E-CARES Research Project:
Utilizing Dynamic Information

André Marburger
Aachen University of Technology
Department of Computer Science III
Ahornstr. 55, 52074 Aachen, Germany
marand@i3.informatik.rwth-aachen.de

Dominikus Herzberg
Ericsson Eurolab Deutschland GmbH
CNM – Node Product Unit MSC
Ericsson Allee 1, 52134 Herzogenrath, Germany
Dominikus.Herzberg@eed.ericsson.se

Abstract

There are many reasons for reverse engineering or re-engineering legacy systems. So far, many approaches concerning re-engineering of legacy systems have been made. The majority of these approaches are dealing with systems in the field of business applications. In contrast to business applications embedded systems, e.g. telecommunication systems, have additional requirements regarding fault tolerance, reliability, availability, and response time. We found that these requirements have a significant impact on the software part of an embedded system. It has different characteristics concerning structuring, inter-program communication, etc. Therefore, in the E-CARES research project an approach is developed that explicitly includes usage of "dynamic" information in combination with multi-level abstraction/visualization to improve the understanding of telecommunication systems. This paper describes how dynamic information can help to understand telecommunication systems.

1. Introduction

The E-CARES[1] research cooperation between Ericsson Eurolab Deutschland GmbH (EED) and the Department of Computer Science III, RWTH Aachen, deals with understanding complex legacy telecommunication systems. The subject of study is Ericsson's Mobile-service Switching Center (MSC) called AXE10. The cooperation aims to develop methods, concepts, and tools to support the processes of understanding and restructuring this kind of embedded systems.

In contrast to business applications embedded systems in general, and telecommunication systems in particular, have additional requirements regarding fault tolerance, reliability, availability, and response time. While analyzing the static structure, the control flow, and parts of the data flow of our subject of study, we found that these requirements have a significant impact on the software part of this embedded system. It has different characteristics concerning structuring, inter-program communication, etc. Legacy business applications have a (more or less) static linking of different parts of the system. That is, at compile-time we already know which part of the system will call another one to realize certain functionality at runtime.

Ericsson's telecommunication system is different. It is a highly dynamic system. The AXE10 switching system handles thousands of calls at the same time. It has to adjust itself to different situations very fast, that is, it has to be very flexible. Therefore, the systems functionality is spread over a large number of small programs. Each program realizes a special part of the system's functionality. In addition, every program can be seen as a process template. That is, at runtime there can be several instances of a single program. Accordingly, a service, i.e. a simple voice call, is realized by connecting the program instances at runtime. In many cases, a single program, or one its instances respectively, is used to realize different services. This is indicated i.e. by a highly interconnected call graph. Therefore, it is not possible to uniquely predict which program/process within the system will call which other program/process if the user requests a certain functionality by just "reading the code" (static analysis). That is, to understand a telecommunication system and to derive a system's architecture, static information extracted from design documents etc. are not sufficient.

In the context of the E-CARES project, we have the possibility to get runtime information. This information on a system's runtime dynamics can be used in addition to the results from static analysis. It provides several capabilities to extend the information basis and to get new insights of

[1] The acronym E-CARES stands for **E**ricsson **C**ommunication **AR**chitecture for **E**mbedded **S**ystems.

the analyzed system. In this paper we briefly describe how this dynamic information can be utilized to improve system understanding.

2. Obtaining Dynamic Information

We derive dynamic by tracing the AXE10-emulator machine. This emulator works in a virtual time mode that allows complex tracings without getting into timing problems. Tracings of the emulator are initiated by defining which kind of information and which part of the system is of interest. Once the machine is started, different call scenarios can be stimulated using the testing equipment connected to the emulator. The tracing information is continously written to a text file (see figure 1). Currently, the following information is obtained using the emulator:

- signal (name, identification, type)
- sender of a signal (program)
- receiver of a signal (program)
- data transferred with a signal
- signal priority

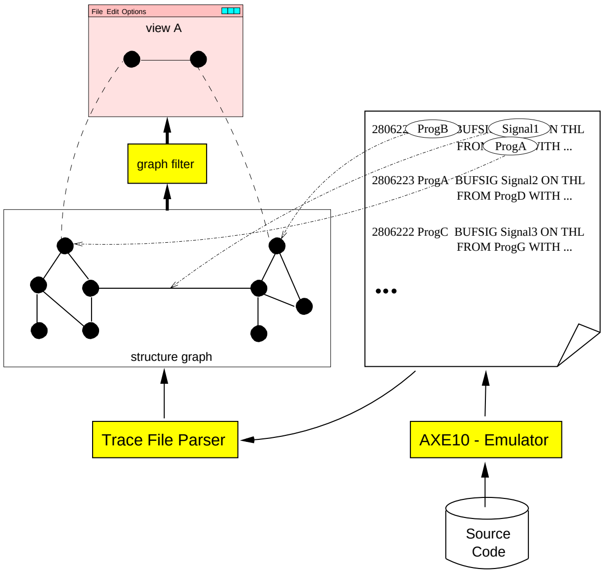

Figure 1. Obtaining "dynamic information"

Each trace file contains the runtime information obtained from the emulator for a special call scenario. The trace files are well-structured and easy to process automatically using some simple text processing functions or a small parser.

