Specification and Allocation of Reliability and Availability Requirements

Per-Erik Hagmark, PhD, Tampere University of Technology
Seppo Virtanen, PhD, Tampere University of Technology

Key Words: availability, failure tendency, generalized fault tree, product requirements, repair time, simulation.

SUMMARY & CONCLUSIONS

Our model for allocation of requirements is based on a generalized fault tree approach, where the TOP represents the product to be designed. The other parts of the fault tree represent entities, which affect essentially the failure tendency and the repair time of the product. Relations between parts are modeled by two mechanisms. The "gates" determine the partly logical and partly stochastic propagation of faults (primary states). The "strategies" define other relations between TOP and the deepest entities. A consequence of the strategies is that two types of "waiting" (secondary states) can occur.

Customer and/or manufacturer data influences the design of product reliability, availability and repair time. The proposed methods can deal with quite different types of requirements. Requirements related to failure tendency can involve number of failures, time between failures, reliability and availability as a function of age, or data concerning first failure. Requirements related to product's repair time again could involve mean time to repair, standard deviation, minimum repair time (0%), and maximum repair time (with corresponding quantile %).

The allocation of the failure tendency of a gate (entity) down to its input entities is guided by assessing "importance" and "complexity". Importance takes into account customer's perspective and complexity represent the technical standpoint. The aim is that the more important an entity is, the less it is allowed to fail, and the more complex an entity is, the more it is allowed to fail. The repair time allocation again is based on a direct assessment of repair time ratios between the input entities. The failure tendency and the repair time of an entity can also be locked, whereas the designer can focus only on the unlocked entities.

The requirements for TOP are summarized in two "dependability functions" - one for failure tendency and one for repair time. A stepwise allocation process downward in the fault tree leads gate by gate to equivalent dependability functions for other entities. These functions are in every stage tested via simulation and comparison to TOP requirements.

The last simulation confirms the final dependability of entities, especially of those to which attention will be paid in a later design process. The simulation produces also a complete list of events, states of entities, their duration, etc. This "logbook" is of course detailed raw material for various supplemental calculations, conclusions, and even further programming.

1. *INTRODUCTION*

This paper presents, a computer-supported method for specifying reliability, repair time, and availability requirements for a product and allocating them into the product's design entities. The general term "entity" can stand for function, system, equipment, mechanism, or any kind of part.

The developed method is one of the main results from the research project, which lasted about nine years and was carried out by Tampere University of Technology. Since 1996 eleven Finnish companies have participated in the research project, which objective was to develop computer supported probabilistic based method for the development of the equipment's and systems' reliability and safety. The participating companies are both manufacturers and users of equipment, in metal, energy, process and electronics industries. Their products and systems have to correspond to high safety and reliability demands. The research project was completed in February 2005.

The corresponding software (RAMalloc) forces the designer to work out which customer and manufacturer needs should be used to determine the product's quantitative reliability, availability and repair time goals, early in the design stage. Rather detailed product specific requirements can be modeled. For example, there is from both the customer's and the manufacturer's perspective, an opportunity to accept a different probability of failure during the burn-in phase than after it, or there is possibility to accept different failure tendencies during the warranty and the post warranty periods. With the software, the requirements can be allocated to functions, systems, mechanisms or any parts as the design work proceeds.

The effect of reliability, availability and repair time requirements defined by the customer and manufacturer on the known technical solution of a product can be demonstrated with the developed method and software. This connection is important in order to avoid promising something that cannot be achieved or something, which is very expensive to achieve. The applicability of the developed methods and software has been tested in companies that have been involved in the research project. At this moment, most of the participating

1-4244-0008-2/06/$20.00 (C) 2006 IEEE

companies use the method and the software in their products' and systems' requirements management.

Figure 1 describes schematically our allocation procedure. The details will be explained throughout the paper. Some details can also be found in [1] of the same authors, but it is worth noting that some concepts and definitions differ or cannot even be found there.

