Requirement Engineering Education in the UK, an Empirical Study

Russell Lock
Department of Computer Science, Loughborough University, Loughborough, United Kingdom
r.lock@lboro.ac.uk

Abstract

The quality of software is critically dependent on the quality of Requirements Engineering activities undertaken during software development. This paper outlines a survey of Higher Education (HE) institutions in the UK undertaken to determine the nature of the topics covered relating to Requirement Engineering, and the extent to which such topics are practically taught and assessed. Very few surveys of Requirement Engineering within HE have been conducted, and, to the authors knowledge this will be the first significant one published which focussing on HE in the UK. The paper concludes that a number of key issues exist in the UK provision for teaching in this area, which impact on the ability of industry to leverage the skills gained by students whilst studying at university.

Keywords: Requirement Engineering, Survey, Higher Education

1.0 Introduction

Within Software Engineering the concept of Requirements Engineering (RE) encompasses topics relating to the elicitation, analysis, documentation and ongoing management of software requirements, as well as its integration within software development methodologies. The effectiveness of RE as an activity is undoubtedly a major factor in the eventual cost of an IT systems [1], and a major cause of system failures [2, 3]. Indeed, Brooks famously stated that “The hardest single part of building a software platform is deciding precisely what to build” [4]. It is also clear that organisations are becoming considerably more cost conscious, with increased trending towards cost reduction measures in software development since the global recession started in 2008 [5]. The relevancy of the skillset of on-boarders graduating from Computer Science related degrees is therefore of increasing importance to industry.

The principle objective of the research is to investigate the state of Requirement Engineering within Higher Education (HE) for Computer Science in the UK, in order to determine to what extent it supports the needs of graduate employers engaged in in RE activities. In order to achieve this, a survey was developed for HE to determine whether RE was explicitly taught as a topic and, if so, the subjects it included, the depth of coverage, how it was taught, and how it was assessed.

The structure of the paper is as follows. Previous research relating to HE surveys and curriculum design will be explored, along with an examination of current best practice for the teaching of Requirements Engineering in Higher Education, from a pedagogical perspective, and in terms of expected content and depth. The methodology for the survey will then be discussed. The key results from the survey are then examined, followed by discussion on the limitations of the research. Finally, conclusions are drawn on the research conducted.

2.0 Background

Theory on the Diffusion of Innovations [6] emphasises the importance of five key characteristics in relation to the rate of uptake for technology: relative advantage, compatibility, complexity, trialability and observability. While HE cannot ensure comprehensive understanding of the five characteristics outlined by Rogers in relation to techniques in every industry, it can cover generic usage. It is also arguable that the strategic alignment of an organisation to the benefits of new approaches is an important consideration [7]. Through the material taught to undergraduate students, the skillset of graduate on-boarders can impact on the uptake of new techniques by industry. Further, beyond mainstream commercially accepted techniques the introduction of students to cutting edge processes / techniques can be achieved from a pedagogical viewpoint through the use of approaches such as Research Informed Teaching, which is explored later on.

The work of Boyle [8] concluded that Computer Science as a discipline is currently in a pre-paradigmatic state, a concept developed by Kuhn [9] , which may partially explain the opposing views held by academics on the content of the Computer Science Syllabus for HE. The differing opinions on Computer Science Syllabus design certainly cannot be considered a recent one, with Denning [10] reporting in 1985 similar conclusions relating to the lack of taught material on (then) new topics such as relational databases and the needs of industry. In terms of the overall syllabus for CS the comparative analysis of curriculum developments composed by Glass [11] in 1992 indicated recognition of the importance of both SE and RE as topic areas for CS from a relatively early point in the disciplines development. However, the majority of studies on CS syllabus design continue to place the most emphasis on programming language provision rather than on aspects of the syllabus relating to RE or SE in general.

