ESRI: Proposal of a Methodology Development of Educational Software Rich in Interaction

Misael N. CRUZ-PÉREZ, Carmen C. MARTÍNEZ-GIL, Aurea J.VICENTE-PINACHO, Zulma J. HERNÁNDEZ-PAXTIÁN.
Departamento de Informática, Universidad de la Cañada.
Teotitlán de Flores Magón, Oaxaca C.P 68540, México.

The educational software are emerging to be one of the pillars of the education how tools to facilitate the teaching-learning process. However, in recent years, a lot of software has been developed in a disorganized manner and poorly documented, so it is necessary to establish a disciplined methodology that guides its formal development.
In this paper we propose a methodology called ESRI (Methodology Development of Educational Software Rich in Interaction). This methodology is the result of following a methodological framework that includes the combination of ISE methodology (Software Engineering Education) and the methodology UCD (User Centered Design). Moreover, in addition to these methodologies ESRI is enriched with models such as MoProSoft (Process Model for Software Industry), quality standards for software products (ISO / IEC 9126) and contributions based in software development using methodologies such as TSP (Team Software Process) and object-oriented modeling by UML.
ESRI follows the scheme of Evolutionary Process Model and consists of five phases: Analysis, Specification of requirements, Design, Development and Evaluation which aims to develop educational software in an organized way, that transmit the educational content effectively according to user characteristics, friendly interface and easy enough to use.

Keywords: educational software, software development methodology, interaction, software process, teaching-learning.

1. INTRODUCCIÓN

Márquez, Gross and Mena, referenced in Fernández and Delavaut [1], they agree that educational software are computer programs developed for the specific purpose of being used as a teaching tool to facilitate the teaching and learning of a specific topic. The characteristics of educational software, are generally based on the materials to assess, however, Márquez [2] synthesizes them into two groups: 1) the educational and functional characteristics, which include ease of use, versatility didactic capacity motivation, fitness users, potential teaching resources, assessment, explanatory and creative approach, and promoting the initiative and learning and 2) specifications, which include the quality of the visual environment, quality and quantity of multimedia elements, quality, structure and content navigation, interaction and reliable execution.
Currently educational software has become one of the pillars of distance education system and this is shaping up to be a basic tool for future generations, however given its growing development in recent years, much of it has been developed so disorganized and poorly documented, so it became necessary to establish a disciplined methodology for development that meets the specific needs of educational software [3]. That is why authors like Galvis, by his ISE Methodology for the selection or development of Computerized Educational Materials [4], and Cataldi, by his methodology of design, development and evaluation of educational software [5], have been established first formal guidelines for the development of educational software.
This paper presents a proposed methodology to develop educational software by combining the best features of the methodology ISE Galvis and UCD methodology.

2. ENRICH A METHODOLOGY FOR DEVELOMENT OF EDUCATIONAL SOFTWARE

Because the educational software is intended to serve as a teaching tool, it must be designed thinking always ensure that students learn the content educational software attempts to transmit at the same time to generate interest and motivation to learn with this medium.
The ISE methodology develops educational materials for computer and consists of five phases: Analysis, Design, Development, Pilot test and Field test, these phases can identify if there is a problem and proposes a pedagogical solution that considers the adaptation or development software. This methodology allows identifying if there is a pedagogical problem and it proposes a solution that considers the adaptation or development of software. ISE incorporates pedagogical aspects, tests and constantly adapts the software to the specific needs to ensure that students learn effectively. ISE is characterized by deepening the analysis phase include compliance with educational objectives in design and theories of learning during the development process [4].
Meanwhile, UCD methodology is focused on designing and developing user-centric software, its phases are (needs analysis, user and task analysis, functional analysis, requirements analysis, usability Match Specifications, design, prototype and Evaluation). UDC allows developing interactive systems easy to use with friendly interfaces that are of interest to users [6].
Both ISE and UCD methodology have elements that complement each other to develop software that meets the features synthesized by Marquis (see above p.1). Having studied both methodologies there is concern us create a methodology to develop educational software that encourages meaningful learning by incorporating the educational component and learning theories, it is easy to use and respond to users' specific characteristics, and it is interactive enough to arouse the interest of users to continue using the software. To achieve this, we propose a software development methodology resulting from the hybridization of the ISE and UCD methodologies: the Methodology of Educational Software Development Rich in Interaction (ESRI).

