AQuIS '98

PROCEEDINGS

4th International Conference on
ACHIEVING QUALITY IN SOFTWARE

Venice, Palazzo Giovannelli
March 30 - April 2 1998

EDITED BY
Carlo Ghezzi - Mario Fusani

ORGANIZED BY

Istituto di Elaborazione
dell'Informazione del CNR

QUALITAL
CONSORZIO UNIVERSITARIO
IN INGEGNERIA DELLA QUALITÀ

Towards Setting Up a Reliability Testing Process

*Antonia Bertolino**, Paolo Di Benedetto*, Gaetano Lombardi*, EmiliaPeciola**

* Ericsson Telecomunicazioni SpA, Rome, Italy
* * Istituto di Elaborazione della Informazione del CNR, Pisa, Italy

Abstract:
We report on an ongoing experience at Ericsson Telecomunicazioni SpA aimed at establishing a reliability guided testing process based on the state-of-practice SRET methodology. The motivation was to improve the current Ericsson Function Test process in terms of the reliability measured in a fixed period of operation after product delivery. We have thus started a case study to evaluate the hypothesis that operational testing can actually achieve higher reliability than conventional deterministic approaches, with no or limited additional costs. Through this case study we would thus like to investigate reasons why a mature and effective quality technology such as software reliability engineering is now claimed to be, has not yet been taken up on a large scale within the industrial world. In this paper we describe the plan of the experiment and report about some preliminary insights collected during the set up of the case study.

Keywords: Operational profile, software reliability.

1. Introduction

Among the approaches that have been proposed in the last several decades for achieving and controlling quality in software, software reliability testing is certainly the most mature and the best understood from the research side. Several reliability growth models and, most importantly for the practitioner, tools for using them are readily available [2]. Moreover, recent theoretical advances in the field [1] have largely eliminated some difficulties that arose in the choice and calibration of these models for a specific case. The opinion of the experts in the academic world is that it is now possible (they add "in most cases", some caution is due) to obtain reasonably accurate reliability measures in industrial contexts requiring relatively modest levels of reliability [1]. Yet it is a fact that reliability testing remains poorly practised in the industrial domain. Software companies for the most part either ignore reliability engineering technology, or remain sceptical. Are there reasons for this? Why is it that a technology that is mature according to the academics, and cost/effective according to those very few exceptions that adopted it, remains not practised? If it is true that reliability engineering is worth the investment, why has not it been spread quickly amongst the industrial domain, as it happened with other technologies (e.g., OO)? Is it just a question of time, or are there practical limitations that hamper its deployment on a large scale? Which are the practical issues involved in adopting it? With these questions in mind, we have recently launched an experimental case study to evaluate the costs and the improvements gained in setting up a reliability testing process within the industrial context of Ericsson Telecomunicazioni S.p.A. in Rome (TEI in the following).

In this paper we describe the case study, which is still going on. We discuss the plan of the experiment and some preliminary insights. At the time of writing, we have not yet collected the results forecast in the experiment plan; thus we will not fully address the above questions. However, another goal of this paper is that of exposing the above questions to AQuIS audience and to get the opinion about and hopefully the experience with reliability engineering of other software developers.

In the next section we describe TEI motivations and objectives in adopting reliability testing techniques. Then in Section 3 we briefly describe the SRET methodology, which we chose for the case study. In Section 4 we give a detailed description of the pilot project and the experimental plan. In Section 5 we discuss current status and finally, in Section 6, some preliminary insights.

2. Motivations and Objectives

We report about an ongoing project at TEI aimed at experimenting operational statistical testing. The project consists of a case study involving the application of reliability testing techniques to a selected baseline project.

2.1 The Need for Improvement

TEI test strategy includes four different test phases:

- Basic Test, testing the smallest module (test object) in the system. The goal is to verify design specification;
- Integration Test, testing a functional area; all modules in that area are integrated;
- Function Test, verifying system functions;
- System Test, verifying system performance and architectural requirements.

