Towards an effective approach for Reverse Engineering

Vinicius C. Garcia, Daniel Lucrédio and Antonio F. do Prado
Federal University of São Carlos, Department of Computer Science
São Carlos, Brazil
(vinicius, lucredio, prado)@dc.ufscar.br
Alexandre Alvaro and Eduardo S. de Almeida
Federal University of Pernambuco, C.E.S.A.R.
Recife Center for Advanced Studies and Systems, Recife, Brazil
(aa2, esa2)@cin.ufpe.br

Abstract

Currently, the demand for the reverse engineering has been growing significantly. The need of different business sectors to adapt their systems to Web or to use other technologies is stimulating the research for methods, tools and infrastructures that support the evolution of existing applications. In this paper, we present the main research trends on reverse engineering, and discuss how should be an efficient reverse engineering approach, aiming at higher reuse levels.

1. Introduction

Current reengineering methods and approaches do not address the main problems faced by organizations when maintaining their systems. Although reengineering can decrease some aging symptoms of legacy systems [5], simply by reconstructing them does not guarantee that the new system will be more maintainable than before. For instance, the object-orientation, often considered the “solution for the software maintenance” [1], created new problems for the maintenance [4] and should be used with very care to assure that the maintenance will not be more problematic than in the traditional legacy systems. Current reengineering approaches leave these issues to the Software Engineer’s expertise and experience, when ideally they should be treated in a repeatable, controllable process. Next we present the main research trends on reengineering.

2. Research Trends

Reengineering is a way to achieve software reuse and to understand the concepts underlying the application domain. Its usage makes it easier to reason about and reuse information in analysis and design documents not always available in legacy systems. Reverse Engineering is the process of analyzing a subject system to identify their components and interrelationships, in order to create representations in another form or at a higher abstraction level. A great number of techniques and methods have been proposed to face the reconstruction software problem:

Source-to-Source Translation. The source program is first transliterated into the target language on a statement-by-statement basis. Various refinements are then applied in order to improve the quality of the output. Specially, it tends to be insufficiently sensitive to global features of the source program and too sensitive to irrelevant local details.

Object recovery and specification. The idea of applying object-oriented reverse engineering provides a simple way to create models of an existing system. The Object-Oriented paradigm offers some desirable characteristics, which significantly help in improving software reuse.

Incremental approaches. There are several benefits associated with iterative processes: by using “divide et impera” (“divide-conquer”) techniques, the problem is divided into smaller units, which are easier to manage; the outcomes and investment return are immediate and concrete and the risks associated to the process are reduced, among other benefits.

Component-Based approaches. The extraction of reusable software components from entire system is an attractive idea, since software objects and their relationships incorporate a large amount of experience from past development. It is necessary to reuse this experience in the production of new software.
3. Towards an effective approach for Reverse Engineering

The research on reverse engineering covers many different issues. Each research trend has its own focus, giving its own contribution to the area. An effective approach for Reverse Engineering should consider all of them, in a wide set of requirements, which are presented next:

Generic process. At some level of detail, reverse engineering projects will differ in terms of their processes. Different tasks are needed to achieve different types of changes. A process model created to guide a specific reverse engineering project must be based on characteristics of the system to be reengineered and the planned changes.

Define objectives first. A strategic planning, which establishes the main objectives of the reengineering process, must be defined before its execution. All modifications on a system should be based on strategic objectives, expressed in a strategic plan and later implemented.

Form a reverse engineering team. Within every project there must be an organizational structure [3] to assign project personnel to project roles. Each project member fills one or possibly several roles within a project. These roles include project manager, team leader, database management system administrator and tester, among others. The team will also include adjunct members, such as legacy system specialists and support staff, with knowledge on the technologies involved in the process.

Define metrics. Metrics constitute a baseline to measure the results of the reverse engineering project. Metrics should directly correspond to the strategic goals and objectives, that were defined earlier. These metrics should also include a means to measure the project status/progress during each phase.

Analyze all possible information. In order to capture the information that will be used in the reengineering, it is necessary to perform a full analysis of the system elements. There are a variety of elements to be collected, including: programs, library routines, business rules, user manuals, among others. These constitute the knowledge that must be reused.

Organize the legacy system. Since the reengineering product must be prepared for future maintenance and reuse, an intermediate step, which organizes the recovered information in order to achieve larger modularity, is required.

Reconstruct Documentation. The software engineer always walks into a fine line between documentation that is critical for maintenance, without creating excessive documentation that will be burdensome, ignored or rapidly obsolete. In order to achieve this, good documentation is still the best way to inform to the Software Engineer about the functionality contained in some asset. Hence, the reengineering process must obtain such documentation from legacy systems, where they are often of poor quality or inexistent.

"Divide et impera". To reconstruct the whole system in a single step brings many problems, such as delay on results and investment returns. Incremental approaches have proven to be the best way to reduce these problems. However, to correctly divide legacy systems is not easy. The interdependence between modules must be considered. The modules importance and relevance, which are specific to each project, may also influence in this decision.

Automated support. Automation tools offer valuable help in reengineering process, saving time, effort and costs. However, tools must be chosen with basis on the project’s strategic goals/objectives and available resources (funding, personnel, and time).

4. Concluding remarks

Current research on reengineering is divided in four trends (source-to-source, object-orientation, incremental and component-based). Future research should focus on ways to make the process of reverse engineering more easily repeated, defined, managed, and optimized [2].

A new tendency is that the research moves toward the separation of concerns area, behind AOSD, integrated with already existent reverse engineering and reengineering techniques. This will represent one step forward in the direction of the post object-orientation, towards higher maintainability and flexibility.

References


