are embedded in social systems and the resulting sociotechnical systems' boundaries and interactions can be hard to identify. For example, social networks, travel booking and online shopping applications have had far-reaching effects on the way people form relationships, how they travel and what they buy. Becker et al. (2016) have argued that software's critical role in society demands a paradigm shift in the software engineering mind-set. We argue that our students need to be prepared for this shift which will focus on architectural issues that can be addressed through Information Systems Modelling.

8.2 Introduction

This chapter primarily draws on a number of years' experience teaching an Information Systems Modelling module jointly to postgraduate students on an MSc Information Systems Management and an MSc Advanced Computer Science. The module concerns the application of the Unified Modelling Language (UML) throughout the development lifecycle from requirements analysis to implementation. All the students arrive on the module with some background in modelling but those on the MSc Information Systems Management tend to think in terms of business models whereas those on MSc Advanced Computer Science tend to view modelling as high-level programming. This presents the challenge of moving students into a deeper understanding from different starting points and with different preconceptions about the nature of the subject.

In the process of delivering this module we have engaged in the following activities each of which will be described in more detail in the remaining sections of this chapter:

- The design of a pattern language to organise best practice in the application of systems development techniques. In developing the pattern language we were mindful of the need to encourage maximum student ownership of the development process. The patterns could not therefore comprise simple lists of instructions to be followed slavishly.
- The development of teaching materials to document the pattern language.
- Running the module. Observing the progress of the module week-by-week in a number of ways including a range of on-going student feedback mechanisms.

The remaining sections of this chapter reflect on what we have learned from engagement in these activities.

8.3 The Pattern Language

Patterns have been widely used in information systems design over the last ten years. A pattern in this context is a generic solution to a recurring problem expressed in a literary form. The approach has its roots in architecture specifically the work of Alexander (1979). In ISM patterns have been used to ease communication problems and the thinking behind complex design (Gamma et al. 1995). Patterns are usually described by templates which specify the style and structure of a pattern description. Typically the template will include sections for a description of the problem to be addressed, the forces acting to create the problem, a generic solution, a specific example of how this solution might be applied and a discussion of the benefits the solution should provide.

The following example relates to a common (and hopefully familiar) problem in domain modelling where students represent a many-to-many relationship between two objects when the relationship would be better represented by a third object.

Problem

How to model the relationship between two classes that have a many-to-many association with each other.

Forces

- Many-to-many relationships occur often in the real world.
- It can be difficult to implement many-to-many associations in some object oriented programming languages.
- Many-to-many relationships have no direct implementation in relational database systems.
- A many-to-many relationship is usually complicated enough to warrant the addition of an extra class.

Solution

Transform the many-to-many association between two classes into a trio of classes by creating an intermediary class with two one-to-many relationships. The name of the intermediary class should describe the type of relationship being captured.

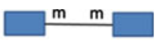
Example

A many-to-many relationship between Student and Module is reconstructed as two one-to-many relationships. One between Student and Work Record and the other between Module and Work Record.

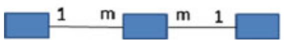
Discussion

We can now store details of module grades for each student as attributes of the new "work record" class.

Summary

If you find this: 

consider replacing it with this:



The idea is that patterns such as this can be used to guide students away from common problems and into good practice. We have specified many more patterns related to commonly occurring problems. Initially we used patterns drawn from the publications of Ambler (1998, 1999) and Evitts (2000). We spent some time re-working and shaping the documentation for these patterns to give coherence to the collection. A key feature of the collection is that relationships are drawn between patterns. When a number of patterns are related to each other in this way we describe the result as a "pattern language". We are therefore trying to develop a pattern language to support the teaching of information systems modelling. Further examples of specific patterns and their relationships will be provided later in this chapter.

8.4 A Framework for the Pattern Language

Most modern courses in Information Systems Modelling are based on the Unified Modelling Language (UML). The UML provides a suite of diagrams that help us to visualise the design of a system. It is published by the International Organization for Standardization (ISO) as an approved ISO standard. Having decided to use the UML we needed to decide which development method to follow. The most popular model-centric approach currently in use is the Unified Software Development Process (USDP) but this is both large and complex. Instead of following the USDP we devised our own simplified method based on our earlier research into the design of a multi-method framework (Salahat and Wade 2009). In that research we proposed a framework for bringing together principles from object oriented approaches to designing software systems and the "soft systems" approach to analysing social systems as part of Business Analysis (Checkland 1999). This approach requires a few words of explanation.

The teaching of Information Systems Modelling tends to focus on issues related to 'hard' systems design. Hard systems are the technical systems that are produced during a development project. Each hard system will be embedded in its social context. This context can be seen as a "soft" densely interconnected system of human activities. It can be argued that hard systems should not be analysed in isolation from the soft systems within which they reside. In analysing the present system or designing a new system, there is the need to consider both the hard system that will be the product of the development, and the soft system within which it will be used. This is challenging because the workings of the soft system are often difficult to understand and the needs of the organisation can be difficult to predict. The UML covers all aspects of hard systems design but has much less to say about the soft system. In contrast Soft Systems Methodology (SSM) focusses on the soft approach.

In addition to augmenting UML modelling with techniques from SSM we also wanted to introduce students to Persona Analysis as a means of developing empathy for users.

Defining personas is an established practice in user-interface design. Blomkvist (2002) describes personas as follows:

> A persona is a model of a user that focuses on the individual's goals when using an artefact. The model has a specific purpose as a tool for software and product design. The persona model resembles classical user profiles, but with some important distinctions. It is an archetypical representation of real or potential users. It's not a description of a real, single user or an average user. The persona represents patterns of users' behaviour, goals and motives, compiled in a fictional description of a single individual. It also contains made-up personal details, in order to make the persona more 'tangible and alive' for the development team. (Blomkvist 2002)

We ask students to develop personas that include a fictional name and life story, a picture, and a 'tag line'—a phrase, supposedly written by the persona, that represents the character of the persona as related to the development project. The case studies that we use in teaching relate to our own department. Accordingly we encourage students to develop a persona for each of the following: A Student of Computer

Science, a Student of Information Systems, a Course Administrator, a Lecturer and a representative of an organisation providing industrial placement opportunities.

Although the primary application of personas has been in the context of user-interface design, we have found spending time developing them focusses attention on requirements in a concrete way. Rather than referring to users in an abstract form, students refer to personas by name. So our student of Computer Science becomes Jo Smith who has a first degree in Software Engineering a great deal of confidence in his programming ability but lacks confidence in writing essays and reports—his tag line is "I would rather write code than prose". In contrast our student of Information Systems becomes Sue Rachel who has a first degree in Law, is fascinated by the impact of technology on society but lacks confidence in her ability to write code. Her tag line is: "technology will never replace great people but it can help ordinary people to achieve great things".

Following the basic structure of this framework we developed patterns and teaching materials based around the following topics:

- How to use Persona Analysis to help the developer focus on the needs of the user.
- How to use Soft Systems Methodology to learn about a problem situation.
- How to extract Use Cases from the soft systems models.
- How to develop sequence diagrams related to each Use Case.
- How to develop a domain model from the collection of sequence diagrams.
- How to convert the domain model into a class diagram and database design.
- How to implement the class diagram as an object oriented software system using the, "naked objects", implementation pattern.

We have used the step-by-step approach implied by these questions as the basis for a course structure and an assignment specification. We have adopted a "scaffolded" approach to teaching (Larkin 2001). which involves working through a number of exercises following the structure implied above then asking students to apply the techniques to related case studies for their coursework. The case studies used for assessment were based around the needs of an academic department like our own.

It is beyond the scope of this chapter to discuss each of these topics in detail but for those unfamiliar with the techniques, the following examples are intended to give an idea of what deliverables are produced during each step of the method.

The example used here relates to the decision to introduce a Peer Tutoring System into an academic department to provide extra help to students on a programming module. The idea being that students who are confident in their programming skills would run support sessions for their less confident colleagues. We initially asked students to develop a simple persona for the type of people who would use this system. As explained above a persona is a fictional character that typically has a name, a picture, behavioural traits, common tasks, and a goal that describes the problem the persona wants to see solved or the benefit the character wants to achieve. Personas are not considered in the UML so we devised our own simple template based on the facets listed above. In filling out this template the developer is encouraged to visualise the user in a concrete, tangible way. So the personas mentioned above,

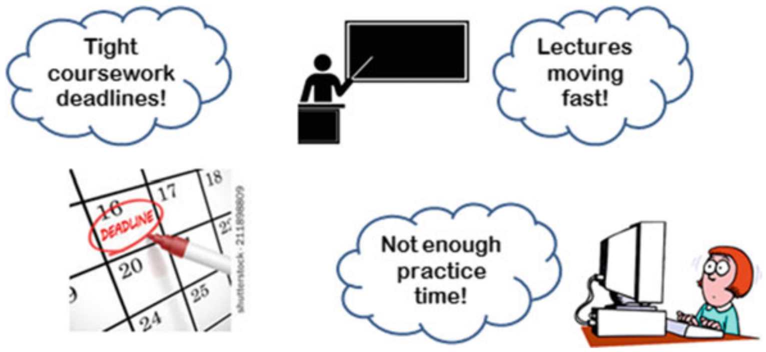


Fig. 8.1 Initial rich picture raising issues for the peer tutoring system

named Jo Smith and Sue Rachel, may be involved in the peer tutor system as a peer tutor and peer tutee respectively.

Moving on from Persona Analysis we next ask students to conduct a simple business analysis—with no attention to the design of software. Again this is somewhat outside the scope of the UML so, as explained above, we have introduced techniques from Soft Systems Methodology. The first of these is a rich picture. The figure below shows a rich picture that we might use to get things started (Fig. 8.1).

We have found rich pictures to be a good way to encourage discussion about the problem situation without focus on any proposed solution. The discussion leads to the development of a root definition: This is defined in SSM as a succinct description of the Human Activity System (HAS) that is required. It is possible to develop multiple root definitions each offering a different perspective on what is required. The following is a possible root definition for the peer tutoring system:

> A system owned by the course that provides programming skills support to students using volunteers with programming experience from the student cohort. The quality of this support will be monitored by academic staff.

Once we have developed a root definition, or set of definitions, we move to developing more detailed activity models. These are called Conceptual Models in SSM and we would develop one for each root definition. The following example, originally presented in Wade et al. (2012), includes activities that might be supported by software and others that will be enacted by humans without the assistance of software (Fig. 8.2).

In developing this type of diagram we encourage discussion about the social system we are supporting. In this simple case discussion might focus on the following questions:

- Will weaker students attend these sessions or will they be primarily attractive to students who are already competent programmers but want further opportunities to develop their skills?

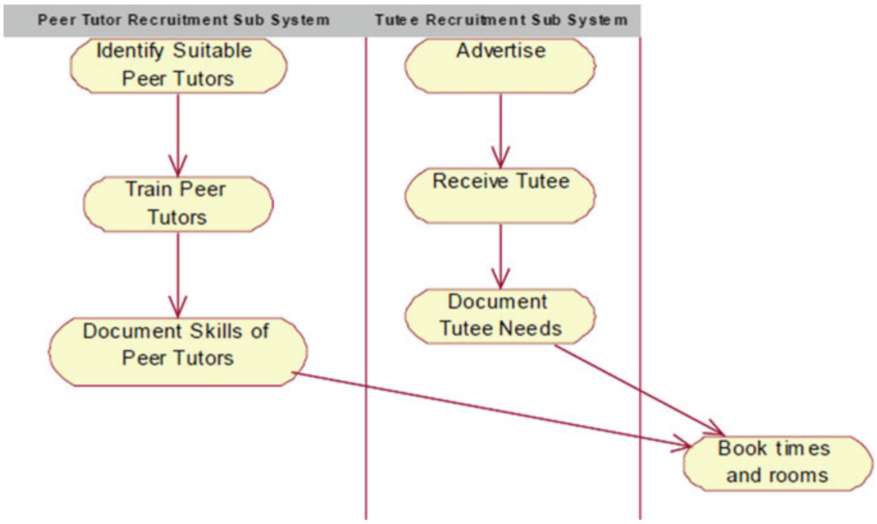

Peer Tutor Recruitment Sub System
Tutee Recruitment Sub System
Identify Suitable Peer Tutors
Train Peer Tutors
Document Skills of Peer Tutors
Advertise
Receive Tutee
Document Tutee Needs
Book times and rooms

Fig. 8.2 Activity diagram

- Should some (weaker) students be required to attend the sessions? If so how do we identify people in this position?
- Should we pay peer tutors? Are their alternatives to rewarding them with money?

This discussion might lead us to develop additional activity models. For example we might consider the need to monitor attendance at the sessions and develop the following for an "attendance monitoring" system (Fig. 8.3).

In developing these models we have not concerned ourselves with the question of how software might be able to assist the people involved. This would be the role of a Use Case Model. Use Cases are part of the UML and represent activities that require software support. If we were to develop a Use Case Diagram from our activity models we would be making the transition from business analysis to software design. The following Use Case diagram could be derived from the conceptual model above (Fig. 8.4).

If we focus on the "Print Class List" Use Case we might prototype a simple user interface like this (Fig. 8.5).

Behind this interface our software system might be composed of collaborating objects. The following high-level sequence diagram depicts the role that a number of objects might have, "behind the scenes" (Fig. 8.6).

We found that students found the transition from Use Case Models to Sequence Diagrams difficult so we provided the following pattern and discussed it in class.

Problem

It is hard to develop sequence diagrams from the Use Case Module. What can I do to make this transition easier?

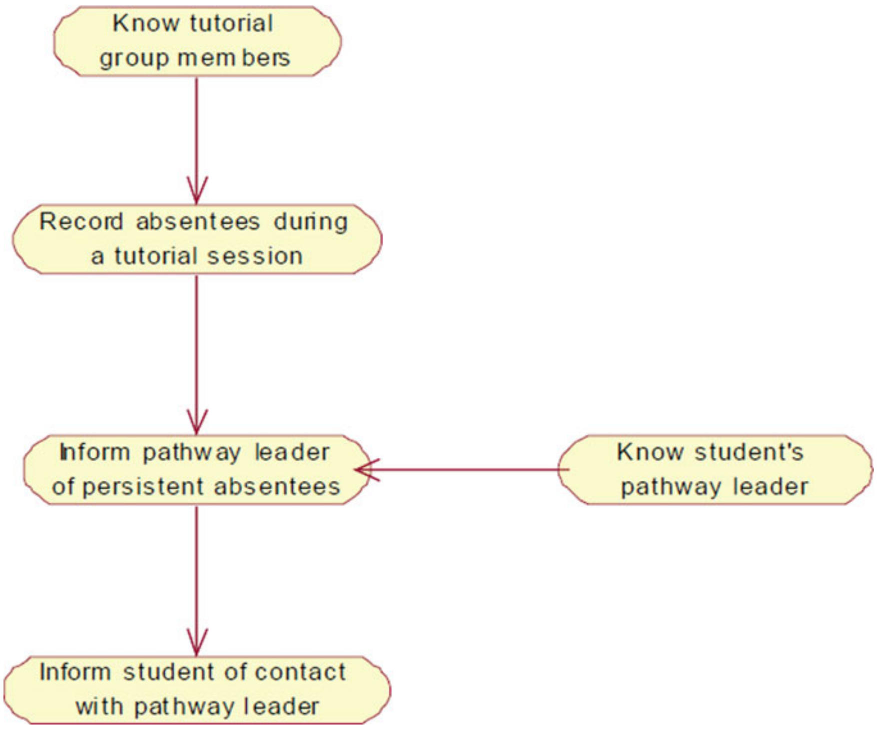

Know tutorial group mem bers
Record absentees during a tutorial session
Inform pathway leader of persistent absentees
Know student's pathway leader
Inform student of contact with pathway leader

Fig. 8.3 Activity diagram for attendance monitoring. Taken from Wade et al. (2012)

Forces

A high level Use Case Diagram (such as the one presented above) is fine for a, "mile high", view of the computer systems behaviour. For many stakeholders, such as sponsors and managers, this will be enough. As designers however we need to open these up and define them in detail. We know what the system presents to the various users (or actors), we need to define in fine detail the, "how", of that interaction; until we have done this we cannot begin to develop a sequence diagram. There is no prescription in UML regarding what detailed information should be recorded about a use case

Solution

Document the detailed logic of a Use Case as a series of steps. Where appropriate each step should include reference to one or more domain classes and identify the role that this class should play in the implementation of the Use Case. We can use this description as the basis for developing an initial sequence diagram. The way in which this is done is specified in the "Develop Sequence Diagram from Primary Path" pattern.

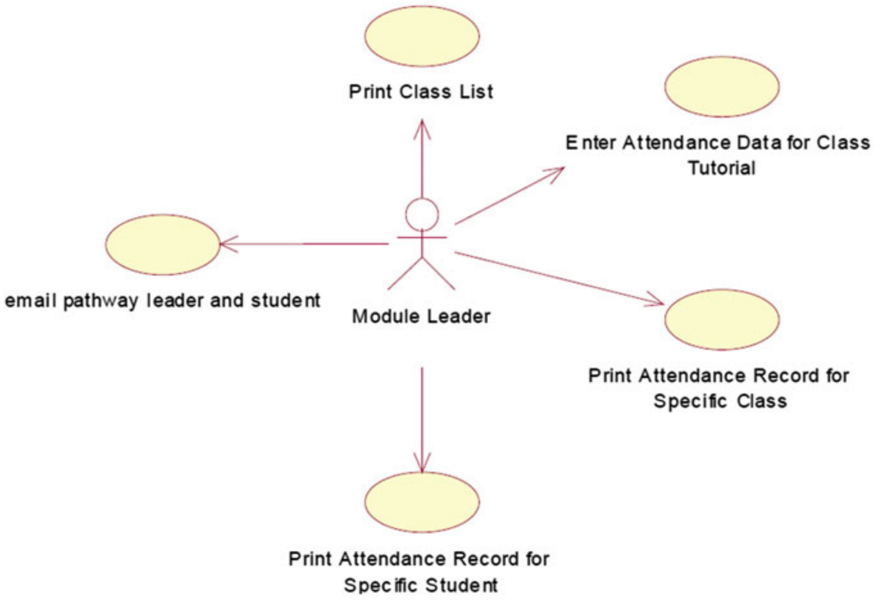
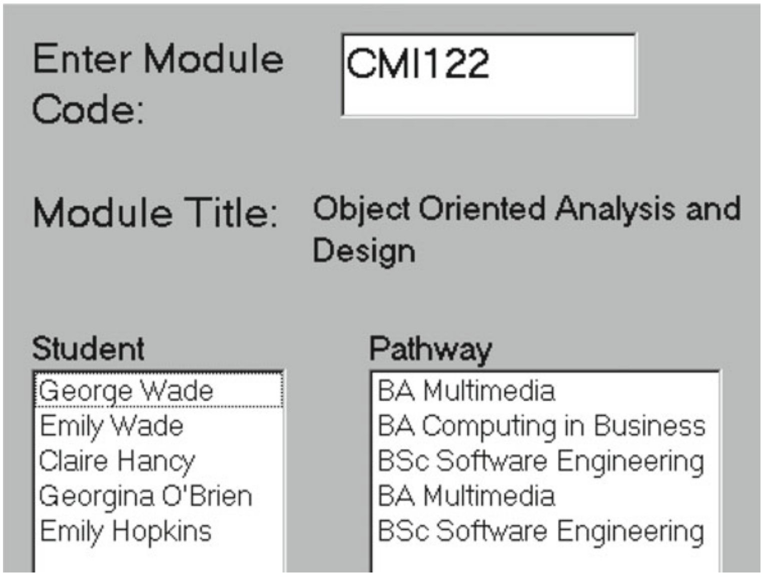

Print Class List
Enter Attendance Data for Class Tutorial
email pathway leader and student
Module Leader
Print Attendance Record for Specific Class
Print Attendance Record for Specific Student

Fig. 8.4 A use case model. Taken from Wade et al. (2012)



Enter Module Code:	CMI122
Module Title:	Object Oriented Analysis and Design

Student	Pathway
George Wade	BA Multimedia
Emily Wade	BA Computing in Business
Claire Hancy	BSc Software Engineering
Georgina O'Brien	BA Multimedia
Emily Hopkins	BSc Software Engineering

Fig. 8.5 Screenshot for a use case. Taken from Wade et al. (2012)

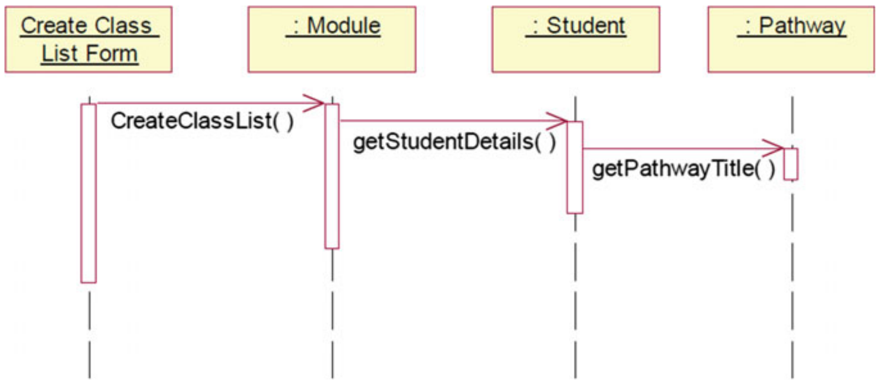


Fig. 8.6 A sequence diagram. Taken from Wade et al. (2012)

Example

This Use Case is concerned with enrolling an existing student in a peer tutor session for which she is eligible.

Key Steps:

1. The use case begins when a student wants to enrol in a peer-tutor session.
2. The student inputs her name and student number into the system.
3. The system verifies the student is eligible to enrol in classes at the university.
4. The system displays the list of available peer tutor sessions.
5. The student indicates the session in which she wishes to enrol.
6. The system checks that the student is enrolled on the appropriate module to join the session.
7. The system asks the student to confirm that she wants to enrol in the session.
8. The student indicates she wants to enrol in the session.
9. The system creates an enrolment the student in the session.

These steps can then be mapped to messages passed between objects in a sequence diagram like the one presented above. It may be that each line in the Use Case description would map to a single message passed between objects. We would develop a sequence diagram for every use case then develop a domain model consistent with all of these sequence diagrams. A domain model derived from this single sequence diagram might look like this (Fig. 8.7).

This domain model can be used as the basis for an object oriented software system design and a relational database structure. We have developed patterns to translate the domain model to a physical database design principally by adding primary and foreign keys to create relationships between tables. A separate pattern discusses ways of mapping inheritance relationships to relational structures.

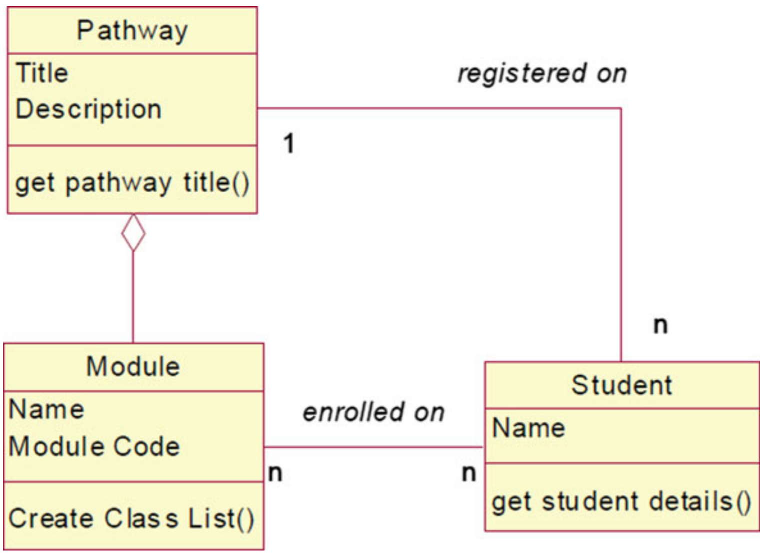


Fig. 8.7 A domain model. Taken from Wade et al. (2012)

In developing User Interfaces we encourage students to use the Naked Objects architectural pattern Pawson (2002) to generate a graphic user interface directly from the domain model. The pattern uses reflection to automatically generate an initial user interface. Typically this interface will present a series of windows containing icons representing each of the domain classes. A class can then be accessed by double-clicking on its icon to reveal its operations. In the above example I can select, "Module," select a specific module then right-click on that module's "Create Class List" operation to see a list of students currently enrolled on the module. Other functionality can be achieved by dragging and dropping; so for example if I wish to enrol a student onto a module I can drag the icon representing that student on to the icon representing the module thereby creating the relationship between them. An advantage of applying this pattern is that the resulting relationship between the domain model and what appears in the interface is very direct. A change in the domain model (e.g. the addition of an operation on, "Student", named "Get Coursework Marks") feeds through into the code and then directly into the user interface. From a teaching perspective this helps to reinforce the idea that modelling is both about representing the real world and designing software.

An important part of our teaching has been to identify specific issues that cause difficulties for students then provide specific, detailed guidance of how to ameliorate these difficulties. More specifically we have considered the difficulty of conducting a thorough business analysis before transitioning to design, the transition from a Use Case view to a behind-the-scenes view of the software architecture, the transition from design to implementation with specific attention being paid to the design of an interface that reflects the structure of the software. We have discussed these transitions in terms of patterns that capture good practice and the relationships between these patterns.

We present the patterns in a manner based around the metaphor of different development "rooms". The first room is concerned with developing user personas it is adjacent to a room for developing an "analysis" model based on Soft Systems Methodology. This room contains patterns for developing a range of soft systems models including rich pictures, root definitions and conceptual models. An adjoining room contains patterns for making the transition from analysis to design by translating the analysis model into a Use Case Model with carefully structured documentation of each use case. The next room contains patterns for moving into physical design and then into code. These include: "Develop a sequence diagram showing how domain classes may co-operate in the implementation of a use case". This will involve ensuring that the detailed steps in our use case description relate to the messages being passed on the sequence diagram. A related pattern will explain how to assign operations to classes that map to messages on the sequence diagram.

As mentioned above when a number of patterns are related to each other in this way we describe the result as a "pattern language". We are therefore trying to develop a pattern language to support information systems modelling. We would argue that patterns are particularly suited to this purpose. They are descriptive, not prescriptive (unlike most detailed development methods). They capture expertise in an open-ended format that lends itself to a "hypertextual" structure of resources with links between related patterns that can be explored without forcing a specific sequence of activities. The patterns can also be used as the basis for developing assignment specifications. We will say more on this latter topic in the closing sections of this chapter.

8.5 Running the Module

In light of the above discussion we have been able to propose the following guidelines for developing a module in this area:

1. Design a portfolio-based assessment that can be completed in instalments each instalment being aligned to patterns used in teaching. For each pattern we specify deliverables that can be represented in an assessment grid. In the case of the patterns described above, one instalment could be a Use Case model that is consistent with earlier conceptual models and which is described in steps that map to messages in a sequence diagram. The patterns then become part of the explanation of what is required and are clearly linked to the feedback grid.
2. Provide formative in-class surveys that encourage students to reflect on their understanding of key patterns. In the first example above can they provide an example of a many to many relationship between two classes that could be better represented by a third class? Can they see how the proposed solution would help? Can they apply this learning to the coursework case studies?

3. Encourage students to discuss the individual patterns and how they may be applied to case studies before they complete the in-class surveys or work on the assignment. Students should be encouraged to identify new patterns and fit them into the pattern language or to improve the documentation of existing patterns.
4. Collect data on a regular basis by inspecting samples of student coursework on a week-by-week basis and in-class surveys. Use the feedback to inform improvements to pattern descriptions and the identification of new patterns.

These four guidelines work together to steer the students through the assessment process by frequently monitoring their progress. Hopefully this will lead to continuous improvement in the clarity of the coursework specification and the teaching materials.

With respect to Step 4 above we employ a variety of different ways to collect evaluative information. These include: a pre-course questionnaire that we distribute before teaching begins this is intended to establish the background knowledge and expectations of our students; a series of anonymous in-class surveys to test students understanding and self-confidence in applying the patterns under discussion; short reflective essays were made part of the coursework portfolio in which students were asked to give their personal opinions about the usefulness of the pattern-based approach and focus group discussions were held in class.

In addition to the above, during marking, we carried out an analysis of the most common mistakes made by students in their coursework. We discovered a number of recurring mistakes and made changes to the pattern language to discourage these. A number of examples are given below.

- Inconsistencies between diagrams. For example operations appearing in the sequence diagram that are not present in the class diagram.
- Failure to use domain-specific vocabulary as presented in the case study materials. In the above example we referred to "Pathway" where others may have used the term "Course". It is important that the language used in the models can be found in the user documentation.
- Operations that have ambiguous or misleading names. We have seen operations with names like "Update all" or "Reconsider" these names are almost meaningless to anyone but the original programmer.
- Database concepts (e.g. primary and foreign key dependencies) used in the domain model. The domain model is meant to be an abstract representation that might be used in the design of object oriented software or an ontology it is not a physical database design.
- Operations not supported by attributes or relationships. Some operations depend on the availability of connections to other classes or of data properties that must be included in the domain model.
- A lack of consistency between the SSM models and the Use Case Model. For example Use Cases that cannot be inferred from the activities in conceptual models.

We continue to work on developing patterns that will steer students away from these types of mistake. We plan to present these via a website based on our "rooms" metaphor with hyperlinks between related patterns. We would argue that working in this way has encouraged us and our students to consider important aspects of information systems design that are often overlooked in courses that teach Information Systems Modelling. A few of these are listed below:

- We encourage students to adopt multiple system viewpoints from different personas. The acknowledgment and exploration of these viewpoints emphasises the important point that typically most systems have more than one purpose and many unexpected consequences;
- We are encouraging our students to consider the total problem situation and to be aware of the need to ensure that all, not just the most obvious, significant issues are addressed;
- Our approach is strongly goal-oriented. The central focus on Use Cases ensures that derived requirements are justified with respect to stated goals;
- All the techniques documented in our pattern language are well established and well tried. We haven't presented patterns for anything that has not, in one form or another, proved useful to developers. We hope the manner of presentation is more effective than user manuals or detailed methodology documentation but the intention is not to present anything new but to organise and document the distilled wisdom of the many talented software engineers and systems analysts who have worked in this area over the years.

8.6 Conclusion

This paper has described our approach to teaching Information Systems Modelling over a number of years. We have described the approach as being built around the scaffold of a multi-method systems development framework which we have documented in the form of a pattern language. This basic structure has been tried and tested through a number of feedback mechanisms (including in-class surveys, focus group discussions and reflective essays) and a concordant assessment strategy. The results obtained through these feedback mechanisms have encouraged us to continuously refine our teaching materials and assessment strategies—we believe these changes have all been improvements.

References

Alexander C (1979) The timeless way of building. Oxford University Press, New York
Ambler SW (1998) Software process patterns. Cambridge University Press
Ambler SW (1999) More software process patterns. Cambridge University Press
Becker C, Betz S, Chitchyan R, Duboc L, Easterbrook SM, Penzenstadler B, Venters CC (2016) Requirements: the key to sustainability. IEEE Softw 33(1):56–65

Blomkvist S (2002) The user as a personality: using personas as a tool for design. Position paper for the workshop 'Theoretical perspectives in Human Computer Interaction' at the Interaction and Presentation Laboratory of the Royal Institute of Technology, Sweden, September 3, 2002. Available at http://www.nada.kth.se/~tessy/Blomkvist.pdf

Checkland P (1999) Soft systems methodology: a 30-year retrospective. Wiley, Chichester

Evitts P (2000) A UML pattern language. Macmillan Technical, Indianapolis, Ind

Gamma E, Helm R, Johnson R, Vlissides J (1995) Design patterns: elements of reusable object-oriented software. Addison Wesley, Reading, MA

Larkin MJ (2001) Providing support for student independence through scaffolded instruction. Teaching Exceptional Children 34(1):30–34

Pawson R (2002) Naked objects. IEEE Softw 19(4):81–83

Salahat M, Wade S (2009) A systems thinking approach to domain-driven design. In the proceeding of UKAIS2009 conference. Oxford University, Oxford, UK

Wade S, Salahat M, Wilson D (2012) A Scaffolded Approach to Teaching Information Systems Design. Innovation Teaching Learn Info Comput Sci 11(1):56–70

Chapter 9
Promoting Design Thinking Through Knowledge Maps: A Case Study in Computer Games Design and Development Education

Carlo Fabricatore and Maria Ximena López

Abstract Modern computing is pervaded by human-centric technologies which potentiate people's capabilities to address complex problems and needs in contextualised and meaningful ways. Creating such technologies requires thinking approaches that overcome the limitations of traditional paradigms that focus on specific aspects of human-computer interaction in well-defined problem contexts. Design thinking is an ill-defined problem-solving strategy which addresses this need through a systematic and iterative process that integrates exploration, ideation and testing of possible solutions based on the participation of stakeholders, and the investigation and accommodation of their often-conflicting needs. Developing design thinking in students is key to face the challenges of an ever-changing and increasingly complex world, and it is therefore crucial to have approaches and tools that can support educational endeavours aimed at this. In this chapter we describe the use of knowledge maps to promote design thinking for game design and development students. Knowledge maps are a variant of hierarchical concept maps created by domain experts to support learners' knowledge construction processes. Game design knowledge maps were conceived to integrate and structure multidisciplinary knowledge regarding game systems, players, player engagement principles, and design and testing processes. Their structure was planned so that students could explore imparted knowledge iteratively and incrementally, driven by a human-centric focus. The evidence collected from students so far indicates that knowledge maps integrate large amounts of information in an easily accessible structure which fosters students' design thinking processes. The maps seem to support students in connecting themes and ideas, and guide them through

C. Fabricatore (✉) · M. X. López
University of Huddersfield, Huddersfield, UK
e-mail: C.Fabricatore@hud.ac.uk

M. X. López
e-mail: m.x.lopezcampino@hud.ac.uk

J. Carter et al. (eds.), *Higher Education Computer Science*,
https://doi.org/10.1007/978-3-319-98590-9_9

the whole design thinking process. This suggests that properly structured imparted knowledge can be effective in helping students to learn "how" to think, not just "what" to think.

Keywords Design thinking · Concept maps · Game design · Education Problem solving

9.1 Introduction: The Need for Human-Centred Design Thinking in Computer Science Education

Nowadays computing devices permeate and shape many aspects of our lives. They influence the way we work, socialize, learn, play, and engage in the many other types of activities that define our daily living, and through which our identities are defined. Nowadays more than ever before computational devices and computer programs are conceived because of people and for people, to address their complex needs in complex contexts.

The defining power of computational technologies is due to how these are designed to mediate human activity, putting abstract technological features at the service of meaningful human needs and wishes (Jaimes et al. 2007). Accordingly, modern perspectives on computing emphasise the importance for Computer Science—henceforth referred to as CS—to fully embrace human-centric approaches (e.g. Taylor 2000; Jaimes et al. 2007; Grudin 2012). This implies that CS should be regarded as a scientific endeavour aimed at studying in equal measure what computational artefacts "can do" in abstract, how they can mediate human life, and how they can be designed and implemented for this purpose. By extension, CS education should aim at enabling students to design, develop and evaluate computational artefacts embracing human-centric perspectives.

User-centredness is not new to CS. In fact, focus on computer-human interaction has been key to drive the evolution of CS and CS education since the 1970s (Grudin 2012). "Human-centrism", however, entails more than attention to the process of interaction between humans and computers. Adopting a human-centric approach requires focussing on the whole human experience, rather than purely on abstract features of technological artefacts (Taylor 2000; Giacomin 2014). Computational technologies should be treated as mediators of contextualised and purposeful human activity, focussing on how they can augment the users' capabilities to interact with their environments in desirable ways, and how technological potentialities are perceived by the "user" (cf. Bannon 1991; Buchanan 1992; Giacomin 2014). For this, the user, the technological artefacts, and the context and purpose of technology uses should be treated in an integrative way, accounting for the possible effects that technology may have on users and their environments (Bannon 1991; Taylor 2000; Giacomin 2014). This requires a systematic way of thinking, suitable to tackle ill-defined problems through iterative processes of creation of solutions driven by user-centred evaluations: human-centric design thinking. By extension, promoting design thinking should be a key priority in education focussed on human-centred creation of technology (cf. Cross 1982; Oxman 1999; Dym et al. 2005; Dunne and Martin 2006; Fabricatore and López 2015, in press).