Lethbridge [12] surveyed 168 software professionals to determine whether the university curriculum they were taught covered the materials needed for their current software engineering jobs. The study highlighted key areas of shortfall for teaching in Computer Science, namely: testing, quality assurance, requirements gathering and analysis, project management, user interface design and configuration management. All of these topics relate to Software Engineering, and many of them to Requirement Engineering. This paper highlights that many of these gaps in topic coverage still appear to exist within UK Higher Education nearly two decades later.

With regard to industry needs the work of Shaw [13] provides a fitting summary of one of the key dilemmas which face Computer Science Departments in teaching for the 21st century, that:

“Universities have long felt the tension between an internal value system that emphasizes education in enduring principles and the demands of employers who want focussed training in current technology”

Although very little research exists which focusses primarily on RE education the work by Macaulay [14] is particularly interesting as it combined small scale data collection from both academic representatives and industry. While the research itself was limited in scope, the depth of skillset expected by industry proved considerably more than that provided by the participating universities with the authors noting that in future “the education of a requirements engineer will require something considerably greater than a standard twenty hour lecture course”. However, in order to understand how and why topics are taught in RE it is necessary to look beyond the limited academic literature to the role of accreditation and standards bodies.

2.1 Accreditation and Guidance for Higher Education

Whether discretionary or mandated within a given country universities worldwide are encouraged to abide by the requirements placed on them by accreditation bodies, and to follow the guidelines put in place by standards bodies. The following sections explore firstly the influences on curriculum design from within the UK, then at the international influence on curriculum design.

2.1.1 The UK

Within the United Kingdom while HE institutions have freedom to develop and deliver the course material they wish, however, in reality they are guided by a number of organisations, including the QAA, and for Computer Science, largely national organisations such as the BCS (British Computer Society). One aspect of the QAA’s work is to publish *subject specific benchmarks* which are used as a source of information for curriculum development in HE institutions. The benchmarks indicate the core topics to be covered by UK HE institutions teaching in 59 subject areas, with compliance checked through periodic inspection by the QAA at an institutional level. The QAA benchmark for Computing [15] was last revised in 2007, and is currently in the process of being updated. Designed to provide a high level overview of topics expected to be covered, rather than a low level prescriptive one, the benchmark documentation as it stands has some notable issues, including:

- The fact they represent single honours Computer Science but only very briefly address the concept of joint honours programmes (ie programmes which combine multiple disciplines).
- The limited recognition of different types of single honours degree within the computing domain, such as specialisms in Software Engineering, Games development etc. (it is understood that these distinctions will be made in the revised version).
- The fact they are updated infrequently. For example, within the 2007 benchmark for Computing Grid Computing is listed as a developing technology, which has now been largely subsumed into the newer more advanced Cloud Computing paradigm.

The concept of Requirements arises nine times within the 2007 Benchmark Statement largely in terms of the concept, with specific mention of, elicitation, specification, constraints, types (functional & non-functional) and evaluation. The benchmark is entirely technique agnostic, for example, making repeated references to the importance of modelling as a concept without mandating the use a specific approach such as the Unified Modelling Language (UML). In summary, the Computing Benchmark is not designed to give guidance on specifics, and has relatively little content to guide the preparation of materials in this area. Unlike the ACM guidance which will be explored later, the benchmarks do not give guidance on the time spent on different topics, and as such raise significant questions relating to the balance between topics in a Computer Science degree.

The BCS accredit Computer Science related degrees, largely in the UK but also overseas. Accreditation is optional for HE institutions, with periodic inspections by nominated panels of BCS members. Unlike the QAA the BCS update their documentation, which outline subject areas and procedures [16] on a more regular basis, with the most recent version being June 2015. However, the core subject areas are aligned with the QAA subject benchmarks, and as such do not add additional guidance on suitable Requirements Engineering topics.

2.1.2 The Wider World

ABET [17] are one a key provider of accreditation for Engineering related topics within North America, with over 300 accredited courses in the Computer Science domain. ABET accreditation places emphasis on the process of Higher Education effectively delegating decisions relating to the applicability of different topics to discussions between the assessor and university in question. ABET curriculum specifications are therefore brief, with an emphasis on 'design' and 'modelling' skills.

