Requirements Specification Reuse

Silvana Castano and Valeria De Antonellis

Dipartimento di Elettronica e Informazione

Politecnico di Milano

P.za Leonardo Da Vinci,32 - 20133 Milano - ITALY

Tel: +39-2-23993651

Email: deantone@ipmel2.elet.polimi.it

Fax: +39-2-23993411

Abstract

An environment to support reuse at the requirement specification level of the application development lifecycle is presented.The reuse model is introduced, and a methodology for defining reusable conceptual components is illustrated. Finally, the architecture of the tools supporting the reuse process is described.

Keywords: Conceptual Schema, Classification, Design-for-reuse, Design-by-reuse, Requirements Reuse, Requirements Specifications.

1 Introduction

Reusability of previous artifacts is considered a fundamental aspect in the application development life cycle, since it can shorten the development process, from the early requirements engineering to the detailed design and implementation [4,5,20,26,31]. For what concerns reusability of software artifacts, problems related to the identification, classification, retrieval, adaptation, and composition of components have been widely studied, and solutions based on techniques and paradigms of the software engineering field have been proposed [6,18,23,24,25]. Our research focuses on reusability of requirement specifications during the Requirements Engineering activity of the development life cycle[10,11,16,17,22,30]. Experimentation and validation of our approach are being performed in the framework of the Esprit project n.6612 F^3 (From Fuzzy to Formal) aiming at defining a methodology and a software environment that integrates into a coherent whole a selection of adequate techniques, to manage and improve the requirements engineering process.

Two separate processes are required to enforce requirements reuse: design-for-reuse and design-by-reuse. *Design-by-reuse* is performed when a new conceptual schema is constructed not from scratch, but tailoring reusable components retrieved from a reuse repository. *Design-for-reuse* is necessary to populate the reuse repository with reusable components, properly defined to encapsulate the domain knowledge, exploitable for subsequently developing new applications not from scratch. The reuse-based development process requires a reuse engineer, who is responsible for the population of the reuse repository, and several application designers, which utilize the reuse repository (Fig.1).

The reuse engineering team defines reusable conceptual components in the form of generic components and associated guidelines, extracted from a collection of existing application conceptual schemas, and stores them in a reuse repository. Available components can then be retrieved and reused by application designers to specify new schemas not from scratch, but personalizing and tailoring retrieved components, according to the suggested guidelines.

The paper is organized as follows. In Sect.2, our reuse environment is described, based on the development lifecycle presented in Fig.1, while in Sect.3 the related research work is discussed.

Fig. 1. The reuse-based application development process

2 An environment for requirements reuse

Within the F^3 project, we propose models, methods, and tools to support both the design-for-reuse and the design-by-reuse, specifically oriented to the requirement specifications reuse problem [10,13].
In Sect.2.1 the reuse meta-model is described. In Sect.2.2 the reuse methodology is presented, and, finally, in Sect.2.3, the architecture of the tools supporting the proposed methodology is illustrated.

2.1 Reuse meta-model

In order to present in a systematic way the reuse model underlying both the design-for-reuse and the design-by-reuse processes, we introduce a reuse meta-model, describing the key concepts characterizing our approach.

The reuse meta-model has been defined using the Entity-Relationship model [3] to give an integrated an coherent description of all the needed inputs and results of the reuse environment. The reuse meta-model is shown in Fig.2. A reusable component is defined as a Generic Component, with an associated Guideline-Component. Reusable components are defined by analyzing existing conceptual schemas of the applications in a given domain. We consider a conceptual schema as a collection of (application) components, expressed in a suitable conceptual model, describing the entities of the real world, their relationships, properties, and, possibly, the associated processes [3,15,32]. Schemas produced by application designers are selected, analyzed,clusterized, and properly arranged inside the repository. Precisely, to each schema a set of schema descriptors is associated [14]. In the repository, schemas are grouped by similarity levels into clusters, on the basis of the shared schema descriptors. For each schema cluster, affinity sets are defined, grouping the most similar schema components, that are modified and engineered into reusable components. For instance,in the Extended Entity-Relationship model, we consider entities as reusable components,together with their attributes, their relationships, which show how the component can be inserted in a schema, and is-a relationships.

