Chapter VI
Requirements Engineering: A Review of Processes and Techniques

Fernando Flores
Autonomous University of Aguascalientes, Mexico

Manuel Mora
Autonomous University of Aguascalientes, Mexico

Francisco Alvarez
Autonomous University of Aguascalientes, Mexico

Rory O'Connor
Dublin City University, Ireland

Jorge Macias-Luévano
Autonomous University of Aguascalientes, Mexico

ABSTRACT

Requirements engineering is the process of discovering the purpose and implicit needs of a software system that will be developed and making explicit, complete, and non ambiguous their specification. Its relevance is based in that omission or mistakes generated during this phase and corrected in later phases of a system development lifecycle, will cause cost overruns and delays to the project, as well as incomplete software. This chapter, by using a conceptual research approach, reviews the literature for developing a review of types of requirements, and the processes, activities, and techniques used. Analysis and synthesis of such findings permit to posit a generic requirements engineering process. Implications, trends, and challenges are then reported. While its execution is being mandatory in most SDLCs, it is done partially. Furthermore, the emergence of advanced services-oriented technologies suggests further research for identifying what of the present knowledge is useful and what is needed. This research is an initial effort to synthesize accumulated knowledge.

INTRODUCTION

In the field of software engineering several process models have been formulated to guide the development of software systems (e.g., software or system development life-cycle). Independent of what process model is selected by a development team, all activities conducted can be grouped into three main macro-phases: **s*ystem definition*** (*software specification of functional and constrain requirements),* ***system development*** *(design and building),* and ***system deployment*** *(software implementation, software validation (to confirm that the new software system satisfies the users' needs)* and *software evolution (evolution of the users' requirements as the users' reality evolves))* (Sage & Armstrong, 2000; Sommerville, 2002).

First, macro-phase's activities have been studied by the requirements engineering (RE) discipline, which can be defined as: "the process of discovering the purpose of the software system by identifying stakeholders and their needs, and documenting these in a form that is amenable to analysis, communication, and subsequent implementation" (Nuseibeh & Easternbrook, 2000). The overall goal of RE is to elicit valid users´ requirements because the strong impact on quality and cost of the final software product. Accordingly to Jin et al (1998): "…errors made at this stage are extremely expensive to correct when are discovered during testing or during actual working." However, even though such evidence of relevance and that RE has been identified (Sommerville, 2005) as essential for successful software development, these activities are often overlapped, uncompleted, or missed in development projects. As Sumano (1999) alerts "it is a general practice not to do it well, or do it faster and careless, because they do not have enough time or because they do not know a good methodology to do it." Consequently, it is possible multiple errors are introduced in early activities and not discovered until later phases of the lifecycle raising the project costs and exceeding the project deadlines.

In this chapter, we use a conceptual research methodology (Glass, Vessey, & Ramesh, 2002; Mora, 2004) to review the state of the art on the process and techniques used in software requirements engineering for software products to answer the following research questions: (a) How can the software requirements be classified?, (b) How can the main processes, activities, and techniques proposed by the software requirements engineering, be organized ?, and (c) Can these processes, activities, and techniques be synthesized in a theoretically-developed generic process of software requirements engineering? According to Mora (2004), despite several sources report the utilization of a conceptual research approach and its wide usage in the domain of the software engineering (43%) (Glass et al., 2002), there is little detailed literature on how to use this research method. Counelis (2000) quoted by Mora (2004) indicates that conceptual research is part of the research methods that study ideas, concepts or constructs on real objects rather than study them directly. Despite scarce literature, Mora (2004) reports that several studies consider the conceptual research method as common as the survey, experimental, and case study methods. This chapter then uses the process described in Mora (2004) that consists in the following phases: (1st) formulation of the research problem; (2nd) analysis of related studies; (3rd) development of the conceptual artifact; and (4th) validation of the conceptual artifact. The first phase and second phases are similar to other research methods. In the third phase, two activities are conducted: the development of a high-level framework/model and the development of low-level details of specific components selected from the high-level framework/model. This third phase is a creativity-intensive process guided by the findings, contributions, and limitations found in the second phase and a set of preliminary proforms that are fixed through an iterative process (Andoh-Baidoo, White, & Kasper, 2004). In the last phase, the conceptual artifact's validation is developed through: face validity from a panel of experts, logical argumentation, or proof of concept developing a prototype or pilot survey.

The objectives of this research are: (a) to develop an updated classification of software requirements,