While there are a number of initiatives within Europe aimed at ensuring equivalent standards for HE these are predominantly designed to provide quality assurance rather than prescriptive subject specifications, including the Bologna Process [18], ECTS (European Credit Transfer and Accumulation project) [19] etc, which are outside the scope of this piece of research.

In terms of guidance the main sources internationally are the combined ACM/IEEE Computer Science Curricula, last updated in 2013. The curricula decomposes topics into Tier 1 (core), Tier 2 (important) and Tier 3 (elective) categories in terms of the number of hours recommended tuition. While the topic of Requirements Engineering is mentioned within the curricula the number of hours associated with it would only support a small number of lectures rather than an entire module on the subject. Much of the material listed is elective in nature, with only one hour outlined at the core Tier 1 level, and three hours at Tier 2. The quote below provides the descriptor for the core Tier 1 material expected to be covered as part of a computer science degree.

"Describing functional requirements using, for example, use cases or users stories. Properties of requirements including consistency, validity, completeness, and feasibility" - IEEE/ACM Curricula Computer Science 2013

While the descriptor above may be considered very limited, it can however be noted that many of the topics which this survey has considered broadly part of Requirements Engineering have been listed under other headings within the curricula. For example UML, which has been classed under the separate heading of data modelling.

2.1.3 Summary on Accreditation and Guidance

Given the breadth of abstract high level guidance, and the lack of detailed guidance available to universities within the UK, HE institutions are reliant on the skillset of their individual staff members, limited industry input, and that of the content of current HE oriented textbooks to determine appropriate topics. It is therefore beneficial to gather data regarding how RE material is currently taught at universities to determine the effectiveness of the approach taken by the HE sector to teaching in this area.

2.2 Requirement Topics

In order to determine which topics were being taught relating to RE, and in what way they were being assessed, a list of common RE topics had to be compiled. This was achieved by analysing the sources of accreditation and guidance in the previous section, and through examination of the major topics covered by current RE/SE textbooks. The RE specific textbooks examined included the Volere technique oriented work by Robertson [20], Laplante's text [21] and the older text by Sommerville [22]. In order to determine which topics were most likely to be covered the authors did make an assumption that the majority of institutions were more likely to teach RE within combination modules that also covered SE and/or software development (an assumption which was supported by the results of the survey). As such the author would argue that lecturers would also be likely to use general SE texts. Therefore the list was supplemented by the RE topics covered by two longstanding SE textbooks, Sommerville [23] and Pressman [24].

3.0Survey Methodology

The survey data (the questions for which can be found in Appendix A), was gathered electronically, and was distributed by the CPHC (College Professors and Heads of Computing). The survey had a return rate of 43 respondents from a total of 104 potential universities. In all cases the departments represented were either Computer Science, Computing, Informatics or departments combining these with allied disciplines such as Engineering, Mathematics etc. In several instances, individuals from the same institution responded to the survey. Recipients were requested to direct the survey to the most appropriate member of staff in their department for completion. The most appropriate was deemed to be the member of staff with greatest responsibility towards assessment of Requirement Engineering within their department. Of the responses that represented multiple responses from a given institution:

- In twelve cases responses were discarded as clearly partially completed. The vast majority of these involved the completion of initial demographic question data but no further involvement (including both of the two responses from one institution).
- Contact details were available for two respondents from the same institution allowing resolution of the official response by email.
- In one case the same individual responded twice, in which case the later time-stamped response was counted.
- In three cases where the university was known, but contact details were not available the later time-stamped entry was counted.

This therefore left 26 institutions with responses that could be analysed further. Seventeen of the responding institutions reported information for their Computer Science/Computing degree scheme. A further seven reported their responses for a specialised Software Engineering degree. Of the remaining two one was Games related, the other Business related but taught by a Computing department.