3. METHODOLOGY

The process for developing the ESRI methodology is summarized in Figure 1.
In accordance with point 1 of the methodology (Figure 1), we performed a synthesis of the phases and activities that make both the ISE and UCD methodology to facilitate the identifica-

tion of similarities and correspondences of both methodologies. See Table 1.

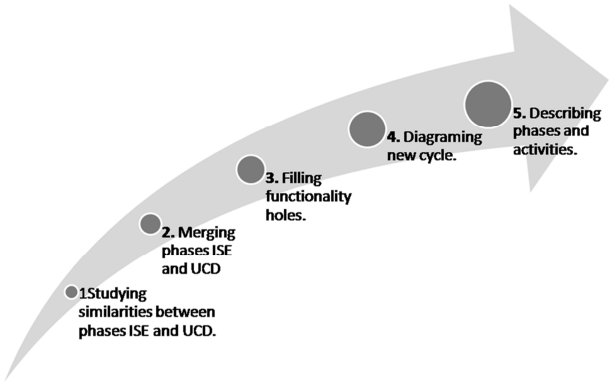

Fig. 1. Process for developing the ESRI methodology.

As shown in Table 1, there is a very close correspondence between the ISE methodology and UCD considered as both phases are very similar and can be fused to make the best of ESRI containing both. In accordance with section 2 of the methodology (see Figure 1) were merged phases and activities of Table 1, these phases and activities are the result of study and analyze the differences and similarities, adapt and combine ISE and UCD methodologies, Table 2 shows the result.

Table 1. Correspondence between phases of ISE and UCD.

ISE methodology	UCD methodology
P1. Analysis of educational needs	P1. Needs Analysis.
	P2. Analysis users.
	P3. Functional analysis.
P2. Desing	P4. Requirements analysis.
	P5. Match Specifications usability.
	P6. Desing.
P3. Develop.	P7. Prototype..
P4. Pilot test.	P8. Evaluation.
P5. Field test.	No correspondence.

In accordance with section 3 of the methodology (see Figure 1) underwent a second revision to incorporate mechanisms to ensure the functionality of the system. To achieve functionality, we made contributions based on experience in developing software systems using different methodologies such as TSP (Team software Process) [7] and UCD [6], object-oriented modeling using UML (UnifiedModelingLanguage) [8], and the study of development models for software industry as MoProSoft [9] and standards for product quality software such as ISO / IEC 9126 [10]. In the design phase included modeling using UML diagrams (at least by use case diagrams, class, and sequence) and Entity-Relationship diagrams of the database, allowing the system to model in a unified language understandable to almost any software engineer and programmer. In the development phase incorporated unit testing and integration complemented with usability testing, allows the system to be functional enough to be released and its user friendly interface and interactive with the user.

According to section 4 of the methodology (see Figure 1) is represented graphically the process of developing the methodology ESRI, although in point 2 of the methodology already have a first approximation of the phases and sub-phases that shape, determine the outline of the software process model to follow, for software process model means the workflow between processes, activities, tasks and products required to develop high quality software without detailing specific activities (own definition based on [11] and [12]). Pressman considers four types of software process models that are called prescriptive process models: The waterfall model, Incremental process models, Evolutionary process model and specialized models in the process. After studying in detail the methods we determine that SERI development methodology will follow the outline of the Evolutionary Process Model considering that educational software should be constantly evolving to suit the emerging needs of users, this model of evolutionary processes to follow a cyclical path and nonlinear, in each iteration can add new features to the software, developing increasingly complex versions that meet the new requirements and adapt to emerging new needs.