What are the characteristics of design thinking? What can be done to promote its development through formal education? To address these questions, in this book chapter we will first discuss the distinctive properties that characterize design thinking as a systematic reasoning strategy. Then, drawing from key research on design thinking education we will outline key strategies to foster design thinking through formal education. Accordingly, we will analyse a key challenge that frequently hampers design thinking education: the provision of knowledge to students. We will then introduce our response to this challenge: design knowledge maps. These are a variant of concept maps that we have created in order to provide structured knowledge suitable to scaffold the development of design thinking in the domain of computer game design and development education. Through a case study we will outline the structure of a game design knowledge map, discussing its underpinning rationale and exemplifying its implementation. We will then report preliminary impacts investigated so far. We will conclude this chapter with some reflections on the need to increase efforts to promote design thinking within CS education, and the possibility to use design thinking knowledge maps outside the domain of game design and development.

9.2 The Engine of Human-Centredness: Design Thinking

Design thinking can be regarded as an applied reasoning strategy that integrates investigation, strategic planning, construction and testing of systems to address open-ended, ill-defined problems involving multiple interacting stakeholders (Buchanan 1992; Taylor 2000; Dorst 2011). The core purpose of design thinking is to formulate solutions aimed at generating impacts valuable for all the stakeholders involved in the problem situation (Buchanan 1992; Dunne and Martin 2006). Unlike well-defined problem-solving approaches, design thinking addresses problem situations in which there exist no optimal or definitive solution, and there is no possibility to gain full and stable knowledge of requirements and working principles underpinning problem scenarios. Stakeholders may be (and often are) influenced by conflicting needs and constraints. These needs and constraints are likely mutable, not fully knowable nor predictable, as are the environmental conditions that define the contexts in which stakeholders exist and interact. Consequently, design thinking explores and models problem situations through different perspectives, ideating, implementing and testing alternative solutions based upon incomplete and uncertain information. This contrasts with well-defined problem-solving strategies that aim at formulating “optimal” and “definitive” solutions based on full and stable understanding of the problem situation. This distinguishes design thinking as an iterative problem-solving process which systematically integrates abductive, deductive and inductive reasoning

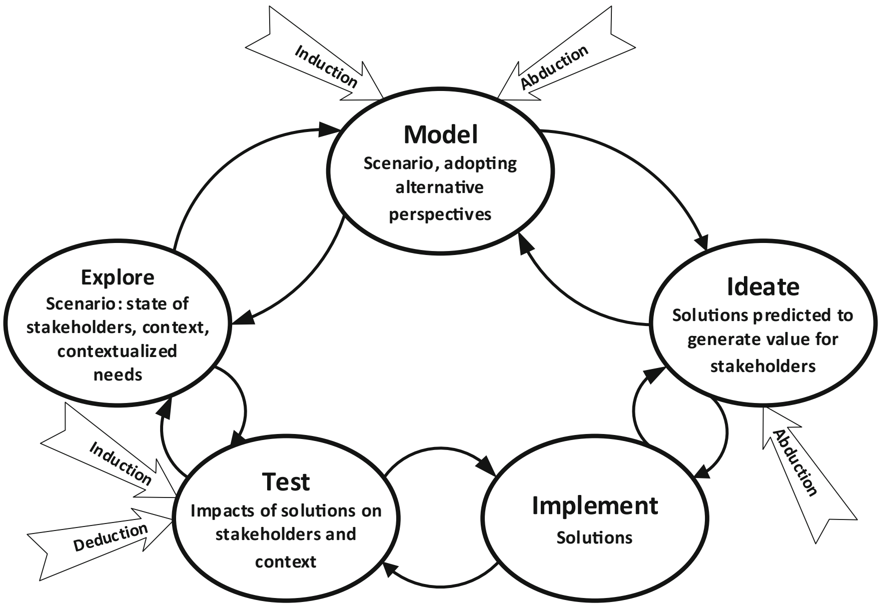

Induction
Abduction
Model
Scenario, adopting alternative perspectives
Explore
Scenario: state of stakeholders, context, contextualized needs
Ideate
Solutions predicted to generate value for stakeholders
Induction
Deduction
Test
Impacts of solutions on stakeholders and context
Implement
Solutions
Abduction

Fig. 9.1 The design thinking process

in a cycle of progressive approximation to an "acceptable", "good enough" solution (Fig. 9.1). The structure and rationale of the design thinking process, and its focus on stakeholders and their context characterize design thinking as a human-centric problem-solving approach suitable to tackle complex problem situations (Buchanan 1992). Furthermore, the way design thinking integrates generation of novel ideas with their practical implementation and testing makes of design thinking a process of innovation (Dunne and Martin 2006).

9.3 Design Thinking as a Systematic Problem-Solving Approach

As previously mentioned, a key and distinctive feature of design thinking is how it systematically integrates abduction, induction and deduction in problem-solving. A problem can be generally regarded as discrepancy between a current state of matters and a desirable goal state in a given situation. A problem situation can be viewed as a system of interacting elements. Accordingly, a problem is defined by information regarding: the type and state of elements involved in the system; the contextual conditions in which they interact; the working principles that regulate their interactions and state changes; and the desirable goal to attain.

When a problem situation is stable, it can be reliably investigated and defined through direct exploration, using deductive and inductive reasoning (Dorst 2011). First of all, stability implies that the state of system elements can be known through exploration of the problem situation at any point in time. Then, if the working principles are known, given a present state of the problem situation it is possible to apply deductive reasoning to predict future states: knowing "what" is involved in a situation and "how" it functions allows predicting "outcomes" of the functioning (Dorst 2011). If the working principles are not known, inductive reasoning can be applied to hypothesize them based on observation of patterns in system state changes: knowing "what" is involved in the situation and observing the "outcomes" of its functioning allows hypothesizing "why" it may have changed and, by extension, "how" it functions (Dorst 2011). These mechanics allow to apply a linear problem-solving approach to define how to modify a system in a desirable way. If current state, working principles and goal state are defined, abductive reasoning allows hypothesizing which aspect of the system should be modified so that the goal state will be attained: knowing "how" things work and knowing the desired outcome of their working allows defining "what" things should be modified to achieve such outcome (Dorst 2011). Deductive reasoning can then be used to confirm or refute hypotheses, through implementing the planned changes—i.e. the "solution"—and testing their effects against the predicted outcomes. For example, imagine that one wanted to improve the strength of tennis players' swings (goal state). To achieve this, knowing that the weight of a racket affects the strength of tennis strokes according to well-defined mathematical models (working principles) one might design a lighter racket, using for the very first time a revolutionary material: graphene (Already used for racquets!?! Too bad…).

In the case of open-ended problem scenarios, the situation may be significantly more complex, unstable and ill-defined. In these cases, the elements involved in the problem situation and the working principles that regulate them may be unknowable, changing and uncontrollable to some relevant degree. This requires using abductive reasoning for the iterative formulation and testing of parallel hypotheses on working principles and solutions, through a systematic process which is distinctive of design thinking (Fig. 9.1). Given a desirable value (goal) to attain, the designer frames the problem situation adopting alternative perspectives and formulating different and likely complex models (Buchanan 1992; Dorst 2011). Each model includes inductively inferred and abductively hypothesized working principles, and a thesis that associates these with the desired value to attain: "IF we look at the problem situation from this viewpoint, and adopt the working principle associated with that position, THEN we will create the value we are striving for" (Dorst 2011, p. 525). A model is then selected and, based on the hypothesized working principles, a solution to attain the desired value is abductively formulated and implemented. Deductive reasoning is then used to test the hypothesized solution and, by extension, the underpinning working principles. If the solution generates the predicted value, its validity and the underpinning working principles are inductively generalized (Dunne and Martin 2006). Otherwise, abductive hypotheses regarding solution and underpinning working principles may be reformulated and tested through a new iteration of the entire process, or the framing may be changed all entirely (Dunne and Martin 2006; Dorst 2011).

To exemplify the process, let us consider an oversimplified fictional case. Pretend that the value pursued was enhancing wellbeing in the context of rural jungle communities in Nowhereland. After exploring the problem situation one might hypothesize that people living in those communities enjoy socializing (working principle). Therefore, one might theorize that if socialization opportunities were enhanced, then wellbeing would increase (abductive thesis, relating hypothesised working principles and desired value). To complete the framing, one might have observed that most people have old mobile phones whose usage is limited to voice calls and SMS. At this point one might have sufficient information to hypothesize that a good solution could be an innovative product: JungleTxTClub®, the first ever SMS-based social networking platform interfaced with larger, non-SMS-based platforms (Already done?!? Oh no…) Assume that, embracing the “social networking framing” of the problem situation, one developed and tested this solution for a month, only to discover that the average use decreased drastically day after day. This apparent failure would then lead to reviewing the working hypotheses and the solution. Through this process, one might interview a sample of users, discovering that the idea of digital social networking is appreciated. However, users found typing on the mobile keyboard cumbersome. One might then infer inductively that there is a problem with the user interface, reviewing the solution accordingly, and initiating a new cycle of the design thinking process.

9.4 Design Thinking as Human-Centric, Social Process

Stakeholders are the fulcrum of the design thinking process (Buchanan 1992; Taylor 2000; Dunne and Martin 2006). Design thinking tackles scenarios which are “problematic” first and foremost because they involve stakeholders who have latent or manifest needs that require addressing. Stakeholders do not consider themselves mere “users” of products and services: they are human beings concerned with the quality of their lives, their activities, and the impacts that products and services can have on these (Giacomin 2014). This is why the primary focus of the design thinking process is not “the solution”, but rather “the need in context” and “the value” to be generated to address contextualised needs. By extension, design thinking is not only concerned with the direct impacts of a “solution”. Driven by social and ethical considerations, it accounts for broader implications that “solutions” may have on stakeholders and their contexts (Buchanan 1992; Taylor 2000).

Design thinking also acknowledges that stakeholders are diverse. Their perspectives and circumstances may be different. As a consequence, their needs may conflict. A given scenario will likely not represent the same “problem” for all stakeholders, and what could be perceived as a “solution” by some, might represent a “problem” for others (Buchanan 1992; Taylor 2000; Dorst 2011). It is in order to address diversity that design thinkers frame problem scenarios through different perspectives, exploring alternative frames in search of models that can accommodate as much as possible all the involved stakeholders.

Stakeholders' needs and circumstances may be mutable and not fully manifest: not everything may be knowable through observation, and what has been observed may change. To address this, design thinking integrates observation, dialogue and inquiry in order to involve stakeholders throughout the problem-solving process (Buchanan 1992; Dunne and Martin 2006). Background information on stakeholders is obviously a key starting point to frame the problem situation tackled. Observation serves to infer relevant behavioural patterns of the stakeholders, salient aspects of their circumstances, and impacts of implemented solutions. Inquiry and dialogue are used to gather information directly and intentionally provided by stakeholders, which closely represents their perceptions of the problem situation. Stakeholders are not just a source of information. Through dialogue stakeholders are involved in joint decision-making and evaluation processes, as in the case of the definition of the value to be attained, the selection of framing perspectives, and the evaluation of impacts attained (Buchanan 1992; Dunne and Martin 2006). This promotes the accommodation and integration of conflictive views in case of conflict.

9.5 Design Thinking as an Innovation Process

Design thinking cannot be fully outlined without highlighting its intimate connection with creativity and innovation. Exploring alternative perspectives and accommodating conflicting needs requires thinking outside existing and evident alternatives, creating and testing new options, and embracing constraints as challenges, rather than barriers and causes for compromise. This makes of creativity an intrinsic element of design thinking. The exploration of non-yet-existing possibilities is not purely conceptual. As possibilities are iteratively developed through cycles of design and testing, design thinking integrates conceptual analysis and practical synthesis in a dialectical process. Thus, design thinking fully mirrors what innovation is: a process of generation and implementation of novel and useful ideas in response to open-ended problems and opportunities (Fabricatore and López 2013). Design thinking as a whole IS a process of innovation.

In sum, design thinking can be regarded as a human-centric, systematic and creative process ideal to address meaningful human needs through the generation of innovative technological solutions. This is of paramount importance nowadays, when people need solutions to improve their lives in increasingly complex contexts, industries compete to generate innovative solutions, and formal education strives to equip students to tackle the challenges of our complex world (cf. Dunne and Martin 2006; Dym et al. 2005; Fabricatore and López 2015; Koh et al. 2015). How can design thinking be fostered within formal education?

9.6 Fostering Design Thinking Through Formal Education: Implications and Challenges

Since the 1990s there has been a growing interest in studying the cognitive aspects of design, and educational research on design thinking has developed accordingly (Cross 1982; Oxman 1999, 2004). It has been acknowledged that learning design thinking means enhancing real-world problem-solving capabilities (cf. Cross 1982; Oxman 1999; Dym et al. 2005; Dunne and Martin 2006). Researchers have consequently investigated strategies to promote the development of the thinking skills and attitudes involved in design thinking, and foster the conscious assimilation of the reasoning patterns underpinning the design thinking process (cf. Oxman 1999, 2004; Dym et al. 2005; Fabricatore and López 2014, in press). A significant consensus has been reached regarding key features that educational strategies should incorporate for these purposes. Drawing from our past research (Fabricatore and López 2014, in press) and echoing key trends in design thinking educational research (cf. Cross 1982; Oxman 1999, 2004; Dym et al. 2005; Dunne and Martin 2006; Koh et al. 2015), we suggest that these features can be summarised as follows:

i. Focus on project-based learning activities, aimed at addressing open-ended, ill-defined problems involving multiple interacting systems, and underpinned by multiple knowledge domains.
ii. Contextualization of problem situation mirroring real-wold scenarios, involving: diverse stakeholders; properties of contextual conditions and stakeholders which are neither fully knowable nor fully predictable; social as much as technological implications for the involved stakeholders.
iii. Iterative and incremental organisation of project work, integrating multiple cycles of design, implementation and evaluation of solutions.
iv. Promotion of students' collaboration and self-organisation of project work, involving: team-based project work motivated by the intrinsic complexity of the problem tackled; roles and responsibilities organised by students, based on iterative exploration of problem scenarios and provisional outcomes of project work.
v. Promotion of direct and indirect exploration of and interaction with key stakeholders involved in the problem scenarios tackled.
vi. Provision of adaptive pedagogical support, involving: upfront provision of explicit core knowledge, firstly useful to promote shared understandings, and later on freely accessible to learners as a scaffold; adaptive provision of explicit supplementary knowledge, ad hoc, depending on project progression and learner profile/learning style; enabling, non-prescriptive tutor feedback, promoting the adoption of alternative perspectives to frame problems, and the prediction and critical analysis of implications of possible solutions; mediation of team dynamics, to promote mutual understanding and self-organisation.
vii. Tailorization of project constraints to maximize their enabling value, ensuring that: project-specific constraints promote the exploration of alternative perspectives and the formulation of different solutions to the problem tackled; admin-

istrative constraints (e.g. academic policies and regulations, etc.) do not hinder students' self-organization and possibilities to embrace varied approaches to project work.

All the above features are underpinned by a pivotal element: the knowledge imparted to learners. Its contents and structure should reflect the comprehensive type of knowledge involved in design thinking, and support knowledge building processes similar to those carried out by expert designers (Cross 1982; Oxman 1999; Fabricatore and López in press).

Designers develop solutions through constructing knowledge regarding a problem situation which can be generally viewed as a system of interacting elements (cf. Cross 1982; Fabricatore and López in press). As they explore problem situations, designers observe, identify, classify and infer concepts and their connections, building meanings in order to hypothesize working principles that govern the problem system tackled, and ideate possibilities to influence it in desirable ways (cf. Buchanan 1992; Oxman 1999; Dorst 2011). This can be assimilated to a purposeful process of construction of knowledge based on meaning-making: meaningful learning (Novak 2010). The knowledge acquired and constructed through the design thinking process concerns the "what" of design, as much as the "how" and the "why" (cf. Cross 1982; Oxman 1999, 2004; Dym et al. 2005). This knowledge can be regarded as a system of concepts and propositions (Novak 2010). It may be factual, inductively inferred, abductively hypothesized, and deductively corroborated through the design thinking process. Regarding the "what", knowledge involved in design thinking relates to attributes and relationships of key elements of the problem system investigated—including stakeholders and other relevant contextual elements—and the solution being designed. Regarding the "how", knowledge regards techniques and tools useful to explore and modify the system being tackled, and strategies to organize and apply techniques and tools. The "why" concerns working principles regulating interactions between system components. This knowledge allows designers to reflect on the suitability of tools, techniques and strategies in relation to the effects that they might have on the systems tackled. It serves to predict or critically evaluate direct effects and broader implications of possible solution approaches.

The contents of the knowledge imparted to students should be integrative, comprehensive and multi-disciplinary, covering the "what", "how" and "why" of design (Cross 1982; Oxman 2004; Dym et al. 2005; Dunne and Martin 2006). For this, the structure of knowledge is as important as its contents (Oxman 1999, 2004; Novak and Cañas 2008). The knowledge imparted should be organised connecting information about the "what", "how" and "why" in scaffold structures which students can assimilate and use as a basis to build their own conceptual structures (Oxman 1999). The knowledge imparted should also be mainly explicit and objective, rather than subjectively defined and verbally conveyed by tutors (Oxman 2004). Non-explicit knowledge is only temporarily embodied in the words of the tutor, and is dependent on the tutor's experience, personality, cognitive style, and on the student's immediate interpretation. Instead, explicit and objective knowledge provides to all students equal possibilities to access and share the same knowledge whenever needed (Novak and Cañas 2008; Fabricatore and López in press).

The definition and provision of structured knowledge thus represents a pivotal challenge in design thinking education. We tackled this challenge in the domain of computer game design and development education through creating a bespoke knowledge representation tool, which we labelled game design knowledge map.

9.7 Game Design Knowledge Map: Structure and Rationale

A design knowledge map is a variant of concept map. Concept maps are visual representations of structured knowledge, articulated as graphs in which concepts are usually represented as nodes, and relationships between concepts as links between nodes (Novak and Cañas 2008). Concept maps are suitable to explicitly represent knowledge structures, and can help students to learn how to learn (Novak 2010). For this, concept mapping can be used in two ways: representation of evolving student knowledge, and mapping of expert knowledge provided to students (Novak and Cañas 2008; Novak 2010). In the first case, concept maps created and updated by learners serve to make their knowledge explicit as its construction process unfolds. This allows students to reflect on their knowledge structures, share them with others, and consciously modify them (Novak and Cañas 2008), facilitating an interplay between individual and collective learning processes which is key in open-ended problem solving (Fabricatore and López 2014). In the second case, expert skeleton maps are prepared by domain experts to facilitate the initiation of a knowledge construction process, and to scaffold the progressive assimilation and generation of new knowledge (Novak and Cañas 2008; Novak 2010). Knowledge construction unfolding through the exploration of a domain should be supported by reliable initial knowledge of that domain (Novak and Cañas 2008). Expert skeleton maps can be effectively used to model initial knowledge, creating scaffold structures based on which learners can construct new knowledge, assimilating, ideating and integrating new concepts and relationships (Novak and Cañas 2008). Scaffolding initial learning minimises the risk of introduction of misconceptions in the knowledge structures built by learners, and facilitates their remediation (Novak 2010). Furthermore, expert skeleton maps can foster the development of thinking strategies: by integrating knowledge about "what" experts think as well as "how" and "why" they think, skeleton maps can help students learn "how" to think as much as "what" to think.

To tackle the challenge of knowledge provision in the context of game design thinking education, we conceived the game design knowledge map as an expert skeleton map aimed at supporting knowledge construction throughout the game design and development process. For this, we conceptualised game design thinking as a human-centric problem-solving process (Fabricatore and López 2014, in press), aimed at addressing a specific core question:

- How can a game attract players, involve them in the game and keep them playing?

This question was formulated to explicitly represent the desired outcome of the game design problem-solving process (i.e. an engaged player), and therefore serve as focus question to support a meaningful exploration of the knowledge map (Novak and Cañas 2008).

In design thinking terms, our conceptualisation considers players as the key stakeholders of the problem scenario tackled by the game designer. Sustained engagement is then the key value to deliver to the stakeholders, and the game represents the system to be designed in order to deliver such value. Finally, the context of use is represented by the physical and social space which frames a gameplay activity but is not directly involved in it. The game design thinking process then requires to abductively hypothesize working principles defining, for a given target audience, which abstract gameplay activity aspects could be accessible, attractive and engaging, and why (e.g. challenge, reward, support, control, narrative, etc.) Games can be consequently created as systems of interacting elements suitable to promote the desired gameplay activity aspects, defining concrete features based on the hypothesized working principles. Designed features can be finally implemented and tested, in order to confirm or (re)defined game system features and engagement working principles, as need be. This creation process is carried out iteratively and incrementally, through cycles of design, implementation, testing, and evaluation, until the desired engagement value is generated.

According to this conceptualisation, our design thinking knowledge map integrated multi-disciplinary expert knowledge from the domains of computer science, game design and development, systems design and the social sciences, extrapolated from leading academic and practitioner literature. Embracing a human-centric approach, we determined that the knowledge to be included in the map had to be relevant to support design decision-making processes and cover the "what", "how" and "why" regarding:

(i) Users
(ii) Elements and patterns of organization of game systems
(iii) Elements and patterns of organization of natural contexts of use
(iv) Engagement principles that allow predicting and explaining how users may perceive and consequently interact with systems, in relation to their needs, preferences and contexts of use
(v) Approaches for testing impacts of a system on user and context of use

Nodes in the game design knowledge map encapsulate concepts reflecting key subproblems or activities involved in the user-centric game design endeavour, along with guiding questions conceived to orient and stimulate related reasoning, investigation, ideation of alternative options, and decision-making (Fig. 9.2).

In order to promote iterative and incremental exploration of the game design knowledge map, we defined its general structure based on a hierarchical star topology pattern. The core of the map covers four key activities involved in the game design endeavour:

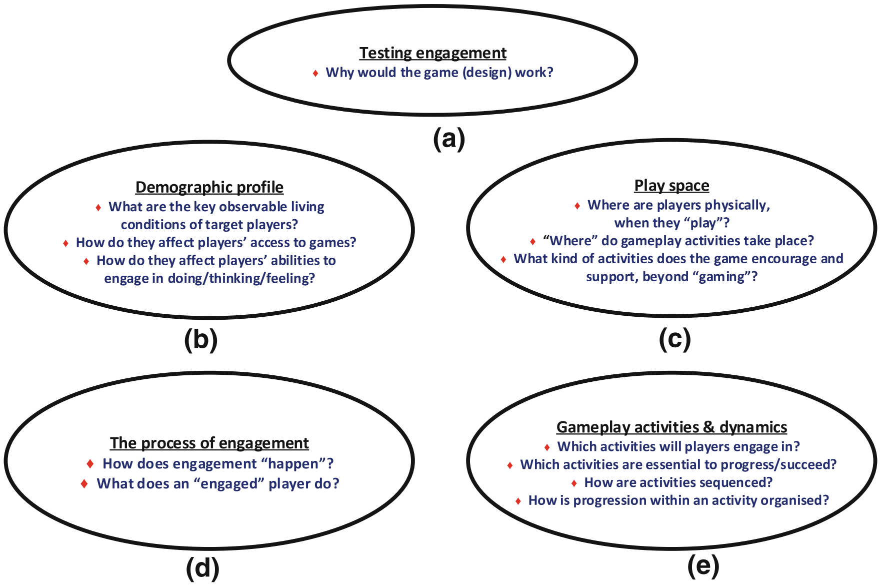

Testing engagement
♦ Why would the game (design) work?
(a)
Demographic profile
♦ What are the key observable living conditions of target players?
♦ How do they affect players' access to games?
♦ How do they affect players' abilities to engage in doing/thinking/feeling?
(b)
Play space
♦ Where are players physically, when they "play"?
♦ "Where" do gameplay activities take place?
♦ What kind of activities does the game encourage and support, beyond "gaming"?
(c)
The process of engagement
♦ How does engagement "happen"?
♦ What does an "engaged" player do?
(d)
Gameplay activities & dynamics
♦ Which activities will players engage in?
♦ Which activities are essential to progress/succeed?
♦ How are activities sequenced?
♦ How is progression within an activity organised?
(e)

Fig. 9.2 Examples of nodes representing: **a** activity of the design endeavour; **b** sub-problem concerning the player; **c** sub-problem concerning context of use; **d** sub-problem concerning engagement working principles; **e** sub-problem concerning features of the game system

(i) Profiling target players: defining key characteristics of the target audience which may affect the way they perceive and interact with a game system.
(ii) Planning player engagement: identifying working principles suggesting which game aspects could trigger and sustain player engagement and why.
(iii) Designing the environment for engagement: analysing and defining features of the game system and its context of use to promote player engagement according to the working principles identified through (ii)
(iv) Testing engagement: prototyping/implementing features of the game system according to what designed through (iii), and testing to verify effects predicted according to the working principles identified through (ii)

Each node in the core represents a design activity with related guiding questions. Relationships between activity nodes represent their key logical relationships. Activity nodes and relationships were conceived to promote comprehension and exploration of what needs to be done, for what purpose, and how activities support one another throughout the iterative design process. Thus, the core of the knowledge map is overall aimed at supporting the student's assimilation of the whole design process, fostering the understanding of its core activities as much as the thinking underpinning their planning and execution (Fig. 9.3).

Each activity node is connected to a hierarchy of relevant design sub-problem nodes, along with guiding questions to orient related investigation and decision-making (e.g. Fig. 9.4). Sub-problems nodes reflect objectives to be tackled through

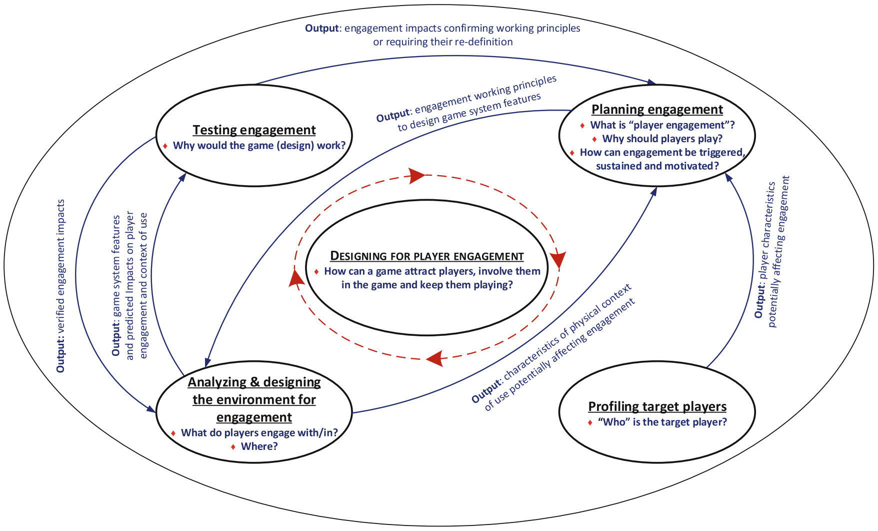

Output: engagement impacts confirming working principles or requiring their re-definition
Testing engagement
Why would the game (design) work?
Output: engagement working principles to design game system features
Planning engagement
What is "player engagement"?
Why should players play?
How can engagement be triggered, sustained and motivated?
Designing for player engagement
How can a game attract players, involve them in the game and keep them playing?
Output: verified engagement impacts
Output: game system features and predicted impacts on player engagement and context of use
Output: player characteristics potentially affecting engagement
Output: characteristics of physical context of use potentially affecting engagement
Analyzing & designing the environment for engagement
What do players engage with/in?
Where?
Profiling target players
"Who" is the target player?

Fig. 9.3 Core of the game design knowledge map

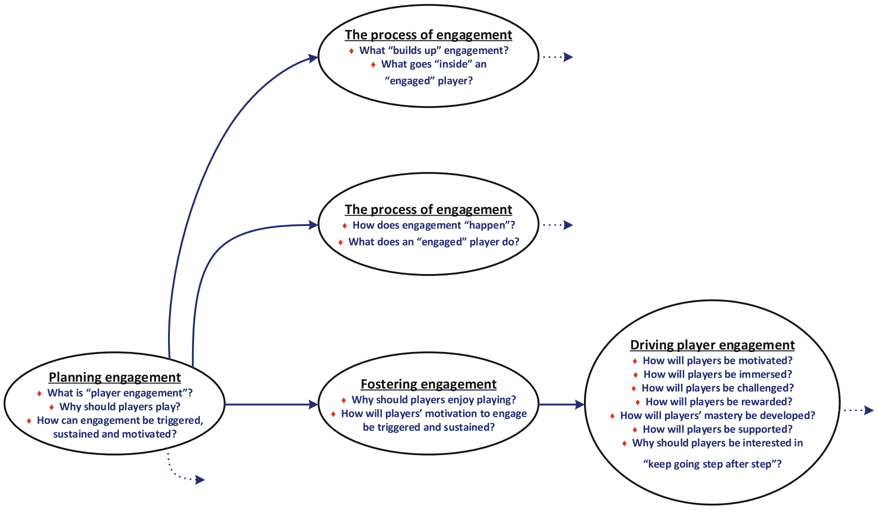
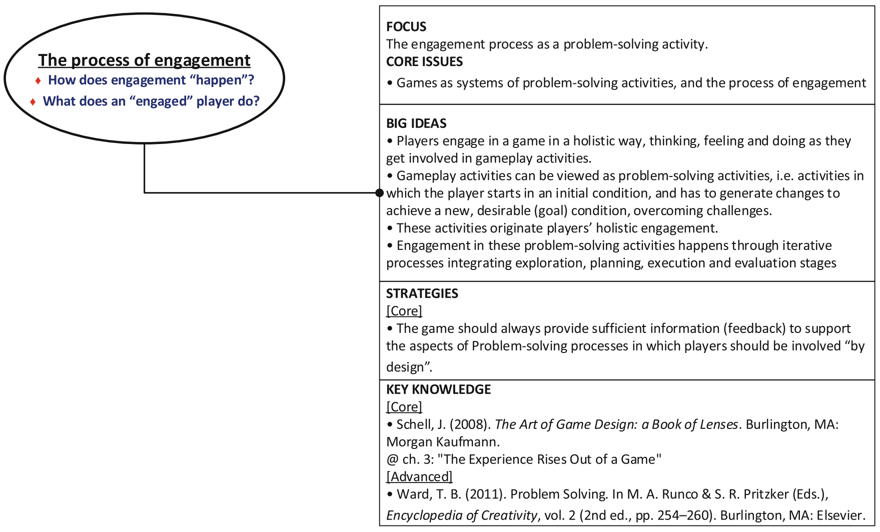

The process of engagement
What "builds up" engagement?
What goes "inside" an "engaged" player?
The process of engagement
How does engagement "happen"?
What does an "engaged" player do?
Planning engagement
What is "player engagement"?
Why should players play?
How can engagement be triggered, sustained and motivated?
Fostering engagement
Why should players enjoy playing?
How will players' motivation to engage be triggered and sustained?
Driving player engagement
How will players be motivated?
How will players be immersed?
How will players be challenged?
How will players be rewarded?
How will players' mastery be developed?
How will players be supported?
Why should players be interested in "keep going step after step"?

Fig. 9.4 Example of hierarchy of sub-problems

each activity, and have been defined to promote comprehension of which sub-problems a given activity might have to address, and reflection on what should be considered in order to address them.

For a given node, a game design knowledge map may provide additional knowledge to fully cover "what" to address, "how" and "why", through a 4-layered knowledge block (Fig. 9.5). The first layer of the block aims at promoting deeper reflection



The process of engagement
♦ How does engagement "happen"?
♦ What does an "engaged" player do?

FOCUS
The engagement process as a problem-solving activity.
CORE ISSUES
• Games as systems of problem-solving activities, and the process of engagement

BIG IDEAS
• Players engage in a game in a holistic way, thinking, feeling and doing as they get involved in gameplay activities.
• Gameplay activities can be viewed as problem-solving activities, i.e. activities in which the player starts in an initial condition, and has to generate changes to achieve a new, desirable (goal) condition, overcoming challenges.
• These activities originate players' holistic engagement.
• Engagement in these problem-solving activities happens through iterative processes integrating exploration, planning, execution and evaluation stages

STRATEGIES
[Core]
• The game should always provide sufficient information (feedback) to support the aspects of Problem-solving processes in which players should be involved "by design".

KEY KNOWLEDGE
[Core]
• Schell, J. (2008). *The Art of Game Design: a Book of Lenses*. Burlington, MA: Morgan Kaufmann.
@ ch. 3: "The Experience Rises Out of a Game"
[Advanced]
• Ward, T. B. (2011). Problem Solving. In M. A. Runco & S. R. Pritzker (Eds.), *Encyclopedia of Creativity*, vol. 2 (2nd ed., pp. 254–260). Burlington, MA: Elsevier.

Fig. 9.5 Example of knowledge block

on "what", through outlining key foci and issues that should be considered when addressing the concept encapsulated in the node. The second layer aims at promoting deeper reflection and comprehension of "what" and "why", through presenting big ideas explaining the nature and importance of the issues presented in the first layer. The third layer aims at promoting deeper comprehension of "how" and "why" to tackle the issues presented in the first layer, accounting for the big ideas presented in the second layer. For this, appropriate strategies are provided (e.g. formal techniques, guidelines and principles). Finally, the fourth block aims at providing knowledge that directly underpin issues, big ideas and strategies, through presenting references to relevant literature and other sources (e.g. podcasts, webinars, etc.).

Overall, the organisation of the game design knowledge map was conceived to facilitate incremental exploration, assimilation and construction of operational knowledge throughout the design process, driven by its core activities and their outcomes. For this, knowledge was clustered so that the apprentice designer could easily assimilate, find and access it depending on the activity being planned or carried out, and the sub-problems tackled. Nodes and their related knowledge blocks were hierarchically organised so that their incremental exploration could drive incremental understanding of activities and sub-problems, and acquisition of conceptual tools to tackle them. Finally, the structure of the knowledge map was underpinned by a rationale of "operational incrementality and immediacy". On the one hand, the deeper the exploration of the hierarchy of nodes and related knowledge blocks, the greater the possibility to gain a deeper, more detailed understanding of design sub-problems, possibilities to tackle them and underpinning rationales. On the other hand, each node and each layer of the related knowledge block was planned to contribute by itself knowledge immediately useful to progress the design process: students to not need to explore the whole map to gain benefits, as each node may represent an opportunity to reflect and design "more" or "better".

9.8 Impacts of the Game Design Knowledge Map: Some Preliminary Results

We developed the game design knowledge map and used it in a range of game design and development modules delivered at the School of Computing and Engineering of the University of Huddersfield (UK). The version of the map discussed in this study was developed in 2014, and used in 2014–15, 2015–16, 2016–17 and 2017–18, in a total of eight level five and four level six modules of a BA Game Design programme (second and third undergraduate year, respectively).

A master map was created to comprise and integrate all the contents covered by all modules, to promote continuity and transferability of learning for students attending different modules at different levels. Then, for each module there was a specific map, consisting in the portion of the master map that addressed the intended learning outcomes of that module. Thus, for each module the map was made available in two versions: module-specific, strictly covering the contents of the module's syllabus, and the entire master map.

Each year the master map has undergone updates to ensure currency of contents (e.g. updating examples), address clarity issues (e.g. clarify phrasing), or update examples. This, however, has not significantly changed the structure of the knowledge map, nor its core contents.

For each module both versions of the knowledge map were made available to students online, through the University virtual learning environment (Blackboard). Students had the option to consult the maps online or download and use them offline.