The responding institutions were ranked to determine their distribution amongst Computer Science Departments in the UK, in order to check a suitable spread of responses had been received. Institutions were ranked using the Guardian 2016 subject league tables, one of the key league tables for universities within the UK. The presentation of the results shown in Figure 1 is deliberately coarse grained in order to ensure the anonymity of the universities who responded, in line with the nature of the survey. The results showed a slight skew, with the higher ranking quartiles being slightly better represented than the lower two quartiles. However, two responding institutions were unranked by the Guardian, and one was submitted anonymously.

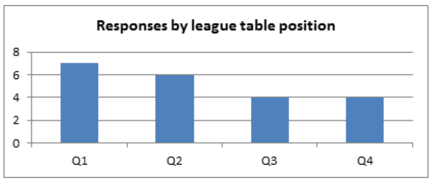


Figure 1: Responses by League Position as determined by the Guardian 2016 subject rankings

In terms of teaching approach within HE, the survey was designed to determine the mode of delivery for topics so, for example whether a given topic was being practiced within lectures, through coursework etc, or simply summatively assessed through assessment mechanisms such as exams. The survey results were processed through the IBM SPSS analytics suite, and, due to the monotonic, ordinal nature of the questions Spearman correlations were used for the correlations reported in the following section.

4.0Key Survey Results

The following subsections explore the results of the data analysis relating to five key themes which emerged from the results.

4.1 Dedicated Requirement Engineering Modules

Respondents were asked whether their degree scheme had a dedicated Requirement Engineering Module rather than teaching the subject within Software Engineering content, or taught in a decentralised manner throughout their programmes. 15 (58%) of the respondents reported dedicated modules for this topic. It is possible that the dedicated RE modules are more likely to be used within Software Engineering degree schemes (given their bias towards material in that sub-discipline), however a test for correlation between degree scheme and the use of dedicated RE modules showed a significant unexpected negative correlation between them (-0.52) in the data set. Information was collected within the survey relating to module titles under which RE material was taught, and while not complete in reporting (this question was optional), this showed that RE is more commonly taught under the module heading of Development, implying the bundling of Requirements-Design-Implementation within Software Engineering degree schemes rather than isolated standalone modules.

4.2 Hours of Tuition

Respondents were also asked how many hours of tuition were given for RE topics. The results showed significantly varying amounts of tuition ranging from 3 hours to 120 hours during a degree scheme with an average of 35 hours, which appears to be a slight improvement on the evidence gathered by the 1995 study by Macaulay [14].

While the number of hours used to teach RE related topics varies greatly between institutions, this arguably indicates discrepancies in determining the scope of RE activities within university teaching. The research by Macaulay indicates that some institutions consider transferable soft skills as part of their RE material.

4.3 Use of Computer Based Tools

All responding institutions required students to be able to write requirements. However, only 16 institutions (61%) expected students to be able to write requirements by hand, implying nearly 40% were unable to ask exam questions which required students to write software requirements. 18 (69%) used generic office software such as Excel & Word in documentation of requirements. Only 1 respondent had developed software especially for this purpose (a Volere template based electronic tool for requirement specification). 7 (26%) used industry standard tools such as IBM DOORS, Objectiver etc.

Schumann et al [25] showed that some segments of industry persist with generic tools such as Excel to document requirements. However, it is arguable that the ongoing dominance of such tools in teaching the new generation of Software Engineers is reinforcing this, rather than acting as a force for change within organisations.

4.4 Research Informed Teaching

The concept of Research Informed Teaching gained traction within academic pedagogical literature in the 1990s [26]. It promotes the use of research techniques within the classroom, and posits that undergraduate students may benefit from involvement in ongoing research activities, either through project work or through module coursework. Although not one of the key focusses of the study in question respondents were asked whether they, as the identified leader for teaching RE materials had a research background which covered these areas. Of the 29 who answer 12 stated they were working in this research area (Approximately 41%). Without further data the reason for this relatively low level of research expertise is open to interpretation, however the number of members of staff within computer science departments, coupled to the breadth of computer science as a domain of research is likely to be a factor. Regardless of the interpretation it does limit the level of Research Informed Teaching that can take place for RE teaching.

