COTSRE: a COmponenTs Selection method based on Requirements Engineering

Miguel A. Martinez and Ambrosio Toval
Computer and Systems Department. University of Murcia (Spain)
mmart@um.es;atoval@um.es

Abstract

Nowadays Components-Based Development and Requirements Engineering (RE) are very active and growing fields in Software Engineering (SE). In this paper we present the preliminary ideas for a proposal of a component selection method whose development is fast, agile and requirements driven. The presented method is called COTSRE and it is based on SIREN, which is a method of RE based on the standards of this discipline and the use of reuse requirements catalogs. The final aim of our proposal is to achieve an RE method that guides the selection, the development and the composition of a set of independent software components, and whose application is the simplest possible. In this paper we also present some simples examples related to the domain of electronic mail applications.

1. Introduction

Requirements Engineering is a growing Software Engineering area, which has demonstrated its capacity to improve the productivity and quality of the processes and software products [1]. RE offers techniques and methods to tackle the initial tasks in the development cycle of Information Systems (IS). RE [2] includes elicitation, analysis and negotiation, documentation and maintenance of the requirements established for IS.

Component-based Software Engineering (CBSE) allows us to reuse pieces of previously elaborated code (software component) so that the system requirements are satisfied. The benefits of the CBSE are mainly a reduction in the cost and the delivery times.

In both disciplines, i.e. RE and CBSE, a common problem is the need to be able to select both requirements and components with different purposes, and one of the critical processes of the CBSE is the components selection which will comprise of the final product and which must fulfil the functional requirements defined by the user [3].

In the development process in CBSE [4] (identification, evaluation, selection, integration and update of the components), the requirements are the most important part for any effective COTS (Commercial Off-The-Shelf) components acquisition. Requirements will give the criteria to evaluate and select the component candidates, and even provide acceptance criteria that the client will use to verify if the given product satisfies his needs [5].

A crucial aspect, and that will be the center of attention of our research, is the importance of requirements when "rapid" software developments are carried out, since these are the key to success in this type of developments.

The components selection method presented in this method is based on SIREN (SImple REuse of software requiremeNts), a general method of RE.

In Section 2 the SIREN method is shown briefly. In Section 3 the proposed component selection method is described. Section 4 shows a simple example to illustrate the practical application of the method. Finally, Section 5 gives the conclusions and further work.

2. SIREN: A Requirements Engineering Method

SIREN method [6] is a practical approach to select and specify the requirements of a software system based on SE standards and above all in reuse.

SIREN encompasses a spiral process model, requirements document templates and a reusable requirements repository. The requirements templates proposed are hierarchical and based on IEEE standards. The repository has requirements organized by catalogs in different application domains or profiles.

Catalogs and PRD (Product Requirements Document) in use can share a sub-document common hierarchy or use different structures. Initially, a hierarchy of empty specification templates is established. Those templates fill up with generic requirements (easier to reuse), proceeding in the general case from different domains of application or fixed profiles. For example, at the moment we have defined the following catalogs in SIREN: *Personal Data Protection (PDP)* [7]; *Security in Information Systems* [6]; *Teleoperated Systems* and *Electronic Medical Record*, which are already in development.

Catalog requirements, in addition to the text itself, contain associate meta-information (attributes with information on each requirement) that enriches the requirement. At the moment there are 20 attributes defined, among which the following are emphasized: source, security level, rationale, priority, fulfilment.

An important question to consider is how SIREN manages traceability, since generally in a PRD there appear many requirements related to each other. At the moment, in SIREN only traceability relations between requirements (inclusive, parent-child and exclusive) are defined.

2. COmponenTs Selection method based on Requirements Engineering (COTSRE)

In order to provide the development team with the components search and design that verify the requirements established in the requirements specification document, we initially propose an "agile" approach, in which the requirements of the catalog, will have an attribute that indicates the source where one software component (or several) can be found which, when incorporated into the system that is being developed, cause the requisite to be validated. The values introduced in this attribute will depend on where the components are (url, components repository, etc.).

When a requirement is incorporated into the final requirements specification document, it may happen that it contains references to components that validate it or it may not contain such references.

If it has not registered any component, the development team has only one option, to begin to develop the project considering that one of the developed components will have to validate that requirement.

On the other hand, it may also happen that the requirement does not correspond exactly with the necessities of the project and it is necessary to adapt it so that such thing happens. In this case the components that validated it possibly no longer do so, or do so partially and therefore it is also necessary to redesign such components.

For example, we suppose that we are in a situation where we find one or several software components that fulfil each one of the requirements (or at least the most critical) included in the PRD and extracted from the templates of the generic requirements PDP catalog. Each one of these requirements has an initial list of links to components, which does not imply that they are all valid for the current project, since we only can affirm that with these initial components associated to the requirement the functionality of this one, can be implemented.

