A method for software reliability analysis and prediction, application to the TROPICO-R switching system.

Karama Kanoun, Marta Bastos-Martini, Georges Moreira de Souza

A METHOD FOR SOFTWARE RELIABILITY ANALYSIS AND PREDICTION APPLICATION TO THE TROPICO-R SWITCHING SYSTEM

K.KANOUN*, M.R.BASTOS MARTINI, J.MOREIRA DE SOUZA****

* LAAS-CNRS, 7 Avenue du Colonel Roche, 31400 Toulouse, FRANCE
** CPqD-TELEBRAS, CP 1579, 13100 Campinas-SP, BRAZIL

ABSTRACT

The aim of this paper is to:

- develop a method of analysis and evaluation of the software reliability,
- apply these methods in order to (i) follow up the evolution of reliability during the test phases, (ii) plan the maintenance effort and (iii) predict the reliability of the Brazilian switching system: TROPICO-R in operational life.

This method is primarily based on the results of reliability growth tests. Reliability data (concerning validation test, field trial and operational life phases) are then partitioned according to these results and two kinds of reliability growth models are applied according to the data trend. When the data exhibit reliability decay followed by reliability growth, an S-Shaped model is applied in order to predict the evolution of the cumulative number of failures which can be used for test and maintenance planning. In operational life, the hyperexponential model allows to predict the software residual failure rate that can be used as a qualification index for the software product.

Key words : Software reliability, reliability growth, software reliability modeling, trend analysis, reliability prediction.

Mailing address : K. Kanoun, LAAS, 7 Av. du Colonel Roche, 31077 Toulouse Cédex, France

Phone : +33 / 61 33 62 00,

Electronic mail: Kanoun@laas.laas.fr

INTRODUCTION

Quality control is one of the key factors in the success of a project which must meet stringent reliability requirements. Quality control techniques consist of the analysis of failure data and the quantitative evaluation based on reliability models.

Evaluation of the reliability of computer based systems requires paying particular attention to the hardware, the software and the system as a whole.

The software reliability is evaluated so that:

- in terms of the product itself, the software reliability satisfies the requirements of system specifications,
- in terms of the production process, the development can be managed in order to obtain a dependable software within the scheduled delays.

Management of the development/maintenance requires realistic time schedules of life cycle phases: test, qualification and maintenance phases for instance. To do this, it is necessary to evaluate continually the reliability attained following the debugging effort in order to review the initial planning. Reliability techniques allows assessment of resource allocation, quantification of the development/maintenance efficiency and monitoring of the reliability growth in order to ensure that the software reaches a sufficient level of quality within the schedules and planned costs.

During the operational life it is important to have confidence in the service delivered by the system and to show quantitatively that the software has reached the required level of reliability. Software quality objectives can be stated by means of usual reliability measures such as the mean time to failure or the failure rate; reliability modeling and evaluation help estimate these quality measures from the observed failure data.

In this paper an evaluation method is proposed for software reliability analysis and prediction. It is illustrated through the application to the TROPICO-R switching system (which is a telephone switching system). It will be shown (i) how reliability growth models can help manage of the

development and maintenance processes, and (ii) how these models can be employed to estimate the software failure rate in operational life.

The data regarding the software failures detected in the TROPICO-R system were recorded in appropriate Failure Report (FR) sheets drawn up by the product development and follow-up engineers. 461 software FRs have been established during 27 months including the last two phases of the development (validation and field trial tests) and operational life (during which fifteen sites were installed).

This paper presents the method used to follow up the software reliability of the TROPICO-R switching system during the development and operational life phases (this method is more general in purpose and can be applied to other systems). Section 2 deals with the evaluation method, more specifically (i) the reliability growth tests and (ii) the choice of the evaluation model. The following section describes the test environment and the failure data recorded. The application of the models to the TROPICO-R failure data and their use as an aid tool for planning and reliability prediction are addressed in section 4.

1. RELATED WORK

Numerous papers deal with software reliability growth modeling and evaluation. However, papers following a global method are rare [15]. A theoretical method for the statistical aspects study has been given in [5, 23] and application examples can be found in [11, 13 or 27] .

Theoretical aspect concerning reliability trend tests are studied in [3,7] and their application to real data can be found in [17].

Definition and application of the reliability growth models have been discussed in numerous papers such as [1,12, 16, 18 or 23].

Application to concrete cases are much fewer since results are often confidential. However there are interesting applications of reliability growth models in [2, 8, 9, 22].

The above list of references is not exhaustive, only few ones are listed in order to give an idea about the work that has already been achieved in this field.

2. EVALUATION METHOD

The method of analysis and evaluation of the software reliability corresponds to the global method defined in [16] and [17]. It consists of two main stages:

- thorough analysis of the system and the data collected,
- reliability trend analysis and software reliability prediction.

As in the case of any reliability study, the first stage basically consists of gathering information on the system itself, the functions to be performed, the system structure and utilization environment, the life cycle phases under study, the software correction and the maintenance policies, the manner in which data is (or have been) collected... This stage is system related and has to be adapted to the nature of the system being studied.

The second stage is more systematic and is applicable to any software data. It is a two step process:

- first, a preliminary data processing provides (i) qualitative information on the software behavior and (ii) useful quantitative information on the evolution of reliability (i.e. trend analysis of the collected data), in order to perform the model applications,
- then, reliability growth models are applied.

This section consists of extending the work presented in the previous references. It will consist of:

- the Laplace test which is well-suited to testing the growth of the random variable: *time between failures* and which will be adapted to the random variable: *number of failures*, the latter form is more suitable to a great number of collected data (as in the case of the TROPICO-R),
- the notion of local and global reliability growth which will be introduced,
- two kinds of reliability growth models which will be used depending on the measures to be evaluated and the objectives of the evaluation: (i) development/maintenance management and planning or (ii) reliability in operational life.

It is worth noting that these objectives lead to consider two different random variables :

- for maintenance management and planning, the random variable will be the number of failures per unit of time,

- for reliability evaluation, the random variable "interval between failures" is more suited to estimate the MTTF (Mean Time To Failure) of the software or its failure rate.

These two random variables are not independent: the first one can be deduced from the second one (the inverse is not true). the second form needs more precise data collection.

2.1.Trend Analysis

Existing reliability growth models allow to model three kinds of reliability behavior according to the nature of the Rate of OCcurrence Of Failures (ROCOF) which is the number of failures per unit of time:

- reliability growth with a decreasing ROCOF tending to zero when t tends to infinity [12, 14, 21],
- reliability growth with a decreasing ROCOF tending to a non zero value when t tends to infinity [17, 18],
- reliability decay prior to reliability growth, i.e., the number of failures per unit of time first increases and then decreases to zero (the ROCOF is bell-shaped): these models [10, 28] are called S-Shaped models because the curve representing the cumulative number of failures is S-shaped.

It is therefore very important to identify the trend (growth/decay) in order to use the appropriate reliability growth models. The Laplace test, proposed in [3], can be used for this purpose. This test was developed for the random variable interval between failures, when the random variable is the number of failures per unit of time, the expression of the Laplace test factor u(k) at time unit k can be deduced as suggested in [7, p.54]. The details of u(k) establishment are presented in the annex. This factor is then given by:

$$u(k) = \frac{c - m}{\sqrt{\frac{k^2-1}{12\, y_k}}}, \qquad k=2,\ldots,p$$

where: $$c = \frac{\sum_{i=1}^{k} (i-1)\, n_i}{y_k}$$

$$m = \frac{k-1}{2}$$

p: is the total number of units of time,

y_k : the cumulative number of failures up to the k^{th} time unit,

n_i : the number of failures during the i^{th} time unit,

The positive values of u(k) indicate a decay of the software reliability whereas negative values indicate reliability growth.

The factor u(k) is evaluated among all the data collected up to the considered time unit: it indicates the *global* variation of reliability. It is worth noting that even in the presence of global reliability growth (decay), *local* reliability decay (growth) may occur. Local fluctuations can be detected by studying the variation of u(k): when u(k) is positive and tends to decrease it suggests a decrease in the number of failures observed over the considered period which means that, locally, reliability tends to increase although a global decay is observed.

This is summarized in figure 1 where the units of time during which the local tendency of u(k) changes correspond to inflexion points of the curve giving the cumulative number of failures.

Two types of tendency changes (inflexion points) can be distinguished:

- type 1: corresponding to the transition A-B which is common and expected during testing and validation phases,
- type 2: corresponding to the transition C-D which denotes a reliability regression, i.e., an increasing number of detected failures per unit of time.

It is worth noting that the evolution of the ROCOF indicates the local reliability growth or decay only.

Local reliability growth or decay may result from:

- dependency of faults: some software faults can be masked by others, i.e. they cannot be activated as long as the latter are not removed [25],
- the variation in time delay between the detection of an error and its removal; this delay is closely dependent on the nature of the activated faults: some faults are more difficult to identify than others and take longer time to be removed,

- the variation in the utilization environment: the variation in the testing effort during the debugging phase, the changing in the test sets, the adding of new sites during the operational life...

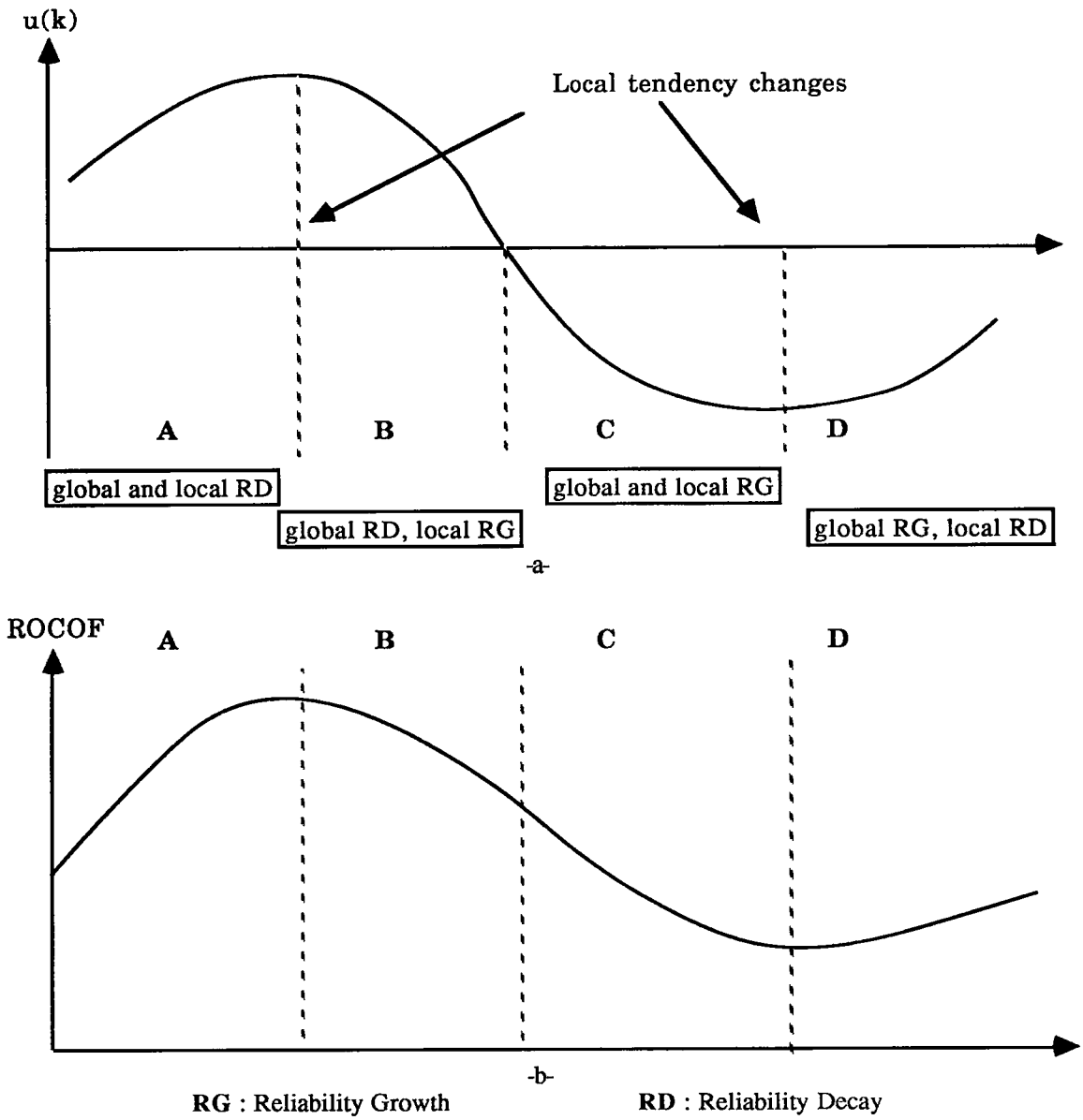


Figure 1 - Trend analysis using Laplace test and the ROCOF evolution.

These phenomena can act either separately or in combination.

Reliability regression can also result from slight specification changes.

2.2. Reliability growth models

The choice of the reliability growth model to be applied depends on a great extent on (i) the objectives of the study as well as on (ii) the nature of the trend displayed by the collected data.

With respect to the objectives, they are generally expressed in terms of measures. When the software is in operation, two kinds of measures are useful:

a) from the user's viewpoint: the mean time to the next failure (or the failure rate of the next failure), or/and the expected residual failure rate of the software in operational life in order to evaluate the reliability of the whole system,

b) from the designer's viewpoint: the expected number of failures among all the installed sites and then the number of corrections to be introduced in the software in order to estimate the maintenance effort still needed.

When the software is under development, the interesting measures are:

c) the evolution of the number of detected faults in the software in response to the test and debugging effort,

d) the expected number of failures for the following periods of time so as to plan the test effort and thus the time and the numerical importance of the test team.

Objective **a** can be reached through a model allowing evaluation of the mean time to (or the failure rate of) next failure and particularly the residual failure rate expected in operational life; the **hyperexponential** model [17, 18] seems the only suitable model enabling evaluation of the second measure. In fact all the other existing models assume that the failure rate tends to zero when t tends to infinity (nil residual failure rate in operational life).

Objectives **b**, **c** and **d** can be obtained through a model based on counting; in this case Non Homogeneous Poisson Process models (NHPP) seem suitable. Many NHPP models have been developed, our choice goes to three among them allowing three kind of reliability growth tendency to be modeled :

- the exponential model [12], allowing to model a continuously decreasing ROCOF tending to zero when t tends to infinity,
- an S-shaped [28] model, due to its ability to model a reliability decay prior to undergoing reliability growth; it can thus be applied to an observation period where the trend test has the form of the A-B-C period of figure 1,
- the hyperexponential model, due to its ability to predict the steady behavior in operational life.

The advantage of the NHPP models is that they are relatively independent of many hypotheses necessary for the derivation of several models which have been published, for instance no relationship between failures and corrections is assumed,

The hyperexponential and exponential models can only be applied to data displaying reliability growth: that is, period C of figure 1. However, in figure 1a, global reliability decay for A and B is due to the data corresponding to A, if the latter are not considered for the evaluation, B will display local and global reliability growth. These models can thus be applied to period B if the data corresponding to period A are not taken into account. Nevertheless, applying the models only in case of reliability growth (periods B and C) leads to discarding data pertaining to A and D from any data set, therefore prediction is delayed until observing a period such as B or C is observed.