4.5 Specific Topic Areas

Respondents were asked the proportion of time spent on four key topics within Requirements Engineering, namely, Elicitation, Analysis, Specification, Verification and Management. The results showed a near even split between these for the respondents with approximately 20% of taught hours going to each of the four topics.

The topic areas generally associated with RE are therefore covered, however, the number of hours put to RE as a whole is likely to be a factor in the quality of learning for students on a given course, as could the techniques discussed/applied/assessed under these four headings.

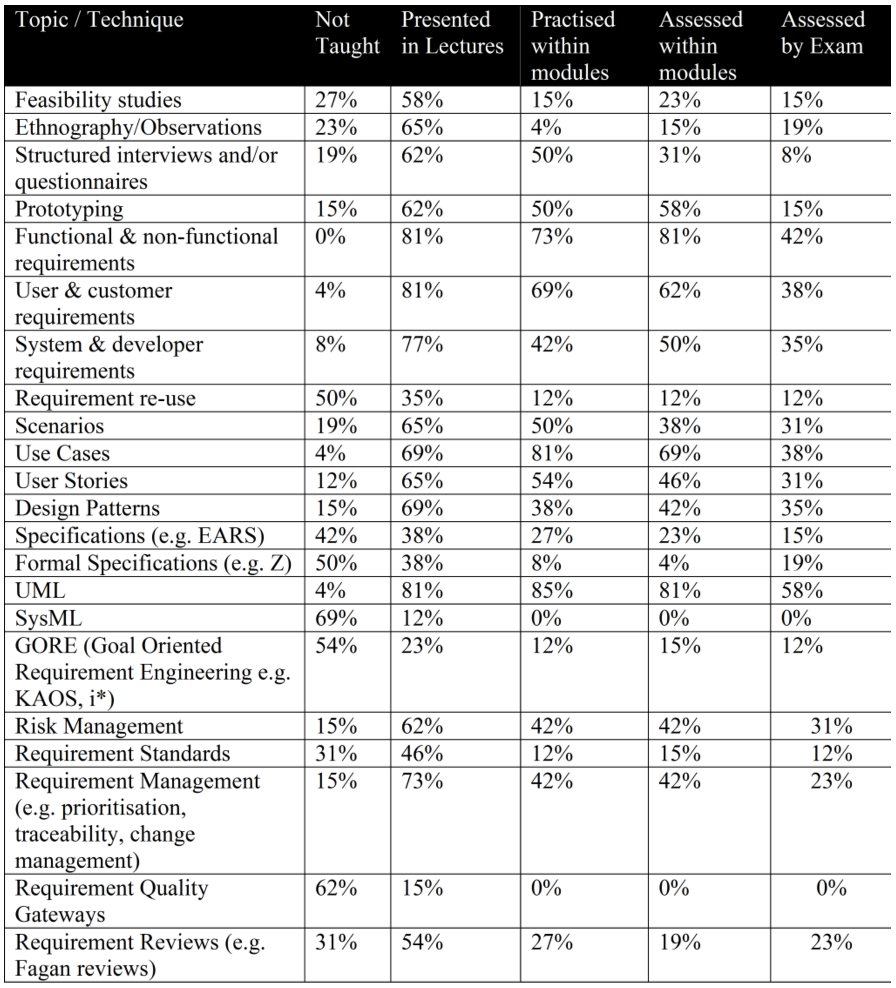
In order to understand which specific techniques were being used by universities in teaching information was requested on 22 specific techniques. Rather than rely on a Boolean measure of *taught* or *not taught* information was gathered on the way in which the topics were covered, applied and assessed. In order to develop these categories some assumptions/decisions did need to be made which are outlined below:

- Not taught: While the author expected that the majority of the topics would be covered by the majority of institutions at some level, some of the techniques listed represent alternative competing ways of working, and so it is not necessarily the case that lack of coverage represents a deficiency.
- Presented in lectures: Coverage in lectures does not preclude the possibility of background reading, or materials covered through tutorials etc, however these were not listed separately as categories in order to streamline the response time for respondents.
- Practised within modules: To indicate whether the material was covered in such a way as to promote levels of learning beyond basic recall, for example through small group work, coursework etc.
- Assessed within modules: To determine whether coverage of a given topic was also covered by assessment processes. While it may seem preferable for all topics to be assessed within modules, in practice some materials may be provided for background information, or delivered by guest lecturers in such a way as to preclude assessment.
- Assessed by Exam: Exams are commonly used within Higher Education, however, their nature does prevent the practical assessment of group activities by their very nature, which are common to many Software Engineering techniques.

The measures are discrete rather than ordered, though it would be unexpected if, for example a topic was assessed but not taught. While it would have been valuable to determine the amount of time spent on the different topics, especially in the case of complex model based techniques such as UML it was not felt that respondents would be willing to enter so much data.

The results revealed a number of interesting points, which are discussed further below. An aggregated summary table of the results from the 26 respondents, converted to the percentage of respondents is shown in Table 1.

Table 1: Processed survey results



Topic / Technique	Not Taught	Presented in Lectures	Practised within modules	Assessed within modules	Assessed by Exam
Feasibility studies	27%	58%	15%	23%	15%
Ethnography/Observations	23%	65%	4%	15%	19%
Structured interviews and/or questionnaires	19%	62%	50%	31%	8%
Prototyping	15%	62%	50%	58%	15%
Functional & non-functional requirements	0%	81%	73%	81%	42%
User & customer requirements	4%	81%	69%	62%	38%
System & developer requirements	8%	77%	42%	50%	35%
Requirement re-use	50%	35%	12%	12%	12%
Scenarios	19%	65%	50%	38%	31%
Use Cases	4%	69%	81%	69%	38%
User Stories	12%	65%	54%	46%	31%
Design Patterns	15%	69%	38%	42%	35%
Specifications (e.g. EARS)	42%	38%	27%	23%	15%
Formal Specifications (e.g. Z)	50%	38%	8%	4%	19%
UML	4%	81%	85%	81%	58%
SysML	69%	12%	0%	0%	0%
GORE (Goal Oriented Requirement Engineering e.g. KAOS, i*)	54%	23%	12%	15%	12%
Risk Management	15%	62%	42%	42%	31%
Requirement Standards	31%	46%	12%	15%	12%
Requirement Management (e.g. prioritisation, traceability, change management)	15%	73%	42%	42%	23%
Requirement Quality Gateways	62%	15%	0%	0%	0%
Requirement Reviews (e.g. Fagan reviews)	31%	54%	27%	19%	23%

On cursory inspection there may appear to be some anomalies in the data collected, seen with some of the less popular techniques, such as SysML, which show less than 100% if the not taught and taught columns are combined. However, there is an assumption that in a situation where the reader did not know the answer to the question they may have left it blank. These, non-responses have not been removed from the table in order that the reader can see the reported result from all respondents for any given technique. The effect of these non-responses is discussed in more detail below.

The majority of courses were found to include observational techniques such as ethnography (65%), however, the assessment of such techniques is noticeably weaker (34%). Strong industrial case studies are important in order to ground the theories put forward in HE [27], especially for approaches that rely on direct observation of real world issues not evident from existing documentation/models.

