Development of Interface Design for Malaysian Interactive Courseware: A Proposed Framework

Asia Pacific Journal of Academic Research in Social Sciences
Vol. 2, 8-13
November 2017
ISSN 2545-904X

Norfadilah Kamaruddin
Creative Visual Exchange Group (CREATe), Faculty of Art and Design, Universiti Teknologi MARA, 40450 Shah Alam, Selangor Darul Ehsan, MALAYSIA
norfadilah@salam.uitm.edu.my

Date Received: September 1, 2016; Date Revised: July 24, 2017

Abstract— *This paper introduces a new proposed framework for the development process of interface design for Malaysian interactive courseware by exploring four established model in the recent research literature, existing Malaysian goverment guidelines and Malaysian developers practices. In particular, the study looks at the stages and practices throughout the development process. Significant effects of each of the stages are explored and documented, and significant interrelationships among them suggested. The results of analysis are proposed as potential model that helps in establishing and designing a new version of Malaysian interactive courseware.*

Keywords— *Development processes, Interaction with interface, Interface design, Interactive courseware*

INTRODUCTION

The software development process can be described as a development life cycle. In order to manage and ensure the success of computer software, a series of structured development activities has been developed over many years and these have been formulated into development processes. Throughout the literature, a variety of systems development models can be found and each outlining detailed structures and processes and a variety of activities that are closely linked. While many models are acceptable, depending on the circumstances and the goals of the product, it should be noted at the outset that variant approaches impact on aspects of the outcome [1], [2], [3]. Therefore, an appropriate development model should provide the developer with a systematic, well-disciplined and practical approach to the design, development and maintenance of the software [4], [5]. In additional, in current context of Malaysian perspective, ceveral issues related to the level of interface design performance within interactive educational courseware been highlighted by several researchers. Thus, based on this undestanding, this paper introduces a new proposed framework for the development process of interface design for Malaysian interactive courseware by exploring four established model, existing Malaysian guidelines and the actual process pratices by Malaysian courseware developers.

BACKGROUND ON SOFTWARE DEVELOPMENT PROCESS

Literature established that the classic waterfall model is the best-known and perhaps oldest process for software development. This model moreover has been used since it was proposed by Winston W. Royce in 1970 [4]. It involves a phased process, which includes requirements analysis and planning, design, development, implementation and evaluation. In particular, the requirement analysis and planning phase, involves defining the target users of the potential product and identifying their requirements and needs. The design phase therefore focuses on determining the design solutions to be used by considering approaches to delivery formats, structure (which is commonly referred to as information architecture and is guided by a flowchart and storyboard), interface and screen design, and elements of the look and feel. The development phase is where the design components and programming are assembled as a functioning prototype. Alpha and beta testing occur at this stage. An implementation phase, where the end product (or in some models, a prototype) is implemented and provided to the target user (or a representative sample group). Finally, an evaluation phase is focuses on gathering feedback on the end product from actual users.

The development process of the waterfall model in detail was identified moves from one phase to the next phase only when the previous phase is completed and

perfected. According to [6] this model attempts to separate the development life cycle into discrete activities by creating a series of linear actions in which each step forms the basis for the next step, and the correctness of each step can be checked. For example, when analysis is complete, the process proceeds to the design stage and, when it is complete, development commences and so on. Moreover, in this model, the interface design development occurs in the design and development phase when an instructional plan is provided by the development team, including a set of requirements and flow charts that show a general path of connections in the interface design. At this point, designers carefully consider the arrangement of screen layout, a style guide for graphic elements, and a navigation system that can engage users in meaningful and authentic tasks [7], [8]. To make this process efficient, effective communication between members in the development team such as a graphic designer, programmer, instructional designer, and a content expert is essential.

