Chapter 6
A Bayesian Network for Predicting the Need for a Requirements Review

José del Sagrado Martínez
University of Almería, Spain

Isabel María del Águila Cano
University of Almería, Spain

ABSTRACT
One of the major problems when developing complex software systems is that of Requirement Engineering. The methodologies usually propose iterations in which requirements are to be reviewed and re-written until the final version is obtained. This chapter focuses on the construction of “Requisites”, a Bayesian network designed to be used as a predictor that tells us whether a requirements specification has enough quality to be considered as a baseline. Requisites have been defined using several information sources, such as standards and reports, and through interaction with experts, in order to structure and quantify the final model. This Bayesian network reflects the knowledge needed when assessing a requirements specification. The authors show how Requisites can be used through the study of some use cases. After the propagation over the network of information collected about the certainty of a subset of variables, the value predicted will determine if the requirements specification has to be revised.

INTRODUCTION
Product software is a worldwide industry providing the software applications found in countless types of systems, such as telecommunications, military, industrial processes, and entertainment. This software must be developed just as any other industrial product, by applying engineering methods to ensure a quality final product. Software Engineering (SE), appears in this context, and has been defined as: “The application of a systematic, disciplined, and quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software.” (IEEE, 1990, p. 67).

SWEBOK (Software Engineering Body of Knowledge) (Abran et al., 2004), one of the major works in SE, gives us an agreed description of scope of SE, defining ten related areas of knowledge: software requirements, software design, software construction, software testing, software maintenance,
A Bayesian Network for Predicting the Need for a Requirements Review

software configuration management, software engineering management, software engineering process, software engineering tools and methods, and software quality.

This chapter deals with how to apply a specific formalism, Bayesian Network (Pearl, 1988; Jensen, 2001), which originated in artificial intelligence and knowledge engineering fields for the purpose of enhancing specific activities related to the one of these SE knowledge areas, namely software requirements. Requirements play a key role in determining the success or failure of software projects and in determining the quality of software to be developed. This knowledge area has been selected for two basic reasons. On the hand, this area is different from the others because requirements reside basically in the problem space whereas other software artefacts reside in the solution space (Cheng & Atlee, 2007), and on the other hand, when requirement-related tasks are poorly defined or executed, the software product is typically unsatisfactory (Sommerville, 2006; Standish Group 1994, 2003), and therefore, any improvement in requirements will favourably affect the whole software lifecycle.

The requirements in any particular software project are usually a complex combination of requirements from stakeholders, and their identification is not a simple single step, but rather a cyclic activity (Sommerville, 2005; Nuseibeh & Easterbrook, 2000; Sawyer, Sommerville, & Viller 1997). Moreover, these requirements evolve and change, but at some point, a contract, called the requirement specification, needs to be agreed upon to generate a starting point for the project. This chapter explores the use of Bayesian networks to represent the relationships between various factors and the quality of requirements specification.

The rest of the chapter is organized in six sections. Section 2 describes how the main tasks involved in defining software requirements have evolved, resulting in the field known as requirement engineering. Section 3 outlines how requirement engineering can be improved, specifically focusing on Bayesian networks. This formalism is described in Section 4. Section 5 describes the development of our Bayesian network model, Requisites. Once it is obtained we show how to use it in order to give advice on the goodness of requirement specifications. Finally, we present the future research directions and the conclusions.

FROM SOFTWARE REQUIREMENT TO REQUIREMENT ENGINEERING

Software requirements express the needs and constraints placed on a software product that contribute to the solution of some real world problem (Kotonya & Sommerville, 2000). The process or set of tasks applied to find out, analyse, document and check these needs and constraints, have nowadays become the category of a subject called Requirement Engineering (RE).

Traditionally, obtaining requirements has been considered a fuzzy step in the software development lifecycle, in which a set of informal ideas must be translated into formal expressions. Nevertheless, tasks related to requirements are regarded as a key stage in software development, because “the primary measure of the success and quality of a software system is the degree to which it meets the purpose for which it was intended” (Nuseibeh & Easterbrook, 2000, p. 37), which is client satisfaction.

Requirements have different characteristics or properties that can be used as classification criteria, so usually, the meaning of “requirement” is narrowed down with adjective labels: system, hardware, software, user, client, functional, non-functional, performance, etc. (Duran & Bernárdez, 2000; Sommerville, 2006). Requirements can be classified according to their scope, leading to system requirements, hardware requirements or software requirements. They can also be classified by the nature of the feature they describe, leading to functional requirements, non-functional requirements or domain requirements. Finally, depending
Related Content

Particle Swarm Optimization-Based Session Key Generation for Wireless Communication (PSOSKG)
Arindam Sarkar and Jyotsna Kumar Mandal (2015). Handbook of Research on Swarm Intelligence in Engineering (pp. 640-677).
www.igi-global.com/chapter/particle-swarm-optimization-based-session-key-generation-for-wireless-communication-psoskg/131265?camid=4v1a

A Multi-Objective Fuzzy Ant Colony Optimization Algorithm for Virtual Machine Placement
www.igi-global.com/article/a-multi-objective-fuzzy-ant-colony-optimization-algorithm-for-virtual-machine-placement/170558?camid=4v1a

DEAL: A Distributed Authorization Language for Ambient Intelligence
Irini Genitsaridi, Antonis Bikakis and Grigoris Antoniou (2013). Pervasive and Ubiquitous Technology Innovations for Ambient Intelligence Environments (pp. 188-204).
www.igi-global.com/chapter/deal-distributed-authorization-language-ambient/68937?camid=4v1a

Fuzzy Logic-Based Round Trip Time Scaling and Scheduling in High Speed TCP Stacks
www.igi-global.com/article/fuzzy-logic-based-round-trip/77863?camid=4v1a