In general, we assume to work within a given application domain, e.g.office domain, air traffic control domain, banking domain, education domain. For a domain is essential to identify and define the terms frequently used in the domain (synonyms, homonyms, and so on), to determine both schema and component similarity. Thus, werefer to a Thesaurus as the base for all this information.

Fig.2 The reuse meta-model

2.2 Reuse methodology

In F^3 we are specificallyconcerned with the design-for-reuse process. To this purpose, a reuse methodology for the construction of reusable components and their associated guidelines has been defined. Starting from properly selected application schemas, common and similar conceptual components are identified and classified; from these components generic components and associated guidelines are extracted and stored in the reuse repository.

The phases of the F^3 reuse methodology are the following:

- selection of candidate schemas
- classification of candidate schemas
- selection and classification of candidate components
- design of reusable components
- assimilation of reusable components

Goals and characteristics of each phase are briefly described in the following.

During the first phase, the reuse engineer selects the schemas judged significant for both their design quality and prospect reuse. Criteria andmetrics to evaluate the reuse goodness of a schema in a given domain are under study.

The remaining methodological phases are based on the concept of similarity[30,22]. It is widely accepted that, in order to be reusable, a component should be apt to be used in more than one application, possibly with adaptations and modifications according to the specific application requirements. To minimize the required operations of adaptation and tailoring, and to provide powerful components, similarities in different application schemas should be identified, and commonalties abstracted into generic components. To facilitate this process, we identify firstly similar schemas, and secondly similar components within similar schemas. The classification model adopted for schemas and components is based on indexing criteria and clustering techniques, derived from the Information Retrieval area [29].

Schemas are indexed by means of schema descriptors, selected among the names associated to the schema components, for their capability of describing the schema subject [7,14]. Using the selected schema descriptors, the schemas are properly clusterized according to their level of similarity, giving origin to several schema clusters inside the repository, using information retrieval clustering techniques [28,29].

The schema clusters constitute the starting point for defining reusable components. Selected a cluster, semantic affinity between schema components is computed taking into accountthe structural and contextual properties of the components (e.g.,attributes, generalization hierarchies, relationships). Schema components presenting a semantic affinity greater than a predefined threshold are grouped into affinity sets, i.e.,hierarchical structures whose leaves are the examined components, and the intermediate nodes are numerical values expressing the level of affinity for the corresponding components.

Affinity sets are used by the reuse engineer to extract generic components, in that the properties and relationships common to all the affinity set members are identified and factored out into the corresponding generic component. For each generic component the corresponding guidelines are defined, including reuse suggestions to personalize the generic componentin specific contexts. Adaptations and modification suggestions are given in the form of design operations, applicable to the generic component (e.g.,refinement, specialization, aggregation operations) [15,9].

Finally, newly created reusable components must be assimilated within the reuse repository, to enrich the existing reusable components and to augment their description capabilities. To this purpose, we are studying view integration techniques to properly integrate the new components with the existing ones [2].

2.3 Reuse tools

A couple of tools characterize our reuse environment, precisely the EXTRACT tool, supporting the design-for-reuse process and the RECAST tool supporting the design-by-reuse process, interacting with the reuse repository, and with other tools in the F^3 requirement engineering support environment (Fig.3).

Fig.3 Global F^3architecture

The RECAST tool [1,17], developed first within the ITHACA ESPRIT project, and now under refinement to be integrated within the F^3 reuse environment, supports the design-by-reuse process. Such a tool provides a workspace where components are graphically represented and interconnected according to composition rules [12]. More precisely, it enforces the retrieval of reusable components from the reuse repository via browsing and querying functionalities, providing also guidelines to tailor and compose, at various refinement levels, the retrieved components, according to given strategies and application contexts.