This waterfall model therefore offers an advantage to developers because it breaks a complex task down into smaller tasks and it allows everyone involved in the process to see exactly what has been completed and what remains to be done. Nevertheless, a number of problems are also be identified in the literature. The main weakness is that it does not include an opportunity for end-users to review and evaluate the potential product being created until it is complete as it does not include interim feedback between stages. Moreover, once the process has progressed, there is no way the product under development can go back to the previous stage because it always moves in a single direction. In response to this problem, some alternative models have been introduced, some of which involve the steps of the waterfall model as a foundation.

The second model been reviewed is the iterative model. This model was introduced by Boehm in 1985 as an alternative model for the software development process. In terms of the phases involved, it incorporates planning, requirements and analysis, design, evaluation, and implementation. It is therefore similar to the waterfall model but it involves a cyclic process which includes multiple iterations or versions in a repetitive process. This model introduces prototyping, testing and analysis during the design phase and involves redesigning on the basis of the feedback received, in order to refine the quality of the final product.

The most important advantage of the iterative model is that prototyping in the design stage offers the developers the opportunity to elicit periodic, objective feedback on the appropriateness of the solution, which allows the developer to make incremental adjustments and make corrections based on a better understanding of the needs and requirements of the user. By implementing this cyclic process, which involves input from the end-users at an early stage on aspects like the proposed look and feel of the application, developers can capture and fix possible weaknesses in the underlying interface design at the prototype phase, rather than after all the components have been integrated into the design. This model therefore emphasises the importance of collaborative work between developers and users, which is necessary for the product under development. Early input from end-users not only has the potential to highlight problems earlier in the design phase, this method can shorten the development time of a project [9]. Besides helping to ensure that the actual needs of users are met, it minimizes the risk of conflict between users and the development team.

In 1988, Boehm proposed further modifications to overcome the limitations of the waterfall model and the iterative model. This model is referred to as the Spiral Model or Boehm's Model. The steps employed are similar to the steps in the traditional waterfall model because, like the iterative model, it is an improved version of the waterfall model on which it is based. It combines the advantages of prototyping within the progression of the waterfall model and involves a number of iterations as it passes through four main steps: analysis of requirements, planning, development and evaluation.

Like the iterative model, from the early stages through to final product development, the process is highly dependent on prototyping designs and testing, which are evaluated by representative users. With this focus, the spiral model emphasises flexibility in meeting end-user requirements to minimise the risk of ineffectiveness. The first iteration of the spiral model is considered most important because at the first iteration, normally most potential risks, obstacles and needs are identified. In the next iteration, all the factors and problems discovered in the first phase will be resolved with great care before being re-tested. Because the process emphasises risk analysis, and because the knowledge of end-users is actively incorporated and contributes to the design solution, not only representative user groups, but highly skilled people in the areas of planning, risk analysis and

ISSN 2545-904X

mitigation, development and customer relations are involved in the process outlined in the spiral model. In addition, because it involves several periods of consultation and multiple cycles, the model requires more time to complete.

The last model reviewed is ADDIE model. In the literature, the ADDIE model is one of hundreds of instructional design models proposed for guiding the process of development for educational materials. The process involves similar phases to other generic models. These include Assessment, Design, Development, Implementation, and Evaluation. While these core steps of the ADDIE model may follow generic models, the ADDIE activities are not organized in a linear or straight-forward way. They are cyclical [10] and recursive like the spiral and iterative model, but their central focus is on evaluation, which sits at the heart of the model. The iterative aspect of this model is represented by the line and arrows running vertically down the left side of the model and the two-headed arrows between each component, as depicted in Fig. 1. Each step has an outcome that feeds into the next step in the sequence but evaluation is a persistent interim step to ensure perpetual quality improvement.