The use of the map was not mandatory, and alternative resources were made available to access the required information through the virtual learning environment (e.g. lecture slides and notes from former editions of the modules, official module readings, etc.).

For all the modules the academic year was articulated in 24 sessions, on a weekly basis (excluding holiday breaks). During the first seven to nine sessions tutors used the map to introduce core contents, i.e. the ones essential to fulfil the module learning outcomes and corresponding to the module's syllabus. Core contents were presented through a combination of seminars, case studies and guided workshops.

For the remaining weeks students worked exclusively on a game project based on a brief provided at the beginning of the academic year. The project was articulated in two or three of formative milestones (depending on the module) and one final summative milestone. Formative milestones had the main purpose of reviewing project progression and providing formative tutor feedback to provisional project outcomes. Such feedback was based on contents of the knowledge directly related to provisional project outcomes, as well as contents not apparently leveraged in the project, but which could orient future developments. Project workshops were run between milestones. These represented an instance to promote peer testing and discussion of projects, as well as further provision of tutor formative feedback, all based on the contents of the knowledge map (module-specific or master, depending on what individuals or groups of students decided to use).

In each module students were required to document their design work through a combination of written and visual materials, according to some module-specific requirements. For this, a module-specific template was provided to them, with guidelines to define and organise contents in a structure that reflecting the contents and organisation of the appropriate module-specific version of the game design knowledge map. Students were also required to produce a structured project journal, documenting key decisions and reflections on the production process and state of the project after each milestone. The journal was structured so that students were encouraged to reflect on their project using contents of the module-specific knowledge map as "lenses" to evaluate their design decisions and their implications.

We are currently investigating data regarding the impacts of this strategy, and the specific influences that the game design knowledge map has had on the student learning experience. In this chapter, we present the results from 51 second-year students that used the knowledge maps in two game design and development modules during the academic year 2014–2015. Participants were predominantly male (84.3%) with a mean age of 21.2 (SD = 2.49). Students who were enrolled in both courses had access to the module-specific knowledge map and to the same master map. At the end of the academic year, students completed a module evaluation questionnaire, which included an open-ended question asking about the main positive aspects of the knowledge maps. The qualitative data gathered through this question were analysed using a content analysis procedure as described by O'Cathain and Thomas (2004). Thematic categories were identified and coded by the two authors, who jointly decided the categorisation scheme, individually classified the comments and discussed coding disagreements until consensus was reached. Coded comments were then subject to descriptive quantitative analysis to identify trends in students' opinions.

Results are presented in Table 9.1. Several positive aspects were perceived by students, among which three can be identified as major trends:

a. **Enhance accessibility of knowledge**. More than 60% of students expressed an appreciation for the quantity, clarity and ease of access of the information available in the knowledge maps. Many students valued the presentation of a large amount of information in a well-structured format, which made contents easy to understand and quick to find. Examples of students' comment were: "*It provided a quick and easy access to information to help us*"; "*Easy to follow*" and "*It had all the information in an easy way to understand it*". These results are in line with literature indicating that organisation of knowledge is as important as the content, since the structure of information implicitly embeds methods to approach and manipulate that knowledge (Cross 2001; Kokotovich 2008; Oxman 2004). The results suggest that the knowledge maps were structured in a way that

Table 9.1 Positive aspects of the Knowledge maps identified by student

Category of comment	N (total n = 51)	Frequency (%)
Cognitive and operational accessibility of knowledge	33	64.7
Quantity, clarity and structure of content	30	58.8
Ease of access	3	5.9
Support for thinking	22	43.1
Support for conceptualisation and analysis of the design situation	13	25.5
Support for formulation of rationale	5	9.8
Support for ideation of solution	5	9.8
Support for planning and execution of design process	10	19.6
Support to structure and produce design documents	9	17.7
Support to fulfil module requirements	4	7.8
Support for collaborative processes (organisation of teamwork, definition of shared goals)	3	5.9
Support for interestingness and enjoyment of design process	1	2.0
Transferability to other courses/domains	1	2.0
No comments	5	9.8

enabled the cognitive and operational access to the information, facilitating its use and application in the design process.

b. **Support for thinking**. About 44% of students perceived that the knowledge maps supported their thinking processes, helping them to structure their ideas and analyse the design problems tackled. Some students made general comments about the support provided by the knowledge maps, like "[It] a*sks enough questions to get you thinking*" and "*Helps to guide your thought processes*". Other students reported more specific descriptions of the cognitive process aided by this tool. Among these, most students' comments were related to the usefulness of the knowledge maps to conceptualise and analyse the design situation. They observed that "*The breakdown of questions made it easier to understand and analyse [the problem]*", "*helped me to concisely explain my ideas*", "*link different topics*" and "*made my study easier—more informed ideas + clearer rationale in my work*". Previous studies (e.g. Kokotovich 2008; Mathias 1993) have compared thinking strategies of novices and experts, finding that novices often rush towards a design solution based on insufficient analysis and understanding of the design problem to be tackled, evidencing shallower and less structured thinking processes. Conversely, the experts' approaches presented most of the features of design thinking, described as iterative, systemic-oriented, and acknowledging

> the complex relationship between design elements. In this regard, the knowledge maps emerged as valuable tools to support students in their design thinking, promoting the organisation of their analyses in a more reasoned, systematic and deeper way.

Students also highlighted the role of the knowledge maps in supporting the planning and execution of the design process, as well as the production of design documents. Representative examples of comments were that it "*offered a lot of information to help in the design process*", "*the work was more manageable by using them*", and "*the map allows organization of the project and makes it more efficient*". Although these topics were less frequently mentioned by students, we believe they represent further indicators of benefits of using the knowledge maps. These results suggest that knowledge maps helped student not only to understand "what" to do, but also "how" to do it, directing students towards more appropriate strategies and methods for accessing and handling the design knowledge.

9.9 Conclusive Reflections

In this chapter we have discussed the importance of human-centrism in modern computing, and the consequent need of adopting human-centric perspectives in CS education. We have emphasized the necessity for CS education to foster design thinking, being this a systematic problem-solving strategy key to conceive human-centric computing technologies suitable to address meaningful human needs in complex, ill-defined problem contexts. Accordingly, we have highlighted the importance and difficulty of imparting explicit, structured knowledge to support student learning regarding "what" to think, as much as "how" to think and "why". We have then proposed our approach to tackle such challenge of in the context of game design and development education: game design knowledge maps.

Game design knowledge maps are a variant of hierarchical concept maps, conceived to integrate and structure multidisciplinary knowledge regarding game systems, players, player engagement principles, and design and testing processes. We have designed the structure of game design knowledge maps so that students can explore imparted knowledge iteratively and incrementally, driven by the design problem-solving processes tackled and their provisional outcomes.

We have developed and tested game design knowledge maps for over four years, with encouraging results. The evidence analysed so far indicates that knowledge maps integrated large amounts of information in an easily accessible structure which fostered students' thinking processes, helped them connect themes and ideas, and guided them through the design process. All accounted for, this suggests that properly structured imparted knowledge can be effective in helping students to learn "how" to think, not just "what" to think.

Looking onwards, we plan of course to complete a thorough investigation of impacts of game design knowledge maps that we have registered so far. We believe, however, that the applicability and potential benefits of design knowledge maps should be investigated in other domains as well. Although the information contained in game design knowledge maps is obviously domain-specific, their prototypical structure and underpinning rationale are not. Many are the fields in which design can be conceptualised as a human-centric problem-solving process that accounts for the needs of key human stakeholders, context of use and possible solution to address stakeholders' contextualised needs (Buchanan 1992). Similarly, analysis of users' profiles, needs and context, definition of engagement strategies, and design and testing of solutions are activities common to design processes in diverse fields (Buchanan 1992; Dunne and Martin 2006). Hence, we believe that design knowledge map could be valuable tools to impart structured knowledge and foster design thinking in fields many other fields focussed tackling human-centric ill-defined problems, such as engineering, environmental planning, economics, architecture, etc.

Furthermore, the rationale underpinning the knowledge maps is based on the assumption that human-centric design is a process that relies on the abductive formulation hypotheses linking contextualised stakeholders' needs, value to be generated in order to satisfy those needs, current state and functioning of stakeholders, and systems to be designed in order to generate the desired value. Therefore, we believe that variants of the knowledge map could be developed for any other fields of education in which students are required to tackle open and human-centric problems.

References

Bannon L (1991) From human factors to human actors: the role of psychology and human-computer interaction studies in system design. In: Greenbaum JM, Kyng M (eds) Design at work: cooperative design of computer systems. Lawrence Erlbaum Associates, Hillsdale, pp 25–44

Buchanan R (1992) Wicked problems in design thinking. Des Issues 8(2):5–21

Cross N (1982) Designerly ways of knowing. Des Stud 3(4):221–227

Cross N (2001) Designerly ways of knowing: design discipline versus design science. Des Issues 17(3):49–55

Dorst K (2011) The core of "design thinking" and its application. Des Stud 32(6):521–532

Dunne D, Martin R (2006) Design thinking and how it will change management education: an interview and discussion. Acad Manage Learn Educ 5(4):512–523

Dym CL, Agogino AM, Eris O, Frey DD, Leifer LJ (2005) Engineering design thinking, teaching, and learning. J Eng Educ 94(1):103–120

Fabricatore C, López X (in press) Education in a complex world: nurturing chaordic agency through complexity science and game design. In: Visser J, Visser M (eds) Seeking understanding: the lifelong pursuit to build the scientific mind, Sense Publishers

Fabricatore C, López X (2013) Fostering creativity through educational video game development projects: a study of contextual and task characteristics. Creativity Res J 25(4):418–425

Fabricatore C, López MX (2014) Complexity-based learning and teaching: a case study in higher education. Innovations Educ Teach Int 51(6):618–630

Fabricatore C, López X (2015) Higher education in a complex world: nurturing "chaordic" influencers. In :Proceedings of the sixth advanced international colloquium on building the scientific mind (BtSM2015), Learning Development Institute, Jupiter, pp 1–11. Retrieved from http://www.learndev.org/BtSM2015.html

Giacomin J (2014) What is human centred design? Des J 17(4):606–623

Grudin J (2012) A moving target: the evolution of HCI. In: Jacko JA (ed) The human-computer interaction handbook: fundamentals, evolving technologies, and emerging applications, 3rd edn. Taylor & Francis Group, New York, pp xxvii–lxi

Jaimes A, Gatica-Perez D, Sebe N, Huang TS (2007) Human-centered computing—toward a human revolution. Computer 40(5):30–34

Koh JHL, Chai CS, Wong B, Hong H-Y (2015) Design thinking for education: concepts and applications in teaching and learning. Springer, Singapore

Kokotovich V (2008) Problem analysis and thinking tools: an empirical study of non-hierarchical mind mapping. Des Stud 29(1):49–69

Mathias JR (1993) A study of the problem solving strategies used by expert and novice designers (Doctoral Dissertation). Aston University, Birmingham, UK.

Novak JD (2010) Learning creating and using knowledge: concept maps as facilitative tools in schools and corporations. J E-Learn Knowl Soc 6(3):21–30

Novak JD, Cañas AJ (2008) The theory underlying concept maps and how to construct them. Technical report IHMC CmapTools 2006-01 Rev 01-2008, Florida. http://cmap.ihmc.us/publications/researchpapers/theoryunderlyingconceptmaps.pdf

O'Cathain A, Thomas KJ (2004) "Any other comments?" Open questions on questionnaires—a bane or a bonus to research?, BMC Medical Research Methodology

Oxman R (1999) Educating the designerly thinker. Des Stud 20(2):105–122

Oxman R (2004) Think-maps: teaching design thinking in design education. Des Stud 25(1):63–91

Taylor P (2000) Designerly thinking: what software methodology can learn from design theory. In: Gray J, Croll P (eds) Proceedings international conference on software methods and tools, smt 2000. IEEE Computer Society, Los Alamitos, pp 107–116

Chapter 10
Fostering Inclusivity Through Dynamic Teaching Practices

Arjab Singh Khuman

Abstract Teaching within the context of Higher Education (HE), often involves interacting with a wide variety of students, who come from, and have varying backgrounds, not just in their capabilities, but also with regards to their understanding. As such, dynamic teaching styles and practices need to be adopted in order to allow for inclusivity. This chapter highlights a particular case study involving the author, and a module that he currently leads—IMAT3406: Fuzzy Logic and Knowledge Based Systems. The teaching style and approaches adopted allow for a better understanding of core concepts, from which better executed applications can be garnered. Making sure that *it makes sense to all*, ensures that the foundational knowledgebase needed from which to build upon is adequately in place, so that everyone in the cohort is on a level playing field. This can be achieved through dynamic teaching practices, often involving acclimation and assimilation to the cohort. Making sure that a concrete understanding exists before the students are encouraged to undertake their coursework, has proved to cater for exceptional output, not only in terms of detail, but both in quality and substance. Through tried and tested means, the case study used in this chapter sheds light on the attributes of a successful approach; describing how the author's own accession to harbouring inclusivity is adopted and executed for the module IMAT3406.

Keywords Dynamic teaching practices · Inclusivity · Assimilate · Acclimate

10.1 Introduction

Teaching with regards to Higher Education (HE) often does not involve a single track of execution. By that, personnel and staff members alike are encouraged to cater for the addition of students whom may not satisfy the general norm of an *expected* student. There may be students whom are on the spectrum; students with

A. S. Khuman (✉)
De Montfort University, Leicester, England
e-mail: arjab.khuman@dmu.ac.uk

J. Carter et al. (eds.), *Higher Education Computer Science*,
https://doi.org/10.1007/978-3-319-98590-9_10

physical disabilities; students with underlying mental disabilities; students whom are not aware of issues they may currently be experiencing, and so on. Within these categories, there are many sub-categories, which splinter off into specific nuances and niches. Regardless of what issues are presented the learning objectives of the module must be attainable by all. To ensure this, the teaching practices need to allow for a non-stringent adaptability, practices that are fully inclusive to all enrolled. Exclusion of a single learner due to circumstance is morally abhorrent as well as a blatant disregard to institutional and government policy. The qualities of a good practitioner of pedagogy should always have the ability to be dynamic, resourceful and patient. Regardless of the context of the module, programme, course or any other instance of classroom based learning, a dynamic approach to teaching is beneficial for all. By being adaptable allows for absolute inclusion of all in the cohort, ensuring that the delivery of teaching objectives are presented and more importantly, that it is understood.

The author of this chapter is the current module leader for IMAT3406: Fuzzy Logic and Knowledge Based Systems, a module belonging to the Faculty of Technology at De Montfort University, United Kingdom. The weighting of the module is 50% coursework and 50% end of year exam, this will become 100% coursework for the 2018/19 academic year onwards. This particular module has been chosen as a case study due to its relevance and advocacy of adapting to change. It is noteworthy to extend a mention that the author received all his degrees from the institute he now lectures for, De Montfort University, where he obtained his B.Sc., M.Sc. and Ph.D. This is a key point, being a student to transitioning to post graduate study, to then transition to academic staff member, all the while at the same HE institute, provides for a unique perspective. It is precisely this perspective that has allowed for the IMAT3406 module to flourish in recent years, with consistent praise and recognition to the teaching practices being adopted. Being more aware of the culture and coupled with the fact that the author was a student on a variation of the module, has all played a part in shaping the module as it is seen today. The changes the author would have liked to have seen been implemented as a student are the changes that he has implemented himself; practice what you preach, so to speak. This is the main reason why the module itself is so well received by so many, it is the embodiment of what the author would have liked to have received as a student himself, somewhat subjective, although effective.

It is also noteworthy to mention that the author was a recipient of a 2018 Vice Chancellor's Distinguished Teaching Award (VCDTAs). The VCDTAs were established in 2005 in order to recognise and celebrate the lecturers who have inspired and motivated their students to succeed. Not only do students nominate lecturers, they sit on the awards panel, helping to recognise the creativity and outstanding quality of De Montfort's teaching staff. This in itself validates the effectiveness of the practices that the author adopts for the delivery of the module.

The author is also the programme leader for G50052: Intelligent Systems, where the fuzzy module is compulsory for all final year (Level 6) students. The module itself is a 15-credit module and is currently undertaken during Term 1 (October–December) of the academic year. The module itself is available to several other programmes in addition to Intelligent Systems, such as; Computer Science, Software Engineering; Computer Security; Forensic Computing; Computing; Computer Games Programming; Erasmus Exchange; and Maths. Year in and year out, the module has become more popular, with current enrolment numbers reaching roughly 170 students. The enrolment figures for the 2018/19 academic year are currently at 196 students. Maths is the latest programme to allow for the module to be an option for final year students. The module is very popular with students from the Erasmus Exchange programme, which is essentially a student swap from an international institute. These students will be well versed in the English language, nonetheless, additional teaching practices are implemented to ensure an absolute understanding by all.

10.2 What Is Fuzzy Logic?

The module itself is best understood as being a mathematical means to model uncertainty. Fuzzy logic is ideally suited to modelling vague, abstract concepts such as linguistic information. To crisply define a notion like that of *'Tall'* is inherently difficult when only considering a crisp, classical understanding of membership. If one was to model the notion of *Tall* using conventional crisp means, it can be assumed that 6 foot would indeed be classified as *Tall*, and would absolutely, unequivocally belong to the set *Tall*. Fuzzy logic however, using the application of fuzzy sets, allows for one to use a degree of membership, allowing one to quantify the belongingness to a particular set. A classical crisp set approach has only 2 possible outcomes; the observed object belongs to the set, or the observed object does not belong to the set and belongs to its complement. Black and white, up and down, left and right, very clear cut, no grey areas. It's either you are or it's either you're not.

A fuzzy set allows for an observed object to have partial belongingness to a set, a degree of truth; a single object can belong to several sets to varying degrees. This is in contrast to the classical approach as we are no longer restricted to either a; yes it does belong, or a no it does not, we now have the option to how much does it have an association. The rationale of fuzzy is that it allows for one to better model humanistic nature so that better computational models can be created, a more accurate model leads to better inference and better decision making based on said inference. The more detail one can capture foundationally, the more accurate the output will ultimately be.

Classical set theory makes use of Boolean logic (Boole 1847, 1854), whereby an object is classified as absolutely belonging, or absolutely not belonging to a particular set (Cantor 1895). The use of crisp boundaries applies an inherent level of strictness to what the set can model, instances where only two outcomes are allowed, such as an integer being either odd or even; such instances are easily handled using this classical approach. However, there is a need to encapsulate uncertainty that is associated to vagueness when considering human based perception. Human nature and inferencing does not work in such a precise and crisp manner, a humanistic approach needs to cater for the existence of imprecision based uncertainty, along with vagueness. The understanding of a set from a classical perspective is not a fitting synthesis for human intuition. The notion of *mereology* described by (Lesniewski 1929), considered the idea of an object being partially included in a set, this was the basis for the formulation of Max Black's vague set (Black 1937), created in the 1930s.

The building blocks of any fuzzy implementation involves the use of fuzzy sets, first proposed by (Zadeh 1965). A fuzzy set can be seen as an extension of the ideology of a vague set. From its inception fuzzy logic has been further expanded upon to establish itself as a powerful and successful paradigm for modelling uncertainty (Zadeh 1973). As logic is associated to propositions, fuzzy logic can be seen as the calculus of fuzzy propositions. Mathematical applications for precise reasoning will often need crisp understandings, however, this becomes problematic when concepts such as natural language are involved. Linguistic vernacular can be inherently vague, with a prevalent amount of ambiguity. Our daily existence will often be littered with varying degrees of uncertainty, further invoking various aspects of specific uncertainties (Zadeh 1999).

The use of a classical set for the modelling of unclassical behaviour will often fall foul when considering the vagueness of uncertainty. For example, the abstract concept of *Tall* cannot be universally defined, a single crisp value cannot be put forward as an indicative representation that is agreed upon by all. What is *Tall* to some may not be as *Tall* to others. Figure 10.1 provides a visualisation of what a typical crisp bounded set may look like. The plot in the figure describes any person being $6'$ or taller as a validated member of the set *Tall*. However, using this precise and strict definition, it would neglect any instance of anything less than $6'$. It can be generally assumed that $6'$ is indeed *Tall*, but so too is $5'\ 11''$, at least to some extent. The only association one can attribute to a value is if it is included in the set *Tall* or not. The problem now becomes one of determining the bounding of the set, to realistically encapsulate all common held assumptions of what satisfies the notion of being *Tall*. This echoes the sentiment of Sorites paradox, arising from a vague predicate. A single grain of sand does not constitute a heap, nor does two grains of sand. However, when we have a billion grains of sand, we then certainly do have a heap of sand, at what point do we transition from not being a heap, to becoming a heap? A fuzzy perspective will allow for a more forgiving approach, one that enables an object to have partial belongingness.

The most fundamental aspect of fuzzy set theory is its understanding of numbers. A fuzzy number is ideal for describing linguistic phenomena, where an exact description of its state is unknown. Fuzzy numbers were first introduced by (Zadeh 1975), for the purpose of approximating real numbers which deal with uncertainty and imprecision associated to quantities. It has great scope when approximating *height*, or *weight* and other such uncertain abstractions. The apex of a fuzzy number will generally be the only point where an object can be given a maximum degree of inclusion equal to 1. The varying degrees of membership for other objects will be indicative to their proximity to the apex. Fuzzy sets extend the notion of fuzzy numbers to allow for more versatility.

If one was to describe the set *Tall* as seen in Fig. 10.1, using a fuzzy approach, a possible visualisation may look like the plot contained in Fig. 10.2. In this plot, the inclusion of other possible values that could be deemed as *Tall*, values such as 5′11″ would also be included, but to a lesser degree when compared to that of 6′. Following this understanding, 5′10″ would also be a viable candidate for inclusion, but to a lesser degree than 5′11″, and so on. Using a fuzzy perspective for encapsulation, one is able to relax the expected strictness one would associate with a crisp set. Not only does a fuzzy set allow for a more harmonic understanding of uncertainty, but it is also able to fall back to a classical interpretation if need be. The degree of belongingness may be that of absolution, or absolutely not, in which case, a fuzzy set can replicate a crisp set (Klir and Folger 1998). In essence, the process of mapping a membership value to an object is known as *fuzzification*. It is only when considering that an object may have partial belongingness, does the strength and applicability of a fuzzy set become apparent.

A fuzzy set on its own can only allow for a certain amount of functionality, the combination of multiple fuzzy sets allows one to extensively model an abstract concept, that would otherwise be very difficult to represent using a crisp understanding. As it is a set of ordered pairs, the object (x) is associated to a degree of inclusion $\mu_A(x)$, the same (x) may belong to more than one set, and as such may be attributed to multiple degrees of inclusion. A fuzzy set goes against the traditional approach of classical set theory, by allowing an object to belong to different sets by varying degrees of membership. Such is the methodology of fuzzy sets, the law of the excluded middle and the law of contradiction are ignored. These two prevalent laws would stop an object from belonging to more than one set if a crisp perspective was used. Continuing with the notion of *Tall*, Fig. 10.3 demonstrates how using an additional fuzzy set, one can allow for a more humanistic approach in understanding the significance of any given value. This plot contains an additional set labelled *Short*.

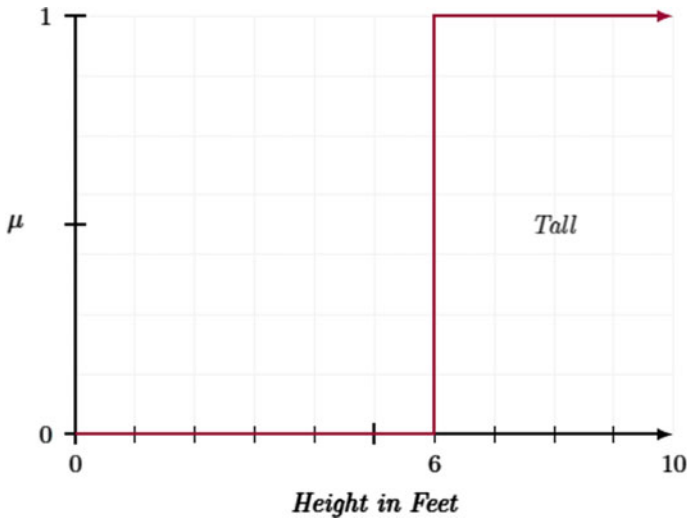


Fig. 10.1 An example of a crisp set

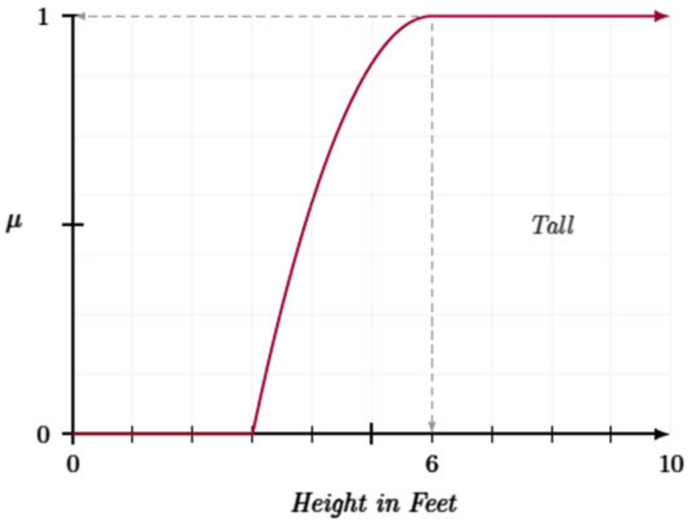


Fig. 10.2 An example of a single fuzzy set

The value 6’ in this instance can be seen to have an absolute degree of association to the set *Tall* with a returned degree of membership of 1, and a partial degree of inclusion to the set *Short* with a returned degree of membership of 0.14. If one inspects the object value 7′, it can be seen to have absolute inclusion to the set *Tall*, and complete non-inclusion to the set *Short*. This logical assumption that being 7′ would never be regarded as being *Short* can be easily catered for, as too can a plethora of other abstract notions. What has been presented is the foundational underpinnings that make up the fuzzy module, without delving deeper into the mechanics, this is what the students will have to grasp in order to make gains on their coursework. There is no prerequisite for the module, as the author presents the syllabus using the assumption that no one has been introduced to fuzzy before; which is often the case.

Fuzzy logic itself is well respected in the academic and research community, with many dedicated conferences and journals. The real world applicability of fuzzy has seen it be deployed in many different and varied industries, further reinforcing the effectiveness of such an approach. Given all this, the need to have a module dedicated to fuzzy is clear to see. From fuzzy one can delve into a multitude of different hybridised concepts and off-shoots (Khuman et al. 2015, 2016a, b, c). The author is a fuzzy researcher, so therefore the students have an academic expert on the subject. This pays dividends in that it allows for core concepts to be explained in a variety of different ways, as opposed to someone just reading off the slides or from a lab sheet. The author’s expert knowledge and enthusiasm for the subject were highlighted in the feedback given by students on the module, this also constituted and contributed to being awarded a 2018 VCDTA.

The aforementioned foundational aspects of fuzzy logic are extended to explain how when piecing it altogether, one can then create a fuzzy inference system (FIS).

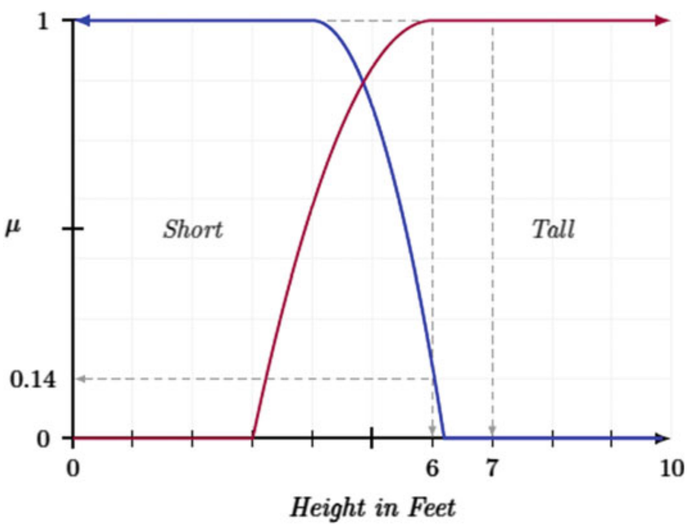


Fig. 10.3 An example of multiple fuzzy sets

The coursework component of the module involves each student creating their own FIS, in doing so, the student will appreciate the intricacies in articulating on their subjective understanding of their chosen application domain.

10.3 Teaching Practices

The configuration of the teaching timetable with regards to the module is; 1*1 h lecture each week, with 1*2 h lab session. The lab sessions utilise the use of MATLAB, which is a multi-paradigm numerical computing environment. A proprietary programming language developed by *MathWorks*. Contained within the install is the Fuzzy Logic Toolbox, which the students are allowed to experiment with to mock up prototypes (Grasha and Yangarber-Hicks 2000). The author anticipates that some of the cohort will be competent programmers and some may not, this does not affect the learning objectives. Most of the students will have no experience with MATLAB, and for those that do, they appreciate the starting from the ground up approach, and see it somewhat as a refresher course on the software. Every lab session in the first few weeks is spent becoming familiar with the software and gaining confidence with the fuzzy library, building up on previous sessions by understanding more functionality. By the end of the module every student will be able to code up in MATLAB and describe the technicalities of their coursework submission.

The capacity of each lab is 20 students, with most labs being timetabled so that there is a least 1 or 2 spaces left spare just in-case there are students attending from other lab sessions. The author leads the entire module, including all lectures and all labs. This was deliberately done so that consistency in teaching style and expertise throughout could be maintained. This commitment was very well received when it came to students providing their feedback. As the current cohort size is roughly 170 students, this results in 9 timetabled lab sessions each week, the times of which varying from first thing—0900:1100, to midday, to late afternoon—1400:1600. The lectures are always Friday—0900:1000. It is noteworthy to mention, module leaders, programme leaders and the like, generally have no say with regards to when and where their modules occur in the timetable. This is an optimisation problem beyond the module leader's control. With this being the case, this adds an additional facet that needs to be taken into consideration when delivering one's lectures (Benett 1993).

10.3.1 Teaching Practices—Lectures

Referring to Table 10.1, one can see that the lecture for the module is Monday morning first thing—0900:1000, this is to the displeasure of the majority of the students. Regardless of the module or type of lecture, students are rarely, fully engaged at that time in the morning. It is the job of the lecturer to engage as much as possible, all the while to not be too overbearing. Expecting the students to absorb everything that is presented to them is an unrealistic expectation given the time of the lecture. This highlights the importance of lecture material being made time specific in accordance to when it is to be delivered; it is advantageous to include content relative to the learning objectives, but not to overwhelm, as this can be counterintuitive. The current method of delivery is to ask the students in the lecture room, '*if they have understood a particular aspect of the lecture material*'. If so, we proceed onto the next aspect, if not, we collectively repeat, presenting the notion using a different perspective.

The learning objectives of the module are a super-set of the learning objectives of each lecture. As such, there is more flexibility in arriving and satisfying the lecture objectives, in so doing, encompassing as much collective participation as possible. Given that this is a third year module, the students are more likely to have already formed a familiarity with the cohort. This can be used as additional tool when incorporating various teaching practices and styles. The module itself, has many times, made use of the students in explaining core concepts vital to the understanding of fuzzy logic. Through a variation of peer-to-peer tutelage, the author encouraged the students who understand the concepts to explain it using their own words to the rest of the cohort. Sometimes hearing it through the words of a fellow student, allows for a more concrete understanding. This is not to imply that all aspects of the lectures are undertaken in this manner, but rather when the situation is called for, this only becomes apparent when the room is gauged. By incorporating student inspired explanations instils an obtainable quality, for the quality of the student doing the explanation can be extremely varied; from the stand-out high fliers, to the well reserved and conservative. Seeing and hearing a fellow student explain can positively affect the learning curve of the room. It should be expected that a student will always

Table 10.1 Module timetable

Tuesday								
Day	Start	End	Weeks	Activity	Type	Module title	Room	Staff
Tue	9:00	10:00	1–5, 7–11	IMAT3406/Y L/01	Lecture	Fuzzy Logic and Knowledge Based Systems	BI0.05	Khuman A
Tue	11:00	13:00	1–5, 7–11	IMAT3406/Y P/01	Practical	Fuzzy Logic and Knowledge Based Systems	GH5.82 Lab	Khuman A
Tue	14:00	16:00	1–5, 7–11	IMAT3406/Y P/02	Practical	Fuzzy Logic and Knowledge Based Systems	GH2.83 Lab	Khuman A
Wednesday								
Day	Start	End	Weeks	Activity	Type	Module title	Room	Staff
Wed	9:00	11:00	1–5, 7–11	IMAT3406/Y P/09	Practical	Fuzzy Logic and Knowledge Based Systems	GH2.81 Lab	Khuman A
Thursday								
Day	Start	End	Weeks	Activity	Type	Module title	Room	Staff
Thu	9:00	11:00	1–5, 7–11	IMAT3406/Y P/03	Practical	Fuzzy Logic and Knowledge Based Systems	GH5.82 Lab	Khuman A
Thu	11:00	13:00	1–5, 7–11	IMAT3406/Y P/04	Practical	Fuzzy Logic and Knowledge Based Systems	GH2.83 Lab	Khuman A
Thu	14:00	16:00	1–5, 7–11	IMAT3406/Y P/05	Practical	Fuzzy Logic and Knowledge Based Systems	GH5.82 Lab	Khuman A

(continued)

Table 10.1 (continued)

Friday

Day	Start	End	Weeks	Activity	Type	Module title	Room	Staff
Fri	9:00	11:00	1–5, 7–11	IMAT3406/YP/06	Practical	Fuzzy Logic and Knowledge Based Systems	GH2.82 Lab	Khuman A
Fri	11:00	13:00	1–5, 7–11	IMAT3406/YP/07	Practical	Fuzzy Logic and Knowledge Based Systems	GH6.52 Lab	Khuman A
Fri	14:00	16:00	1–5, 7–11	IMAT3406/YP/08	Practical	Fuzzy Logic and Knowledge Based Systems	GH6.51 Lab	Khuman A

be willing to explain to a potential population of roughly 170, they are never forced to, or made to feel uncomfortable. They do so using their own volition.

There are 3 distinct teaching styles that are adopted: visual, auditory, and kinaesthetic. These are not necessarily undertaken individually, but rather they are amalgamated. One can move quite quickly from a lecture slide to a group discussion highlighting a real world example of the lecture slide, to then demonstrate this through a thought experiment. The free flowing and adaptable learning approaches do play dividends, as they allow for potentially more cohort participation. Not everyone learns and understands in the same way, being able to present and transcribe key concepts using varying teaching styles does benefit the learning quality of the group (Hsieh et al. 2011). By being able to connect what the students are learning to information they may already know, allows for them to fit new knowledge into their understanding of the topic. By allowing them to have the 'eureka' moment gives them more incentive to encourage future engagement and understanding. This is all possible, and enforced via good communication, the most essential quality when participating in a teaching environment. For the communication can be the difference in explaining something adequately well to get it, or explaining it very well to understand it and articulate from it.

Other teaching practices that the author makes use of are to be enthusiastic. From the received comments regarding the fuzzy module, the author's enthusiasm was constantly remarked upon, this can at times be infectious. However, this is a double edge sword, as too much enthusiasm in a 0900 lecture can be counterintuitive, so there must be an equilibrium between too much and too little. Regulating the enthusiasm as and when needed is a beneficial quality, students on the autistic spectrum do not take too kindly to staff that are too loud and too animated, however, given enough time and reassurance, they will often warm to you, as has been the case for the author.

