Investigating Heuristic Evaluation as a methodology for evaluating pedagogical software: an analysis employing three case studies

Mike Brayshaw[1], Neil Gordon[1], Julius Nganji, Lipeng Wen, Adele Butterfield

[1]Department of Computer Science, University of Hull, Hull, HU6 7RX
{m.brayshaw, n.a.gordon}@hull.ac.uk

Abstract. This paper looks specifically at how to develop light weight methods of evaluating pedagogically motivated software. Whilst we value traditional usability testing methods this paper will look at how Heuristic Evaluation can be used as both a driving force of Software Engineering Iterative Refinement and end of project Evaluation. We present three case studies in the area of Pedagogical Software and show how we have used this technique in a variety of ways. The paper presents results and reflections on what we have learned. We conclude with a discussion on how this technique might inform on the latest developments on delivery of distance learning.

Keywords: Heuristic evaluation, pedagogy, pedagogical software, disability, technology enhanced learning, flexible learning.

1 Introduction

This paper addresses practical concerns about evaluation methods and techniques in the context of educational software. In it we discuss methods of evaluating software and pragmatic approaches about how this can be done in practice. We note that whilst this is often hard to do, it is a vital part of the software development lifecycle. Formal experimental empirical studies can be difficult to set up, organise and run. This paper discusses our experiences with a lightweight alternative. What this paper does is reflect on issues we have found and provides details through case studies to demonstrate the application of Heuristic Evaluation as an alternative possible solution route. Heuristic evaluation is a well established method for quickly evaluating the efficacy of new media solutions to interface issues [12-14]. One type of new media is that pertaining to pedagogy and educationally motivated software. Squires and Preece [16] first proposed using heuristic evaluation as a way of measuring quality, learning potential, and usability in educationally motivated applications. Albion [1] and Benson et al [3] are examples of this methodology being applied.

In this paper we will reflect on three case studies that have used this evaluation technique and the value and insight that it has afforded. The heuristics used here were developed from Squires and Preece [16] so all three studies used the same questions.

The first was a Semantic Web motivated Virtual Learning Environment, proposed as a general learning architecture, but instanced here as a computer science tutor [17]. This case study used heuristic evaluation at the end of a long software build to evaluate the usability and learnability of the final product. The second case study [8-9] looked at ontology-driven personalisation of services in the context of university students with disabilities. Thus the context and nature of the subjects of the evaluation was notably different from that of the first approach. Heuristic evaluation was carried out by both subjects with disabilities and without. The task was to locate resources and manage the learning packages in the context of a university using a specially designed software help and navigation tool. The third case study used heuristic evaluation to gain an understanding of the usability of a rolling prototype tutoring system for C# [4]. The prototype was evaluated by a group of experts who had established experience in teaching programming at First Year University level to an introductory computer science cohort. The technique was thus used not as an end user evaluation but as part of a rolling software development to get expert input into the evolving software tutoring tool.

This paper will report results from all three studies. In all the cases reported heuristic evaluation was found to be an effective tool and one that was easy and fairly fast to use. An evaluation of the effectiveness of all three case studies will be given and we will give individual examples of some of the outcomes and effectiveness of this type of activity. We will conclude with a discussion about what we have learnt about employing this technique and compare the type of outputs achieved from the approach taken here to that which we might have got from more formal or traditional evaluation methods. Indeed where we aim to turn next is to carry out a traditional empirical evaluation experiment with one of our case studies to compare the results. We look forward to being able to report on that in the future.

2 Case Study: A schema-driven flexible virtual learning environment

2.1 The Schema-driven approach

This case study is based on the evaluation of a system that was designed to provide a software platform for the provision of personalised content for virtual learning environments [17]. The system was designed to use a schema - that is machine-readable versions of instances of an educational process - providing a management system to allow the system to provide flexibility to the user in terms of building a personalised learning schema based on the users' needs. For this approach, semantic web concepts were utilised so that the users would be able to make their choices in terms of options such as the style of learning, and then the system would compile – i.e. combine and build – a suitable overall learning schema for the user.