Table 2. Result of merging the ISE and UCD methodology

Phases and sub-phases.	Activities.
P1. Analysis.	
SubP1.1 Analysis of educational needs.	Consult sources, analyze possible causes to the problem identified, analyze alternative solutions, define the type of software and the target population, pedagogical and didactic principles apply and develop a plan of development.
SubP1.2 Analysis users.	Identify and define the characteristics of primary and secondary users.
P2. Requirements Specification.	Software overview, functional and non-functional requirements, restrictions.
P3. Desing.	
SubP3.1 Educational Design.	Specify the content and educational need to be treated, raise learning objectives, evaluation mechanisms and motivation.
SubP3.2 Interface Design.	Determine ES devices to use, interface design microworlds (scenarios), perform visual and content organization, navigation map.
SubP3.3 Computational design.	Functional and structural specification by UML, implementation, evaluation and adjustment paper prototype.
P4. Development.	
P4.1 Program modules.	Select tool development, program microworlds and supplementary materials to develop software.
P4.2 Unit Testing and integration.	For each module coded white box testing, implement corrections to errors detected, the system integration module and integration testing.
P.5 Evaluation.	Black box testing and implement corrections to errors found, plan and conduct usability testing and implement corrections to errors detected States, eventually writing a user manual.

4. ESRI: METHODOLOGY OF EDUCATIONAL SOFTWARE DEVELOPMENT RICH IN INTERACTION

The methodology ESRI is characterized by incorporating educational issues as elements to develop functional and usable educational software that meet the users' needs, always seeking to teach that content to be learned by the students effectively. This is what differentiates ESRI methodology as the ISE methodologies that focuses on developing educational materials but neglects structural modeling system software needed to develop a

stable and complex. Furthermore, the UCD methodology focuses on interface design and usability but explicitly incorporates pedagogical and structural aspects of the system and consolidated methodologies and Team Software Process (TSP) and Personal Software Process (PSP). TSP is aimed to build a project team with different roles to develop software in software development companies large or PSP where the whole development process lies with one person and therefore difficult software development issues is enriched effective teaching educational content.

For this and as a result of applying the logical methodological process of Figure 1, Figure 2 shows the scheme ESRI methodology consists of five phases and seven subphases.

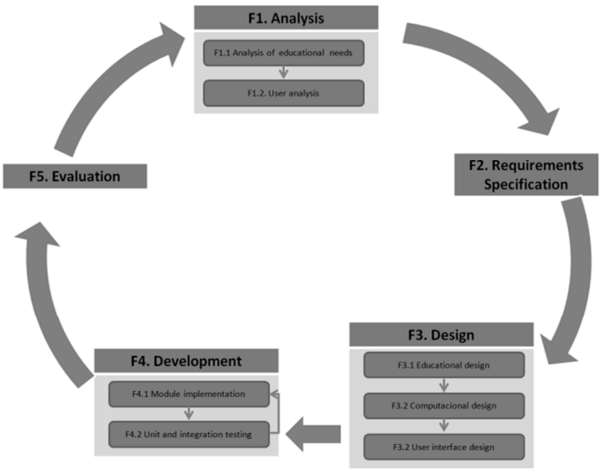

Fig. 2. ESRI development methodology under the scheme of model evolutionary processes.

In accordance with section 5 of the methodology (Figure 1), following the methodology described formally ESRI detailing each of the phases and subphases inside them, and the activities necessary to carry it out and the products that are generated . It is clear that the general diagrams of activities follow the notation of Process Change Method Guidance Resourse the SEI (Software Engineering Institute) which provides a practical guide to determine the current state of the process and its activities [13].

For the development of educational software under the ESRI methodology are required at least four roles: *Team Leader*, *Team Development*, *Professor of the area* and a *pedagogue*.

Phase 1. Analysis.

In this first phase are detected educational problems, possible solutions are proposed and analyzed in detail the characteristics of the users. It consists of the subphases 1.1 and 1.2.

Subphase 1.1. Analysis of educational needs: lies in consulting various sources of information to identify educational problems that are occurring and propose alternative solutions that include software development, you must also make clear the role to play the computer, characterizing the software, also must define the target population to which the software will consider the pedagogical principles applicable in the latter should make an analysis of the teaching-learning process is implemented in the classroom and on this basis selecting teaching techniques, learning tools, educational models, etc. to use in software development. Based on the above defined system type, users and creating the development plan, Figure 3 shows the general diagram of activities subphase 1.1.

Subphase 1.2. Analysis users: is to get to know the characteristics of users to create a profile (such as age, gender, computer experience, skills, interests, etc.) And thus to design and develop software according their needs and specific characteristics also need to know their educational characteristics (such as grade level, learning styles, prior knowledge, school environment, etc.) for adapting educational content according to their current knowledge and at the school level that meet. Figure 4 shows the diagram of operations.

Phase 2. Requirements Specification.