The expression of the ROCOF of these models is as follows:

Exponential : $h(t) = N \Phi e^{-\Phi t}$, parameters: N, Φ, (1)

S-Shaped : $h(t) = N \Phi^2 t e^{-\Phi t}$, parameters: N, Φ, (2)

Hyperexponential: $$h(t) = \frac{\omega_1 Z_1 e^{-Z_1 t} + \omega_2 Z_2 e^{-Z_2 t}}{\omega_1 e^{-Z_1 t} + \omega_2 e^{-Z_2 t}},$$ (3)

with $0 \le \omega_1 \le 1, 0 \le \omega_2 \le 1$ et $\omega_1 + \omega_2 = 1$.

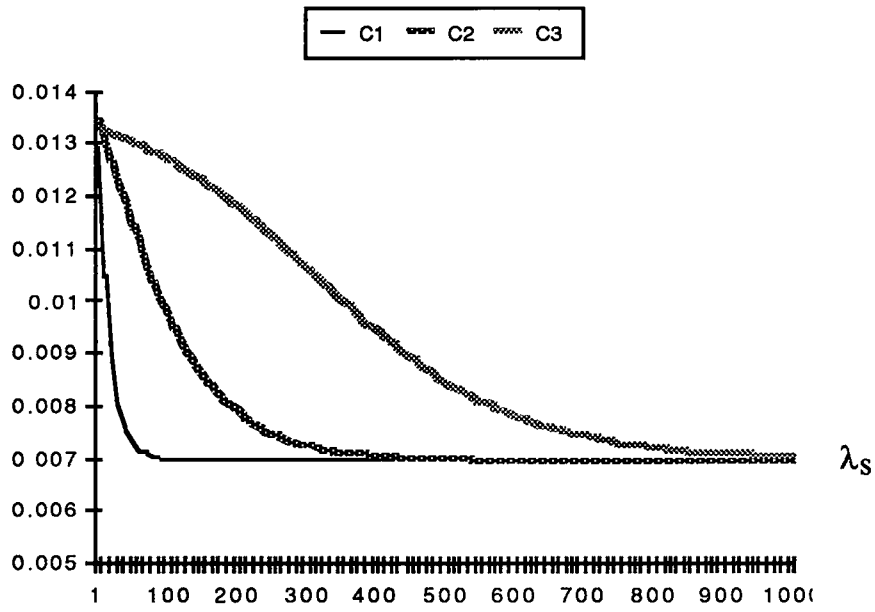
Let $\omega_1 = \omega$ and $\omega_2 = 1 - \omega = \varpi$. The corresponding ROCOF continuously decreases from an initial value $(\omega Z_1 + \varpi Z_2)$ to $\lambda_s = \inf \{Z_1, Z_2\}$ which is the **residual failure rate** of the software.

The $MTTF_i$ which is the expected time to failure i, given that failure (i-1) occurred at time s, is:

$$MTTF_i = \frac{\frac{\omega}{Z_1} \exp(-Z_1 s) + \frac{\varpi}{Z_2} \exp(-Z_2 s)}{\omega \exp(-Z_1 s) + \varpi \exp(-Z_2 s)} \qquad (4)$$

The relevant features of this model are:

- the rate of decrease with respect to time is adjustable through parameter (ω, Z_1, Z_2) tuning (figure 2),
- the fact that it asymptotically tends towards a non zero limit λ_s,
- the fact that it also enables availability growth modelling [17, 18].



Curve	ω	Z_1	Z_2
C1	0.1	$7\ 10^{-2}$	$7\ 10^{-3}$
C2	0.5	$2\ 10^{-2}$	$7\ 10^{-3}$
C3	0.9	$1.4\ 10^{-2}$	$7\ 10^{-3}$

Figure 2 : Hyperexponential model ROCOF for a set of chosen parameters.

The generalization of this model would be to take :

$$h(t) = \frac{\sum_{i=1}^{k} \omega_i Z_i e^{-Z_i t}}{\sum_{i=1}^{k} \omega_i e^{-Z_i t}}, \qquad \text{with} \sum_{i=1}^{k} \omega_i = 1, \qquad k \geq 2.$$

2.3. Model validation

A model can be analysed according to its repetitive and predictive capabilities:

- **repetitive capability**: indicates the ability of a model to reproduce the observed behavior of the software,
- **predictive capability**: indicates the extent to which the model, based on the observed data up to a given time, correctly predicts the failure behavior in the future.

This paper deals with the predictive features; repetitive features concerning the TROPICO-R application are given in [5].

Prediction of the failure behavior after a given time interval is based on failure data observed among this interval. The predictive capability is evaluated in the following manner:

1 use the failure data observed up to the time unit i=j (or a subset, say from i=k up to i=j), from which the failure behavior will be predicted,

2 estimate the unknown parameters of the model using this subset of data,

3 evaluate H(i), the expected cumulative number of failures up to unit time i, i>j,

4 evaluate the residue i.e., the *mean error* between the observed failure data and those foreseen by the model for time units i, $j<i\leq p$ (y_i - H(i)):

$$E_{k,p} = \frac{\sum_{i=k}^{p} |y_i - H(i)|}{p-j}, \qquad k \geq j+1 \qquad (5)$$

Steps 1 to 3 represent the so called prediction system [1].

The previous steps can be adapted to the case where the random variable is the time between failures, the residue in this case being equal to : (t_i-MTTF$_i$) where t_i is the observed time between failure (i-1) and failure i, and MTTF$_i$ the estimated one. In this case, the *relative residue* $R_{k,n}$ will be evaluated:

$$R_{k,n} = \frac{\sum_{i=k}^{n} (t_i - MTTF_i)}{S_n - S_k}, \qquad k \geq j+1 \qquad (6)$$

where S_i is the time of occurrence of failure i.

Expression (5) enables to check whether the model gives results close to the observed ones; on the other hand, expression (6) enables to see the bias in the evaluation: a positive residue means that globally the model underestimates or overestimates the times between failures.
These choices are mainly due to the very nature of the random variables: y_i or t_i.

2.4. Conclusions

The results issued from the the Laplace test application are interesting for two reasons :
- they allow to follow up the evolution of the reliability of the software and in case of reliability decay, one has to explain the reasons for the regression and try to react to the non explained phenomena,
- the inflexion points allow to partition data into subsets (called stages in the sequel) and to apply "reliability growth models" to these stages in order to enhance the quality of the estimation; type 2 inflexion points constitute the boundaries between these stages:

  - an S-Shaped model is applied to data situated between two type 2 inflexion points and contains a type 1 inflexion point,
  - models corresponding to reliability growth only are applied to data from a type 2 inflexion point to a type 1 inflexion point.

The idea of dividing data into subsets have been used in [15, 16 and 17] and in [26] (in an empirical way, for the latter), however the Laplace test used in the preceding manner helps this partition: it tells where it is better to do that.

3. TROPICO-R TEST ENVIRONMENT AND FAILURE DATA

3.1. Test Programme

The software test programme for TROPICO-R is a four-step process: (i) unit test, (ii) integration test, (iii) validation test,and (iv) field trial. The first three steps correspond to the test phases usually defined for the software life cycle. The last step consists of testing a prototype in a real environment that therefore approaches the normal operating environment. It uses a system

configuration (hardware and software) that has reached a sufficient level of quality after completing the laboratory testing programme.

The test programme carried out during validation test and field trial is decomposed into four tests:

1. *functional tests* verify the correct operation of all the specified telephone, maintenance and operation functions.
2. *quality tests* ensure that a specified rate of call loss is not exceeded. Call loss may be due either to a failure in the software and/or hardware of the system or to missing call resources (due to resource failure or because all the resources are already in use). At the beginning of the test we ensure that all the resources needed for making the calls available so any lost call can only be due to the failure of the software and/or hardware.
3. *performance tests* ensure the compliance with the quality of service requirements under a nominal traffic.
4. *overload tests* check the correct operation of traffic overload control mechanisms so that the calls accepted by the telephone exchange subject to this exceptional traffic comply with the specified quality of service.

The description of the whole quality control programme for TROPICO-R is given in [29 and 30].

3.2. Failure Report (FR)

The handling of failure data that affect the TROPICO-R product is achieved by utilizing an appropriate FR sheet. The FR contains the following information:

- date of failure occurrence,
- origin of failure: the system configuration in which the failure was detected and the conditions of occurrence,
- type of FR: hardware, software, documentation and the affected modules,
- analysis: identification and classification of the fault which led to the failure (coding, specification, interface,...),
- solutions: those proposed and the solution retained,
- modification control: control carried out by the person in charge of the module,

- edition control: control of the corrected modules,
- regression testing: results of the tests applied to the corrected module(s).

Only one FR is kept per detected failure: rediscoveries [2] are not recorded. In other words, if several FRs, caused by the same fault, are drawn up by different testing groups only, one FR is entered into the data base: generally the first one to be reported. In fact an FR is both a *failure report* and a *fault (or a correction) report* since it also contains the information on the fault that caused the failure.

Only software failures that can be simulated on the prototype are recorded: detected failures which are not reproducible are returned to the original site for more information.

Two software versions of the TROPICO-R switching system have been developed: a 1500 and a 4096 subscribers version. The failure data that have been analysed correspond to the former version. The software volume is about 300K written in Assembly language.

3.3. TROPICO R failure data

The cumulative number of failures per periods of ten days (units of time) are indicated in figure 3: data corresponding to the validation test go from time unit 1 up to 30, the field trial test goes from time unit 31 up to 42. The validation test lasted about ten months and resulted in 297 corrections. The field trial was performed for four months and led to 55 corrections whereas during the first year of operation 108 corrections were achieved.

Time corresponds to calendar time excluding vacation periods. Even though execution time seems more suitable than the calendar time [23], it was not easy in our case to obtain this information* .

---

* The software of the TROPICO-R is divided into several implementation modules. Some modules (such as the Operational System modules) are more executed than other (e.g. the application modules). As the TROPICO-R is a distributed system, it is very difficult to evaluate the CPU time expend by the software system. Accordingly it is useful to adopt calendar time as the time basis for the reliability evaluation. During the validation phase there was only one prototype operating all day long, so we can consider calendar time as a good time basis.

4. EVALUATION OF TROPICO-R RELIABILITY

The method of evaluation proposed begins with the analysis of the trend of the failure data before determinating which reliability model will be used during the reliability growth analysis.

4.1. Application of the Laplace trend test

The Laplace test is first applied to the whole set of data, the results are shown in figure 4.

Time unit	Validation	Time unit	Field trial	Time unit	Operation
1	7	31	301	43	356
2	8	32	302	44	367
3	36	33	310	45	373
4	45	34	317	46	373
5	60	35	319	47	378
6	74	36	323	48	381
7	82	37	324	49	383
8	98	38	338	50	384
9	106	39	342	51	384
10	115	40	345	52	387
11	120	41	350	53	387
12	134	42	352	54	387
13	139	-	-	55	388
14	142	-	-	56	393
15	145	-	-	57	398
16	153	-	-	58	400
17	167	-	-	59	407
18	174	-	-	60	413
19	183	-	-	61	414
20	196	-	-	62	417
21	200	-	-	63	419
22	214	-	-	64	420
23	223	-	-	65	429
24	246	-	-	66	440
25	257	-	-	67	443
26	277	-	-	68	448
27	283	-	-	69	454
28	286	-	-	70	456
29	292	-	-	71	456
30	297	-	-	72	456
				73	457
				74	458
				75	459
				76	459
				77	459
				78	460
				79	460
				80	460
				81	461

Figure 3 - Cumulative number of failures per periods of 10 days

This figure gives the Laplace factor u(k) v.s. k, the number of time unit, and where a high global reliability growth can be noticed at the end of the observation period.

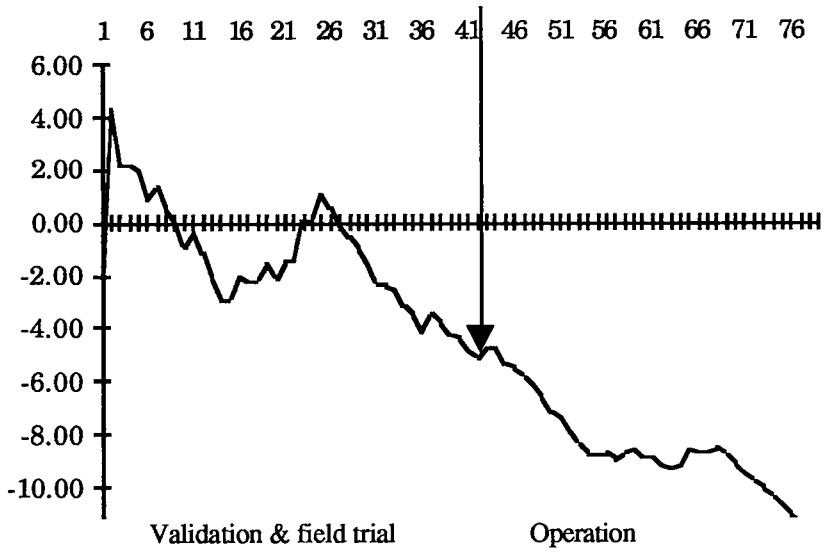


Figure 4 - Result of Laplace trend test for the whole set of data.

The results obtained separately for each phase (using data pertaining specifically to the phase considered), are shown in figure 5, and presented in a chronological order.

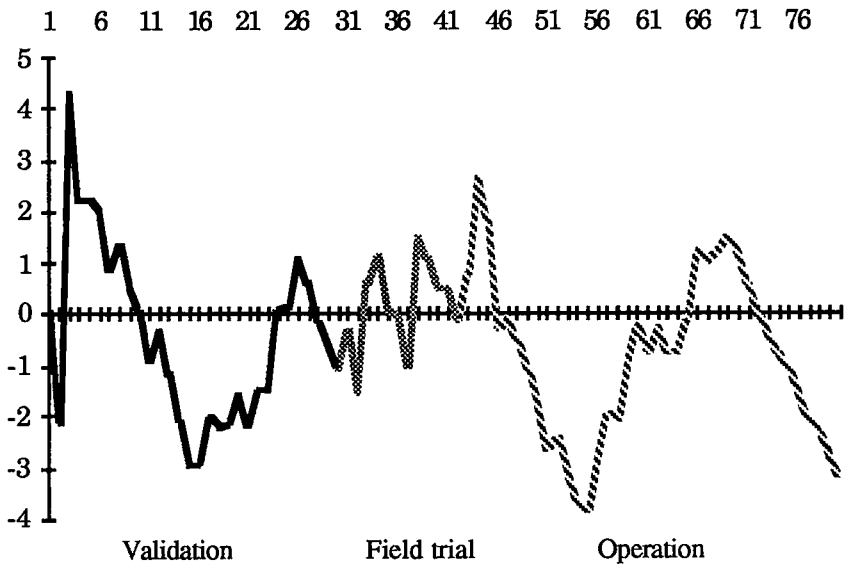


Figure 5 - Results of Laplace trend test for each phase

It is interesting to comment both figures at the same time:

- at the beginning of the validation, a reliability decay took place, as a result of the correction of 28 faults during the third unit of time whereas only 8 faults were removed during the first two time units and 24 during the next two time units; applying the trend test without considering the data belonging to the three first time units leads to a negative Laplace factor for the whole period (however the shape of the u(k) curve is not changed),
- the local reliability decay from k=14 to k=25, was induced by the changes in nature of the tests within the validation phase: this period corresponds to the application of the quality and the performance tests after the functional tests in the previous period; this decay is due to their dynamic nature (traffic simulation) which have activated new parts of the program,
- transitions from validation to field trial and from field trial to operation did not give rise to a reliability discontinuity: reliability was still improving,
- figure 3 indicates that from k=55 up to k=70 reliability growth tends to diminish: u(k) is almost constant, suggesting a local reliability decay; this behavior is reinforced when considering the trend results obtained for operational data only in figure 4 where reliability decay is more evident; from k=70 this trend is reversed; this failure behavior is directly related to the number of installed exchanges over the periods considered (see figure 6), during which about twelve exchanges have been installed and the number of failures reported by the users increased; by time unit 70, the 4096 version has been released and no new 1500 version has been installed, which corresponds to the period of local reliability growth (less failures reported).