The concepts of functional and non-functional requirements are the only topics universally taught (these are explicitly listed in the 2007 QAA Subject Specific Benchmark for Computing) with UML, use cases and user requirements having near complete coverage across institutions. However, the complexity of UML raises concerns regarding depth of coverage when considering the average number of hours put to RE topics as a whole as discussed earlier. With the exception of UML, the Model Based Software Engineering (MBSE) approaches coverage was noticeably poorer, including approaches such as GORE. SysML in particular fared badly (69% of courses had no content on this), industry take-up of SysML has also been limited compared to UML. Further investigation into the data showed that SysML was not being used as a substitute for UML. Unless the teaching of MBSE broadens beyond UML the next generation of RE practitioners may have to rely on androgogical activities, self-interest / internal / external courses etc upon completing their on-boarding processes in order to leverage these approaches going forward. Unfortunately the shortfall in MBSE related teaching currently coincides with a global recession that has seen industry training budgets reduced or even removed unless mandated by regulatory or legal bodies [28].

One of the key roles of HE is to set the students on the path to best practice, and it is therefore surprising that the role of standards in RE does not comprise a part of nearly 1/3rd of the courses investigated. The results also indicated a relatively low level of teaching (half of the courses contained none) relating to re-usability, which is a concern given the recent studies on how likely organisations are to actively re-use requirements for new or modified systems [29, 30,31]

The results overall showed a role for both coursework and exams in teaching RE, however there was a clear bias in favour of coursework. However, in cases of practical techniques such as UML, prototyping and scenarios lecturers do make use of lecture based exercises in addition to coursework. For subject matter where small scale lecture based exercises cannot be constructed as easily, such as promoting standards and requirement reviews this material was mostly presented in lecture format.

In summary, it is clear that the basics of RE are being covered almost universally, however emphasis on the role of standards and reuse are weak. There may be implications for practice in terms of forming “bad habits” (ignoring standards) as well as under-emphasising the re-use organisations commonly undertake. It is also clear that when considering model based specifications there may be neither the time nor the appetite to teach languages other than UML. This in turn hinders the coverage of more recent academic advances in RE, such as the GORE based techniques which have only seen limited industrial use so far.

5.0Limitations and Future Work

While the survey was distributed to all relevant computer science departments by an established body within the Computer Science HE domain (the CPHC) the return rate was still lower than expected. As a result although the results themselves have been analysed and are valid for the responding institutions, the return rate, combined with the limited number of UK Computer Science departments (104) do mean that statistical confidence for the research findings in terms of application to the wider group of computer science departments in the UK is not ideal with a potential 16% margin of error at a 95% confidence level. One of the steps taken to increase confidence in the results shown was to map the responding institutions in terms of league table position, as outlined in the methodology section.

As part of the survey response respondents were asked if they would be prepared to take part in future communication on this issue. As a result a number of contacts have been made which could be leveraged through the development of a suitable focus group. The results also point to interesting issues relating to the need of industry to retrain incoming graduate employees to use specific software, and newer paradigms such as GORE. As such an additional survey will be created for industry to follow up these points.

6.0Conclusions

In conclusion, the results of the survey highlight a number of potentially interesting issues, including the limited use of industry RE tools in use for teaching purposes, and the ongoing dominance of UML for modelling in both HE. The results also showed that the QAA 2007 subject benchmark for computing was adhered to, but that it was insufficient by itself in providing outline guidance on RE topics. The future of Research Informed Teaching in this area is also hampered by the niche nature of this topic of research. Given the importance of RE activities in keeping projects on time and in cost HE appears to be providing a broad but relatively shallow depth of teaching in this area with considerable variation between institutions.

7.0 Acknowledgments

The author would like to thank the assistance of the CPHC (Council of Professors and Heads of Computing), and Prof Sally Fincher in particular for helping to disseminate the RE survey.