This phase is intended to translate the formal requirements of the software before being designed and built by describing the software, specifying functional requirements, non-functional and software constraints. The importance of this phase lies in assisting the team leader and the development team to better understand the problem whose solution will work and you make clear the result they expect. Figure 5 shows the diagram of operations for this phase.

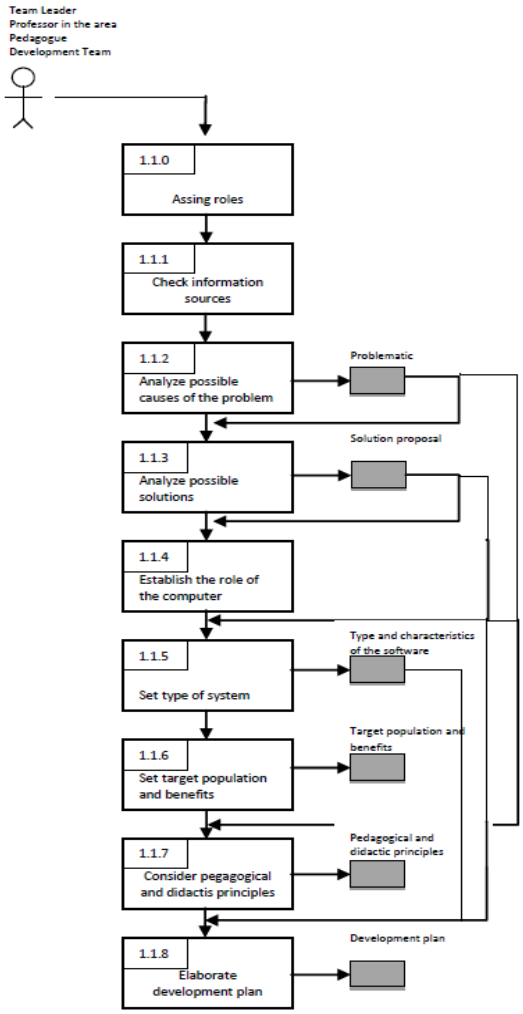

Fig. 3. General diagram of activities subphase 1.1.

Phase 3. Design.

This phase consists of designing the software on three levels: Educational design, Computational design and Interface design.

Subphase 3.1. Educational desing: at this stage should be set content to teach and attend educational need that software for each pose content learning objectives, motivation mechanisms to generate interest in the students as they use the software because otherwise find it useful motivator and hardly reach the learning objectives also should raise mechanisms to

evaluate the learning gained by the students and quantitatively determine the level of achievement for each learning objective raised. See Figure 6.

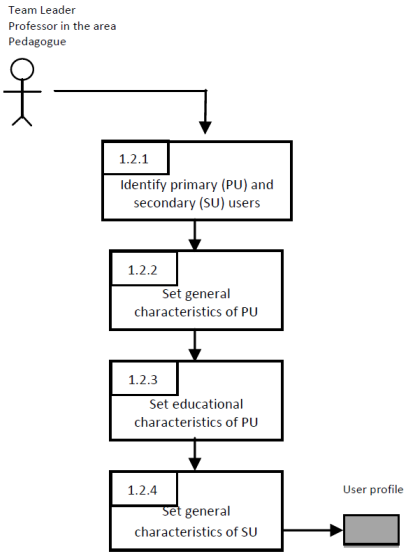

Fig. 4 General diagram of activities subphase 1.2.

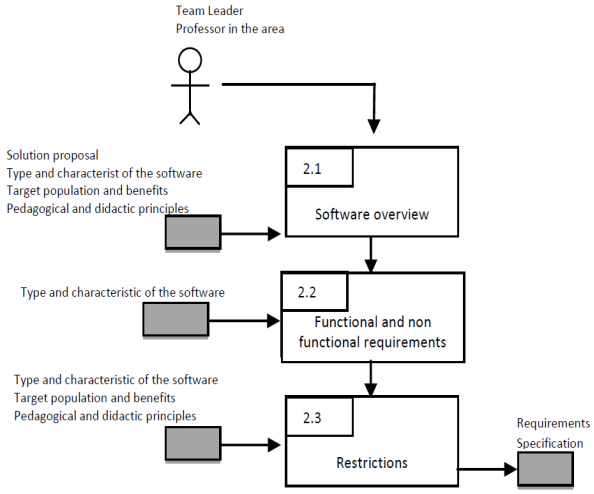

Fig. 5 General diagram of activities phase 2.