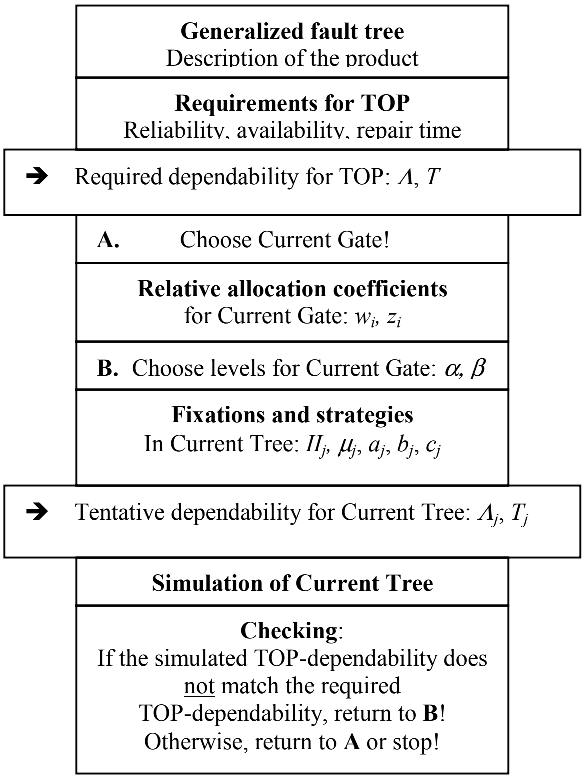
Generalized fault tree
Description of the product
Requirements for TOP
Reliability, availability, repair time
➔ Required dependability for TOP: Λ, T
A. Choose Current Gate!
Relative allocation coefficients
for Current Gate: w_i, z_i
B. Choose levels for Current Gate: α, β
Fixations and strategies
In Current Tree: Π_j, μ_j, a_j, b_j, c_j
➔ Tentative dependability for Current Tree: Λ_j, T_j
Simulation of Current Tree
Checking:
If the simulated TOP-dependability does not match the required TOP-dependability, return to B!
Otherwise, return to A or stop!

Figure 1. The Allocation Process

2. *MODELING REQUIREMENTS*

We present the parameters and the mathematical models used for the critical customer data that influences product reliability, availability, and repair time. The cumulative working time of the product will be called *age*, and repair time is not included in age.

2.1 *Failure Tendency*

We assume that the customer product requirements for reliability and the number of failures can be concentrated in the following set of parameters:

Age at the end of the burn in period	t_a
Age at the end of the warranty period	t_b
Age at the end of the useful life period	t_d
Length of age period (for Rel below)	t_c
Reliability in age periods $(t, t+ t_c] \subseteq (t_a, t_d]$	*Rel*
A parameter for warranty period	*s*

The last condition means that the expected number of failures in the warranty period $(0,t_b]$ must not exceed s times the number of failures in an equally long post-warranty period $(t_b, 2t_b]$.

Our failure tendency model $\Lambda(t)$ denotes the expected cumulative number of failures during the age period $(0, t]$. We assume the age at failures follows a non-homogeneous Poisson process (NHPP) with intensity $\Lambda'(t)$. Practically this compromise means that the failure tendency of a repaired object is statistically the same as it was just before it failed. (For more details on NHPP we refer to [2].)

In terms of this model, the requirements adopt the following inequality form:

$$\Lambda(t+t_c)-\Lambda(t)\le -\ln(Rel) \quad \text{for} \quad t_a \le t \le t_d - t_c \tag{1}$$

$$\frac{\Lambda(t_b)}{\Lambda(2t_b)-\Lambda(t_b)} \le s \tag{2}$$

More explicitly, the following expression involves the inequalities (1) & (2):

$$\Lambda(t)=\lambda\cdot t+a_1\cdot\left(1-Z\left(1-\frac{t}{t_b}\right)^b\right)+a_2\cdot Z\left(\frac{t}{2t_b}-1\right)^c \tag{3}$$

$$Z(t)=\begin{cases}0 & for \quad t\le 0\\ t & for \quad t>0\end{cases},$$

where

$$a_1=\frac{\lambda\cdot t_b\cdot(s-1)}{1-2^{-b}(s+1)}, \tag{4}$$

$$a_2=-\frac{\lambda\cdot t_c+\ln(Rel)}{\left(\frac{t_d}{2t_b}-1\right)^c-\left(\frac{t_d-t_c}{2t_b}-1\right)^c}, \tag{5}$$

and $t_a + t_c < 2t_b < t_d - t_c$. Expression (3) fills the needs of allocation, and the choice of the shape parameters λ, b, c is supported by the corresponding software. (More details can be found in [1], pp.89-93, though the Λ-expression there is not exactly the same.)