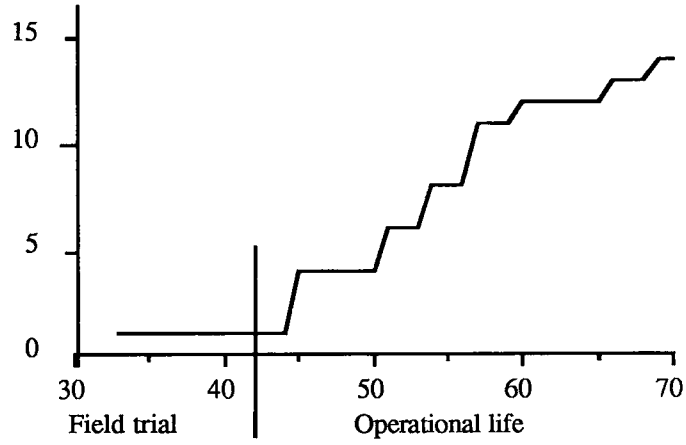


Figure 6 - Number of installed sites per unit of time

These results show that the systems installed have reached a sufficient level of quality after the new software releases which incorporate the necessary corrections. It is worth noting that, for the same calendar time used as reference in the sequel, the increase of the running software copies lengthens the software execution time. Reliability decrease does not necessarily mean that the software is less reliable, however it implies an increase in the maintenance effort that must be considered for maintenance planning.

4.2. Model application

The results of the trend analysis evidence changes in the trend over the considered phases and two periods of reliability decay have been identified. These results will guide the model application stage: (i) the exponential model will not be used given its inability to deal with reliability decay [6], (ii) only S-Shaped model (**SS**) will be applied to follow up the number of detected failures. The hyperexponential model (**HE**) will be applied at the end of operational life in order to estimate the residual failure rate of the software.

4.2.1. Model application to the whole data

Let us apply the SS model to validation data: data from k=1 to k=8 are used to estimate the model parameters and predict the cumulative number of failures. Results are given in figure 7 where N(i) denotes the observed cumulative number of failures up to unit time i and H(i) the estimated one. Predictions are very good up to k= 15 following which they rapidly become extremely inaccurate. Indeed k=14 corresponds to a type 2 inflexion point, the model cannot predict this change and, as a result, it continues to assume reliability growth underestimating the cumulative number of failures.

The same thing can be done with operational data: the cumulative number of failures after k=55 is more and more underestimated (figure 8).

These two model applications show that the S-Shaped model cannot follow the modifications in trend due to type 2 inflexion points. According to § 2.4, data will be partitioned into stages.

Data corresponding to validation and field trial are thus divided into two stages:

S1: data up to k=14 for which u(k) is continuously decreasing.

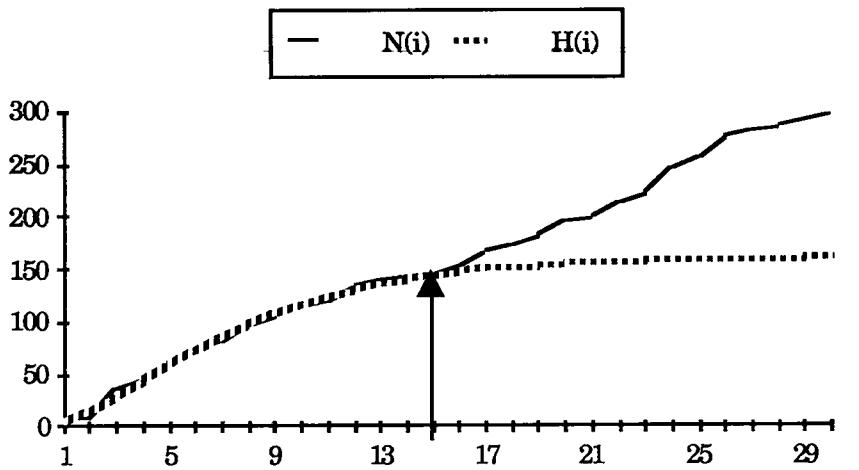


Figure 7: Application of the S-Shaped model to validation data.

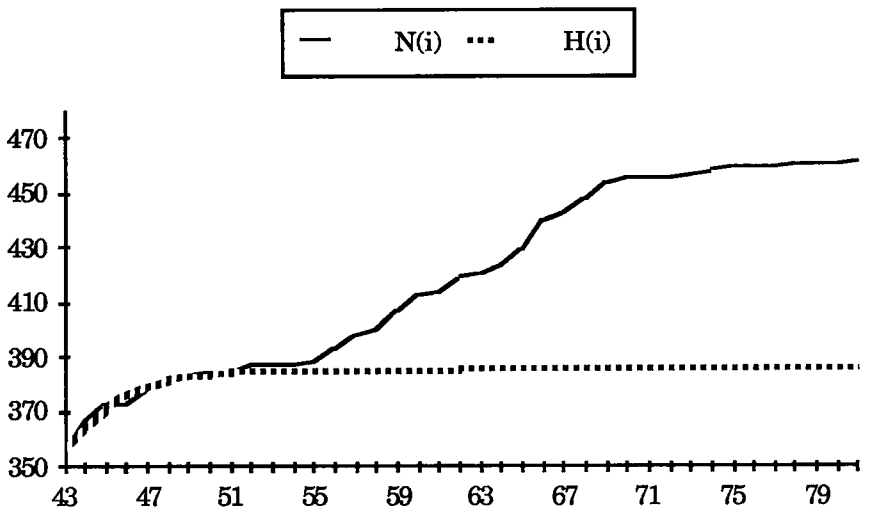


Figure 8: Application of the S-Shaped model to operational data.

S2: data counted from k=15 up to the end of field trial testing, for which u(k) is continuously increasing and then continuously decreasing.

Data corresponding to operation are also divided into two stages:

S3: data from k=43 up to k=54 for which u(k) is continuously decreasing.

S4: data counted from k=55 up to the end of the observation period, for which u(k) is continuously increasing and then continuously decreasing.

Reliability growth models will be applied according to these stages. Previous predictions for S1 and S3 remain valid.

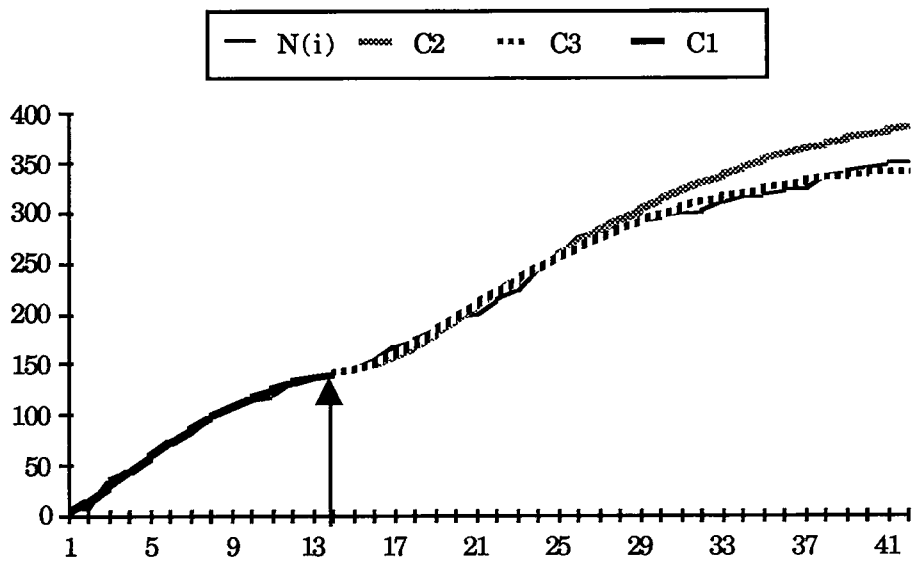
4.2.2 Validation and field trial phases

Figure 9 shows the cumulative number of failures estimated by the S-Shaped model (C1, C2, C3) and the cumulative number of observed failures N(i), where:

C1 prediction for S1 achieved from the failure data recorded over the first eight units of time of S1, k=1 to k=8,

C2 prediction for S2 performed using the failure data recorded up to 2 time units after the type 1 inflexion point which corresponds to time unit 25 (i.e., data from k=15 up to k=27 of S2),

C3 prediction for S2 performed using data including 4 time units after the inflexion point (i.e., data from time unit 15 up to 29 of S2).



	$E_{9,14}$	$E_{28,42}$	$E_{30,42}$
C1	2.6		
C2		28.4	31.2
C3			5.8

Figure 9 -Validation and field trial prediction using the Laplace test results.

The accuracy of the prediction is given by the mean prediction errors which are indicated by the table of figure 9. C1 and C3 have low residue whereas C2 has a relatively high residue.

The last prediction (C3) yields remarkable results over approximately 4 months (13 units of time of ten days each).

How to use these results during validation?

From data corresponding to the first two months and a half of validation (k=1 to 8), curve C1 indicates that if the same kind of tests is continued (i.e. functional tests) these tests will be inefficient from the 12th or 13th unit of time (one month later) since H(i) will reach a stable value. In other words, the number of detected failures per unit of time will reach an extremely low value at this stage, the designer can thus plan to change the type of tests to be applied the following month.

From time unit 14, quality and performance tests were applied leading to a reliability decrease. Since a change in the tendency is observed, prediction will not be very accurate. Some time is therefore needed prior to applying any reliability model. Indeed, these models can only be applied again when reliability growth is noticed, that is, in our particular case, after the next inflexion point.
Curves C2 and C3 indicate that the best predictions are obtained when applying models 4 units of time after tendency change.
The shape of H(i) is more interesting than the precise values of the figures: both curves show that for k=41 (date scheduled for software release), H(i) will begin to be stabilized, meaning thus that the software will not be in a stable state at the beginning of the operational life. This was confirmed later.

4.2.3 Operational life phase

When the software is in operation, two kinds of measures are interesting: **the number of failures** over all the installed sites in order to estimate the maintenance effort to be made during the operational life and **the failure rate of one site** in order to evaluate the whole system failure rate (i.e., hardware and software). These two measures are successively addressed in the sequel.

4.2.3.1 Maintenance effort prediction

The maintenance effort can be estimated through the evaluation of the number of corrections to be performed on the software for the next period of time. Since the RFs are at the same time failure and correction report, the number of corrections is estimated through the application of a "reliability growth model" to the collected data over all the installed sites, i.e. the corrections achieved on the software product.

The prediction of the cumulative number of failures applied to the operational life failure data is given in Figure 10 where (C4, C5, C6) are estimated using the SS model and N(i) the cumulative number of observed failures (i.e., achieved corrections):

C4 prediction for S3 based on the first eight units of time of S3, k=43 to k=50,

C5 prediction for S4 carried out from the failure data recorded from k=55 to k=73, that is, 2 time units following the type 1 inflexion point (k=71, figure 5),

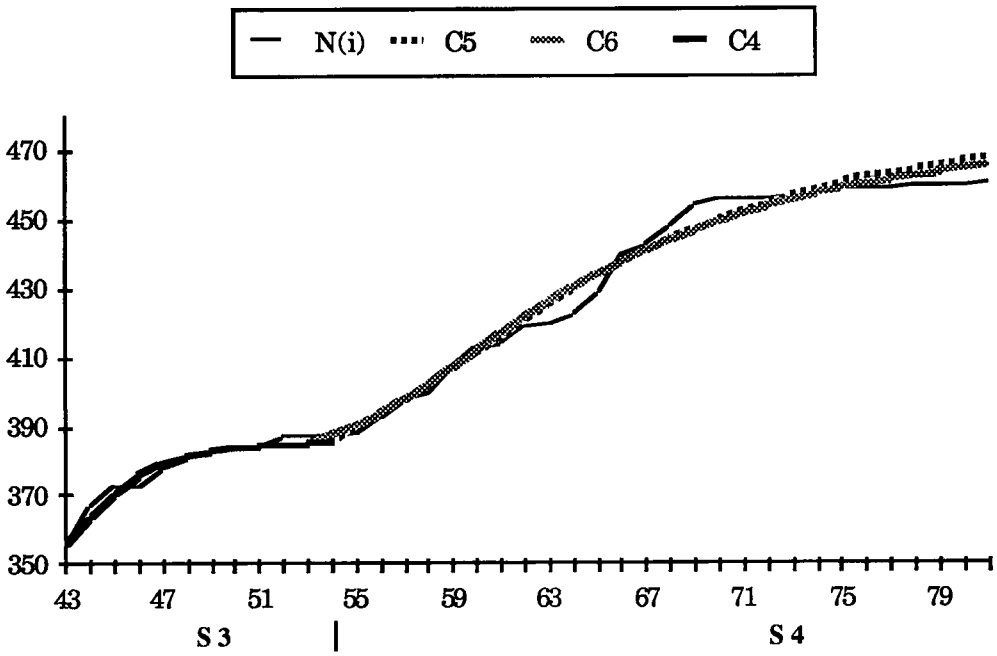
C6 prediction for S4 carried out from the failure data recorded from k=55 to k=75, that is, 4 time units following the type 1 inflexion point.

C5 and C6 are very close to each other and the difference in the residue is not significant. Both show that from the 75th unit of time H(i) is almost linear and that the hyperexponential model can be applied from this stage in order to predict the residual failure rate.

Application of the SS model until time unit 90 (three months after the considered period) indicates that for the whole installed sites, the expected number of corrections to be performed is about 2 corrections the next month and 1 correction per month for the next two months, i.e., 4 corrections during the next quarter. This estimation was confirmed later.

4.2.3.2 Failure rate and mean time between failures in operational life

In order to evaluate the failure rate and the mean time between failures, the random variable interval between failures has to be considered.



	$E_{51,55}$	$E_{74,81}$	$E_{76,81}$
C4	1.8		
C5		4.3	5.3
C6			3.5

Figure 10 -Operational prediction using the Laplace test results.

The S-Shaped model was applied using data gathered in an increasing number of sites in order to have an idea of the number of corrections to be performed on the software product. When the MTTF is addressed, the user is interested in the reliability of one site using all the collected data. Data have thus been modified so as to integrate the number of installed sites. Figure 11 gives, from left to right, the observed intervals between failures (in days) for one site (an average site) during the last months of operation where 34 corrections took place.

14	14	0*	0	14	14	42	0	0	14	0	0
42	42	14	0	90	15	0	15	0	120	15	
90	0	0	0	90	30	420	180	90	555	480	

* k successive zeros mean that (k+1) failures took place in the same day.

Figure 11 - Time (in days) between failures during the last months of operation.

Laplace test

The Laplace test is then applied to this subset (the considered random variable is the time between failures). The results are given in figure 12 and indicate an almost steady reliability at the beginning of this period (with u(k) oscillating between +2 denoting a low reliability decay and -2 denoting a low reliability growth) followed by a relative reliability growth. The time axis is added to the figure in order to show the correspondence between the failures and their occurrence time, this figure is in accordance with figure 5 (considering results for k = 65 to k = 71).