At De Montfort University, a Universal Design for Learning (UDL) is adhered to, an approach to enhance learning and teaching for *all* our students. It provides a framework to identify and promote existing good practices across the institution, many of which already address the principles of UDL. Apart of this framework is DMU Replay, where it is expected that all modules across all levels, have their respected lectures recorded. This recording is undertaken using the Panopto software, which records the monitor and in doing so the current presentation, along with audio via the embedded microphone found on the lectern. Enrolled students have access to all recorded material and play them back in their own time. This has a huge benefit, as students who were unable to attend a lecture, will not miss out on the learning of the session. This is also hugely beneficial to the students who may be a part of the Erasmus Exchange, where English may not be their first language, for those students, having access to recorded sessions is invaluable.

Good communication and trust between teachers and students is also important. Saying that you will get something done is different to actually getting it done. Maintaining a good a rapport with students reduces the likelihood of insubordination, and therefore making the learning environment more enjoyable. Engaging in discussion needs to be done in such a way as to not come across as domineering, forcing perspectives onto students, even though that is technically what is happening. An understanding should be as harmonic as possible, involving the student as much as possible in the 'journey' of understanding for themselves. Connecting the dots for them is different from letting them connect the dots for themselves, as this takes away from their accomplishment.

10.3.2 Teaching Practices—The Labs

The labs for the IMAT3406 module involve the use of specialised software, in particular: MATLAB. This was chosen for several reasons; included as an additional installation package is the Fuzzy Logic Toolbox, which we make use of for quick prototyping. The toolbox is also needed for its incorporated function library which is used in the coursework when the students create their own fuzzy inference systems. MATLAB was also chosen as not many students have used it. This allows for the cohort to be on a level playing field, programming experience is not needed as the environment is unconventional. The labs are structured so that everyone is taught the necessary skills needed in order to properly be equipped for when they embark on their coursework.

The labs themselves are all 2 h lab sessions. With the main lecture being first thing Monday morning at 0900, the attentiveness of the students in attendance needs to be taken into consideration. Students are expected to attend all timetabled sessions, but as is the nature of early morning starts, especially on a Monday, one can be forgiven for NOT expecting a full turnout every time. If a student was to miss a lecture, the content that was covered would be available via the Virtual Learning Environment (VLE) , in our case, Blackboard. One has to be aware when the lectures are to be undertaken and also to what level of attentiveness the room may be. Having a 2 h lab allows for the learning objectives expected for each lecture, to be reinforced via lab work, for those that may have missed the lecture entirely. Those who did attend the lecture and also the lab, will benefit from a more concrete understanding. Having the lectures and the lab material align somewhat, facilitates this aspect of learning. Making sure that everyone knows what is needed in order to understand what will be asked of them for when they start the coursework component of the module.

It is in the labs where the author is able to be more proactive as the cohort size is a maximum of 20 students. With a smaller population, and 2 h for each session, a more involved approach is adopted. Students are expected to follow along with the lab sheet as the author will be doing the same on the projector. Seeing the lab instructor also undertaking the lab exercises provides for a more inclusive environment because they are able to follow along, or do it on their own. The time is utilised very affectively, as the demonstration of the lab material and lab sheets do not take the entirety of the 2 h session up. What the author will do is to make sure that concepts from the Monday lecture have been understood and if not, a brief recap is undertaken for the benefit of the room (Vogt and Rogalla 2009). This may involve everyone, or only the students who need it.

The author will also take the time to talk to each student individually to make sure they are comfortable with the lab exercises and the lecture content, as sometimes students may not feel compelled to raise their hand if they have a question or concern, so proactiveness on part of the lecturer should always be encouraged. As the lab groups are smaller; the attentiveness to which students are more vocal; which prefer to just get on with it and be left alone; which benefit from one-to-one, which benefit from small group discussions and so on becomes very apparent. It is playing to the strengths of the room that allow for a fruitful and rewarding lab session.

It is also in the lab sessions that the author can spend more time with students from the Erasmus Exchange. For these students may not feel entirely confident in speaking in front of the lecture group, so being in a smaller size setting should be utilised as effectively as possible. The Erasmus Exchange students will already have a good grasp of the English language so they are very capable of asking and questioning given the opportunity.

These are all aspects of dynamic teaching practices that the author adopts in presenting the fuzzy IMAT3406 module. With the background of the author and the past experiences at DMU, this has all played a part in creating and structuring a module with embedded teaching practices that allow for understanding and execution of fuzzy concepts. This has and continues to prove effective, with module gaining more popularity year in and year out.

10.4 Conclusion

There are many facets when considering dynamic teaching practices, the main objective of which should always be to foster for inclusivity, without creating an apartheid in the cohort where more energy and attention is given to a specific group. What has been put forward is a somewhat itemised list of adopted teaching practices. With that being said and done, the proof will be in the pudding, so what is to follow is a breakdown of module statistics, showing the reader how well the students actually fared. As the coursework marks were already finalised and released to the students at the end of Term 1 (December 2017), the exam was the only thing still to be undertaken, which was completed in May 2018. Final marks and weights have been calculated and as such, the following statistics can be inferred.

Of the original 165 enrolled students, 160 remained. The slight drop in number is due to interruptions rather than the module being dropped. In actuality, more students decided to join onto to module within the first 2 weeks of the academic year commencing. Of these 160 students, 3 did not attend any lectures or labs, nor did they submit coursework or complete the exam, they are however still included as being enrolled. With any module there is always a chance that you will have students whom have asked and have been granted a deferral for their coursework, the deferral date means that their coursework deadlines are now August 17th 2018. Which means, at the time of writing this chapter, they are still yet to be submitted and yet to be marked and graded. Of the 150 students whom submitted their coursework, the average mark of the cohort was an impressive 70%. The number of students whom completed the exam was 152, the average score obtained by the cohort was an equally impressive 77%. As the module is weight as 50% coursework and 50% exam, the overall module score obtained was 74%. What is to be contextualised is the module pass rate for level 6 modules here at DMU should aim to be at least 90%, the module pass rate for IMAT3406 including all students, some of which did not attend, some of which have still the coursework to submit, the pass rate is 94%.

If one was to only include the students whom submitted their coursework and sat the exam, the pass rate jumps to 99%. Given the cohort size, either figure: 94 or 99, are very impressive.

A breakdown of the mark range for the overall score for the module considering every enrolled student is as follows:

- 63% gained an overall module score of 70+.
- 21% gained an overall module score of between: 60–69.
- 8% gained an overall module score of between: 50–59.
- 3% gained an overall module score of between: 40–49.
- 1% gained an overall module score of between: 30–39. Which constitutes a marginal fail.
- 5% gained an overall module score of between: 0–29. Which constitutes a major fail. This does include the students whom still have to submit their coursework, so this value will decrease and the overall pass rate will increase accordingly.

As one can imply, the effectiveness of the teaching practices adopted have proved to be effective in obtaining the module scores. As with any HE institute, all marks and exam scripts are subject to moderation, which has already occurred, so the values you see before you are final indicators. A dynamic approach to teaching, one which fosters inclusivity will always fare well and be commended, as has been the case for IMAT3406: Fuzzy Logic and Knowledge Based Systems.

References

Benett Y (1993) The validity and reliability of assessments and self-assessments of work-based learning. Assess Eval High Educ 83–94

Black M (1937) Vagueness. An exercise in logical analysis. Philos Sci 427–455

Boole G (1847) The mathematical analysis of logic. Philo Libr

Boole G (1854) An investigation of the laws of thought: on which are founded the mathematical theories of logic and probabilities. Walton and Maberly

Cantor G (1895) Beiträge zur Begründung der transfiniten Mengenlehre. Mathematische Annalen 481–512. https://doi.org/10.1007/bf02124929

Grasha AF, Yangarber-Hicks N (2000) Integrating teaching styles and learning styles with instructional technology. Coll Teach 2–10

Hsieh SW, Jang YR, Hwang GJ, Chen NS (2011) Effects of teaching and learning styles on students' reflection levels for ubiquitous learning. Computers Educ 1194–1201

Khuman AS, Yang Y, John R (2015) A significance measure for R-fuzzy sets. In: 2015 IEEE international conference on Fuzzy systems (FUZZ-IEEE), pp 1–6. https://doi.org/10.1109/fuzz-ieee.2015.7337808

Khuman AS, Yang Y, John R (2016a) R-fuzzy sets and grey system theory. In: 2016 IEEE international conference on systems, man, and cybernetics

Khuman AS, Yang Y, John R (2016b) Quantification of R-fuzzy sets. Expert Syst Appl 374–387

Khuman AS, Yang Y, John R, Liu S (2016c) Quantification of perception clusters using r-fuzzy sets and grey analysis. In: 2016 international conference on grey systems and uncertainty analysis (GSUA2016)

Klir GJ, Folger TA (1998) Fuzzy sets, uncertainty, and information. Physica-Verlag

Lesniewski S (1929) Grundzüge eines neuen Systems der Grundlagen der Mathematik. Fundamenta Mathematicae 1–81

Vogt F, Rogalla M (2009) Developing adaptive teaching competency through coaching. Teach Teac Educ 1051–1060

Zadeh LA (1965) Fuzzy sets. Inf Control 338–353

Zadeh LA (1973) Outline of a new approach to the analysis of complex systems and decision processes. IEEE Trans Syst, Man, and Cybern 28–44

Zadeh LA (1975) The concept of a linguistic variable and its application to approximate reasoning—I. Inf Sci 199–249

Zadeh LA (1999) From computing with numbers to computing with words. From manipulation of measurements to manipulation of perceptions. In: IEEE transactions on circuits and systems I: fundamental theory and applications, pp 105–119

Chapter 11
Semi-automating the Marking of a Java Programming Portfolio Assessment: A Case Study from a UK Undergraduate Programme

Luke Attwood and Jenny Carter

Abstract Recent changes in Higher Education including larger numbers of students and larger staff student ratios mean that assessments need to be marked quickly and consistently, whilst also benefitting the students' learning experience. This paper initially introduces a portfolio assessment adopted three years ago on a software development module, with the aim of improving student engagement, and then goes on to discuss the inspiration behind attempting to automate the marking of this assessment. The paper then focuses on how the JUnit testing framework was used to achieve this, dissecting the challenges faced along the way, and how these were individually addressed. The end result was considered to be successful, with a significant reduction in marking time, and a guaranteed high consistency between markers. Students also benefitted through ongoing feedback from the unit tests they were provided during the assessment. In future, using such an approach provides the potential to assess students more frequently, giving them more regular feedback on their progress, and further helping them to engage with their studies.

Keywords Technology in education · Automatic marking · Unit testing
Java · JUnit

L. Attwood (✉)
De Montfort University, Leicester, UK
e-mail: Lattwood@dmu.ac.uk

J. Carter
University of Huddersfield, Huddersfield, UK
e-mail: j.carter@hud.ac.uk

J. Carter et al. (eds.), *Higher Education Computer Science*,
https://doi.org/10.1007/978-3-319-98590-9_11

11.1 Introduction

OO Software Design & Development is a second year undergraduate module delivered on the B.Sc. Computer Science and Software Engineering courses at De Montfort University (DMU) that teaches Object Oriented development principles using the Java programming language. For the 2016/17 academic year there were 170 students enrolled on the module, almost twice that of the previous academic year.

Three years ago a portfolio assessment was adopted with the goal of encouraging students to further engage with lab material by completing a selection of questions across a variety of key topics. Ideally, we wanted to assess as many questions as possible, however, there was a trade off with what would be feasible to mark. This led to the idea of marking through sampling. Students were told that we would only mark one of their portfolio questions, but if they did not submit the one we chose to assess, then their mark would be capped at 40%. A further idea was to conclude the assessment with a lab test conducted under exam conditions, where students were asked to modify one of their portfolio questions before submitting it. This would help to an extent with overcoming potential issues relating to plagiarism, as a student would find it hard to modify a program that they had not fully written themselves.

The pass rate and average mark for students on this module has had a notable increase ever since this style of assessment was used (see Fig. 11.1), which somewhat proved our inclination that increasing engagement in the first term, where many fundamental principles are delivered, would help students in the module as a whole.

Whilst the assessment was clearly working, student feedback suggested they would appreciate a wider selection of their work to be marked. Due to the increasing number of students on the module and the general changes happening in Higher Education, as discussed in Sect. 11.2, this presented a challenge. Additionally, the British Computing Society (BCS) that accredits the associated courses updated their documentation with an emphasis on incorporating trustworthy software into the curriculum. One advised technique of delivery, as outlined in the PAS 754:2014 software trustworthiness specification (PAS 754:2014 2014), was to adopt unit testing. As such, it was identified that unit testing could be used to semi-automate the marking of the portfolio assessment, integrating good practice, and also allowing more work to be marked. The target was to maintain the existing successful assessment structure whilst gaining the obvious benefits of automating the marking process and the production feedback.

11.2 Motivation

Higher Education has seen a number of changes in recent years. Notably, the NSS and the new TEF initiative have put a greater focus on learning and teaching methods and associated activities such as assessment. The HEA study ‘A Marked Improvement’,

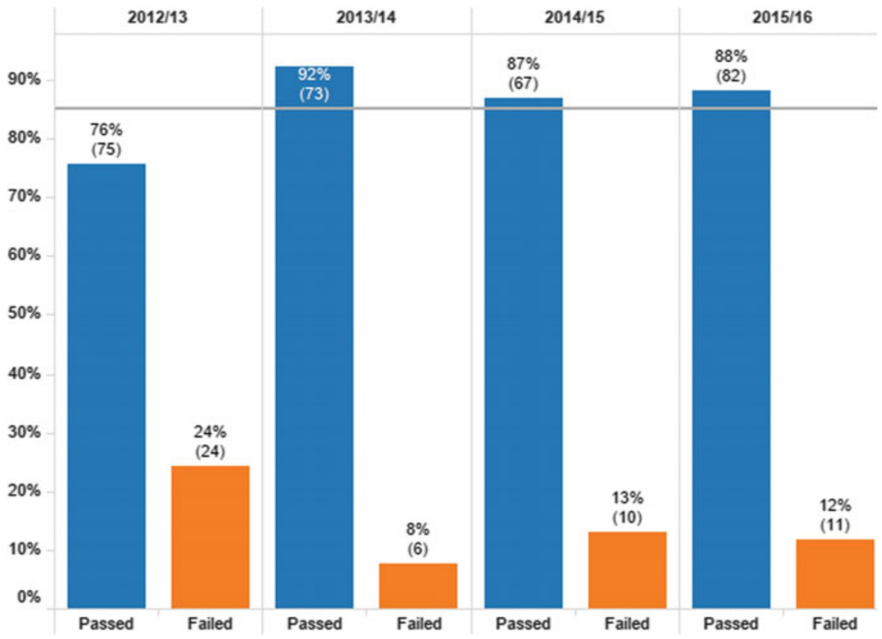


Fig. 11.1 Module pass rates and number of students (in parenthesis) over the past four years

(Ball et al. 2012) identifies a number of issues specifically related to assessment. It emphasises how fee paying students have different expectations from their courses and associated assessments. Large numbers of students and larger staff student ratios mean that assessments need to be marked quickly, consistently, and the marks need to have integrity. The HEA report suggests better assessment practise will result in: "..greater confidence in academic standards and improved safeguarding of the reputation of UK higher education" (Ball et al. 2012).

The greater number and wider range of students also means that a variety of learning styles are evident and assessments need to cater for this variability. At DMU this has been addressed by adopting principles of Universal Design for Learning (UDL). This is an approach that encourages the design of learning materials and assessments that suit different learning styles and do not therefore need to be adapted to suit special needs (Al-Azawei et al. 2016). The UDL approach is supported by the UK's Quality Assurance Agency (QAA), in which the quality code for higher education has a series of indicators that reflect sound practice. Indicator ten states: "Through inclusive design wherever possible, and through individual reasonable adjustments wherever required, assessment tasks provide every student with an equal opportunity to demonstrate their achievement" (QAA 2013).

The HEA report (Ball et al. 2012) states that "The increasing size of student cohorts and a shrinking unit of resource mean that tutor time has become disproportionately spent on summative assessment. Students can be taught in larger groups, but each assignment or exam script still requires individual attention." This suggests the use of more formative assessments but we can also be creative about how we set, mark and give feedback on summative work. The report goes on to discuss how the use of technology enhanced approaches can help to accommodate diverse needs as well as enabling automated and/or timely feedback. This view is supported in (Biggan 2010), which reports on a case study that uses automation to enhance the quality of feedback on assessments for a large module with 500 students. This article illustrates in particular that the marking was more consistent, the comments less superficial, as well as being timely. The findings were validated by both student and external examiner feedback on the project outcomes. The Jisc report (JISC 2016) 'Transforming Assessment and Feedback with Technology' similarly supports the use of technologically enhanced approaches for marking and feedback, ensuring: "the work of each student is marked fairly" and "consistency of approach across different cohorts of students."

The computer programming assignment in question at DMU was previously manually marked by tutors. The nature of checking code is such that it lends itself to automation, however, it also presents many challenges. The approach presented here provides a case study of a novel, largely automated means of marking and assessing all aspects of the portfolio assessment. The remaining sections consider how this was achieved, problems faced along the way, and the positive gains that have been made.

11.3 Semi-automating the Portfolio Assessment

Programming assignments that are automatically marked are not new and this opportunity has been tackled in a variety of ways, over a number of years, within the computing discipline (Douce et al. 2005; Ala-Mutka 2005). It is possible to develop or utilise existing web-based automated assessment tools (English 2006), which often entail free-form data entry for part of a program (English and English 2015). As mentioned at the end of Sect. 11.1, we specifically wanted to expose students to an industry standard testing framework, and have them develop entire Java classes showcasing they understood standard conventions and key design principles. Therefore we wanted specific control over what and how we tested. It was decided that the emphasis would be on having students use prebuilt test cases as a means of informing their development (Janzen and Saiedian 2005) and verifying their classes had been designed correctly. This would help to instil a further level of critical thinking as supported in (Rosen 2016). As the language being used was Java, it was decided that JUnit (also being used on a parallel module) would be most suitable. Furthermore, there were many existing cases of it being used effectively for automatic marking (Helmick 2007; Tremblay and Labonte 2003; Tremblay et al. 2008; Khalid 2013). JUnit is a testing framework for Java where individual *unit tests* are grouped inside of *test cases*, which can be executed individually or as part of a *test suite* that contains multiple test cases.

Table 11.1 Number of portfolio questions compared with those submitted for marking

Year	Portfolio questions	Java classes	Test cases	Submitted questions
15/16	6	6	N/A	1
16/17	6	2	2	3

One decision was which, and how many, questions should be assessed. Previously, there had been 6 independent questions from 3 different topic areas—aggregation, interfaces, and inheritance. Utilising this approach would require independent test cases to be written for each of these questions, a potentially onerous task. It was decided that a better approach would be to have two portfolio scenarios: A (a Register class) and B (a Player class), for which there are each 3 questions. Each scenario would provide students with an initial question along with a test case (containing all unit tests) for the associated Java class, and then 2 further questions would act as extensions to this. Table 11.1 shows the different structure between this year and last, clearly highlighting how more questions would be submitted and assessed than before. Although only 2 classes were now used, the same design principles as before were introduced across the 6 portfolio questions.

As in (Schmolitzky 2004), we also introduced the notion of Java *interfaces* at an early stage in the module but decided to not provide interfaces for the two classes linked to the portfolio work. Although this approach has been shown to be successful (Helmick 2007), we instead wanted students to interpret a UML specification in conjunction with the unit tests provided. As in previous years, the assessment criteria used were: 40% for their Java class design, 10% for a demonstration program, 20% for their documentation (i.e. javadoc), and 30% for their lab test modifications. The aim was to automate the marking of all but the javadoc (i.e. 80% of the assignment) by providing unit tests.

11.4 Constructing the Unit Tests

Normally it is considered good practice to have a single assertion per unit test (Aniche et al. 2013), however, in doing this a few issues became apparent. Some methods would require multiple unit tests, whilst others would only require one. Furthermore, some unit tests could pass without any code having been written. An example of this would be unit tests associated with testing that a Java equals method had been overridden correctly as the inherited version may pass one or more tests without the student having written any code. This issue can be overcome by using the Java *reflection* library (Forman and Forman 2004), which allows source code to be inspected at run time, however, this would result in having multiple asserts per unit test.

Having some unit tests passing prematurely may not normally be an issue, but the plan was to directly map the percentage of unit tests passed to the mark awarded for

Table 11.2 Overview of the test cases, classes, and unit tests in a typical portfolio scenario

Test case	Classes tested	Unit tests	Weighting (%)
RegisterTest	Register	15	40
RegisterDemoTest	RegisterDemo	1	10
RegisterSubTest	Register and RegisterSub	3	30

a given assessment criterion. Considering this, it also seemed most logical to avoid having a varying number of unit tests per method, as this would result in a higher weighting being attached to one method over another. Whilst it could be argued that some methods were more challenging than others, there was not a consistent ratio between this. It was therefore decided that the most sensible approach would be to write one unit test per method, and thus potentially have more than one assert statement per unit test. To overcome the ambiguity this may create, every assert statement had an associated unique message to ensure that students could clearly identify which assertion had caused a unit test to fail.

In previous years, students had been given credit for selecting the correct data types and access modifiers for their fields, e.g. private to ensure data encapsulation. Initially it was felt that this aspect of their design may have to be manually checked by the marker, but then it became apparent that the previously mentioned reflection library could be used to assess this. Whereas in other examples such as (Helmick 2007) reflection was used to assist the housekeeping of a custom-built testing framework, in this case it was used to assess the structure and quality of their code.

Whilst the main focus was on assessing students' ability to implement a class using a provided UML specification, the demonstration program had traditionally assessed that they could use their class, by creating object instances and invoking methods within the main method of a Java application. The issue was how this could be assessed with unit tests as whilst the main method could be executed by a unit test, it would be very difficult, if not impossible, to accurately assess if a student had used their class in a specific way. Instead, it was decided that a class containing a single static method would be used (e.g. RegisterDemo), similar in nature to one that students were used to, but instead with a string return type and a parameter list that accepted an object instance of the class being assessed.

In the past the 40 min lab test had involved students updating one of their portfolio questions under exam conditions. The goal was to also automate the marking of this component. A new test case (e.g. RegisterSubTest) was provided for this, which required them to further modify an existing class, and to extend it with a subclass. Table 11.2 shows a breakdown of the test cases for the Register portfolio scenario along with their associated number of unit tests and assessment weighting.

11.5 Avoiding Unintended and Hardcoded Solutions

When writing the unit tests it became apparent that it may be possible for students to pass a given unit test without following the intent of the lab exercise question. In particular, there was an issue with students being able to hardcode values initialised in their class, or returned from their methods, as they could see the expected result within the provided unit tests. One potential solution to this would be to have multiple asserts, each with different expected results, although theoretically this was still subject to a hardcoded attempt. A solution that was found, and used in some cases, was to take advantage of the assertSame method, which instead of assessing the state of two objects (as assertEquals does) assesses two object references for equality, making it impossible to hardcode a value set or returned.

Another example of a method subject to hardcoding was one that had to search a Register object by using a given input parameter. If different input parameters were used then the student could ensure their search returned true or false accordingly, without actually searching through the inner collection. Therefore, multiple asserts were used, each with the same input parameter, but with the Register containing different data sets.

When assessing inheritance, even if a subclass was created, a student could simply update the superclass method and not override it. To overcome this, the unit test invoked the method on instances of both the superclass and subclass to check the corresponding behaviour differed. Other issues were overcome by again using reflection to ensure the superclass's field remained private and the subclass did not have any fields. Without these checks it would have been possible to bypass the public interface of the superclass by either accessing the superclass's field directly or by working on a duplicate mirrored field in the subclass.

11.6 Marking and Feedback

Whilst a JUnit test suite could have been used to execute each test case, it was instead decided to create a test runner that would allow a finer level of control over how the results were summarised, and additionally provide a means of giving feedback for the students' javadoc comments. A method was defined that accepts a JUnit test case and an associated weighting (e.g. RegisterTest and 40), and then produces a statistical summary of the total number of unit tests passed and failed, and their mark for that component. If 14 of the 15 unit tests passed, a rounded mark of 37/40 would be shown. Using the org.junit.runner.Result class, a failure overview was then provided by iterating through any failures and recording details of these.

So that all feedback could be inside a single file, a rubric-like set of predefined comments were defined relating to the quality of javadoc. Tutors would simply need to assess students' documentation before executing the test runner, and then be prompted to select one of these. The mark and comment could then be merged with

Table 11.3 Comparison of this and last year's portfolio assessment

Year	Cohort	Mean (%)	STDEV (%)	Questions assessed	Marking time per student *
15/16	93	64	1	1	12 min 15 s
16/17	170	62	20	3	1 min 45 s

those associated with the unit tests and uploaded for students to view. This would usually be in excess of a page of A4, without a single comment having to be written.

There were a total of three tutors on the module, one of whom was inexperienced, and ensuring consistency between markers is usually a challenge. As 80% of this assignment was marked automatically, this was now largely a nonissue. The statistics presented in Table 11.3 show even though the cohort size has nearly doubled, and the number of questions marked increased, the marking time has been significantly reduced (* measured via a sample of 20 students with an experienced marker). The mean and standard deviation have remained in a similar band to previous years.

11.7 Conclusion

The evolution of the original portfolio assessment to incorporate unit testing has clearly been a success. Some obvious benefits include a significant reduction in marking time and guaranteed high consistency between markers, both of which are very important factors in the design of an assessment when student numbers and the number of markers are both increasing. This also allows the potential to assess students more frequently, giving them more regular feedback on their progress, and further helping them to engage with their studies.

There was a larger amount of source code assessed than in previous years, and this could very easily grow. As well as further exposing students to the importance of unit testing [as advocated by (PAS 754:2014 2014)], they also benefitted from ongoing feedback during their assignment based on the number of tests they had passed at any given time.

Many of the difficulties came in the construction of the unit tests, particularly in ensuring that tests cannot be passed in unforeseen or unintended ways. There were a variety of techniques deployed to overcome these hurdles, and the Java reflection library was particularly helpful. In the future, having different input data sets used for the unit tests the students are provided during their assessment, and those that are actually used to mark their work, may further help combat this. Ultimately a lot of the work involved in the construction of this assessment was frontloaded and the challenge now is to see how easily this approach could be adapted across different topics or even onto other software development modules.

References

Ala-Mutka K (2005) A survey of automated assessment approaches for programming assignments. Comput Sci Educ 15(2):83–102

Al-Azawei A, Serenelli F, Lundqvist K (2016) Universal Design for Learning (UDL): a content analysis of peer reviewed journals from 2012 to 2015. J Sch Teach Learn 16(3):39–56

Aniche M, Oliva GA, Gerosa MA (2013) What do the asserts in a unit test tell us about code quality? A study on open source and industrial projects. In: 17th European conference on software maintenance and reengineering (CSMR 2013)

Ball S, Bew C, Bloxham S, Brown S, Kleiman P, May H, McDowell L, Morris E, Orr S, Payne E, Price M, Rust C, Smith B, Waterfield J (2012) A marked improvement: transforming assessment in higher education, The HEA

Biggan J (2010) Using automated assessment feedback to enhance the quality of student learning in universities: a case study. In: Technology enhanced learning. Quality of teaching and educational reform. Communications in computer and information science, vol 73. Springer, Berlin, Heidelberg. JISC (2016) Transforming assessment and feedback with technology [online]

Douce C, Livingstone D, Orwell J (2005) Automatic test-based assessment of programming: a review. J Educ Resour Comput, 5(3)

English J (2006) The Checkpoint automated assessment system. In: Proceedings of e-learn 2006, pp 2780–2787

English J, English T (2015) Experiences of using automated assessment in computer science courses. J Inf Technol Educ: Innovations Pract 14:237–254

Forman IR, Forman N (2004) Java reflection in action. Manning Publications 2004

Helmick M (2007) Interface-based programming assignments and automatic grading of java programs. In: ITiCSE '07: proceedings of the 12th annual conference on innovation and technology in computer science education, ACM Press, Dundee

Janzen D, Saiedian H (2005) Test-driven development: concepts, taxonomy, and future direction. IEEE Comput 2005 38(9):43–50

JISC (2016) Transforming assessment and feedback with technology [online]

Khalid A (2013) Automatic assessment of Java code. The Maldives Natl J Res 1(1):7–32

PAS 754:2014 (2014) Software trustworthiness. Governance and management. Specification, British Standards Institution, May 2014

QAA (2013) UK quality code for higher education—Chapter B6: assessment of students and the recognition of prior learning

Rosen C (2016) A philosophical comparison of chinese and european models of computer science education (A discussion paper). In: Software engineering education going agile (CEISEE 2015), pp 9–14

Schmolitzky A (2004) "Objects first, interfaces next" or interfaces before inheritance. In: OOPSLA '04: companion to the 19th annual ACM SIGPLAN conference on object-oriented programming systems, languages, and applications, pp 64–67

Tremblay G, Labonte E (2003) Semi-automatic marking of java programs using junit. In: International conference on education and information systems: technologies and applications (EISTA '03), pp 42–47

Tremblay G, Guerin F, Pons A, Salah A (2008) Oto, a generic and extensible tool for marking programming assignments. Softw: Pract Experience 38(3), 307–333

Part III
Employability and Group Work

Chapter 12
The Enterprise Showcase Experience

Gary Allen and Mike Mavromihales

Abstract The School of Computing and Engineering at the University of Huddersfield have, for the last 2 years, been experimenting with an Enterprise Showcase Event, in which multidisciplinary teams of students work intensively on real-world projects for local companies, and then present their work at a day long trade-show style event. This paper outlines the rationale for the project, explains how the event operates, discusses the advantages of the approach over more traditional kinds of team project, and analyses the student feedback.

Keywords Group work · Team work · Enterprise showcase · Employability Multi-disciplinary · Collaboration

12.1 Introduction

The School of Computing and Engineering at the University of Huddersfield is made up of two departments, Computer Science and Engineering. Students in both departments engage in group work at various levels within their respective courses, but the students do not normally engage in collaborative, cross-departmental project work. The work undertaken is mainly group work in years 1 and 2, followed by an individual honours project in the final year. Students on an Integrated Masters degree (MEng, MComp, or MSci) also undertake a group project in their additional Integrated Masters year. There is a rich body of educational literature in support of group work, and it is widely accepted that group work offers many advantages to the students. In the computing discipline, group work is seen as good for interpersonal skills including team working, project management, and presentation skills, as well as for the development of technical skills such as requirements capture, system

G. Allen (✉) · M. Mavromihales
The University of Huddersfield, Huddersfield, UK
e-mail: g.allen@hud.ac.uk

M. Mavromihales
e-mail: m.mavromihales@hud.ac.uk

design, coding, testing and evaluation. In the engineering discipline, students are encouraged to undertake group work from an early part in their undergraduate studies through several modules. This helps them develop interpersonal skills which are greatly desirable by prospective future employers. Through discipline, organisation and effective communication students learn from each other in modules such as professional development, in which they research and present on a topical issue such as sustainable transport. In a mechanical design module students collaborate to develop a conceptual design prior to developing various aspects of it to a detailed stage. The Enterprise Showcase Event, for the mechanical and electrical engineering students, is linked to the intermediate, year 2 module of manufacturing and enterprise. For these students the significance of the Showcase Event is that, post event, the students are expected to complete a business plan which outlines vital information that is required in order to bring the product to market. By this stage they have already been made aware of the commercial constraints associated with product development by partaking in the Showcase Event. Although teamwork in undergraduate studies is not new, what is unusual in this case is the operation of multi-disciplinary teams. This has allowed for learning from fellow students with disparate areas of interest and expertise within the technology sector. This emulates exactly what they are likely to encounter in a real world working environment, as they gain an appreciation of the required expertise for a balanced product development.

Effective collaborative problem solving and its benefits has been documented by many educational researchers (Nelson 1999), as has learning by doing (Schank et al. 1999). Group work is also a required component of many of our courses, by the British Computer Society (BCS) (Projects and Group Work 2018; Guidelines on Course Accreditation 2018) for accreditation of Computing degrees, and by the Institute of Mechanical Engineers (I.Mech.E) and the Institute of Engineering and Technology (I.E.T) for Engineering courses, as well as by our industry partners who require graduates with soft skills and capable of working in teams. Both Engineering Institutions, who are the main governing bodies for our accredited engineering courses, apply the UK Standard for Professional Competence (UK-SPEC) (UK Standard for Professional Competence (UK-SPEC) 2018).

The UK-SPEC is based on the demonstration of key competences and is the UK Standard for Professional Engineering Competence. It describes the Science and Mathematics, Engineering Analysis, Design, and Engineering Practice competences in the economic, legal and social, ethical, and environmental context that have to be met in order to attain Engineer status at Technician, Incorporated or Chartered level.

Over the past two academic years the School has been experimenting with an Enterprise Showcase Event, whereby students from both departments have been placed into multidisciplinary teams and set to work on real world problems put to them by our industrial collaborators. The event takes place in an intensive one week block and culminates in a trade-show style Showcase Event where the students present their work to both academics and to the industrial collaborators. In this paper we outline the motivation for the Showcase Event (Sect. 12.2), explain how the event operates (Sect. 12.3), analyse the reaction and feedback from our students (Sect. 12.4), and outline several ideas for how we might improve the event in future years (Sect. 12.5).

12.2 Limitations of Existing Project Work

Current approaches to team based project work suffer from several limitations and weaknesses. Timetabling restrictions make it difficult or impossible to schedule interdisciplinary groups, so groups tend to be drawn from one course, one course suite, or at best one subject area. This means that the students are rarely, if ever, expected to work outside of their comfort zone in terms of working with students they don't know or working with students from different courses who bring a wide range of unfamiliar knowledge and skills. Additionally, when creating project ideas for students within a single discipline, the tendency is to come up with project ideas that fit the skill set of those students. This potentially stifles creativity and reduces the range of opportunities for learning afforded to these students. If we give the same problem to all students or groups within a module then there is little variation to keep the students interested and there is potential for collusion and plagiarism. If, on the other hand, we let students come up with the problems themselves then there tends to be a lot of variation in the level of complexity, making is difficult to keep the learning experience equitable and the marking fair. Students are unaware of what they don't know, so they tend not to consider projects outside their comfort zone. Similarly, students often have limited commercial and industrial experience, so are unable to easily consider realistic problems. Collectively considered, these issues often mean that the project work undertaken in team project modules is rather repetitive, unoriginal and unimaginative and does not provide the students with a realistic experience of working on commercial projects. An additional concern is that long time-scales (typically these modules are year-long, starting in September/October and due for submission in the following May) often implies that students postpone assignment completion until urgency hits, thus producing rushed and sub-standard work at a late stage. These projects do not reflect the "need it yesterday" urgency of many real commercial situations. Taken together these problems led us to reconsider how team-based project work, combined with activity based learning (ABL) and problem based learning (PBL) (Barrows 1985; Perrenet et al. 2000; Nkhoma et al. 2017), should operate, and led to the introduction of the Enterprise Showcase Event.

12.3 The Enterprise Showcase Event

12.3.1 Timetabling and Assessment of the Event

The University of Huddersfield introduced a "Consolidation Week" into the timetable several years ago. Consolidation Week is the first week after the Christmas vacation, when normal timetables are suspended to allow other activities to take place, such as in-class tests, catch-up and revision, and practical sessions before the start of term 2 teaching. The University has been keen to find interesting and innovative ways to utilise this time, and offered the opportunity to bid for small amounts of funding via the university's Teaching and Learning Institute (TALI) to support such projects. Our Enterprise Showcase Event has, for the past two years, taken advantage of this Consolidation Week. As normal classes do not operate in this week it gives us the opportunity to timetable events across subject areas or (as in this case) across the departments within the school. This allows us to create interdisciplinary teams made up of (almost[1]) all students in the Computer Science and Engineering departments. This includes students of Automotive Engineering, Electrical Engineering, Mechanical Engineering, Computing Science, Software Engineering, Computing, Information Systems, ICT, and Web Design and Technology, who can all be mixed together within the project teams.