2.2 *Repair time*

Concerning repair time we assume the following model parameters can be extracted from the customer requirements:

Minimum repair time (0-quantile)	$t_{\min}$
Mean time to repair	μ
Q-quantile (often Q=0.95)	$T(Q)$

To this data, we apply a modified Beta model. The upper bound of the domain $(t_{\min}, t_{\max})$ will be dynamically connected to the variance $\sigma^2 > 0$:

$$t_{\max}= \mu + \left(\mu - t_{\min}\right)\cdot \max\left\{20, \left(\frac{\sigma}{\mu - t_{\min}}\right)^2 + 1.0001\right\}, \quad (6)$$

Now, if $betf\left(x,p,q\right)$ denotes the standard cumulative Beta probability distribution on the interval (0,1) with parameters

$$p = \left(\mu - t_{min}\right)\cdot s\,, \qquad q = \left(t_{max} - \mu\right)\cdot s\,, \quad (7)$$

$$s = \frac{\mu\cdot\left(t_{\min} + t_{\max}\right) - t_{\min}\cdot t_{\max} - \sigma^2 - \mu^2}{\sigma^2\cdot\left(t_{\max} - t_{\min}\right)}\,, \quad (8)$$

then the quantile function to be used in simulation takes the form

$$T\left(u\right) = t_{\min} + \left(t_{\max} - t_{\min}\right) + betf^{-1}\left(u, p, q\right),\ 0 \le u \le 1 \quad (9)$$

The standard deviation σ can of course be a parameter of its own, but here it serves rather as an auxiliary parameter, via which we iterate the desired value to the quantile $T(Q)$. Formula (6) is formulated for the purpose of being flexible enough for applications, where the ratio $T(0.95)/\mu$ can be large. A suitably chosen σ can give ratios up to 8...10. (In [1], p. 93, we employed a less flexible Weibull model.)

2.3 *Availability*

The customer requirements are often described in terms of availability. For such cases, our model and software offers the following parameters:

Age at the end of the warranty period	t_b
Age at the end of the useful life period	t_d
Average availability in age period $(0, t_b]$	A_b
Average availability in age period $(t_b, t_d]$	A_{bd}
Availability at age $t = 0$	A_0
Availability at age $t = 2\ t_b$	A_{m}

The notion of availability is a combination of failure tendency (2.1) and repair time (2.2). A flexible definition of availability in an age period $(t, s]$ is given by

$$A\left(t,s\right) = \left(1 + \mu\cdot\frac{\Lambda\left(s\right) - \Lambda\left(t\right)}{s - t}\right)^{-1}. \quad (10)$$

Note that unavailability caused by any kind of planned stops is not included here. When s approaches t, we obtain the point wise availability in the form:

$$A\left(t\right) = \left(1 + \mu\cdot\Lambda'\left(t\right)\right)^{-1} \quad (11)$$

Combining formulas (10) and (11) with the availability requirements, the requirements are automatically transformed to the corresponding failure tendency given by formula (3), where this time

$$a_1 = \frac{2\cdot t_b}{\mu\cdot b}\cdot\left(\frac{1}{A_0} - \frac{1}{A_m}\right) \quad (12)$$

$$a_2 = \left[\frac{t_d - t_b}{\mu}\cdot\left(\frac{1}{A_{bd}} - \frac{1}{A_m}\right) - a_1\cdot 2^{-b}\right]\cdot\left(\frac{t_d}{2t_b} - 1\right)^{-c} \quad (13)$$

and $2t_b < t_d$, $A_0 < A_b < A_{0\mathrm{m}}$. Again, the corresponding software supports the choice of the shape parameters b and c. (Availability requirements were not considered in [1].)

2.4 *Elementary Requirements*

The failure tendency function $\Lambda(t)$ for the product must sometimes be constructed using other types of data. Still assuming NHPP, the following possibilities can be mentioned. If the requirements in an age period $(t_1, t_2]$ are given in terms of average number of failures v, or average failure rate λ, or reliability R, then $\Lambda\left(t_2\right) - \Lambda\left(t_1\right) = v = \lambda\cdot\left(t_2 - t_1\right) = -\ln(R)$. Further, the hazard function for the first failure age is given by the derivative $\Lambda'(t)$, and the reliability function is $e^{-\Lambda(t)}$.

3. *THE ALLOCATION PRINCIPLE*

The customer requirements for reliability, availability and repair time of the product ended in the *dependability functions*: $\Lambda(t)$ for failure tendency, and $T(u)$ for repair time, (3), (4), (5), (9), (12), (13). Allocation of requirements can be carried out gate by gate in the frame of a *generalized fault tree*, the TOP of which represents the product.