The growth of reliability at the end is mainly due to the last five observations for which the mean observed time between failures is about 345 days for an average site whereas it was about 24 for the previous five observations. These intervals are 23 days and 1.6 days respectively for the 15 installed sites.

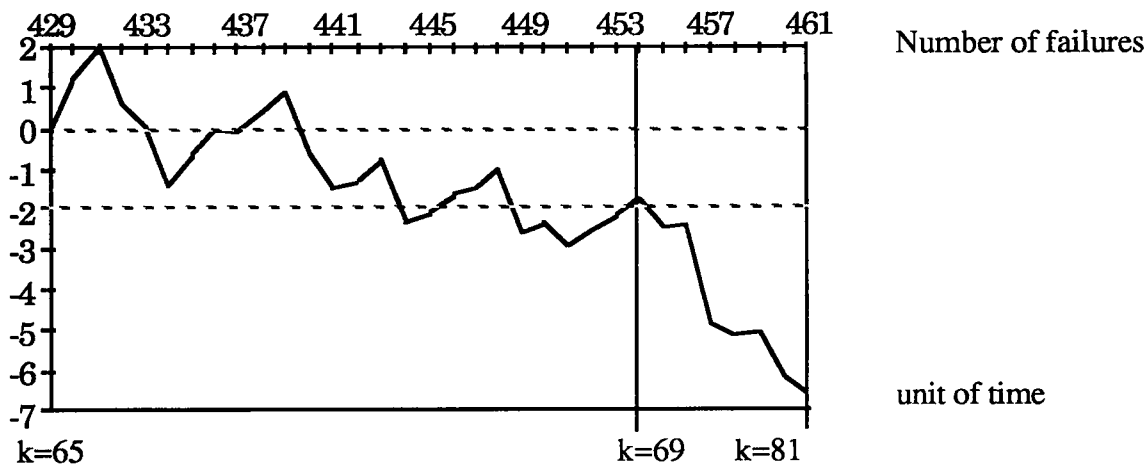


Figure 12 - Laplace factor for the last months of operation.

HE model application

The results of the HE model application to the considered period are summarized in figure 13 where the prediction is done step by step: data from i=441 to i=j-1 are used to estimate model parameters and $MTTF_j$. It can be seen that the model gives very pessimistic (and bad) results for the last five data (as prescribed by the Laplace test results). The residue goes from 0.65 when considering these data to 0.10 without these data.

The estimated residual failure rate λ_s is 3.4 10^{-4}/h (8.3 10^{-3}/day), i.e. one failure every 4 months per site. This time interval is much lower than the one observed at the end of the considered

period (which corresponds to about one failure every 345 days, for an average site during the last five data).

Considering the 15 installed sites, this means 15 failures per 4 months for all the sites or one failure every eight days, to be compared to one failure every 23 days.

i	t_i	$MTTF_i$
441	42	23.35
442	14	38.52
443	0	28.77
444	90	18.77
445	15	36.05
446	0	32.23
447	15	27.23
448	0	25.56
449	120	22.41
450	15	32.87
451	90	31.16
452	0	36.22
453	0	33.38
454	0	30.91
455	90	28.77
456	30	32.71
457	420	32.54
458	180	54.78
459	90	61.64
460	555	62.81
461	480	88.03

$R_{441,456} = 0.10$ $R_{441,461} = 0.65$

$\lambda_S = 8.3\ 10^{-3}$/day

Figure 13- Results of the HE model application.

The estimated residual failure rate without taking into account the last data is about $3\ 10^{-2}$/day (i.e., $1.25\ 10^{-3}$/h). Although in accordance with the observed one during this period it may seem very high. One way to use these results is as follows: the residual failure rate may be considered very high, if this is so the users may ask the designer to enhance the software reliability, the latter has thus to develop new test sets, these tests can either be run in the laboratory or on some installed sites in parallel with normal system utilization.

It can be noticed that the HE model experiences great difficulties in following a sharp and sudden reliability growth: it underestimates the last time between failures and over-estimates the

residual failure rate. In order to overcome these difficulties, data will be partitioned into two sets as for the SS model:

S5: data from 427 up to 447 for which reliability growth is difficult to ascertain.

S6: data counted from 448 for which reliability growth is clear.

Application of the HE model to S6

The results obtained when using data from 448 to 453 to estimate the parameters of the model for the first step (figure 14) are better than the previous one: prevision for the last observations are better and the residual failure rate seems more realistic.

i	t_i	$MTTF_i$
454	0	37.50
455	90	32.14
456	30	39.37
457	420	38.33
458	180	323.89
459	90	279.01
460	555	216.17
461	480	298.43

$R_{454,456} = 0.31$

$\lambda_S = 2.98\ 10^{-3}$/day

Figure 14 - Results of the HE model application to S6.

The residual failure rate is about $1.3\ 10^{-4}$/h ($2.98\ 10^{-3}$/day) for one site, i.e., one failure every 336 days which is more in keeping with the time between failures observed at the end of the data set and later also.This failure rate is quite high when compared to the hardware failure rate which is about $4\ 10^{-6}$/h. Indeed, the hardware is fault-tolerant and the failure rate corresponds to failures leading to system unavailability, whereas the software is not fault-tolerant and all the consequences of the failures on the delivered service are included in this figure. It is worth noting that most of the software failures do not lead to system unavailability or call loss. Collected data have thus to be partitioned into subsets according to failure consequences on the delivered service, the HE model will be applied to each of the subsets in order to evaluate the contribution of the software to the whole system unavailability. To do this a thorough analysis of the failure reports is needed in order to see what are the effects of each failure; this analysis has not been undertaken yet. Application of the same approach

to software failure data concerning another switching system [16] showed that the failure rate leading to system unavailability is about 5% of the global software failure rate.

4.3. Practical recommendations

What precedes leads to the following practical recommendations :

- depending on the nature of the collected data (concerning failures or corrections or both of them) and the objectives fixed, two kinds of dependability measures can be derived :
    - the number of corrections to be performed during test or operational life,
    - the failure rate of the software,
- the evaluation method does not depend on the evaluated measure and can be summarized by the following steps :
    1. apply the Laplace test trend to the collected data,
    2. identify the periods of reliability growth and the periods of reliability decay,
    3. if a reliability growth trend is observed then apply an appropriate model to forecast dependability measures as long as the model assumptions are verified,
    4. if local reliability decay is noticed then apply an S-Shaped model to the data set including the inflexion point, the predictions stand a good chance of being more accurate,
    5. for the next collected data, proceed with the analysis in the same way, from step 1 to 4.

5. CONCLUSIONS

Software reliability growth models constitute an important aid tool for test/maintenance planning and reliability evaluation when used with reliability trend tests.

Two models, that are, the hyperexponential and the S-shaped models, have been applied to the failure data of the TROPICO-R telephone switching system. These models have been analyzed according to their predictive capabilities.

The results obtained show that:

- the trend test is a major tool for guiding the partitioning of failure data according to the assumptions of model reliability growth; it also indicates the unit of time from which the occurrence of future failures can be predicted more accurately,
- the prediction approach proposed for the validation and the field trial phases yields good results over a time period of a few months, showing that reliability modeling constitutes a major aid tool for test/maintenance planning and follow up,
- the residual failure rate prediction given by the hyperexponential model is in accordance with the observed failure data at the end of the considered period.

Unfortunately, since the TROPICO-R 1500 version is no more produced, failure data collection is no more achieved for this system and it is not possible to check the accuracy of the predictions. Our current work concerns the TROPICO-R 4096 version and is directed towards:

- the application of reliability growth models to hardware design failures recorded during the test and operational phases in order to have a system reliability model to guide test/maintenance planning and follow up,
- the analysis of software components failure data in order to deduce system dependability from the components measured reliability.

ANNEX

LAPLACE FACTOR DERIVATION

The expression of the Laplace factor u(k) when the random variable considered is the cumulative number of events (failures for our purpose) is derived as indicated in [7, p. 54].

Let us consider k time intervals of equal length h and let :

Ni be the random variable that represents the number of events occurred in the time interval [(i-1)h,ih], i=1,...,k.

ni the realization of Ni, i.e., the number of events occurred in the time interval [(i-1)h,ih], i=1,...,k,

The occurrence of the events follow a non-homogeneous Poisson process whose occurrence rate is given by the expression:

$$\lambda(t) = e^{a+bt} \tag{A1}$$

If b=0 the Poisson process becomes homogeneous and the occurrence rate is time independent. If we observe k time intervals, the likelihood is :

$$L(k) = \prod_{i=1}^{k} P\{N_i = n_i\} = \prod_{i=1}^{k} \frac{\left[\int_{(i-1)h}^{ih} \lambda(u)\, du\right]^{n_i}}{n_i!} \exp\left(-\int_{(i-1)h}^{ih} \lambda(u)du\right) \tag{A2}$$

For the occurrence rate (A1), the likelihood becomes:

$$L(k) = \frac{(e^{bh}-1)^N\, e^{aN+bh\sum_{i=1}^{k}(i-1)n_i}}{b^N \prod_{i=1}^{k} n_i!} \exp\left[\frac{e^a\,(1-e^{bkh})}{b}\right], \text{ where } N=\sum_{i=1}^{k} n_i \tag{A3}$$

Thus the conditional probability density function of the observations, given that N events have occurred in k time intervals, is obtained dividing expression (A3) by the probability:

$$P(k) = P\{\sum_{i=1}^{k} n_i = N\} = \frac{\left[\int_0^{kh} \lambda(u)\,du\right]^N}{N!} \exp\left(-\int_0^{kh} \lambda(u)du\right)$$

The result is given by (A4) and only depends on the b variable:

$$\frac{L(k)}{P(k)} = \frac{N!\,(e^{bh}-1)^N\, e^{bh\sum_{i=1}^{k}(i-1)n_i}}{(e^{bkh}-1)^N \prod_{i=1}^{k} n_i!} \qquad \text{(A4)}$$

The conditional log likelihood $\mathcal{L}(b)$ is, from (A4):

$$\mathcal{L}(b) = \log N! - \log \prod_{i=1}^{k} n_i! + bh\sum_{i=1}^{k}(i-1)n_i + N\left[\log(e^{bh}-1) - \log(e^{bkh}-1)\right]$$

so that:

$$\mathcal{L}'(b) = \frac{d\mathcal{L}(b)}{db} = \begin{cases} h\sum_{i=1}^{k}(i-1)n_i + N\left[\frac{he^{bh}}{e^{bh}-1} - \frac{khe^{bkh}}{e^{bkh}-1}\right] & , b\neq 0 \\ h\sum_{i=1}^{k}(i-1)n_i - \frac{Nh(k-1)}{2} & , b=0 \end{cases} \qquad \text{(A5)}$$

and the information function $\mathcal{I}(b)$, is:

$$\mathcal{I}(b) = E\{-\mathcal{L}''(b)\} = \begin{cases} Nh^2\left[\frac{e^{bh}}{(e^{bh}-1)^2} - \frac{k^2e^{bkh}}{(e^{bkh}-1)^2}\right] & , b\neq 0 \\ \frac{Nh^2(k^2-1)}{12} & , b=0 \end{cases} \qquad \text{(A6)}$$

Under the null hypothesis b=0 (constant occurrence rate), the statistics:

$$u(k) = \frac{\mathcal{L}'(0)}{\sqrt{\mathcal{I}(0)}} = \frac{\frac{\sum_{i=1}^{k}(i-1)n_i}{N} - \frac{(k-1)}{2}}{\sqrt{\frac{(k^2-1)}{12N}}}$$

is approximately normal distributed with zero mean and unit variance.

Positive value of u(k) means that the considered statistics is above the mean and therefore indicates b>0, i.e. an occurrence rate increasing with time. On the other hand, negative value of u(k) suggests b<0 (a decreasing failure rate).

REFERENCES

1 A.A.Abdel-Ghaly, P.Y.Chan, B.Littlewood, "Evaluation of Competing Software Reliability Predictions", *IEEE Trans. on Soft. Eng.*, vol SE-12, n° 9, Sept. 1986.

2 E.N.Adams, "Minimizing cost impact of software defects", IBM Research division, Rep.RC 8228 (35669), April 11, 1980.

3 H.Ascher, H.Feingold, "Application of Laplace's Test to Repairable System Reliability", Proc. *1st International Conference on Reliability and Maintainability*, Paris, 19-23 June 1978, pp. 219-225.

4 H.Ascher, H.Feingold, *Repairable Systems Reliability: Modeling, Inference, Misconceptions and Their Causes*, Lecture notes in statistics, Volume 7, 1984.

5 V.R.Basili, D.M.Weiss, "Methodology for Collecting Valid Software Engineering Data", *IEEE Trans. on Soft. Eng.* vol. SE-10, n °6, Nov. 84, pp. 728-738.

6 M.R.Bastos Martini, K.Kanoun, J.Moreira de Souza, "Software Reliability Evaluation of the TROPICO-R Switching System", LAAS Report n° 88-336, Nov. 1988.
To appear in *IEEE Trans. on Reliability*, June 1990.

7 D.R.Cox, P.A.W.Lewis, *The Statistical Analysis of Series of Events*, Chapman & Hall, 1978.

8 P.A.Currit, M.Dyer & H.D.Mills, "Certifying the Reliability of Software", *IEEE Trans. on Soft. Eng.*, Vol SE-12, N° 1 Jan. 1986, pp. 3-11.

9 J.Favrot, C.Lamy, F.Michel, "Reliability evaluation of Programs for Irradiated Nuclear Fuel Testing during the Qualification phase", *Technique et Science Informatiques*, Vol. 4, n° 2, 1985, pp. 209-223, in French.

10 K.Fukushima, Y.Kishida, "Estimation of the Bug Curve Using Experimental Regression Analysis of Office Automation Equipment Software", *5th Int. Conf on Reliability and Maintainability*, Biarritz 6-10, Oct 86, pp 87-91.

11 R.L.Glass: "Persistent Software Errors", *IEEE Trans. on Soft. Eng.*, Vol SE-7, N°2 March 1981.

12 A.L.Goel, K.Okumoto, "Time-dependent Error-detection Rate Model for Software and other Performance Measures", *IEEE Transactions on Reliability*, vol.R-28, n°3, Aug 79, pp 206-211.

13 Z.Jelinski, P.B.Moranda: "Software Reliability Research", *Statistical Computer Performance Evaluation*, W.Freiberger(ed), Academic Press 1972, pp. 465-484.

14 R.K.Iyer, S.E.Butner, E.J.Mac Clusky, "A statistical Failure Load Relationship : Results of a multicomputer study", *IEEE Transactions on Computers*, vol.C-31, n°5, 1982, pp 696-706.

15 K.Kanoun, T.Sabourin, "Software Dependability of a Telephone Switching System", Proc. *17th IEEE Int. Symp. on Fault-Tolerant Comp. (FTCS-17)*, Pittsburgh, Pennsylvania, 6-8 July, 1987, pp. 236-241.

16 K.Kanoun, J.C.Laprie, T.Sabourin, "A Method for Software Reliability Growth Analysis and Assessment", Proc. of *Le génie logiciel & ses Applications*, Toulouse, France, 5-9 Dec., 1988, pp. 859-878.

17 K.Kanoun, "Software Dependability Growth characterization, modeling and evaluation", Doctorat ès-Sciences thesis, Institut National polytechnique de Toulouse, LAAS report N° 89-320, Sept. 1989, in French.