The primary objective of this case study is to improve TEI Function Test process with respect to the reliability achieved at the end of the Function Test phase. This improvement is in alignment with the continuously running Ericsson System Software Initiative (ESSI), which comprehends and supports a variety of improvement programs. One of the ESSI objectives is reducing the fault density figures obtained by monitoring the first 6 months in operation of released products. Fault density is measured by the number of failures detected over the product size, expressed in lines of code. Within the frame of the ESSI initiative, regular assessments are being performed at all Ericsson Software Design Centers, in order to identify key areas for process improvement and to propose a framework for subsequent improvement actions. A software process assessment has been recently performed at TEI organisation. The assessment covered the AXE10 (multi-application, open-ended digital switching product for public telecommunications networks) software development area. The software processes at the Design Center were found to be at defined level of maturity (level 3) of Capability Maturity Model [5]. Although the result is very satisfying, TEI is going to initiate some of the level 4 practices. The organisation is going to improve its competence in the area of statistical process control and prediction methods. This case study is one of the improvement activities in that area.

Additionally, at TEI a Root Cause Analysis of failures found during the first 6 months in operation is performed. The goal is to monitor failures with respect to the phase in which they are originated. The results of this analysis for TEI products showed that an important percentage of failures (48%) corresponds to software faults that could have been discovered during Function Test. Both ESSI and Root Cause Analysis thus point to the Function Test phase as a good attack point for improving the quality and the reliability of TEI software products.

2.2 Expected Benefits

The hypothesis we intended to test in this case study is that the Function Test process can be made more effectiveby introducing explicit reliability objectives to guide it. To this purpose a new approach will be used for test case selection: the new approach is Musa's "software-reliability-engineeredtesting" (SRET) [1, 2]. We describe SRET methodology in section 3. Testing in the SRET approach is intended to reproduce the expected usage of the system in operation, so those functions which are the most critical for the user are tested more thoroughly. In this way we expect higher reliability levels for the tested product. SRET provides a systematic, step-wise procedure for the definition of an operational profile for the product under test. The developed operational profile is then used to guide test case selection.

Although they are not primary objectives for this case study other measurable benefits are expected as well:

- SRET will provide a means for predicting product reliability in operation (via analysis of test results under a reliability growth model).
- Increase of reliability in operation will reduce maintenance costs after product delivery.

3. Musa's SRET Approach

Software reliability is a measure of the probability that a software will execute without failure for a specified time period within a specified environment. Software Reliability Engineering (SRE) encompasses state-of-practice techniques and tools for analysing, managing and improving the reliability of software products. The central focus is on the quantitative evaluation of the operational behaviour, as contrasted with qualitative evaluation of conventional (non reliability based) validation and verification activities. Thus a key step of SRE practice is to define quantitatively the quality objectives of the development process. SRE activities embrace all phases in the software life cycle, from the feasibility and requirement stages up to maintenance after delivery.

Our interest in SRE is in the application of SRE activities to the testing stage. Musa has recently introduced the SRET methodology, whereby "SRET is testing guided by reliability objectives and expected usage and criticality of different operations in the field" [4]. Reportedly [2, Cap. 6], Musa's SRET approach has been successfully applied to many projects with documented strong benefit/cost ratio results. For this reason, we decided to adopt this approach for application to our process improvement experiment via reliability testing. For the sake of completeness we report a very brief summary of SRET features; we, however, recommend that the readers go to the original sources for reference[2, 3, 4].

The SRET approach consists of five principal activities:

1. Define the reliability objectives: this implies defining possible failures in the distinct modes of operation and identifying their potential impact on users (severity class); then the reliability level to be achieved for each operational mode and for each severity class must be stated quantitatively.
2. Develop the operational profile: Musa defines an operational profile as "the set of operations and their probability of occurrence". To obtain the list of operations, one can follow a five step procedure, starting from identifying a list of possibly differentcustomer types, and progressively breaking them down into different user types, then into the different modes in which a user can invoke the system. In turn, for each system mode a functional profile is defined, i.e., a list of the functions needed by the user in each mode and their occurrence probability. Finally the functional profile, that has been defined from the user's point of view, has to be converted to the operational profile, that is system oriented: an operation is associated with running the system to accomplish a defined task.
3. Prepare the tests: this involves specifying the test cases to be executed. In this phase the scripts for automatically launching the test cases are prepared.
4. Execute the tests: the execution must be automated with the use of a test management system. In this phase we must carefully record failures and their time of occurrence.
5. Interpret test results: failure data collected during test execution are interpreted differently depending on whether we attempt to resolve detected failures or not. In the first case, we are doing debug test, i.e., our goal is to increase the product reliability. We then have to select a suitable reliability growth model and monitor the system failure

intensity values as we find and remove faults. We are finished when we reach the originally set reliability objectives. In the second case, we are doing acceptance test, i.e., our goal is to decide if a product is acceptable or must be rejected, and we decide based on the product reliability. We then simply evaluate the product failure behaviour against the required reliability levels.