2.2 Evaluation of the system

Once the prototype system was developed, the issue remained of how to evaluate its effectiveness. It was thus decided to develop an evaluation based around a set of usability elements, considered against the user's context i.e. the usability of the e-learning software, considering the designer/learner models, learner control, teacher customisation, and pedagogy. The heuristic evaluation approach in this case study was based around comparing the prototype against the perceived effectiveness of 3 other existing systems (for details of this see [17]).

In terms of Human-Computer Interaction, the system usability of the prototype was assessed as part of the overall heuristic evaluation, with regards to Nielsen's heuristics, for example "user control and freedom" , "flexibility and efficiency of use" and "help users recognize, diagnose, and recover from errors" [14]. Further, the prototype was considered against the assessment criteria for eLearning systems designed by Squires and Preece [16] that was developed from Nielsen's [14] heuristics. These criteria included "learner control", "teacher customisation" and "pedagogy".

For the prototype eLearning system in this case study, the participants were provided with information at least 3 days before the evaluation, which consisted of a hand-out with instructions, identifying the goals of the study, a set of definitions used in the evaluation and what they would be expected to do. The evaluation focussed on the typical use of the system – i.e. as users within the context of eLearning – not on general usage. To be specific, the study concerned the usability regarding flexibility of the selected eLearning tool around, for example, learner control, teacher customisation and pedagogy.

The participants were provided with instructions on using the different systems to carry out the requisite tasks, e.g. to select learning units, to collect these together to develop an overall learning process, and to review how they were able to manage that process. The study itself had 10 participants, ranging from 3 professors to a range of computer science PhD students. Interaction was constrained with the use of scripts to try to constrain the range of actions so that each system could be compared in a systematic way.

After each participant had used each system, there was a heuristic evaluation session to gather data on their use and perceptions of the system. There was also an opportunity to provide views on the overall process.

2.3 Results of the evaluation

The evaluation produced data on the participants' views of using all 4 systems – that is the prototype system and the 3 comparison systems. The participants used the same evaluation heuristics enabling comparison across the systems. By collating data into Agree, Disagree or Maybe the different systems could be compared, especially in terms of the usability of the different systems. This enabled the ease of use of the prototype to be evaluated against the other systems. This provided a good indicator of the usability of the system, with 70% agreeing the prototype enabled the user to develop a suitable learning schema for them (with only 10%, 20% and 10% agreeing this for the comparison systems). All 4 systems received positive (70% or more) agreement on the question of whether the software provided multiple views and representation. For full details of the evaluation see [17]. The use of heuristic evaluation for the flexible virtual learning environment enabled the system to be compared against 3 other systems. The evaluation showed the benefits and some limitations in the prototype system.

The participants in the evaluation were asked to comment on the evaluation methodology itself. The results from this showed that 90% felt the evaluation did show differences between the 4 systems, and the limitations of each, and 80% agreed that the evaluation results provided useful statistics by which to compare the systems.

3 Case Study: The ONTODAPS e-learning system

3.1 What is ONTODAPS?

To respond to the current need for inclusive blended learning in contemporary education which is greatly driven by information and communication technologies (ICTs), a disability-aware e-learning system needed to be designed, implemented and evaluated. The Ontology-Driven Disability-Aware Personalized E-Learning System (ONTODAPS) responds to that need and personalizes learning resources for students (both disabled and non-disabled) based on their disability type and severity whilst considering their learning goals and the preference of the formats in which learning resources should be presented.

3.2 The need for a disability-aware e-learning system

Although ICTs greatly facilitate learning for students without disabilities, it could pose a significant challenge to disabled students when the technologies are not accessible. With the success of the Web, learning is now being delivered online and numerous e-learning systems have been developed to facilitate learning. Nevertheless, some of these learning systems are not accessible to learners with disabilities. This problem could arise when the developers of the systems do not adhere to guidelines for accessible design such as WCAG 2.0 and do not consider the needs of disabled learners during the design process [9]. By engaging with disabled students at the University of Hull, directly in lectures and through mentoring sessions, we have been able to understand the difficulties these students face with using existing e-learning environments and to obtain their recommendations for increased accessibility in e-learning systems.