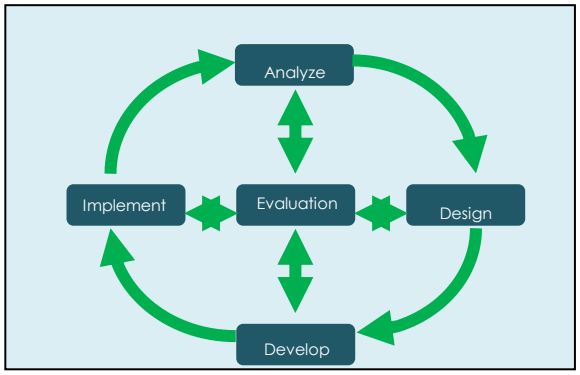

Fig. 1. The ADDIE model: Core elements of the phases of quality improvement (Source: Smith & Ragan, 1998)

As a useful model for creating effective interactive learning material for education, this model highlights analysis as the most important step in the process. The analysis of users needs and requirements, as well as input of instructional designers, helps developers (particularly the designers) to establish a clear understanding of the gaps in the users' existing knowledge and skills and strategies to bridge them. Therefore, during the analysis stage, the designer identifies the learning problem, the goals and objectives, the users' needs, instructional strategies and methods. This step is moreover necessary in order to define the parameters for the production of the product, including the learning environment, any constraints on limitations or opportunities, delivery options, and the timeline for the project.

In this model, the creation of the content and production of learning materials occurs during the design phase. Generally, rapid prototyping of the interface design is conducted at this stage to facilitate user feedback. Some scholars describe this as a using information gathered from analysis and design phase, the development stage begins, complete with evaluation and measurement of how well the product achieves its objective. Therefore, the evaluation of the ADDIE model consists of two parts: formative and summative. The formative evaluation may involve peer review, a walk-through of a rapid prototype, observations of the target group using the software and interviews (individually or in focus groups) with potential users. In each stage of development, these evaluations inform all aspects of the design including the interfaces, navigation, and how the software's functionality supports student learning. Summative evaluations consist of tests, based on performance, and provide formal opportunities for feedback from the users. Both tiers of evaluation provide insights into any false assumptions on the part of designers and developers, as well as design errors, and any barriers to achievement of the desired outcomes.

A software development process must therefore be understood as a phased and rigorous endeavour comprised of many separate but inter-related activities that each have a bearing on the creation of products. While all of the four models discussed above help to ensure this, all have advantages and disadvantages. If the problem is well defined and well understood and needs little change, the life cycle of the waterfall model may be sufficient to produce a product in the most cost-effective way. However, if the developer lacks a clear understanding of needs of users, they are unlikely to be able to produce comprehensive and entirely appropriate specifications at the beginning of the process. In this case, the developer must choose a longer life cycle and more complex approach, which involves stakeholder participation (e.g. spiral model or iterative model). And, to promote the use of technology as an effective tool to support learning, the instructional design process of the ADDIE model is perhaps the best system design process currently available that can be practiced by the developers.

ISSN 2545-904X

THE SOFTWARE DEVELOPMENT PROCESS IN MALAYSIA

Interviews with ten Malaysian courseware developers revealed that the current flow of software development practised is similar to the product life-cycle of the waterfall model. Five particular steps are being practiced, which are similar to those most commonly referred to in the literature. They include the requirements and planning stage, analysis stage, design stage, development and testing stage, evaluation and feedback stage and finally, the delivery and implementation stage. However, the responses show that most of them also initiate a requirements and planning phase up front, and include iterative phases in the development stage and evaluation stage to refine the courseware. Therefore, it is not carried out in an entirely, linear way.

Based on the data gathered, it is important to clarify that the current practices of software development in Malaysiaidentified differences from the waterfall model (the inclusion of a requirements and planning phase and an iterative phase), does not bring the process in line with the Spiral, Iterative or ADDIE models. While the process includes iterative phases in the development stage and evaluation stage to refine the courseware, the interviews revealed that the iterative phase is not entirely similar to the iterative process suggested in the Spiral, Iterative or ADDIE models in which, if an error or mismatch is noticed at any phase of the project by target group feedback, the previous stage is repeated from the beginning. A revision process only occurs if an error is revealed within the development and evaluation stages. The reason provided by the participants is that to make changes then is less time consuming and expensive than the ADDIE model where a change means a total rework.

