Introducing a reliability measurement program into an industrial context

Antonia Bertolino, Gaetano Lombardi, Eda Marchetti, Emilia Peciola

Abstract

We report on an experience in the industrial employment of software reliability engineering techniques at Ericsson Telecomunicazioni S.P.A.. The experience includes a case study on the application of operational statistical testing techniques to a selected baseline project. This case study is aimed at evaluating the hypothesis that operational testing can achieve higher reliability than conventional deterministic approaches, with lno or imited additional costs. We have adopted Musa's SRET approach for test design and guidance. At the same time, we are also investigating the application of software reliability models on some available failure data, in order to understand the feasibility of the more comprehensive objective of introducing a reliability measurement program. While the experience is still going on, we describe some preliminary results.

1. Introduction

Software Reliability Engineering (SRE) encompasses state-of-practice techniques and tools for analysing, managing and improving the reliability of software products [4]. The central focus is on the quantitative evaluation of the operational behaviour, as contrasted with the qualitative evaluation provided by conventional (non reliability based) validation and verification activities. In this paper we report about an ongoing experience at Ericsson Telecomunicazioni S.p.A. in Rome (TEI in the following) aimed at experimenting the applicability of SRE techniques in this industrial context.

In agreement with corporate policies, the TEI software development process undergoes assessment and improvement actions on regular bases and moreover failures in the field of released products are routinely monitored for the first six months of operation. Though, so far, specific reliability objectives have not been explicitly set for the development process, nor any attempts to measure the achieved reliability of TEI products have been tried. TEI is now going to improve its competence in the area of statistical process control and prediction methods. In particular, it feels to be at a right stage of maturity for trying SRE techniques.

In this perspective, the objective of this experience is twofold. On the one hand we intend to experiment a reliability guided testing process in contrast with the purely functional approach currently taken: the hypothesis we want to test is that operational statistical testing can achieve higher reliability than functional (deterministic) testing. On the other hand, we would also like to make a preliminary (and modest) attempt to introduce reliability measurement techniques: we have started by analysing available field failure data of completed projects, as well as operational test results which are being collected in the course of this experience.

In the next section, we describe the TEI development process and the motivations for this experience. In Section 3 we illustrate a case study aimed at evaluating operational statistical testing within the TEI testing process. In Section 4 we report some results of the failure data analysis and reliability model selection activities. Finally, in Section 5, we outline some preliminary insights and hint at further work.

2. TEI process

Within the frame of the company-wide Ericsson System Software Initiative (ESSI), regular assessments are being performed at all Ericsson Software Design Centers according to the Capability Maturity Model (CMM) for software, Version 1.1 [8]. Main objectives of ESSI are to identify key areas for process improvement and to propose a framework for subsequent improvement actions. An 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 TEI were found to be at the Defined level of maturity (level 3). Although this result was very satisfying, TEI is now going to initiate some of the level 4 practices. The organization intends to improve its capabilities in statistical process control and in prediction methods. This experience is one of the improvement activities launched in that area.

One objective placed by ESSI is reducing below a determined value the fault density figures that are obtained by monitoring the first six months of operation of released products. Fault density is measured by the ratio between the cumulative number of failures observed in that period over the product size, expressed in lines of code. Root Cause Analysis (RCA) of reported failures is routinely performed, to track back failures to the phase in which they have been originated. An important finding of RCA for TEI products was that a high percentage of failures (48%) corresponded to software faults that could have been discovered during the Function Test phase, one of the four test phases in TEI test strategy, namely:

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

Currently, Function Test is performed along a function test specification, with the goal of testing conformance of the target function to its specifications. Test cases are derived manually by testers, by making a systematic analysis of the specification documentation and trying to cover all the specified functionalities (or use cases). 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 testing in the next test stage. Function Test stops when all the 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.

3. The case study

In agreement with ESSI objectives, and specifically in order to reduce the fault density figures via a more efficient Function Test, we have set up a case study. The hypothesis we intend to test is that the cost/effectiveness ratio of the TEI Function Test process can be improved by introducing explicit reliability objectives to guide it. To this purpose, a baseline project has been selected (described in Section 3.2), and reliability guided testing techniques will be applied to the Function Test phase of this baseline project, i.e., the test selection will be performed so as to reproduce the expected usage of the system in operation. In this way we expect to achieve higher reliability levels for the tested product after release. Although they are not primary objectives for this case study, other important measurable benefits are expected as well. Specifically, SRE techniques will provide a means for predicting product reliability in operation (via analysis of the test results under a reliability growth model). In parallel to this case study, we have also started the evaluation of failure data available from past projects using well known reliability growth models. The aim of this analysis is to assess current reliability levels achieved with the conventional testing approach for similar products. Some results are reported in Section 4.