Over the past decade, our research has focused on how we deliver learning to students using educational software. With increasing awareness on the challenges faced by the increasing numbers of disabled students while trying to use blended learning technologies, we spread our tentacles into inclusive education, seeking novel ways to deliver courses in an inclusive way. From searching literature, we have found that very little research is being done to improve the learning of disabled students from a personalization approach particularly employing semantic web technologies and considering the needs of students with multiple disabilities. This response conforms to contemporary legislations in various countries that call for service providers to include the needs of disabled people through accessible systems. Such is the case with the Special Educational Needs and Disability Act, 2001 in the UK, the Accessibility for Ontarians with Disabilities ACT, 2005 and the Americans with Disabilities Act, 1990 just to mention these. The design process through which experts in disability and disabled students and other stakeholders are involved in designing inclusive and accessible blended learning systems adopted in the ONTODAPS design is in accordance with the disability-aware design approach [11].

3.3 Main functionalities of the ONTODAPS system

ONTODAPS functions as a multi-agent e-learning system that is driven by the ADOOLES (Abilities and Disabilities Ontology for Online LEarning and Services) ontology. The architecture of ONTODAPS is presented in Fig. 1.

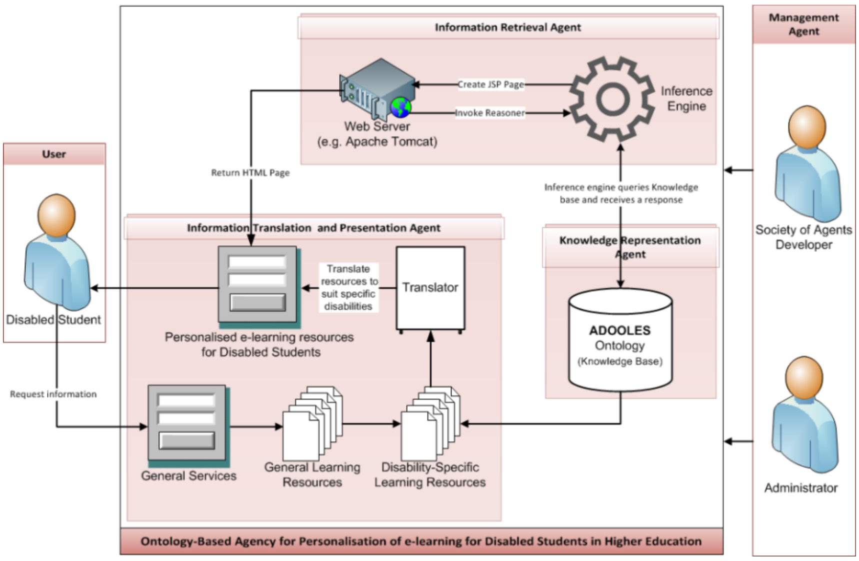
Information Retrieval Agent
Create JSP Page
Invoke Reasoner
Inference Engine
Web Server
(e.g. Apache Tomcat)
Management Agent
User
Return HTML Page
Inference engine queries Knowledge base and receives a response
Society of Agents Developer
Information Translation and Presentation Agent
Knowledge Representation Agent
Translate resources to suit specific disabilities
Translator
Disabled Student
Personalised e-learning resources for Disabled Students
ADOOLES
Ontology
(Knowledge Base)
Request information
General Services
General Learning Resources
Disability-Specific Learning Resources
Administrator
Ontology-Based Agency for Personalisation of e-learning for Disabled Students in Higher Education

Fig. 1. Architecture of the ONTODAPS e-learning system [8]

