

Double Life Cycle

Alejandro Canales¹, Rubén Peredo¹ and Ivan Peredo¹

¹ Computer Science Research Center of National Polytechnic Institute, Col. Nueva Industrial Vallejo, Del. Gustavo A, Madero, D.F. 07738, Mexico City, Mexico
{acc, peredo}@cic.ipn.mx

Abstract. In this paper a new life cycle called “double” for the development of Web-Based Education Systems (WBES) is presented. These WBES are centered in the student and adapted to their personal necessities in intelligent form. The double life cycle assembles to the patterns of software development and instructional design. On the one hand, the software development pattern is supported under the methods and technical of the Domain Engineering (components), Learning Technology System Architecture of IEEE 1484, Sharable Content Object Reference Model (SCORM) of ADL, a Multi-Agents System and Service-Oriented Architecture for the reusability, accessibility and interoperability of the services. On the other hand, the instructional design pattern incorporates a mental model as the Conceptual Maps to transmit, build and generate appropriate knowledge to this educational environment type.

Keywords: WBES, SiDeC, IRLCOO, Evaluation System.

1 Introduction

Student-centered education pursues that sequencing, authoring content, pedagogic patterns, assessment, and evaluation processes meet the learning goals of the students. Also, the content and evaluation repositories must be suitable for the particular requirements of each individual. But at the same time, they have to be flexible and available for being tailored and used by a wide community of developers and students, respectively. It is necessary the development of a WBES that considers a whole diversity of requirements (technological and pedagogic) and provides the needed functionalities based on the facilities of the Web.

As well, the Domain Engineering has to consider the particular specifications claimed by the WBES in order to tailor solutions. Furthermore, authoring reusable components for content and evaluation tasks have to be fulfilled automatically as well as possible. The components help us to reduce the complexity, managing change, and reuse [1]. Thus, in order to deliver teaching-learning experiences, it is necessary the use of mental model as Conceptual Maps (CM).

Wherefore, the purpose of this paper is to show a new life cycle for the WBES development. In order to achieve this goal, this paper is organized as follows: In Section 2, the double life cycle is presented; whereas in Section 3, the instructional design pattern are analyzed. In Section 4 the software development pattern is

described. Afterwards, in Section 5 the SiDeC (authoring content) and Evaluation System are respectively depicted. Finally, in Section 6 the conclusions and future work are discussed.

2 Double Life Cycle

If the actual WBES are reviewed, we can note that the pedagogic aspect is considered but not in significant form. We believe in the incorporation of pedagogical aspects or an instructional design pattern inside the applications development process. Because the instructional design is a process that allows translating pedagogic principles of learning, in an action plan to develop learning content and activities as well as information and evaluation resources. It is a systemic process that allows developing the plan step by step and the results of each one serve as input to the consequent steps [2].

In conclusion, this suggests the incorporation of an instructional design pattern inside the software development process for WBES, and consequently we need a life cycle with this characteristic. But if the international standards as ISO/IEC 9001, 9003, 12207 and 15504 of Software Engineering are reviewed [3], they only contemplate the software development from a technological perspective basically and they do not deepen inside the requirements of the software application field, such as in this case is the education. In all life cycles referenced, only in the requirements step is possible to include the pedagogical requirements. We consider that the pedagogical aspects must intervene in a direct way in all processes inside the software development.

In Fig. 1 a double life cycle pattern is shown. It joins the software development pattern and the instructional design pattern, with the purpose of not only developing educational software based on technological or computational questions, but by adding, also, cognitive elements that collaborate in the acquisition of the students' knowledge.

The double life cycle provides to the developer facilities to go and come from a pedagogic extreme to other technological for the application design. This is because the software has an educational purpose; therefore the pedagogic principles guide the software technological development. Also, the double life cycle helps to the development and delivery fasts of software parts.

In the first phase of the double life cycle the software objectives are defined. For the second phase the software requirements so much pedagogic as technological are obtained. In the third phase the learning activities and the global design of the system are developed. It can be observed that the double life cycle allows in these three phases to go back and to advance among them during the software development with the purpose of incorporating the necessary information to design the software until reaching the goals traced for each phase.

In phase 4, a software version is developed that later in fifth phase is shown to the user. The result is refined in function of the user's feedback. This cycle is repeated until reaching appropriate software, since the characteristic of the double life cycle is in fact to be evolutionary. This characteristic is very important, because it provides

the possibility to change the product direction to halfway in response to the user's petitions (learner and tutors' developers). If it is used the evolutionary delivery carefully, it can improve the product quality, reduce the code size and produce a more uniform distribution of the development and test resources [4]. Finally, during sixth phase the final version to the user is delivered.