3.1. Testing guided by reliability objectives

Our interest in SRE techniques is mainly in the application of SRE activities to the testing stage. Musa has recently introduced the Software Reliability Engineered Testing (SRET) methodology, whereby "SRET is testing guided by reliability objectives and expected usage and criticality of different operations in the field" [7]. The SRET approach consists of five principal activities (see [4,6,7]):

1. Define the reliability objectives;
2. Develop the operational profile;
3. Prepare the tests;
4. Execute the tests;
5. Interpret test results.

Reportedly [4, Chapter 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 guided testing. Hence, more precisely, this case study is aimed at testing the conjecture that Function Test conducted according to Musa's SRET approach leads to cost-effective reliability improvements over current TEI practice.

In this case study, we have set up a testing process precisely reflecting the above five stages approach. In a sense, the most critical part was the second activity, i.e., the development of an operational profile for the software under test. Musa defines an operational profile as "the set of operations and their probability of occurrence". To obtain the list of operations, SRET outlines a five steps procedure, starting from identifying a list of possibly different customer types, progressively breaking them down into different user types, and then into the different modes in which a user can invoke the system. In turn, for each system mode a functional profile is defined, and finally the functional profile is converted to the operational profile.

We emphasise that this is the first time that reliability guided testing is attempted within TEI and consequently in some occasions during the implementation of the SRET process 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.

As already said, other benefits are expected from adopting the new approach. In particular, unlike the current Function Test approach, SRET provides a means of predicting product reliability in operation (last activity "Interpret test results"). Within the experiment, we thus 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 which would have been obtained by the conventional approach, as well as to that experienced in operation for similar products. Also, we intend to keep track of maintenance costs after product delivery, to check whether SRET actually reduces them, as it claims.

3.2. The baseline 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. CTM 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 baseline project consists of the administration functionalities of the links between CCFP and CTM Exchange.

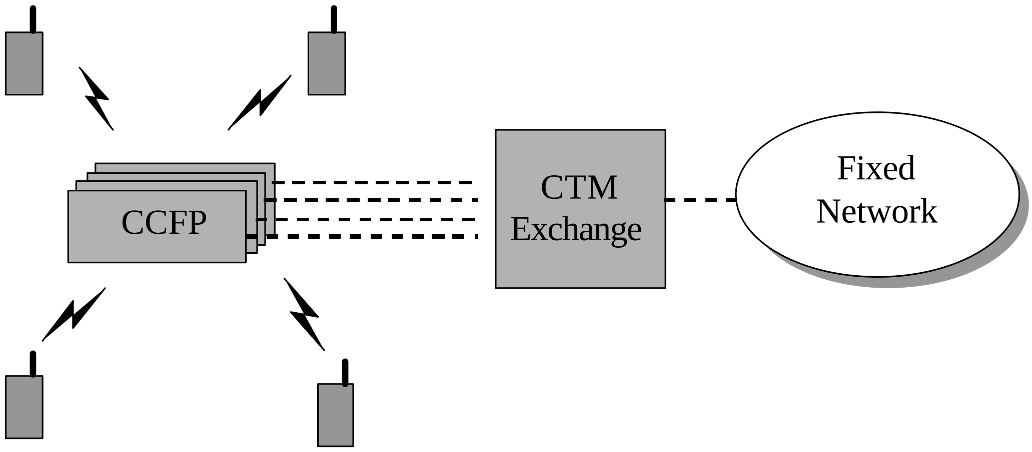

Figure 1. Ericsson CTM architecture

3.3. Experiment plan

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 baseline project will undergo two Function Test tracks in parallel:
   a) track F (for TEI Function Test), to be tested according to the conventional TEI method: test cases are deterministically 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 3.