In our case study, we have set up a process precisely reflecting the above five activities. However, we emphasise that this is the first time a reliability guided testing is attempted within TEI and consequently in some occasions during the implementation of the SRET process (that is still ongoing) we found ourselves in the necessity to act as pioneers and to decide for 'acceptable' engineering compromises, where we missed part of the information or the background required. Undoubtedly, subsequent future applications of SRET within TEI would result facilitated now that we have opened up the route.

4. Description of the Case Study

4.1 Purpose

The case study is aimed at testing the conjecture that Function Test conducted according to the SRET approach leads to cost-effective reliability improvements over current TEI practice. The reason for expecting that effort will not increase significantly is that, although many more test cases are expected to be run, the process will be largely automated. Cost is measured by computing total testing effort.

As explained in section 2.2, other benefits are expected from adopting the new approach. The case study will also provide for evaluating these other side benefits. Specifically, unlike the current Function Test approach, SRET provides a means of predicting product reliability in operation. Within the experiment, we intend to select and adopt a suitable reliability growth model to analyse failure results during the debugging period. The predicted reliability will then be compared to that experienced in operation. Also, we intend to keep track of maintenance costs after product delivery, to check whether SRET actually reduces them, as it claims.

4.2 Generalities

4.2.1 The CTM Project

The project used as a base to evaluate the SRET approach is the 'CTM project'. The CTM project implements the service Cordless Terminal Mobility (CTM) in the Ericsson AXE architecture. Cordless Terminal Mobility is a service that allows users of cordless terminals to be mobile within and between networks. Where radio coverage is provided and the cordless terminal has appropriate access rights, the user will be able to make calls from, and to receive calls at, any location within the fixed public and/or private networks, and to move without interruption of a call in progress. The solution adopted by Ericsson is to connect the mobile terminal to the fixed network via a Central Control Fixed Part (CCFP), whose main aim is to concentrate the traffic towards the CTM Exchange in a more capacious link called a 'device'. All the devices between a CCFP and a CTM Exchange are grouped, for administrative reasons, in an entity called a 'route'. The Ericsson CTM architecture is shown in figure 1. The pilot project consists in testing the administration functionalities of the links between CCFP and CTM Exchange.