3. Utilizing Dynamic Information

The information on a system's runtime behavior can be utilized in different ways. First of all, tracing information is sufficient to "simulate" the dynamic behavior of an analyzed system. For this purpose, we use a small parser that is connected to our reverse engineering prototype. To simulate a call scenario within this prototype, the corresponding trace file is parsed stepwise. The information read from the file is visualized by means of a graph. Indeed, the information resulting from code analysis is stored as a attributed graph. Therefore, we only need to highlight the corresponding parts of the graph to achieve a "simulation" within one of the views provided by the reverse engineering prototype. The prototype comprises three different kinds of views, each visualizing the information stored in the information base (attributed graph) on a different level of detail. That is, simulation can take place on different levels of "abstraction" at the same time. For more details refer to [2]

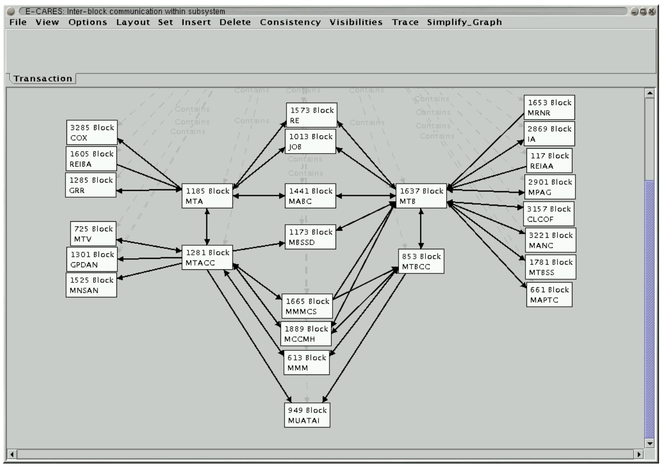

Figure 2. Mobile-to-Mobile call trace

After a simulation is finished, the displayed graphs represent a kind of simple multi-level collaboration diagram with respect to UML [3] (see figure 2). If the order of edges is preserved by labeling them appropriately the result would be a full collaboration diagram. In addition every trace file can easily be transformed into a sequence diagram. This way of defining or describing program inter-communication is very common within the telecommunications domain. These collaboration diagrams or sequence diagrams can be used for re-documentation purpose. According to some Ericsson experts this is very useful, because there is no equivalent up-to-date documentation of the system.

The simple collaboration diagrams resulting from the simulation of a call scenario are the basis for the identification of different roles with respect to the GSM[2] standard. This standard defines, more or less detailed, how a GSM-network switch should be organized, which features must be implemented, and how different interfaces should look like. For example, the whole system is organized in different layers. Each layer is responsible for a certain aspect of the realization of communication. The vertical communication of layer N and layer N-1 is realized via protocols[3] comprising a defined set of messages. Therefore, at each "end" of this communication link there has to be a special unit performing message handling. That is, the role of these units is message handling. By analyzing different call scenarios restricted to certain parts of a system, it is possible to identify these units.

The capabilities to "simulate" different call scenarios and to visualize the components of the system that are involved in a call-case is as well very useful with respect to on-the-job training of new employees. Considering a complex legacy telecommunication system like Ericsson's AXE10, it would take hours studying the system's source code and lots of pen-and-paper drawings to achieve similar results. To support on-the-job training, inter-expert discussion, and re-documentation, these collaboration diagrams or sequence diagrams can be stored as XML-files, including layout. They can be reloaded into the reverse engineering tool at any time. Including these diagrams into an online documentation could be another way of post-processing dynamic information.

With respect to maintenance an interesting topic is *feature location* [4][1]. Feature location is used to identify those parts of a system's source code that implement a certain functionality. This technique bases on analysis and comparison of different traces of a system. The results from feature location can guide a maintenance engineer to that parts of the source code, that he most probably will need to modify to fix a bug or to implement add-on features.

Coming back to sequence diagrams, there is another utilization of dynamic information. As mentioned before, sequence diagrams are often used within the telecommunications industry to define and document interaction between different units of a system. In some cases, these sequence diagrams are stored in a well defined and computer processable textual representation. Assuming computer processable "legacy" sequence diagrams, it is possible to automatically compare them to corresponding traces of the current version of the system. The result is the identification of deltas between the current version of the system and that one underlying the available documentation.

4. Conclusion

The E-CARES research project is concerned with the understanding and restructuring of complex legacy telecommunication systems. The research is mainly driven by the analysis of Ericsson's mobile services switching center, called AXE10. So far, we have concentrated on the understanding aspects. We found, that in the case of telecommunication systems, the information on the structure of the system is not sufficient to understand the system. Therefore, we extended our approach to include additional information on the runtime behavior and the runtime structure of these kind of embedded systems.

In this paper, we indicated different possibilities to utilize this additional information to support the understanding of telecommunication systems, ranging from simulation of runtime behavior to re-documentation of the existing system. Of course, this approach is neither limited to telecommunication systems nor to the aspects mentioned so far. In general, we can conclude that utilization of information on a systems runtime behavior can help to identify execution patterns that occur regularly. Execution patterns on the other hand can help to identify higher level concepts within a systems that are not extractable from static analysis.

References

[1] T. Eisenbarth, R. Koschke, and D. Simon. Derivation of Feature Component Maps by means of Concept Analysis. In *Proceedings of the 5th European Conference on Software Maintenance and Reengineering*, pages 176–179. IEEE Computer Society Press, 2001.

[2] A. Marburger and D. Herzberg. E-CARES Research Project: Understanding Complex Legacy Telecommunication Systems. In *Proceedings of the 5th European Conference on Software Maintenance and Reengineering*, pages 139–147. IEEE Computer Society Press, 2001.

[3] J. Rumbaugh, I. Jacobson, and G. Booch. *The Unified Modelling Language Reference Manual*. Addison Wesley, 1999.

[4] N. Wilde, H. Page, and V. Rajlich. A Case Study of Feature Location in Unstructured Fortran Code. In *Proceedings of the 5th European Conference on Software Maintenance and Reengineering*, pages 68–76. IEEE Computer Society Press, 2001.

[2] GSM is an acronym for **G**lobal **S**ystem for **M**obile communication

[3] Note, in this case the term protocol is not restricted to horizontal peer-to-peer communication.