Moreover, the changes are based on internal reviews and requests by the Ministry only. No testing is undertaken with user groups, so changes are not based on their feedback. In this regard, it can concluded that the iterative process being practiced by the developers in Malaysia is simply to ensure that all Ministry requirements are signed off, rather than to produce a product that optimally fulfils end-user needs. Example of developers' feedback regarding this:

> *"We are the developers of the interactive educational courseware so we should know the basic requirements of the courseware. So I don't think we need to do a user needs analysis."* (Developer Interview, respondent 2)

> "*So far, we haven't yet conducted a user needs analysis with the students. This is because we are familiar with the Ministry project. We also don't have time to concentrate on that because we are not only taking jobs from the government. At the same time we also have to complete other jobs."(*Developer Interview, respondent 1)

> *"We conduct product testing but only among our team in order to determine the weak points of the courseware, and we modify it before sending it to the Ministry. Generally, this testing is primarily focused on evaluation of the courseware functionality."* (Developer Interview, respondent 3)

> *"Actually, the authorization of decisions, especially on interface design, is in the Ministry hands, not ours."* (Developer Interview, respondent 4)

In summary, the courseware developers do not consider end-user involvement necessary or possible within the timeframe, nor their responsibility. However, self-claimed familiarity with the tasks required is not enough to ensure that the interactive courseware is embraced by end-users. As the literature has established, compared with other models of software development, such an approach carries with it a high potential for deficiencies in the content, interface and interaction design of software products.

THE PROPOSED FRAMEWORK FOR THE MALAYSIAN SOFTWARE DEVELOPMENT PROCESS

The analysis identified issues of a users needs and invovlement arising from the current development process in Malaysia particularly in testing the product. However, the literature contains a store of proposed solutions based on user participation and engagement in software development. Thus, by concerning this issue, a new framework are proposed here with the aim is not to simply put forward another model for software development. Rather, it is intended to counter the weak relationships that currently exist between the main stakeholders and the lack of user involvement in the design and evaluation process. Thus this proposal first involves including teacher's involvement with a view to acknowledging and benefiting from their expertise in the domain of classroom teaching, pedagogy in general and understanding of interactive learning concepts, and incorporating this knowledge into the design considerations. The best way to ensure that end-user needs are taken into account in the software

ISSN 2545-904X

development process is to continuously incorporate their involvement in the development process.

In proposing a new framework, it is therefore important to first consider the various models that incorporate user input. There are several approaches to user involvement established in the literature. For instance, users can be involved at the early stages of the process when the requirements are determined. At this level, they are commonly involved in providing insights into their specific needs, and consequently contributing to the user needs analysis. The second type of user involvement is during the prototyping phase, in which the user is involved in testing low fidelity and high fidelity prototype versions of the interaction and interface design. At this level, users might be observed interacting with the prototype or may complete a survey about the various features to provide their feedback and offer critique and suggestions to the designers. This type of feedback is incorporated into the iterative model discussed in the previous chapter, and it contributes to redesign and improvements in the next iteration.

Another phase of user involvement is after the production of the system. Here, users interact with a Beta version of the actual product and give their opinions and report any problems they encounter. Revisions at this stage are then included in a revised version of product. At this point, however, only minor changes are possible rather than structural or major interface design changes, as these underpin the entire application. Beyond these bracketed forms of user input, approaches to extended user involvement are available. They include user-centred design (UCD), participatory design (PD) and, most recently, co-design. These three approaches are driven by the impetus of empowering stakeholders in the design process and involve more intensive collaboration between designers and other stakeholders. In short, the end users who will use the outcome contribute to the design process, with the designer acting as a facilitator of this contribution. As shown in Fig. 2 these models were developed some years ago but they have recently gained ground in terms of popularity and implementation.

