

E-CARES Project: Reengineering of PLEX Systems

Christof Mosler

Department of Computer Science 3, RWTH Aachen University

Ahornstr. 55, D-52074 Aachen

`christof.mosler@rwth-aachen.de`

Abstract

One important field of application for embedded real-time systems is in the telecommunications industry. The first phase of the E-CARES reengineering project concerned the architecture modeling and the reverse engineering of telecommunication systems implemented in the programming language PLEX. Now, we study the restructuring of such systems including their re-design and re-implementation. This paper presents the current status of the E-CARES project and describes some interesting challenges of PLEX.

1 Introduction

There exist many approaches concerning reengineering of legacy systems, but the majority of these approaches deals with systems in the field of business applications. Our project concerns understanding and restructuring of complex legacy systems from the telecommunication domain. Such systems are embedded real-time systems using the signaling paradigm, thus they pose additional requirements regarding fault tolerance, reliability, availability, and response time. Corresponding reengineering tools should take into account these performance aspects and provide adapted visualization and modeling methods for their analyzes.

The E-CARES research project is a cooperation between Ericsson Eurolab Deutschland GmbH (EED) and Department of Computer Science 3, RWTH Aachen University. The acronym E-CARES stands for **E**ricsson **C**ommunication **A**Rchitecture for **E**mbded **S**ystems. The current system under study is Ericsson's AXE10, a mobile-service switching center (MSC) comprising more than ten million lines of code written in PLEX (**P**rogramming **L**anguage for **E**Xchanges) [4].

2 Approach

In the first phase of our reengineering project, the reverse engineering of telecommunication software was regarded [2]. In order to extract information about the systems and to regain the actual state of their architectures, various techniques were developed. Besides the source code, other sources of information were regarded and not only the static aspects but also behav-

ioral and dynamic characteristics were analyzed.

Current work concerns the restructuring of legacy telecommunication systems including their re-design and re-implementation. The aim is to provide concepts, languages, methods, and tools to improve the architecture of software, make software easier to understand, and help to find bugs. First, we generate and analyze the structure graph of the given software system, comprising its communication, control, and data flow. Then, we perform graph transformations to improve the system. Finally, we propagate all changes back to the source code level.

The reengineering prototype is based on the graph rewriting system PROGRES [3]. The PROGRES language is a strongly typed specification language for complex data structures. It is based on a graph model and allows the definition of graph schemes and corresponding graph transformations. The PROGRES programming environment consists basically of a syntax-directed editor and a C-code generator, by which the underlying application logic of the E-CARES prototype is generated. For building graph-based applications, we use a framework which automatically offers an adapted user interface to access the graph operations generated by PROGRES. Thus, not much additional functionality needs to be coded outside the PROGRES specification. Graph grammars offer a convenient and efficient way to define the required architecture transformations [1].

3 Challenges of PLEX

Our goal is to analyze the AXE10 system with respect to problem descriptions stated by Ericsson experts and to provide suggestions how to improve the software. In this section we discuss some aspects of PLEX posing interesting challenges for the restructuring process.

The programming language PLEX was developed in about 1970 at Ericsson and is still in wide use within the company. It is an asynchronous concurrent real-time language designed for programming of telecommunication systems. It has a signaling paradigm which means that only incoming signals can trigger code execution. The main unit of a PLEX program is a block and each block is stored in its own file.

PLEX provides only very simple means for program

structuring, therefore, engineers at Ericsson often use special implementation patterns to obtain a clear program structure. For example, it is not possible to define subsystems in the PLEX language. To group a set of blocks to a subsystem, they use predefined name conventions and comments. However, the concept of subsystems is still very important for the organisation and finds a wide application in the AXE10 context. For such complex legacy systems a clear and understandable architecture is essential. Each subsystem consists of blocks, which all should be related to one particular functionality of the system. But, subsystems often contain blocks which are assigned to them only because of historical reasons. Such situations make the software very difficult to understand and especially the maintenance of such systems is very painful. For that reason, we are studying reengineering techniques improving the software architecture in terms of decomposition and restructuring of subsystems. First algorithms based on the analysis of static signal flows have been successfully implemented.

Moving blocks from one subsystem to another does not cause any changes on the source code level. However, in order to obtain a more understandable software, also the restructuring of blocks can be useful. Here, some complex source code transformations can be required. For example, we can search through a block for subroutines and statement sequences which could be moved to other blocks. A similar, more general problem is the decomposition of blocks which have become too large for efficient maintenance.

As described in the introduction, the specific challenges of PLEX comprise primarily performance aspects. In this context, the restructuring of blocks plays an important role, too. For example, if we find code parts of a block which are mainly used by another block, we can reduce the number of exchanged signals and temporal variable allocations by moving these parts to the second block. Alternatively, we can merge blocks with related functionalities to form a new, more complex block and avoid inter-block signals. Corresponding algorithms should calculate the improved runtime performance before suggesting an architecture transformation.

Another promising approach to improve system performance is the analysis of communication buffers. Instead of transferring a large amount of data between a number of blocks, we can utilize a global memory buffer and send signals containing a pointer to such a communication buffer. We are developing algorithms supporting decisions on where communication buffers should be used instead of usual signal communication. For this analysis, Ericsson provides tables with capacity breakpoints depending on the data size and the number of blocks involved in the communication.

4 Outlook

The aim of the E-CARES project is to provide a flexible and interactive reengineering environment for analyzing PLEX systems with respect to problems stated by Ericsson experts and for providing suggestions how to improve the software. As described in this paper, current challenges concern the restructuring of subsystems and blocks, and the optimization of signal communication, in order to improve the software architecture and runtime performance. As already explained in the introduction, the characteristics of embedded real-time systems strongly impact the structuring and implementation of such reengineering tools. The graph rewriting system PROGRES offers a convenient and efficient way to define the required source code transformations. The E-CARES reengineering prototype already implements some restructuring algorithms and the provided results are discussed in regular meetings.

References

- [1] CREMER, KATJA, ANDRÉ MARBURGER and BERNHARD WESTFECHTEL: *Graph-based tools for re-engineering*. Journal of Software Maintenance and Evolution: Research and Practice, 14(4):257–292, August 2002.
- [2] MARBURGER, ANDRÉ and BERNHARD WESTFECHTEL: *Behavioral Analysis of Telecommunication Systems by Graph Transformations*. In PFALTZ, JOHN L., MANFRED NAGL and BORIS BÖHLEN (editors): *Proceedings of the 2nd Workshop on Applications of Graph Transformations with Industrial Relevance AGTIVE 2003*, LNCS 3062, pages 202–219, Charlottesville, Virginia, USA, September 28 – October 1 2003. Springer: Heidelberg, Germany.
- [3] SCHÜRR, ANDY, ANDREAS J. WINTER and ALBERT ZÜNDORF: *The PROGRES Approach: Language and Environment*. In EHRIG, HARTMUT, GREGOR ENGELS, HANS-JÖRG KREOWSKI and GRZEGORZ ROZENBERG (editors): *Handbook on Graph Grammars and Computing by Graph Transformation: Applications, Languages, and Tools*, volume 2, pages 487–550. World Scientific: Singapore, 1999.
- [4] WENNERSTEN, J.: *PLEX-C Language Description*. Ericsson Telecom AB, 1999. EN/LZB 101 1903 R4B.