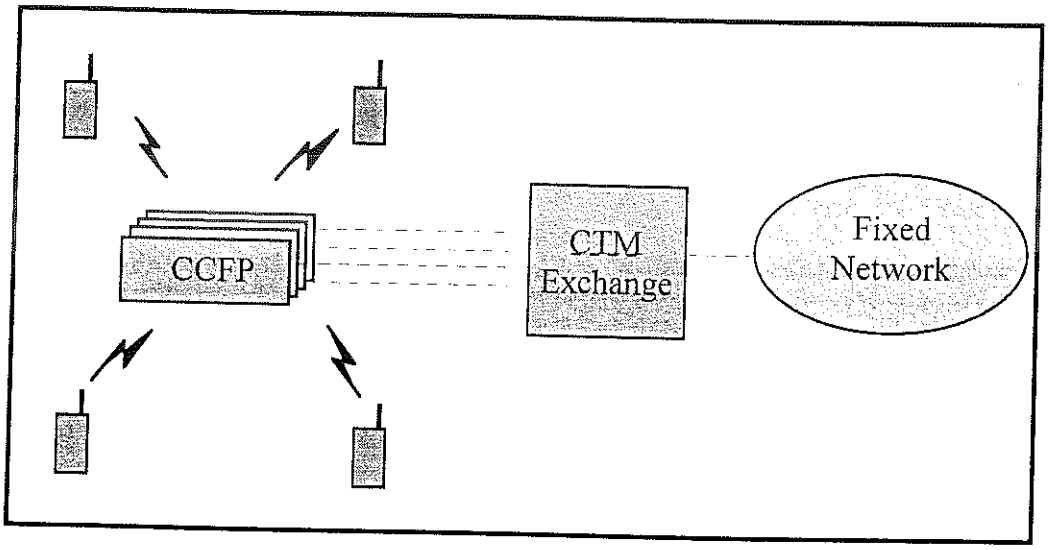

Figure 1. Ericsson CTM architecture

4.2.2 TEI Function Test Process

Currently, Function Test is performed along a function test specification, with the goal of testing conformance of the target function to its specifications. A function may be described as a set of subfunctions. A set of test cases is derived aimed at covering all the specified functionalities (or use cases). Test cases are derived manually by testers by systematic analysis of the specification documentation. The function test phase is organised in a specified number of stages. All the failures discovered within a stage are logged and reported to software designers, who trace failures back to code and correct them. A new software version is then released, which is resubmitted to test in the next test stage.

Function Test stops when all test cases defined in the test case specification have been successfully performed, either at first try or after suitable code correction. Specific reliability objectives are not explicitly considered in test planning, and no estimation of achieved reliability is currently performed.

4.3 Description of the experiment

The execution steps of the proposed experiment are:

1) Define the reliability required for stopping the testing in the SRET approach, based on TEI quality objectives.
2) Analyse the baseline project to produce a list of its functions (meaning, according to Musa's usage, a specific task or part of the overall work to be performed by the system).
3) Define an operational profile for the baseline project, using the SRET method. In particular, identify the individual operations and determine their respective occurrence rates. In addition, define, where appropriate, severity classes for expected failure modes, so as to test more thoroughly those functions for which higher-severity failures are possible.
4) The subsystem analysed in step 1 will undergo two parallel Function Test exercises; we will speak of two test tracks, namely:
   a) track F (for TEI functional test), to be tested according to the standard TEI method: test cases are ad hoc selected to obtain 100% coverage of all the functionalities identified in the manual analysis of the functional specifications.
   b) track R (for random test), to be tested following the new approach, based on the operational profile defined in step 2.

   The two tracks will be assigned to two different groups of persons that will operate in a totally independent way in order not to bias the results of the experiment.

   The duplication of the Function Test phase into modified and unmodified test tracks, run side by side in the same case study, will allow us to derive significant measures of improvement through the comparison of performanceof the two groups.
5) Set up a test environment for the new test method. It requires random test selection, as described in Section 3, and handling of random variables. Automated tools to help launch the tests and log test results are already used in TEI. The evaluation of test outcomes (success/failure), which is always a difficult problem, is here exacerbated by the uncertainty of the randomly selected inputs. We will adopt a viable solution to automate it as far as feasible, exploiting the relative simplicity of the subsystem at hand.
6) Execute the test cases. The track R will be tested using the SRET approach until the requested reliability is obtained. In parallel, starting from the first release of the software, the track F will be tested using the standard TEI organisation into successive releases after corrective actions until full coverage is obtained.
7) Experimental results are evaluated. We will evaluate the cost/effectivenessof the SRET approach in the context of TEI development process with regard to direct improvements, such as increase of reliability, and also indirect benefits, such as reduction of maintenance costs or improvement of the design phase.

5. Current situation

According to the experiment described in Section 4, the following steps have been executed.

◆ *Defining the Required Reliability*

TEI software quality objectives require that the number of failures over the product size found in six months of operation is less than 0,15. We have estimated that only 1% of the activity of a CTM-AXE10 switching is spent for administrative functions. From this, we derived that the reliability required for the system under test is in the order of 10^{-3} failures per hour of operation.

◆ *Defining the Operational Profile*

Starting from the analysis of the administrative functions, the operational profile has been defined. Operator manuals and previous experiences in similar systems have been used as input to derive the operational profile. In the following tables we describe the results of the obtained profiles. Regarding the operational profile, we used an implicit approach: the profile is represented by a behavioural tree, of which in Fig. 2 we show only a branch. The severity of the failures is considered when assigning the probabilities.