8.0 References

1 Boehm B, Basili VR (2001) Top 10 list [software development]. Computer (Long Beach Calif) 34:135–137
2 The STANDISH Group (2013) The CHAOS Manifesto 2013.
3 Dorsey P (2010) Top 10 Reasons Why Systems Projects Fail.
4 Brooks FP. J (1987) No Silver Bullet Essence and Accidents of Software Engineering. Computer (Long Beach Calif) 20:10–19

5 Derksen B, Luftman J (2014) European key IT and Management Issues & Trends for 2014.
6 Rogers EM (2010) Diffusion of innovations. Simon and Schuster
7 Rifkin S (2001) Why software process innovations are not adopted. IEEE Softw 18:112–111
8 Boyle R, Carter J, Clark M (2002) What Makes Them Succeed? Entry, progression and graduation in Computer Science. J Furth High Educ 26:3–18
9 Kuhn T (2012) The Structure of Scientific Revolutions, 50th Anniv. The University of Chicago Press
10 Denning PJ (1985) Ruminations on Education. Computer (Long Beach Calif) 18:105–110
11 Glass RL (1992) A comparative analysis of the topic areas of computer science, software engineering, and information systems. J Syst Softw 19:277–289
12 Lethbridge TC (1998) A survey of the relevance of computer science and software engineering education. In: Proc. 11th Conf. Softw. Eng. Educ. IEEE Comput. Soc, pp 56–66
13 Shaw M (2000) Software engineering education. In: Proc. Conf. Futur. Softw. Eng. - ICSE '00. ACM Press, New York, New York, USA, pp 371–380
14 Macaulay L, Mylopoulos J (1995) Requirements engineering: An educational dilemma. Autom Softw Eng 2:343–351
15 (QAA) TQAA for HE The Quality Assurance Agency for Higher Education (QAA).
16 British Computer Society Academic accreditation | Qualifications and Certifications | BCS - The Chartered Institute for IT. http://www.bcs.org/category/5844. Accessed 7 Aug 2015
17 ABET (2015) ABET. In: ABET. http://www.abet.org/. Accessed 7 Aug 2015
18 EHEA Documents | Main documents | EHEA. http://www.ehea.info/article-details.aspx?ArticleId=73. Accessed 10 Aug 2015
19 European Communities (2004) European Credit Transfer and Accumulation System (ECTS): Key features. In: 2004. http://ec.europa.eu/education/ects/ects_en.htm. Accessed 10 Aug 2015
20 Robertson S, Robertson J (2012) Mastering the Requirements Process: Getting Requirements Right. Addison-Wesley
21 Laplante P (2013) Requirements Engineering for Software and Systems, Second Edition - CRC Press Book, 2nd ed. CRC Press
22 Sommerville I, Sawyer P (1998) Wiley: Requirements Engineering: Processes and Techniques - Gerald Kotonya, Ian Sommerville, 1st ed. Wiley
23 Sommerville I (2015) Pearson - Software Engineering, 10/E - Ian Sommerville, 10th ed. Pearson

24 Roger P (2014) Software Engineering: A Practitioner’s Approach: Roger Pressman, Bruce Maxim: 9780078022128: Amazon.com: Books, 8th ed. McGraw-Hill Education

25 Schumann H, Wendel H, Braukhane A, Berres A, Gerndt A, Schreiber A (2010) Concurrent systems engineering in aerospace From excel-based to model driven design. Proc. 8th Conf. Syst. Eng. Res.

26 Elton L (2010) Research and Teaching: Conditions for a positive link. Teach. High. Educ.

27 Ramsden P (2003) Learning to Teach in Higher Education. Routledge

28 Israilidis J, Lock R, Cooke L (2012) Ignorance management: An alternative perspective on knowledge management in multinational organisations. In: Proc. Eur. Conf. Knowl. Manag. ECKM. pp 493–501

29 Daneva M (1999) Measuring reuse of SAP requirements. In: Proc. 1999 Symp. Softw. reusability - SSR ’99. ACM Press, New York, New York, USA, pp 141–150

30 Nuseibeh B, Easterbrook S (2000) Requirements engineering. In: Proc. Conf. Futur. Softw. Eng. - ICSE ’00. ACM Press, New York, New York, USA, pp 35–46

31 Frakes WB (ed) (2000) Software Reuse: Advances in Software Reusability. doi: 10.1007/b75206