As ONTODAPS currently employs an interface technology, a student seeking personalized learning resources interacts with the system through a visual interface implemented with Java. The student identifies himself through an authentication mechanism involving a username and password and the system then recognizes the student’s disability type based on information stored in his profile. The student selects his learning goals and an indication of the severity of his disability. The ADOOLES ontology which represents knowledge about the student and the learning resources and goals is queried and information is passed onto the information retrieval agent which makes inference and then presents information to the information translation and presentation agent. The outcome is learning resource presented to the student in a format that is compatible with his disability and also meets his learning needs. If a student is visually impaired for instance, learning resources could be presented as audio or text and the text in the interface is also read out to the student through an inbuilt screen reader.

It is noteworthy that ONTODAPS has an administrative interface where administrators could manage the learning of the students and upload learning resources, setting up goals and also adding users to the system. There is also the possibility of a new user to register on the system and thence to manage their profile.

3.4 Heuristic evaluation of ONTODAPS

Heuristic evaluation, fundamentally based on Nielsen's ten heuristics [14], is a cost effective evaluation method used in finding problems with the design of a piece of software before its release. The aim of this is to fix the problems found before users can use the system. Heuristic evaluation can also be used to evaluate educational software and has been successfully employed to do this.

No	Heuristic
1	Ensures visibility of system status
2	Provides match between the system and the real world
3	Flexible enough to provide the user enough control and freedom
4	Is consistent and follows common operating system standards
5	Prevents errors
6	Supports recognition rather than recall
7	Supports flexibility and efficiency of use
8	Uses aesthetic and minimalist design
9	Helps users recognise, diagnose and recover from errors
10	Provides help and documentation
11	Has clear goals and objectives
12	Context is meaningful to domain and learner
13	Content clearly and multiply represented and multiply navigable
14	Provides navigational fidelity
15	Provides appropriate levels of learner control
16	Supports personally significant approaches to learning

Table 1. Heuristics used to evaluate ONTODAPS

We evaluated ONTODAPS after *learning with software heuristics* [16], also incorporating some *educational design heuristics* [15] in a similar manner to [1] and [5]. To heuristically evaluate ONTODAPS, we used ten experts in pedagogy at the University of Hull who had at least three years' experience in evaluating educational software and in human-computer interaction.

Before the evaluation, the experts were given a guide to ONTODAPS which was presented in three formats: audio, video and text format. The experts could choose the format they preferred. After going through the guide, they were asked to interact with the system at least twice and then to perform specific tasks before evaluating the system based on specific heuristics as shown in Table 1.

3.5 Results of heuristic evaluation of ONTODAPS

For each heuristic presented in Table 1, evaluators were asked to rate the software on a scale of 10. The mean scores for each heuristic obtained from the ten expert heuristic evaluators are presented in Fig.2.