This model moreover provides a life cycle structure that, like established models, focuses on five main phases in the software development process. As illustrated in the figure however, the iterative and linear approaches are combined. The repeated iteration approaches were adopted from the spiral and iterative model and from the ADDIE model, which places feedback in the centre of the process so that it becomes an integral and integrated step. In particular, collaborative approaches to iterative design are suggested for the phases of design, development and evaluation to cement relationships between the three key stakeholders in the current software development process (the Ministry, courseware developers and teachers) and embrace and integrate their many skills, insights and areas of expertise into the development process.

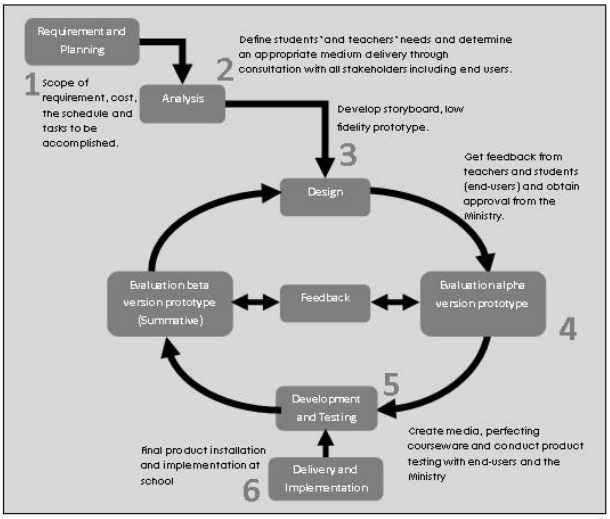

Fig. 2. A new proposed framework for the software development process in Malaysia

CONCLUSION

In response to the concerns of stakeholders involved in the development process and weaknesses in the current development process in Malaysia, the study has proposed a new framework for the software development process. A rationale for, and explanation of, the proposed framework has been presented. In essence, it highlights the importance of end-user involvement in the software development process, which is overlooked at present. Significantly, the proposed framework helps to support the development of relationships between all stakeholders and outlines the activities that should be undertaken by each of them. This suggested framework for courseware production that includes guidelines which promote user involvement can be useful for the development future courseware.

REFERENCES

[1] Bradler, J. (1999). Developing on-line Learning Material for Higher Education: An Overview of Current Issues. *Educational Technology Social*, 2, 112-121.

ISSN 2545-904X

[2] Bostock, S. (1996). *Courseware Engineering: An Overview of Courseware Development Process.* Retrieved Sept 18, 2010 from http://www.keele.ac.uk/depts/cs/stephen_bostock/docs/atceng.html

[3] Lee, P.M. & W.G. Sullivan, (1996). Developing and Implementing Interactive Multimedia in Education. *IEEE Transactions on Education*, 39, 430-435.

[4] Kamaruddin, N. (2010). Challenges of malaysian developers in creating good interfaces for interactive courseware. Turkish Online Journal of Educational Technology, 9(1), 37–42.

[5] Kamaruddin, N. (2010). Interface design development: Malaysian's practice. International Journal of Learning, 17(8), 195–204.

[6] Beynon-Davies, P. (1993). Information Systems Development: an Introduction to Information Systems Engineering. London: Macmillan

[7] Wilson, B., & Cole, P. (1991). A Review of Cognitive Teaching Models. *Journal of Educational Technology Research and Development*, 39(4), 47-64

[8] Wilson, B. G., Jonassen, D. H., & Cole, P. (1993). Cognitive Approaches to Instructional Design. In G.M. Piskurich (ed.), *The ASTD Handbook of Instructional Technology*. New York: McGraw-Hill

[9] Connell, J. & Shafer, L. (1989) *Structured Rapid Prototyping: An evolutionary Approach to Software Development*. New Jersey: Prentice-Hall, Inc. Smith & Ragan, 1998

[10] Gustafson, Kent L., & Branch, Robert M. (2002). *Survey of instructional development models (4th ed.)*. Syracuse, NY: ERIC Clearinghouse on Information & Technology, Syracuse University.

ISSN 2545-904X