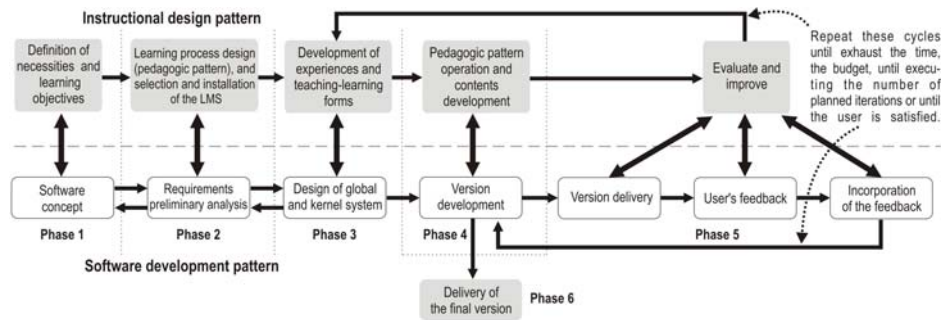


Fig. 1. Double life cycle.

3 Instructional Design Pattern

The instructional design pattern represents a substantial advance in the teaching-learning process (see Fig. 1). On the one hand, it incorporates cognitive elements (as CM) with the purpose of helping the students to maximize the knowledge acquisition. On the other hand, it is specifically designed to the Web environment.

The instructional design pattern allows generate a process where pedagogic principles of learning can be translated in an action plan for development of activities and learning contents, as well as evaluation and information resources [5].

A CM is a graphical technique used during the teaching-learning process. CM as instructional and learning strategy and as schematic resource or navigation map is used [6].

4 Software Development Pattern

The software development pattern is based on the Domain Engineering (see Fig. 1). The Domain Engineering aims to identify, build, classify and divulge software components [7]. While the traditional Software Engineering aims to the development of software system or application to satisfy some specific requirement for a particular necessity [8].

A software component is "a unit of composition with contractually specified interfaces and explicit context dependencies. A software component can be deployed

independently and is object to composition by third parties” [9]. Between the key issues of Domain Engineering is the aim for developing reusable software. Thus, components are widely seen by software engineers as a main technology to address the “software crisis”. The Industrial Software Revolution is based upon component software engineering. Between the reasons that explain the relevance of the component-oriented programming are: the high level of abstraction offered by this paradigm, and the current trends for authoring reusable component libraries, which support the development of applications for different domains. In addition, the three major goals pursued by Component-Oriented Programming are considered: Conquering complexity, managing change, and reusability [1].

4.1 IRLCOO Profile

With regards to the SCORM terminology, Intelligent Reusable Learning Components Object Oriented (IRLCOO [10]) are a special type of Sharable Content Object (SCO) that represent an alternative approach to content development, which is based on the Reusable Learning Object Strategy Definition stated by Advanced Distributed Learning (ADL) [11], IRLCOO are self-contained learning components that are organized as learning resources, which are accessed independently. IRLCOO are digital resources that can be reused to support WBES thru: live, streaming and prerecorded video and audio, a course module, animations, graphics, applications, Web pages, PDF and Office documents, and other pieces devoted to deliver complete experiences.

IRLCOO were developed with Flash 8. Flash is an integrator of media and have a powerful programming language denominated ActionScript 2.0 [12]. This language is completely Object Oriented and enables the design of client components that allows multimedia content. In addition, this IRLCOO development platform owns certain communication functionalities inside the Application Programming Interface (API) of the LMS, Multi-Agent System (MAS), and different frameworks, as AJAX [13], Hibernate [14], Struts [15], etc.), and dynamic load of Assets in Run-Time.

IRLCOO are meta-labeled with the purpose of complete a similar function as the product bar codes, which are used to identify the products and to determine certain characteristics specify of themselves. This contrast is made with the meta-labeled Resource Description Framework (RDF/XML) [16].

From a pedagogical perspective, each IRLCOO might play a specific role within an instructional design pattern. IRLCOO can be re-assembled to create new courses or sequenced to tailor individual learning paths. The use of IRLCOO deals with the following key issues: (1) The IRLCOO must be able to communicate with learning management systems (LMS) using a standardized method that does not depend on the system. (2) The sequence system, that usually is a module of the LMS, defines the navigation rules that a learner uses to move between IRLCOO. (3) IRLCOO own a description that enables designers to seek and find the appropriate IRLCOO for the right job. These considerations offer clear benefits, such as: IRLCOO enable mass-customization of learning with more personalized and content ‘just for the learner’, and for authors, there is the opportunity to seek existing IRLCOO within the organization or from external providers in order to reuse them, save time and money.

