A Quality Constraint Model to be Used during the Test Phase of the Software Life Cycle

Wohlin, C. and Vrana, C.

Proceedings 6th International Conference on Software Engineering for Telecommunication Switching Systems, pp. 136-141, Eindhoven, The Netherlands, 1986.

A Quality Constraint Model to be Used During the Test Phase of the Software Lifecycle

by

Claes Wohlin, Lund Institute of Technology

and

Ctirad Vrana, Telelogic AB

Sweden

SUMMARY

This paper deals with the problem of when to stop the test phase in the software lifecycle, in order to get an efficient, reliable, manageable and maintainable software product. A quality constraint model is introduced and formulas for the means and variances of the number of remaining errors at a specific time and the time when a specific number of errors are remaining are derived. A natural approach to the use of the model and results for some different failure time distributions are presented. The impact of the different failure time distributions is discussed and some ideas of what to do about the situation that the model (and the reality) leads to are presented.

A QUALITY CONSTRAINT MODEL TO BE USED DURING THE TEST PHASE OF THE SOFTWARE LIFECYCLE

C. Wohlin*, C. Vrana**

*Lund Institute of Technology, **Telelogic AB, Sweden

1. BACKGROUND

The software industry is an industry in crisis. The reason for this is not the same as for many other industries, where jobs are disappearing. On the contrary, the problems and work are growing faster than we are able to cope with.

The problems in the software industry are of several different types:
- The software projects are often missing their target dates (due to lack of appropriate methods), and even more often not fulfilling the quality constraints, and consequently the software product will be difficult to manage and maintain during operation.
- The software projects are often way over budget, and at the same time it is getting worse because resources are missing.
- The software projects are often badly planned, if planned at all, and out of control. This leads to unmanageable, unmaintainable, unreliable and inefficient software products and, in the worst case, to no product at all.

The list of problems could be made much longer, but the point is that it is necessary to increase the effectiveness of software handling routines and of the tools, both during construction, testing and operation. The condition for doing this is to improve understanding of laws and relations which concern the structure and handling processes of software products.

During work in different projects we have studied economical and quality (here in the sense of correctness and reliability) aspects, metrics and models describing different relations concerning the software product and its handling. At the moment the studies focus on some local phenomena, but in the long run the aim is to lower the total cost for handling software products throughout the whole lifecycle.

This paper presents a model, which describes some of the relations in the lifecycle. The model deals with the problem of when (if ever) to let the product enter the field. We will here consider the transition from the test phase to the operation phase.

Specifically the studies focus on the following aspects, which will be surveyed and analyzed:
- planning
- a realistic decision (estimation) of the (optimal) time of release
- problems we meet (sometimes unprepared and without understanding) while performing the transition from the test phase to the operation phase
- some inevitable consequences that the model leads to (and we can find in reality as well).

2. INTRODUCTION TO THE PROBLEM

Common sense and all experiences points to the fact that only software products which fulfil the given (whatever they are) quality constraints could be successfully used, managed and maintained.

One of the most important (and fundamental) quality constraints is reliability. An important condition, to obtain the desired reliability, is that the errors do not appear, i.e. that the software product is correct (i.e. that the quality constraint of correctness is fulfilled).

There are four ways of making the programs correct:
- to use better methods and tools for design and construction.
- to spend more time on design and construction.
- to use better methods and tools for testing (correcting errors).
- to spend more time testing (correcting errors).

The first point is, for obvious reasons, the best one, but in reality the solution of the problem is a combination of all four ways.

Very early in the lifecycle, often when the system and project specifications are written, one usually decides the different target dates to meet. Primarily the time to put the software product into operation is decided, see Figure 1.

Because reliable tools for planning are still missing, (both in planning the need of resources and obtainable quality in a given time), it will probably be impossible, at the predetermined time to put the product into operation, and obtain the planned quality. Or, at best, the estimation is true as far as the mean value, but because of the stochastic nature for many of the processes the estimation will be problematic. The stochastic behaviour in which the errors occur will be taken into account in the model and we will not only consider the mean values but also the variances.

For example, if we are forced to keep to the planned target date the number of remaining errors in the software product will partly be uncontrollable. The worst thing to do, without any doubt, is to accept the product just because the target date is reached. This may cause serious consequences while the system is in operation. It will probably be very expensive to maintain the system, if it is at all maintainable. An extended test phase does not have to be viewed as a failure, often rather as an inevitable consequence of the stochastic aspects in the handling processes.

3. MODEL DEVELOPMENT

The derivations of formulas found in this section are made in Wohlin and Vrana (1).

3.1 Considering the number of errors at a specific time

We want to solve the following problem: How many errors remain after testing a time t and how do the number of remaining errors vary?

The problem will be solved with the following three assumptions:
1. The initial number of errors, when the test phase is started, are known through estimation.
2. The failure time distribution is known.
3. If a failure is found, it is corrected and no new failures are introduced.