The EXTRACT tool has been specifically designed and implemented to support the reuse methodology described in Sect.2.2. In the following, its functionalities are briefly described. Basically, EXTRACT is an interactive tool, such as the reuse tools proposed in [30,22], assisting the reuse engineer in defining new reusable components.

In order to assist the reuse engineer during selection of candidate schemas, EXTRACT will provide a functionality of retrieval and selection ofexisting schemas, according to suitable reuse metrics and selection criteria. Selection functionalities will be built on top of the browsing and retrieval functionalities provided by the reuse repository (see [13] for further details about the reuse repository implementation).

To properly arrange selected candidate schemas within the repository, EXTRACT provides a schema classification functionality, whereby the reuse repository is populated in a systematic way. EXTRACT classifies reuse candidate schemas according to the following steps:

- automatic selection of schema descriptors, using the procedure detailed in [10].The reuse engineer can adjust in an interactive way the threshold value for descriptor selection according tothe schema contents, and can force the inclusion of specific/meaningful descriptors, in order to improve the quality of the schema indexing;
- computation of the schema similarity for all the indexed schemas;
- definition of the similarity clusters, using the single link clustering technique.

Starting from the schema clusters, EXTRACT selects candidate components and classifies them according to their semantic affinity. The following steps are performed by EXTRACT:

- pairwise semantic affinity comparisons for conceptual components that are descriptors of schemas in a given cluster;
- automatic definition of affinity sets, using the complete link clustering technique [29] and the affinity coefficients. The reuse engineer can examine the affinity sets, and interactively modify the affinity threshold, to improve and/or refine their composition.

The reuse engineer can then invoke the definition of reusable components functionality of EXTRACT, to extract the reusable component for defined affinity sets. Precisely, EXTRACT performs the following operations:

- automatic definition of a Generic Component, by considering the structural properties, and the contextual properties (relationships, generalization hierarchies) common to all the affinity set components. The reuse engineer can interactively modify the number of affinity set components to be considered for the commonalities' extraction, in order to refine the definition of the generic component;
- interactive definition ofreuse guidelines for a Generic Component, in form of design operations to be applied to the generic component to adapt and tailor it to specific contexts.

3 Related work

Several research efforts are currently in progress in the requirements reuse area. They are mainly related to the design-by-reuse process, and most of the currently available systems provide some form of support for the application developer, in retrieving and tailoring selected reusable components, exploiting either the analogy paradigm, or the artificial intelligence techniques. In particular, in [16] a reuse analogy-based model is proposed, and the TRUE tool based on that model is described, supporting the reuse of transaction specifications. Starting from the target application requirements, a set of reusable specifications are retrieved,exploiting different retrieval strategies, based either on analogical reasoning, oron browsing of similar components. Retrieved components can then be used through restructuring, editing, and tailoring operations, to fit the new requirements. In [22] an analogy based approach is proposed for reusing requirements specification, and reusable specifications are given in form of abstract domain models, supporting reuse across several applications. A prototype tool, called Ira, is described, enforcing the design-by-reuse, by providing retrieval functionalities, and customization guidance to tailor the retrieved candidate specifications to the target domain. Abstract domain models proposed in [22] are similar to the *cliché* proposed in [27], where a tool to assist the acquisition of requirements is proposed, based on artificial intelligence techniques to support the knowledge acquisition process. Acquired requirements are matched against existing validated cliché,describing the commonly occurring structures in one or more domains. In [30] the concept of similarity is exploited to allow requirements reuse. Here an interactive environment is described, where a set of tools helpapplication designers in retrieving and reusing object-oriented requirements specifications similar to the application at hand, properly selected among the stored specifications, on the basis of their level of similarity. In [19] an approach is proposed, to develop new information systems requirements making use of generic structures properly instantiated to the current context, with more emphasis for the engineering process rather than the definition process of generic structures.