Furthermore, ActionScript 2.0 adds the component `WebServiceConnector` to connect to Web Services (WS) from the IRLCOO. The `WebServiceConnector` component enables the access to remote methods offered by a LMS through SOAP protocol. This gives to a WS the ability to accept parameters and return a result to the script, in other words, it is possible to access and join data between public or own WS and the IRLCOO.

5 SiDeC

In order to facilitate the development of learning content, it was built an authoring tool called eCourses Development System (SiDeC - Sistema de Desarrollo de eCursos) [17]. SiDeC is a tool based on components, which facilitates the authoring content by tutors who are not willing for handling multimedia applications. In addition, the Structure and Package of content multimedia is achieved by the use of IRLCOO, as the lowest level of content granularity.

According to the IEEE 1484 LTSA specification [18], SiDeC is used to construct Web-based courseware from the stored IRLCOO (Learning Resources) through of the coach in the way illustrated in Fig. 2.

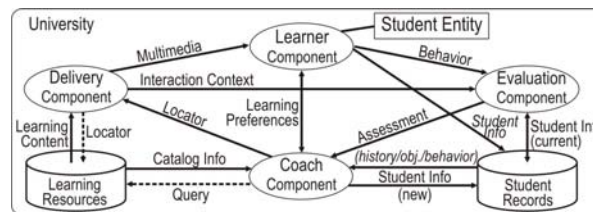


Fig. 2. LTSA of IEEE 1484 (layer 3).

SiDeC has a metadata tool for the generation of IRLCOO and on-line courses (see Fig. 3). This courseware estimates learners' metrics with the purpose to tailor their learning experiences. These deliverables are compliance with the specifications of the IRLCOO and with learning items of SCORM 1.2 Models (Content Aggregation, Sequencing and Navigation, and Run Time Environment) [11]. Metadata represent the specific description of the component and its contents, such as: title, description, keywords, learning objectives, item type, and rights of use. The metadata tool provides templates for entering metadata and storing each component in the SiDeC or another IMS/IEEE standard repository.

At this moment, the SiDeC lesson templates are based on the cognitive theory of CM [6], but in the future we will consider others theories such as: Based-problems learning (BPL), the cases method, and the project method.

In Fig. 4 the SiDeC implements the conceptual map as a navigation map, allowing to the learner interacts with content objects along the learning experiences. These experiences follow an instructional-teaching strategy. There kind of strategies carry

out modifications of the learning content structure. Such modifications are done by the learning experience designer with the objective of provide significant learning, and to teach the learners how to think [2].



Fig. 3. Learning content generated for the SiDeC.

Based on a CM the SiDeC represents the course structure that the student follows. The delivery process identifies a learning content for the student. The learning content owns IRLCOO associated with it. Afterwards, the delivery process launches (see Fig. 2) the IRLCOO and presents them to the student. Fig. 4 depicts how the course structure is organized as a manifest, and the learning content can be interpreted in a Learning Content Tree. A Learning Content Tree is a conceptual structure of learning activities managed by the delivery process for each learner. The Tree representation is just a different way for presenting content structure and navigation. This information is found in the manifest that is defined into the SCORM Content Aggregation Model (CAM) [11].

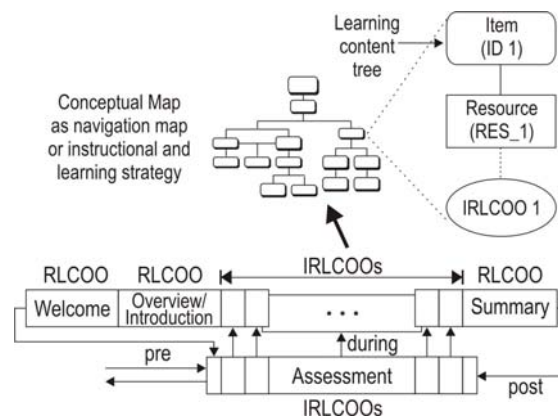


Fig. 4. The course structure.