Comments to the assumptions:
1. The initial number of errors are supposed to be known through estimation from some complexity measure and/or experiences from earlier projects, this is discussed further in Vrana and Wallander (2) and Lennselius (3). A complexity measure is a measure of the degree of difficulty of the product, e.g. from a graph or a program. There are many different complexity measures presented in the literature, e.g. McCabe's Cyclomatic Number (4) and Halstead's Software Science (5).
   The estimation of the initial number of errors could be improved during testing by using failure rate models, some results are presented in (2). Two of the most used and accepted failure rate models are:
   * Jelinski-Moranda's De-Eutrophication Model (briefly described later in this paper, (6))
   * Goel-Okumoto's Non-Homogenous Poisson Process Model (7)
2. The failure time distribution could be found through measurements on earlier projects. There are two different ways to approach the problem: either we could use an existing failure rate model, or develop a model of our own that fits our environment and applications. Results for different failure time distributions will be presented below, and we will see that the failure time distribution will have a great influence on reliability, number of remaining errors and the times considered.
3. This assumption is quite natural, despite the fact that it is not always true. This is, however, always the aim when correcting errors. The derivation of results will be easier with this assumption, it is, however, possible to find some results if we assume that the error is corrected with the probability α, but we will not consider this case here.

To be able to derive the desired formulas, we introduce the following notation:

* $F(t)=P(\tilde{t}\leq t)$ - the probability distribution function (PDF)
* $f(t)=dF(t)/dt$ - the probability density function (pdf)
* $F^*(s)$ - the Laplace transform of f(t)
* $P(z)$ - the z-transform of $p_k=P(\tilde{n}=k)$
* $E(X)=\bar{x}$ - the mean value of X
* $V(X)$ - the variance of X
* $E(X^k)=\overline{x^k}$ - the k:th moment of X
* N_0 - the initial number of errors, i.e. when the test phase begins
* ΔN_0 - the uncertainty in the estimation of the initial number of errors
* N_t - the number of remaining errors at time t, (pdf $g_k(t)$)
* X_i - the time between error number (i-1) and error number i, (pdf $f_i(x)$)
* S_n - the time to error number n, (pdf $k_n(t)$)

For those who need an introduction or a refresher to the probability theory or the transform theory, we would like to recommend Feller (8), Kleinrock (9), Cox (10) and Cox and Miller (11).

With these definitions we obtain Figure 2. It should be noted that the starting point for the analysis, (in Figure 2), is any suitable point after the beginning of the test phase.

We need to derive the z-transform for N_t and then, by using the moment generating properties of the z-transform, we are able to obtain the Laplace transform of the mean and variance of the number of errors remaining at t, which can be inverted numerically and in some special cases analytically, for more details see Wohlin (12).

If we let $H(t)=E(N_t)$ and $M(t)=E(N_t^2)$ with Laplace transforms $H^*(s)$ and $M^*(s)$, we obtain

$$H^*(s)=\frac{1}{s}\left(N_0-\sum_{n=1}^{N_0}\prod_{i=1}^{n}F_i^*(s)\right) \qquad (1)$$

and

$$M^*(s)=\frac{1}{s}\left(N_0^2-\sum_{n=1}^{N_0}(2(N_0-n)+1)\prod_{i=1}^{n}F_i^*(s)\right) \qquad (2)$$

$H^*(s)$ and $M^*(s)$ can be inverted numerically. The inversion is done by computer programs. Some methods and example of Fortran programs are found in Karlsson and Stavenow (13).

The variance is now easy calculated as

$$V(N_t)=E(N_t^2)-E(N_t)^2 \qquad (3)$$

We have now obtained a possibility to calculate the mean and variance of the number of remaining errors in the software product at time t.

Particularly we could invert $H^*(s)$ and $M^*(s)$ at the target date, and get a good idea of the number of remaining errors and how they vary at the target date. Before we release

the product, we have to ask ourselves the question: Are we satisfied with the quality of the product?

If $F_i^*(s)=F^*(s)$, i.e. f(x) is independent of the number of errors that have occurred. This will give us a little simpler formulas for $H^*(s)$ and $M^*(s)$ and if we then let $N_0 \to \infty$, hopefully not realistic, we could obtain asymptotic results.

3.2 Considering the time when a specific number of errors are remaining

We discussed above the variation of N_t. We should now look at the variation of t, when we specify the number of errors remaining (or those which have occurred). This would be a better way to do the testing, i.e. to put a quality constraint on the product and then adjust the time to put the product into operation. An example of how to decide the quality constraint will be discussed below.

Using the notation and assumptions from above we can derive formulas for the mean and variance of the time when a specific number of errrors are remaining. This is done by deriving the Laplace transform for S_n and then, by using the moment generating properties of the Laplace transform, we are able to obtain $E(S_n)$ and $V(S_n)$.

When using the moment generating properties of the Laplace transform, we find

$$E(S_n)=\sum_{i=1}^{n} \bar{x}_i \qquad (4)$$

and

$$E(S_n^2)=\sum_{i=1}^{n} (\overline{x_i^2}+2\bar{x}_i \sum_{k=i+1}^{n} \bar{x}_k) \qquad (5)$$

which leads to

$$V(S_n)=E(S_n^2)-E(S_n)^2 \qquad (6)$$

We have now obtained a possibility to calculate the mean and variance of the time when a specific number of errors are remaining.