18 J.C.Laprie, "Dependability Evaluation of Software Systems in Operation", *IEEE Trans. on Soft. Eng.* Vol. SE-10, N° 6, Nov. 1984, pp. 701-714.

19 J.C. Laprie : Dependability: a unifying concept for reliable computing and fault tolerance, *Resilient Computing Systems,* Vol. 2, T.Anderson Editor, Collins et Wiley, 1987; LAAS report n° 86.357, Dec. 1986.

20 J.C.Laprie, "Towards an X-ware Reliability Theory", *TSI*, Vol. 7, n° 3, 1988, pp. 315-330; in French (also available in English as LAAS report).

21 B. Littlewood, "Stochastic Reliability-Growth: A Model for Fault-Removal in Computer-Programs and Hardware-Designs", *IEEE Transactions on Reliability*, vol.R-30, n° 4, Oct 81, pp. 313-320.

22 J.D.Musa, "Software Reliability data", Data and Analysis Centre for Software Rome Air Development Centre (RADC) Rome, NY, 1979.

23 J.D.Musa et al., *Software Reliability Measurement, Prediction, Application*, Mc Graw-Hill International Edition, 1987.

24 P.M.Nagel, J.A.Skrivan, "Software Reliability: Repetitive Run Experimentation and Modeling, BCS 98124, Boeing Computer Service Company, Seattle, Washington, Feb. 1982.

25 M.Ohba, S.Yamada, "S-Shaped Software Reliability Growth Models", proc. *4 th International Conference on Reliability and Maintainability*, Perros Guirec, France, 1984, pp. 430-436.

26 Tohma et al., "Structural Approach to the Estimation of the Number of Residual software faults Based on the Hyper-geometric Distribution", *IEEE Trans. on Software Eng.*, vol. 15, N° 3, March 1989, pp. 345-355.

27 R.Troy, Y.Romain, "A Statistical Methodology for the Study of the Software Failure Process and its Application to the ARGOS Centre", *IEEE Trans. on Soft. Eng.*, vol SE-12, n° 9, Sept. 1986, pp. 968-978.

28 S.Yamada, M.Ohba, "S-Shaped Reliability Growth Modeling for Software Error Detection", *IEEE Transactions on Reliability.*, Vol. R-32, n° 5, Dec. 1983, pp.475-478.

29 B.Vianna, "R&D at TELEBRAS-CPqD: The TROPICO System", *Int.Conf. on Communications (ICC 88)*, Philadelphia, June 1988, pp. 622-626.

30 B.Vianna, E.C.Cunha, F.F.Boin, "Hardware Quality Control in the TROPICO System", *Int. Conf. on Communications (ICC 88)*, Philadelphia, June 1988, pp. 632-636.

A Method for Software Reliability Analysis and Prediction Application to the TROPICO-R Switching System

Karama Kanoun, Marta Rettelbusch de Bastos Martini, and Jorge Moreira de Souza

Abstract— **It is well-known that no particular reliability growth model is "superior" in predicting software behavior of any software system under any circumstances. The work presented in this paper allows the already existing models to give better predictions since they are applied to data displaying trends in accordance with their assumptions. The method proposed is primarily based on the analysis of the trend exhibited by the data collected on the program (which is determined by reliability growth tests). Reliability data are then partitioned according to the trend and two types of reliability growth models can be applied: 1) when the data exhibit reliability decrease followed by reliability growth, an S-shaped model can be applied and 2) in case of reliability growth most of the other existing reliability growth models can be applied. The hyperexponential model allows prediction of the software residual failure rate in operation, this failure rate being used as a qualification index for the software product.The proposed method is illustrated through its application to the Brazilian Electronic Switching System (ESS), TROPICO-R.**

Index Terms—**Reliability growth, reliability prediction, software reliability, software reliability modeling, trend analysis.**

I. Introduction

QUALITY control is one of the key factors in the success of a project having to meet stringent reliability requirements. Among quality control techniques, the analysis of failure data and quantitative evaluation based on reliability models are of prime importance.

Evaluation of the reliability of computer based systems requires paying particular attention to the hardware, the software and the system as a whole.

Software reliability is evaluated so that:

- in terms of the product itself, it satisfies the requirements of system specifications (if such specifications exist);
- in terms of production process, the development can be managed to obtain a "dependable" software within the allowed delays.

Management of the development/maintenance requires realistic time schedules for life cycle phases (e.g., test, validation, and maintenance phases). To do this, it is necessary to evaluate continually the reliability attained following the debugging effort in order to review the initial planning. Reliability techniques allow assessment of resource allocation, quantification of the development/maintenance efficiency, and monitoring of reliability growth so as to make sure that software reaches a sufficient level of quality within schedules and planned costs.

Manuscript received December 20, 1989; revised November 28, 1990 Recommended by F. B Bastani

K Kanoun is with LAAS-CNRS, 7 Avenue du Colonel Roche, 31400 Toulouse, France

M. R. Bastos Martini and J. Moreira de Souza are with CPqD-TELEBRAS, CP 1579, 13 100 Campinas-SP, Brazil

IEEE Log Number 9042424.

During operational life it is important to be confident in the service delivered by the system and to show quantitatively that the required level of software reliability has been attained (if this level has been defined). Software quality objectives can be stated by means of usual reliability measures such as mean time to failure or failure rate; reliability modeling and evaluation help estimate these quality measures from the observed failure data.

In this paper an evaluation method is proposed for software reliability analysis and prediction. It is illustrated by applying it to TROPICO-R ESS. It is shown 1) how reliability growth models can help manage development and maintenance processes, and 2) how these models can be employed to estimate the software failure rate in operational life.

The data regarding software failures detected in the TROPICO-R system were recorded in appropriate Failure Report (FR) sheets drawn up by the product development and follow-up engineers. Four hundred and sixty-one software FR's were established over 27 months including the last 2 phases of development (validation and field trial tests) and operational life (during which 15 sites were installed).

This paper is divided into four sections. Section II gives an idea about the work already published in the field. Section III deals with the evaluation method, more specifically 1) reliability growth tests (primarily Laplace test) and 2) the choice of evaluation models and the associated validation criteria. The following section describes the test environment and the failure data recorded on the TROPICO-R. The application of the models to the TROPICO-R failure data and their use as a tool for planning and reliability prediction are addressed in Section V. The Laplace factor expression is derived in Appendix A and Appendix B presents the reliability growth models used in Section V.

II. Related Work

Numerous papers deal with software reliability growth modeling and evaluation. However, papers following a global method are seldom found [16]. Methods of data collection and statistical analysis of data are given in [5], [25] and application examples can be found in [11], [14], or [29].

Theoretical aspects concerning reliability trend tests are studied in [3], [4], [7] and their application to real data can be found in [17]. Definition and application of the reliability growth models have often been discussed, see for example [1], [12], [16], [19], or [25]. Applications to concrete cases have seldom been reported since results are often confidential. However, several interesting applications of reliability growth models can be found in [2], [8], [9], [16], [24].

Other aspects concerning software reliability evaluation can be found in [20], [23], and [26], for example. The above list

of references is not exhaustive. It is only intended to show what has already been achieved in this field; note that numerous papers analyzed the data collected and published by Musa in [24].

III. Evaluation Method

The method of analysis and evaluation of the software reliability corresponds to the global method defined in [16] and [17]. It consists of two main stages:

- thorough analysis of the system and the data collected;
- reliability trend analysis and software reliability evaluation.

As in any reliability study, the first stage basically consists of gathering information on the system itself, the functions to be performed, the system structure and the utilization environment, the life cycle phases under study, the software correction, and the maintenance policies, the manner in which data are (or have been) collected. Qualitative data processing provides interesting information (e.g., the nature of faults, the consequence of their activation, their evolution along the considered period, distribution of faults among the different components). This stage is system related and has to be adapted to the nature of the studied system.

The second stage is more systematic and applicable to any software data. It is a two-step process involving:

- first, a preliminary data processing providing quantitative information on the evolution of reliability (i.e., trend analysis of collected data), before model applications;
- then, reliability growth model application; the selection of the reliability growth model depends to a great extent on 1) the objectives of the study as well as on 2) the nature of the trend displayed by the collected data.

This section focuses on the last stage and aims to extend the work presented in [15] and [16]. It will consist of:

- stating of the objectives of software reliability evaluation;
- adapting the Laplace test to the random variable: number of failures and introduction of the notion of local and global reliability growth;
- presenting the various types of reliability growth models and validation criteria;
- discussing the way in which reliability growth models can be applied in a predictive manner.

A. Reliability Evaluation Objectives

The objectives of reliability evaluation are generally expressed in terms of measures. When the software is in operation, two types of measures are useful:

a) from the user's viewpoint: the mean time to failure, MTTF (or failure rate of the next failure), and/or the expected software residual failure rate in operational life so as to evaluate the reliability of the whole system;
b) from the designer's viewpoint: first the expected number of failures among all installed systems and then the number of corrections to be introduced in the software to estimate the maintenance effort still needed.

When the software is under development, interesting measures are:

c) the evolution of the number of detected faults in the software in response to the test and debugging effort;
d) the expected number of failures for the following periods of time so as to plan the test effort and thus the time and the numerical importance of the test team.

It is worth noting that these objectives lead to considering two different random variables :

- for maintenance and test management and planning, the random variable will be the number of failures per unit of time;
- for reliability evaluation, the random variable "interfailure time" is more suited for estimating the MTTF of the software or its failure rate.

These two random variables are not independent: by knowing the interfailure times it is possible to obtain the number of failures per unit of time (the second form needs less precise data collection). Both of them are considered below.

B. Trend Analysis

Existing reliability growth models allow to model either reliability growth with a decreasing ROCOF (Rate of OCcurrence Of Failures) or reliability decrease prior to reliability growth, i.e., the number of failures per unit of time first increases and then decreases to zero (the ROCOF is bell-shaped); these are called S-shaped models [10], [30] because the curve representing the cumulative number of failures is S-shaped.

It is therefore very important to identify the trend (growth/decrease) in order to use the appropriate reliability growth models. Several trend tests can be used, either graphical or analytical [4], [17]; we selected the Laplace test, proposed in [3], [4], [7] and extended in [17]. This test was developed for the interfailure time random variable, when the random variable is the number of failures per unit of time, the expression of the Laplace test factor $u(k)$ at time unit k can be deduced as suggested in [7, p.54]. The details of the derivation of $u(k)$are presented in Appendix A. Then, this factor is given by

$$u(k) = \frac{c - m}{\sqrt{\frac{k^2-1}{12N(k)}}}. \qquad k = 2.\cdots.p$$

where

$$c = \frac{\sum_{i=1}^{k}(i-1)n_i}{N(k)}$$

$$m = \frac{k-1}{2}$$

p is the total number of units of time.

n_i is the number of failures during the ith time unit.

$N(k)$ is the cumulative number of failures up to the kth time unit

$$N(k) = \sum_{i=1}^{k} n_i$$

Positive values of $u(k)$ indicate a decrease of the software reliability whereas negative values indicate reliability growth [3]. The Laplace test can be used with confidence levels to accept or reject the hypothesis: no reliability trend versus reliability trend; however in our case it is used as an indicator for reliability growth or decrease; moreover it is adapted to identifying global and local trends in the following manner.

The factor $u(k)$ is evaluated among all the data collected up to the time unit since it indicates the global variation of reliability. It is worth noting that even in the presence of global reliability growth (decrease), local reliability decrease (growth) may occur.[1]

[1] Global: versus all data collected from the beginning of the observation period to the considered unit of time. Local: versus the data observed during some units of time just before the unit of time considered.

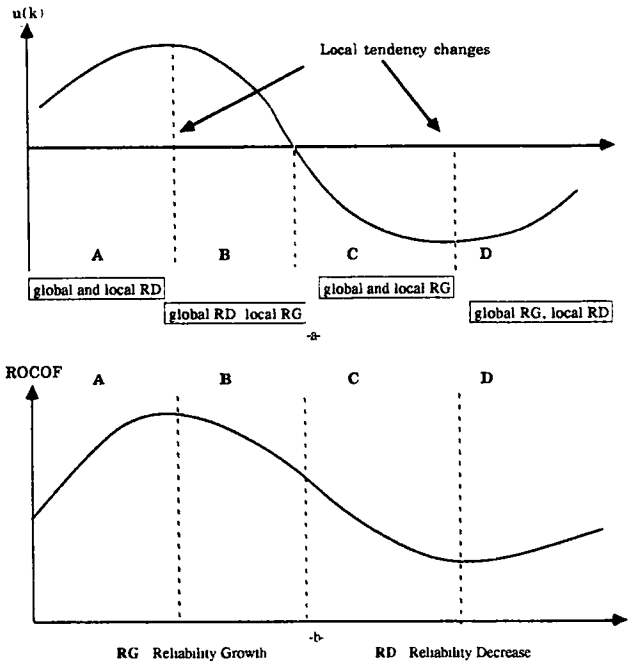


Fig. 1. Trend analysis using Laplace test and ROCOF evolution.

Local fluctuations can be detected by studying the variation of $u(k)$: when $u(k)$ is positive and tends to decrease, it suggests a decrease in the number of failures observed over the considered period which means that, locally, reliability tends to increase although a global decrease is observed.

This is summarized in Fig. 1, where the units of time during which the local tendency of $u(k)$ changes correspond to the inflexion region (called inflexion point) of the curve giving the cumulative number of failures.[2] The evolution of the ROCOF indicates local reliability growth or decrease only.

Two types of tendency changes (inflexion points) can be distinguished:

- type 1: corresponding to transition A-B, which is common and expected during testing and validation phases;
- type 2: corresponding to transition C-D, which denotes a reliability regression, i.e., an increasing number of detected failures per unit of time.

Reliability trend changes may result from a great variety of phenomena, such as:

- dependency of faults: some software faults can be masked by others, i.e., they cannot be activated as long as the latter are not removed [27];
- variation in time delay between the detection of an error and its removal; this is closely dependent on the nature of the activated faults: some faults are more difficult to identify than others and take longer time to be removed;
- variation in the utilization environment: the variation in the testing effort during debugging, change in test sets, addition of new sites during the operational life;
- slight specification changes.

These phenomena can act either separately or concurrently

[2]The curve was smoothed to show the main periods of time: some small (and very local) fluctuations due to the random nature of data may take place, these fluctuations are normal and will not be considered in the following. We only consider those trend changes which occur during a relatively long period of time.

C. Selection and Validation of Reliability Growth Models

The reliability growth model to be applied depends on the objectives of evaluation presented in Section III-A and on the nature of the trend. Objective **a** (evaluation of MTTF and residual failure rate) can be reached through a model allowing MTTF evaluation, such as the models of Jelinski–Moranda [13], Littlewood–Verral [22], or Keiller–Littlewood [18]. With respect to the residual failure rate expected in operational life, the *hyperexponential* model [17], [19], [21] is the only suitable model. In fact, most other existing models assume that the failure rate tends to zero when t tends to infinity (nil residual failure rate in operational life).

Objectives **b**, **c**, and **d** (test planning and maintenance effort) can be obtained through a model based on counting; in this case nonhomogeneous Poisson process models (NHPP) seem suitable. The advantage of the NHPP models is that they are relatively independent of many hypotheses necessary for the derivation of several published models, for instance, no relationship between failures and corrections is assumed.