   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 direct, quantitative comparison of the reliability achieved in the two tracks.
5) Set up a test environment for the new test method. SRET requires random test selection. Automated tools to help launch the tests and log test results are already used in TEI; these will be adapted to allow for handling of random variables. 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. 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, track F will be tested according to the

standard TEI approach into successive releases after corrective actions until all the test cases are covered with no failure.

7) Evaluate experimental results. We will evaluate the cost/effectiveness of 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 efficiency.

3.4. Current situation

With reference to the experiment plan described above, we depict here the current situation.

3.4.1. Definition of the required reliability

ESSI improvement objectives require that fault density, i.e., the number of the failures found in the first six months of operation over the product size, is reduced to less than 0,15. We have estimated that only 1% of the activity of a CTM-AXE10 switching is spent for administrative functions, that are those in the baseline project. From this, we estimated that the reliability required for the system under test is in the order of 10^{-3} failures per hour of operation.

3.4.2. Definition of the operational profile

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, according to the SRET approach. Regarding the operational profile, we used an implicit approach, i.e., the profile is represented by a behavioural tree. In Fig. 2 we show only a branch of this tree. The severity of the failures is considered when assigning the probabilities.

3.4.3. Test case 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 (to be randomly selected).

Table 1. Customers profile

Customer	Probability
Telecom Operator	100%

Table 2. Users profile

User	Probability
Operator	100%

Table 3. System-modes profile

Mode	Probability
Installation	10%
Operation	90%

Table 4. Functional 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%

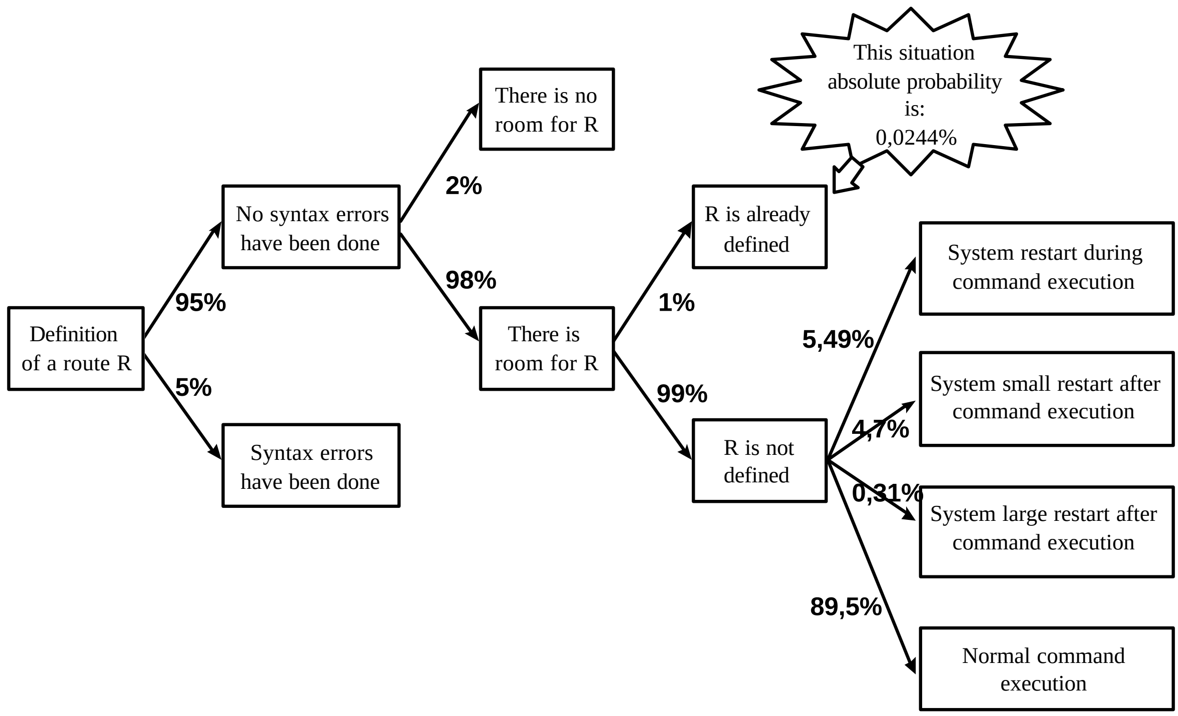

Figure 2. Operational profile for 'Definition of a route' in the ' Installation' system-mode

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

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

AUTOSIS has been extended in order to handle random variables with the integration of STUT. 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 information:

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

The usage of STUT does not require any specific training for TEI testers, because the structure of the input file is quite similar to the structure of the AUTOSIS input file.