Our research addresses the design-for-reuse process, and we propose a constructive approach for defining reusable components, while this aspect of reuse is sometimes given as a *de facto*. We cover the problem of repository population, (generally left open or dealt with using examples), by building reusable generic specifications for both static (see [7,8,10]), and dynamic aspects (see [11,12,13]) of the application requirements, presenting generic features like those of the abstract domain models proposed in [22], and the cliché illustrated in [27].

Acknowledgements

Part of this work has been supported by the ESPRIT III Projectn.6612 F^3 (From Fuzzy to Formal) and by the Italian National Research Council “Progetto Finalizzato Informatica e Calcolo Parallelo”, L.R.C. Infokit.

References

[1] M.Ader, O. Nierstrasz, S. McMahon, G. Muller, A. K. Proefrock, "The ITHACA Technology: a Landscape for Object-Oriented Application Developement", *Proc. ESPRIT'90 Conf.* , Kluwer Academic Publisher, November 1990

[2] C. Batini, M. Lenzerini, S.B. Navathe, “A Comparative Analysis of Methodologies for Database Schema Integration”, *ACM Computing Surveys*, Vol.18, December 1986,pp.323-364

[3] C. Batini, S. Ceri, S.Navathe, *Conceptual Database Design*, Addison Wesley, 1991

[4] D.Batory, S.O'Malley, “The Design and Implementation of Hierarchical Software Systems with Reusable Components”, ACMTOSEM, Vol.1,N.4, October 1992, pp.355-398

[5] T.J. Biggerstaff, A.J. Perlins(eds.), “Software Reusability - Concepts and Models”, vol.I, ACM Press, Addison-Wesley, 1990

[6] G. Caldiera, V.R. Basili, "Identifying and qualifying reusable software components", IEEE Computer, Vol. 24, N. 2, Feb. 1991

[7] S.Castano, V. DeAntonellis, B. Zonta, “Classifying and reusing conceptual schemas”, in *Proc. of ER'92, Int.Conf. on the Entity-Relationship Approach, LNCS n.645*, Springer Verlag, Karlsruhe, October 1992

[8] S.Castano, V. DeAntonellis, “A Model for Reusable Requirements”, F3 Report, F^3.PdM.2-1-3-R1,November 1992

[9] S.Castano, V. DeAntonellis, “Reuse of Conceptual Requirement Specifications”, *in Proc. of RE ’93,ACM/IEEE Int. Conf. on Requirements Engineering*, San Diego, CA, January 1993

[10] S.Castano, V. DeAntonellis, “A Constructive Approach to Reuse of Conceptual Components”, in *Proc. of 2nd ACM/IEEE Int. Workshop on Software Reusability*, Lucca, Italy, March1993

[11] S. Castano, V. DeAntonellis, “Reusing Process Specifications”, in *Proc. IFIP Working Conference on Information System Development Process,IFIP WG 8.1*, Como, Italy, September 1993, North-Holland, pp.267-283

[12] S. Castano, V. De Antonellis, C.Francalanci, M.G. Fugini, B. Pernici, R.Bellinzona, L. Vandoni, “Methodology for reuse”, F^3 Report, F^3.PdM.2-1-3-R2, Politecnico di Milano, April 1993

[13] S. Castano, V. DeAntonellis, P.Sanpietro, “Reuse of Object-Oriented Requirements Specifications”, to appear in *Proc. ER '93, 12th Int. Conf.on The Entity Relationship Approach*, Dallas Arlinghton, TX, USA, December 1993

[14] V. DeAntonellis, B. Zonta,“A disciplined Approach to Office Analysis”, IEEE TSE,Vol. 16, No. 8, 1990, pp. 822-828

[15] V. DeAntonellis, S. Castano, B. Pernici, A. Piovesana, “The ITHACA Object Oriented Methodology: Application Developer Manual”, ITHACA Report, ITHACA.POLIMI-UDUNIV, 1992