All of the students involved in the Enterprise Showcase Event are already enrolled on year-long projects modules. This applies for all disciplines and provides a relatively simple way to build the assessment of the Enterprise Showcase Event into the curriculum. The event has been added as an additional assessment within those project-based modules (as a sub-element of the main in-course assessment). In the Computer Science department the students are all registered on module CII2350 Team Project. The Enterprise Showcase Event forms a sub-element of the module coursework assessment. Similarly, in the Engineering department the students are registered on either module NIM2220, known as Manufacture and Enterprise or NIE2208, known as Enterprise: Electronic Product Design and Manufacture. The former module is for undergraduate engineers studying either Mechanical, Automotive or Energy engineering. The latter module is for undergraduate Electrical or Electronic engineering students. All modules stipulate pre-requisite study that is more technically focussed. The modules associated with Enterprise focus on the commercial aspects of technological products and the soft skills required to develop these. Teamwork and inter-disciplinary awareness are therefore of the essence, which is why the Enterprise Showcase Event has been added as a sub-element of assessment within those modules.

For electrical and electronic engineering students the modules provide an introduction to business, finance, marketing, engineering management and design for manufacture (DFM) in the context of electronic product design and manufacturing.

[1]Students on the Computer Games Design and Computer Games Programming courses do not take part as they already have a range of activities taking place in Consolidation Week.

It is intended to promote an understanding of the lifecycle process of product design and develop the skills required by professional engineers to play an active role in that process.

12.3.2 Clients and Commercial Partners

One of the key objectives of the Enterprise Showcase Event has been to involve practitioner clients providing existing actual problems for the students to work on, in order to make the event as realistic as possible. The problems must have genuine commercial application, and the client companies are encouraged to work with the students with the best ideas after the Showcase Event in order to explore opportunities for further development and for commercial exploitation. The opportunity to develop a real commercial product as a result of the Showcase Event should act as a motivator both for the client companies and for the students themselves.

In the first year of operation one of our client companies, was a leading Health Care provider. They outlined the problems caused within the NHS and private health care by bed sores and ulcers resulting from prolonged periods of being bed-bound. The students were asked to think of innovative ways to minimise bed sores. Some groups focussed on high-technology solutions, such as sensors, monitors, and alarms; some turned to the use of novel materials or shapes and profiles for mattress design, or the use of springs within the mattress to keep a patient moving; some researched the impact of air-flow on bed sore formation and suggested fans or mattress covers designed to minimise moisture; some groups explored the role of Big Data and cloud storage in collecting as much information as possible via a range of sensors (such as temperature, pressure, humidity, and time intervals between the patient being moved) so as to build better models of the conditions that cause pressure ulcers; while some teams identified low-technology solutions such as simple timers to notify nursing staff that it is time to move the patient. The wide range of solutions suggested by the student groups impressed the company, who found several of the ideas worth further investigation.

One of the criticisms of the initial event was that by providing only one client with one problem we did not provide options to the students. In the second year it was therefore decided to seek several companies who could each pitch their problem to the students, thereby giving the students a choice of problems to work on, and making the event more appealing. Three projects were offered, two from local companies and one from the university's Students' Union. In summary these were:

- A local construction sector company presented a problem based on optimising the means by which roofing A-Frames are accommodated on a flatbed truck for delivery to a construction site, in a safe and efficient manner. Each set of A-Frame trusses is unique to the project to which it is intended. According to the company there is no off-the-shelf solution currently available for this problem. By failing to utilise the entire space on the truck, a second truck would be required to be booked through a haulier, at considerable cost;

- A local software house with a passion for encouraging children from schools in the surrounding area to get involved with STEM (Science, Technology, Engineering, and Maths) subjects, and in particular to get them writing computer code. The brief here was very general, to come up with ideas and associated products that could be used with the children to inspire them to get involved with STEM subjects, code clubs, and "hackathons";
- A student society from the Huddersfield University Students' Union called Enactus. Enactus is a global student movement aiming to "use the power of entrepreneurial action to transform lives and shape a better more sustainable world" (Enactus 2018). The Huddersfield Enactus group had taken ownership of a run-down greenhouse on an area of land owned by the university. They were looking for ways to utilise the greenhouse in a cost-efficient way to bring benefit to as many people as possible. Their pitch asked the students to identify low cost ways to support this objective, with a particular focus on energy efficiency and sustainability, whilst also ensuring a secure site.

All of these clients were able to provide opportunities for further work and for the potential commercial development of the ideas generated. The students were happy to be given the range of projects to choose from. All of the clients were available on the first day to allow the students to discuss the projects, and to carry out some requirements capture. This alone, is an important part of the Enterprise Showcase Event, as students working on projects do not normally get the opportunity to perform requirements capture from real clients, nor do they often get to work on projects with such uncertain and volatile requirements. This is an important part of the 'real-world' experience which the Enterprise Showcase Event is designed to provide.

We again received a wide range of suggestions for each of the project ideas. These included:

- Use of 3D modelling software to help with the A-Frame loading problem. Some teams found open source software that could be enhanced or tailored towards the specific problem, while others set about developing prototypes of bespoke software built to solve the problem. One team suggested an Artificial Intelligence (AI) solution based on machine learning;
- Many varied solutions were suggested for the STEM project. These included 3D printed cogs and wheels to allow students to experiment with basic engineering; software development platforms aimed specifically at younger children to allow drag-and-drop development of code; and 3D printed build-it-yourself car kits and associated mobile apps to control the finished car, allowing children to experience both the physical building of the car and the software side of controlling it;
- Greenhouse monitoring and control systems based around Raspberry Pi or Arduino devices, with attached sensors and motors to automatically maintain ideal

conditions through automated watering and opening and closing of vents; intruder detection systems; and remote monitoring coupled with mobile apps to allow the greenhouse to be operated remotely.

The clients were all impressed by the range and diversity of the suggestions, as well as the level of technical detail achieved by many of the teams.

12.3.3 Organisation and Operation of the Event

The Showcase Event is designed to run in a short, intensive time-scale of one working week. The students are placed into groups in advance of the start of the event. Groups are announced at the outset. The students arrange themselves into their groups, and the industrial partners then immediately pitch their problems to the teams. The industrial partners then remain on site during the first day (for at least half the day) to enable the students to ask questions, discuss the problem and any initial ideas they may have for potential solutions, and to carry out some requirements capture. Students then organise themselves in terms of tasks and priorities in order to meet the required outcomes by the end of the week. Initially they are expected to brainstorm ideas and potential solutions, carry out research in the university library or on-line, and to develop their ideas.

The first deadline the students are expected to meet is to submit an electronic copy of an A2 poster to showcase their ideas. This is required during the second half of the week. The poster should visually represent the group's idea or proposal, and will be displayed at the Showcase Event at the end of the week. Note that this is not a formal assessment point. If teams successfully submit by the deadline then the department will arrange and pay for the printing of the poster ready for the Showcase Event. Any team that fails to submit their poster by the deadline will be required to arrange and pay for printing themselves. This 'soft' deadline is in part intended as an encouragement to the students to start work as quickly as possible and make initial progress.

As well as working on the poster, the teams are also required to start work on their prototype product. For this we have available a range of kit that the students can sign out and borrow for use on the project. The kit includes Raspberry Pis, Arduinos, breadboards, a range of sensors such as temperature and humidity measures, motion sensors, cameras, indicators and LCD displays. 3D printing facilities are also available. When designing artefacts for 3D printing, the teams must ensure that the print time is limited to one hour maximum, and that their designs are suitable for the size of printer available. The necessary details are included in a briefing pack given to each team at the start of the week. There is also a small budget available, so the students can reclaim up to £20 per team if they need to purchase any specific equipment which we are unable to provide. For this receipts must be presented, along with a description of the use of the item. Only one member per team can claim back the expenses. The teams have the rest of the Wednesday and all day Thursday to work on their prototype product in preparation for the Showcase Event on the Friday. Students are therefore required to work to constraints and learn by doing (Schank et al. 1999). The learning benefits of undertaking problem-based tasks have been well documented (Barrows 1985).

To ensure fair access to support for all teams there are drop-in advice sessions scheduled throughout the week, where the students can consult academic staff. These sessions are supported by Computing and Engineering academics, who will help the students by providing advice and guidance, suggesting changes or improvements to an idea, directing the students to relevant resources, or whatever other help they may deem appropriate. The details of these sessions, which run for an hour per day on the Tuesday, Wednesday, and Thursday, are again included in a briefing pack provided to each team.

The Showcase Event itself takes place in a large room organised along the lines of a trade fair. Each team has a pre-allocated stand. The posters have been printed and are ready to be displayed. The students bring their prototype product and are given time to set up and prepare for the event. Academic staff and industrial partners visit stands, observing and scrutinising posters and prototypes, listen as the students pitch their ideas, question the students, and then give immediate verbal feedback. The Showcase Score Card used for the event is included in Appendix. The final mark awarded to each team is the average of the best three marks on the day. In the first year that the event operated each member of academic staff was asked to walk around and look at as many projects as possible. This did cause some problems, as the teams at the front of the room found it much easier to get three academics to mark their work than the groups towards the back. To improve this, in the second year of operation each academic was randomly allocated five teams that they were required to visit and assess, and were then encouraged to walk around and look at as many more teams as possible. In this way every team was guaranteed a visit by at least three academics in order to ensure the marking could be completed. Another change in the second iteration of the event was to ensure that the academics could not see the marks awarded earlier in the day by their colleagues. This was to ensure each mark was awarded entirely impartially, and staff were not swayed by the grades awarded by their colleagues. One of the interesting things to emerge was the way in which the teams were able to take on board the feedback given by the academic markers throughout the day to improve their pitch and therefore improve the marks received as the day progressed. Many teams commented on how they welcomed the immediate feedback and the opportunity to use this to refine their pitch (and in some cases the idea itself) so as to maximise their grade.

As with any group work, the students are encouraged to play to their strengths, and so to give team members the roles that they can best fulfil, be it development, documentation, or presenting the idea. Some teams do this well, and ensure that the idea is pitched in a positive light by a team member with good presentation and marketing skills. It can be a valuable lesson to some students to realise that a good idea described badly can accrue a lower grade than an average idea pitched in a positive and enthusiastic manner. However, as can be seen from Appendix A, the academic assessors are required to consider Research, Innovation, Manufacture/Execution, Marketing, Costing, Commercial Potential, and Teamwork. The feedback provided and the grades awarded should therefore summarise all of these areas, and should not be too heavily swayed by the presentation skills of the team members.

12.3.4 Outstanding Issues

As with any new idea or novel event there have been some issues which we need to address going forward. These include: communication of team allocations and other required information; dealing with students who fail to attend or contribute to their groups; the layout of the room for the Showcase Event; and ensuring the industrial partners know which teams are working on their projects. Here we briefly address each of these concerns:

- Communication: Some students complained that the team allocations were not made clear, or that students arriving just a few minutes late on the Monday morning were unsure which team they had been allocated to. We feel that it would not be appropriate to announce the team memberships before the Monday morning, as we want to ensure all teams have a similar experience, and we do not want students to get together in advance. Part of the idea is that the students are working with people they do not know, so we feel it is sensible to keep the team memberships secret until the event begins. We do, however, need to make sure the team allocations are complete before the event begins, and that the information is ready to be given out at the outset of the event;
- Students who don't engage: The overwhelming majority of students did participate in the event. However, as with any student activity, there are always a few who do not. This year we experimented with a reporting system to allow groups to notify the project coordinator of any students who failed to engage. The coordinator then contacted these students to inform them that, if they did not take part in the event, they would receive a mark of 0% for this component of their relevant module. This approach did help to bring a few students into the event, but did not address the issue of students attending but making little real contribution. In future we are likely to adopt the use of Peer Group Assessment, whereby the members of each team rank the contribution of their peers at the end of the project, as this is our established vehicle for managing group work. The perceived difficulty, and the reason we did not try this before, is that the interdisciplinary nature of the teams makes it difficult for the team members to get together after the Showcase Event has finished. We will therefore have to ensure that it is done on the Friday before the event closes;
- Layout of the room: We need to ensure that no team is 'hidden in a corner' or feels that their location at the Showcase Event puts them at a disadvantage when it comes to attracting academics to evaluate their work. This was much better the second time around, when academics were allocated teams to see, as it ensured that all teams saw a minimum of three assessors. There were also some issues

where teams needed access to mains power for their prototype devices, which led to power extension cables being laid across the floor. We need to think carefully about this in order to find an ideal layout for future years;

- Industrial Partners: The industrial partners attend the Showcase Event on the Friday to see the students' work, discuss it, and provide feedback. This year, with three client companies involved, we realised too late that we were unable to tell each company which teams had elected to work on their particular problem. The representatives of the companies therefore had to walk around and identify relevant teams for themselves. This had not been a problem the year before, as we only had one company involved. Next year we will use the Poster Submission on the Wednesday to collect data about which team is working on which project, so that we can then provide a list of relevant teams to each company.

12.4 Student Feedback

Feedback from participants was sought in the form of comments after the event. There was therefore no formal feedback questionnaire to qualify learners' own perceived learning outcomes. The feedback comments were therefore based entirely on learner comments reporting on the experience of participation and engagement. A selection of comments relating to the activity were as follows:

'Very good idea with the enterprise project, given a chance to apply practical skills and improve presentational skills. Nice range of work, offers more information and practical applications than most other modules'.

'Working with students from other courses was a necessary and valuable experience. Support, when questions were asked and answers were required, has been excellent'.

'Facilities are very good and the enterprise activity was a good experience especially in working with people from other disciplines'.

'Provided a positive experience in working within a team-based structure to research, design and develop a product'.

'I had a good team that really wanted to deliver something great to the rest of the class and enjoyed helping create this'.

'I really enjoyed working with other student[s] from different courses and to me it was more interesting than the other assignment in the module'.

Overall, the comments that were received from students were positive and favoured the practical aspect and group work. There were however a number of negative comments which were received by students who could not see the relevance of the practical aspect of the industrial partner's problems to their particular course and discipline—typically such comments came from web design students. Some students were also critical of the structure, organisation and what they considered as erratic scoring depending on assessor. The negative comments were mostly received during the first year of delivery of the Enterprise Showcase Event. We took on board the comments regarding structure and organisation and made improvements for the second event delivery, including the choice of projects. The scoring of posters and prototype solutions by various tutors can by nature be erratic, as what constitutes a good and viable solution to a practical problem can be a subjective matter. It is for that reason that the average of the best three scores is determined and allocated as the final score.

12.5 Conclusions and Further Work

Our experiences in the first two years of operation of the Enterprise Showcase Event have been highly positive. We believe we have identified benefits to our students, benefits to our industrial collaborators, and benefits to the school. We have also identified some areas in need of further work and improvement.

12.5.1 Benefits to Students

The students are given the chance to work in interdisciplinary teams, learn new skills, and experience 'real world' short-term intensive project work. This is particularly useful for their CVs and for discussion at interviews. Several of the students have reported back to us that, during employer interviews for placement jobs, they have been able to draw upon their experiences in the Enterprise Showcase Event to provide examples of working to tight deadlines, working in teams with people they did not previously know, and working on 'real' projects. In several cases the students believe that this has been key to their securing an offer of a placement job as they were able to confidently discuss their role and contribution as part of a team in order to help solve a practical problem. The experience has therefore provided a range of opportunities for the students to impress potential placement or graduate employers. The event also gives the students the opportunity to make new friends and create contacts outside of their usual peer group, and simply to learn a little about the focus and content of courses from across the School of Computing and Engineering. Overall the experience has provided many positive benefits to the students who have taken part.

12.5.2 Benefits to Our Industrial Collaborators

There are also many benefits to the industrial collaborators. These include the opportunity to receive original input to a problem or business need, including the potential to take this further and develop a new product or solution in collaboration with the students involved; the chance to see and meet potential (placement or graduate) employees; and the chance to get involved with the university, which could then lead to a range of other opportunities for collaboration, including student placements, Knowledge Transfer Partnerships, or an invitation to become more actively involved with our Industry Panel, affording the opportunity to contribute to course and curriculum development. To date the industrial partners have all expressed satisfaction with the event, and have found it to have been a worthwhile experience.

12.5.3 Benefits to the School

There are also a range of benefits to the School and to the Departments therein. These include opportunities to build or enhance links with local companies, which could in turn lead to a range of other opportunities for collaboration as mentioned above; the chance to collect and document useful and novel experiences for discussion at professional body validation visits, university subject reviews, quality audits, etc.; and the chance to provide students with opportunities not usually available to them. We can then include this information in marketing materials, and for discussion at open days and applicant visit days. The opportunity to engage in ‘real’ projects has been popular with applicants and with their parents and guardians as examples of such activities are often discussed during applicant visit days. The overhead in terms of running the event is relatively low compared to the range of potential benefits to be accrued.

12.5.4 Areas to Improve

While we, our students, and our industrial collaborators all feel that the events to date have been highly successful, we are aware of a number of areas that could be improved. There is the need to address peer assessment, to ensure the marking is fair and that the students who make the greatest contribution to the projects do receive the recognition they deserve. We must also ensure the layout of the room is clear, that there are no advantages or disadvantages to individual groups based on their location within the room, and that power can be supplied safely to the groups that require it. We must also ensure that our industrial collaborators are able to easily identify and locate those teams which are working on their particular problem. We do feel, however, that none of these issues are particularly complex, and we are confident that we can improve the event year on year.

Acknowledgements Many thanks to our colleague Dr. Leigh Fleming, who came up with the original idea for the Showcase Event and who worked very hard to make it so successful.

Appendix: The Showcase Score Card

Showcase Score Card

Your mark will be based on the assessment of 3 judges, if you are able to attract more than three academic judges to your stand, the average of the top 3 marks will be calculated and taken as your assessment mark.

Feedback on this element of assessment will be verbal during the showcase event.

Judges will be looking for the following attributes:

Research—Background to the project and demonstration of understanding of the problem
Innovation—Your innovative and creative approach to solving the problem
Manufacture/Execution—How well the project has been carried out by the team
Communication—How effectively you communicate your ideas through the poster and orally
Marketing—How well you sell your idea
Costing—Demonstration of an appreciation for the cost and methods of production
Commercial potential—Potential for a commercial market and consideration of pricing
Teamwork—Demonstration of good team-working and overall effort

Print Name	Signature	Mark (--/100)
Comments		

When a judge has completed their scorecard, please place in the envelope provided to ensure that subsequent judging remains impartial.

References

Barrows HS (1985) How to design a problem based curriculum for the preclinical years. Springer Publishing Co, New York

Enactus—Dedicated to making the world a better place through entrepreneurial action http://enactus.org/. Accessed 19 May 2018

Guidelines on course accreditation https://www.bcs.org/upload/pdf/2018-guidelines.pdf. Accessed 19 May 2018

Nkhoma MZ, Lam TK, Sriratanaviriyakul N, Richardson J, Kam B, Hung Lau K (2017) Unpacking the revised Bloom's taxonomy: developing case-based learning activities. Education + Training 59(3):250–264. https://doi-org.libaccess.hud.ac.uk/10.1108/ET-03-2016-0061. Accessed 26 June 2018

Nelson L (1999) Collaborative problem-solving. In: Reigeluth CM (ed) Instructiona design theories and models: a new paradigm of instructional theory. Lawrence Erlbaum Associates, Hillsdale, pp 241–267

Perrenet JC, Bouhuijs PAJ, Smits JGMM (2000) The suitability of problem-based learning for engineering education: theory and practice. Teach High Educ 5(3):345–358. https://doi.org/10.1080/713699144

Projects and group work https://www.bcs.org/content/ConWebDoc/57834. Accessed 19/05/2018

Schank RC, Berman TR, Macpherson KA (1999) Learning by doing. In: Reigeluth CM (ed) Instructional design theories and models: a new paradigm of instructional theory. Lawrence Erlbaum Associates, Hillsdale, pp 161–179

UK Standard for Professional Competence (UK-SPEC). Engineering Council website http://www.engc.org.uk/engcdocuments/internet/Website/UK-SPEC%20third%20edition%20%281%29.pdf. Accessed 12 June 2018

Chapter 13 Task Versus Process: A Taxonomy for Group Projects

Clive Rosen

Abstract Group projects appear to be a mandatory requirement of undergraduate degree programmes in software engineering and other, similar courses. But what are these projects for, and what do we expect students to learn from them? Often the group project exists within the curriculum simply because it is required by various authorities and without a clear pedagogic rationale. The language used in module specifications usually refers to employability skills and collaborative working, but often disguises a wide range of different delivery approaches. Assessment regimes may prioritise outputs over the process of producing those outputs. Even the word "process" is ambiguous. Do we mean the systems development process or the group process? If the former, why do we need a group? If the latter, how can this be assessed to the satisfaction of students and external examiners? The "Task Versus Process" taxonomy firstly suggests a model for categorising the aims and objectives of the group project, secondly, evinces an appropriate assessment strategy and finally suggests some approaches that can be taken if staff teams wish to address the "Personal Development" quadrant of the taxonomy. The chapter contextualise the student group project within broader aspects of curriculum design and then attempts to clarify the multifarious and often contradictory objectives proposed for student group projects. Fully appreciating what it is we, as instructors hope to achieve in group projects takes us beyond resigning ourselves to the compulsory inclusion of the project. Projects offer the opportunity to help our students achieve their full potential and to actually deliver the learning outcomes.

Keywords Learning and teaching strategy · Student group projects · Group project assessment · Teamwork · Peer assessment

C. Rosen (✉)
Passerelle Systems Limited, Newcastle Under Lyme, UK
e-mail: clivecrosen@outlook.com

J. Carter et al. (eds.), *Higher Education Computer Science*,
https://doi.org/10.1007/978-3-319-98590-9_13

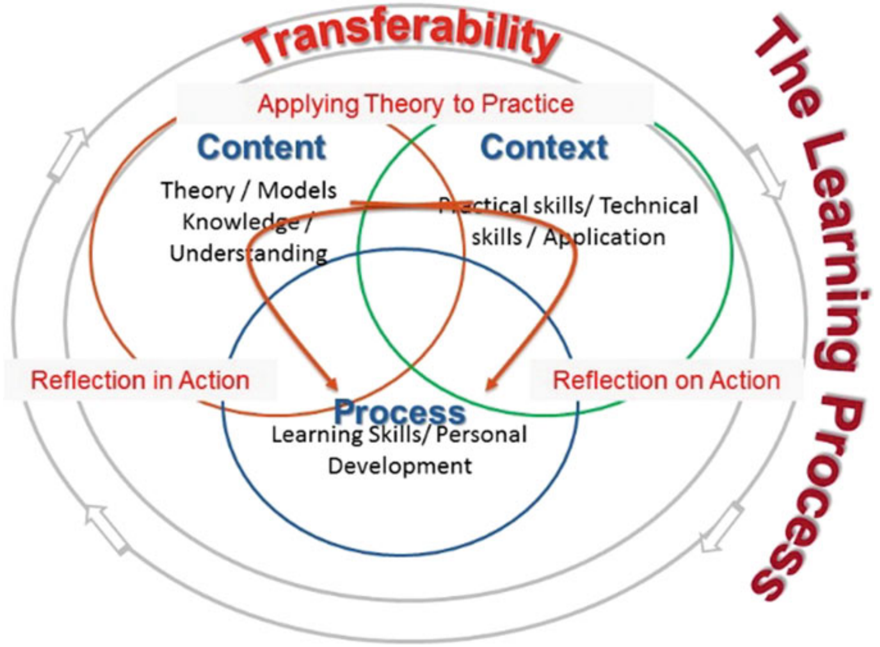


Fig. 13.1 The learning process in higher education (Rosen and Schofield 2011; Rosen 2015)

13.1 Introduction

What is the purpose of student group projects in software engineering courses? Their ubiquity suggests that they must be a very important component in such courses. However, when one explores the range of learning outcomes associated with group project modules, one finds a wide range of potentially conflicting objectives and often assessment approaches that are not wholly compatible with the learning outcomes. This chapter explores the reasons why this might occur and proposes a taxonomy that associates the actual objectives of the group project with the assessment approach taken. The purpose of the group project is situated within the more general model of the learning process (Fig. 13.1) in order to plot clear connective patterns from programme learning outcomes, through module learning outcomes to appropriate assessment methodologies. It is hoped that this approach will enlighten discourse on programme design in IT and computing related courses.

Group project module learning outcomes often cite such employability related requirements as the ability to work in a team, decision making or effective communication. These nebulous terms are often treated as an inexorable consequence of the group project process, avoiding the fact that learning about oneself requires self-reflection and challenging preconceived personal constructs. Students need to be presented with opportunities to learn and also be given support for self-reflection.

Computing students in particular are often reluctant to engage with this sort of process. The group project module provides the opportunity to integrate employability skills into the curriculum. This chapter offers some suggestions of how this might be done. However, if it were that easy, everybody would be doing it. Resources play a part, but there is also a challenge for staff. We cannot expect students to reflect on themselves without having gained some self-awareness and self-reflectiveness ourselves. Many would not see this as part of their lecturing role. It is hoped that some of the suggestions made in this chapter will help to overcome this reluctance on the part of both students and staff.

13.2 The Purpose of Student Group Projects

The demands made on student group projects are often extensive. Projects are often expected to deliver a wide range of outcomes including “real world” experience of industrial weight project work, employability skills, team-working and communication skills. Group projects are also viewed as capstone modules, integrating knowledge from other parts of the curriculum (such as database development, web sites and programming). This range of expectations can lead to students becoming confused regarding the purpose of the module and what is expected of them. It can also result in a situation where the mechanisms used for assessment either fail to test the actual learning outcomes, fail to challenge students to produce their best work, or fail to provide sufficient opportunity for students to exercise appropriate critical self-awareness.

Before considering the role of the student group project, it might be helpful to stand back a little and consider how higher education relates to governmental and industrial expectations of software engineering students and how the group project sits within this context and the software engineering curriculum. Figure 13.1 above helps to illustrate this positioning.

This model identifies three intersecting circles representing models of student learning which can support the development of a learning and teaching strategy at both the curriculum or module level. The three inner circles represent “content”, “context” and “process”. These circles intersect in a “Venn” style diagram suggesting the lack of precise boundaries between the three forms of learning.

“Content” is comprised of the theories pertinent to an academic domain, abstract knowledge associated with it and accumulated experience concerning the subject area. It is abstracted from any given situation. Content can normally be found in text books, journal papers etc. and is often delivered in HE settings through formal classes. Content is the domain that academia traditionally focused on delivering and, more particularly, developing, through on-going research. So an example of content in IT would be normalisation.

“Context” is the application of a particular theory to a given context and requires practical/implementation skills. Vocational subjects generally give more emphasis in this sphere than the humanities, as, being able to deliver is an essential element of success. Context is situated in real world problems and encompasses knowledge and experience associated with practical problem solving. When employers complain about the lack of readiness of graduates to work, it is often issues associated with this sphere that they complain about.

The association of context with competency contrasts with the association of content with understanding. An example of the difference between the two within the computing domain would be the capability of a student to program in a particular language (context) compared to an understanding of the principles of programming languages (content). In reality, both are needed. Students struggle to understand programming principles without knowing how to solve problems in a given language, yet they must understand the principles to be able to learn the range of languages they may be confronted with when they leave university. Most courses contain elements of both content and context. Examples and case studies are often used in teaching for just this purpose. Without context/application, abstract knowledge is interesting, but essentially unproductive.

Whilst both understanding and technical competence are considered to be essential requirements of the higher education system, neither are sufficient either singularly nor together. An implicit demand in recent years, particularly from industry (but arguably often undermined by governmental policies) has been the ability of graduates to transfer knowledge from one context to another. This requires the third sphere, "process".

Students need to develop much more than knowledge and skill; they need the ability to learn independently; to use their knowledge and understanding to solve unfamiliar problems; to understand the processes and standards involved in learning; to recognise which tools and techniques are applicable to a particular context plus other social and learning skills). Students need the problem solving skills of logical reasoning, deduction, research and critical evaluation. These capabilities do not automatically emerge from content and context, but need to be nurtured and encouraged in equal measure through reflective practice. Schön's (1991) concepts of "reflection in action" and "reflection on action" can be applied here to support the learning process. "Reflection in action" is the idea that (in Schön's phrase) the "Reflective Practitioner" (in this case the student) does not simply do work, but thinks about it whilst they are engaged in it to ensure that any particular course of action is the best available at the time. "Reflection on action" is the idea that once a particular project is complete a professional should not give a huge sigh of relief and move on to the next project, but should review the project to see what can be learnt from it. Were there any mistakes made? Could the output have been more effective? Were any inefficiencies introduced into the process? (The parallels between this and the Shewhart Cycle (1986) and Deming (2000) quality improvement process should not be overlooked.) Traditionally, one might argue, that higher education has used research as a vehicle to enculturate students into these activities, but the group project represents an ideal opportunity to encourage students to become reflective, particularly if ongoing progress reviews are included in the module implementation. However, reflection, particularly reflection on one's own performance, involves the ability to be self-critical which necessitates a level of self-awareness and self-confidence; qualities that are often limited amongst computing students.

In practice content, context and process are not independent of each other. Teaching staff switch between them in rapid order, drawing on case studies to expound theory, asking students what they think might be going wrong (theorising) in a practical application and asking them to consider alternative approaches to the problem they are trying to solve. However, this model does offer an idealised conceptualisation of what we might be trying to achieve. This objective for Higher Education may be aspirational and idealistic, but it is expressed in Benchmark statements for all the computing subject areas. And, the group project provides curriculum designers with the opportunity to demonstrate the development of these skills in the programme. This, it might be suggested, is often one of the sources of conflict within the group project module. For how can we help develop self-reflection in students and how do we assess it (one might say particularly in computing students who are often resistant to the notion that self-awareness, personal reflection and personal development are necessary to becoming software or systems engineers)?

Student group projects rarely include new knowledge and skills. They are more likely to focus on applying the skills that students have already acquired in other modules to a particular situation (or "context" in the diagram above) and often in a more complex environment. But the learning outcomes for the group project module will often include "the ability to work in a team", "real world understanding", "creative thinking" and so on. These skills are "Process" skills as defined by the model above. However, students will usually be expected to be able to demonstrate that they have actually understood what they have done. They might be expected to explain why they did something one way and not the other. In other words, to use critical analysis to show that they know why they did something. This activity is situated in the "Content" part of the model above. So, when student group projects are considered in the light of the above model, they sit very firmly at the nexus of the diagram. It is unsurprising therefore that lecturers may feel pulled in several different directions at once. For any particular group project module, should the emphasis be on the real world experience, reasoned decision making, team working of something else? The consideration of "Task Versus Process" may help to articulate the answers to some of these questions by informing the module design process and helping to make decisions more explicit.

13.3 Task Versus Process

The "Task Versus Process Grid" provides a means of considering the primary purpose of the group project in a particular programme. The grid offers two parametric dimensions. On the vertical axis we have the question of, "for this module, on this programme, do we consider students' capability to complete the task more or less important than their understanding of how to undertake the task (i.e. the process)"? However, process can have two different meanings here. The first meaning suggests that the process we are considering is the development process. A second interpretation of process is the team process. Are we more concerned about the way the system is produced or an understanding of the group dynamics and hence the ultimate capability of the student to work in a real industrial team? The second axis helps to clarify this difference. If the former, we would be more concerned with the outputs from the students' activity. If the latter we would pay more attention to the students' ability to think about the way the group has operated, and how the intra-group process has affected the conduct of the task.

The four quadrants therefore provide us with a taxonomy to consider where we wish to place the emphasis for the module. If we are concerned with a student's technical capability, this would place us in the lower left hand quadrant of the model. We would want the student to demonstrate their ability to produce some outputs (probably in the form of a database, website, program or app) and we would concentrate on the knowledge associated with "doing" the task. Alternatively, we may be concerned less with the actually getting the task done than with how the students went about the task. Did they establish the requirements? Did they produce a product design? How rigorous was their testing? These are questions regarding the governance of the project and place us firmly in the "software engineering" quadrant.

If the purpose of the project is to promote employability skills, we would be concerned with how students worked together to produce their outputs. The artefacts themselves would be less important than the students' ability to reflect on what worked, what did not, and how they communicated with each other to arrive at appropriate and timely decisions. What did the students learn from their experiences?

The final quadrant, the "personal development" quadrant is perhaps the most ephemeral and also the least well-articulated quadrant. It may also be the quadrant most aspired to when curriculum designers construct courses, and perhaps the hardest to achieve. Questions relating to students' personal development include how did the group process support or prevent you from contributing to your maximum ability? Were you able to contribute to decision making? Were you able to reflect in and on action (Schön 1991)? These questions can be the most difficult on which to engage students, and the most difficult to assess. Teaching staff are probably least qualified to support student learning in this area and, perhaps therefore least willing to focus on this aspect of the group project.

Figures 13.2 and 13.3 summarises this discussion.

13.4 Assessment

The argument above would be incomplete without a discussion of how to assess team projects within each quadrant as, if each focusses on particular forms of output, each should use those outputs as the means of assessment. Figure 13.4 below outlines the means of assessment that would be best associated with each quadrant.

It can be seen that the range and variety of potential assessments is extensive. Learning outcomes defined in module specifications are often ambiguous and lack

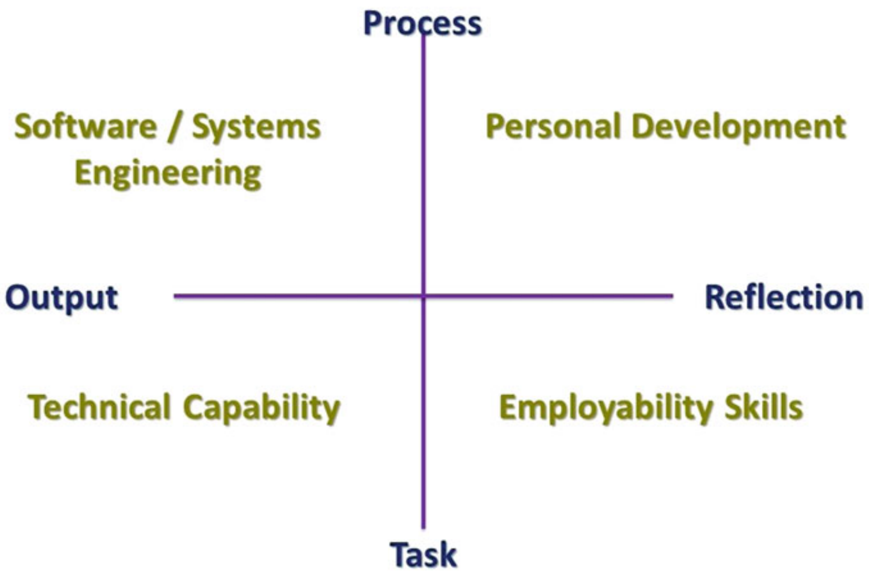


Fig. 13.2 Task versus process grid

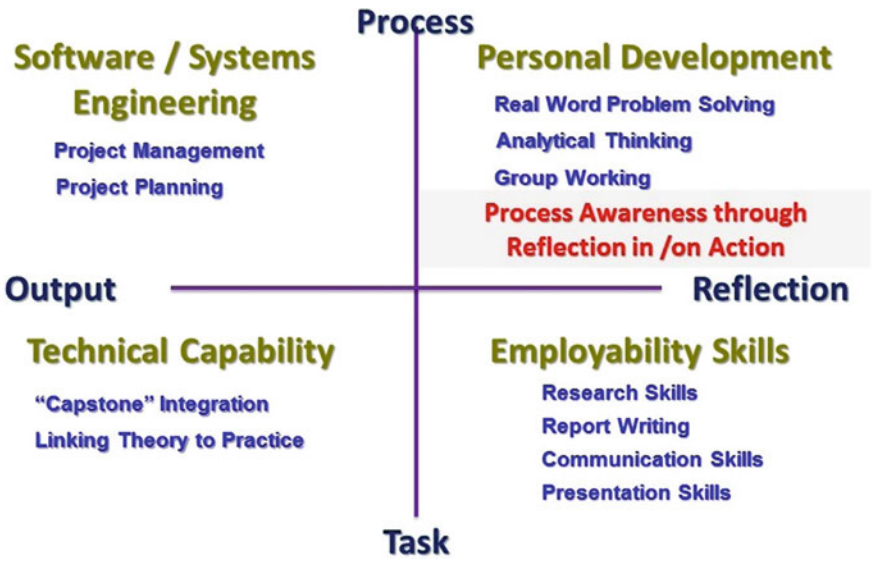


Fig. 13.3 Activities most associated with each quadrant of the task versus process grid

precision, leaving it to the module leader to design her/his own assessments. The breadth of scope however means that it is not possible to assess every outcome which is why having more clearly defined learning outcomes becomes important (notwithstanding the point that if the module team are not clear what they are trying to achieve, the students are likely to be confused). Some may argue with the details of the taxonomy presented here, but this analysis clearly demonstrates the difficulties associated with assessing group projects. Whilst the "Software Engineering", "Technical Capability" and "Employability Skills" quadrants all have relatively self-evident outputs that can be assessed, the problem is deciding on what to focus.

