# Architecture for the Use of Synergies between Knowledge Engineering and Requirements Engineering

José del Sagrado, Isabel M. del Águila, and Francisco J. Orellana

Dpt. Languages and Computation,
Ctra Sacramento s/n, 04120 University of Almería, Spain
{jsagrado,imaguila,fjorella}@ual.es

**Abstract.** The application of Artificial Intelligence techniques in the processes of Software Engineering is achieving good results in those activities that require the use of expert knowledge. Within Software Engineering, the activities related to requirements become a suitable target for these techniques, since a good or bad execution of these tasks has a strong impact in the quality of the final software product. Hence, a tool to support the decision makers during these activities is highly desired. This work presents a three-layer architecture, which provides a seamless integration between Knowledge Engineering and Requirement Engineering. The architecture is instantiated into a CARE (Computer-Aided Engineering Requirement) tool that integrates some Artificial Intelligence techniques: Requisites, a Bayesian network used to validate the specification of the requirements of a project, and metaheuristic techniques (simulated annealing, genetic algorithm and an ant colony system) to the selection of the requirements that have to be included into the final software product.

**Keywords:** Requirement management, bayesian network, computer aided requirement engineering.

## 1 Introduction

The software development has been supported by Artificial Intelligence (AI) techniques for more than 20 years, since the appearance of the first intelligent editors. Currently we are witnessing a reemergence of AI and Software Engineering (SE), which has become a valid and potentially very valuable research field [15]. Expert knowledge is involved in every software development project since developers must face numerous decision tasks during requirements, analysis, design, and implementation stages. Therefore, if expert knowledge could be properly modelled and incorporated in the different processes of software development as well as in the CASE tools that support these processes, that would mean a great advantage for any software development.

Requirements stage is considered a good application domain for AI techniques because of requirements nature. Software requirements express and establish the needs and constraints that contribute to the solution of a real world problem [11]. However, requirements tend to be imprecise, incomplete and ambiguous. In SE it is well known that this area is quite different from others because requirements reside in the problem space, whereas other software artifacts reside in the solution space [5]. Statistical studies and all the Chaos Reports [8,4], published since 1994, point out that tasks related to requirements are the main cause of disaster of software products. When requirement-related tasks are poorly defined or executed, the software product is typically unsatisfactory [22,3,4]. The role played by requirements is essential, as they are the basis for the analysis, design and implementation of the final product. Therefore, any improvement in the requirements stage will favorably affect the whole software life cycle.

Bayesian Networks [17,9,10], have been successfully applied with the purpose of enhancing specific activities related to SE knowledge areas. They have been applied in maintenance [14], defect and effort prediction [7,18,19] or implementation of a software project [12]. In addition, Bayesian networks have been successfully applied in Requirement Engineering (RE) [2], specifically in the prediction of the need for a review of the requirements' document [20].

Other AI techniques that have also been used to enhance the requirement stage are metaheuristic optimization techniques. Specifically, they have been used in the selection of the set of requirements that will be included in the development of a final software product [21].

Therefore, we need a seamless integration of RE and AI techniques to exploit the benefits of collaboration between these two knowledge areas.

By other hand, the biggest breakthrough in requirement management is when you stop thinking of documents and start thinking about information. Moreover, to be able to handle this information you have to resort to databases, particularly documental databases that have evolved into what nowadays are called CARE (Computer-Aided Engineering Requirement) tools. Among this type of tools the best known are the IRqA, Telelogic DOORS, Borland Caliber, and the IBM-Rational Requisite Pro. InSCo Requisite is an academic web CARE tool, developed by DKSE group at the University of Almería, which aids during the requirement development stage [16].

This work presents the architecture for the seamless integration of a CARE tool to manage requirements (i.e. InSCo Requisite) with some AI techniques (i.e. Bayesian networks and metaheuristics). Specifically, a Bayesian network, called Requisites [20], is used in the requirement validation task in order to validate the Software Requirements Specification (SRS) of a software development project. Metaheuristic techniques (Simulated Annealing, Genetic Algorithms and Ant Colony Systems) are used in the problem of selecting the subset of requirements among a whole set of candidate requirements proposed by stakeholders, that will be included in the development of a final software product [21].