In order to select the most suitable component, we must take into account the rest of related requirements, especially, those non-functional requirements that impose some type of restriction in the final application (programming language, operating system, memory, etc.).

In order to help in this task, we make a selection matrix (Table 1) for each requirement, which will contain a column for each one of the related requirements to it, and a row with each of the identified components. The criterion to decide which requirements to consider and which not, it is left to choice of the analyst, although we could apply more sophisticated techniques such as applied metric to the requirements attributes [8] or automatic generation of test cases for the product evaluation [9].

This selection matrix is equipped with greater functionality, to take into account the following considerations:

- A requirement can be fulfilled by a component to a certain degree. Depending on the requirement and the component features, it is possible to score the fulfillment between requirement and component. In this case, the entries of the selection matrix will have a score instead of a simple binary value. This score is calculated using the "Feature Analysis Technique" [10].
- Weights using some multi-criteria decision making techniques [11] can be assigned to a requirement, so we can know how a particular requirement affects the final score of a component.

To help us in the selection, we will apply the technique called "Weighted Scoring Method" (WSM): let $s_1, s_2, \ldots, s_n$ be the scores for the requirements $r_1, r_2, \ldots, r_n$ obtained using "Feature Analysis Technique" for the component c_1; let $w_1, w_2, \ldots, w_n$ be the weights for each requirement obtained using, for example, the multi-criteria decision making technique called "Analytical Hierarchical Process" (AHP) [12]. Final score for c_1 would be: $score\ (c_1) = w_1s_1+w_2s_2+\ldots+w_ns_n$

The result of the selection process will be either a single component or a set of finalist components, which are in the same conditions with respect to the requirements included in the PRD. In this case, the final decision will depend on the analyst's or designer's preferences.

In order to make a quality selection, it is helpful to have for each component a precise specification of its features (datasheet). In COTSRE method we also take into account that other requirements in the final PRD, not only the technical requirements, can impose more limitations (in cost, available licenses, development team qualities) until even the selection procedure finishes in an empty list of components.

4. Illustrative Example

In this section we present a simple example of our first approach to the selection method proposed, using the electronic mail applications as domain.

Let us suppose that a developer team is beginning a project in which the application to develop must have the capabilities of sending a receiving electronic mails. Requirement A has the following textual description:

The application will use protocols SMTP, POP3 and MIME to send and receive electronic mails.

It is obvious that many components[1] can exist that give us capabilities of sending and receiving electronic mails, which is why Requirement A, will have in its "component" attribute a list with all those components that validate this requirement. In particular, in the commercial components repository we found 77 components that fulfilled this requirement.

This set will be the one that constitutes our initial set of candidate components, which we will be leaking progressively when considering the rest of the requirements.

We also want our electronic mail application, to allow the creation of digital signatures, thus fulfilling Requirement B:

The electronic mail application must provide the creation of digital signatures.

As well as fulfilling these functional requirements, the developer team also must fulfil several (non-functional) requirements related to the technology used, the programming language, operating system, etc. Thus, we have a Requirement C, with the following textual description:

The application will have to be implemented using the paradigm C++Builder or Microsoft Visual C++.

On the other hand, Requirement D is defined in the following way:

The application must be able to execute under the Windows XP Operating System.

[1] The components used in this example have been taken from http://www.componentsource.com

These requirements limit rather the components technology that we can use to cover the needs of sending and receiving electronic mails of our application. In this case the list of components that fulfil the required functionality in Requirement A is reduced to two candidates, as we can see in Table 1.

Component	Req. B	Req. C	Req. D
Easy Mail Objects v6.5		X	
Easy Mail .Net v3.0			X
PowerTCP Mail .NET v2.1.4		X	X
PowerTCP Mail Tool v2.9.2		X	
eMail Wizard Toolpack v3	X	X	X
Catalyst Internet Mail v4.5	X	X	X
Rebex Mail .NET v1.0.2537.0		X	X
Active Mail Professional v1.9			X

Table 1. Example of Selection Matrix for Req. A

Once we have made the first pre-selection of those candidates which have the mandatory features (simple features) that we wished our application to have, we must elaborate another decision matrix to determine how it is the most adequate selection, in respect to other desirable features (compound features) that we considered important. These characteristics, a value have assigned corresponding to the importance that these have in the final appearance of the application. This value will be fixed by the development team and the customer at the beginning of the selection process. The values that can be given are: *highly desirable* (3), *desirable* (2) and *optional* (1). In the example we have considered that the quality parameters "component cost" and "component usability" are highly desirable. On the other hand, the parameters "confidence in the vendor" and "component usability" are desirable.