The Personal Development quadrant is more problematic for many reasons. It has already been suggested that computing students often fail to appreciate the importance of the more esoteric skills resident in this quadrant. Often staff feel reluctant to stray into an area they might feel unqualified to assess. The subjectivity associated with assessing personal development is also problematic in a discipline aspiring to achieve engineering status. Furthermore, many staff have developed an antipathy to assessing personal development from the experience of trying to assess "key skills" in HND, "core skills" in apprenticeship schemes and "employability skills" as part of the employability agenda. One lesson many have learnt from these experiences is that if employability skills are not fully integrated into the curriculum in general, and the group project module in particular, students react poorly to their add-on, ad hoc injection into a programme. This raises the question of how can seamless integration be achieved in such a practical subject area such as computing. The next section of this chapter suggests some approaches that can be used as part of the group project. Other chapters in this volume look at other approaches.

13.5 Implementation

This section will concentrate on the introduction and assessment of the skills associated with the upper right, "Personal Development" quadrant of the Task Versus Process Grid (Figs. 13.2, 13.3 and 13.4). Staff may feel more confident implementing some suggestions than others. Logistics and resources also play a part as do the peculiar academic environment surrounding all group projects. With this in mind, staff may choose to introduce changes over a period of time if they feel any of these ideas are appropriate to the pedagogic context in which they are working.

13.5.1 Live Client Projects

A point of dispute amongst group project module leaders is whether or not to trawl for real client projects, allow students to construct their own projects or for the module team to design projects the students can do as a group. The choice may well depend on the particular circumstances in which the project is being conducted and the objectives of the project. The Task Versus Process taxonomy above can help with this decision. However, often the choice boils down to more pragmatic considerations. Do the module team have sufficient time to find and vet potential projects? Can we trust

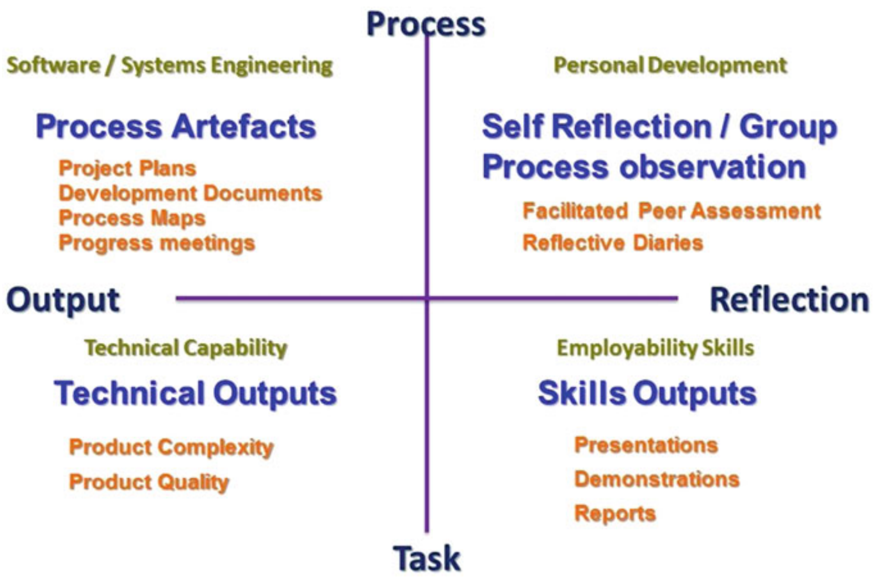


Fig. 13.4 Assessment associated with each quadrant in the task versus process grid

students to deliver for real clients? What mechanisms are available for installation on client systems and ongoing support? Real projects tend to be much messier and more complex than contrived projects. This means that students need to be given much more hands on support during the project; particularly with client management. But this is also the strength of using real projects and real clients. Students discover that part of systems development is client management; working with clients to identify their actual requirements, agreeing with them what is practical and what can be delivered given the time and resource constraints. Clients do not conform to the stereotypes implied in software engineering texts. Working with real clients exposes students to these realities and provides many powerful learning opportunities.

Furthermore, students engaged on real projects experience one of the fundamental differences between employment and student life which often goes unacknowledged; that work is undertaken for an employer, not for oneself (excepting self-employment). Studying is a very self-centred occupation and feeds into many students' egocentricity. Employment implies doing work for other people (even when self-employed) and this involves a radically different mind-set. Real, live projects can help with this transition in a relatively safe environment, at least one where the module staff can provide a safety net if necessary.

One further factor when considering whether live projects should be used, especially when external (to the institution) clients are adopted, is the role of the student as ambassador to the college or university. This can be a very positive experience for students, and the institution, if students embrace the responsibility, but can cause difficulties for the programme staff if things go wrong. Teaching staff must decide for themselves the balance of risk and benefit. Client preparation is essential and staff cannot absolve themselves from continuous client management, particularly when working with external clients. There is undoubtedly less work for the teaching team when contrived projects are undertaken, but they are potentially less rewarding for both students and staff. Group projects present this opportunity. There is rarely a shortage of potential clients. Only the module team can assess the cost/benefit ratio in their environment.

13.5.2 Student Application to Chosen Project

One approach that can help to integrate employability skills with the group project is to require students to make a formal application to a particular project. This works particularly well when external clients are offering projects. The assessment would require students to produce a CV, write a letter of application to the project provider and attend an interview for the project. (Students would only get their first choice project if they pass the interview.) Careers staff and project providers can be recruited to help with the interview process so the experience mirrors more closely a genuine recruitment exercise. An assessment centre can also be included in the process if time and resources permit.

One essential element of this activity is immediate feedback to students from the interview. It is remarkable the learning that can occur even in this relatively contrived situation when a student has underprepared for the interview and then received feedback from an interview panel on their performance. A large majority of students will never have previously experienced a formal interview and the reality and immediacy of such a meaningful interview is a very powerful learning experience. It is important for the interview panel to set the right tone. Hostile interrogation could generate a student's defence mechanisms which would not be helpful, a potential danger when a student fails to take the process seriously or fails to prepare for the interview. Having the external client on the interview panel brings a number of benefits. Not only does it make the experience more real, it also increases the client engagement in the project process. At the same time, being interviewed by a real client promotes greater student commitment to the group project and therefore improves the prospects of their full engagement.

A commonly recognised problem with the assessment of student group project is the evaluation of the student free-rider, i.e. a student who contributes little to the project, but expects to be awarded equal marks to those that have worked on the project. Facilitated Peer Assessment, see below, helps to alleviate this problem, but interviewing students for the project in the first place, also helps student engagement by increasing their initial investment in the project.

13.5.3 Skills Inventory

The skills inventory presented here is a development from a skills matrix that was presented to the HEA BMAF Placements workshop in 2006. The inventory has evolved to its current version since that date. Six principles have emerged during this evolution:

1. To provide a supportive structure for student reflection.
2. To minimise the psychic threat to students.
3. To encourage students to identify specific examples to work on.
4. To focus on learning from personal experience.
5. To make the process of self-reflection intrinsic to the exercise.
6. To use the skills employers say they want as the vehicle for self-evaluation.

As can be seen in Table 13.1, each row represents one of the skills employers consider to be important. The student is asked to identify a specific example of when they have demonstrated each skill. If they have more than one example, they can insert additional rows under the same heading. The "activity description" column requires the students to articulate the specific circumstances. The "when" and "where" columns are included to ensure a particular time and place, and not a generalised activity. In the "what did you do" column, the word to emphasise is "you". Students often experience these activities as part of a group, and it is important that they learn to differentiate themselves as an individual rather than simply a member of a group so that they can attribute the outcome to their personal contribution. Students are often reluctant to appreciate what they do well, and this exercise helps them identify strengths as well as learning experiences. The seventh column, the "learning" column is perhaps the most important. It is these entries where the ideas of self-reflection and self-evaluation are expressed. Having the experience alone is insufficient. Professional development requires personal reflection on what has been learnt from the experience. To experienced professionals this may become integral to their professional practice, but to students and new entrants into the profession, the concept of continuous self-evaluation and reflection are not self-evident and often students are never told that it is required. This exercise makes explicit an activity experts often take for granted.

The final column is optional. It is there to help students develop an action plan and to decide on which skills should be priorities for future development.

The inventory can be used as a standalone exercise, but has greater relevance if students are preparing to apply for placement or post graduate opportunities. UK employers commonly adopt a competency based approach to interviewing applicants (sometimes known as CAR, Context, Action, Result). This takes the form of asking questions such as 'describe a situation when … what did you do … what was the outcome?' The skills inventory helps prepare students for such questions and therefore improves their chances of performing well in interviews.

Table 13.1 Skills inventory

Skill	Activity description	When	Where	What did you do?	Outcome	Learning	Priority
Leadership/Initiative taking							
Influencing/Negotiating							
Team work							
Effective communication							
Self-motivation							
Decision making							
Planning/Organisation							
Working under pressure							
Personal development							
Commercial awareness							
Presentation/Report writing							

13.5.4 Facilitated Peer Assessment

Facilitated Peer Assessment (FPA) was first described in Rosen (1996) as a means of assessing the individual contribution to group projects, but it is also a means of supporting student reflection in that students are required to assess their own performance in relation to other members of the team and to give feedback to other team members. As such, it provides a vehicle for exploring self-reflection both as part of the project and as part of the assessment process.

FPA has the form of a post project review in which students in a face to face group meeting with a module tutor evaluate the success or otherwise of the project. As a group, they must also reach a consensus on the allocation of individual members' contributions to the project. The role of the staff member is to facilitate the discussion rather than to influence it (other than to encourage the students to reach a conclusion).

FPA has several advantages over other forms of peer assessment which usually consist of students completing undisclosed assessment forms about each other. The lack of transparency with this traditional approach undermines the value of peer assessment. Students can attribute any contribution level to their fellow team members, but it is unclear what criteria or perceived equity of treatment they are applying. Group projects can provide excellent opportunities to explore unconscious bias in teams, but undisclosed assessments of each other fail to challenge these biases in the assessment method. As a result, it is often down to the module team to moderate the students' assessments. It can be argued that moderation of marks is the responsibility of the module team, but if the module team are going to moderate the peer assessment, students may question the reliability, and possibly the validity of the process. Furthermore, academics' moderation of the peer assessment process undermines the students' sense of responsibility for the process as they cannot take full ownership of the process.

FPA avoids these problems. Firstly it is open and transparent. Students know what is being said about them by other team members, both positives and negatives. FPA therefore has the capability of being a very affirming activity as well as a functional one. Students retain responsibility for the process as the role of the staff member is facilitation, not moderation. FPA also provides the opportunity for students to reflect on the assessment process and the legitimacy of various criteria. They must decide whether the team's project manager has contributed more or less than its chief designer. What is the nature of contribution? This has the effect of requiring students to reflect on their own performance, as does the receipt of feedback from other team members. FPA does challenge the assertiveness of less articulate students, but it is the role of the facilitator to support students to express themselves (one of the key requirements for employability).

Student ownership of the FPA process has an important side effect. Group project module leaders will be familiar with the concept of the "free riding" student, the one that hopes to benefit from the work other students do without contributing much themselves. Often the only (meaningful) sanction the staff team have is to sack such students from the team. This is very much a nuclear option as it results in the dismissed student failing the module and staff have to have very clear evidence that a particular student has not, and will not, contribute. In group dynamics terms, a student's lack of engagement in the project can be very complex. Gender, race, culture and language can all play a role in peer acceptance or rejection within the team, and an apparently non-contributing student may feel that they are being excluded by the group rather than the student not wanting to contribute. Students have responsibility and authority for FPA so they have greater responsibility for the successful functioning of the group. FPA gives the students themselves a sanction against non-contributing members; one that can be referred to during the project to bring pressure to bear on a student and one that avoids the nuclear option. The result is that contributing students feel more in control, and often respond positively to the responsibility they have for project progress. The nuclear option is still available, but if it does need to be invoked, there is usually a clearer evidence trail to support it.

FPA is problematic as it does depend on the confidence of academic staff to manage the FPA process. Smouldering conflict that has been present within the team can, and sometimes does emerge during the FPA. Staff may feel that managing such conflict is not one of their responsibilities, but if this tension has resulted in the team not achieving its full potential, staff should at least be aware of it, and it must surely be beneficial for students to explore how such tensions affect the progress of a development project. After all they are likely to face these sorts of situations during their working life and if they have reflected on it within the relatively safe and supported group project environment, they will be better prepared for it in the world of work.

Managing conflict in group projects is challenging for staff. It is probably the aspect of student group projects academics find most off putting, and probably one of the reasons why many staff, when charged with the responsibility of running group projects, avoid the "Personal Development" quadrant of the "Task Versus Process" grid. Yet the ability for graduates to be able to work successfully with other team members often marks them out as potential employees, and in a profession where such capability is a rare commodity (Teague 1998; Capretz 2003), suggests that the effort may produce its own rewards.

13.5.5 Team Membership Selection Process

The team membership selection process is somewhat orthogonal to the taxonomy of group projects as any team selection method could be used whichever quadrant the project is focussed on. However it is nevertheless a bone of contention and teaching staff need to be cognisant of the influence the selection process can have on the group dynamics. In group theory, group formation is one of the most significant events in the life of a group, and the method of its formation reverberates through the group process in one form or another.

The debate over the team selection process revolves around whether or not students should be allowed to select their own groups or whether team members should be selected by staff members. The arguments over which is best have been well rehearsed and are not going to be recapitulated here. However, two options are available that may be of interest that somewhat sidestep this problem and provide an opportunity to encourage student reflection.

The first is to make use of an early tutorial session to discuss with the students the method of selection to be used. This involves brainstorming the various ways in which groups can be selected. Students usually identify between 20 and 40 different possibilities ranging from the obvious, that they (the students) should select their own groups or the teacher announces the teams, to geographic proximity to where students live and the physical height of the students. The class then go on to discuss the pros and cons of each method, eliminating some for practical or ethical reasons and looking at the more legitimate approaches critically. This discussion often raises some ethical and moral questions such as 'what happens to the student no one wants in their group?', 'what about the new student who nobody knows?' or 'what happens when friends fall out?'. After this discussion, the students then decide which method to adopt. It has to be said that students usually adopt a self-selection method, though not always. However this approach does place the responsibility of the selection method with the students. They are therefore unable to claim that if the project goes sour it was the teachers fault. Students do seem to like this discursive approach and it is nearly always enlightening, both for students and academic staff.

The second approach that has been alluded to above is the 'project first' approach. This is where the projects are defined first and students then choose the project they want to work on. This approach need not be combined with the interviewing process described above or include external clients, but it does mean that fairly clear specifications of what the project involves need to be provided to the students. Again this approach places the responsibility for the team selection with the students, albeit indirectly in this case. The project first approach subtly changes the initial dynamic of the module emphasising the importance of the project task over being a bit of fun with mates at the end of the year.

13.6 Conclusions

This chapter aims to develop a pedagogic taxonomy for determining the focus of student group projects in the context of the objectives of any given software engineering course. It identifies four quadrants that might become the focus of a group project module on that programme, the “Software Engineering” quadrant, the “Technical Capability” quadrant, the “Employability Skills” quadrant and the “Personal Development” quadrant. For each quadrant, it identifies the type of activities associated with the quadrant and the assessment artefacts that might be used to assess student capability. In providing this mapping, it is hoped that coherence and consistency between assessment methodology and learning outcomes can be maintained, and that, as a result, staff may be better able to articulate their own aims for the module and prepare students for the assessments they might expect. Congruence between learning outcomes and assessment methodology seems to be taken for granted in other modules within a software engineering curriculum. Because there can be such a wide variety of both overt and covert objectives for group project modules, it seems important to explicitly resolve any ambiguities that might exist regarding what the group project module in a given programme is really for. It is hoped that this taxonomy will help in this regard.

The “Personal Development” quadrant in particular can be quite challenging. It takes academic computing staff into unfamiliar territory and areas which it might be argued are not their responsibility. However, group projects are often cited as being key components of the employability agenda, and UK universities and colleges have signed up to this agenda, however reluctantly in some cases. In reality, it is not easy to differentiate elements of personal development from employability in general. Team working for example always involves managing personal differences and sometimes even conflict. If we are genuinely to address such aspects of employability, we probably need to provide students with the opportunities to reflect on, and learn from appropriate experience. The group project module can be such an opportunity. Some suggestions have been proposed in this chapter which can be employed to help to stimulate student personal development. Many may still wish to steer clear of any involvement in this aspect of the current higher education agenda. Whichever approach is adopted, it is hoped that the taxonomy presented here provides a framework for decision making.

References

Capretz LF (2003) Personality types in software engineering. Int J Hum Comput Stud 58(2):207–214
Deming WE (2000) Out of the crisis. MIT Press, Cambridge
Rosen C (1996) Individual assessment of group projects in software engineering—a facilitated peer assessment approach. In: 9th Annual Conference on Software Engineering Education, 1996. IEEE Computer Soc. Press, Daytona Beach
Rosen C (2015) A Philosophical comparison of Chinese and European models of Computer Science education. (A Discussion Paper) 11th CHIINA—Europe International Symposium on Software Engineering Education, Zwickau, Germany, April 2015
Rosen C, Schofield R (2011) Reliability and validity in work-based learning. Work Based Learning Futures V, Derby
Schön D (1991) The reflective practitioner: how professionals think in action. Aldershot, Arena
Shewhart WA (1986) Statistical method from the viewpoint of quality control. Dover, New York
Teague J (1998) Personality type, career preference and implications for computer science recruitment and teaching, ACSE98, Brisbane, Australia

Chapter 14
Realising the Threshold of Employability in Higher Education

Chris Procter and Vicki Harvey

Abstract A substantial body of work has tested and developed ‘Threshold Concepts’. A Threshold Concept may be considered “akin to a portal, opening up a new and previously inaccessible way of thinking about something … it represents a transformed way of understanding, or interpreting, or viewing … without which the learner cannot progress” (Meyer and Land in Threshold concepts and troublesome knowledge 1—Linkages to ways of thinking and practising in improving student learning—Ten years on. OCSLD, Oxford, 2003). Little research however exists on the relevance of the concept to employability. Employability is fundamental to Higher Education, yet its role in the curriculum is unclear and contested. Our practice suggests that developing knowledge about employability is a threshold which, when reached, empowers and gives confidence to the student. To achieve this means embedding this knowledge in the curriculum. The paper discusses the delivery of a large module with this aim, explaining how the design of assessment was fundamental in guiding students to a transformed way of understanding employability.

Keywords Employability · Threshold concepts · Competencies · Professional development

14.1 Introduction

Since the mid 1960s there has been significant change which has paved the way for the development of the current employability agenda in Higher Education (HE) in the United Kingdom (UK). The Robbins report of 1963 (Barr 2014) heralded the creation of many new Higher Education Institutions (HEIs) to ensure that all who were qualified and wished to enter should be able do so (Barr 2014). Figures from

C. Procter (✉) · V. Harvey
University of Salford, Salford, UK
e-mail: c.t.procter@salford.ac.uk

V. Harvey
e-mail: v.m.harvey@salford.ac.uk

the Higher Education Statistics Agency (HESA 2017a) indicate that there have since been consistent increases in student enrolments. With the number of first year, first degree student enrolments standing at 542,575 for 2015/16, this shows a 3% increase year on year since 2006/07.

The UK has the second highest graduation rate in the OECD, with around 47% of school leavers entering HE, but such growth is not replicated in the graduate labour market (The Guardian 2016; CIPD 2015). Government figures suggest that 31% of all graduates are not doing graduate—or high-skilled—jobs (BIS 2016a). Employers report (Archer and Davison 2008; Woods and Dennis 2009) that graduates are not work ready and do not have the requisite competencies. Despite a substantial number of under-employed graduates, employers continuously report a skills shortage (CIPD 2014; CBI 2015), and the role of HE in facilitating this gap is continually called into question (Cranmer 2006). Successive UK governments have responded by steadily increasing the pressure on HEIs to demonstrate the employability of their graduates. When considering Computing graduates specifically, The Tech Partnership report (Matthews 2017) emphasises the gap between student and employer expectations and the need for Universities to develop employability skills. The review conducted by the Higher Education Funding Council for England (HEFCE) led by Nigel Shadbolt (HEFCE 2017) recommended a much greater emphasis on employability and proposed integrating ‘work ready’ skills as an accredited part of the curriculum.

Following a brief explanation of the evolution of policy, this paper discusses how employability is defined as a set of competencies by employers. It then addresses how best HEIs can help students to demonstrate their employability arguing that this is best achieved by initiatives embedded within the curriculum. We explain the relevance of theory concerning Threshold Concepts and how assessment can be used as the lever to enable students to realise what we term the threshold of employability. The paper explains the design and implementation of a large multi-disciplinary module undertaken by undergraduates. Whilst they were largely on business related programmes (including IT) the approach is equally valid for computing students. Student reflection and feedback is used to demonstrate the relevance of our approach. Conclusions are drawn concerning the value of embedding employability in the curriculum to help students conquer their ‘Monsters of doubt’ (Hawkins and Edwards 2013).

14.2 Development of Government Policy on Employability

For over two decades successive governments have sought to intervene to ensure that HEIs addressed the gap between graduate capability and the requirements of the labour market (Artess et al. 2017). The Dearing report (NCIHE 1997) provided a major impetus for UK HEIs to become engaged in employability skills development. There has been a continued policy emphasis on the strong relationship between HE and economic prosperity, and consequent need for the production of ‘employable’ graduates (BIS 2016b).

Since the turn of the century policy measures have sought to inculcate employability into programmes of study by the use of national metrics. The Destination of Leavers from Higher Education (DLHE) survey, which commenced in 2003, has been conducted with all former undergraduate students six months after their graduation. It allows for comparison between HEIs in relation to the quality of career/employment destinations post-graduation. The data from this is used as a key measure in assessing the performance of HEIs in league tables. More recent initiatives in the UK linking the achievement of such measures to student funding (i.e. the Teaching Excellence Framework, TEF) are designed to further force the hand of University management in delivering employability. It is worth noting however that any data applied for such measurements needs to be placed in context, as data collection methodologies are reviewed to meet the scrutiny of a wider range of users. As such the next data set for DLHE, implemented as the Graduate Outcomes collection will not be published until January 2020 (HESA 2017b). Importantly what remains is that Artess et al. (2017) suggest that employability has now become one of the top priorities for HEIs.

14.3 Defining Employability: Understanding Competencies

Understanding the employer approach to employability is fundamental to addressing the issue. Professional bodies have a substantial influence over the curriculum in traditional vocational degrees, such as law or nursing, and the importance of the curriculum in defining employability, knowledge acquisition and training for a career is widely accepted. Many other programmes, such as those in Computing, whilst they are vocational are not training individuals for specific jobs, do aim to facilitate the possession of a mix of employability skills and business knowledge desired by employers. However, today's school children will likely go on to graduate from University and in future work in organisations and job roles that don't yet exist. Upheaval and fluidity in the job market mean that employability is increasingly defined by one's possession of, and ability to articulate, a given set of competencies, alongside requisite technical skills (e.g. Independent 2015; Times Higher 2015).

The focus on competencies in the job application process has become ever stronger in the 21st century: students who can demonstrate these competencies have a substantial advantage in seeking graduate jobs and future promotion.

The Higher Education Academy (HEA 2015) identified a composite list of 34 terms associated with graduate attributes, suggesting that employability is difficult to define, and being able to demonstrate competency is also rather slippery, described as "*a personal state that individuals occupy*" (Artess et al. 2017, p 10). More usefully, Dubois (1998) defines competencies as those characteristics—knowledge, skills, mind-sets and thought patterns—that, when used whether singularly or in various combinations, result in successful performance. There are numerous lists of the competencies most in demand (Diamond et al. 2011): these typically include teamwork, communication & networking, leadership, business awareness, initiative, flexibility, enthusiasm, personality and many others.

14.4 The Importance of Articulating Competencies

Therefore, the challenge in HE is not just to teach these competencies but to help students realise their possession and develop the ability to articulate them. Students, however, may not fully recognise the importance of engaging with employability while studying, beyond managing their part-time work (Tymon 2013; Greenbank 2015). Tomlinson, in his Review of Graduate Employability (2012), suggests that many do not appreciate the competencies sought by employers, nor have the knowledge to articulate these, despite this articulation being the most critical component of a typical recruitment process. He adds that it is not just about individuals possessing certain competencies which enhance their employability, but being able to package this for employers:

> Brown and Hesketh's (2004) research has clearly shown that …for graduates, the challenge is being able to package their employability in the form of a dynamic narrative that captures their wider achievements, and which conveys the appropriate personal and social credentials desired by employers. (Tomlinson, p 420)

Tomlinson et al. (2017) have developed this further arguing the need for HEIs to develop a range of 'capitals' with the student:

> Capitals can be understood as key resources, accumulated through graduates' educational, social and initial employment experiences, and which equip them favourably when transitioning to the job market. (Tomlinson, p 17)

Such 'capitals' not only include intellectual capital but social capital as well.

If, as suggested, employability for recruiters rests upon an individual candidate's ability to demonstrate their competencies, this has great significance for HEIs developing employability. The challenge therefore is not whether or not HE should seek to develop competencies, but *how and in what ways this could be done*, taking into account some students' historical lack of engagement in employability initiatives.

14.5 How to Deliver Employability?

All Universities have traditionally encouraged their students to consider their future employment and provided support for this, managed via an extra-curricular Careers Service. Typically, they organise employer visits, recruitment fairs and offer careers information, advice and guidance. However, optional engagement with such activities is not sufficient to reach all students and ensure graduates have appropriate employability knowledge and understanding. Thus, many institutions look to the curriculum to see whether and how they can embed employability. This has been given recent impetus in the UK by the support of the Higher Education Academy (HEA) which published a framework for employability in tertiary education (Cole and Tibby 2013). The HEA's report on Pedagogy for Employability (Pegg et al. 2012) includes many case studies on embedding employability in the curriculum. None of these however, explicitly link course assessment to the articulation of student competencies in the way described later in this paper.

14.6 Should Employability Be Developed Within the Curriculum?

Cranmer (2006) conducted research amongst a group of UK Universities, looking at different models of delivery, and concluded that there was limited evidence of the development of ‘employability skills’ through classroom teaching. She argued that resources would be more usefully deployed by HEIs in employment based training, employer involvement in the curriculum and opportunities for experience with employers such as placements/internships. Her conclusions suggested that teaching employability skills was not achievable or (even) desirable. Tymon (2013, p 853), acknowledges the complexity of employability and further questioned whether the development of employability was within the capability of HE institutions:

> It is also unclear whether many of these skills and attributes can be developed in practice and, if so, what the role of higher education institutions should be. Putting aside the arguments about whether higher education institutions are able, willing or designed to develop employability, there is evidence to suggest there are alternative options which may be more appropriate.

Tymon (2013, p 853) goes on to declare that with improvement from HEI’s:

> Skills can be developed and are embedded in the curriculum, but many first and second year students appear to lack engagement with these activities. This must reduce their motivation to learn and inevitably impact on successful development.

Tymon concludes that “development of these [employability skills] is possibly outside the capability and remit of higher education institutions.”

14.7 Making Employability in the Curriculum Work

There is a lack of evidence of the efficacy of seeking to ‘teach’ or ‘develop’ the competencies sought by employers. Such efforts can appear to clash with the priority of the curriculum, rightly focused on the subject content of the degree. Our own initiative arose from reflection on the failure of: a) previous efforts to prescribe a Progress File as part of an earlier Personal Development Planning (PDP) initiative and b) extra-curricular employability initiatives to engage the majority of students. In terms of this study, the demographics of the student population were a key driver behind the embedded nature of the module described below. The University has a relatively high proportion of students from Widening Participation backgrounds, currently standing at 42% of the population (University of Salford 2017).

This paper suggests that while it may not be possible to instruct students in employability, we can guide them to construct their employability on the basis of their prior experience and knowledge. This knowledge comes from their learning both within and outside the curriculum. For this to be effective, students not only need to appreciate the importance of employability in general but, most importantly, they need to cross the threshold between *their* understanding *and* that of employers.

14.8 Threshold Concepts and Employability

Meyer and Land developed the theory of Threshold Concepts in 2003 following a research project into the characteristics of effective undergraduate education, particularly in the field of economics. Akin to a portal, achieving a threshold opened up a "new and previously inaccessible way of thinking about something … it represents a transformed way of understanding, or interpreting, or viewing … without which the learner cannot progress" (Meyer and Land 2003). Subsequent investigation has shown that the central tenet of mastery of a subject via Threshold Concepts could be applied to any subject, demonstrating the broader applicability of Meyer and Land's original findings. In particular, work by Cousin (2010) has demonstrated the significance of Threshold Concepts in developing pedagogy as well as facilitating subject specific knowledge. A good example to illustrate new understanding would be a shift from a student of French to a French speaker (Cousin 2010).

This understanding, considered so important by their tutors, is both transformative and irreversible and is not (we would argue) discipline specific. Threshold Concepts have non-subject specific features in common. Their significance in HE has been explored more fully in a more recent collection of work published in 2016 (Land, Meyer and Flanagan 2016). Flanagan (2017) has summarised the features of Threshold Concepts, in Table 14.1.

The concept of liminality resulted from additional research by Meyer and Land, published in 2006. Liminality involves the active engagement of the learner, as this threshold is crossed back and forth as the student experiences both positive and unsettling shifts in comprehension. Cousin (2010) compares this idea to the age of adolescence. In learning it can involve a period of understanding and misunderstanding at the same time: the experience can be very emotional. The first experience of a job interview is a good example: the experience can be very emotional involving a combination of understanding and misunderstanding. Effective performance in face to face interviews and other employer engagement (for example assessment centres and online video interviews) require a clear understanding of the employer perspective which, when achieved, is a threshold of employability. Achieving a clear understanding of employability meets the description of a Threshold Concept well.

Burch et al. (2014), in their paper on 'Identifying and overcoming threshold concepts and conceptions', stress the importance and difficulty of changing curriculum design and delivery in order to apply the theory. Cousin (2010) argues that a teaching strategy informed by the practical application of Threshold Concepts allows

Table 14.1 Summary of threshold Concepts. Adapted from Flanagan (2017)

Threshold feature	Impact characteristics
Transformative	Once understood, a threshold concept changes the way in which the student views the discipline
Troublesome	Threshold Concepts are likely to be troublesome for the student. Perkins (1999) has suggested that knowledge can be troublesome e.g. when it is counter-intuitive, alien or seemingly incoherent
Irreversible	Given their transformative potential, Threshold Concepts are also likely to be irreversible, i.e. they are difficult to unlearn
Integrative	Threshold Concepts, once learned, are likely to bring together different aspects of the subject that previously did not appear, to the student, to be related
Bounded	A Threshold Concept will probably delineate a particular conceptual space, serving a specific and limited purpose
Discursive	Meye, Land and Davies (2006) suggest that the crossing of a threshold will incorporate an enhanced and extended use of language
Reconstitutive	Understanding a threshold concept may entail a shift in learner subjectivity
Liminality	Comparing the crossing of the pedagogic threshold to a rite of passage, involving a potentially messy journey to learning. Liminality requires active engagement of the learner, as this threshold is crossed back and forth as the student experiences both positive and unsettling shifts in comprehension

academics to steer a path between teaching-centred and student-centred education, making it appealing across all disciplines. In practice this involves a change of focus from the teaching to the learning, with a particular focus on how the student can develop their learning on the basis of their prior experience.

14.9 Constructing Employability

In considering our best approach to guide the students towards the threshold of employability, the solution devised follows a constructivist approach to learning and teaching: our approach can be characterised by a saying attributed to Plutarch: “the mind is not a vessel to be filled, but a fire to be lit”. Boud et al. (1985) work on ‘Reflection: turning experience into learning’ was valuable. This advocates the use of assessment and reflection in motivating learning. Wiggins and McTighe (1998) also argue in their book ‘Understanding by Design’ for the idea of deciding first upon the outcome required and then designing the assessment accordingly. Boud suggests further that the teaching and assessment process needs to enable students to draw on their previous experience and knowledge so that they can ‘take some significant responsibility for their own learning over and above responding to instruction’ (Boud

1988, p 32). As Villar and Albertin (2010) suggest, students need to become more actively involved and responsible for their education, investing in their own social capital. Providing students with a better understanding of how to do this and opportunities to participate in student-driven activities can develop and/or demonstrate proactive personality in a practical way. These ideas have become well established in the work on Assessment for Learning (Sambell et al. 2013), promoting the idea of using assessment to promote learning rather than measure it. We were also influenced by arguments concerning the significance of authentic assessment, a term popularised in the paper with that title by Fook and Sidu (2010).

14.10 The Patchwork Approach to Assessment

We adopted a patchwork approach to assessment as a mechanism to force the involvement discussed above. This is similar to an assessment approach possible more widely known as scaffolding. Winter's paper 'Contextualising the Patchwork Text: Addressing problems of coursework assessment in Higher Education' (2003) explains the patchwork approach as follows. Academic staff define the module assessment as a sequence of tasks. The tasks themselves are a process of development designed to guide students to construct their own learning, whilst assimilating new ideas within their existing experience. The tasks are both analytical and experiential.

Winter (2003) likens this to bricolage, where the student is encouraged to improvise for each task according to the social, material and experiential resources they have to hand. In this case students are given quick feedback, and social feedback amongst their peers is encouraged.

Students are asked to synthesise the patchwork through self-reflection at the end. In arguing the value of this, Winter (2003) cites Barnett; "Only in that moment of self-reflection can any real state of intellectual freedom be attained … Only through becoming a continuing 'reflective practitioner' can the student … gain a measure of personal integrity." This offers the opportunity, as Moon (1999) suggests, for engagement in personal or self-development in addition to gaining insights and empowerment. This final review and interpretation thus also holds the possibility of students demonstrating achievement of a threshold of learning about employability.

14.11 Professional Development Module

We now discuss how this was put into action in a large 20 credit module (one sixth or a year of study) undertaken by 600s year undergraduates on 11 different degree programmes.

A patchwork of assessment (Winter 2003) was designed to achieve student enquiry into their own employability; assessments replicated typical recruitment processes. Figure 14.1 illustrates the assessment tasks of the module which closely follow pro-

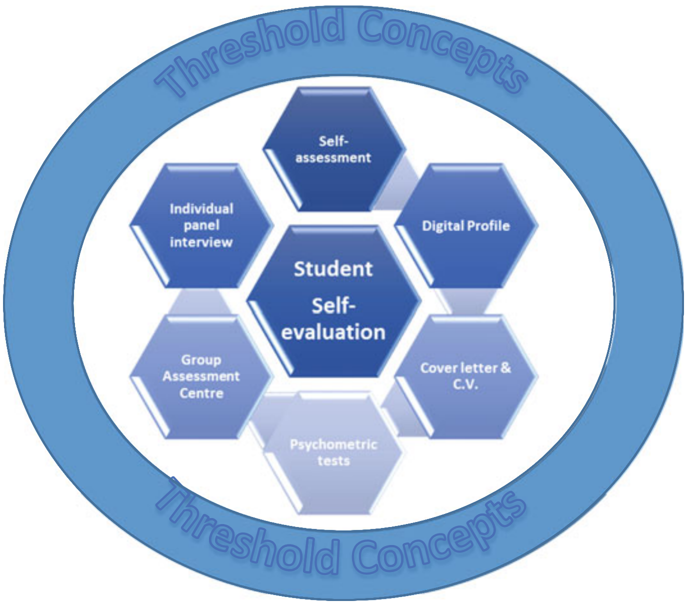


Fig. 14.1 Professional development patchwork assessment

cesses used by employers in an application process. Academic and careers staff together with employers were involved at every stage—significant evidence is available of the benefits of such partnerships (e.g. O'Leary 2017).