Customer	Probability
Telecom Operator	100%

Table 1. Customers Profile

User	Probability
Operator	100%

Table 2. Users Profile

Mode	Probability
Installation	10%
Operation	90%

Table 3. System-modes Profile

Function	Installation-mode Probability	Operation-mode Probability
Definition of a route	26,2%	14%
Deletion of a route	0,78%	7%
Change of route parameters	1,56%	14%
Print of route parameters	31,4%	21%
Connect a device to a route	28,8%	17%
Disconnect a device from a route	0,78%	7%
Print device data	5,24%	10%
Print device connections data	5,24%	10%

Table 4. Functional Profile

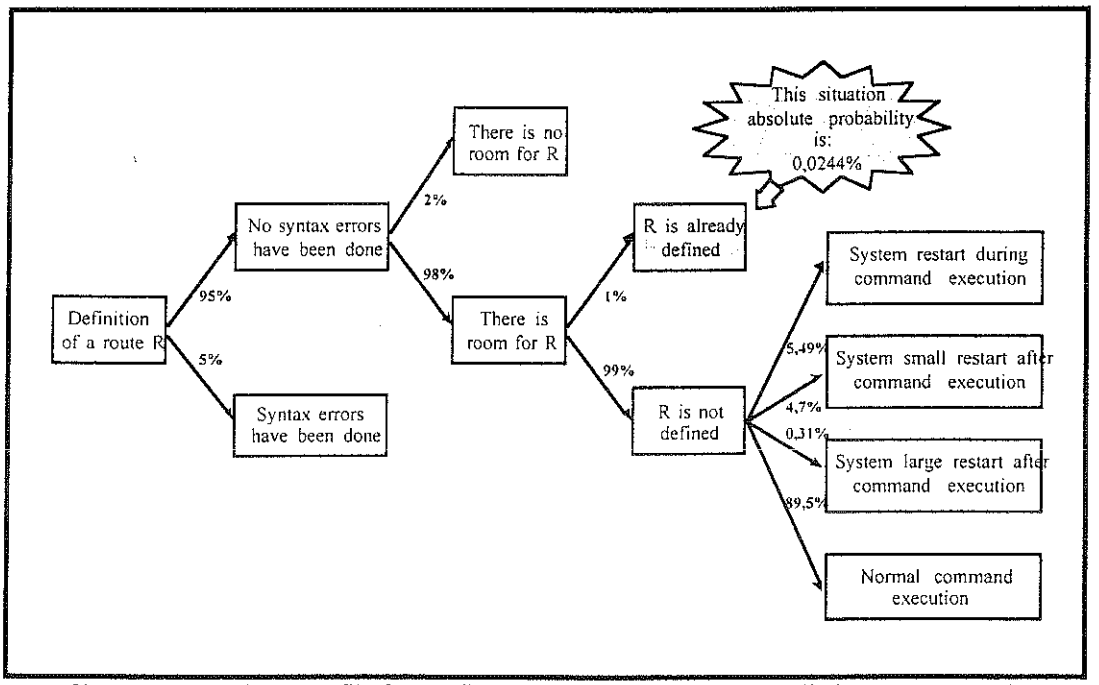

Figure 2. Operational profile for 'Definition of a route' in the ' Installation' system-mode

♦ *Test Cases Definition*

With regard to track F, we used the test instructions prepared according to the standard function test process. With regard to track R, for each leaf of the tree we have specified one or more test cases.

♦ *Set-up of the Test Environment*

To set-up the test environment, we have integrated a proprietary Ericsson tool, called 'AUTOSIS', with an ad-hoc developed tool 'STUT' and a few commercial tools to compute the reliability of the system.

AUTOSIS is a tool used for testing any system possessing an interface for man-machine interaction. The test is performed by sending commands towards the test object and by analysing the resulting printouts. The instructions to perform the test are supplied in the form of an AUTOSIS test instruction (TI) written prior to execution. The TI contains AUTOSIS instructions interpretable by AUTOSIS and commentary text. The tool generates two output files:

- a log file with the trace of the execution to be used in the debugging of errors;
- a report file where the execution time and the result of each test case are recorded.

AUTOSIS has been extended in order to handle random variables.

STUT is a tool that has been developed to select the test cases according to the SRET approach. It generates a file specifying the test instructions to be used as input to AUTOSIS by processing the following input data:

- operational profile and related test instructions for each leaf of the tree;
- definition of random variables;
- required reliability value.

The usage of that tool does not imply any specific training for the testers, because the structure of the input file is

quite similar to the structure of the AUTOSIS input file.

To compute the reliability of the system under test, we are using commercial tools that include the following features:

- collecting failure and test time information;
- calculating estimates of model parameters using the collected information;
- testing the fit of a reliability model against the collected information;
- selecting a model to make predictions of remaining faults, time to test, or other items of interest;
- applying the model.