The rest of this paper is structured as follows. Section 2 depicts the requirement engineering workflow. The architecture for the use of synergies between Knowledge Engineering and Requirements Engineering is explained in Section 3. Finally, the conclusions and future works are exposed in Section 4.

## 2 Enhancement of Requirement Engineering Workflow

The workflow depicted in Figure 1 shows an organization of the tasks that must be done in a software development project during Requirement Engineering stage. This workflow unifies the main classics methodological approaches [22,1,13], starts with a feasibility study that is built in order to determine the project scope and the availability of resources. Once the feasibility report is available, elicitation and analysis, specification and validation tasks are executed in an iterative way.

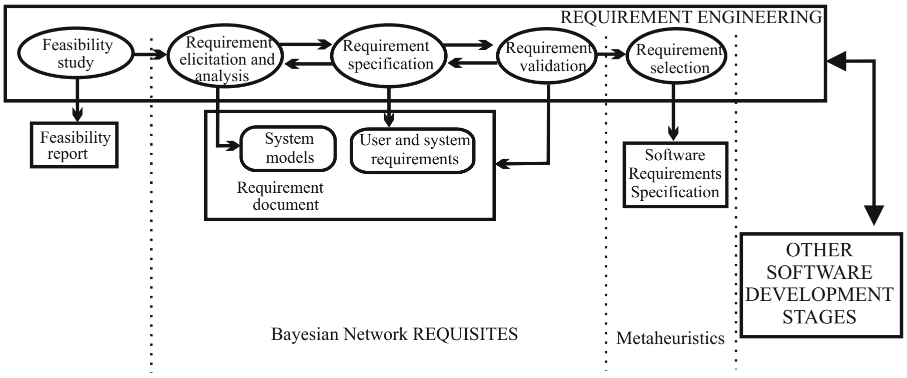
REQUIREMENT ENGINEERING
Feasibility study
Requirement elicitation and analysis
Requirement specification
Requirement validation
Requirement selection
Feasibility report
System models
User and system requirements
Requirement document
Software Requirements Specification
OTHER SOFTWARE DEVELOPMENT STAGES
Bayesian Network REQUISITES
Metaheuristics

**Fig. 1.** Requirement Engineering workflow

Requirements are elicited or gathered from users through interviews and other techniques such as questionnaires or brainstorming. Usually this is a complex task because activities requiring human communication imply problems of understanding, and requirements have to be conceived without ambiguities in order to define what the system is expected to do.

In the next task, requirement specification, captured requirements are gathered in a document or its electronic equivalent, known as Software Requirements Specification (SRS). Early approaches to address this activity using computers involved the use of word processors. This way of supporting requirement specification can be tedious and prone to error for the management and maintenance of large sets of requirements. CARE tools appeared to give a solution to this problem, providing environments that make use of databases, allowing an effective management of the requirements of any software project.

Requirements validation is a task performed in order to check whether the elicited and specified requirements present inconsistencies, the information is incomplete or there are ambiguities in the system definition. The Bayesian network Requisites has been built, through interaction with experts and using several information sources, such as standards and reports, to support validation and specification partially. The aim of this network is to provide developers an aid, under the form of a probabilistic advice, helping them at the time of making a decision about the stability of the current requirements specification. Requisites provides an estimation of the degree of revision for a given requirements specification. Thus, it helps the process of identify if requirements specification is stable and does not need further revision.

The elicitation-analysis-specification-validation cycle is carried out through several iterations in order to correct the defects found, and decide if the requirements specification has been completed in order to move towards the next task.

Finally, requirements selection task has as main objective to choose, from all the requirements defined in the specification, the subset of requirements that will be implemented. This selection is necessary due to the limitations of resources that usually appear in the feasibility report, and prevents development of all defined requirements. Requirements selection is a complex problem where many factors are involved (requirements priorities, development costs, customers' priorities, etc). The goal is to select a subset of requirements by searching for a set of requirements, which maximize satisfaction and minimize development effort considering the project constraints. This problem has been addressed in the literature using techniques from Artificial Intelligence, specifically metaheuristic algorithms [21] which have shown a performance similar to that exhibited by experts at the time of selecting the set of requirements to be developed in the software product.