As regards fulfilment, a scale of 3 possible values is established: *not fulfilled* (-1), *favourable fulfillment* (1) and *highly favourable fulfillment* (2). In Table 2 we see the decision matrix with the fulfilment degree for both preselected component.

Once we have made both matrices, we calculate the score for each one of the preselected components, applying the formula from Section 3. The calculations are shown in Table 3.

Fulfilment	eMail Wizard Toolpack v3.0	Catalyst Internet Mail v4.5
Component cost	1	2
Confidence in vendor	2	1
Component usability	2	2
Documentation quality	1	2

Table 2. Degree of fulfillment of the components.

Components	Score
eMail Wizard Toolpack v3.0	$1·3+2·2+2·3+1·2=15$
Catalyst Internet Mail v4.5	$2·3+1·2+2·3+2·2=18$

Table 3. Score of the components.

We thus obtain the component that we will use to add the functionality to send and to receive electronic mails to the application which we are developing, in this case Catalyst Internet Mail Control v4.5 is selected, since it has obtained a slightly superior score.

4. Conclusions and Further Work

The most important conclusions of this paper are:

1. There are several proposals in the Literature for components selection methods according to the requirements identified for the system, but selection and negotiation requirements techniques are not directly contemplated in any of the proposals found.
2. Most of the current processes and techniques of components selection make the selection, in an individual and independent way. Nevertheless, in real world applications, the decision to select a single component is not very habitual and usually depends on which other components are selected.
3. COTSRE method, based on a RE method such as SIREN, would allow us to solve the components selection problem based on requirements more efficiently, since from the first moment the existence of quality requirements documents is guaranteed, and in theory, is devoid of inconsistencies or conflicts between the requirements of the PRD.

There are several unresolved issues with respect to the use of a components selection method and when these methods give benefits for the customer and for the developer. Some of these open questions are:

1. If the COTS components are considered when developing the architecture of the system, then what are the impacts on the system?
2. What is the smallest granularity for a COTS component that provides a benefit (i.e., overall reduced effort) when used in a system?

We are researching to be able to give an answer to the first of the above questions exhaustively studying the different techniques of definition of architectures and construction of components, with focus on the problem of passing of the system requirements to the components that will fulfil them.

With respect to the second question, the context in which we move in this paper is medium size COTS components, since these are related to the domain of the electronic mail applications.

Acknowledgements

This work has been partially financed by the Spanish Ministry of Science and Technology, project DEDALO (Desarrollo de sistEmas De calidad bAsado en modeLos y requisitOs), TIN2006-15175-C05-03, and by the CALIPSO network (TIN2005-24055-E) supported by the Spanish Ministry of Education and Science.

References

[1] S Robertson, J. Robertson, Mastering the Requirements Process, ed. Addison-Wesley. 1999.
[2] G. Kotonya, I. Sommerville, Requirements Engineering. Processes and Techniques, ed. John Wiley & Sons. 1998.
[3] M. F. Bertoa, J.M. Troya, A. Vallecillo, “Measuring the usability of software components”, Journal of Systems and Software 79 (2006) 427-439.
[4] C. Alves, J. Castro, “CRE: A systematic method for COTS components selection”, in: Proc. of the XV Brazilian Symposium of Software Engineering, Brazil. (2001).
[5] Glass, R.: Software Engineering (6th edition). (2001).
[6] A. Toval, J. Nicolás, B. Moros, F. García, “Requirements Reuse for Improving Information Systems Security: A Practitioner's Approach”, *Requirements Engineering Journal* 6 (2002) 205-219.
[7] A. Toval, A. Olmos, M. Piattini, “Legal Requirements Reuse: A Critical Success Factor for Requirements Quality and Personal Data Protection”, in: Proc. of the IEEE Joint International Conference on RE (2002) 95-103.
[8] M. Piattini, A. Cechich, A. Vallecillo, eds.: *Component-Based Software Quality - Methods and Techniques*. Volume 2693 of LNCS, Springer (2003).
[9] N. A. Maiden, C. Ncube, “Acquiring COTS Software Selection Requirements”. *IEEE Software* 15 (1998) 46-56.
[10] B. A. Kitchenham, L. Jones, “Evaluating Software Engineering Methods and Tools; Part 5 The Influence of Human Factors”, ACM SIGSOFT Software Engineering Notes 22 (1997) 13-15.
[11] E. Triantaphyllou, Multi-Criteria Decision Making Methods. (2000).
[12] L. G. Vargas, T. L. Saaty, *Models, Methods, Concepts and Applications of the Analytic Hierarchy Process* (2001).