Subphase 3.2. Computational Design: focus on modeling the system's internal structure, ESRI recommends using UML as a standard language for expressing the requirements specification and software architecture in a language understandable and unambiguous. UML provides several types of diagrams in grouped structure diagrams, behavior diagrams and interaction, but the methodology suggests ESRI perform computational design with at least one group of diagrams being the highest priority, according to Ambler, cited in [14], the class diagram (Structure), Sequence Diagram (Interaction) and the use case diagram or Activity (Behaviour), these three types of UML diagrams and relational database diagrams are proposed as minimum that should be used to model the independent educational software features, however it is the responsibility of the development team incorporated many diagrams deems necessary to uniquely capture software. Figure 7 shows the general diagram of activities for the subphase 3.2.

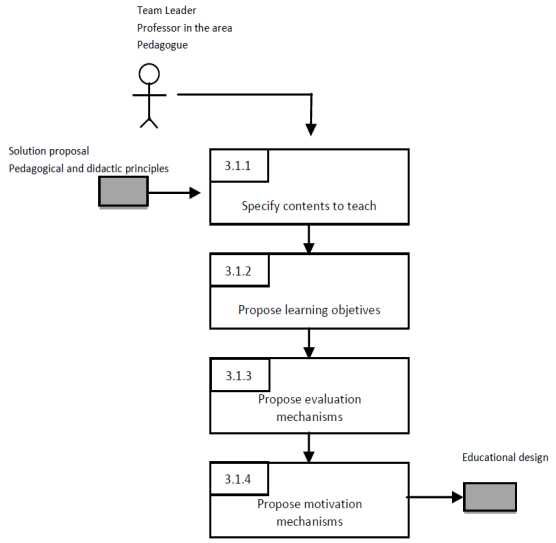

Fig. 6 General diagram of activities subphase 3.1.

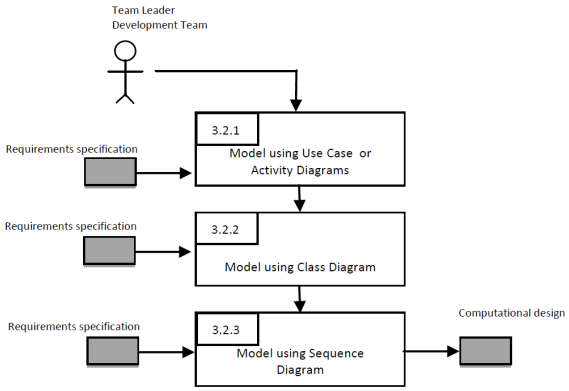

Fig. 7 General diagram of activities subphase 3.2.

Subphase 3.3. Interface design: this phase is concentrated on the area to which the user communicates with the software, emphasizing how the content will be presented, its deployment and organization by the screen, thereby specifying the input and output devices of which will use the software, interfaces are designed to apply usability principles, the content is organized and finally verifies the design at all three levels to detect and correct inconsistencies. Figure 8 shows the general diagram of this subphase activities.

Phase 4. Development.

Is to continue the design phase and implement the system using a programming language, supplemented by a series of tests to ensure that the modules are functional; Once that stage is going to have all the modules tested both individually and integrated and functional version of the system at the code level, logical structure and compatibility between modules. It consists of the subphases 4.1 and 4.2.

Subphase 4.1. Module implementation: it is to code each of the modules that were designed in previous phases, to do this, we selected the programming language and other appropriate technology tools for the implementation of each module designed. Figure 9 shows the general diagram of activities for this subphase.

Subphase 4.2. Unit testing and integration: it is to test modules one by one the previous phase scheduled to test its functionality, performance and coupling testing it performed white box testing and integration testing, during the process of

writing the maintenance manual. Figure 10 shows the general diagram of activities for this subphase.