## 3 Seamless Synergic Architecture

AI techniques described in this paper have demonstrated to obtain interesting results through different tests data [20,21]. However, it is difficult to put them in practice in real software projects. We strongly believe that having these AI techniques available in a CARE tool would be considerably helpful for any development team, making them more accessible even for non-expert people.

InSCo Requisite [16] is a web-based tool developed by DKSE research group at the University of Almería, to manage requirements of software development projects. It provides basic functionality allowing groups of stakeholders to work in collaboration. The fact of having the possibility of make changes to the tool, give us an exceptional opportunity to afford the integration of AI techniques in a CARE tool. This integration cannot be done in a straightforward way, because AI techniques and the CARE tools have been developed independently of each other. Therefore, it is necessary to define a communication interface between them preserving the independent evolution of both areas and achieving a synergic benefic effect between them.

This seamless synergic architecture is shown in Figure 2. The architectural pattern distinguish between three logically separated processes: the presentation (i.e. interface layer), the application processing (i.e. service layer), and the data management (i.e. data layer). This pattern (see upper picture in Fig. 2) provides a framework to create a seamless synergy, between knowledge-based tools and computer aided software engineering tools, at service layer becoming a more flexible management of interface and data.

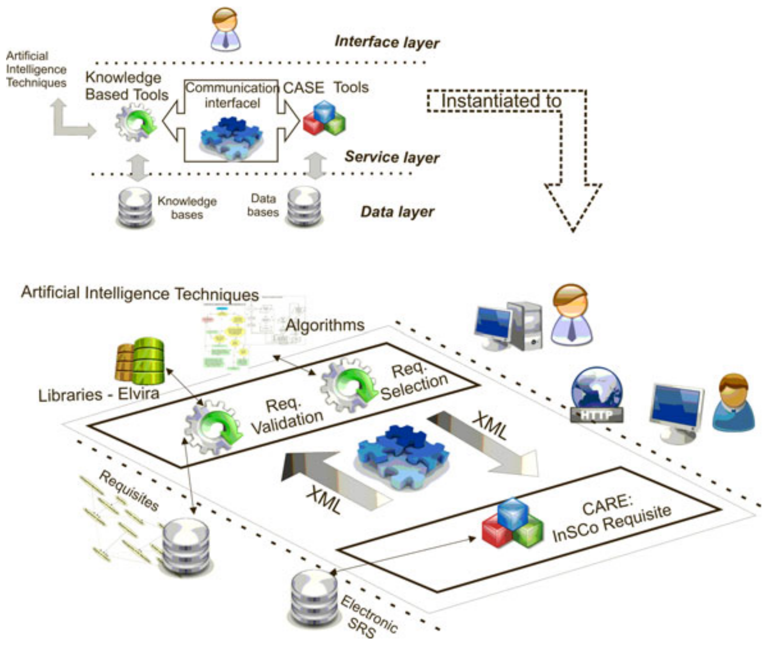

**Fig. 2.** Seamless synergic architecture

The architectural pattern can be instantiated for the enhanced requirement engineering workflow (see the bottom picture in Fig. 2). Interface layer represents the part of the architecture that provides users a mechanism to interact with the system. In our case, the interface is a web environment accessed from a web browser. Data layer is in charge of storing and managing the electronic representation of SRS handled by InSCo Requisite tool and the knowledge base that contains the Bayesian network Requisites. Service layer contains all the functionality provided by the overall system. The layer is composed by the CARE tool (i.e. InSCo Requisite), the AI techniques used to address requirements validation (i.e. Bayesian network Requisites) and requirements selection (i.e. metaheuristic algorithms) tasks, and a communication interface used to send and retrieve information between them.