Using the results obtained here will give us a good idea of the amount of time we have to test the product, to fulfil the quality constraints and how the time varies.

3.3 A natural approach to the use of the model

It is well-known that it costs more to correct an error in the operation phase than in the test phase, but it is not economical to make the test phase too long either, just to get rid of all errors. This simple discussion leads to the observation that there has to be an economical minimum at some time, see Figure 3.

The cost function is hard to find. It depends on many factors, e.g. the system, the environment, testing methods, the market, maintenance routines etc. But if we could find the cost function, then we would be able to find out how many errors we should try to correct during the test phase, before we release the software product to obtain the economical optimum.

A natural approach to our problem, would be to:

1. Decide how many errors we should remove in the test phase to obtain minimum of the cost function.
2. Use the developed model to obtain the mean and variance of the time we have to spend in the test phase.
3. The project management and the marketing department could plan for different cases, from the best to the worst depending on how the test phase goes. We should always be prepared for the worst and not be surprised if it happens, rather, be happy if it does not.

3.4 A short presentation of an important parameter in the model: the failure time distribution

To be able to use the developed model, we have to know the failure time distribution. We are in this paper interested in the principle results when assuming different failure time distributions. The problem of finding a realistic distribution, for a specific environment and application, will not be considered here.

A model which is often used for the time between errors is the Jelinski-Moranda De-Eutrophication Model, (6).

A brief description of the J-M model:

The following notation will be used:

* $\lambda(i)$ - error rate
* $f_i(x)$ - pdf for the time to discover error number i
* N_0 - the initial number of errors
* n - the number of removed (corrected) errors
* ϕ - proportionality factor

The time between error occurrences is assumed to be exponentially distributed.

$$f_i(x)=\lambda(i)\exp(-\lambda(i)t) \qquad (7)$$

The error rate is proportional to the number of remaining errors in the system.

$$\lambda(i)=\phi(N_0-n) \qquad (8)$$

ϕ and a better estimation of N_0 will be found as the test phase progresses, by application of the maximum likelihood method.

The J-M model is developed under the condition that we have an environment similar to the one we have during the operation phase. This is not always the case. The model is not suitable for module, unit or function testing, where we have the problem of how to book the time. An investigation has been made and it seems more reasonable to assume that we have a distribution with a failure rate that is independent of the number or remaining errors while we are doing the module, unit or function testing. This is understandable, because correcting errors in one unit will not make the other units better. The failure time distribution is, however, dependent on other factors, e.g. test methods, number of test cases etc. Problems arising when considering the failure time distribution during functional testing are presented in Wohlin (14).

If we assume $f_i(x)=f(x)$, we could look at some results using well-known distributions. We have chosen to look at the following four distributions a little closer:

1. Exponential distribution
   (pdf $f(x)= \lambda \exp(-\lambda x)$) (9)
2. Two-stage Erlangian distribution
   (pdf $f(x)=4\lambda^2 x \exp(-2\lambda x)$) (10)
3. Two-stage hyperexponential distribution
   (pdf $f(x)=\alpha\lambda_1 \exp(-\lambda_1 x)+ (1-\alpha)\lambda_2 \exp(-\lambda_2 x)$) (11)
4. Deterministic distribution
   (pdf $f(x)=\partial(x-1/\lambda)$) (12)

4. RESULTS FROM THE MODEL USING DIFFERENT FAILURE TIME DISTRIBUTIONS

We should keep in mind that the results presented in this section are example on the inevitable consequences that the derived formulas lead to. When using the formulas we should be aware of:
- we have to find our own failure time distributions, which are applicable to our specific problems.
- we have probably different failure time distributions during different phases in the software lifecycle.

The principal results can best be seen in a figure, see Figure 4. This figure is valid for all failure time distributions, where the number of remaining errors decrease. If we consider a number of distributions with the same mean value, the difference in Figure 4 will be the variance, i.e. the length of the arrows in the figure.

Explanation to the figure:
We have estimated the initial number of errors to be N_0 and we would like to test the software product until we reach N_{opt}. This happens at T_e if we do not consider the variations in N_0 and T_e.

But let us consider variations of both N_0 and T_e and see what happens. If we assume that we have estimated N_0 to be somewhere in the interval $(N_0-\Delta N_0, N_0+\Delta N_0)$, then we observe that we have a best and a worst case, remembering that N_0 was the initial number of errors. If we study the best and the worst case and assuming that T_e varies as marked in the figure by the minor arrows, then we obtain an interval, marked in the figure by the big arrow (ΔT), in which T_e lies. This interval is rather big and to get a good estimate of T_e we have to make this interval much smaller, otherwise we are bound to get a software product that is (at least) partly out of control.

