Educating Software Engineers of the Future: Software Quality Research through Problem-Based Learning

Ita Richardson, Louise Reid
Dept of Computer Science & Information Systems and Lero – the Irish Software Engineering Research Centre, University of Limerick, Ireland
ita.richardson@lero.ie

Bob Pattinson, Yvonne Delaney
Management Development Unit Kemmy Business School University of Limerick, Ireland
yvonne.delaney@ul.ie

Stephen B. Seidman
College of Science Texas State University, U.S.A.
seidman@txstate.edu

Abstract

Software Engineering graduates are expected to enter the workforce with both technical and soft skills. In addition, software quality is a topic that is becoming increasingly important both because of educational and industry requirements. Software engineering lecturers need to bring their research into the classroom. Bringing all of these together can pose the lecturer with a dilemma that is not easily solvable. This paper presents how problem-based learning, a pedagogical methodology that is popular in medicine and other disciplines, can be used to accomplish these goals in a single course module. It describes a research project which analyses the implementation of problem-based learning within a M.Sc. Software Engineering Quality Module, and evaluates the outcomes against published expectations.

1. Essential Skills for Software Engineers

In the four decades since the first use of the formulation "software engineering" [1], its meaning has expanded from a powerful metaphor to designate a new discipline of research, practice, and education. Many writers have sought to capture the broader meaning of software engineering [2][3][4]. Their definitions emphasize that software engineering is an engineering discipline with foundations in computer science and mathematics, implying that software engineering education must give students an engineering perspective along with a solid technical background. Software engineers must develop designs by working within specified constraints, considering alternatives, and trading off costs and benefits. Their required technical background must also include a solid foundation in computer science and discrete mathematics. All engineering programs must prepare students for life as professional engineers, requiring that students become familiar with principles of ethical practice. For software engineers, these are embodied in the ACM/IEEE-CS code of ethics [5]. It is also critical that engineers leave university with the "soft skills" expected by employers: oral and written communication, interpersonal and teamwork skills, motivation and initiative, honesty and integrity, and strong work ethic [6]. However, several authors (e.g., [7] [8] [9]) argue that even though employers are often happy with graduates' technical skills, they are less persuaded that their soft skills are at a sufficiently high standard.

2. Software Quality in the Software Engineering Curriculum

As any effort to develop a software system must pay attention to quality issues, software quality has become an essential part of software engineering practice, now one of the ten knowledge areas within the IEEE Computer Society's Software Engineering Body of Knowledge [10]. Therefore, software quality has become a key topic in undergraduate and Masters' curriculum recommendations for software engineering ([6], [10]), and it appears in software engineering and computer science curricula. For example, in the ACM/IEEE-CS undergraduate software engineering curriculum recommendations [6], software quality includes the quality of work products such as functionality, usability, reliability and safety, and the quality of the work processes used to develop and modify these work products. Similarly, the recent GSwE recommendations [11] for software engineering Master's degree curricula are built around a core body of knowledge (CBOK) that includes software quality, verification and validation, and quality management processes. Given these curricular recommendations, topics covered in university courses and modules on software quality often deal with both aspects of quality. Process quality issues are embedded in IEEE and ISO-IEC quality standards, so the elements of these standards (ISO 90003, CMMI, IEEE 760, IEEE 1061) are usually covered in quality courses, as is process assurance. Product quality can be treated via standards (ISO/IEC 9126), and techniques for product assurance are usually covered, particularly defect prevention, quality metrics, and software testing.

However, the presence of a topic in a particular curriculum gives no guidance as to how the topic should be taught. The traditional approach to university-level instruction involves a lecturer standing in front of students, with the course content divided into a number of topical lectures. The instructor generally tries to stimulate student discussion of the material. This is often difficult! The lecturer will also assign relevant readings – normally text books or research papers. Students are examined on their understanding of the topic and/or an individual or group project for which they are awarded a grade.

