Abstract

A Learning Management System (LMS) is a popular e-Learning tool that helps to easily create, deploy and manage online courses. LMS is widely accepted both in academia (universities, schools, etc.) and corporate (employee training programs, lifelong learning, etc.). The increasing demand of new features and functionalities within LMS form its diverse set of users like learner, teacher, author, institutional management, accreditation bodies, etc. makes the process of software design and development complex and challenging. Unfortunately, most of the LMS development approaches typically follow the same Software Engineering methodologies as used for development of non-educational software systems. Consequently, LMS developments often result in delays, incomplete software and faulty modules, and eventually could fail to meet its educational objectives.

This research work is focused on the software engineering methodologies for development of LMS software. It proposes some methodologies for the requirement engineering and software design phases of LMS development. Pertinent frameworks are introduced that employ established graphical, semi-formal and formal methods of software engineering in an integrated way. The requirement engineering methodology includes identification of the requirements from different stakeholder's perspectives, analyzing the requirements to group them on the level of abstractions and to map them with the quality attributes of the software. A model-based approach for checking traceability of requirements at different levels of abstraction is proposed.
A high-level design of LMS, represented by the software architectural design of the system, is developed by extending the UML meta-model based on the formal modeling style of ACME. The proposed approach also includes testing conformance of the design with the learning technology standard—Learning Technology System Architecture (LTS A). The architectural design is followed by a detailed design methodology that uses a semi-formal method—UML and a formal method—VDM in an integrated way. A verification technique to check consistency between different UML constructs and VDM-SL, and traceability of the requirements between the requirement specification and the design artifact is also proposed. The design methodology also includes a systematic approach for analysis and design of interface requirements by using formal methods in a uniform way for functional as well as interface requirements. Some add-ons to VDM-SL syntaxes are proposed to cover the interface-requirements of web-based applications like LMS.

Some contribution is also endeavored towards enriching LMS with some augmented features like learning path, personalized learning, and adaptive learning with help of application of some advanced computing techniques of Data Mining and Artificial Intelligence in the domain of LMS. Finally, a prototype implementation, CompTutor, is developed to implement some of the proposed features.