We have now looked at the principle result, let us consider some results using some specific failure time distributions. Results are presented in Table 1 for the J-M model and the four distributions mentioned above. We have chosen the parameters for the distributions, so that we have the same mean value until error number N_0 is detected. We have chosen this approach to get an opportunity to compare the variances dependence on the failure time distribution. We have to assume something about the distribution for S_n and N_t, to obtain the length of the arrows, in Figure 4. It is possible to show that S_n and N_t are asymptotically normally distributed, more details on this are found in (10). Assuming a normal distribution for S_n and N_t, we can find confidence intervals, that is we can, with a certain probability, find the length of the arrows in Figure 4. In the table we have chosen a 95 % confidence interval.

TABLE 1 - The mean time (T_e) to reach N_{opt}, and its variance due to the variances in N and the failure time distribution.

Suppose: $N_0=200$ and $\Delta N_0=10\% \cdot N_0=20$

Distribution \ N_{opt}		100	50	20	10	0
Exp. $\lambda=1$	$E(T_e)$	100	150	180	190	200
	$V(T_e)$	100	150	180	190	200
	ΔT	62.5-141.5	107.7-195.6	135.2-227.7	144.4-238.4	153.7-249.1
Erlang. $\lambda=1$	$E(T_e)$	100	150	180	190	200
	$V(T_e)$	50	75	90	95	100
	ΔT	67.6-135.2	114.2-188.1	142.5-219.6	151.9-230.1	161.4-240.6
Hyper.* $\alpha_1=0.5$ $\alpha_1=0.5$ $\lambda_1=5$ $\lambda_2=5/9$	$E(T_e)$	100	150	180	190	200
	$V(T_e)$	232.0	348.0	417.6	440.8	464.0
	ΔT	53.3-152.7	96.0-208.9	122.2-242.2	131.1-253.3	139.9-264.3
Hyper.* $\alpha_1=0.8$ $\alpha_2=0.2$ $\lambda_1=37/10$ $\lambda_2=37/90$	$E(T_e)$	100	150	180	190	200
	$V(T_e)$	1911.8	2867.8	3441.3	3632.5	3823.7
	ΔT	3.35-213.9	32.2-281.7	51.5-321.2	58.3-334.2	65.0-347.1
Deter. $\lambda=1$	$E(T_e)$	100	150	180	190	200
	$V(T_e)$	0	0	0	0	0
	ΔT	80-120	130-170	160-200	170-210	180-220
J-M $\lambda=0.02939$	$E(T_e)$	23.5	46.9	77.6	100.3	200
	$V(T_e)$	5.75	17.1	50.7	104.4	1898.6
	ΔT	15.5-31.6	35.3-58.4	60.1-94.8	76.8-123.7	111.0-288.6

* $\alpha_1 + \alpha_2 = 1$ and we have chosen λ_1 and λ_2 so that $\lambda_1/\lambda_2 = 9$.

5. CONCLUSIONS

We have looked at the situation today during the test phase, (see Figure 4), and something has to be done about it. There are four ways of making the situation better, two primary ways and two secondary ways. The two primary ways deal with the problem of making the test phase shorter and they are:

1. To lower the number of initial errors. This could be done by developing better methods and tools for the design and construction. It is necessary to look at design and construction (programming) as a part of an industry and not as a form of art. The work and documentation have to be standarized.
2. To develop better methods and tools for the test phase, i.e. we need to find more errors in a shorter time and this calls for more systematic testing routines.

The two secondary ways of making the situation better deal with the problem of making the variances smaller and they are:

1. To obtain a better estimate of the initial number of errors. This could be done by developing and applying better methods and tools for the estimation of N_0. We have to find methods and tools that are suitable for our environment and our applications.
2. To decrease the variance in t. This also calls for better methods and tools during testing and more systematic testing routines. If we could decrease the variance in the times between failures, then we automatically obtain a smaller variance for t. In the model this will be shown by the fact that we have to find another failure time distribution. The optimal distribution is a deterministic distribution with a small mean value.

If we do not make the estimated interval for t smaller this could lead to two things, both of them uneconomical, namely:

1. We stay in the test phase longer than necessary, just to be on the safe side.
2. We leave the test phase too early, which could lead to the software product being very expensive to maintain and if the worst comes to the worst, we will end up with an inefficient, unreliable, unmanageable and unmaintainable software product.

6. ACKNOWLEDGEMENT

This project is supported by Telelogic AB and the Swedish Telecommunication Administration, Sweden.

7. REFERENCES

1. Wohlin, C., and Vrana, C., 1985, "Derivation of Formulas for the Quality Constraint Model", Technical Report, Lund Institute of Technology, Lund, Sweden.

2. Vrana, C., and Wallander, A.,1983,"S/W Quality and Complexity - Different Aspects and Measurement Results", Proc. 5th Int. Conf. on SETSS, 121-127.

3. Lennselius, B., 1986, "Software Compexity and its Impact on Different Software Handling Processes", Proc. 6th Int. Conf. on SETSS.