There are several major problems with this approach to teaching. First, despite efforts to encourage student interaction, it is fundamentally passive. Second, in the specific area of software engineering, this mode of instruction seems very distant from the goal of designing and constructing software systems. Finally, it is disconnected from the industrial experience of many students in software engineering programs.

These objections apply equally in many disciplines that are oriented to practice. One of the most significant attempts to find a new modality of instruction is problem-based learning (PBL), which was first developed in the teaching of medicine [12]. Additionally, as research becomes increasingly important within universities, and funding becomes scarcer, there is an increasing requirement to ensure that students are gaining in knowledge and education from research projects. One of the authors (IR) has been teaching software quality for the past 10 years, and, in her recent implementation of PBL, she has given students the opportunity to be educated in software quality knowledge and to develop their soft skills. The problem which she has used is based on healthcare systems, and has resulted from her involvement in a European Union funded project, TRANSFoRm.

3. The TRANSFoRm Research Project

TRANSFoRm, Translational Medicine and Patient Safety in Europe, is a project funded under the European Union's Framework 7 program. The purpose of this project is "to develop a 'rapid learning healthcare system' driven by advanced computational infrastructure that can improve both patient safety and the conduct and volume of clinical research in Europe" [13]. Within this, Lero – the Irish Software Engineering Research Centre is developing and evaluating a quality assurance framework for healthcare software (e-Health). Data must be secure, accurate and timely so that patients and clinicians can believe and use available and secure computerised data. Additionally, there are regulatory requirements to

be followed. Due to Lero's involvement in TRANSFoRm, we have commenced research with local hospitals and are working with hospital personnel to ensure that patient care is not compromised due to a lack of software quality. Investigating where underlying problems exist, we develop solutions for implementation and evaluation with the hospitals, ultimately developing an evaluated e-Health quality assurance framework.

Software is becoming more pervasive throughout healthcare. It is embedded within medical equipment, enables storage of patient records and creation of diagnostics, and is used to schedule appointments. The increasing pervasiveness of e-Health systems gives rise to a requirement to 'keep the software safe'. Often, the non-functional requirements of e-Health systems seem to focus on privacy and security rather than data integrity. When data is inaccurate, "reliance on information systems can be dangerous" [14]. This can give rise to a general lack of trust in e-Health systems "since possible errors may cause serious problems to patient health" [15]. Improved data quality will provide benefits to patients, particularly as "accurate clinical data" is needed for "high standards of care and monitoring" [16]. Indeed, "a hospital would be better off not having certain data, than having inaccurate data especially if those relying on the data are not aware of its inaccuracy."[17]. In our primary research within TRANSFoRM, we observe that quality standards, although relatively commonplace for medical devices, are not always implemented during e-Health systems development. This is despite awareness that "computerisation can significantly increase the risk of poor data leading to misinterpretation and adverse events" [18]. Some of the challenges faced by e-Health systems include adequate data definition, collection and confidentiality [19]. Despite the recognition of these issues, little has been done to find solutions.

Improvement of data integrity can improve healthcare practices. When data relating to cardiac arrest was reported uniformly, it improved understanding of the elements of resuscitation practice [20][21]. We argue that software quality practices must be implemented to match the increase in software usage. However, the development and management of clinical software is challenged by size, complexity of practice, parallel management structures, and resistance to change [22]. For example, when software systems are developed internally to manipulate medical device output, they conflict with the regulatory activities that have been used in the medical device manufacturing process. Additionally, software systems are often not seen as being governed by the health standards used in Irish hospitals [23]. We suggest that health standards will need to be modified to take account of software standards.

4. Teaching Software Quality

PBL was implemented in the Software Engineering Quality module within the M.Sc. in Software Engineering at the University of Limerick. Fourteen full-time and part-time students, some with industry experience, some recent graduates, participated in the module. In addition, 6 were from Asia and are not native speakers of English. The variety of student backgrounds had the potential to cause problems when teaching the module.