In parallel, we are also evaluating a set of reliability models on TEI historical data, in order to select the model that best suits to TEI software process. These models are: Jeliski-Moranda, Goel-Okumoto, Hyperexponential, S-Shaped, Musa-Okumoto.

♦ *Execute the Test Cases*

We have now started to execute test cases by performing the following steps:

1. using STUT we select a set of test cases;
2. using AUTOSIS we execute the selected set of test cases: each time a failure is observed, we fix it, update the reliability of the system under test and return to step 1.

We will continue this process until the required reliability is reached.

6. First Remarks

Even if the pilot project is not finished yet, it is possible to bring out some results from the comparison between SRET and the traditional coverage approach.

Some remarks point out the effort required to apply the SRET approach:

- the SRET strategy requires extra effort with respect to current process due to the definition of the operational profile. If new systems have to be modelled it will be quite difficult to find usage data to assign the right probabilities;
- the definition of test instructions can be more difficult because in SRET they have to be specified in a more abstract way;
- considering the high number of test cases to be executed, a completely automated environment is required. The Ericsson AXE10 target environment is not completely automated, so we had to limit the application of the case study to the simulated environment. As a consequence, it was not possible to execute some categories of test cases that required to be run on the target system, like tests for evaluating performance or involving traffic.

However, we could improve the standard coverage approach by applying some lessons learned from the application of the SRET strategy. In fact, the actual use of the SRET approach has been carrying out the following benefits:

- a better understanding of the function/feature in the early phase of the development process;
- a structured approach to identify the test cases (we have used a behavioural tree), that made the identification of test cases easier. We also identified some new cases not considered following the coverage approach. By executing them, two new faults were discovered in the released product;
- a prioritisation of functions and test cases to shorten the lead time in date-driven projects. In fact, the exit criterion of the standard Function Test process requires that the whole set of test cases has to be executed successfully. This criterion involves an execution time related to the number of test cases. The introduction of a priority criterion and a classification of incident severity levels allows a tester to identify exit criteria to shorten the lead time in date-driven projects. This priority criterion may be identified by using the operational profile, i.e.: the most critical set of test cases as experienced by the customers has to be executed and passed without any failures of major incident severity levels;
- an improved work organisation between developers and testers, both involved in the operational profile design. The close collaboration between testers, system engineers and product users has produced valuable side benefits, such as a deeper understanding of user needs, less ambiguity in the specification of system requirements, and the possibility for testers to contribute to system reviews;
- expertise has been gained during this case study both with respect to the construction of an operational model, and with the use and tuning of reliability models, thus enhancing TEI's staff awareness of reliability issues, and allowing a reuse of those competencies in future projects. Should the new technique prove successful, applying the new test technique will allow TEI to control the reliability of its software products and the associated test cost.

These first remarks are so far only qualitative. We look forward to collecting quantitative results from the prosecution of the experiment, so that we can confirm or reject our conjectures on more concrete grounds.

References

[1] S. Brocklehurst and B. Littlewood, "New Ways to Get Accurate Reliability Measures", *IEEE Software*, Vol. 9, n. 4, pp. 34-42, July 1992.

[2] M. R. Lyu (Ed.), *Handbook of Software Reliability Engineering*, McGraw-Hill, 1996.

[3] J. D. Musa, "Operational Profiles in Software-Reliability Engineering", *IEEE Software*, March 1993, pp.14-32.

[4] J. D. Musa, "Software-Reliability Engineered Testing", *Proc. of the 9th Int. Software Quality Week*, S. Francisco, USA, May 21-24, 1996, paper 2Q2.

[5] M. C. Paulk, B. Curtis, M. B. Chrissis and C. V. Weber, "Capability Maturity Model for Software, Version 1.1", SEI CMU Int. Rep. CMU/SEI-93-TR-24, February 1993.

Acknowledgements

The authors are grateful to Norman Fenton and Lorenzo Strigini of the CSR, City University, for commenting earlier versions of this work, and to John Musa, the founder of SRET, for useful references and hints.

Corresponding author:
Lombardi Gaetano
Ericsson Telecomunicazioni SpA
Via Anagnina, 203
00040 Roma, Italy
Tel. +39 6 7258.3012; Fax +39 6 7258.3940
e-mail: G.Lombardi@rd.tei.ericsson.se