The CARE tool is in charge of the management of all the information related to the development project (requirements, customers, etc) which is stored in a database. The knowledge-based tools carry out requirements validation and requirements selection tasks. Communication interface connect CARE and knowledge-based tools passing the required information needed for the execution of the appropriated processes. Thus, requirement validation receives metrics on the SRS and returns an estimation of the degree of revision for SRS; requirement selection receives resources effort bound and specific measures on individual requirements and identifies the set of requirements selected for implementations. All of these communication processes are performed through XML files. Next subsections explain the AI techniques applied in requirement validation and requirement selection.

## 3.1 Requirements Validation Module

Bayesian networks [17,9] are a well-known Artificial Intelligence technique suitable in handling decision-making problems involving uncertainty. They have the advantage of having rich semantics and can be interpreted by the stakeholders without a high background on Statistics. From user's point of view, Bayesian networks provide a natural framework for relevance analysis and prediction tasks. Therefore, a Bayesian network can be used in requirement validation as a predictor that determines whether a requirements specification has enough quality to be considered as a baseline of a software project, establishing a contractual agreement between customers and developers about what is needed to be developed [20]. Bayesian network Requisites (see Figure 3) has been designed though interactions with experts and using several information sources, such as standards and reports, to tell us whether we can stop the iterations needed in order to define the SRS. To assess the goodness of a SRS, Requisites uses the following variables:

– *Stakeholders' expertise*: Degree of familiarity with expertise respect to tasks related to RE. Previous experience would lead to commit fewer errors.
– *Domain expertise*: Level of knowledge reached by the development team about the project domain. If developers and other stakeholders handle the same terminology, communication will be more effective.
– *Reused requirements*: If the number of requirements from reusable libraries is high, the overall specification of the requirements may not need new iterations.
– *Unexpected dependencies*: Unexpected dependencies between requirements usually involve a new revision of the specification of the requirements.
– *Specificity*: Number of requirements sharing the same meaning for all stakeholders. A higher specificity implies less revision and a shorter process of negotiation in order to reach a commitment.
– *Unclear cost/benefit*: Requirements included by stakeholders or developers whose benefits cannot be clearly quantified.
– *Degree of commitment*: Number of requirements that required a negotiation in order to be accepted.
– *Homogeneity of the description*: A good requirement specification must be described using the same level of detail. If there is no homogeneity, the specification will need to be revised.
– *Requirement completeness*: Indicates if all significant requirements are elicited and/or specified.
– *Requirement variability*: Represent that requirements have suffered changes. When a requirement specification changes, it needs an additional revision.
– *Degree of revision*: Value predicted by Requisites Bayesian network.

The structure of Requisites models the dependence relationships between variables. Note that each node in the network has attached its own conditional probability distribution given its parents. In the qualitative structure of Requisites, specificity, unexpected dependencies, reused requisites, stakeholder's expertise and domain expertise are not affected by any other variables. Degree of commitment and unclear cost benefit are related because if the degree of commitment (i.e. the number of requirements that have to be agreed) increases, then the level of specificity will be low. If stakeholders have little experience in the processes of RE, then it is more likely to lead to requirements which are unclear in terms of cost/benefits.

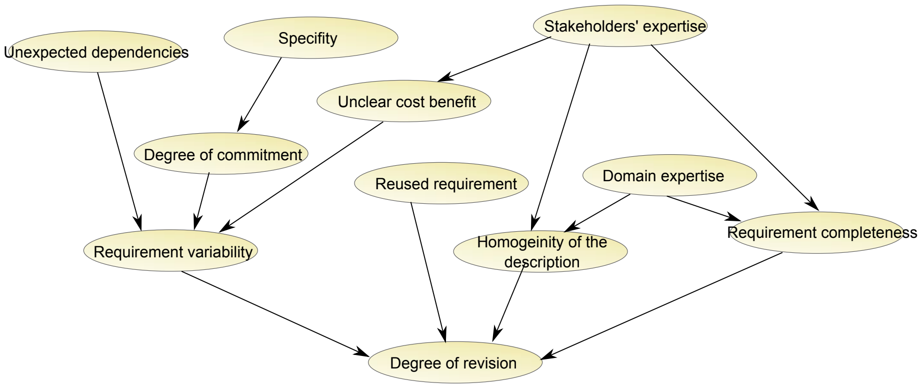

**Fig. 3.** Requisites Bayesian network [20]

The requirement completeness and homogeneity of the description are influenced by the experience of software and requirement engineers in the domain of the project and by stakeholders in the processes or tasks of RE. If experience is high, the specification will be complete and homogeneous because developers have been able to describe the requirements with the same level of detail and have discovered all requirements.