4.1 Previous implementation of module

The module was previously implemented as a two-hour lecture per week over 12 weeks, with a 15 minute break after 50 minute. The lectures were generally presented on powerpoint slides with occasional guest lecturers. Although discussion was encouraged, the lecturer did most of the talking. There was even less inter-student interaction. Students rarely read assigned papers. Overall, this resulted in uninteresting classes for all involved. Students found it difficult to learn software quality concepts and completed the class without an in-depth understanding of what is really meant by 'software quality'! Assessment for the module was divided between a team project (40%) and final exam (60%). The project involved self-selected groups working together outside of class time, and presenting a final

paper at the end of semester. The paper was not discussed in class, and any group learning was not shared with the rest of the class. There was no record of individual involvement in the project, nor was individual's participation indentified. The project was normally a case study which required the students to learn about a domain outside of their interests. The final exam dealt with concepts presented in class and while those that did reading outside of class performed well in the exams, there was no incentive for students to research topics on their own. No advantage was taken of student background and experience.

4.2 Introducing Problem-Based Learning

In an effort to overcome these difficulties and focusing on finding a solution to the problem, IR implemented PBL in her M.Sc. in Software Engineering software quality module, during the 2009-2010 academic year. The intention was to:

- Provide students with an understanding of software quality concepts;
- Provide students with soft skills such as teamwork, communication and problem solving;
- Introduce research which was being undertaken within Lero and demonstrate how this could potentially be useful to students' in the future.

In addition, one of the authors, YD, is researching the implementation of PBL in the classroom situation. The software quality class would become one of her case studies.

5. What is problem-based learning?

PBL is an approach to learning [24] which constructs and teaches courses using problems as the stimulus for student activity [25] [26]. This is further reiterated by [27]: "PBL is characterized as an approach to learning in which students are given more control over their learning than a traditional approach, and asked to work in small groups, and most importantly acquire new knowledge only as a necessary step in solving authentic, ill-structured, and cross-disciplinary problems representative of professional practice." PBL allows lecturers to meet educational and industry-specific objectives.

Real situations are used to present challenging operational problems which generate learning outcomes that incorporate the professional knowledge, skills and competencies required by graduates. Students take responsibility for their own learning; they develop critical thinking, self-directed learning, leading teams, problem solving, decision making, and communication. Schmidt [28][29] and Coles [30][31] stress the importance of linking theory with practice. This allows students to make connections, linking prior knowledge or learning with new knowledge. For example, they suggest that students should participate in activities such as group problem solving, preparing and presenting papers, and reflection. In doing this, PBL presents several critical shifts from traditional educational models [32]. The PBL goal is long-term learning that results in behavioral change and not just conceptual mastery [33].

A major study by Walker and Leary [34] concluded that PBL students either did as well as, or better than, their lecture-based counterparts. This analysis suggests that students are not being disadvantaged by the implementation of a new process [35]. The need to prepare graduates for professional life is reviewed in Armarego [36], who states that PBL holds great promise for preparing students to operate in environments where the development of new technologies and the increasing rates of changes in levels of complexity has resulted in organisations requiring multi-faceted employees. The process of collaborative self-directed, authentic learning, characterised by problem-finding, problem-solving, reiteration and self-evaluation distinguishes true PBL from similarly named methods that use a problem of any sort somewhere in the teaching/learning sequence [37]. PBL makes students better able to develop competencies in the affective learning dimension as they gain a more balanced understanding of complex situations. Furthermore, affective learning serves as a critical link

between cognitive and behavioural learning that motivates students and enhances educational outcomes.

6. Introducing PBL to the Software Quality Module

Critical to the success of the PBL curriculum, is the development of a good problem. For the software quality module, the problem needed to be engaging and interesting to the students, motivating them to look for a clear and deep understanding of the concepts and relating to a familiar situation. The TRANFoRM e-Health software quality research was identified early in the module design process as a source. e-Health topics have the advantage of being personalized quickly and easily and would fulfill each of the criteria. This also had the advantage of bringing Lero's research to the postgraduate students. The problem trigger in Figure 1 was presented to the students during the second week of the module.