Many NHPP models have been developed, allowing three types of reliability tendency to be modeled:

- decreasing ROCOF tending to zero when t tends to infinity (e.g., the exponential [12] or logarithmic models [25]);
- decreasing ROCOF tending to a nonzero value when t tends to infinity (allowing steady behavior in operational life to be evaluated): the hyperexponential model;
- ROCOF increasing prior to undergoing ROCOF decreasing; it can thus be applied to an observation period where the trend test has the form of the A-B-C period of Fig. 1.

The first two types of models yield better results when applied to data displaying reliability growth: that is, period C of Fig. 1. However, in Fig. 1(a), global reliability decrease for A and B is due to the data corresponding to period A; if the latter are not considered for evaluation purposes, period B will display local and global reliability growth. These models can be applied to period B if the data corresponding to period A are not taken into account. Nevertheless, applying the models in case of reliability growth only (periods B and C) leads to discarding data pertaining to A and D from any data set, therefore prediction is delayed until observing a period such as B or C is observed.

Model Validation: A model can be analyzed according to its retrodictive and predictive capabilities:

- *retrodictive capability*: the ability of a model to reproduce the observed behavior of the software,
- *predictive capability*: the extent to which the model, based on the observed data up to a given time, correctly predicts the failure behavior in the future.

This paper deals with the predictive features; retrodictive features concerning the TROPICO-R application are given in [6].

Prediction of the failure behavior after a given time interval is based on failure data observed in this interval. The predictive capability is evaluated in the following manner.

1) Use the failure data observed up to the time unit $i = j - 1$ to estimate the unknown parameters of the model (model calibration).
2) Evaluate $H(i)$, the expected cumulative number of failures up to unit time $i, i \geq j$.
3) Evaluate the residue, i.e., the *mean error* between the observed number of failures, $N(i)$, and that foreseen by the model, $H(i)$, for time units $i, j \leq i \leq p$:

$$E_{j,p} = \frac{\sum_{i=j}^{p} |N(i) - H(i)|}{p - j + 1}. \quad (1)$$

Operations 1 and 2 represent the so-called prediction system [1].

When the random variable is the interfailure time, operations 2 and 3 have to be adapted; in this case the residue is equal to: $(t_i\text{-MTTF}_i)$ where t_i is the observed time between failure $(i-1)$ and failure i, and MTTF_i the estimate. The *relative residue* $R_{j,n}$ will be evaluated:

$$R_{j,n} = \frac{\sum_{i=j}^{n} (t_i - \text{MTTF}_i)}{S_n - S_j}, \quad (2)$$

where S_j is the time of occurrence of failure j.

When the random variable is the interfailure time, the Kolmogorov–Smirnov distance [1], [7] can also be used as a test of goodness of fit of the considered model.

Equation (1) enables us to check whether the model yields results close to the observed ones; on the other hand, (2) permits us to see the bias in the evaluation: a positive residue (resp. negative) means that globally the model underestimates (resp. overestimates) interfailure time. These choices are mainly due to the very nature of the random variables: $N(i)$ or t_i.

D. How to Obtain Predictions of Future Reliability

In a predictive situation, statements have to be made regarding the future reliability of software, and we can only make use of the information available at that time. A trend test carried out on the available data helps choose the reliability growth model(s) to be applied and the subset of data to which this (or these) model(s) will be applied.

The models are applied as long as the environmental conditions remain significantly unchanged (changes in the testing strategy, specification changes, no new system installation . . .). In fact even in these situations, reliability decrease may be noticed. Initially, one can consider that it is due to a local random fluctuation and that reliability will increase some time in the near future: predictions are still made without partitioning data. If reliability keeps decreasing, one has to find out why and new predictions may be made by partitioning data into subsets according to the new trend displayed by the data.

If a significant change in the development or operational conditions takes place, great care is needed since local reliability trend changes may result, leading to erroneous predictions. New trend tests have to be carried out:

- if there is insufficient evidence that a different phase in the program's reliability evolution has been reached, application of reliability growth models can be continued;
- if there is an obvious "reliability decrease," reliability growth models application has to be stopped until a new reliability growth period is reached again; the observed failure data have to be partitioned according to the new trend.

1) Number of Models to Be Applied: With respect to the number of models to be applied, previous studies indicated that there is not a "universally best" model and suggest that we "try several models and examine the quality of prediction being obtained from each" and that even doing so, we "are not able to guarantee obtaining good predictions..." [1], [18]. During development, it is not always possible to apply several models, because of lack of time, experience, and analytical and practical tools. Usually people only apply one, two, or three models to their data. Analysis of the collected data and of the environmental conditions helps us understand the evolution of software reliability, and data partitioning into subsets helps us improve the quality of the predictions [15], [28].

2) Model Calibration: The model may be calibrated either after each new observed data (step-by-step) or periodically after observation of a given number of failures, say y, (y-step ahead). Step-by-step predictions seem more interesting. But, in practice, this may be not feasible, particularly in an industrial environment when the predictions are made by a team different from the development team and the collected data are not always available immediately. In operation, longer interfailure times allow step-by-step predictions.

E. Conclusion

The results of Laplace test application are interesting, as:

- they allow follow-up of the evolution of the reliability of the software;
- the inflexion points allow data to be partitioned into subsets (called stages) and reliability growth models to be applied to these stages enhance the quality of the estimation made; type 2 inflexion points constitute boundaries between these stages. In a predictive situation:
  - — an S-shaped model is applied to data including a type 1 inflexion point (reliability decrease followed by reliability growth);
  - — models corresponding to reliability growth can be applied to data from a type 1 inflexion point to a type 2 inflexion point (reliability growth region).

The idea of dividing data into subsets has been used in [15], [16], [17], and [28] (empirically, in the latter case); however, the Laplace test used in the preceding manner helps this partition: it tells you where to do it.

IV. TROPICO-R Test Environment and Failure Data

A. Test Program

The software test program for TROPICO-R is a four-step process: 1) unit test, 2) integration test, 3) validation test, and 4) field trial. The first three operations correspond to the test phases usually defined for the software life cycle. Field trial consists of testing a prototype in a real environment, which approaches the normal operating environment. It uses a system configuration (hardware and software) that has reached a sufficient level of quality after completing the laboratory testing program.

The description of the whole quality control program for TROPICO-R is given in [31] and [32]. The test program carried out during validation test and field trial is decomposed into four tests:

1) *Functional tests* verify the correct operation of all specified telephone, maintenance, and operation functions.
2) *Quality tests* ensure that a specified rate of call loss is not exceeded. Call loss may be due either to a failure in the software and/or hardware of the system or missing call resources (resource failure or all resources are already used). At the beginning of the test we ensure that all resources needed for calls are available so that any lost call can only be due to software and/or hardware failure.

3) *Performance tests* ensure compliance with the quality of service requirements under a nominal traffic.
4) *Overload tests* check the correct operation of traffic overload control mechanisms so that the calls accepted by the telephone exchange subject to this exceptional traffic comply with the specified quality of service.

B. Failure Report (FR)

Handling of failure data affecting the TROPICO-R product is achieved through use of an appropriate FR sheet containing the following:

- date of failure occurrence;
- origin of failure: the system configuration in which the failure was detected and the conditions of occurrence;
- type of FR: hardware, software, documentation, and the affected modules;
- analysis: identification and classification of the fault which led to the failure (coding, specification, interface, . . .);
- solutions: the proposed solutions and those retained;
- modification control: control carried out by the person in charge of the module;
- edition control: control of the corrected modules;
- regression testing: results of the tests applied to the corrected module(s).

Only one FR is kept per detected failure: rediscoveries [2] are not recorded. In other words, if several FR's, caused by the same fault, are drawn up by different testing groups, only one FR is entered into the database: generally the first one. In fact, an FR is a *failure report* as well a *fault (or correction)* report since it also contains information on the fault that led to failure.

Two TROPICO-R ESS software versions have been developed, namely 1500 and 4096 subscriber versions. The analyzed failure data correspond to the former version. Software volume is about 300 kb written in Assembly language.

C. TROPICO R Failure Data

The cumulative number of failures per periods of ten days (units of time) is indicated in Fig. 2: data corresponding to the validation test go from time unit 1 to 30, the field trial test goes from time unit 31 to 42. The validation test went on for about ten months and resulted in 297 corrections. The field trial was performed for four months and led to 55 corrections, whereas 108 corrections were made during the first year of operation. Time corresponds to calendar time excluding vacation periods. Even though execution time seems more suitable than the calendar time [25], it was not easy to obtain this information[3] in our case.

Validation		Field trial		Operation	
Time unit	CNF	Time unit	CNF	Time unit	CNF
1	7	31	301	43	356
2	8	32	302	44	367
3	36	33	310	45	373
4	45	34	317	46	373
5	60	35	319	47	378
6	74	36	323	48	381
7	82	37	324	49	383
8	98	38	338	50	384
9	106	39	342	51	384
10	115	40	345	52	387
11	120	41	350	53	387
12	134	42	352	54	387
13	139			55	388
14	142			56	393
15	145			57	398
16	153			58	400
17	157			59	407
18	174			60	413
19	183			61	414
20	196			62	417
21	200			63	419
22	214			64	420
23	223			65	429
24	246			66	440
25	257			67	443
26	277			68	448
27	283			69	454
28	286			70	456
29	292			71	456
30	297			72	456
				73	457
				74	458
				75	459
				76	459
				77	459
				78	460
				79	460
				80	460
				81	461

Fig. 2 Cumulative number of failures (CNF) per periods of 10 days (time unit)

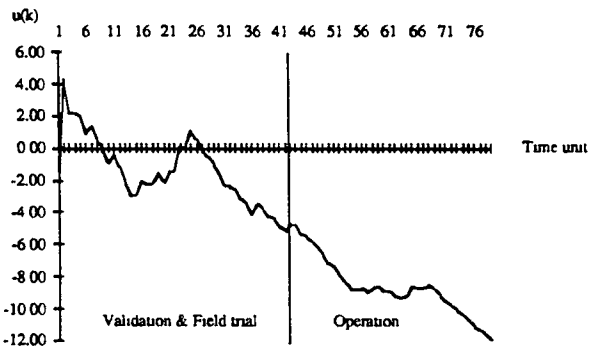


Fig 3. Result of Laplace trend test for the whole set of data.

V. Evaluation of TROPICO-R Reliability

When this study took place, all the failure reports were available, and it was not possible to apply the method presented in Section 2 in a truly predictive situation. However, we have tried to show how the results would have been used in real time.

First, the Laplace test is employed to identify the trend. Then, reliability growth models are used to predict the number of failures and the residual failure rate in operation. The characteristics of the S-shaped model and the hyperexponential models are given in Appendix B.

A. Application of the Laplace Trend Test

First, the Laplace test is applied to the whole set of data. The results are displayed in Fig. 3, which gives the Laplace factor $u(k)$ versus k, the number of time unit, and where a high global reliability growth can be noticed at the end of the observation period.

The results, obtained separately for each phase (using data pertaining specifically to the phase considered), are shown in Fig. 4, and are presented in chronological order.

It is interesting to comment on both figures at the same time:

- at the beginning of the validation, "an apparent" reliability decrease took place as a result of the correction of 28 faults during the third unit of time, whereas only 8 faults were removed during the first two time units and 24 during

[3]TROPICO-R software is divided into several implementation modules. Some modules (such as the Operational System modules) are more executed than other (e.g application modules). As TROPICO-R is a distributed system, it is very difficult to evaluate CPU time. Accordingly it is useful to adopt calendar time as the time basis for the reliability evaluation During the validation phase there was only one prototype working all day long, so that calendar time can be regarded as a good time basis.

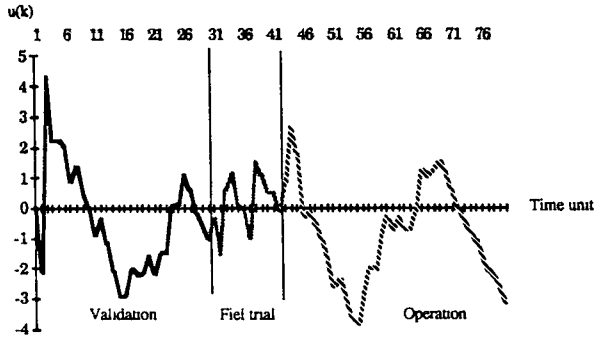


Fig 4. Results of Laplace trend test for each phase (considering all sites in operation).

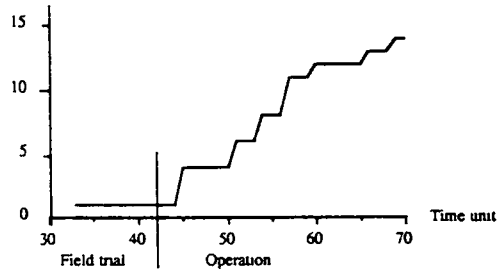


Fig. 5. Number of installed sites versus time.

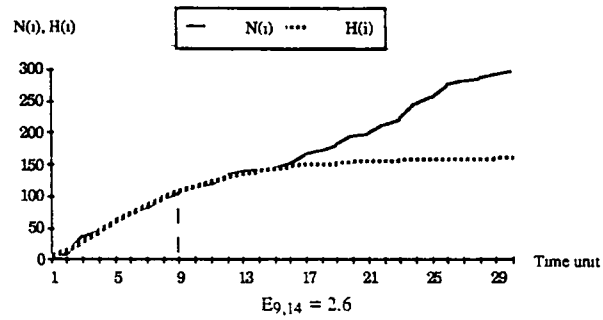


Fig. 6. Application of the S-shaped model to validation data.

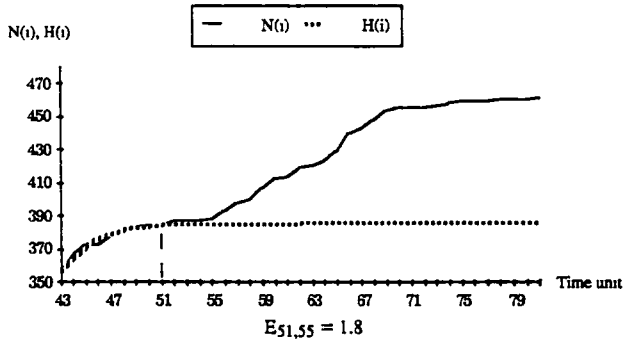


Fig. 7. Application of the S-shaped model to operational data.

the next two time units; applying the trend test without considering the data belonging to the three first time units leads to a negative Laplace factor for the whole period (however the shape of the $u(k)$ curve is not changed);

- the local reliability decrease from $k = 14$ to $k = 25$ was induced by the changes in nature of the tests within the validation phase: this period corresponds to the application of the quality and the performance tests after the functional tests in the previous period; this decrease is due to their dynamic nature (traffic simulation) which has activated new parts of the program;
- transitions from validation to field trial and from field trial to operation did not give rise to a reliability discontinuity: reliability was still improving;
- Fig. 3 indicates that from $k = 55$ up to $k = 70$ reliability tends to be stabilized: $u(k)$ is almost constant, suggesting a local reliability decrease; this behavior is reinforced when considering the trend results obtained for operational data only in Fig. 4, where reliability decrease is more evident; from $k = 70$ this trend is reversed; this failure behavior is directly related to the number of installed exchanges over the periods considered (see Fig. 5), during which about 12 exchanges were installed and the number of failures reported by the users increased; by time unit 70, the 4096 version had been released and no new 1500 version had been installed, which corresponds to the period of local reliability growth (fewer failures reported).