According with Fig. 4, the next fragment code shows how the course structure is organized as *imsmanifest.xml*.

```
<manifest>
  <organizations>
    <organization>
      <item>
        <item identifier="ID1" identifierref="RES_1">
          <adlnav:presentation>
            <adlnav:navigationInterface>
              <adlnav:hideLMSUI>previous
            </adlnav:hideLMSUI>
              <adlnav:hideLMSUI>continue
            </adlnav:hideLMSUI>
            </adlnav:navigationInterface>
          </adlnav:presentation>
          <imsss:sequencing>
            <imsss:controlMode choice="false"
              flow="true"/>
            <imsss:rollupRules
              rollupObjectiveSatisfied="false"/>
          </imsss:sequencing>
        </item>
        <item> ... </item>
      </organization>
    </organizations>
    <resources>
      <resource identifier="RES_1"> ... </resource>
      ...
    </resources>
  </manifest>
```

6.1 Evaluation System

The Evaluation System is designed under the same philosophy used for the SiDeC. The functionality of the Evaluation System lays on the analysis of the learner's profile, which is built during the teaching-learning experiences. The profile is based on metrics that elicited from the learner's behavior at Run-Time. These measures are stored into the learner records that compose the profile (see Fig. 2). The generation of new sequences of courses is in function of the results obtained, besides the account of the adaptation level.

The Evaluation System combines IRLCOO, additional meta-labels, and a Java Agent platform. Also, some technologies of the Intelligence Artificial field are considered in order to recreate a Semantic Web environment. Semantic Web aims for assisting human users to achieve their online activities. Semantic Web offers plenty of advantages, such as: reduction of the complexity for potential developers, standardization of functionalities and attributes, definition of a set of specialized APIs, and deployment of a SWP.

All resources have a Universal Resource Identifier (URI). An URI can be a Unified Resource Locator (URL) or some other type of unique identifier. An identifier does not necessarily enable access to a resource. The XML layer is used to define the

SCORM metadata of IRLCOO that are used to interchange data over the Web. XML Schema tier corresponds to the language used to define the structure of metadata [19]. The RDF level is represented by the language used for describing all information and metadata sorts [16]. The Meta-ontology tier is devoted to define the semantic for establishing the usage of words and terms in the context of the vocabulary. Logical level corresponds to the reasoning used to establish consistency and correctness of data sets and to infer conclusions that are not explicitly stated [20].

In resume, the components and operation of the SiDeC and Evaluation System are outlined in Fig. 5. Basically the Evaluation System is fulfilled through two phases. The first phase is supported by the LMS, and is devoted to present the course and its structure. All the actions are registered and the presentation of the contents is realized with IRLCOO content. The evaluations are done by evaluating IRLCOO and in some cases by simulators based on IRLCOO. These processes are deployed by the Framework of Servlets/Java Server Pages/JavaBeans.

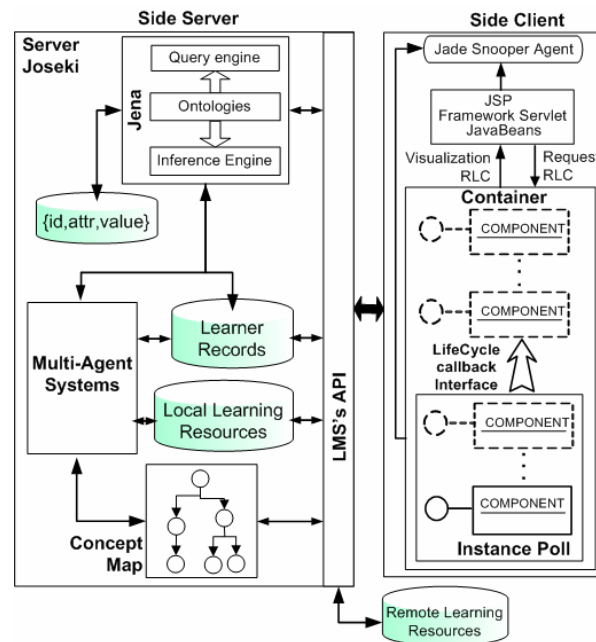


Fig. 5. Semantic Web Platform for WBE.

The second phase analyzes the learner's records carried out by the Server based on JADE MAS. This agent platform owns seven agents: Snooper, Buffer, Learner, Evaluation, Delivering, Coach, and Info. The fundamental idea is to automate the learner's analysis through the coach, and to give partial results that can be useful for the learner's final instruction. These agents are implemented as Java-Beans programs, which are embedded in the applications running both at the client and server sides. These agents employ the dynamic sequencing to change the course or assessment

sequence. The sequencing is defined for the instructional strategy based on CM and it employs the SCORM Sequencing/Navigation. Once the necessary information is received (sequence, kind of IRLCOO and localization, etc.), this is represented as a string dynamically constructed by the rule-based inference engine known as JENA [21] and JOSEKI server [22], to generate dynamic feedback.