Requirement variability represents the number of changing requirements. A change in the requirements will be more likely to occur: if unexpected dependencies are discovered, if there are requirements that do not add any value to the end software, if there are missing requirements or if requirements have to be negotiated.

Bayesian network Requisites makes its prediction in order to indicate whether a requirement specification is sufficiently accurate or require further revision, performing an inference process in which some evidences or observations are used to calculate the marginal probability distribution of the variable degree of revision. The evidences (i.e. variables observed) are provided by the CARE tool that is in charge of extracting values of variables from the data about projects, requirements, users's activity and so on. Evidence is transferred from the CARE tool to Requisites through the communication interface as a XML file. The Bayesian network Requisites is implemented using Elvira [6], a Java software package for the creation and evaluation of Bayesian networks. Then, an inference process is launched on Requisites computing the posterior marginal probability distribution of the unobserved variables given evidence. The results are sent back to the CARE tool via the communication interface as other XML file.

The main advantage of applying this architecture, besides the synergy between AI and RE, resides in the fact that the Bayesian network model can be modified without affecting the CARE tool and vice versa, providing a great flexibility.

### 3.2 Requirements Selection Module

The use of meta-heuristics techniques can help experts who must decide which is the set of requirements that has to be considered in the next development stages when they face contradictory goals. The main aim is to combine computational intelligence and the knowledge experience of the human experts with the idea of obtaining a better requirements selection than that produced by developer's judgment alone.

The selection of a set of requirements between all those previously defined and validated can be addressed using metaheuristic optimization techniques. Specifically, simulated annealing, genetic algorithms and an ant colony system [21] have been adapted to solve this problem. In order to work these metaheuristics algorithms for requirement selection need: a representation of the problem, which is amenable to symbolic manipulation, a fitness function based on this representation and a set of manipulation operators. InSCo Requisite generates an interface file in XML format containing all data needed for the execution of each of the metaheuristic algorithms. This file is transferred through the communication interface. Note that input adopts the same format for each of the algorithms, so it would be a simple task to add new algorithms. Each algorithm searches for a subset of requirements which maximize the customers satisfaction and minimize the required implementation effort within the given project constraints (see [21] for details). After its execution, the set of selected requirements obtained is sent as an XML file through the communication interface and is presented in the interface of InSCo Requisite. In this way, developers receive a feedback when perform the task of requirement selection.

## 4 Conclusions

The purpose of this work is to define a three-layer architecture with two objectives. On the one hand, allow the seamless collaboration between Requirement Engineering tasks and some Artificial Intelligence techniques in order to perform a software development project. And on the other, facilitate their parallel and independent evolution.

During a software development project, the tasks belonging to the requirements stage workflow are inherently difficult and uncertain, and are considered a good domain for the application of AI techniques. In this work, we have structured the requirement workflow into several tasks, paying special attention to those tasks that can be enhanced or supported by knowledge-based techniques: requirement validation and requirement selection.

The generic seamless architecture has been instantiated taking advantage of the synergy between requirement management tools (InSCo Requisite), Bayesian networks (Requisites) and metaheuristic algorithms (Simulated Annealing, Genetic Algorithms and Ant Colony Systems). In this architecture, the communication interface is responsible for making the connection between CARE and knowledge-based tools, and has to pass the information required and needed for the execution of the appropriated processes for requirement validation and requirement selection tasks.

In the next future, we plan to instantiate our seamless synergic architecture to other Software Engineering stages (e.g. software maintenance or project management) by adding other knowledge models already developed in order to enhance these development stages. Also, we plan to enhance and automate the definition of the communication interface by defining languages that support it.

**Acknowledgments.** This work was supported by the Spanish Ministry of Science and Innovation under project TIN2010-20900-C04-02 and by the Junta of Andalucía under project TEP-06174.