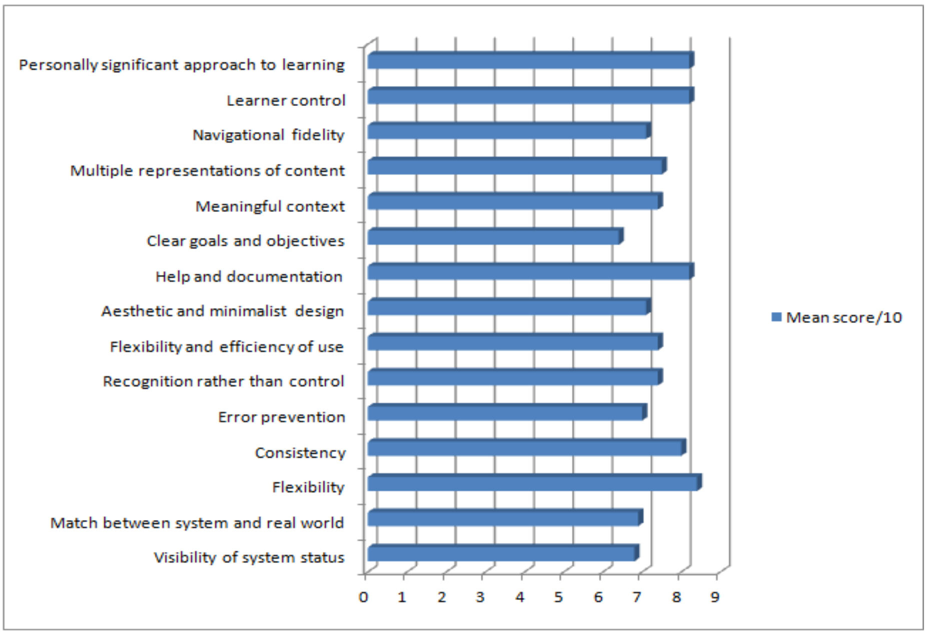

Fig. 2. Results of heuristic evaluation of ONTODAPS

Generally, ONTODAPS was seen to follow common operating system standards with its menus following platform standards. It provided appropriate levels of learner control by allowing learners to self-direct their learning through personalizing learning resources and allowing them to choose the format in which the resources are presented to them. Nevertheless, there were some usability problems in the area of system visibility status, needing much improvement in keeping users informed of what is going on through appropriate and timely feedback. Heuristic evaluation also helped find the need to fully grant keyboard only users access to the system by activating keyboard functionality. Thus, by carrying out a heuristic evaluation of the software, various usability problems categorized into major, minor and cosmetic problems were found and the system improved upon. By employing specific heuristics suitable for educational software, we ensured that the evaluated system meets the standards of educational software and focusing on special educational needs, we also ensured that the software is fully inclusive, meeting the needs of disabled users.

4 Case Study: An inquiry-based C# tutor

The third case study was an Inquiry based tutoring system for C#. Inquiry based learning [7] aims to put the user as the centre of the learning process and through their exploration acquire, at their own pace, through their own efforts, and in their own language acquire the desired skills. The overall make of the tutoring system is heavily influenced by Anderson [2]. The novel thing in this case study is that the approach taken here was one of iterative refinement of the prototype. We engaged with the experts as soon as the first robust prototype was available. So instead of using heuristic evaluation as an end evaluation technique, we embedded it into the very body of the development process. The target audience for the tutoring system was novice programmers but the aim was to get expert teachers evaluating the interface from the start. Every usability and learning problem found should be recorded and the violated heuristic found and given a severity rating scale.

Severity Scale
0) I don't agree that this is a usability problem at all
1) Cosmetic problem only; need not be fixed unless extra time is available
2) Minor usability problem; fixing this should be given low priority
3) Major usability problem; important to fix; so should be given a high priority
4) Usability catastrophe; imperative to fix before this product is released

Once the usability errors have been found then the expert will need to go back through the C# tutorial to ascertain the extent of the problem by applying a scale [3].

Extensiveness Scale
1) This is a single case
2) This problem occurs in several places in the program
3) This problem is widespread throughout the program

The Heuristic evaluation proved to be successful and led to significant updates in the software at each iteration. When initially carrying out the evaluation. it was believed that the software was almost at a stage of completion. Luckily the software evaluation was started early enough so from the feedback given, there was enough time for a significant amount of updates to take place.

An alpha study was carried out using PhD students which led to initial important modifications. The updated software was then sent to seven experts (who have at least 5 years teaching programming experience), in order to carry out the heuristic evaluation. After four evaluators replied, the collated results were extremely interesting and found some issues that had not been considered in the development of the project. The experts really tested the extremities of the program by not following set paths or reading the instructions: - an oversight which had been taken for granted. From the feedback given it led to more software updates to try and improve the system based on the comments provided.

This led to the completion of prototype 6. This version was then sent out to the same seven evaluators asking if they had time to carry out a heuristic evaluation on the current prototype. One of the evaluators responded promptly and their feedback pointed out some fundamental issues which needed to be dealt with immediately. The updates were duly carried out and led to the completion of prototype 7. This prototype was sent to five of the evaluators who had already been sent the previous versions and a new external (to the University) evaluator who also has had at least 5 years experience teaching programming. Thus heuristic evaluation informed an iterating evolution of prototypes. Some experts commented on a single version but others were involved over the whole process and were able to give insights into the evolution of the whole. Heuristic evaluation provided a key role in a rolling evolution of a software project.