These were initiated by an 'as advertised' job description for which students were set a mixture of tasks, commencing with a self-evaluation presentation to their class. They had to judge their stronger and weaker competencies in relation to the job description, drawing upon a wide range of experiences. They were required to give evidence of their strengths using the Situation, Task, Action, Result (STAR) approach, and an action plan to address areas of improvement. Individual feedback was given and (where necessary) support provided.

Students then created a digital profile using LinkedIn. Unlike other forms of social media, the focus here is upon a professional outlook that illustrates the students' current and future employability, highlighting their skills, experience, extra-curricular activity, as well as a summary of their education.

Students then submitted a CV and covering letter for the job description provided. Evaluation of this work focussed on the quality and relevance of the application to the criteria specified, rather than the calibre of the student achievement.

All students were required to complete a series of industry standard Psychometric tests. These tests were treated as a learning opportunity to be reflected upon in the final assessment.

Students from many degree programmes were randomly mixed for an assessment centre which was conducted together with a company partner, aiming to be as authentic as possible, simulating standard practice in recruitment as used by approximately 90% of medium to large employers (AGR 2016). Each student was allocated to a group to tackle a business problem followed by a team presentation. The following competencies: leadership, teamwork and communication were assessed.

By once again using the 'as advertised' relevant job description provided, each student undertook a panel interview. The preparation of relevant answers and positive engagement was key for this element. The process was conducted as realistically as possible with appearance, punctuality and body language forming part of the assessment criteria.

At the end students were required to synthesise and reflect upon their experience and discuss how they would continue to develop their employability and articulate their competencies. The potential to reach the Threshold Concepts for employability permeated the whole development process. Importantly students were asked to reflect *not* upon the module delivery but rather their own experience of professional development. This could be a critical incident, or perhaps consequence of their performance, that they considered influential to their future development.

This sequence is illustrated in Fig. 14.1, commencing with the self- assessment, moving clockwise around the hexagon to the interview and then concluding with the student self-evaluation:

Every part of the assessment followed the cycle:

(1) Introduction in lectures by academic staff and explanation by employers and other visitors of the thresholds they expected,
(2) Seminar with opportunity for exercises and detailed explanation, plus questions and answers and feedback on draft work,
(3) Submission of assessment,
(4) Quick feedback online with opportunity for personal feedback in the following seminar or by appointment, allowing for development between tasks,
(5) Final summative feedback following the final written self-reflection submission.

The delivery was resource intensive. Sixty staff, employers and postgraduate students were involved in delivering and assessing the module involving many issues of co-ordination, equity and moderation. Substantial deployment of technology in the assessment process (for example in psychometric testing), numerous different opinions, different cultural backgrounds, and many other 'business as usual' issues created significant complexity.

14.12 Presentation of Data

The data presented in this paper is drawn verbatim from reflective statements submitted by students undertaking the Professional Development module. It is presented anonymously with their permission. The reflective statements used formed the final element of the patchwork assessment developed for this programme of study (see Fig. 14.1). In this final element, students are encouraged and required to examine both positive and negative aspects of their professional development and are assessed upon quality of reflection rather than specific performance outcomes.

We have deliberately selected those statements (i.e. purposive sampling) that illustrate the argument of the paper. Threshold features were used as a framework for qualitatively analysing this data, taking a discourse analysis approach (Alvesson and Karreman 2000). It should be pointed out at this stage that a significant minority of students did not consider employability to be a subject worthy of time in the curriculum: these views are not used in the following section. The sample reported upon in this paper has not been broken down by specific degree programme or any other demographic parameter. In combination we suggest that the comments used demonstrate the attainment of intellectual freedom referred to by Barnett cited in Winter (2003) above. The selected quotes are intended to illustrate student insights that suggest the development of an appreciation of employability based upon threshold and constructivist concepts. They are presented as distinct sections but the content is all inter-related:

1. A number of students discussed how the module helped them come to understand the processes of employment:

"During the study of professional development module, I realised the importance of employability skills. I came to know that set of qualities, skills and knowledge that all newly graduates should obtain to ensure they have the skills of being persuasive in the workplace for their own benefit and their employers." (S1)

"From this module, I have found that I have developed my key employability skills a great deal and now feel I would have a lot more confidence when applying for a job role. I have a greater understanding of what employers are looking for in an ideal candidate … I have learnt that whilst it is important to possess skills such as teamwork, commercial awareness and leadership skills.., it all depends on how you can demonstrate them to the employer by using key experiences and situations to evidence them." (S2)

Both Student 1 and 2 make the point, also raised by others below, that first they needed to understand what was required, and then demonstrate their attainment of this. For some students, including S2, it was patchwork of the module assessment that gave her confidence.

2. Students discussed from their learning that despite initial misgivings, employability was a process of development:

"At the beginning I was very apprehensive about it working as it is just a University module. However, from completing it I can say that I have improved in all aspects especially compared to the weaknesses I pointed out I had during the personal skills assessment at the beginning of the module." (S3)

"My journey through Professional Development has been an enlightening and welcome experience. My understanding of self-employability has had a boost and I now feel more confident with the knowledge of what makes me employable such as knowing that being able to demonstrate a skill is just as important as possessing it." (S4)

"By taking this module my employability skills have improved drastically… When I started this module in January I had been rejected by five different companies that I had applied for a placement with. Towards the end I had two placement offers… My employability has increased … and my confidence has grown. By using Gibbs reflective model (1988) "to promote self-improvement and to link practice to theory." I am able to improve my employability even further." (S5)

It is interesting to see how many students refer to the development of their confidence, a theme discussed further below. What can also be seen is a level of commitment to the task and perseverance as the patchwork assessment progresses and that the relationship between the different recruitment processes was being understood.

3. As with Student 4 above, others discussed the importance of reflection and self-evaluation in the employability journey:

"I have learned from this module that regular self-evaluation is essential, that employability is vital to develop continuously, and that seeking and recording experiences that will enhance employability will enable me not to just have a career for life but be employable for life." (S6)

4. Although students didn't use the term 'threshold' the quotes below suggest that this is indeed what some students reached.

"At the beginning of this module, I was terrible at reflective thinking and writing and didn't see the point in it. Then I began to understand the importance of reflection on everything I do within my life. It made me begin to think about why reflection was important and how it linked with my personal employability. It is important, within an ever changing world, to conduct regular reflections in both thought and in writing, to enable us to be the best version of ourselves that we can be." (S7)

"Looking back at the whole experience of the professional development module, I have realised that the subject is a life changing experience. The module has made me become a better person with a strong persona… Although things were hard at first, it got better with time. This module obviously gave me confidence… I believe after accomplishing this Professional Development module I am now able to manage myself as a professional. I have identified areas that need to be developed and I have already started building on them."(S8)

5. As can be seen throughout the feedback, gaining confidence was the single most important achievement for many students. This in turn led to a development of their employability and realisation of a threshold:

"The module has helped me transition from an introvert to an extrovert in the working environment by developing my employability skills. I feel that everything I have learnt and developed throughout this module I can develop it by putting it into context when I am on my placement year and further down the line in my working career. From this moment on it is down to me as an individual to develop on the skills I have gained." (S9)

"When I originally learnt the details of the course I was incredibly nervous as I never identified myself as a person who excelled in areas such as group and face to face interviews… However, having reviewed feedback from each assessed piece of work I have completed I realise my worries were without basis. … This allowed me to identify my main weakness; and it wasn't my time management, my level of business awareness or my Microsoft skills; it was in fact my level of self-confidence. This module has enabled me to build confidence and has allowed me to demonstrate to myself that I can excel in a recruitment process… It has helped me grow in terms of my skills but also as a person and I believe my future employment prospects are improved thanks to the strengths and weaknesses I have identified and built upon." (S10)

"Before starting this module I had low self-confidence and underestimated myself…Overall my feelings at the beginning of this module were negative, however after completing assessments, attending lectures and doing my own research I have learnt a lot about myself in relation to employment. Through feedback and self-evaluation I have determined my strengths [and] this module has helped me develop along with weaknesses which I could improve. As a whole I have achieved well throughout the module which has helped build my self-confidence." (S11)

"I had not given any thought to the expectations that employers will have from me and was focussed on attaining a good degree. This module has been an enlightening experience to my professional character… The self-reflection I have continuously made in this module has made me realise where my strengths and weaknesses lie." (S12). Whilst quantitative analysis of this module has not been conducted it is probably relevant that in the two years for which destination data is available for students who have undertaken this module, i.e. 2015–16 and 2016–17, there has been a 4% increase from 2014–15 in the proportion of graduates reporting themselves in work or further study (increase from 84 to 88%). In the absence of further research it isn't possible to claim an association with the module.

14.13 Discussion

14.13.1 Constructing Employability

The evidence provided by these reflective statements illustrates how the module experience helped students actively construct their employability rather than passively learning about careers. Authentic assessment (Fook and Sidu 2010) was used as an opportunity for learning (Sambell et al. 2013) rather than a simple measure. The Patchwork of assessment, linked to expert advice, feedback and reflection allowed students to build upon the development of competencies. This accords with Perkins (1999, p 8) characterisation of constructivism as an energising process of discovery, one that yields deeper understanding. In attempting to capture the form of active learning described by Perkins, the teaching and learning activities sought to allow students to discover or re-discover principles that fostered understanding with practical results. This is illustrated by comments from S4 who discusses using reflection for self-improvement, and consequently changing from receiving repeated employer rejections to securing new job offers.

It can also been seen that students (S5 and S11 for example) were learning from employer rejection and assessment feedback, experiencing phases of liminality within the process. By continually engaging in various activities, such as job applications and CV improvement, students as learners entered this liminal space, engaging with the mastery of employability (Meyer et al. 2006).

14.13.2 Achieving a Threshold

The reflective statements illustrate the transformative and irreversible aspects of achieving a threshold of understanding of employability. Understanding why an employer requires a combination of competencies is integrative, and can be morale boosting since the threshold of employability is bounded. For example, the employer may not expect detailed technical expertise from an employee but they may require someone with enthusiasm to learn and an appreciation of their own employability. Student statements highlight the transformation, with S4, S5, and in particular S10 stating their change in attitude towards the module, its teaching and their personal learning. The statements also demonstrate liminality aspects relative to attaining thresholds. The process of understanding is not without difficulty for learners: comments such as feeling "nervous" (S11), having "negative feelings" (S12) or "insecurities" (S13) were common and illustrate the disorientation typical of a state of liminality as discussed by Meyer et al. (2006). Students alternated between embracing and rejecting the module, between anxiety and confidence. Because liminality indicates a period of oscillation, by definition some students will not have realised the threshold of understanding of employability until later in their learning journey.

For students, this new understanding can be troublesome in a very specific way. For many employers, demonstration of competencies is valued just as highly as qualifications. Some major employers have explicitly said that qualifications are not their primary recruitment metric (Times Higher 2015); rather evidence of employability is of greater significance. Most students have been brought up to believe that the fundamental purpose of their participation in education is the achievement of the highest marks possible. As the final student (S13) says above "my focus was on attaining a good degree", but as S2 comments; "I have a greater understanding of what employers are looking for in an ideal candidate". Empathising with the employer perspective and achieving the threshold of employability involves assimilating this troublesome knowledge and this is an essential part of self-realisation and transformation.

14.13.3 The Importance of Confidence

The student feedback also demonstrates that a constructivist approach can foster confidence, which is an essential ingredient in achieving the threshold of employability. Other researchers have also noted the significance of confidence in the achievement of a Threshold (e.g. White et al. and also Berg et al, cited in Land, Meyer and Flanagan 2016). The module helped students understand the perspectives and requirements of employers, understand their own development, appreciate the importance of reflection, gain confidence and thus realise a threshold. As S8 comments, “I have realised that the subject is a life changing experience. The module has made me become a better person with a strong persona… This module obviously gave me confidence”. Hawkins and Edwards (2013) discuss the ‘monsters of doubt’ that students experience when learning about leadership. These same monsters equally apply to employability. A key moment is when students understand clearly the competencies employers seek, why they are sought, and how to articulate them. This threshold may be reached after numerous job applications, assessment centres, interviews, or through real experience in employment. Discovering how to effectively articulate their competencies can be a lightbulb moment for students.

Many of the reflective statements submitted for assessment across the cohort include comments relating to increasing confidence. Once students grasp the linkage between their own skills and those required by employers, i.e. an integrative concept, they are creating knowledge *for* and *of* themselves that can be applied across the various aspects of employability. The confidence and perseverance we saw from many of the students was also an important aspect of maintaining engagement that Cranmer (2006) and Tymon (2013) suggested was lacking in other initiatives concerned with the development of employability in the curriculum. This is particularly important as it feeds into the benefits of an active and constructivist learning approach as discussed in our research.

Contrary to the work of Tymon (2013) and others, this paper suggests that HE can and should help guide students toward this self-realisation through the curriculum. This is demonstrated via a collaborative approach. To accomplish this does require a re-organisation of the curriculum (as suggested by Burch et al. 2014), starting with a focus on the final aim (the articulation of competencies), and working backwards through design of multiple assessments to achieve this aim. Substantial teamwork—including the involvement of employers—was essential in guiding students to this final aim or threshold. This approach succeeded in engaging the great majority of students. It also enabled a mind-set change for some of the significant minority to become engaged, to realise the importance of employability and how it applies to them. Overall it was thus much more effective than relying solely on ‘traditional’ extra-curricular support.

14.14 Conclusions

Demonstrating employability has become a central component of UK government policy in regards to Higher Education, and HEIs have made employability a key component of their mission. However, the ability to articulate employability skills, more commonly now known by employers as competencies, is not equally distributed and is related to the social capital that is held (Villar and Albertin 2010). Students have or can nurture competencies, but may not appreciate how to do this or how to express the same. This is where Higher Education can provide interventions and, to some degree, light the fire. If well designed, HE can help students construct their own employability, and thus enhance their experience of education more broadly. This paper has argued for the importance of embedding employability in the curriculum, guiding students to realise the threshold of employer expectations, and as a consequence construct their employability in more effective ways. The use of assessment and specifically, a Patchwork Approach (Winter 2003), is valuable in this process.

The paper explains the design and implementation of a module that has allowed students to recognise their competencies so widely cited as requirements by employers (Tomlinson 2012; Tymon 2013). This module has now been undertaken over a period of five years. Evidence of the efficacy of this approach is provided in excerpts from student reflections: further longitudinal and quantitative research would be necessary to gauge the long term impact. The single most important lesson to come out of the feedback from students involved is that confidence is fundamental to expressing competencies and demonstrating employability. Confidence is key to achieving the Threshold of Employability. Whilst the significance of social capital to employability is widely accepted, we were not aware of the importance of emotional (Cousin 2006) and psychological capital (Luthens 2007) for students in navigating the liminal space between HE and employability. These issues are clearly presented by Rattray (2016) opening up a whole body of work in Positive Psychology (e.g. Ivtzan and Lomas 2016) which we haven't addressed in the paper.

Valuable further research can be conducted on employability initiatives in HE which take as a starting point the development of student confidence built within the curriculum.

References

AGR (2016) The Association of graduate recruiters annual survey at: http://justoncampus.co.uk/wp-content/uploads/2016-AGR-Annual-Survey-2.pdf. Accessed 20 June 2018

Alvesson M, Kärreman D (2000) Varieties of discourse: on the study of organizations through discourse analysis. Human Relat 53(19):1125–1149

Archer W, Davison J (2008) Graduate employability: the view of employers. Council for Industry and Higher Education, London

Artess J, Mellors-Bourne R, Hooley T (2017) Employability : a review of the literature 2012–2016. Available at https://www.heacademy.ac.uk/knowledge-hub/employability-review-literature-2012-2016. Accessed 20 June 2018

Barr N (2014) Shaping higher education: 50 years after Robbins. In: Barr N (ed). LSE, London
BIS (2016a) Graduate market labour statistics 2015. Department for Business, Innovation and Skills, London
BIS (2016b) Success as a knowledge economy: teaching excellence, social mobility and student choice. Department for Business, Innovation and Skills, London
Boud D, Keogh R, Walker D (1985) Reflection: turning experience into learning. Routledge Farmer, Oxon
Boud D (1988) Developing student autonomy in learning, 2nd edn. Kogan Page, London
Brown P, Hesketh AJ (2004) The mismanagement of talent: employability and jobs in the knowledge-based economy. Oxford University Press, Oxford
Burch G, Burch J, Bradley T, Heller N (2014) Identifying and overcoming threshold concepts and conceptions. J Manage Educ 39(4):476–496
CBI (2015) Inspiring growth CBI/Pearson Education and skills survey 2015. London, UK
CIPD (2014) Industrial strategy and the future of skills policy. CIPD, London, UK
CIPD (2015) Policy report: over-qualification and skills mismatch in the graduate labour market. CIPD, London, UK
Cole D, Tibby M (2013) Defining and developing your approach to employability: a framework for higher education institutions Higher. Education Academy, available at: https://www.heacademy.ac.uk/system/files/resources/employability_framework.pdf. Accessed 20 June 2018
Cousin G (2006) Threshold concepts, troublesome knowledge and emotional capital: an exploration into learning about others. In: Meyer JHF, Land R (eds) Overcoming barriers to Student understanding: threshold concepts and troublesome knowledge. Routledge, London and New York
Cousin G (2010) Neither teacher-centred nor student-centred: threshold concepts and research partnerships. J Learn Dev Higher Educ (2)
Cranmer S (2006) Enhancing graduate employability: best intentions and mixed outcomes. Stud High Educ 31(2):169–184
Diamond A, et al (2011) Global Graduates into Global Leaders. AGCAS and others
Dubois D (ed) (1998) The competency casebook. International Society for Performance Improvement, Amherst, MA, HRD, & Silver Spring MD
Flanagan M (2017) Threshold concepts: undergraduate teaching, postgraduate training, professional development and school education. Available at: https://www.ee.ucl.ac.uk/~mflanaga/thresholds.html accessed 20 June 18
Fook CY, Sidu GK (2010) Authentic assessment and pedagogic strategies in higher education. J Soc Sci 6(2):153–161
Greenbank P (2015) Still focusing on the "essential 2: 1": exploring student attitudes to extra-curricular activities. Educ+Training 57(2):184–203
Guardian (2016) Editorial: The guardian view on graduates and employment: degrees but not destinations. https://www.theguardian.com/commentisfree/2016/may/03/the-guardian-view-on-graduates-and-employment-degrees-but-not-destinations. Accessed 30 Mar 2017 and 20 June 2018
Hawkins B, Edwards G (2013) Managing the monsters of doubt: liminality, threshold concepts and leadership learning. Manage Learn
HEA (2015) https://www.heacademy.ac.uk/knowledge-hub/graduate-attributes-framework. Accessed 29 Mar 2018 and 20 June 2018
HEFCE (2017) Computer science employability and accreditation. Review led by Nigel Shadbolt http://www.hefce.ac.uk/skills/gradstemreview/csreview/. Accessed 21 May 2018
HESA (2017a) Higher education student enrolments and qualifications obtained at higher education providers in the United Kingdom 2015/16. https://www.hesa.ac.uk/news/12–01-2017/sfr242-student-enrolments-and-qualifications. Accessed 15 Dec 2017
HESA (2017b) NewDLHE: destinations and outcomes review. Accessed 27 June 2018

Independent (2015) Leading employers prefer value work experience among graduates over grades. http://www.independent.co.uk/news/education/education-news/leading-employers-prefer-work-experience-over-grades-says-new-research-10286829.html. Accessed 31 May 2015

Ivtzan I, Lomas T (2016) Mindfulness in positive psychology. Routledge

Land R, Meyer JHF, Flanagan MT (2016) Threshold concepts in practice. Sense Publishers, Rotterdam, Taipei & Boston

Luthans F, Youssef CM, Avolio BJ (2007) Psychological Capital. Oxford University Press, New York

Matthews S (2017) Tech City Talent and Computer Science Graduate Employability: an independent learning review for the Tech Partnership and the JPMorgan's Chase Foundation. https://www.thetechpartnership.com/globalassets/pdfs/research-2017/techcitytalentandcomputersciencegraduateemployability_report_may2017.pdf. Accessed 24 Apr 2018

Meyer JHF, Land R (2003) Threshold concepts and troublesome knowledge 1—Linkages to ways of thinking and practising in improving student learning—Ten years on. In: Rust C (ed). OCSLD, Oxford

Meyer JHF, Land R, Davies P (2006) Implications of threshold concepts for *course design and evaluation.* In Meyer JHF, Land R (eds) Overcoming barriers to student understanding: threshold concepts and troublesome knowledge. Routledge, London and New York

Moon J (1999) Reflection in learning and professional development: theory and practice. Routledge-Falmer, Abingdon

NCIHE (1997) Higher education in the learning society http://www.educationengland.org.uk/documents/dearing1997/dearing1997.html. Accessed 20 June 2018

O'Leary S (2017) Enhancing graduate attributes and employability through initiative with external partners. Pract Evid Sch Teach Learn High Educ 12(3)

Pegg A, Waldock J, Hendy-Isaac S, Lawton R (2012) Pedagogy for employability. Available at https://www.heacademy.ac.uk/system/files/pedagogy_for_employability_update_2012.pdf

Perkins D (1999) The many faces of constructivism. Educ Leadersh 57(3)

Rattray J (2016) Affective dimensions of liminality. In: Land R, Meyer J, Flanagan M (eds) Threshold concepts in practice. In: Peters MA (Series Editor) Educational futures: rethinking theory and practice, vol 68. Sense Publishers, Rotterdam/Boston/Taipei

Sambell K, McDowell K, Montgomery C (2013) Assessment for Learning in Higher Education. Routledge, London

Times Higher (2015) Ernst and Young drops degree classification threshold for graduate recruitment. Available at https://www.timeshighereducation.co.uk/news/ernst-and-young-drops-degree-classification-threshold-graduate-recruitment. Accessed 20 June 18

Tomlinson M (2012) Graduate employability: a review of conceptual and empirical themes. High Educ Policy 25:407–431

Tomlinson M, McCafferty H, Fuge H, Wood K (2017) Resources and readiness: the graduate capital perspective as a new approach to graduate employability. J Natl Inst Career Educ Counselling 38(1):28–35

Tymon A (2013) The student perspective on employability. Stud Higher Educ 38(6):841–856

University of Salford (2017) TEF 2 provider submission. University of Salford

Villar E, Albertin P (2010) 'It is who knows you'. The positions of university students regarding intentional investments in social capital. Stud High Educ 35(2):137–154

Wiggins G, McTighe J (1998) Understanding by design. Association for Supervision and Curriculum Development, Alexandria, VA

Winter R (2003) Contextualizing the patchwork text: addressing problems of coursework assessment in higher education. Innovations Educ Teach Int 40(2):112–122

Woods A, Dennis C (2009) What do UK small and medium sized enterprises think about employing graduates? J Small Bus Enterp Dev 16(4):642–659

Chapter 15
Baseline Skills—Scaffolding Soft Skills Development Within the Curriculum

Sue Beckingham

Abstract To enable Computer Science students to develop employability and 'work-ready' skills it is important to consider both the technical skills aligned to their discipline and the soft skills desired by employers. Research has identified that students in Computer Science would benefit from further support to develop the latter. This chapter considers how these skills can be developed through a variety of work experience opportunities including work-based learning and work-related learning; in class activities and alternative teaching approaches such as project, inquiry and problem-based learning; and through scaffolding both soft skills development and reflective practice, how students can become more confident in articulating these skills when applying for graduate work.

Keywords Employability · Soft skills · Problem based learning · Reflective practice

15.1 Introduction

Over recent years there have been numerous accounts in the news proclaiming that graduates are not work ready as they don't possess the baseline soft skills required by employers. Despite the growth in Computer Science and IT-related degrees and their subsequent graduates, unemployment was running at just over 10% (Shadbolt 2015). As a result the 'Shadbolt Review of Computer Sciences Degree Accreditation and Graduate Employability' was commissioned and the research published in 2016.

This chapter will consider the recommendations made within the Shadbolt Review and other research to highlight:

S. Beckingham (✉)
Sheffield Hallam University, Sheffield, UK
e-mail: S.Beckingham@shu.ac.uk

J. Carter et al. (eds.), *Higher Education Computer Science*,
https://doi.org/10.1007/978-3-319-98590-9_15

- what elusive soft skills employers are looking for;
- provide guidance and recommendations on how soft skills development can be integrated effectively within the curriculum; and
- how students can go on to apply and showcase these skills confidently and effectively through a professional online presence.

15.2 Employability

Over two decades ago Dearing (1997: 133) recommended that higher education focus on key skills which were the ‘key to the future success of graduates whatever they intend to do in later life’ (p. 133). Skills identified included: communication skills, numeracy, information technology, learning how to learn/personal development planning, problem solving and team-working. Leckey and McGuigan (1997) lamented on the allegations of a gap between the generic skills fostered by higher education, and those that the labour market need. Whilst subject specific knowledge and skills are strong, the transferable knowledge, skills and attitudes essential for the world of work are weak. They go on to cite the European Commission (1991: 44) who argue “One feature of current skills shortage is the widespread lack of important generic skills and social skills such as quality assurance skills, problem-solving skills, learning efficiency, flexibility and communication skills”.

To understand what makes individuals employable Knight and Yorke (2004: 5) defined employability as “A set of achievements—skills, understandings and personal attributes—that make individuals more likely to gain employment and be successful in their chosen occupations, which benefits themselves, the workforce, the community and the economy”. Cole and Tibby (2013: 5) add that employability is about “supporting students to develop a range of knowledge, skills, behaviours, attributes and attitudes which will enable them to be successful not just in employment but in life”.

In 2016 the Shadbolt Review was commissioned by Ministers from the Department for Business, Innovation and Skills (BIS), where one of the key strategic priorities was to support and develop science and engineering talent coming through the education system and ensure that the UK has access to ‘the skills and knowledge that it needs to drive economic growth and the development of a more innovative, productive and information-driven economy’. (Shadbolt 2016: 13). To realise this ambition, it is vital to align the skills graduates have with those required by employers. In order to do this and prepare work ready graduates, a recommendation for employers and higher education providers to work more closely together was made.

The Shadbolt Review (2016: 52) highlighted the following as the main issues impacting on graduate employability:

- Graduates lacking 'softer skills'
- Graduates lacking specific knowledge
- Graduates lacking computer programming skills of specific programming languages
- Graduates lacking business/commercial awareness
- Graduates lacking work experience.

15.3 Skills

The terminology used to describe skills most frequently refers to soft and hard skills. Generally speaking hard skills tend to be those that have been or can be tested; may be recognised with a professional, technical or academic qualification; and are quantifiable. Soft skills are personal life skills and tend to be more subjective. However there are variations as can be seen in Fig. 15.1.

In 2004 the Pedagogy for Employability Group (p. 5) as cited by Dacre-Pool and Sewell (2007) collated the following skills, advocating that these were the skills that employers expect graduates to have:

- imagination/creativity
- adaptability/flexibility
- willingness to learn
- independent working/autonomy
- working in a team
- ability to manage others
- ability to work under pressure
- good oral communication
- communication in writing for varied purposes/audiences

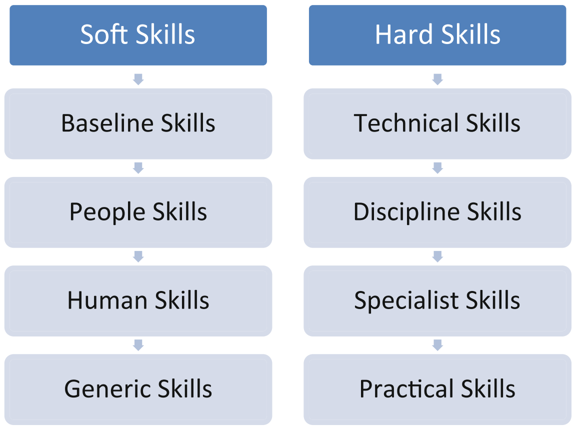


Fig. 15.1 Alternative names for soft and hard skills

- numeracy
- attention to detail
- time management
- assumption of responsibility and for making decisions
- planning, coordinating and organising ability.

A series of scoping interviews with employers carried out by Lowden et al. (2011: 12) recognised the need for skills and knowledge for specific roles (which would include technical skills); however all agreed the following *transferable* skills were considered most relevant:

- team working
- problem solving
- self-management
- knowledge of the business
- literacy and numeracy relevant to the post
- ICT knowledge
- good interpersonal and communication skills
- ability to use initiative but also to follow instruction.

The UK Commission for Employment and Skills (UKCES) a publicly funded, industry-led organisation provides leadership on skills and employment issues across the UK. UKCES carry out an Employer Skills Survey at regular intervals, the last conducted in 2015. Within this report it states that skills can be defined within two groups and named these: ‘people and personal skills’ and ‘technical and practical skills’ (UKCES 2015a: 44).

Hawkins (1999) splits soft skills into three groups: people skills, self-reliance skills, and generalist skills; then includes a fourth group as technical skills. People skills include team working, leadership, interpersonal skills, and customer orientation. Self reliance skills include self awareness, confidence, self-promotion, initiative, proactivity, networking, willing to learn and action planning. Generalist skills include problem solving, IT/computer literacy, flexibility, numeracy, business acumen and commitment (Hawkins 1999: 12).

Kaplan (Pedley-Smith 2014) published a white paper on graduate recruitment, learning and development in which a survey compiled a list of competencies. These were grouped as Knowledge (e.g. numeracy, literacy, technical knowledge), Skills (e.g. effective communication, analytical, problem solving) and Attitude (e.g. team player, confidence, positive mental attitude). Thurner et al. (2012) consulted software companies (potential employers of graduates) to identify desired skills and these are grouped as Self-Competencies (e.g. openness to constructive criticism, striving for life-long learning), Practical and Cognitive Competencies (e.g. analytical thinking, diligent and accurate work style) and Social Competencies (e.g. empathy and understanding of others, ability to work and cooperate in a team).

The World Economic Forum (WEF) report (2016: 21) categorises core work-related skills as abilities, basic skills, and cross-functional skills. Within these groupings there are subgroups:

- Abilities: cognitive abilities and physical abilities
- Basic Skills: content skills and process skills
- Cross Functional Skills: social skills, systems skills, complex problem solving skills, resource management skills and technical skills.

An area of concern is the ability for students and graduates to recognise and evidence vital soft skills. Being able to articulate transferable skills such as problem solving with clear examples requires the students to catalogue a collection of examples and to develop the confidence to sell themselves (Shadbolt 2016). One of the issues that students and graduates appear to struggle with is being able to provide clear examples of when and how a given skill has been applied in an authentic situation. However given the complexity of how skills are referred to, it should not come as a surprise that students and educators trying are struggling to know what skills they should focus on and the language to use to describe them.

15.4 Skills Gaps

A crucial problem faced is gaining a clear understanding about *what* key skills employers are looking for. The Council for Industry and Higher Education (CIHE 2010) are cited by UKCES (2015b: 48) stating "the importance of individuals possessing a 'fusion' of technology, business, creative and interpersonal skills". The Association for Computing Machinery (ACM) states that each computing discipline must "articulate its own identity, recognize the identities of the other disciplines, and contribute to the shared identity of computing" (Association for Computing Machinery 2005:8). In addition to having foundational knowledge in Computer Science, the Burgess Report (2007: 70) also highlights the significance of 'soft' or 'work readiness' skills. CBI's report (2017) refers to work-related attitudes, with communication, teamworking and a positive attitude to work being critical to attaining career opening.

It is therefore important for students to not only build and develop soft skills, but to be applying these skills and knowledge gained in a professional and business context. The Office of Students (2018) an independent regulator of higher education in England states within its strategy that as an outcome of the higher education experience, students will be able to progress into employment or further study. Universities UK (2016: 30) argue that "while a degree may be a baseline requirement for attaining a job, an applicant's ability to demonstrate skills and competencies that they can bring to the workplace will allow them to stand out from their graduate counterparts".

The UKCES Employer Skills Survey (2015a: 45) acknowledging there is a need to understand what skills are in poor supply, looks to identify skills lacking in the labour market. Employers taking part in the survey selected from a list of skill descriptors presented within the two aforementioned groups:

1. people and personal skills
2. technical and practical skills

To some extent these groupings could be seen to complicate matters. For example making speeches or presentations are under people and personal skills and yet basic numerical skills and writing instructions, reports etc. are placed within technical and practical skills. There is an overlap here if communication skills were used to describe a typical soft skill. That aside it still provides an indication of skills gaps and areas to focus on.

The 2017 Tenth CBI Education and Skills Survey (run in partnership with Pearson) received responses from 340 UK organisations. Within this research a similar set of skills were identified. Areas of particular weakness included international cultural awareness (39%), business and customer awareness (40%), and attitudes/behaviours e.g. resilience and self-management (32%) (CBI 2017: 93). Furthermore being able to demonstrate personal qualities such as resilience, attitude and confidence is inextricably linked to soft skills, and therefore also of importance (Fincher and Finlay 2016).

Looking to the future the anticipated skills needs are even harder to predict. On average, by 2020, more than a third of the desired core skill sets of most occupations will be comprised of skills that are not yet considered crucial to the job today (World Economic Forum 2016: 20).

In the Council of Professors and Heads of Computing (CPHC/HEA 2015) report on Computing Graduate Employability, it refers to an employer invited to speak to students about expectations, and is cited as stating that those students graduating with a first or 2:1 demonstrate the aptitude to learn so any additional technical skills required can be taught. However what they cannot teach is the soft skills.

The employers consulted during the Shadbolt Review indicated that whilst a range of soft skills were desired, the top two were **communication** and **project management skills**, stating that these were "crucial for working in teams, developing successful working relationships and contributing positively to an employer's strategic vision" (Shadbolt 2016: 55). Interestingly similar to the findings two decades prior by Leckey and McGuigan (1997: 368) who refer to the importance of 'personal transferable skills' and categorise these as communication skills, problem analysis and solving, interactional skills, initiative and efficiency.

Overall there is evidence that students benefit from authentic work experiences, ideally in an organisation that provides them with the opportunities to develop these soft skills and of particular importance communication as this is a fundamental overarching skill that will enable students to articulate the range of skills and experience they have in subsequent job interviews. Being able to communicate is clearly a vital skill, but to do so confidently needs development, practice and ongoing feedback.

15.5 Work Experience

One of the key recommendations from the Shadbolt Report (2016) is to extend and promote work experience. Placements and internships can provide rich opportunities to apply knowledge and current skills, as well as developing new skills. Such authentic work experience can provide a context for learning (Pegg et al. 2012). The report also suggests that the National Centre for Universities and Business (NCUB), the Council for Professors and Heads of Computing (CPHC), and the National Union of Students (NUS) should work closely to try to identify what barriers Computer Sciences students are facing when trying to gain work experience.

There is a correlation between those that have undertaken a placement (or other work experience) and being employed once graduated. It is therefore very important to prepare and encourage students to ensure they are in the best possible position to apply for posts. However 'work awareness' is also valued. This is where employers have an expectation that new employees will have "a useful awareness of the world of work … a feel for the market, and what's going on in the world" (Bennet et al. 2000: 101).