TRIGGER: *Students viewed a video (accessed November 2010):*
http://www.youtube.com/watch?v=-xrrk-XhgVc *and were then asked to:*
Develop and write the software quality plan for a hospital.

Figure 1. Problem presented to the students

This trigger is interesting for a number of reasons. As the video commences, a patient is taken in from ambulance on a trolley into a hospital. This is the last time we see any patient. The focus is on hospital computer hardware systems, such as bedside monitors, and on staff discussions around computer screens. This video heightened students' awareness of the presence of computing equipment within hospitals today. Students' initial discussions highlighted that where there is hardware, software is also present. From this point, they started to focus on the software quality required by safety-critical healthcare systems.

PBL's introduction led to changes in class organization. Although the classes continued with the two-hour lecture format, the lecturer split the students into groups of four. Given the international make-up of the class, each team had at least 2 Asian student members. During each scheduled session, students joined their groups immediately to work on the problem. Only one student in the class had previously participated in a PBL module. The lecturer now served as facilitator. She circulated between the groups, discussing issues that arose, ensuring that all groups worked towards a relevant software quality plan by directing them towards relevant research. This facilitation allowed students to benefit from the lecturer's knowledge. On occasion, she gave 10-15 minute lectures on specific topics. For example, one lecture ensured that students understood the characterisation of processes as Organisation, Management, Engineering, Customer-Supplier and Maintenance processes, thus removing the exclusive focus on Engineering processes. Additionally, at the end of class, group discussions were summarised during a short 5-10 minute discussion with all students.

During group discussions, as in PBL theory, students filled specific roles: discussion leader, recorder, observer and team member. They kept minutes of meetings and reviewed these each week. Actions from the previous week were discussed and students circulated papers that they read since the previous session. They had internet access and freely viewed papers or other information they needed during the meetings. Some groups stayed in the classroom to conduct meetings; others moved to the adjacent café to hold their meeting and discussion. At the end of each meeting, actions for the following week were distributed.

The knowledge which IR had gained from her TRANSFoRm research was continually being used to subtly guide the students. She discussed various guidelines and health related software quality initiatives. To ensure an understanding of quality requirements, and to give students an insight into the hospital quality system, a subject-matter expert (SME) visited the class after they had researched the problem for 4 weeks. This SME, a co-author of this paper

(LR), a PhD student on TRANSFoRm, oversees quality in the local hospital. She gave a very short presentation, after which students questioned her for 90 minutes about software processes and data quality within the hospital. She imparted an understanding of the importance of software quality in e-Health to the students and gave them some insights into TRANSFoRm research.

The module was continually assessed; there was no final exam. Each group prepared a paper[1] worth 25% of the module. Presentations to the full class – once half-way through and once at the end of the module – were worth a total of 25% of the grade, with 50% of this allocated to individual presentation skills. Ten percent was allocated to participation within their group during class time and 10% to four individual oral exams. An individual portfolio was submitted, worth 30%, which included summaries of papers read, a personal reflective journal, meeting minutes, and an outline of individual project participation.

7. Research Methodology

Prior to this module, IR's only previous PBL experience was with a second year undergraduate class [38]. For the software quality module, IR compiled a reflective journal within 24 hours of each weekly class. Participating students maintained an assessed reflective journal. Students were surveyed via e-mail about 70% of the way through the module. They were presented with open-ended questions in which they were asked to discuss the positive and negative experiences arising from their attendance. During module delivery, YD observed the implementation of PBL within the class, which allowed her to collect data about student and lecturer participation. In addition, she followed this up by interviewing IR, providing an insight into the reflections of the lecturer. All data collected was analysed using content analysis.

8. Research Results

Student learning changed and student knowledge increased as a result of implementing PBL. The students were now part of a more interactive environment, particularly at meetings, where they disseminated new knowledge learned. This gave them a real sense of solving a problem, even knowing that they …. *have put in more work…*[2]. Student attendance improved and was consistent at almost 100% as students were conscious that they would disrupt their group if they were unable to attend. Students worked consistently, and groups made steady progress on their software quality plans. Students were given regular feedback from the lecturer by discussion and from the immediate availability of assessment results.