To summarise the changes heuristic evaluation made overall for this process important revisions were made as follows:

- Important HCI navigational changes
- HCI Conformity and Consistency Changes
- Important Changes to Wording
- Updates to Pedagogic Issues
- Tutorial Learning Content

5 Evaluation, discussion and conclusion

The paper has considered some of the practical concerns about evaluation methods and techniques for educational software. It is not our intention to find a substitute for classical empirical studies. In an ideal world it would be great to be able to run full empirical evaluations of our target software using large numbers of subjects in a balance empirical design. Indeed this assumes that sufficient numbers of suitable users are available. The design of these studies is not a trivial task in addition to pragmatic issues. In the rapidly changing world of computing, technology functionality, interaction model, and the continuing evolution of the affordances of the technology, we need to evaluate the artifacts that we create. Indeed the emergence of things like ubicomp (Ubiquitous Computing, [18]) and how we interact with technology may well be on the move and we need a way of evaluating the effectiveness as we move to new delivery surfaces. Thus a quick and light touch way of being able to evaluate the pedagogical software is a timely thing. As time is an essence heuristic evaluation has here provided us with a way to answer these questions. We saw how this could be done in Case Studies 1 and 2 as a way of evaluating the pedagogic tool that was produced at the end of the project. Case Study 3 does demonstrate how it can be used actively to drive the very design process so that the output of the expert study feeds directly into the next generation of prototype, demonstrating how this approach could be deployed in a fast changing interface design.

There has been criticism in the past that heuristic evaluation just tests the opinion of those who are carrying out the evaluation [12]. It may not actually spot any real interface issues or problems, but just instead the preferences of the experts employed. Care has been taken in all these studies to choose experienced computer scientists in all these, students who we hope can be relied on to make expert judgement about HCI and pedagogy judgments. Indeed all three of the studies took place in a university context so the experts were in a position to comment on their effectiveness. Furthermore the use of the set series of questions [16] would act to focus the dialog onto the matter in hand.

Another issue that raised concerns was false alarm – where the expert identifies an interface issue which is actually of no concern to the user.   Now it can be argued that false alarms are actually good things if it leads to more rigorous testing and investigation. As the audiences for online courses increases so may the range of users – possibly cutting down on false alarms by increasing varieties of behaviour.

One area that we are currently keen to compute is to compare our results that we have got from these quick and light evaluations with the types of results that would be obtained from a traditional empirical evaluation in the context of pedagogically motivated software (as opposed to more generally in HCI, e.g. [6]. We are currently in the process of designing and running such a study for Case Study 2 to see how closely these two results match or otherwise.

The move to learning online and therefore by computer mediated means we are accelerating even at the time of writing. In the past year or so we have seen a vast explosion in the use of MOOCs (Multiple Open Online Communities, ref needed). These are now offered by a very large number of universities from all over the world offering courses, where student numbers are measured in their 100 000s, offering distance education for users in many different countries. Now trying to evaluate these systems, based on student experience, would present a problem, both due to the logistics of organising a multi-continent study, but also of issues of the extremely variable skill sets/life experiences of those who sign up for such courses. Anyone can sign up irrespective of whether these courses are appropriate given their study level. Thus any study would have to work who was being evaluated, how they were doing their study, and balance for such things in any study undertaken. However to extend the approach taken here will be an appropriate use of our light weight pedagogical evaluation developed here. Rather than casting around in the wide diaspora of users instead we can focus on the education material itself and carry out the evaluation by heuristic evaluation using a college of experts. Thus heuristic evaluation of pedagogic software has the potential to allow us to evaluate brand new ways of learning.