These results show that the systems installed have reached a sufficient level of quality after the new software releases which incorporate the necessary corrections. It is worth noting that, for the same calendar time used as a reference in the following, the increase in the number of software copies is equivalent to a stress test in terms of execution time. Reliability decrease does not necessarily mean that software is less reliable, however it does imply an increase in the maintenance effort that will have to be taken into consideration for maintenance planning.

B. Model Application

The results of the trend analysis evidence changes in the trend over the considered phases and two periods of reliability decrease have been identified. These results guide the model application:

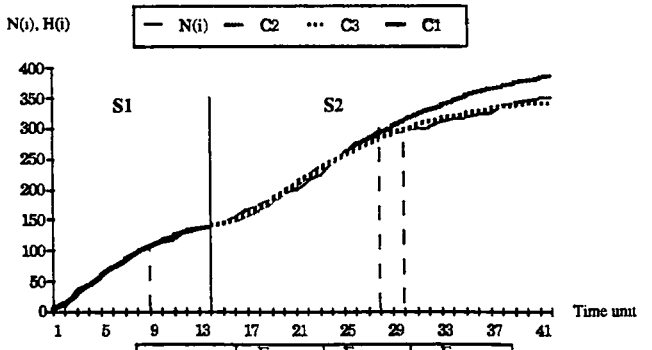
- an S-shaped model (SS) is applied to follow up the number of detected failures;
- the hyperexponential model (HE) is applied at the end of operational life in order to estimate the residual failure rate of the software.

1) Model Application to the Whole Data: Let us apply the SS model to validation data: data from $k = 1$ to $k = 8$ are used to estimate the model parameters and predict the cumulative number of failures. Results are given in Fig. 6 where $N(i)$ denotes the observed cumulative number of failures up to unit time i and $H(i)$ the estimated one. Predictions are very good up to $k = 15$ following which they soon become highly inaccurate. Indeed, $k = 14$ corresponds to a change in the condition of use (type 2 inflexion point), the model cannot predict this change and, as a result, it continues to assume reliability growth and underestimates the cumulative number of failures.[4]

The same may apply to operational data: the cumulative number of failures after $k = 55$ is increasingly underestimated (Fig. 7).

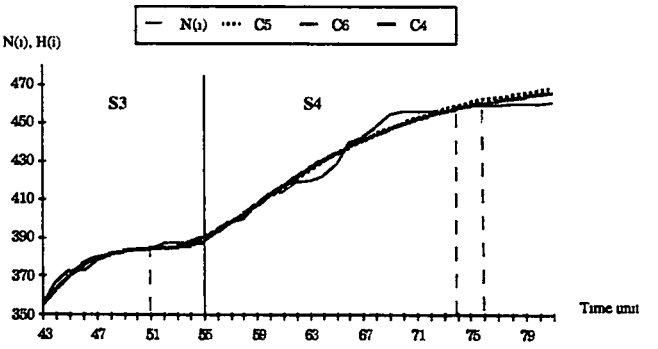
These two model applications show that the S-shaped model cannot follow the modifications in trend due to type 2 inflexion points. According to the previous section, data will be partitioned into stages.

[4]Step-by-step prediction from $k = 9$ does not significantly improve the results up to $k = 14$ (the residue is not affected); however better results are obtained after $k = 15$ but they are still erroneous due the type 2 inflexion point.



	$E_{9,14}$	$E_{28,42}$	$E_{30,42}$
C1	2.6		
C2		28.4	31.2
C3			5.8

Fig. 8. Validation and field trial prediction using Laplace test results.



	$E_{51,55}$	$E_{74,81}$	$E_{76,81}$
C4	1.8		
C5		4.3	5.3
C6			3.5

Fig. 9. Operational prediction using Laplace test results.

Data corresponding to validation and field trial are thus divided into two stages (S1 and S2):

S1: Data up to $k = 14$ for which $u(k)$ is continuously decreasing.

S2: Data counted from $k = 15$ up to the end of field trial testing.

Data corresponding to operation are also divided into two stages (S3 and S4):

S3: Data from $k = 43$ up to $k = 54$ for which $u(k)$ is continuously decreasing.

S4: Data counted from $k = 55$ up to the end of the observation period.

Reliability growth models are applied according to these stages.

2) Validation and Field Trial Phases: Fig. 8 shows the cumulative number of failures estimated by applying the S-shaped model three times to different data sets (C1, C2, C3) and the cumulative number of observed failures $N(i)$, where:

C1: Prediction for S1 from the failure data recorded over the first eight units of time of S1, $k = 1$ to $k = 8$.

C2. Prediction for S2 using the failure data recorded up to 2 time units after the type 1 inflexion point which corresponds to time unit 25 (i.e., data from $k = 15$ up to $k = 27$ of S2).

C3: Prediction for S2 using data including 4 time units after the inflexion point (i.e., data from time unit 15 up to 29 of S2).

Prediction accuracy is given by mean prediction errors listed in Fig. 8. C1 and C3 have a low residue whereas C2 exhibits a relatively high residue.

The last prediction (C3) yields remarkable results over approximately 4 months (13 units of time of 10 days each): the mean error is 5.8 and points to a relative error between time units 30 and 42 which is less than $\frac{5.8}{29}$, the latter being less than 2%.

How to Use These Results During Validation

From data corresponding to the first two and a half months of validation ($k = 1$ to 8), curve C1 indicates that if the same type of tests is continued (i.e., functional tests) these tests will be inefficient from the 12th or 13th unit of time (one month later) since $H(i)$ will reach a stable value. In other words, the number of detected failures per unit of time will reach an extremely low value at this stage. The designer can thus plan to change the type of tests to be applied in the following months.

From time unit 14, quality and performance tests were applied leading to a reliability decrease. Since the condition of use are changing (inducing local tendency changes), prediction will not be very accurate. Some time is therefore needed prior to applying any reliability model. Indeed, these models can only be applied again when reliability growth is noticed, that is, in our particular case, some units of time after the next inflexion point (since curves C2 and C3 indicate that the best predictions are obtained when applying models, 4 units of time after tendency change).

The shape of $H(i)$ is more interesting than the precise values of the figures: both curves show that for $k = 41$ (date scheduled for software release), $H(i)$ will begin to be stabilized, meaning that the software will not be in a stable state at the beginning of operational life. This was confirmed later.

3) Operational Life Phase: When the software is in operation, two types of measures are interesting, that is: *the number of failures* over all the installed sites to estimate the maintenance effort to be made during the operational life and *the failure rate of one site* to evaluate the whole system failure rate (i.e., hardware and software). These two measures are successively addressed below.

a) Maintenance Effort Prediction: Maintenance effort can be estimated through evaluation of the number of corrections to be performed on the software during the next period of time. Since FR's are failure as well as correction reports, the number of corrections is estimated by applying a "reliability growth model" to the collected data over all installed sites, i.e., corrections made on the software product.

The prediction of the cumulative number of failures applied to the operational life failure data is given in Fig. 9 where (C4, C5, C6) are estimated by application of the SS model to three different data sets and $N(i)$ the cumulative number of observed failures (i.e., corrections made)

C4: Prediction for S3 based on the first eight units of time of S3, $k = 43$ to $k = 50$.

C5: Prediction for S4 based on the failure data recorded from $k = 55$ to $k = 73$, that is, 2 time units following the type 1 inflexion point ($k = 71$, Fig. 4).

C6: Prediction for S4 based on the failure data recorded from $k = 55$ to $k = 75$, that is, 4 time units following the type 1 inflexion point.

C5 and C6 are very close to each other and the difference in residue is not significant. Both show that from the 75th unit of time $H(i)$ is almost linear and that the hyperexponential model

i	t_i
428	14
429	14
430	0 *
431	0
432	14
433	14
434	42
435	0
436	0
437	14
438	0
439	0
440	42
441	42
442	14
443	0
444	90
445	15
446	0
447	15
448	0
449	120
450	15
451	90
452	0
453	0
454	0
455	90
456	30
457	420
458	180
459	90
460	555
461	480

* k successive zeros mean that (k+1) failures took place the same day.

Fig. 10. Interfailure times during the last months of operation (in days).

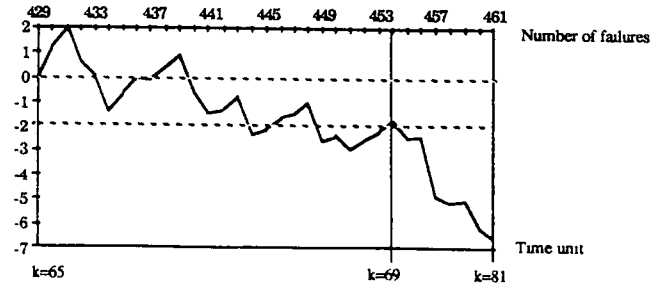


Fig 11 Laplace factor for the last months of operation.

i	t_i	$MTTF_i$
441	42	23 35
442	14	38 52
443	0	28.77
444	90	18.77
445	15	36.05
446	0	32.23
447	15	27.23
448	0	25.56
449	120	22.41
450	15	32.87
451	90	31.16
452	0	36.22
453	0	33.38
454	0	30.91
455	90	28.77
456	30	32.71
457	420	32.54
458	180	54.78
459	90	61.64
460	555	62.81
461	480	88.03

$R_{441,461} = 0.65$ $R_{441,456} = 0.10$
Kolmogorov-Smirnov distance: 0.56
$\lambda_s = 8.3\ 10^{-3}$/day

Fig. 12. Results of the HE model application.

can be applied from this stage in order to predict the residual failure rate.

Application of the SS model until time unit 90 (three months after the considered period) indicates that for the whole installed sites, the expected number of corrections to be made is about 2 during the next month and 1 per month for the next two months, i.e., 4 corrections during the next quarter. This was confirmed later.

b) Failure Rate and Mean Time between Failures in Operational Life: To evaluate the failure rate and mean time between failures, the random variable interfailure time has to be considered.

The S-shaped model was applied using data gathered in an increasing number of sites in order to determine the number of corrections to be performed on the software product. When the MTTF is addressed, the user is interested in the reliability of one site using all the collected data. Data have thus been modified so as to integrate the number of installed sites. Fig. 10 gives the observed interfailure times t_i (in days) for one site (an average site[5]) during the last months of operation where 34 corrections were made.

Laplace Test: The Laplace test is then applied to this subset (the considered random variable is the interfailure time). The results are given in Fig. 11 and indicate an almost steady reliability at the beginning of this period (with $u(k)$ oscillating between $+2$ denoting a low reliability decrease and -2 denoting a low reliability growth) followed by a relative reliability growth. The time axis is added to the figure in order to show the correspondence between failures and their time of occurrence and to allow comparison to Fig. 4.

[5]The conditions of operation of a telephone exchange may vary greatly according to the traffic characteristics and the number of sites. Unfortunately data related to traffic change and to the environment are not available for this system. Also failure data do not give the identification of that site where the failure occurred. This is why it was assumed that all sites are submitted to the "same" environmental conditions and an average site is one of these identical sites. A more refined evaluation would certainly be obtained by taking into account the different types of use. The notion of an "average site" has also been introduced in [25].

Reliability growth at the end is mainly due to the last five observations for which the mean observed interfailure time is about 345 days for an average site whereas it was about 24 for the previous 5 data. These intervals are 23 days and 1.6 day, respectively, for the 15 installed sites.

HE Model Application: The results of the HE model application to the considered period are summarized in Fig. 12, where the prediction is done step-by step: data from $j = 428$ to $j = i-1$ are used to estimate model parameters and $MTTF_i$. It can be seen that the model gives very pessimistic (and bad) results for the last five data (as prescribed by Laplace test results).

The residue varies from 0.65 when considering these data to 0.10 without these data.

The Kolmogorov–Smirnov distance is 0.56 which is not good, since the result is significant at the 1% level.

The estimated residual failure rate λ_s is $3.4 10^{-4}$/h (8.310^{-3}/day), i.e., one failure every 4 months per site. This time interval is much less than that observed at the end of the considered period (which corresponds to about one failure every 345 days, for an average site during the last five data). Considering the 15 installed sites, this means 15 failures for the next 4 months for all the sites or one failure every 8 days, to be compared to the obseved one which is about 1 failure every 23 days.

The estimated residual failure rate without taking into account the last data is about 310^{-2}/day (i.e., 1.2510^{-3}/h); although in accordance with the observed failure rate during this period it may seem very high. These results may be used as follows. The residual failure rate may be considered as very high; if this is so

i	t_i	$MTTF_i$
454	0	37.50
455	90	32.14
456	30	39.37
457	420	38.33
458	180	323.89
459	90	279.01
460	555	216.17
461	480	298 43

$R_{454,461} = 0.31$
Kolmogorov-Smirnov distance: 0.29
$\lambda_s = 2.98\ 10^{-3}$/day

Fig. 13. Results of the HE model application to S6.

the users may ask the designer to enhance software reliability, and the latter has to develop new test sets that can be run either in the laboratory or on some installed sites in parallel with normal system utilization.

It can be pointed out that the HE model experiences great difficulties in following a sharp and sudden reliability growth: it underestimates the last interfailure times and overestimates the residual failure rate (this property is also true for other reliability growth models, as shown in [15]). To overcome these difficulties, data will be partitioned into two sets, as for the SS model.

S5: Data from 428 up to 447 for which reliability growth is difficult to ascertain.

S6: Data counted from 448 for which reliability growth is clear.

Application of the HE Model to S6: The results obtained when using data from 448 to 453 to estimate the parameters of the model (Fig. 13) are better than the previous ones. Predictions for the last observations are better (the residue is lower, the Kolmogorov-Smirnov distance is better, and the result is significant at the 20% level) and the residual failure rate seems more realistic.

The residual failure rate is about 1.310^{-4}/h (2.9810^{-3}/day) for one site, i.e., one failure every 336 days. This is more in keeping with the interfailure times observed at the end of the data set and subsequently. This failure rate is quite high compared to the hardware failure rate which is about 410^{-6}/h. Indeed, the hardware is fault-tolerant and the failure rate corresponds to failures leading to system unavailability, whereas the software is not fault-tolerant and all the consequences of the failures on the delivered service are included in this figure. It is worth noting that most of the software failures do not lead to system unavailability or call loss. Collected data thus have to be partitioned into subsets according to failure consequences on the delivered service. The HE model will be applied to each of the subsets in order to evaluate the contribution of the software to the whole system unavailability. To do this, a thorough analysis of the failure reports is needed to ascertain the effects of each failure: this analysis has not yet been undertaken. Application of the same approach to software failure data concerning another ESS [15], [16] showed that the failure rate leading to system unavailability is about 5% of the global software failure rate.

VI. Conclusions

Software reliability growth models are an important aid to test/maintenance planning and reliability evaluation. However, it is well-known that no particular model is better suited for predicting software behavior for all software systems in any circumstances. Our work helps the already existing models to give better predictions since they are applied to data displaying trends in accordance with their assumptions.

With respect to the application of the proposed method to the failure data of the TROPICO-R ESS, two models, namely, the hyperexponential and the S-shaped models, have been analyzed according to their predictive capabilities.

The results obtained show that:

- the trend test helps partition the observed failure data according to the assumptions of reliability growth models; it also indicates the segment of data from which the occurrence of future failures can be predicted more accurately;
- the prediction approach proposed for the validation and the field trial phases yields good results over a time period of a few months, showing that reliability modeling constitutes a major aid tool for test/maintenance planning and follow up;
- the residual failure rate prediction given by the hyperexponential model is in accordance with the observed failure data at the end of the considered period.