8.1 Software Quality Knowledge

The interactivity in the classroom created an interesting and innovative situation where prior industrial, medical and cultural experiences play a role in student learning. From the mid-semester presentations, it was obvious that student knowledge has increased, demonstrating an understanding of software quality concepts and standards which had not been observed in previous incarnations of this module. Students themselves recognize this: *Personally, I believe that I have learnt more through PBL in the first 8 weeks than I would have in a standard classroom environment.* Student presentations, written papers, and final results, have demonstrated that students have learned more about software quality than in the traditional mode.

---

[1] Based on the output from one group, TRANSFoRm researchers have subsequently had a short paper published [38]

[2] Quotes from interviews are in italics.

8.2 e-Health and TRANSFoRm

The PBL methodology required students to review academic papers. While this is an expectation for a module at this level, it has not generally been achieved in the past. The incorporation of results from the literature ensured that students understood the strengths and weaknesses of e-Health quality systems. In particular, students now understand that software has become increasingly important for patient treatment, which in turn which has increased the need for software quality systems to be used in e-Health development. They acquired a deep insight into a European Union funded research project, TRANSFoRm, which they would not have received in the traditional lecture methodology. Their understanding of the role played by software quality within e-Health has made students aware of the essential role played by software quality in safety-critical system. They can also now see that software quality is important in what might be considered to be 'less-serious' e-Health domains.

8.3 Transferrable Skills

Table 1 compares the skills cultivated by PBL with and the soft skills expected from graduate software engineers and with two management skills models [39] [40].

PBL Tutorial	Software Engineers Skills (Section 1)	Skills of Executives [39]	Management and Leadership Skills [40]
Communication	Communication Interpersonal	Communication	Social skills
Problem solving		Problem solving	Problem solving
Decision making		Decision making	Decision making
Teamwork	Teamwork	Teamwork	
Time and task management	Strong work ethic	Managing time and stress	Time and task management
Leadership influencing	Motivation	Motivating and influencing	
		Self awareness	Self awareness
Self-directed learning	Ethics	Setting goals and articulating a vision	
Critical thinking	Honesty and integrity	Managing conflict	
		Delegating to others	

Table 1: Core Soft Skills Matrix

PBL provided students taking the software quality module with important soft skills. We will begin by discussing the soft skills called for by software engineering curricula. With respect to **communication**, students presented their work to the class twice during the module. Students learned to interact and draw from each other's prior experiences. We observed this during team meetings when they were actively volunteering to present their data; these activities contributed to the development of communication and **teamwork** skills. We also observed that teamwork developed as students took on ownership of their personal contributions. Student attendance improved as the semester progressed. Required work allocations were e-mailed to all members. The improved **work ethic** was obvious throughout the semester. Group tasks were completed between classes. Groups also had regular contact and discussion over the internet. This resulted in steady progress; students were aware that they …. *have put in more work*…. The overall workload undertaken, output produced and interest in the software quality e-Health topic could not have existed without the **motivation**

of the individual students. For example, they arranged team meetings outside of class time independently of direction from the facilitator. This was paramount to the success that each student had with the module. We saw no explicit evidence of **Ethics** or **Honesty and integrity** being fulfilled through the use of PBL.

The management skills models [39] [40] indicate that PBL can provide other required skills to graduates. Students' **problem-solving** skills improved, as they belonged to *a more interactive environment, actively participating in the meetings, and providing new knowledge acquired between meetings.* This gave them *a real sense of solving a problem*, and they were learning from each other in a *student way*. They have realized that the PBL experience will be *beneficial in industry as we are expected to work on our own, look at the problem and provide solutions for it.* As part of students' **decision-making** process, PBL automatically led them to review academic writings. They drew on their academic reading to support decisions taken. They demonstrated decision-making skills in organization of the discussion groups, role and work allocation. This can also be seen to help improve their **leadership and management** skills. Minutes and action items were recorded, research papers were discussed and decisions were taken. Allocating workload and managing time expectations for problem completion improved t**ime management** skills.