## References

1. Abran, A., Moore, J., Bourque, P., Dupuis, R., Tripp, L.: Guide to the Software Engineering Body of Knowledge 2004 Version. IEEE Computer Society, Los Alamitos (2004)
2. Barry, P.S., Laskey, K.B.: An Application of Uncertain Reasoning to Requirements Engineering. In: 15th Conference on Uncertainty in Artificial Intelligence, pp. 41–48. Morgan Kaufmann, Stockholm (1999)
3. Standish Group: Chaos Report. Technical report, Standish Group International (1994)
4. Johnson, J.: CHAOS chronicles v3.0. Technical report, Standish Group International (2003)
5. Cheng, B.H., Atlee, J.M.: Research directions in requirements engineering. In: Future of Software Engineering, FOSE 2007, pp. 285–303. Institute of Electrical and Electronics Engineers, Minneapolis (2007)
6. Elvira Consortium: Elvira: An environment for probabilistic graphical models. In: First International Workshop on Probabilistic Graphical Models (PGM 2002), Cuenca, España, pp. 222–230 (2002), `http://leo.ugr.es/elvira/`
7. Fenton, N., Neil, M., Marsh, W., Hearty, P., Marquez, D., Krause, P., Mishra, R.: Predicting software defects in varying development lifecycles using Bayesian nets. Information and Software Technology 49(1), 32–43 (2007)
8. Glass, A.R.L.: Facts and Fallacies of Software Engineering. Pearson Education, Inc., Boston (2002)
9. Jensen, F.V.: Bayesian Networks and decision graphs. Springer, New York (2001)
10. Jensen, F.V., Nielsen, T.: Bayesian networks and decision graphs. Springer, New York (2007)

11. Kotonya, G., Sommerville, I.: Requirements Engineering: Processes and Techniques. Wiley (1998)
12. Lauria, E.J., Duchessi, P.J.: A Bayesian Belief Network for IT implementation decision support. Decision Support Systems 42(3), 1573–1588 (2006)
13. Loucopoulos, P., Karakostas, V.: System Requirements Engineering. McGraw-Hill, Inc., New York (1995)
14. de Melo, A.C., Sanchez, A.J.: Software maintenance project delays prediction using Bayesian Networks. Expert Systems with Applications 34(2), 908–919 (2008)
15. Meziane, F., Vadera, S. (eds.): Artificial intelligence applications for improved software engineering development: new prospects. IGI Global, Hershey (2010)
16. Orellana, F.J., Cañadas, J., del Águila, I.M., Túnez, S.: INSCO requisite - a Web-Based RM-Tool to support hybrid software development. In: International Conference of Enterprise Information System ICEIS, Barcelona, Spain, vol. (3-1), pp. 326–329 (2008)
17. Pearl, J.: Probabilistic reasoning in intelligent systems: networks of plausible inference. Morgan Kaufman, San Mateo (1988)
18. Pendharkar, P., Pendharkar, P., Subramanian, G., Rodger, J.: A probabilistic model for predicting software development effort. IEEE Transactions on Software Engineering 31(7), 615–624 (2005)
19. Radlinski, L., Fenton, N., Neil, M.: Improved Decision-Making for Software Managers Using Bayesian Networks. In: 11th IASTED Int. Conf. Software Engineering and Applications (SEA), pp. 13–19. Acta Press, Cambridge (2007)
20. del Sagrado, J., del Águila, I.M.: A Bayesian Network for Predicting the Need for a Requirements Review. In: Meziane, F., Vadera, S. (eds.) Artificial Intelligence Applications for Improved Software Engineering Development: New Prospects, pp. 106–128. IGI Global, Hershey (2010)
21. del Sagrado, J., del Águila, I.M., Orellana, F.J.: Requirement selection: Knowledge based optimization techniques for solving the next release problem. In: 6th Workshop on Knowledge Engineering and Software Engineering (KESE 2010), pp. 40–51. CEUR-WS, Karlsruhe (2010)
22. Sommerville, I.: Software Engineering. Addison-Wesley Longman Publishing Co., Inc., Boston (2006)