4. McCabe, T., 1976, IEEE Trans. on Software Eng., Vol. SE-2, No 4, 308-320

5. Halstead, M., 1977, "Elements of Software Science", Elsevier North Holland Publishing Co., New York, USA.

6. Jelinski, Z., and Moranda, P., 1973, Statistical Computer Performance Evaluation, Academic Press, 465-484.

7. Goel, A., 1983, "A Guidebook for Software Reliability Assessment", Syracuse University, Syracuse, USA.

8. Feller, W., 1957, "An Introduction to Probability Theory and its Applications", John Wiley and Sons, New York, USA.

9. Kleinrock, L., 1976, "Queueing Systems, Vol 1. Theory", John Wiley and Sons, New York, USA.

10. Cox, D.R., 1962, "Renewal Theory", Methuen Co, London, England.

11. Cox, D.R., and Miller, H.D., 1965, "Stochastic Processes", Methuen Co, London, England.

12. Wohlin, C., 1985, "An Analytical Comparison between two Software Reliability Models", Technical Report, Lund Institute of Technology, Lund, Sweden.

13. Karlsson, J., and Stavenow, B., 1980, "Methods for Numerical Inversion of Laplace- and z-transform", Technical Report, Lund Institute of Technology, Lund, Sweden.

14. Wohlin, C., 1986, "Software Testing and Reliability for Telecommunication Systems", Proc. 6th Int. Conf. on SETSS.

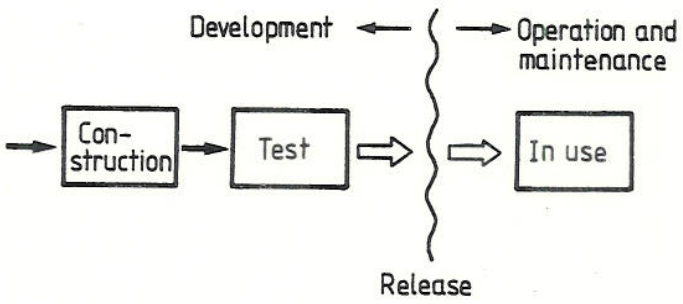

Figure 1 A simple software lifecycle

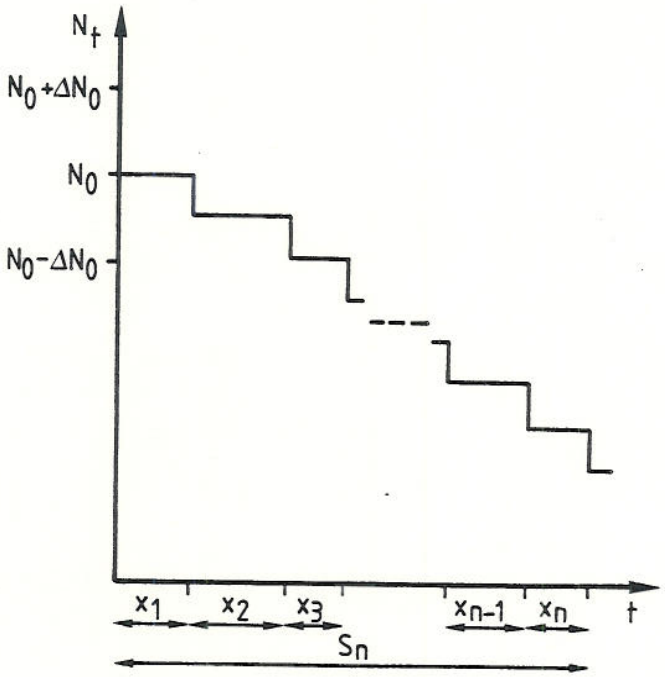

Figure 2 An example of how the number of remaining errors decreases

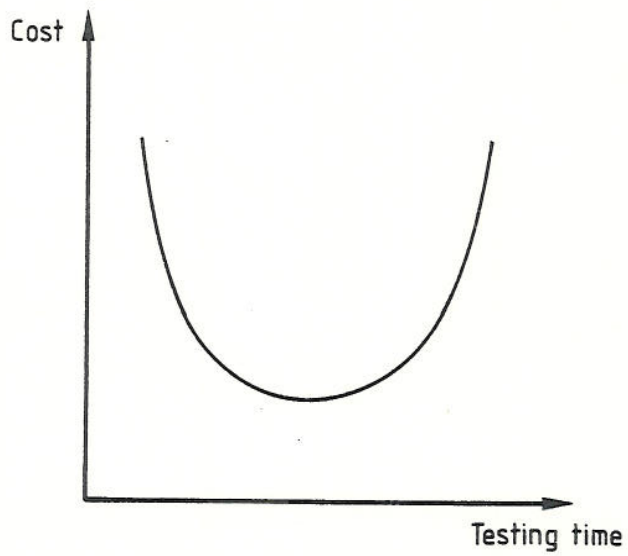

Figure 3 The principal relationship between testing time and the cost for correction of errors