9. Discussion

The aims of introducing PBL to the software quality module were to improve students' learning experience while reinforcing the soft skills required in software engineering and computer science courses. The module was also used to introduce a research project to the M.Sc. students.

PBL works particularly well if a good problem is presented. The problem used was based on the ubiquity of software within e-Health. It effectively highlighted the need for software quality, while giving the students a familiar domain. Another advantage of this problem was that it aligned the students with the TRANSFoRm project, bringing them into a research situation with which they were unfamiliar. From a pedagogical perspective, PBL can be considered a threshold concept, a troublesome space for learners [41], but once the students understood that PBL provides a framework, things settled down and real learning began.

Novices and experts address problems in different ways. Gijselaers and Woltjer [42] state that "novices tend to organize their knowledge representations around the specifics of the problem, whereas experts move to the more abstract level to see the general principles". The way this group of PBL novices operated supports this theory. The students requested more direction and identification of the learning outcomes. While it is difficult for a lecturer (now facilitator) to do this, it was important to provide them with only the information that is critical to get them started. This well-structured problem allowed them to reach their learning outcomes independently.

In this software quality module, the PBL tutorial process was followed. The students worked in groups facilitated by the lecturer. Students benefited from experienced and informed input and were given relevant guidance. Additionally, the SME who met with the class is experienced in quality for e-Health systems and a part-time PhD student on the TRANSFoRm project. Following the PBL scenario, students had a very interactive discussion session with LR. Observing this, it was clear that the problem was well-understood by the students, mainly because of the familiarity of the domain.

The development of soft skills in the PBL process is almost like a hand in a glove. It is practically impossible to develop one skill without impacting another. This overlap was evident in the group activities, where students that were unable to attend class communicated with their team, conscious of the fact that non-attendance would disrupt the team dynamics. A similar overlap was evident when the students took on the team roles of discussion leader,

recorder and observer. They demonstrated the mutual development of their leadership and time management skills by allocating tasks and setting deadlines.

Problem-based learning was a positive experience for both the students and the lecturer involved. From the lecturer's perspective, it provided a different approach to teaching which is both interactive and interesting. It provided students with the experience of solving a real-world problem and the opportunity to understand software quality and its effects. Students achieved defined learning outcomes and gained soft skills that will be important to them in their software engineering careers. In addition, students were given an active insight into a research project.

10. Conclusion

This paper reports on the use of PBL in a software quality module within a software engineering course. While the experience was clearly positive for both student and lecturer, the broad application of PBL to software engineering modules requires that the experience reported be repeatable and generalizable. Repeatability requires the ability to generate interesting and useful problems. We would like to see such problems arise from existing research projects, and we have learned that it is important to situate the problems in a context which is known to the student. This is not always an easy task. Software engineering lecturers need greater understanding and awareness as to the structure of the PBL process. Therefore it is important for those introducing PBL to their classes to be aware of relevant literature and developments in this discipline. The larger success of PBL in software engineering education depends on the further development of PBL problems and concepts for use within this discipline. It also depends on the usefulness of the outcome within industry. This is something which requires research that we have not yet undertaken.

Acknowledgment

The authors would like to thank the students who participated in this module and provided us with feedback. This research was partially supported by Science Foundation Ireland grant 03/CE2/I303_1 to Lero - the Irish Software Engineering Research Centre (www.lero.ie) and by the European Commission – DG INSFO (FP7 247787) through TRANSFoRm.