[16] A.Finkelstein, “Reuse of Formatted Requirements Specifications”, Software Engineering Journal, September 1988,pp.186-197

[17] M.G. Fugini, O. Nierstrasz, B. Pernici, “Application Development through Reuse: the Ithaca Tools Environment”, SIGOIS Bulletin, vol. 13, n. 2,August 1992

[18] M.G. Fugini, S. Faustle, “Querying a Software Information Base for Component Reuse”, to apper in *Proc. of 2nd ACM/IEEE Int. Workshop on Software Reusability*, Lucca, Italy, March1993

[19] G.Grosz, “Buiding Information System Requirements Using Generic Structures”, Nature Report Series N.92-06, 1992

[20] C.W.Krueger, “Software Reuse”, ACM Computing Surveys, Vol.24, N.2, June 1992, pp.131-183

[21] Y.S. Maarek, D.M.Berry, G.E. Kaiser, “An Information Retrieval Approach For Automatically Constructing Software Libraries”, IEEE TSE, Vol.17,No.8, August 1991, pp.800-813

[22] N.A. Maiden, A.G. Sutcliffe,“Exploiting Reusable Specifications Through Analogy”,Communications of the ACM, Vol.35, N.4, April 1992, pp.55-64

[23] J.W. Hooper, R. O. Chester, *Software Reuse: Guidelines andMethods*, Plenum Press, Division of Plenum Publishing Corporation, NewYork, 1991

[24] E. Ostertag, J. Hendler, R. Prieto-Diaz, C. Braun, “Computing Similarity in a Reuse Library System: An AI-Based Approach”, ACM TOSEM, Vol.1, No.3, July 1992, pp.205-228

[25] R. Prieto-Diaz, “Implementing Faceted Classification for Software Reuse”, Comm. of the ACM, vol. 34,n. 5, May 1991

[26] Proceedings of REBOOT Workshop on Reuse, Esprit Project 5327,Grenoble, September 1991

[27] H.B. Reubenstein, R.C.Waters, “The Requirements Apprentice: Automated Assistance for Requirements Acquisition”, IEEE TSE, Vol.17, N.3, March 1991

[28] G. Salton, *Automatic Textt Processing - The Transformation, Analysis and Retrieval of Information by Computer* , Addison-Wesley, 1989

[29 G. Salton, Ch. Buckley, “Term Weighting Approachesin Automatic Text Retrieval”, Information Processing andManagement, vol. 24, Nr. 5, 1989, pp.513-523

[30 G.Spanoudakis, P.Constantopoulos, “Similarity for Analogical Software Reuse: A Conceptual Modelling Approach”, in*Proc. of CAiSE '93*, *Int. Conf. on Advanced Information Systems Engineering*, Paris, June 1993

[31] W. Tracz ed., *Software Reuse: Emerging Technology*, IEEE Computer Society Press, Washington D.C., 1988

[32] R. Wirfs-Brock, B. Wilkerson, L. Wiener, *Designing Object-Oriented Software*, Englewood Cliffs, NJ, Prentice Hall, 1990

4 Biography

Valeria De Antonellis is a professor of computer science at the Politecnico di Milano, Engineering Faculty. Her major areas of research include conceptual models, methodologies and tools for databases and information systems design. She is coordinator of the AICA WG on Databases. She is co-author of Relational Database Theory (Benjamin Cummings) and co-editor of Computer-Aided Database Design (North-Holland). She is currently responsible of the research team from Politecnico di Milano in the ESPRITProject n. 6612 - F^3 (From Fuzzy to Formal).

Silvana Castano is PhD student at the Politecnico di Milano,Engineering Faculty. Her major areas of research are related to models and methods for database and information system design. She is currently member of the research team from Politecnico di Milano in the ESPRIT Project n.6612 - F^3 (From Fuzzy to Formal).