Our current work concerns the TROPICO-R 4096 version and is directed toward:

- the application of reliability growth models to hardware design failures recorded during the test and operational phases in order to derive a reliability model for the whole system;
- the analysis of software component failure data in order to deduce system dependability from the components measured reliability.

Appendix A
Laplace Factor Derivation

The expression of the Laplace factor $u(k)$ when the random variable considered is the cumulative number of events (failures for our purpose) is derived as indicated in [7, p. 54].

Let us consider k time intervals of equal length h and let :

N_i be the random variable that represents the number of events that occurred in the time interval $[(i-1)h, ih]$, $i = 1, \ldots, k$.

n_i be the realization of N_i, i.e., the number of events that occurred in the time interval $[(i-1)h, ih]$, $i = 1, \ldots, k$.

The occurrence of the events follows a nonhomogeneous Poisson process whose occurrence rate is given by the equation

$$\lambda(t) = e^{a+bt}. \tag{A1}$$

If $b = 0$ the Poisson process becomes homogeneous and the occurrence rate is time independent. If we observe k time intervals, the likelihood is

$$L(k) = \prod_{i=1}^{k} P\{N_i = n_i\} = \prod_{i=1}^{k} \frac{\left[\int_{(i-1)h}^{ih} \lambda(u)du\right]^{n_i}}{n_i!} \exp\left(-\int_{(i-1)h}^{ih} \lambda(u)du\right). \tag{A2}$$

For the occurrence rate (A1), the likelihood becomes

$$L(k) = \frac{(e^{bh}-1)^{k} e^{ak+bh\sum_{i=1}^{k}(i-1)n_i}}{b^{k}\prod_{i=1}^{k} n_i!} \exp\left[\frac{e^{a}(1-e^{bkh})}{b}\right]. \tag{A3}$$

where $N = \sum_{i=1}^{k} n_i$.

Thus the conditional probability density function of the observations, given that N events have occurred in k time intervals, is obtained dividing (A3) by the probability

$$P(k) = P\left\{\sum_{i=1}^{k} n_i = N\right\} = \frac{\left[\int_0^{kh} \lambda(u)du\right]^N}{N!} \exp\left(-\int_0^{kh} \lambda(u)du\right).$$

The result is given by (A4) and only depends on the b variable

$$\frac{L(k)}{P(k)} = \frac{N!(e^{bh}-1)^N e^{bh} \sum_{i=1}^{k} (i-1)n_i}{(e^{bkh}-1)^N \prod_{i=1}^{k} n_i!}. \quad \text{(A4)}$$

The conditional log likelihood $\mathcal{L}(b)$, is, from (A4),

$$\mathcal{L}(b) = \log N! - \log \prod_{i=1}^{k} n_i! + bh \sum_{i=1}^{k} (i-1)n_i + N\left[\log\left(e^{bh}-1\right) - \log\left(e^{bkh}-1\right)\right]$$

so that

$$\mathcal{L}'(b) = \frac{d\mathcal{L}(b)}{db} = \begin{cases} h\sum_{i=1}^{k}(i-1)n_i + N\left[\frac{he^{bh}}{e^{bh}-1} - \frac{khe^{bkh}}{e^{bkh}-1}\right], & b \neq 0 \\ h\sum_{i=1}^{k}(i-1)n_i - \frac{Nhk(k-1)}{2}, & b = 0 \end{cases} \quad \text{(A5)}$$

and the information function $\mathcal{I}(b)$ is

$$\mathcal{I}(b) = E\{-\mathcal{L}''(b)\} = \begin{cases} Nh^2\left[\frac{e^{bh}}{(e^{bh}-1)^2} - \frac{k^2e^{bkh}}{(e^{bkh}-1)^2}\right], & b \neq 0 \\ \frac{Nh^2(k^2-1)}{12}, & b = 0. \end{cases} \quad \text{(A6)}$$

Under the null hypothesis $b = 0$ (constant occurrence rate), the statistics

$$u(k) = \frac{\mathcal{L}'(0)}{\sqrt{\mathcal{I}(0)}} = \frac{\frac{\sum_{i=1}^{k}(i-1)n_i}{N} - \frac{(k-1)}{2}}{\sqrt{\frac{(k^2-1)}{12N}}}$$

are approximately normal distributed with zero mean and unit variance. A positive value of $u(k)$ means that the considered statistics is above the mean and therefore indicates $b > 0$, i.e., an occurrence rate increasing with time. On the other hand, a negative value of $u(k)$ suggests $b < 0$ (a decreasing failure rate).

Appendix B
Presentation of the Models

The mean function (cumulative number of failures) of the S-shaped model is given by

$$H(t) = [1 - (1 + \Phi t)]e^{-\Phi t}, \quad \text{(A7)}$$

where the parameters are N and Φ.

The intensity function (ROCOF), which is the derivative of $H(t)$, is as follows:

$$h(t) = N\Phi^2 t e^{-\Phi t}. \quad \text{(A8)}$$

The inflexion point is situated at $t = \frac{1}{\Phi}$.

The expression of the mean function of the hyperexponential model is

$$H(t) = Ln\left[\omega_1 e^{-\zeta_1 t} + \omega_2 e^{-\zeta_2 t}\right] \quad \text{(A9)}$$

with $0 \leq \omega_1 \leq 1.0 \leq \omega_2 \leq 1$. and $\omega_1 + \omega_2 = 1$.

Let $\omega_1 = \omega$ and $\omega_2 = 1 - \omega = \overline{\omega}$: the parameters of the model are: ω. ζ_1. and ζ_2.

The ROCOF is thus

$$h(t) = \frac{\omega\zeta_1 e^{-\zeta_1 t} + \overline{\omega}\zeta_2 e^{-\zeta_2 t}}{\omega e^{-\zeta_1 t} + \overline{\omega} e^{-\zeta_2 t}}. \quad \text{(A10)}$$

which is continuously decreasing from an initial value $(\omega\zeta_1 + \overline{\omega}\zeta_2)$ to $\lambda_s = \inf(\zeta_1, \zeta_2)$ (λ_s is the residual failure rate of the software).

The MTTF_i, which is the expected time to failure i, given that failure $(i-1)$ occurred at time s, is

$$\text{MTTF}_i = \frac{\frac{\omega}{\zeta_1}\exp(-\zeta_1 s) + \frac{\overline{\omega}}{\zeta_2}\exp(\zeta_2 s)}{\omega\exp(\zeta_1 s) + \overline{\omega}\exp(\zeta_2 s)}. \quad \text{(A11)}$$

The relevant features of this model are:

- the rate of decrease with respect to time is adjustable through parameter $(\omega, \zeta_1, \zeta_2)$ tuning;
- the fact that it asymptotically tends toward a nonzero limit λ_s;
- the fact that it also enables availability growth modeling [17], [19], [21].

Acknowledgment

This paper benefitted from discussions with J. C. Laprie from LAAS. The authors are also grateful to A. Costes and M. Kaâniche from LAAS and to an anonymous reviewer for their constructive suggestions.

References

[1] A. A. Abdel-Ghaly, P. Y. Chan, and B. Littlewood, "Evaluation of competing software reliability predictions," *IEEE Trans. Software Eng.*, vol. SE-12, no. 9, Sept. 1986.
[2] E. N. Adams, "Minimizing cost impact of software defects," IBM Res. Division, Rep. RC 8228 (35669), Apr. 11, 1980.
[3] H. Ascher and H. Feingold, "Application of Laplace's test to repairable system reliability," in *Proc. 1st Int. Conf. Reliability and Maintainability*, Paris, France, June 19-23, 1978, pp. 219-225.
[4] H. Ascher and H. Feingold, *Repairable Systems Reliability: Modeling, Inference, Misconceptions and Their Causes (Lecture Notes in Statistics*, Vol. 7), 1984.
[5] V. R. Basili and D. M. Weiss, "Methodology for collecting valid software engineering data," *IEEE Trans. Software Eng.*, vol. SE-10, no. 6, pp. 728-738, Nov. 1984.
[6] M. R. Bastos Martini, K. Kanoun, and J. Moreira de Souza, "Software reliability evaluation of the TROPICO-R switching system," *IEEE Trans. Rel.*, vol. 39, no. 3, pp. 369-379, Aug. 1990.
[7] D. R. Cox and P. A. W. Lewis. *The Statistical Analysis of Series of Events*. London: Chapman & Hall, 1978.
[8] P. A. Currit, M. Dyer, and H. D. Mills, "Certifying the reliability of software," *IEEE Trans. Software Eng.*, vol. SE-12, no. 1, pp. 3-11, Jan. 1986.
[9] J. Favrot, C. Lamy, and F. Michel, "Reliability evaluation of programs for irradiated nuclear fuel testing during the qualification phase," *Technique et Science Informatiques*, vol. 4, no. 2, pp. 209-223, 1985 (in French).

[10] K. Fukushima and Y. Kishida, "Estimation of the bug curve using experimental regression analysis of office automation equipment software," in *Proc. 5th Int. Conf. Reliability and Maintainability*, Biarritz, France, Oct. 6–10, 1986, pp. 87-91.
[11] R.L. Glass, "Persistent software errors," *IEEE Trans. Software Eng.*, vol. SE-7, no. 2, pp. 162-168, Mar. 1981.
[12] A.L. Goel and K. Okumoto, "Time-dependent error-detection rate model for software and other performance measures," *IEEE Trans. Rel.*, vol. R-28, no. 3, pp. 206-211, Aug. 1979.
[13] Z. Jelinski and P.B. Moranda, "Software reliability research," in *Statistical Computer Performance Evaluation*, W. Freiberger, Ed. New York: Academic, 1972, pp. 465-484.
[14] R.K. Iyer, S.E. Butner, and E.J. MacClusky, "A statistical failure load relationship : Results of a multicomputer study," *IEEE Trans. Comput.*, vol. C-31, no. 5, pp. 696-706, 1982.
[15] K. Kanoun and T. Sabourin, "Software dependability of a telephone switching system," in *Proc. 17th IEEE Int. Symp. Fault-Tolerant Computing (FTCS-17)*, Pittsburgh, PA, July 6–, 1987, pp. 236-241.
[16] K. Kanoun, J.C. Laprie, and T. Sabourin, "A method for software reliability growth analysis and assessment," in *Proc. Le Genie Logiciel et Ses Applications*, Toulouse, France, Dec. 5–9, 1988, pp. 859-878.
[17] K. Kanoun, "Software dependability growth characterization, modeling and evaluation," Doctorat ès-Sciences dissertation, Institut National Polytechnique de Toulouse, France, LAAS Rep. 89-320, Sept. 1989 (in French).
[18] P.A. Keiller, B. Littlewood, D.R. Miller, and A. Sofer, "Comparison of software reliability predictions," in *Proc. 13th IEEE Int. Symp. Fault-Tolerant Computing*, Milano, Italy, June 1983, pp. 128-134.
[19] J.C. Laprie, "Dependability evaluation of software systems in operation," *IEEE Trans. Software Eng.*, vol. SE-10, no. 6, pp. 701-714, Nov. 1984.
[20] ——, "Dependability: A unifying concept for reliable computing and fault tolerance," in *Resilient Computing Systems*, vol. 2, T. Anderson, Ed. New York: Collins and Wiley, 1987; also LAAS Rep. 86-357, Dec. 1986.
[21] J.C. Laprie, K. Kanoun, C. Béounes, and M. Kaâniche, "The KAT (Knowledge-action-transformation) approach to the modeling and evaluation of reliability and availability growth," this issue, pp. 370–382.
[22] B. Littlewood and J.L. Verral, "A Bayesian reliability growth model for computer software," *J. Roy. Statist. Soc. C*, vol. 22, pp. 332-336, 1973.
[23] B. Littlewood, "Software reliability model for modular program structure," *IEEE Trans. Rel.*, vol. R-28, no. 3, pp. 241-246, Aug. 1979.
[24] J.D. Musa, "Software reliability data," Data and Analysis Center for Software, Rome Air Development Center (RADC), Rome, NY, 1979.
[25] J.D. Musa *et al.*, *Software Reliability Measurement, Prediction, Application*. New York: McGraw-Hill International, 1987.
[26] P.M. Nagel and J.A. Skrivan, "Software reliability: Repetitive run experimentation and modeling," Boeing Computer Service Co., Seattle, WA, Rep. BCS 98124, Feb. 1982.
[27] M. Ohba and S. Yamada, "S-shaped software reliability growth models," in *Proc. 4th Int. Conf. Reliability and Maintainability*, Perros Guirec, France, 1984, pp. 430-436.
[28] Y. Tohma *et al.*, "Structural approach to the estimation of the number of residual software faults based on the hyper-geometric distribution," *IEEE Trans. Software Eng.*, vol. 15, no. 3, pp. 345-355, Mar. 1989.
[29] R. Troy and Y. Romain, "A statistical methodology for the study of the software failure process and its application to the ARGOS Center," *IEEE Trans. Software Eng.*, vol. SE-12, no. 9, pp. 968-978, Sept. 1986.
[30] S. Yamada and M. Ohba, "S-shaped reliability growth modeling for software error detection," *IEEE Trans. Rel.*, vol. R-32, no. 5, pp. 475-478, Dec. 1983.
[31] B. Vianna, "R&D at TELEBRAS-CPqD: The TROPICO system," in *Proc. Int. Conf. Communications (ICC 88)*, Philadelphia, PA, June 1988, pp. 622-626.
[32] B. Vianna, E.C. Cunha, and F.F. Boin, "Hardware quality control in the TROPICO system," in *Proc. Int. Conf. Communications (ICC 88)*, Philadelphia, PA, June 1988, pp. 632-636.

Karama Kanoun received the Certified Engineer degree from the National School of Civil Aviation, Toulouse, France, in 1977, and the Doctor-Engineer and Doctor-ès-Science degrees from the National Institute Polytechnique of Toulouse in 1980 and 1989, respectively.

She is currently Chargée de Recherche at LAAS-CNRS, Toulouse. She joined LAAS in 1977 as a member of the Fault-Tolerance and Dependable Computing group. She has conducted several research contracts and has been a consultant for several French companies and for the International Union of Telecommunications. Her current research interests include modeling and evaluation of computer system dependability, considering hardware as well as software. Her Doctorat ès-Science dissertation is devoted entirely to software dependability growth characterization, modeling, and evaluation, including theoretical as well as practical aspects.

Dr. Kanoun is a member of the working group of the European Workshop on Industrial Computer Systems (EWICS): "Technical Committee 7—Reliability, Safety, and Security" and a member of the AFCET working group on dependability of computing systems.

Marta Rettelbusch de Bastos Martini received the Electric Engineer degree from Pará Federal University in 1980, the M.S. degree from Campinas State University—UNICAMP in 1983, and is currently working on the Doctorate degree in electric engineering.

She has been working in hardware and software reliability for six years as a researcher on the TROPICO project at TELEBRAS R&D Center, Campinas, Brazil, and is a member of the Quality Assurance team.

Jorge Moreira de Souza received the B.S. and M.S. degrees in electric engineering from the Catholic University of Rio de Janeiro in 1971 and 1975, respectively, and the Doctor ès-Sciences degree in 1981 from the Institut National Polytechnic de Toulouse, France.

He spent a sabbatical year at the Laboratoire d'Automatique et d'Analyse des Systèmes of CNRS, Toulouse, in 1988 as an invited researcher. He is responsible for the Quality Assurance Group of the Switching Department of TELEBRAS Research and Development Center, Campinas, Brazil. His current interests are switching systems and performance and reliability evaluation.