References

[1] Naur, P. and Randell, B. (eds.), (1969) Software Engineering: Report on a Conference Sponsored by the NATO Science Committee, (7 – 11 October 1968), Scientific Affairs Division, NATO, Brussels.
[2] Bauer, F.L., "Software Engineering", Information Processing, 71, 1972.
[3] Ford, G., (1990) 1990 SEI Report on Undergraduate Software Engineering Education, CMU/SEI-90-TR003.
[4] IEEE STD 610.12-1990, IEEE Standard Glossary of Software Engineering Terminology, IEEE, 1990.
[5] ACM/IEEE-CS Joint Task Force on Software Engineering Ethics and Professional Practices, Software Engineering Code of Ethics and Professional Practice, Version 5.2, September 1998.
[6] ACM/IEEE-CS Software Engineering Computing Curriculum, 2004, sites.computer.org/ccse.
[7] Davies, Lloyd, Why kick the "L" out of "learning"? The development of student employee ability skills through part-time working Education þ Training Vol. 42 No. 8, 2000 pp. 436-444
[8] Cotton, K. (2001), Developing Employability Skills, Northwest Regional Educational Research Laboratory, Portland, OR
[9] Connor, H. and Shaw, S. (2008), "Graduate training & development: current trends and issues", Education þ Training, Vol. 50 No. 5, pp. 357-65
[10] Software Engineering Body of Knowledge, IEEE Computer Society, 2004, www.computer.org/portal/web/swebok.
[11] Graduate Software Engineering 2009, www.gswe2009.org.
[12] Barrows, H. S., Tamblyn, R.M. (1980) Problem-Based Learning: An Approach to Medical Education. New York: Springer Publishing Company
[13] TRANFoRm EU Project http://137.73.82.45/Home.html (accessed November 2010)
[14] Laurence, P.D., Spink, A., Jameson, R. (2002): Variation in EPR Clinical Support Systems: Data, Decision and Disparity, Health Informatics Journal 8, 34-38