6 References

1. Albion, P. Heuristic evaluation of educational multimedia: from theory to practice. ASCILITE 1999: *16th Annual Conference of the Australasian Society for Computers in Learning in Tertiary Education: Responding to Diversity*. Brisbane, Australia. (1999)

2. Anderson, J. R. The expert module. Foundations of intelligent tutoring systems, 21-53. (1988)

3. Benson L, Elliott, D., Grant, M., Holschuh, D., Kim, B., Kim, H., Lauber, E., Loh, S., and Reeves, T., Heuristic Evaluation Instrument and Protocol for E-Learning Programs, (Online), Available: http://it.coe.uga.edu/~treeves/edit8350/HEIPEP.html (Accessed 31st of October, 2013) (2001)

4. Butterfield, A., and Brayshaw, M. A Pedagogically Motivated Guided Enquiry Based Tutor for C#, Submitted Article. (2014)

5. Granic, A. & Cukusic, M. Usability Testing and Expert Inspections Complemented by Educational Evaluation: A Case Study of an e-Learning Platform. *Educational Technology & Society*, 14, 107-123, (2011)

6. Jeffries, R., and Desurvire, H., Usability Test versus Heuristic Evaluation, *SIGCHI Bulletin*, 24(4), October 1992 (1992)

7. Kahn, P., & O'Rourke, K. Guide to curriculum design: enquiry-based learning. Higher Education Academy, 30-3. (2004)

8. Nganji, J. T., Brayshaw, M, and Tompsett, B. C. Ontology-Based E-Learning Personalisation for Disabled Students in Higher Education, *ITALICS*, 9(4), February, 2011, ISSN 1473-7507 (2011)

9. Nganji, J.T., Brayshaw, M. and Tompsett, B. Ontology-Driven Disability-Aware E-Learning Personalisation with ONTODAPS. *Campus Wide Information Systems*, vol.30, No. 1, pp. 17-34. (2013)

10. Nganji, J. T., & Brayshaw, M. Designing Personalised Learning Resources for Disabled Students Using an Ontology-Driven Community of Agents. In P. Isaias, & M. Baptista Nunes (Eds.), *Information Systems Research and Exploring Social Artefacts: Approaches and Methodologies* (pp. 81-102). Hershey, PA: Information Science Reference. doi:10.4018/978-1-4666-2491-7.ch005 (2013)

11. Nganji, J.T. and Nggada, S.H. Disability-Aware Software Engineering for Improved System Accessibility and Usability. *International Journal of Software Engineering and Its Applications (IJSEIA)*, Vol. 5, No.3, pp.47-62, July 2011. (2011)

12. Nielsen, J., and Molich, R. Heuristic evaluation of user interfaces, *Proc. ACM CHI'90 Conf.* (Seattle, WA, 1–5 April), 249-256 (1990)

13. Nielsen, J., and Mack, R., *Usability Inspection Methods*. New York: John Wiley. ISBN 0-471-01877-5. (1994)

14. Nielsen, J. *Usability inspection methods*. In: Nilesen, J. & Mack, R. (eds.) Usability Inspection Methods. New York: John Wiley. (1994)

15. Quinn, C. N.. Pragmatic evaluation: lessons from usability. In: Christie, A. & Vaughan, B. (eds.) *13th Annual Conference of the Australasian Society for Computers in Learning in Tertiary Education (ASCILITE 96).* Adelaide: Uni SA: Australia Society for Computers in Learning in Tertiary Education. (1996)

16. Squires, D. & Preece, J. Predicting quality in educational software: Evaluating for learning, usability and the synergy between them. *Interacting with Computers*, 11, 467-483. (1999)

17. Wen. L , Brayshaw, M., and Gordon, N. Personalized Content Provision for Virtual Learning Environments via the Semantic Web, *ITALICS*, ISSN 1473-7507. (2012)

18. Weiser, M., Gold, R., and Brown, J. S. The origins of ubiquitous computing research at PARC in the late 1980s. *IBM systems journal*, 38(4), 693-696. (1999)