3.1 *The Structure of Failing*

The primary *state* of TOP, either 1 (failed), or 0 (non-failed), is the value of a *gate* (a stochastic Boolean function), whose inputs are the equivalent states of the first-level entities. The first-level entities in turn can be divided into their own input entities through gates, and so on. Construction of the fault tree, consisting of entities and corresponding gates, is continued to a depth needed for the aim in question.

The mechanism of a gate is partly logical and partly stochastic. A gate is characterized by giving the data row

$$\left(ID \quad k \quad m \quad P \quad \pm I_1 \quad \pm I_2 \quad .... \quad \pm I_n\right). \quad (14)$$

Here ID is the number of the entity and the corresponding gate, $0 \le k \le m \le n$, $0 < P \le 1$, I_{i} are the ID-numbers of the input entities, and a minus–sign denotes that the input is first negated. Now, if

$$k \le \text{the sum of } \{0,1\}\text{-inputs} \le m, \quad (15)$$

then the gate ID adopts the state 1 with probability P, otherwise its state is 0.

3.2 *Preliminary Description of the Allocation Step*

In the first "allocation step", the *current gate* (entity) is TOP, and the *current tree* consists of TOP and the first level entities. Dependability functions (Λ, T) for the first level entities are then found through an iterative simulation process. The criterion is that the simulated TOP-dependability fulfills the requirements.

In the next allocation step the new current gate can be some of the first level entities (if it is a gate). The inputs of this entity are added to the current tree, and their dependability functions (Λ, T) are determined.

This process goes gate by gate deeper in the fault tree. In the general allocation step, a new current gate *ID* (14) is first chosen. This is possible if its dependability functions, $\Lambda_{ID}(t)$ and $T_{ID}(u)$, have been constructed in an earlier allocation step. The corresponding new current tree consists of the previous current tree *and* the input entities of the current gate. With repeated simulations of the current tree, dependability functions (Λ_i, T_i) for the gate's inputs I_i are obtained such that the simulated TOP-dependability fulfills the requirements. The general allocation step is thus described with the formula

$$\Lambda_{ID}(t),\, T_{ID}(u) \;\rightarrow\; \Lambda_i(t),\, T_i(u), \quad i=1, 2,\ldots, n. \tag{16}$$

The details will be clarified in the following sections, 4 & 5.

4. *RELATIVE ALLOCATION COEFFICIENTS*

The first part of the allocation step (16) is to build relative allocation coefficients for failure tendency and repair time of the inputs of the current gate.

4.1 *Importance*

The allocation of failure tendency starts by assessing such "importance" that forbids failures of the current gate. When the gate fails, it might cause some kind of damage, for example,

Property damage	$100{\cdot}D$ %
Environmental damage	$100{\cdot}E$ %
Human damage	$100{\cdot}F$ %
Business damage	$100{\cdot}G$ %
	$(D+E+F+G = 1)$

The designer has the freedom to choose these types of importance factors, and even different types for different gates.

Each type of damage must be shared among the input entities of the gate. Let input I_i be responsible for

$100{\cdot}d_i$ %	of "property damage"
$100{\cdot}e_i$ %	of "environmental damage"
$100{\cdot}f_i$ %	of "human damage"
$100{\cdot}g_i$ %	of "business damage"

$(d_i,\, e_i, f_i,\, g_i \geq 0,\;\; \Sigma d{=}\Sigma e{=}\Sigma f{=}\Sigma g = 1)$

An approximate assessment of the *relative importance* x_i of the input entity I_i is given simply by matrix multiplication:

$$\begin{pmatrix} x_1 \\ x_2 \\ \ldots. \\ x_n \end{pmatrix} = \begin{pmatrix} d_1 & e_1 & f_1 & g_1 \\ d_2 & e_2 & f_2 & g_2 \\ \ldots. & \ldots. & \ldots. & \ldots. \\ d_n & e_n & f_n & g_n \end{pmatrix} \cdot \begin{pmatrix} D \\ E \\ F \\ G \end{pmatrix} \tag{17}$$

Strictly speaking, this model coming from elementary conditional probability is exact only if the possible damage types are non-overlapping and the gate is XOR (cf. (14) & (15): a=b=1, P=1). Notwithstanding, we accept (17) as a general model.