6.2 Semantic Web Platform

The overall architecture of Semantic Web Platform, which includes three basic engine representing different aspects, is provided in Fig. 5.

1. The query engine receives queries and answers them by checking the content of the databases that were filled by info agent and inference engine.
2. The database manager is the backbone of the entire systems. It receives facts from the info agent, exchanges facts as input and output with the inference engine, and provide facts to the query engine.
3. The inference engine use facts and Meta-ontologies to derive additional factual knowledge that is only provided implicated. It frees knowledge providers from the burden of specifying each fact explicitly.

Again, Meta-ontologies are the overall structuring principle. The info agent uses them to extracts facts, the inference engine to infer facts, the database manager to structure the database, and query engine to provide help in formulating queries.

JENA was selected as the inference engine. It is a Java framework for building Semantic Web applications. It provides a programmatic environment for RDF, RDFS and OWL, SPARQL and includes a rule-based inference engine [21].

While JOSEKI was selected as Web API and server. It is an HTTP and SOAP engine supports the SPARQL Protocol and the SPARQL RDF Query language. SPARQL is developed by the W3C RDF Data Access Working Group [22].

7 Conclusions

This paper has introduced an instance of an adaptive and intelligent WBES. Our approach focus on: reusability, accessibility, and, interoperability of the learning contents, which are built as IRLCOO, as the main component for delivering teaching and evaluation content.

IRLCOO offer a common interface and functionality that makes easy the authoring of learning content that is delivered by dynamic sequencing. The IRLCOO accept feedback via assessment based upon MAS platform. The information provided is considered as a rough data because it is based on parameters elicited from the behavior of the student.

References

1. Wang, A., Qian, K.: Component-oriented programming. John Wiley & Sons, Inc., Publication (pp. 3–5), USA (2005)
2. Díaz-Barriga, F.: Educational strategies for a significant learning (Estrategias docentes para un aprendizaje significativo). 2nd ed. DF, Mc Graw Hill Publication, México (2002)
3. International Standards Organization, ISO 9001, 9003, 12207 and 15504, <http://www.iso.org>
4. Mc Connel, S.: Informatics Projects - Develop and Management (Desarrollo y Gestión de Proyectos Informáticos). Mc-Graw Hill Editorial, Spain (1997)
5. Díaz, J., Ramírez, V.: Instructional Design pattern (Modelo de diseño instruccional), <http://www.uv.mx/jdiaz/DisenoInstrucc/ModeloDisenoInstruccional2.htm>
6. Novak, J., Gowin, B.: Learning How to Learn, Cambridge University Press, Cambridge, USA (1984)
7. Simos, M., Klingler, C., Levine, L., Allemang, D.: Organization Domain Modeling (ODM), Guidebook –Version 2.0. Technical Report, Lockheed Martin Tactical Defense Systems, USA (1996)
8. Pressman, R.: Software Engineering – A practical vision (Ingeniería de Software - Un Enfoque Práctico). McGraw-Hill Editorial, Spain (2002)
9. Szyperski, C.: Component Software. Beyond OPP. Addison-Wesley Editorial, USA (1998)
10. Peredo, V. R., Balladares, O. L., Sheremetov, L.: Development of intelligent reusable learning objects for web-based education systems. *Expert Systems with Applications*, 28(2), 273–283 (2005)
11. Advanced Distributed Learning Consortium, <http://www.adlnet.org>
12. Macromedia, Inc., <http://www.macromedia.com>
13. Grane, D., Pascarello, E., James, D.: Ajax in Action. Manning Publications, Greenwich, USA (2006)
14. Peak, P., Heudecker, N.: Hibernate Quickly. Manning Publications, Greenwich, USA (2006)
15. Holmes, J.: Struts: The Complete Reference. Mc Graw Hill – Osborne Publications, Edited by Herbert Schild, California, USA (2004)
16. RDF specification, <http://www.w3.org/RDF/default.htm>
17. Canales, A., Peredo, R., Fabela, O., Sossa, H.: Architecture for development of WBES based on components and agents. *International Conference on Computing*, 223–228 (2006)
18. IEEE 1484.1, Draft Standard for Learning Technology - Learning Technology Systems Architecture (LTSA), <http://ieee.ltsc.org/wg1>
19. XML specifications, <http://www.w3.org/XML/>
20. Antoniou, G., Van Harmelen, F.: A Semantic Web Primer, The MIT Press, USA (2004)
21. JENA, <http://jena.sourceforge.net/>
22. JOSEKI server, <http://www.joseki.org/>