Evaluation of Case-Based Maintenance Strategies in Software Design

Paulo Gomes, Francisco C. Pereira, Paulo Paiva, Nuno Seco, Paulo Carreiro, José Luís Ferreira, and Carlos Bento

CISUC - Centro de Informática e Sistemas da Universidade de Coimbra.
Departamento de Engenharia Informática, Polo II, Universidade de Coimbra.
3030 Coimbra
pgomes@dei.uc.pt,
http://rebuilder.dei.uc.pt

Abstract. CBR applications running in real domains can easily reach thousands of cases, which are stored in the case library. Retrieval times can increase greatly if the retrieval algorithm cannot cope with such an amount of cases. Redundancy can also be a problem, focusing retrieval alternatives in a very restricted search space. Basically, the system’s performance starts to degrade with the increase of the case-base size. Case-base maintenance allows CBR systems to deal with this problem, mainly through the use of case selection criteria. In this paper we present an experimental study about several case-base maintenance policies developed till now. We adapted and implemented these policies to a CBR system for software reuse and design, testing the applicability of these policies to cases with a complex representation (combination of tree and graph representations).

1 Introduction

The performance of Case-Based Reasoning (CBR, [1,2]) systems depends on several factors. From the similarity measures used for ranking cases, to the adaptation and verification mechanisms, and particularly the content of the case library - the cases used for reasoning. Case-base maintenance (CBM) is an important issue in a CBR system [4]. Several problems can arise if the case-base is not fine tuned, for instance, the system can take too much time to provide a solution to a problem, due to the high number of cases in the library, or the right cases are not retrieved due to a redundancy problem of the case-base, or more complex issues, like the diversity of a case-base.

CBM has been the focus of several CBR researchers for some time. Smyth and Keane [10] describe a case deletion policy for CBR systems which is based on the idea of preserving the competences of a case-base. They have showed that traditional deletion policies solve the performance problem (when retrieval time performance is poor, due to the high number of cases in the library), but they may fail to preserve the case-base competence. Racine and Yang [8] stress two important issues in CBM, the existence of redundant and inconsistent cases,
presenting methods to deal with these cases. One important notion in CBM is the competence of a case-base and was introduced by Smyth [11]. This notion is based on the observation that some cases are crucial to the competence of a case-base, while others redundant. Using the notion of case-base competence, Zhu and Yang [13] presented a case addition policy rather then deletion. Smyth and McKenna [12] presented a technique for constructing compact competent case-bases, this technique is based on the relative coverage of a case. Leake and Wilson [5] developed a new CBM policy based on the relative performance of a case. Recently McKenna and Smyth [8] have presented a study using several case-base editing techniques.

In our research work, we are developing a CASE tool capable of providing the software designer with several functionalities that we consider intelligent. Some of these capabilities are: retrieving similar designs, suggesting alternative designs, revising designs, learning design knowledge, and others. Our tool, called REBUILDER, basis it’s reasoning abilities on cases, which are the main knowledge source and a way of storing the design knowledge of a software development company. REBUILDER is intended to be used at a company level, centralizing the design knowledge and enabling it’s reuse. One important issue is the CBM strategy to follow, since the case-base performance is crucial to the system’s success. In this paper we explore several CBM policies having in mind REBUILDER as a target application.

The main CBM problem is: what cases should be in the case-base? But for this question to be answered, another question arises: what is the main purpose of the CBR system? and subsequently: what are the main properties that the case-base must have? In order to select an appropriate CBM strategy for REBUILDER, we have chosen some of the CBM strategies developed, and we have adapted and implemented them in our system. Most of these strategies are used with a case representation based on vectors of attribute/value pairs. Our approach uses a more complex case representation, where cases are UML (Unified Modelling Language, [9]) diagrams, represented by a tree-like structure, where each node can be a class diagram, which is basically a graph. We are interested on testing the CBM strategies implemented and to study the properties and performance of REBUILDER’s case-base.

The next section describes REBUILDER, detailing its architecture and case representation. Section 5 describes nine different CBM criteria that are implemented in REBUILDER, with the purpose of testing them. Then section 4 presents the performed experiments and the obtained results. Finally, section 5 analyzes the performance of the different CBM criteria, showing the advantages and disadvantages of each criteria. We also draw some conclusions about our case-base properties, and what are the desirable characteristics of REBUILDER case-base.