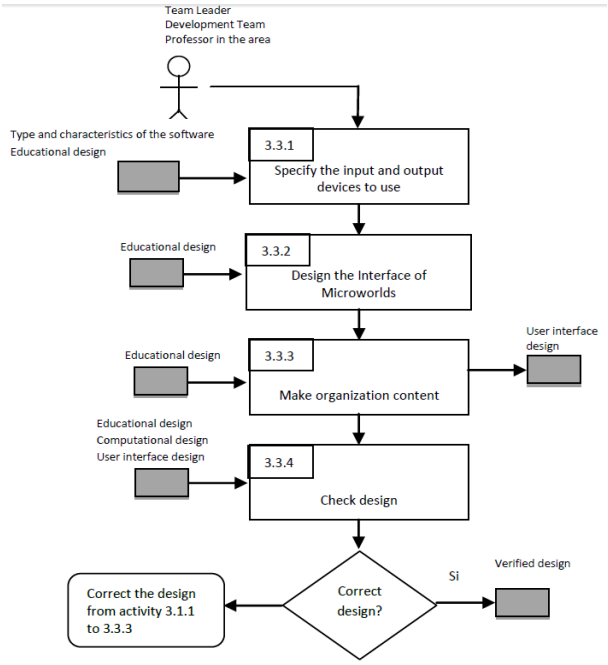

Fig. 8 General diagram of activities subphase 3.3.

Phase 5. Evaluation.

In the later phases it was possible to have the modules developed and tested, however, since is incorporated into the development methodology an interaction component, it is necessary do interface test to increment the user-system interaction and check that the system developed adapt to user characteristics; this is achieved by black box testing and usability test at the same time be implement the improvements detected in each test. Figure 11 shows the general diagram of activities for this phase.

To conclude the phase 5, an evaluation is obtained the first stable version of the software and her documentation, so it can be released for end users.

The SERI methodology currently is implementing to develop bilingual educational software for teaching-learning of indigenous language Cuicateca as a strategy for preservation and dissemination of the original languages through the TIC´s. At the moment have been applied the Analysis, Requirements specification and part of Design phases and has been observed that the proper application of the methodology can guide the development of the project without ambiguities and effectively.

5. CONCLUSIONS

Since educational software is shaping up to be a pillar of the modern education system, must be built under of software development methodologies that incorporate the technical components of software development, the didactic and interactive part. This was achieved by merging the ISE and UCD methodology for obtain the SERI methodology that responds to the demands of the development of educational software.

With the SERI methodology composed of five phases under a scheme of evolutionary processes, it is expected to develop educational software in an organized way, documented and that respond to emerging needs of users and changes in educational environment, developed under the formal modeling software guidelines and above all encouraging greater user-system interaction, hoping that the content taught by software effectively be learned by students.

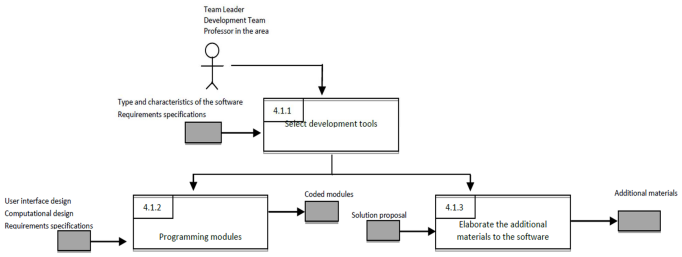

Fig. 9 General diagram of activities subphase 4.1.

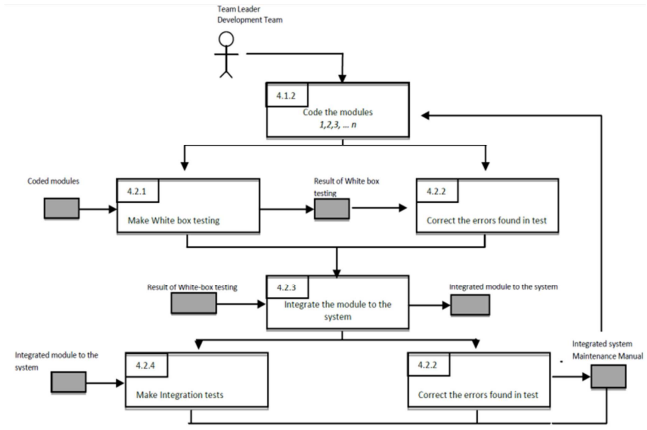

Fig. 10 General diagram of activities subphase 4.2.

