بسم الله الرحمن الرحيم

University of Science and Technology

College for Graduate Studies and Academic Advancement

Towards SCORM-Conformant Integrated Learning Management System

Student Module: student profile, Announcement, Chat tools, View learning material

A Thesis Submitted to the University of Science and Technology as Partial Fulfilment Requirements for the Degree of Master of Science in Information System

Prepared By:

Doaa Magdi Eisa.

Supervised By:

Dr. Azharī Qism Allah Jād Al-Ṭayyib Muḥammad.

October 2014

Abstract:

The research focusing on development of systematically SCORM-Conformant Integrated Learning Management System (ILMS) for student module (Student Profile, Chat Tool, View Learning Material, View Announcement).

The research used software development method it is eXtreme Programming (XP) - for system development.

The research results indicated the ability of development such an SCORM Conformant ILMS in Student Module.

المستخلص:

البحث يُركّزُ على تطويرِ نظام ادارة التعليم الالكتروني المتكامل متوافقا مع اسكورم لوحدة الطالبِ (بيانات الطالبِ، أداة دردشةِ، عرض المواد التعليمية، عرض الاعلان).

إستخدم البحثُ طريقةَ تطويرِ البرامجِ وهي البرمجة الرشيقة (XP) – لتطويرِ النظامَ.

اشارت نتائج البحث القدرة على اجراء عمليات الطالب في نظام ادارة التعليم الالكتروني المتكامل متوافقا مع اسكورم

1. Introduction

The chapter presents the motivations, the objectives, the methodology, the expected outcomes and overview research.

2. Motivations

Most learning management systems are not integrated. To overcome existing problems occurring in traditional education of student module. Also many of LMS not conformant with SCORM.

3. Objectives

To systematically develop an SCORM-conformant integrated learning management system student module focusing on student profile, announcement, chat tools, view learning material services. Efficient utilization of human resource.

4. Methodology

The project uses Extreme Programming (XP) methodology, XP is the most widely used approach to agile software development which is intended to improve software quality and responsiveness to changing customer requirements, it advocates frequent releases in short development cycles, which are intended to improve productivity and introduce checkpoints at which new customer requirements can be adopted (Roger, 2010).

4.1. What is Extreme Programming? (XP)

Extreme Programming (XP) is a software engineering methodology, the most prominent of several agile software development methodologies. Like other agile methodologies, Extreme Programming differs from traditional methodologies primarily in placing a higher value on adaptability than on predictability. Proponents of XP regard ongoing changes to requirements as an often natural and

often inescapable aspect of software development projects; they believe that being able to adapt to changing requirements at any point during the project life is a more realistic and better approach than attempting to define all requirements at the beginning of a project and then expending effort to control changes to the requirements.

XP prescribes a set of day-to-day practices for managers and developers; the practices are meant to embody and encourage particular values. Proponents believe that the exercise of these practices—which are traditional software engineering practices taken to so-called "extreme" levels—leads to a development process that is more responsive to customer needs ("agile") than traditional methods, while creating software of similar or better quality.

The main aim of XP is to lower the cost of change. In traditional system development methods (like SSADM) the requirements for the system are determined at the beginning of the development project and often fixed from that point on. This means that the cost of changing the requirements at a later stage will be high.

4.2. Activities of XP

Extreme Programming uses an object oriented approach as development paradigm and encompasses a set of rules and practices that occur within the context of four framework activities: planning, design, coding, and testing (Roger, 2010).

4.2.1 Planning

The planning activity (also called the planning game) begins with listening- a requirements gathering activity that enables the technical members of the XP team to understand the business context for the software and to set a broad feel for required output and major features and functionality. Listening leads to the

creation of a set of "stories" that describe required output, features, and functionality for software to be built.

4.2.2 Design

XP design rigorously follows the KIS (keep It Simple) principle. A simple design is always preferred over a more complex representation, in addition, the design provides implementation guidance for a story as it is written nothing less, nothing more. XP encourages the use of CRC (Class-Responsibility-Collaborator) cards as an effective mechanism for thinking about the software in an object-oriented context. The CRT cards are the only design work product produced as part of the XP process.

If a difficult design problem is encountered as part of the design of a story, XP recommends the immediate creation of an operational prototype of that portion of the design, the design prototype is implemented and evaluated. The intent is to lower risk when true implementation starts and to validate the original estimates for the story containing the design problem.

XP encourages Refactoring, which is the process of changing a software system in such a way that it does not alter the external behavior of the code yet improves the internal structure. It is a disciplined way to clean up code and modify, simplify the internal design that minimizes the chances of introducing bugs, in essence.

Because XP design uses virtually no notation and produces few, if any, work products other than CRC cards and spike solutions, design is viewed as a transient artifact that can and should be continually modified as construction proceeds. The intent of refactoring is to control these modifications by suggesting small design change that can radically improve the design. It should be noted, however, that the

effort required for refactoring can grow dramatically as the size of an application grows.

4.2.3 Coding

After stories are developed and preliminary design work is done, the team does not move to code, but rather develops a series of unit tests. Once the unit test has been created, the developer is better able to focus on what must be implemented to pass the test. Nothing extraneous is added KIS (keep It Simple).Once the code is complete, it can be unit-tested immediately, thereby providing instantaneous feedback to the developers.

A key concept during the coding activity (and one of the most talked about aspects of XP) is pair programming. XP recommends that two people work together at one computer workstation to create code for a story, this provides mechanism for real-time problem solving (two heads are often better than one) and real-time quality assurance (the code is reviewed as it is created). As pair programmers complete their work, the code they develop is integrated with the work of others.

4.2.4 Testing

We have already noted that the creation of unit tests before coding commences is a key element of the XP approach. The unit tests that are created should be implemented using a framework that enables them to be automated, hence, they can be executed easily and repeatedly. This encourages a regression testing strategy whenever code is modified (which is often, given the XP refactoring philosophy).

As the individual unit tests are organized into a universal testing suite integration and validation testing of the system can occur on a daily basis. This provides the XP team with a continual indication of progress and also can raise warning flags

early if things go awry. Fixing small problems every few hours takes less time than fixing huge problems just before the deadline, this is known as XP Acceptance Tests.

5. The Research Expected Outcomes

Developed the following student module's functions within an integrated learning management systems LID.

- Student profile
- Announcement
- Chat tools
- View learning material.

6. The Research Overview

Chapter 2(Background): contains definitions and some related works.
Chapter 3 (**Planning and Modelling Phases)**: presents the planning and designing of the specified functions.
Chapter 4(Construction Phase): present the coding and testing of the specified functions.
Chapter 5 (Conclusion): presents results summary and recommendations