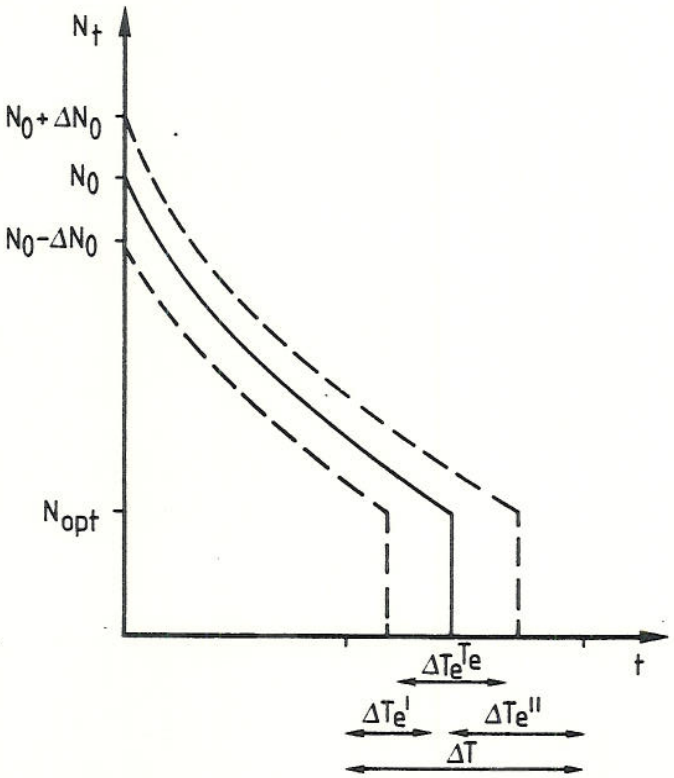

Figure 4 The principal result

LUND INSTITUTE OF TECHNOLOGY
Dept of Communication Systems

CODEN:LUTDEX(TETS-7097)
/1-7/(1985) & local 10

Derivation of Formulas for the Quality Constraint Model

by

Claes Wohlin, Lund Institute of Technology

and

Ctirad Vrana, Telelogic AB

Sweden

Lund in october, 1985

1. INTRODUCTION

The model is introduced in Wohlin and Vrana (1).
The problem is to derive formulas for:

- the mean and variance of the number of remaining errors at a specific time
- the mean and variance for the time until a specific number of errors remain

when the failure time distribution is known.

The following notation is used in (1):

* $F(t)=P(\tilde{t}\leq t)$ - the probability distribution function (PDF)
* $f(t)=dF(t)/dt$ - the probability density function (pdf)
* $F^*(s)$ - the Laplace transform of f(t)
* $P(z)$ - the z-transform of $p_k=P(\tilde{n}=k)$
* $E(X)=\bar{x}$ - the mean value of the stochastic variable X
* $V(X)$ - the variance of the stochastic variable X
* $E(X^k)=\overline{x^k}$ - the k:th moment of the stochastic variable X
* N_0 - the initial number of errors, i.e. when the test phase begins
* ΔN_0 - the uncertainty in the estimation of the initial number of errors
* N_t - the number of remaining errors at time t, (pdf $g_k(t)$)
* X_i - the time between error number (i-1) and error number i, (pdf $f_i(x)$)
* S_n - the time to error number n, (pdf $k_n(t)$)

With these definitions we find:

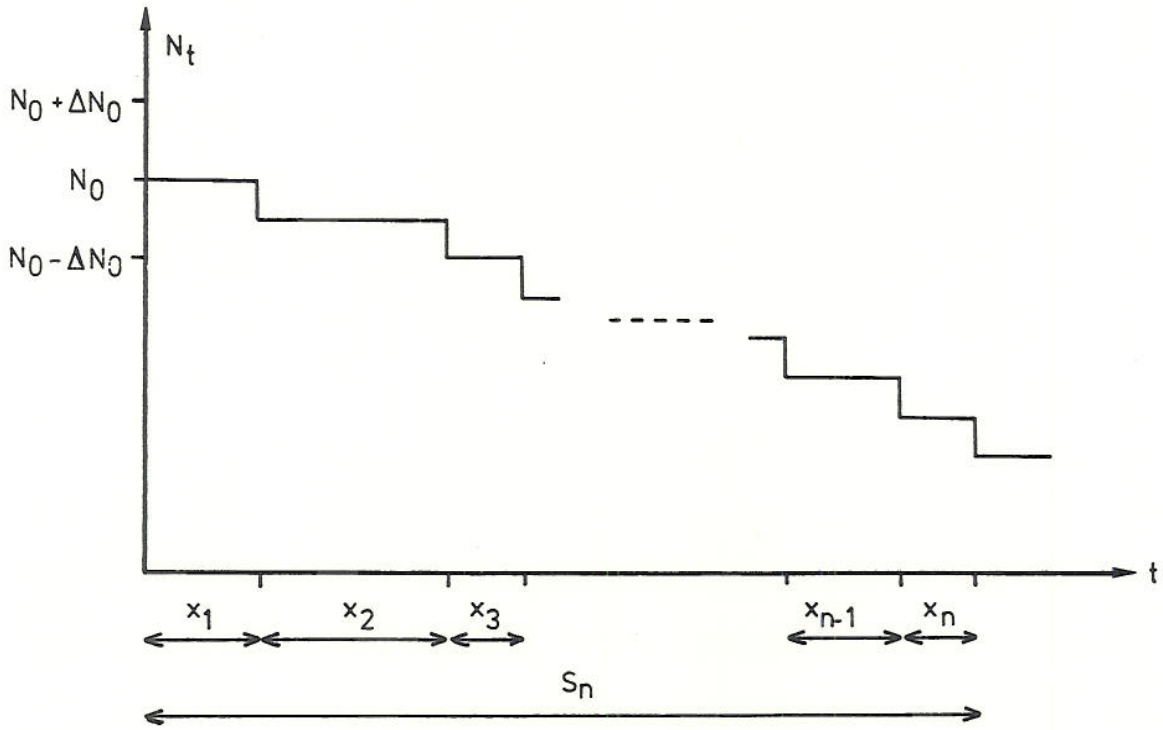

Figure 1. An example of how the number of remaining errors decrease.

The derivations will be made with the following three assumptions:

1. The initial number of errors, when the test phase is started, are known through estimation.
2. The failure time distribution is known.
3. If a failure is found, it is corrected and no new failures are introduced.

For comments to the assumptions, see reference (1).

2. SOME DEFINITIONS

Let us first look at some definitions:

Laplace transform (for continuous stochastic variables):

$$F^*(s) = \int_{x=0}^{\infty} e^{-sx} f(x)dx \qquad (1)$$

z-transform (for discret stochastic variables):

$$P(z) = \sum_{k=0}^{\infty} p_k z^k = \sum_{k=0}^{\infty} P(X=k) z^k \qquad (2)$$

The k:th moment for a stochastic variable:

Continuous stochastic variable:

$$E(X^k) = \int_{x=0}^{\infty} x^k f(x)dx \tag{3}$$

Discret stochastic variable:

$$E(X^k) = \sum_{k=0}^{\infty} x^k p_k \tag{4}$$

Variance:

$$V(X) = E(X^2) - E(X)^2 \tag{5}$$

The definitions of the transforms give us the moments for the stochastic variable as:

Continuous stochastic variable:

First moment (mean value):

$$E(X) = - \lim_{s \to 0} \frac{\partial F^*(s)}{\partial s} \tag{6}$$

Second moment:

$$E(X^2) = \lim_{s \to 0} \frac{\partial^2 F^*(s)}{\partial s^2} \tag{7}$$

The k:th moment:

$$E(X^k) = \lim_{s \to 0} (-1)^k \frac{\partial^k F^*(s)}{\partial s^k} \tag{8}$$

Discret stochastic variable:

First moment (mean value):

$$E(X) = \lim_{z \to 1} \frac{\partial P(z)}{\partial z} \tag{9}$$

Second moment:

$$E(X^2) = \lim_{z \to 1} \left(\frac{\partial^2 P(z)}{\partial z^2} + \frac{\partial P(z)}{\partial z} \right) \tag{10}$$

3. DERIVATION

We are now able to derive the desired formulas.
For fuller details on the theories used in this section, we would like to recommend Cox (2).

3.1 Derivation of the mean and variance of the number of remaining errors at a specific time

We introduce the time-dependent z-transform for N_t as

$$G(t,z) = \sum_{k=0}^{N_0} P(N_t=k) z^k \tag{11}$$

but

$$P(N_t=k) = P(N_t>k-1) - P(N_t>k) \tag{12}$$

and we realize, from Figure 1, that

$$P(N_t>k) = P(S_n>t) \qquad \text{because} \quad k=N_0-n \tag{13}$$

=>

$$\begin{aligned} P(N_t=k) &= P(N_t=N_0-n) = P(N_t>N_0-n-1) - P(N_t>N_0-n) = \\ &= P(S_{n+1}>t) - P(S_n>t) = K_n(t) - K_{n+1}(t) \end{aligned} \tag{14}$$

=>

$$\begin{aligned} G(t,z) &= \sum_{n=0}^{N_0} (K_n(t)-K_{n+1}(t)) z^{N_0-n} = \\ &= K_0(t) z^{N_0} + \sum_{n=1}^{N_0} z^{N_0-n}(1-z)K_n(t) \end{aligned} \tag{15}$$

but knowing that

$$K_0(t) = P(S_0 \leq t) = 1 \tag{16}$$

and then taking the Laplace transform on t, we obtain

$$G^*(s,z) = \frac{1}{s} \left(z^{N_0} + \sum_{n=1}^{N_0} z^{N_0-n}(1-z)K_n^*(s)\right) \tag{17}$$

It is easily seen that

$$S_n = \sum_{i=1}^{n} X_i \tag{18}$$

and it can be shown that a sum of stochastic variables form a product, when applying the Laplace transform on the sum.