Finally, to compute the current reliability of the system under test, we are using readily available tools (see Section 4).

3.4.5. Execute the test cases

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

i) using STUT, we select a set of test cases;

ii) 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 i).

This process is being repeated until the required reliability is reached.

4. Evaluation of reliability

Before starting to experiment the SRET approach, we thought it necessary to assess the current situation in terms of the reliability that is achieved by TEI development process. In fact, the hypothesis at the basis of this experiment is that operational statistical testing can achieve higher reliability than conventional (purely functional) TEI approach. Although, for the testing of this hypothesis, the experiment plan includes the comparison of the reliability obtained in the two parallel test tracks, one according to the conventional approach and the other following the new one, we think that assessing the reliability achieved on other similar products can also provide useful indications. As already explained, TEI monitors and logs all failures found in the field for the first six months of operations. At present, such data are not used directly for reliability estimation; on the contrary, they are used to evaluate the failure densities, and the timing of failures is thus not directly accessible. However, we have got access to the detailed failure report, including timing information, for a completed product which is similar to the baseline product.

To these data, we have applied standard techniques for reliability evaluation; the findings of this analysis are summarised in the next two subsections. We are the first to admit that these first evaluations are quite limited in their scope. Though, we hope that we can soon augment these first results with more data, even because we feel that this experience can promote a more collaborative attitude towards SRE techniques in the producing units.

4.1. Data analysis

We performed first a detailed analysis of the data available, in order to verify their suitability for applying reliability growth models.

The very first assumption for reliability assessment is that the analysed product is exercised according to the operational profile. This is not the case for the Basic, Integration and Function Test phases (at least, not so far). Thus, for the product examined, only the failure data relative to the six months of monitored operation are meaningful. For these six months, we had a reported number of 35 failures, collected over a set of five installations run in parallel. The products run continuously and failures are reported on a daily basis. For this reason, and also for complying with corporate established attitudes, we opted for reliability growth models in the class of number of failures per time period. The period to be considered as the time unit naturally corresponded to one calendar day, in turn corresponding to five days of execution time, by considering the five plants monitored.

Another very basilar assumption for using reliability growth models is that the failure data exhibit a growth in reliability. Somewhat surprisingly, this was not the case if we considered the data set as a whole. At a closer inspection, though, we noticed an anomalous behavior in the first two months: just two failures in the first month, and a set of eleven failures all concentrated in the second month. Thus, we decided to filter the data, by discarding the first month of operation. After this filtering, the failure data relative to the last five months (33 failures) exhibited an increase in reliability, although this remained very small. These informal analyses were confirmed by the Laplace Test, which has been applied to the data to observe the trend in reliability [3]; the Laplace test confirmed that reliability was increasing, but very slowly. We are studying causes for this.

4.2. Model fitting

A major step in predicting software reliability is to decide which model is the best one for a particular context. It is now well known that there is no software reliability model that performs better than any other one in any case. Fortunately, recent theoretical advances in the field [1,2] have largely eliminated some difficulties that arose in the choice and calibration of these models for a specific case.

More importantly from the practitioner's perspective, several software reliability tools are today available that help users to make this choice without requiring them a deep knowledge of the mathematical aspects of the software reliability models. With these tools, in fact, the user may readily apply the best known software reliability models at his/her set of failure data and then choose, by analysing the produced results, the model that gives the best predictions.

In our case, we adopted the well known tool CASRE [4,5], that includes, for the failure per time period data, the following software reliability models: Brooks/Motley (BM), Schneidewind (SM), NHPP (also known as Goel-Okumoto), Generalized Poisson (PM), Yamada S-Shaped (YM).

For each model, the tool automatically computes the prequential likelihood function (PL), which is a general means of comparing the accuracy of predictions provided by more models when applied to a same failure data set [1]. In particular, the best model is characterized by the highest value of PL (we must specify that for convenience CASRE computes -ln(PL), so in this case the best model is that showing the lowest value).