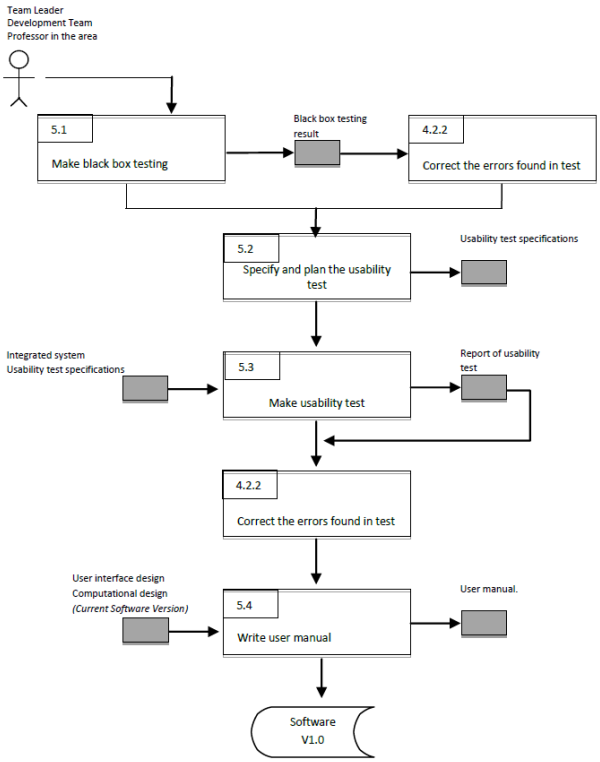

Fig. 11 General diagram of activities phase 5.

6. FUTURE WORK

As future work, we will continue with the implementation of the remaining phases of the ESRI methodology, to prove their effectiveness in developing bilingual educational software in the teaching and learning of indigenous language Cuicateca.

7.-REFERENCIAS

[1] R. Fernández and M. Delavaut, **Educación y tecnología un binomio excepcional,** Argentina: Grupo Editor K, 2008.

[2] P. Marqués, "Evaluación y selección de software educativo", **Comunicación y pedagogía**, No. 185, 2002, pp. 37-37.

[3] Z. Cataldi, F. Lage, R. Pessacq, and R. García, **Ingeniería de Software Educativo**, V Congreso Internacional de Ingeniería Informática (Argentina), pp. 185-199, Universidad de Buenos aires, 1998.

[4] A. Galvis, **Ingeniería de software educativo**. Colombia: EdicionesUniandes, 1992.

[5] Z. Cataldi, **Metodología de diseño, desarrollo y evaluación de software educativo**. Tesis de maestría, Universidad Nacional de la Plata, 2000.

[6] D. McCracken and R. Wolfe, **User-Centered Website Development**, A Human-Computer interaction approach. EstadosUnidos: Prentice Hall, 2004.

[7] W. Humprey, **Introduction to the Team Software Process**, Massachusetts: Addison Wesley, 2000.

[8] M. Fowler and K. Scott, **UML gota a gota**. México: Addison Wesley Longman de México, 1999.

[9] H. Oktaba, C. Alquicira, A. Ramos, G. Quintanilla, M. Ruvalcaba, et al. **Modelo de Procesos para la Industria Software, MoProSoft V1.3,** México: Secretaria de Economía, 2005.

[10] International Standard Organization and International Electrical Technical Commission, **ISO/IEC 9126 Standars for Software Engineering - Product Quality**. Versión resumida extraída el 17 de enero, 2012 de: http://wwwagse.informatik.uni kl.de/teaching/re/ws2010/ISO9126_Abstract.pdf

[11] R. Pressman, **Ingeniería del Software: Un enfoque práctico**, México: MacGraw- Hill, 2006.

[12] I. Sommerville, **Ingeniería de software**, México: Pearson Educación, 2002.

[13] P. Fowler, B.Middlecoat and S. Yo, **Lessons Learned Collaborating on a Process for SPI at Xerox (CMU/SEI-99-TR-006, ADA373332)**, Pittsburg USA: Software Engineering Institute, 1999.

[14] www.economicasunp.edu.ar (s.f) Los Diagramas UML 2.0. Reviewed on March 1, 2012**:** http://www.economicasunp.edu.ar/home/index.php?option=com_content&view=article&id=58:carreras&catid=35&Itemid=13