=>

$$K_n^*(s) = \prod_{i=1}^{n} F_i^*(s) \tag{19}$$

We then finally obtain

$$G^*(s,z) = \frac{1}{s}\left(z^{N_0} + \sum_{n=1}^{N_0} z^{N_0-n}(1-z)\prod_{i=1}^{n} F_i^*(s)\right) \tag{20}$$

Remembering the moment generating properties of the z-transform, we obtain the Laplace transforms for the first two moments of the number of remaining errors.
Taking the first two partial derivatives and then letting $z \to 1$, we obtain

$$\lim_{z\to 1} \frac{\partial G^*(s,z)}{\partial z} = \frac{1}{s}\left(N_0 - \sum_{n=1}^{N_0}\prod_{i=1}^{n} F_i^*(s)\right) \tag{21}$$

and

$$\lim_{z\to 1} \frac{\partial^2 G^*(s,z)}{\partial z^2} = \frac{1}{s}\left(N_0(N_0-1) - 2\sum_{n=1}^{N_0}(N_0-n)K_n^*(s)\right) \tag{22}$$

If we let $H(t)=E(N_t)$ and $M(t)=E(N_t^2)$ with Laplace transforms $H^*(s)$ and $M^*(s)$, we get that

$$H^*(s) = \lim_{z\to 1} \frac{\partial G^*(s,z)}{\partial z} = \frac{1}{s}\left(N_0 - \sum_{n=1}^{N_0}\prod_{i=1}^{n} F_i^*(s)\right) \tag{23}$$

and

$$M^*(s) = \lim_{z\to 1}\left(\frac{\partial^2 G^*(s,z)}{\partial z^2} + \frac{\partial G^*(s,z)}{\partial z}\right) = \frac{1}{s}\left(N_0^2 - \sum_{n=1}^{N_0}(2(N_0-n)+1)\prod_{i=1}^{n} F_i^*(s)\right) \tag{24}$$

If we invert these two transforms, we can obtain the variance as

$$V(N_t) = E(N_t^2) - E(N_t)^2 \tag{25}$$

3.2 Derivation of the mean and variance for the time until a specific number of errors have occurred (or are remaining)

We know from (19), that

$$K_n^*(s) = \prod_{i=1}^{n} F_i^*(s)$$

Using the moment generating properties of the Laplace transform, we immeditely find the first two moments as

$$E(S_n) = - \lim_{s \to 0} \frac{\partial K_n^*(s)}{\partial s} \tag{26}$$

and

$$E(S_n^2) = \lim_{s \to 0} \frac{\partial^2 K_n^*(s)}{\partial s^2} \tag{27}$$

Let us take the derivatives on K*(s)

=>

$$\frac{\partial K_n^*(s)}{\partial s} = \sum_{i=1}^{n} \frac{\partial F_i^*(s)}{\partial s} \prod_{\substack{j=1 \\ j \neq i}}^{n} F_j^*(s) \tag{28}$$

and

$$\frac{\partial^2 K_n^*(s)}{\partial s^2} = \sum_{i=1}^{n} \frac{\partial^2 F_i^*(s)}{\partial s^2} \prod_{\substack{j=1 \\ j \neq i}}^{n} F_j^*(s) + \sum_{i=1}^{n} \frac{\partial F_i^*(s)}{\partial s} \cdot \sum_{\substack{k=1 \\ k \neq i}}^{n} \frac{\partial F_k^*(s)}{\partial s} \prod_{\substack{j=1 \\ j \neq i \\ j \neq k}}^{n} F_j^*(s) \tag{29}$$

From this follows that

$$E(S_n) = \sum_{i=1}^{n} \bar{x}_i \tag{30}$$

and

$$E(S_n^2) = \sum_{i=1}^{n} \overline{x_i^2} + \sum_{i=1}^{n} \bar{x}_i \sum_{\substack{k=1 \\ k \neq i}}^{n} \bar{x}_k = \sum_{i=1}^{n} \left(\overline{x_i^2} + 2\bar{x}_i \sum_{k=i+1}^{n} \bar{x}_k \right) \tag{31}$$

We can now easily obtain the variance as

$$V(S_n) = E(S_n^2) - E(S_n)^2 \tag{32}$$

In the special case, when $f_i(x)=f(x)$ we obtain simpler formulas for $E(S_n)$ and $E(S_n^2)$. We find that

$$E(S_n) = n\bar{x} \tag{33}$$

and

$$E(S_n^2) = \sum_{i=1}^{n} (\bar{x}_i^2+2\bar{x}^2(n-i)) = n\overline{x^2}+2n^2\bar{x}^2-2\bar{x}^2\sum_{i=1}^{n} i =$$

$$= n\overline{x^2}+2n^2\bar{x}^2-\bar{x}^2 n(n+1) = n(\overline{x^2}-\bar{x}^2)+n^2\bar{x}^2 \tag{34}$$

=>

$$V(S_n) = nV(x) \tag{35}$$

This concludes the derivation of formulas for the Quality Constraint Model (QCM) presented in (1).

4. REFERENCES

1. Wohlin, C., and Vrana, C., 1986, "A Quality Constraint Model to be Used During the Test Phase of the Software Lifecycle", Proc. 6th Int. Conf. on Software Engineering for Telecommunication Switching Systems.

2. Cox, D.R., 1962, Renewal Theory, Methuen Co., London, England.