[15] Orfanidis, L., Bamidis, P. D., Eaglestone, B. (2004). Data quality issues in electronic health records: an adaptation framework for the Greek health system. Health Informatics Journal Vol 10(1): 23-36.
[16] Forster, M. (2008). Electronic medical record systems, data quality and loss to follow-up: survey of antiretroviral therapy programmes in resources-limited settings. Bulletin of the World Health Organisation.
[17] Welzer, T., Boltjan, B., Golob, I., Margan, D.. (2002). Medical Diagnostic and Data Quality, Proceeding of the 15th IEEE Symposium on Computer-Based System, pp 97-101.
[18] Simpson, C., Roberts, T., Cooper, D.K., O'Brien, F.: Using Statistical Process Control(SPC) chart techniques to support data quality and information proficiency: the underpinning structure of high-quality health care, Quality in Primary Care, 13:37-43 (2005)
[19] Cretikos, M., Parr, M., Hillman, K., Bishop, G., Brown, D., Daffurn, K., Dinh, H., Francis, N., Heath, T., Grant, H., Murphy, J., Sanchez, D., Santiano, N., Young, L. (2006). Guidelines for the uniform reporting of data for Medical Emergency Teams. Resuscitation 68, 11-25
[20] American Heart Association. Guidelines for cardiopulmonary resuscitation and emergency cardiovascular care-and international consensurs on science. Resuscitation 2000; 46:3 -430
[21] Jacobs, I., Nadkarni, V., Bahr, J., Berg R.A., Billi, J.E., Bossaert, L., Cassan, P., Coovadia, A., D'Este, K., Finn, J., Halperin, H., Handley, A., Herlitz, J., Hickey, R., Idris, A., Kloeck, W., Larkin, L.G., Mancini, M. E., Mason, P., Mears, G., Monsieurs, K., Montgomery, W., Morley, P., Nichol, G., Nonal, J., Okada, K., Perlman, J., Shuster, M., Steen, P.A., Sters, F., Tibbals, J., Timerman, S., Truitt, T., Zideman, D.,. (2004) Cardiac arrest and cardiopulmonary resuscitation outcome reports, Resuscitation; 63:233-49
[22] Shroff, V, L.Reid, S, Hashmi, G. Mulligan, D, Sheehy, K. Xiang and I. Richardson, Development of a Software Quality Plan for Hospitals: An Irish Perspective, HEALTHINF 2011, International Conference on Health Informatics, 26-29th January, 2011, Rome, Italy, pp 587-591.
[23] HSE.(2007). Quality & Risk Management Standard. Available at [Accessed July, 2010]: http://www.hse.ie/eng/About/Who/OQR009_20080221_v3_Quality_and_Risk_Management_Standard.pdf
[24] Saving Badin 2000, Problem-based learning in higher education: Untold stories. Buckingham, UK Open University Press
[25] Boud &Feletti 1997 The challenge of problem-based learning (2nd ed.). London: Kogan Page
[26] Boud & Feletti, 1991, The challenge of problem-based learning.New York: St. Martin's Press.
[27] Barrows, Howard S., 1996,. Problem-based learning in medicine and beyond: A brief overview. New directions for teaching and learning(68), 3-12
[28] Schmidt, H. G. (1983). Problem-based Learning: rationale and description. Medical Education, 17, 1-16.
[29] Schmidt, H. G. (1993). Foundations of problem-based learning: some explanatory notes. Medical Education, 27, 422-432
[30] Coles, C. R. (1990). Elaborated learning in undergraduate medical education. Medical Education, 24, 14-22.
[31] Coles, C. R. (1991). Is problem-based learning the only way? In D. Bound and G. Feletti (Eds.), The Challenge of problem-based learning (Chapter 30). London: Kogan Page
[32] Milter, R. G.,& Stinson, J. E. (1995). Educating leaders for the newcompetitive environment. In W. H. Gijselaers, D. T. Tempelaar, P. K. Keizer, J. M. Blommaert, E. M. Bernard, & H. Kasper (Eds.), Educational innovation in economics and business administration: The case of problem-based learning (pp. 30-38). Dordrecht, the Netherlands: Kluwer
[33] Brownell, Judi &. Jameson, Daphne A (2004) Problem-Based Learning in Graduate Management Education: An Integrative Model and Interdisciplinary Application Journal of Management Education 28; 558
[34] Walker A, & Leary, H (2009) A Problem Based Learning Meta Analysis: Diff erences Across Problem Types, Implementation Types, Disciplines, and Assessment Levels, The Interdisciplinary Journal of Problem-based Learning • volume 3, no. 1
[35] Camp G, (1996) Problem-Based Learning: A Paradigm Shift or a Passing Fad:, Medical Education Online, 1996,1:2.
[36] Armarego, Jocelyn ,(2007) ,Beyond PBL: Preparing Graduates for Professional Practice, 20th Conference on Software Engineering Education & Training (CSEET'07), pp 175-183, 0-7695-2893-7/07
[37] Barrows, Howard S., 1986, A taxonomy of problem-based learning methods. Medical Education, 20(6), 481-486
[38] Richardson, Ita and Yvonne Delaney, Problem Based Learning in the Software Engineering Classroom, IEEE Conference on Software Engineering Education and Training, CSEET 2009, IIIT, Hyderabad, India, 17th-19th February 2009, pp. 174 - 181.
[39] Whetten D.A. & Cameron,K.S Developing Management Skills, 2010, Pearson Publication, New York
[40] Day, D.V. 2001, "Leadership Development: A Review in Context." Leadership Quarterly, 11(4): 581-613
[41] Meyer J H F and Land R 2003 'Threshold Concepts and Troublesome Knowledge 1 – Linkages to Ways of Thinking and Practising' in Improving Student Learning – Ten Years On. C.Rust (Ed), OCSLD, Oxford
[42] Gijselaers,W. H., &Woltjer, G. (1998). Cognitive science perspectives on learning and instructional quality. In D. T. Tempelaar, F.Widersheim-Paul,&E. Gunnarson (Eds.), Educational innovation in economics and business II: In search of quality (pp. 345-354). Dordrecht, the Netherlands: Kluwer