Work experience can be referred to as work-based learning (in the workplace) or work-related learning (workplace and learning space, and also simulated space). Work-based learning (WBL) is the term used to describe a class of university programmes that brings together universities and work organisations to create new learning opportunities in workplaces (Strachan et al. 2011: 134). The Department for Children, Schools and Families (DCSF 2009) define work-related learning as: "Planned activity that uses the context of work to develop knowledge, skills and understanding useful in work, including learning through the experience of work, learning about work and working practices, and learning the skills for work". This is further described as:

- learning ***for*** work by developing skills for enterprise and employability (e.g. problem-solving activities, work simulations and mock interviews
- learning ***about*** work by providing opportunities to develop knowledge and understanding of work and enterprise (e.g. careers education)
- learning ***through*** work by providing opportunities for young people to learn from direct experiences of work (e.g. work experience or enterprise activities).

Work experience can be both paid and unpaid. Opportunities to work within an organisation closely related to the subject discipline can provide access to develop

Table 15.1 Skills development opportunities

Examples of where skills can be developed	
Sandwich placement	year-long industry placements typically in the third or final year
Semester placement	short internships could be as little as a day a week or in a block
Work shadowing	an opportunity to get a feel for the working environment
Summer internships	taken over the summer break and include working abroad
Part-time work	typically unrelated to subject discipline
Course and university initiatives	hackathons and competitions, student led conferences, students as researchers projects
Volunteering	charities, computer clubs, peer assisted learning schemes, course rep and other university committees
Extra-curricular activities	clubs, societies, special interest groups

the broadest set of skills. However much can be gained through engaging with part-time work, volunteering and extra-curricular activities as important transferable soft skills can be developed and evidenced. Wilson (2012: 37) stated that in addition to the course/programme studied, personal skills are developed as "a consequence of social and family background, the environment within which study is undertaken and the extracurricular activities of the student". This might include voluntary work, community work, or part-time work unrelated to the degree subject (Table 15.1).

Cross-discipline work-related learning opportunities can provide two way learning. For example an institution led initiative provided an IT service desk to nursing students. The Nursing students gained IT skills and the Computing students were given feedback on their communication skills. Peer assisted learning (PAL) schemes help to develop leadership, communication and mentoring skills, whereby students mentor students in the year(s) below them.

15.6 Developing Opportunities for Work Experience

The development of an Industrial Advisory Board linking Computer Science academics with industry professional can open up a forum to discuss both work experience prospects and the preparation needed to apply for such posts; as well as forging links for graduate jobs (Universities UK and UKCES 2014, UKCES 2015b). Partnerships with businesses can also provide opportunities for guest lectures, industry visits and projects. Students on placement often go on to focus their final year project or dissertation on a topic related to the organisation they worked at. Employer engagement can include inviting employers to provide information, advice and guidance;

or to contribute case studies or work-based scenarios to the curriculum (University Alliance 2015).

Cole and Tibby's (2013: 10) paper 'Defining and developing your approach to employability' is a useful planning tool and comes with an action plan for course teams. It considers four stages:

1. Discussion and reflection—creating and defining a shared point of reference.
2. Review/mapping—what are we doing/not doing?
3. Action—how do we share and enhance existing practice? How do we address gaps in provision?
4. Evaluate—What does success look like and how is it measured? How can we enhance practice further?

There is much to be gained through academic cross-institutional visits, developing communities of practice between institutions to discuss and share good practice about employability and skills development. A CPHC-funded (Council of Professors and Heads of Computing) initiative focussed on employability took place during 2016/17 called 'GECCO Building a Graduate Employability Community in Computing'. A series of three events took place in London, Manchester and Edinburgh. The evaluation report (CPHC 2017) shared that participants valued:

- Time and space share/discuss practice.
- Discussions between practitioners for ideas and reflection.
- Networking (mentioned three times).
- Exchanging ideas/establishing new contacts.
- Contacts. Ratification that we are not alone!!
- New contacts at other institutions.
- The networking opportunities.
- Talking to colleagues about employability across the sector.

Maintaining alumni networks is key, as when students graduate and maintain connections with their alma mater, they are more likely to let past Tutors know about job offerings and placements opportunities in their workplace. Equally it enables the academics to stay in touch and invite graduates back to give talks to inspire students to take a placement year, undertake extracurricular activities and explain how these develop the skills employers are seeking. Utilising LinkedIn, (regarded as 'professional' networking) to create course alumni groups, provides a useful way to keep in touch with graduates and follow their progression. Encouraging students to also join industry groups and follow Company pages can give an insight into the culture and specialisms within organisations.

15.7 Approaches to Teaching Soft Skills

Teaching soft skills can be challenging. Some computing programmes have specific stand-alone soft skills modules which focus solely on professionalism and communication. Whilst often aligned to the skills identified by employers as important, students do not always value the module. For some it is seen as a stand-alone bolted on addition to the course, containing activities the students feel they are already competent in. Criticism of this approach suggests that activities are not sufficiently contextualised in the computing specialism students are taking.

Knight and Yorke (2004: 199) note the following ways to include employability skills development in the curricula:

- employability through the whole curriculum
- employability in the core curriculum
- work-based learning or work-related learning incorporated as one or more components within the curriculum
- employability-related module(s) within the curriculum
- work-based or work-related learning in parallel with the curriculum.

The British Computer Society's (BCS) best practice model considers the legal, social, ethical and professional issues in computing (LSEPI). Healey (2014) argues that providing opportunities to work through ethical issues can help students to develop critical thinking skills. Connecting current examples of these issues that align with specific computing specialisms are more likely to engage students than providing generic examples. An inquiry-based learning approach can provide students to research exemplars and then link the activity with an exercise to identify the skills they have gained as a result. There is further scope for BCS to collate Computer Science case studies of best practice implementing effective approaches to embed LSEPI in the curriculum; and to liaise with Association of Graduate Recruiters (AGR) and the Association of Graduate Careers Advisory Services (AGCAS) to develop a model for accrediting careers advice provision within the curriculum (Shadbolt 2016).

The capstone or service learning module is an alternative approach whereby students engage in project based learning and key skills are purposefully integrated. For example, written and verbal communication, teamwork and organisation skills (Carter 2011). Mock data and role play where students play the role of the Client set the scene for the project (Vogler et al. 2017). In some instances, real businesses are involved and provide a valuable work-related learning experience. The application of communication skills are demonstrated through meetings with the Client face to face, online, by phone, via email and through presentations and a final report; along with a vast array of other skills needed to undertake and complete the project. When working with a real business, students will experience client management, meeting client expectations and potentially conflict management (González-Morales et al. 2011). Jackson (2014) refers to work-integrated learning and opportunities for service learning where students apply their professional skills through participating in an authentic activity that benefits the community. There are also opportunities to cross disciplinary boundaries, involving students from other courses. Hazzan and Har-Shai (2014) ask students to reflect on one stage in the lifecycle of a company, stage of a project or specific department, and describe the skills required. Yu and Adaikkalavan (2016) expound upon the value of problem-based learning where time is given to coach students and evaluate performance.

Another approach is involving students in the planning of the curriculum in relation to soft skills development. This can be done by asking the students what skills they would like to develop on the course and to suggest or contribute to the design of activities to develop these. Using a card sort exercise, students can engage in an activity to rank skills. For example:

- group activity—identity the ones they feel employers most value
- individual activity—identify the skills they are most/least confident in
- group activity—design activities to develop skills
- Individual activity—articulate what soft skills are and why they are of value.

Innovative approaches take a constructionist approach (Papert and Harel 1991) and use Lego Serious Play to engage students in discussions around topics such as barriers and enablers of effective teams, and skills development. Students use the bricks to build metaphorical representations and share stories based on these. It provides an effective way for students to express themselves in a non-threatening way and to learn from each other (Peabody and Noyes 2017).

Key to all of these activities is helping students identify and then articulate the skills they are developing. Often what can be missing is a portfolio that is owned and valued by the student to capture this information.

15.8 Personal Development Planning (PDP)

The integration of personal development planning (PDP) can be an effective way to engage in reflective practice. PDP is defined by the QAA (2009: 2) as a "structured and supported process undertaken by a learner to reflect upon their own learning, performance and/or achievement and to plan for their personal, educational and career development". Furthermore the QAA states PDP supports the idea that learning is a lifelong and life-wide activity. To achieve skills enhancement requires self-awareness of strengths and weaknesses from personal skills profiles (Wilson 2012).

The 2004 Burgess report 'Measuring and Recording Student Achievement' concluded that whilst advocating the use of personal development planning, further work should be supported to ensure research extended knowledge of the most effective strategies and evaluation of impact. This was followed in 2007 by the Burgess Group Final Report 'Beyond the honours degree classification' which introduced the Higher Education Achievement Report (HEAR). The HEAR (2008) is described as being "designed to encourage a more sophisticated approach to recording student achievement, which acknowledges fully the range of opportunities that higher education institutions in the UK offer to their students".

The recommendation was that students evidence additional skills attained at university which is intended to provide employers with an additional report to their transcript of academic results, providing a more comprehensive record of achievement. For example the HEAR (2008) can capture:

- additional awards—accredited performance in non-academic contexts and individual units/modules studies in addition to the main degree programme
- additional recognised activities—roles and activities undertaken by students which demonstrate achievement but for which no recognition is given in terms of academic credit e.g. volunteering, student union representative roles, representation at national level sport or training course run internally
- university, professional and departmental prizes.

Another way of capturing attainment is through 'Progress Files'. These can provide a way for students to record, reflect and review the skills and experience gained from the curriculum, work related learning, work-based learning as well as casual and voluntary work. In doing so they will develop the confidence to articulate and evidence their skills and knowledge (QAA 2001). In Pegg et al. (2012: 27) a case study by Waldock advocates the use of weekly e-Progress files throughout their time at university, whereby students develop a culture of engagement, value supported learning through regular feedback, and provides staff with regular ongoing feedback on their teaching.

PDP is seen both as a set of process and through a portfolio a valuable product in its own right. Within the portfolio, students are expected to review achievements, identify learning needs, plan how to address these needs, and present achievements (Knight and Yorke 2004). However for this to be achieved it is important that guidance and support is given to students (Beard 2018). It should also be acknowledged that reflection takes practice and encouragement. To move from surface reflection and acceptant thinking to deep reflection and question thinking, students need to learn how to effectively undertake self-analysis and achieve self-awareness. They need to value the process and the outcomes (Carter 2011). When a motivated, there is a risk that they will just go through the motions. Fung (2017) advocates the integration of research and enquiry-based pedagogies, where activities and skills develop over time.

Dacre-Pool and Sewell (2007: 280) created the 'The Key to Employability' a metaphorical model. The CareerEDGE Employability Development Plan aims to develop increased levels of self-efficacy, self-confidence and self-esteem through reflection and evaluation. CareerEDGE is a mnemonic for the five components of the model.

1. **Career** Development Learning
2. **Experience** (work and life)
3. **Degree** Subject Knowledge, Understanding and Skills
4. **Generic** Skills
5. **Emotional** intelligence.

The Centre for Recording Achievement (nd) is a national network organisation and a registered charity which seeks to "promote the awareness of recording achievement and action planning process as an important element to improving learning and progression throughout the world of education training and employment". It provides a useful collection of case studies and CPD opportunities.

15.9 Scaffolding Reflective Practice

Learning how to reflect on experiences and developing a habit of doing so is an important life skill. It can have a profound impact on learning. Boud et al. (1985) consider the journey of turning experience into learning in three stages: What? (experience), So what? (reflection) and Now what? (learning). Developing reflective skills can in itself help individuals with the art of knowing how to learn (Helyer 2015).

The concept and pedagogy supporting the use of PDP has been written about extensively. As practioners we know ourselves it is valuable as part of our own CPD. However it is clear that reflecting effectively for many does not come easily. One of the barriers is finding the language to articulate what needs to be said. 'Blank canvas syndrome' is rife, with students declaring "I don't know what to write!"

There is a tendency to simply write about the 'what I did' but not expand upon what might be done differently, what was learned, what areas of development are needed and a recognition of skills that have been developed. It is therefore helpful to scaffold the process of reflection. This can be done by first of all explaining what reflection is and providing examples of how reflective practice is used outside of academia.

For example reflection is common practice in sport, where after a game the players will review their performance. Reflective practice in sport appraises what was on form and what can be learned from mistakes made. In the military After-Action Reviews (AARs) are conducted during or immediately after each event. They use open ended questions, determine strengths and weaknesses, and link performance to training. These consider: what did we set out to do, what actually happened, why did it happen and what will we do next time.

Then in the context of computing the use of Scrum has been used since the 1990s. Schwaber and Sutherland (2017) define Scrum as "a framework within which people can address complex adaptive problems, while productively and creatively delivering products of the highest possible value". Scrum prescribes four formal events for inspection and adaptation:

1. **Sprint Planning**—The work to be performed in the Sprint is planned at the Sprint Planning. This plan is created by the collaborative work of the entire Scrum Team.
2. **Daily Scrum**—This is a 15 min time-boxed event for the Development Team and is held every day of the Sprint. At it, the Development Team plans work for the next 24 h.
3. **Sprint Review**—This is an informal meeting, not a status meeting, and the presentation of the Increment is intended to elicit feedback and foster collaboration.
4. **Sprint Retrospective**—This is an opportunity for the Scrum Team to inspect itself and create a plan for improvements to be enacted during the next Sprint.

Whilst the language used is different the concept of teams 'inspecting itself', can be aligned with reflection. It can be valuable to invite professionals into describe how Scrum is used in the context of their own organisation.

When it comes to capturing reflective practice, rather than asking students to go away and write 200–300 reflective words, providing aide-mémoires to guide students can be a very helpful and supportive approach. Building on the work of Gibb's (1988) reflective cycle which commences with a description of what happened and then considers feelings, evaluation, analysis, a conclusion and an action plan; it can be helpful provide suggestive prompts for each of the six stages of reflection. These can be presented in the form of a check list to encourage students to reflect beyond *the doing* and reach a point where they can start to identify skills they are developing and those that need further attention. Furthermore they can contextualise them in authentic scenarios, providing examples of how they have been able to apply the skills.

1. **Description: What happened?**
   Begin by describing in detail the activity you are going to reflect upon. Think about including who you were with, where you were, what did you do/read/see; what were your responsibilities and what did you contribute, what others contributed; what were the outcomes.
2. **Feelings: What were you thinking and feeling?**
   Now consider what you were thinking about. Capture what you were feeling at the beginning of the activity. What did you feel when you completed and how do you feel now? Did your feelings change? Consider how others made you feel.
3. **Evaluation: What was good and bad about the experience?**
   The next step is to evaluate your experience. Think about what went well and what didn't go as well as expected. Record both the positive and negative aspects of each stage of the experience. Were there any difficulties? What/who was helpful/unhelpful?

4. **Analysis: What sense can you make out of the situation?**
   Look more closely at why you think aspects went well or didn't go so well. What contributed to things going well? Where things went badly think about how this might have happened. Think about your contribution and how others contributed. How does it compare to other experiences?
5. **Conclusion: What else could you have done?**
   Now you need to consider what you have learned from your experience. It is important to be honest with yourself and think about how you could have done anything in a different way. Have you learned anything from other people's approaches or behaviours?
6. **Action plan: If it rose again what would you do?**
   Finally if you were to find yourself doing this or a similar activity again, how would you approach it? What would you differently? Think about the skills you may need to develop to ensure it went better next time. Plan how and when you will undertake any skills development.

A further consideration is encouraging the students to explore a variety of approaches to capture their reflections. For example multimedia reflective blogs can contain text, but also photos, audio and video, sketch notes and mind maps. For example students learning in groups to programme a Lego Mindstorm in preparation for a race, could capture maths calculations as a photo, and a video of the robot in action. The visuals can aid recall and provide a focal point for reflection on the activity.

15.10 Scaffolding the Articulation of Soft Skills

It is important that student value the broad range of opportunities experienced whilst at university (Knight and Yorke 2004)and learn how to reflect confidently on the skills that they are developing; become able to articulate these; and also realise where their skills development is weak and understand how to develop an action plan to overcome any weaknesses.

As with reflective writing it can be challenging finding the right words to demonstrate skills. Hawkins (1999: 12) presents a 'Skills Portfolio' which captures a useful collection of adjectives for each of the skills listed. Providing students with such a list can act as a prompt as they identify and build their own portfolio of skills. For example: team working: supportive, facilitator, organised, co-ordinator, deliverer, imaginative, delegator, open-minded or willing to learn: motivated, adaptable, enthusiastic, active, keen learner, inquisitive, continual improver (Hawkins 1999: 12).

This could also be developed into a group activity where students in groups are tasked to identify relevant adjectives to describe a skill and the class shares and critiques the outputs. These can then be compared to Hawkin's Skills Portfolio.

15.11 Peer Support

Students can learn about skills development and the value from their peers through peer assisted learning and mentoring schemes. Students as mentors develop leadership, communication and mentoring skills and students as mentees can develop a range skills influenced by the activities they are able to engage with (Pegg et al. 2012).

Another useful approach is either inviting students currently on placement, final years returning from placement or graduates who have experienced placements, to be involved in group or individual speed dating conversations or poster presentations to highlight the skills required for placements and the skills developed whilst out on placement.

15.12 Careers Support

Many universities have a Careers unit within the university. Those working in this area can provide students with one to one personal support or as group activities to further help them prepare for placements and graduate jobs. For example:

- Job applications
  - Skills
  - Competencies
  - Attributes
- CVs and cover letters
  - Skills and STAR statements
  - Technical skills
- Interviews
  - Phone interview
  - Skype video interview
  - Face to Face interview
- Assessment Centres
  - Group exercises
  - Role play
  - Logic tests
- Tests
  - Psychometric testing
  - Numerical testing
  - In tray test (task organisation).

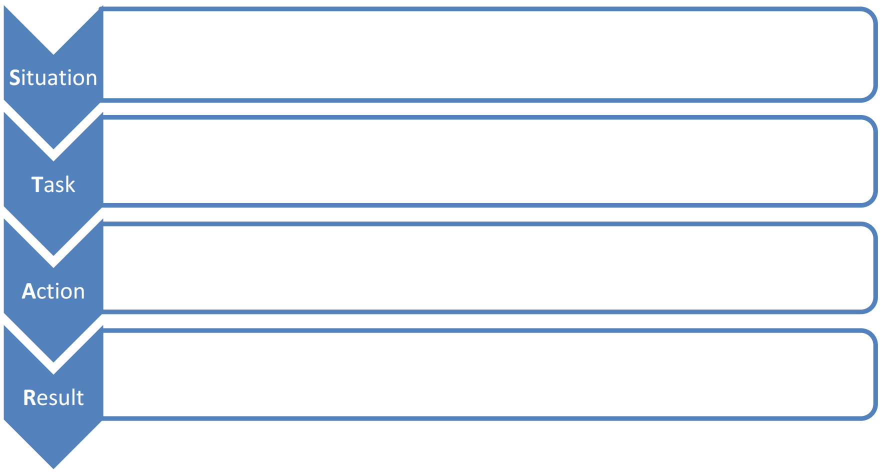


Fig. 15.2 Template for a STAR statement

By engaging in these mock activities, students can practice articulating the skills they have and demonstrate these to receive feedback. This can be further enhanced where there is support through mentoring schemes, Academic Advisors, Personal Tutors, Employability Advisors and PAL (peer assisted learning) schemes.

Other class activities can support the development of presentation skills. This might include the use of PowerPoint (or similar), infographic posters or interactive digital poster, or screencasts.

15.13 Recording Skills

A useful approach to describe any skill is to develop a S.T.A.R. statement for each, whereby a scenario considers each of the four elements: situation, task, action, and result. Using the template in Fig. 15.2 a specific skill is chosen and then evidenced by considering an example of where it was applied. This allows for reflection on examples where things went well and also where they didn't. By considering the result there is an opportunity to further reflect on what was learned from a situation and what might be done differently next time. An interview question for example might be 'Describe a situation where you were working in a team and things didn't go as planned'.

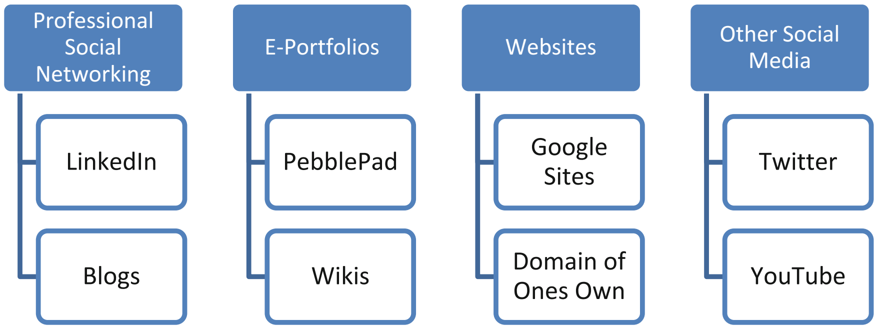


Fig. 15.3 Examples of tools to capture reflect on and showcase learning

Hawkins (1999: 14–15) developed an exercise to record tasks undertaken and skills gained in any type of work experience. The examples given include full-time work, voluntary work, part-time work and community work. The objective is to list all relevant tasks undertaken in the selected role and then for each task consider what skills were developed. To further aid the process skills are to be considered under the following four headings: working with data and information, working with people, working with practical things, or working with ideas. The exercise can be repeated for each role an individual has had. The outcome demonstrating the broad range of skills an individual has when looking at this from a wider perspective.

15.14 Showcasing Skills

In addition to the traditional CV, students applying for placements or graduate jobs can benefit greatly from having a professional online presence to showcase achievements and demonstrate their skills. This can be done by creating a profile on LinkedIn, which is a both a professional networking site and a tool for recruiters to find talent. In addition the use of blogs, wikis, websites and e-portfolios can provide spaces to share skills, achievements and work experience (see Fig. 15.3). Engagement will increase if students can see the purpose (Barrett 2005) of digital portfolios. Clear signposts can help prospective employers find further information. This can easily be done by cross-referencing the online spaces using hyperlinks.

Students need to take ownership of these spaces and value what is presented publicly. It is common practice for employers and recruitment consultants to use search engines like Google to screen applicants. Digital footprints left in social media spaces where open to the public can reflect on the individual positively, but also negatively. Taking ownership of a personal profile is therefore important. The act of 'googling' yourself can be quite revealing. Not only does it help users see what others might see, it can provide a timely prompt to revisit social media profiles to check privacy settings where information is not intended to be openly public, and also to tidy up profiles that are public.

Hundreds of millions of professionals now have a LinkedIn profile to which they add their education, skills and information about their employment. As mentioned, employers will actively search for potential candidates using LinkedIn. Considering what search terms they might use to do this is another useful activity. These keywords might include placement or graduate. Where these words are included in a profile header can increase the chance of being found. For example: '*BSC Computing Science graduates looking for employment*'. Adding skills to a profile can help in this search process as well as providing the reader with a clear picture of the strengths of the individual.

Self-promoting a professional online presence can be done in a number of ways.

- email signature—adding the URL of a LinkedIn page, blog or website
- business cards—as above and converting the link into a QR code
- paper CVs—it is now common to see a LinkedIn URL included at the top of a CV alongside an email address and telephone number
- social media—links to any digital portfolios can be shared via Twitter and LinkedIn.

15.15 Summary

The range of soft skills can be seen as a minefield. However building partnerships with industry can help to highlight what is required as work-ready skills, as this will differ in different specialisms within the sector and is likely to change over time in relation to what is a current priority. It is crucial that graduate skills are aligned with the expectations and needs of employers. Underpinning the wide spectrum of skills is the ability to confidently reflect and communicate examples when faced with a job application and interview. The development of all skills is enhanced greatly when students engage in work experiences.

References

Association for Computing Machinery (2005) Computing curricula 2005: the overview report. Available at: https://www.acm.org/binaries/content/assets/education/curricula-recommendations/cc2005-march06final.pdf

Barrett HC (2005) Researching electronic portfolios and learning engagement. Available at: http://ww.w.electronicportfolios.org/reflect/whitepaper.pdf

Beard C (2018) Dewey in the world of experiential education. New Dir Adult Continuing Educ 158:27–37. https://doi.org/10.1002/ace.20276

Bennett N, Dunne E, Carre C (2000) Skills development in higher education and employment. SRHE/Open University Press, Buckingham

Boud DE, Keough RE, Walker DE (1985) Reflection: turning experiences into learning. Kogan Page, London

Burgess R (2004) Measuring and recording student achievement: Report of the Scoping Group. Available at: https://www.universitiesuk.ac.uk/policy-and-analysis/reports/Documents/2005/measuring-and-recording-student-achievement.pdf

Burgess R (2007) Beyond the honours degree classification: Burgess Group Final Report. Available at: http://www.hear.ac.uk/sites/default/files/Burgess_final2007.pdf

Carter L (2011) Ideas for adding soft skills education to service learning and capstone courses for computer science students. In: SIGCSE '11 Proceedings of the 42nd ACM technical symposium on computer science education. pp 517–522. https://dl.acm.org/citation.cfm?doid=1953163.1953312

CBI (2017) Helping the UK thrive: CBI/Pearson Employment and Skills Survey 2017. http://www.cbi.org.uk/insight-and-analysis/helping-the-uk-thrive/

CIHE (2010) The Fuse: Igniting High Growth for Creative. Digital and Information Technology Industries in the UK, CIHE, London

Cole D, Tibby M (2013) Defining and developing your approach to employability: a framework for higher education institutions. Available at: https://www.heacademy.ac.uk/knowledge-hub/defining-and-developing-your-approach-employability-framework-higher-education

CPHC (2017) GECCO evaluation report: building a graduate employability community in computing. https://cphcuk.files.wordpress.com/2017/08/geccoevaluationreport.pdf

CPHC/HEA (2015) Computing graduate employability: sharing practice. Available at: https://cphc.ac.uk/2016/01/08/cphchea-report-computing-graduate-employability-sharing-practice-published/

CRA (nd) Centre for recording achievement. Available at: https://www.recordingachievement.ac.uk

Dacre-Pool L, Sewell P. (2007) The key to employability: developing a practical model of graduate employability. Educ Training 49(4):277–289. https://doi.org/10.1108/00400910710754435

DCSF (2009) The work-related learning guide (Second Edition): A Guidance Document for Employers, Schools, Colleges, Students and their Parents and Carers. DCSF, London

Dearing R (1997) Higher education in the learning society. *Report of the National Committee of Enquiry into Higher Education*. HMSO, London

European Commission (1991) Memorandum on Higher Education in the European Community. Commission of the European Communities Task Force, Human Resources, Education, Training and Youth, Brussels, Nov

Fung D (2017) A connected curriculum for higher education. UCL Press, London, England

Fincher S and Finlay J (2016) *Computing Graduate Employability: Sharing Practice*. Project report. Council of Professors and Heads of Computing, Kent, UK https://cphcuk.files.wordpress.com/2016/01/computinggraduateemployabilitysharingpractice.pdf

Gibbs G (1988) Learning by doing: a guide to teaching and learning methods. Oxford Brookes Further Education Unit, Oxford

González-Morales D, Moreno de Antonio LM, Roda García JL (2011) Teaching "soft" skills in Software Engineering. In: IEEE global engineering education conference (EDUCON)—"Learning environments and ecosystems in engineering education", pp 630–637. http://ieeexplore.ieee.org/document/5773204/

Hawkins P (1999) The art of building windmills: career tactics for the 21st century. Graduate Into Employment Unit, Liverpool

Hazzan O, Har-Shai G (2014) Teaching and learning computer science soft skills using soft skills: the students' perspective. In: IGCSE '14 Proceedings of the 45th ACM technical symposium on Computer science education, pp 567–572. https://dl.acm.org/citation.cfm?id=2538885

Healey RL (2014) How engaged are undergraduate students in ethics and ethical thinking? An analysis of the ethical development of undergraduates by discipline. Student Engagem Experience J 3(2). http://dx.doi.org/10.7190/seej.v3i2.93

HEAR (2008) Higher education achievement report. https://www.hear.ac.uk

Helyer R (2015) Learning through reflection: the critical role of reflection in work-based learning (WBL). J Work-Appl Manag 7(1):15–27. Available at: www.emeraldinsight.com/2205-2062.htm

Jackson D (2014) Employability skill development in work-integrated learning: barriers and best practice. Stud High Educ 40(2):350–367. https://doi.org/10.1080/03075079.2013.842221

Knight P, Yorke M (2004) Learning, curriculum and employability in higher education. Routledge Falmer, London

Leckey JF, McGuigan MA (1997) Right tracks-wrong rails: the development of generic skills. High Educ Res High Educ 38(3):365–378. https://doi.org/10.1023/A:1024902207836

Lowden K, Hall S, Elliot D, Lewin J (2011) Employers' perceptions of the employability skills of new graduates. Project Report. Edge Foundation, London, UK

Office for Students (2018) Office for Students Strategy 2018 to 2021. Available online at: https://www.officeforstudents.org.uk/media/1435/ofs-strategy-2018-to-2021.pdf

Papert S, Harel S (1991) Constructionism. Ablex Publishing Corporation, New York

Peabody MA, Noyes S (2017) Reflective boot camp: adapting Lego Serious Play in higher education. Reflective Pract 18(2):232–243. https://doi.org/10.1080/14623943.2016.1268117

Pedley-Smith S (2014) Graduate recruitment, learning and development white paper. Kaplan UK. https://kaplan.co.uk/docs/default-source/pdfs/kaplan_graduate_recruitment_whitepaper40FEB3E55577.pdf

Pegg A, Waldock J, Hendy-Isaac S, Lawton R (2012) Pedagogy for employability. Available at: https://www.heacademy.ac.uk/knowledge-hub/defining-and-developing-your-approach-employability-framework-higher-education

QAA (2001) Guidelines for HE progress files, digital education research archive. Available at: http://dera.ioe.ac.uk/8480/1/progfile2001.pdf

QAA (2009) Personal development planning: guidance for institutional policy and practice in higher education. Available at: http://www.qaa.ac.uk/en/Publications/Documents/Personal-development-planning-guidance-for-institutional-policy-and-practice-in-higher-education.pdf

Schwaber K, Sutherland J (2017) The Scrum guide. The definitive guide to Scrum: the rules of the game. Available at: https://www.scrumguides.org/

Shadbolt, N. (2015) Unemployment among computer science graduates—what does the data say? HEFCE Blog. http://blog.hefce.ac.uk/2015/07/08/unemployment-among-computer-science-graduates-what-does-the-data-say/

Shadbolt N (2016) Shadbolt review of computer sciences degree accreditation and graduate employability. https://www.gov.uk/government/uploads/system/uploads/attachment_data/file/518575/ind-16-5-shadbolt-review-computer-science-graduate-employability.pdf

Strachan R, Liyanage L, Casselden B, Penlington R (2011) Effectiveness of technology to support work based learning: the stakeholders' perspective. Res Learn Technol 19:134–145. Available at: https://journal.alt.ac.uk/index.php/rlt/article/view/729/961

Thurner V, Böttcher A, Winter V (2012) Soft skill development along the education path. Evaluating expectations on and perceptions of student competencies in Software Engineering education. Int J Eng Pedagogy 2 (3):19–28. http://www.online-journals.org/index.php/i-jep/article/view/2148/2269

UKCES (2015a) UK employer skills survey 2015: UK results. https://www.gov.uk/government/publications/ukces-employer-skills-survey-2015-uk-report

UKCES (2015b) Sector insights: skills and performance challenges in the digital and creative sector. https://www.gov.uk/government/publications/sector-insights-skills-and-performance-challenges-in-the-digital-and-creative-sector

Universities UK (2016) Higher education in England: provision, skills and graduates. http://www.universitiesuk.ac.uk/policy-and-analysis/reports/Documents/2016/higher-education-in-england-provision-skills-and-graduates.pdf

Universities UK, UKCES (2014) Forging Futures: building higher level skills through university and employer collaboration. https://www.gov.uk/government/uploads/system/uploads/attachment_data/file/356749/FF_FinalReport_Digital_190914.pdf

University Alliance (2015) Mind the gap: engaging employers to secure the future of STEM in higher education. https://www.unialliance.ac.uk/2015/10/09/mind-the-gap-engaging-employers-to-secure-the-future-of-stem-in-higher-education-2/

Vogler JS, Thompson P, Davis DW, Mayfield BE, Finley PM, Yasseri D (2017) The hard work of soft skills: augmenting the project-based learning experience with interdisciplinary teamwork. Instr Sci 46(3):457–488. https://doi.org/10.1007/s11251-017-9438-9

Wilson T (2012) The Wilson review: a review of business–university collaboration. Available at: https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/32383/12-610-wilson-review-business-university-collaboration.pdf

World Economic Forum (2016) The future of jobs employment, skills and workforce strategy for the fourth industrial revolution. [Online] http://www3.weforum.org/docs/WEF_Future_of_Jobs.pdf

Yu L, Adaikkalavan R (2016) Developing soft skills by applying problem-based learning in software engineering education. In: Railean E, Walker G, Elci A, Jackson L (eds) Handbook of research on applied learning theory and design in modern education. Hershey, PA, IGI Global

Index

A
Academic integrity, 59–61, 63, 65, 67–70
Academic misconduct, 59, 61, 63, 65, 67, 69, 70
Academic performance, 4, 12
Active learning, 7, 17–20, 25–27, 215
Assimilate, 29, 135, 140
Attendance, 29, 38–41, 43, 45, 48, 118, 157
Automatic marking, 164

C
Cognition, 6
Collaboration, 33, 70, 134, 183, 184
Computing education, 10, 18, 21, 89, 127, 141, 153, 173, 176, 189, 191, 226, 230
Concept maps, 127, 129, 136, 144
Constructivism, 19, 215
Contract cheating, 59, 63, 64, 66, 68

D
Design thinking, 127–137, 143, 145
Distance learning, 32, 51, 53
Dynamic teaching practices, 147, 158, 159

E
Education, 3–5, 7, 10, 14, 17, 18, 53, 61, 63, 69, 112, 128, 129, 133, 136, 144, 145, 147, 161, 162, 189, 190, 203, 206, 208, 210, 211, 218, 231, 238
Employability, 13, 52, 66, 95, 187–189, 194, 196, 201, 203–209, 211–214, 216–218, 221, 222, 226, 229, 230, 232
Engagement, 10, 12, 20, 22, 23, 29, 31, 34, 39–42, 44–46, 48, 53, 127, 137, 144, 156, 196, 206, 208, 210, 217, 238
Enterprise showcase, 173, 174, 176–178, 182, 183

F
First programming languages, 108
Flipped classroom, 29, 30, 32, 33, 35, 45–48
Formative feedback, 53, 64, 142

G
Game design, 127, 129, 136, 137, 139–142, 144, 145
Group project assessment, 173, 187–189, 191, 194, 196, 198, 199, 201
Group work, 18, 173, 174, 180, 182

I
Inclusivity, 147, 159, 160
Information foraging, 3–5, 14
Intelligent systems, 51, 57, 148, 149

J
Java, 19, 24, 81, 86, 97, 105, 107, 108, 162, 164, 165, 168
JUnit, 161, 164, 167

L
Large lectures, 21
Learning and teaching strategy, 189
Learning edge momentum theory, 97

Learning resources, 33–35, 46, 60
Learning to program, 75, 76, 83

M
Motivating computer science students, 4, 13, 221
Multi-disciplinary, 135, 137, 174, 204
Multitasking, 3–6, 8, 9, 11–14

P
Plagiarism, 59, 60, 63, 68–70, 162, 175
Portfolios, 238, 239
Postgraduate, 53, 113, 212
Problem based learning, 20, 175
Problem solving, 9, 53, 86, 136, 174, 190, 222, 224, 225
Processing language, the, 97, 98, 104, 107

R
Reflective practice, 190, 221, 231, 233
Robotics, 51, 52, 55, 57, 86, 97

S
Soft skills, 174, 176, 215, 221, 223, 226, 228, 230, 240
Student group projects, 187–189, 191, 199, 201

T
Teaching programming, 88, 108
Team work, 64, 173, 191, 201

U
Unit testing, 162, 168

V
VLE, 24, 29, 31, 32, 34, 39, 40, 44, 45, 47, 48, 53, 157