For each model CASRE also shows the results of executing the Chi-Square test, which is a standard way to check the accuracy of predictions with respect to raw data (goodness-of-fit measure). In particular, we decide that a selected analytical distribution can be used to correctly represent a given empirical distribution if the Chi-Square value is within a specified interval.

The results of the accuracy analysis made by the tool to our failure data are shown in Table 5. We can observe that the evaluation of the Chi-Square test (in this case with 4 degrees of freedom) brings us to reject the Yamada S-Shaped model (the models fit well for the analysed data set if the Chi-Square value is within the interval [0.207, 9.49]).

Considering the value computed for (-ln PL), we can see that the NHPP and SM models give the most accurate results for the data source. Indeed, for our data set the observations period have the same length, so NHPP and SM are equivalent [4, Chapter 3].

Table 5. Analysis of models

	BM bin	**BM pois**	**NHPP**	**YM**	**SM**
-ln PL	36,677 (3)	45,998 (5)	36,677 (1)	42,494 (4)	36,677 (1)
Chi-Square	4,913 (3)	4,936 (4)	4,772 (1)	26,050 (5)	4,772 (1)

Finally, we show below in Figure 3 the cumulative number of failures predicted versus the raw data and in Figure 4 the relative error, computed as:

$$\frac{(PredictedNoFailures - ActualNoFailures)}{ActualNoFailures}$$

5. Conclusions

Even if the case study is not finished yet (we have encountered some difficulties not directly related to the implementation of the case study, but which arrested the work anyway), it is already possible to bring out some remarks from the comparison between SRET and the conventional approach.

Some observations regard the effort required to apply the SRET approach:

• SRET requires extra effort with respect to Function Test standard process, in particular for 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 (for the need to consider random variables);

• Considering the high number of test cases to be executed, a completely automated environment is required. The Ericsson AXE10 target environment is not completely automatable, so we had to limit the application of this case study to the simulated environment, that is the administrative part. 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.

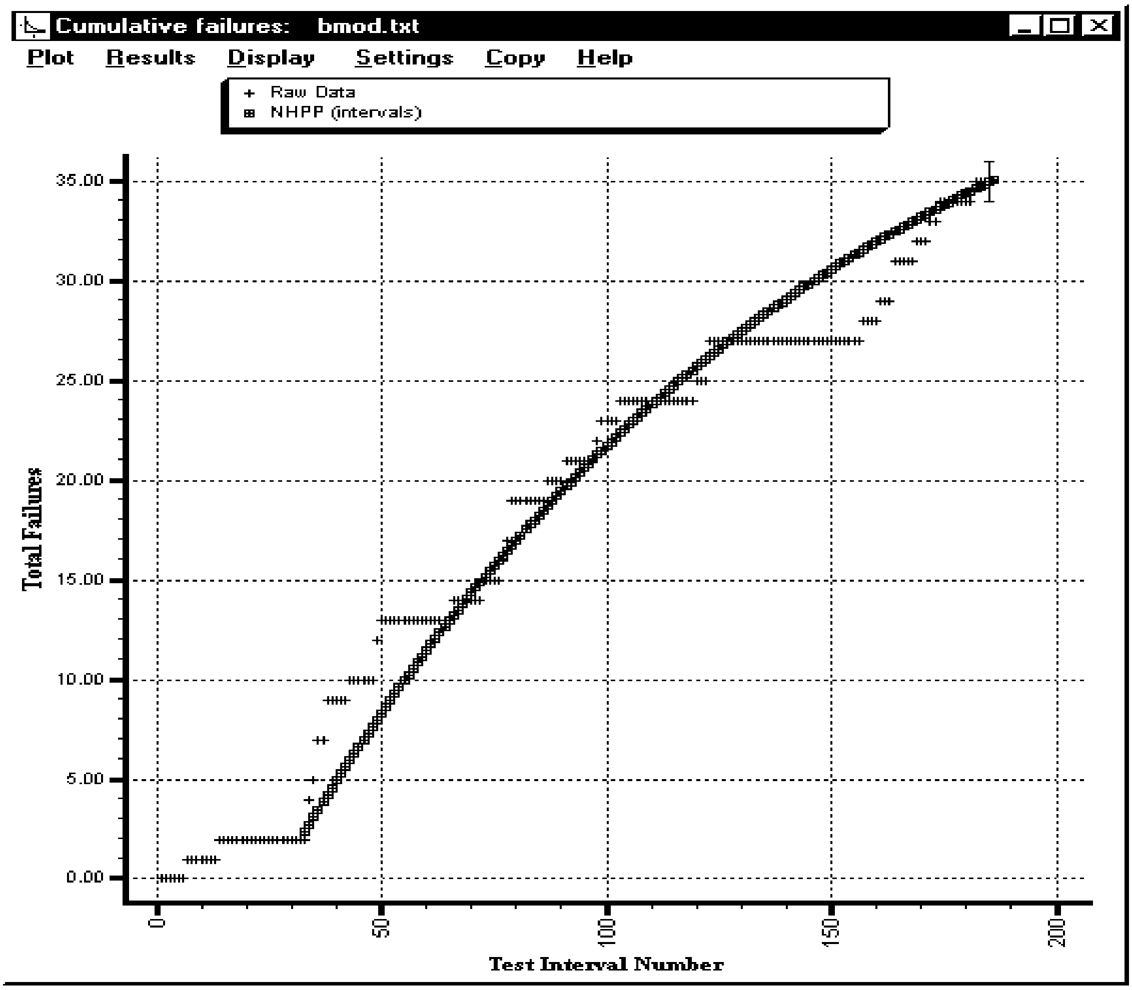

Figure 3: Curves derived by CASRE for the cumulative number of failures relative to the NHPP model

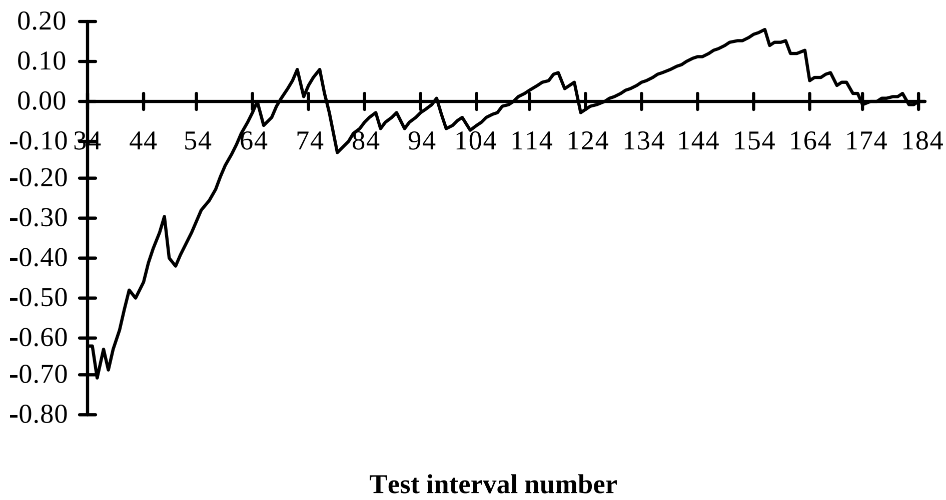

Figure 4: Relative error of NHPP model for the cumulative number of failures

However, we could improve the standard approach by applying some lessons learned from the application of the SRET strategy. So far, 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 test cases that would not have been considered following the conventional 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 derivation of the operational profile. 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 techniques 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.

We also intend to augment the reliability measurement activity with the analysis of more data, and to understand why the showed increase of reliability appears so slow.

6. References

[1] Abdel-Ghaly, A. A., Chan, P. Y., and Littlewood, B., "Evaluation of Competing Software Reliability Predictions", IEEE Trans. Soft. Engineering, Vol. SE-12, No. 9, Sept. 1986.

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

[3] Kanoun, K., Kaaniche, M., and Laprie, J. P., "Qualitative and Quantitative Reliability Assessment", IEEE Software, Vo. 14, No. 2, March 1997.

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

[5] Lyu, M. R., Nikora, A. P., and Farr, W. H., "A Systematic and Comprehensive Tool for Software Reliability Modeling and Measurement", Proc. of the 23rd Int. Symp. on Fault-Tolerant Computing (FTCS 23), Toulouse, France, June 1993, pp. 648-653.

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

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

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

Acknowledgements

Paolo Di Benedetto heavily contributed to this work with the development of the STUT tool, and helped in the phase of definition of the operational profile.