4.2 *Complexity*

We consider such "complexity" that guides the designer to allocate more failures to those inputs, which more probable cause the gate's failure. The complexity of a gate can consist, for example, of the following types:

Number of parts	$100{\cdot}A$ %
Level of human activities	$100{\cdot}B$ %
Level of state-of-art	$100{\cdot}C$ %
	$(A+B+C = 1)$

The designer has again the freedom to choose these types of complexity factors, and even different types for different gates. Experts and experience are here extremely valuable, since these ratios are especially difficult and time consuming to assess. The case dependence is often tricky, and no common unit for measuring exists. (The software can also split up the assessment in pair comparisons.)

Each complexity type is then shared among the inputs of the gate. Suppose the input entity I_i include

$100{\cdot}a_i$ %	of "number of parts"
$100{\cdot}b_i$ %	of "human activities"
$100{\cdot}c_i$ %	of "state-of-art"

$(a_i,\, b_i,\, c_i \geq 0,\; \Sigma a = \Sigma b = \Sigma c = 1)$

If y_i denotes the *relative complexity* of the input entity I_i, then, as in (17), heuristic probabilistic interpretation suggests the approximate formula

$$\begin{pmatrix} y_1 \\ y_2 \\ \ldots. \\ y_n \end{pmatrix} = \begin{pmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ \ldots. & \ldots. & \ldots. \\ a_n & b_n & c_n \end{pmatrix} \cdot \begin{pmatrix} A \\ B \\ C \end{pmatrix} \tag{18}$$

4.3 *Allocation Coefficients for the Current Gate*

Importance forbids failures, i.e., the bigger x (17) for an input entity, the fewer failures it is allowed to cause. Complexity again permits failures, i.e., the bigger y (18), the more failures. These heuristic principles can of course be combined and quantified in many ways. Our suggestion is:

$$w_i = \frac{y_i^{\,g(2-\tau)}}{x_i^{\,g(\tau)}} \quad \text{where} \quad g(\tau) = \begin{cases} \tau & for \quad 0 \le \tau < 1 \\ 1 & for \quad 1 \le \tau \le 2 \end{cases} \tag{19}$$

The coefficients, *w,* describe the relative amounts of failures associated with the inputs of the gate. They are further normalized by $\Sigma w = 1$ and called *failure tendency coefficients.* The parameter, τ, $0 \le \tau \le 2$, which weights importance against complexity, is also to be chosen by the designer. Examples:

	τ	$g(\tau)$	$g(2-\tau)$	w
Importance omitted	0	0	1	y
Complexity omitted	2	1	0	$1/x$
Equally weighted	1	1	1	y/x

Finally, the *repair time coefficients,* $z_1, z_2, \ldots, z_n$ ($\Sigma z = 1$), reflect directly the ratios between the repair times of the input entities. We suppose they can be formed by direct assessment.

(Some additional details can be found in [1], pp. 94-96. For a simpler version of (19) see p. 97.)

5. *SIMULATION OF DEPENDABILITY*

We come to the second part of the allocation step (16). New parameters and concepts (level parameters, fixations, and strategies) are defined. Then iterative simulation of the current tree leads to dependability functions (Λ, T) for the input entities of the current gate.

5.1 *Level Parameters and Fixed Dependability*

Above we described the construction of relative allocation coefficients, *w, z,* for the input entities of the current gate. For adjustment of the general level we need two *level parameters*: α>0 for failure tendency, and β>0 for repair time. The dependability functions of the inputs I_i of the current gate are now defined:

$$\Lambda_i(t) = \alpha \cdot w_i \cdot \Lambda_{ID}(t) \tag{20}$$

$$T_i(u) = \beta \cdot z_i \cdot T_{ID}(u) \tag{21}$$

The values of α and β are so far tentative, but they will be determined in the iterative simulation process (section 5.2). Note the following obvious consequence of (20) and (21): All dependability functions are (vertically) scaled versions of the equivalent functions for TOP, $\Lambda(t)$, $T(u)$. (The principle for repair time (21) was different in [1], p.94.)

The deepest entities of the current tree are called (current) *basic parts* (BP). A useful additional feature of our model is the possibility of locking the failure tendency and/or the repair time for *any* current BP. The data for *fixation* is as follows:

Average number of failures during (0, t_d]	Π_j
Mean time to repair	μ_j

Again, the shapes will be inherited from TOP, so *all* dependability functions are (vertically) scaled versions of $\Lambda(t)$ and $T(u)$.

5.2 *Strategies, Waiting States and Simulation*

In addition to the fault logic, certain interrelations between TOP and current BPs can be modeled by fixing the following *operation strategies* (restrictions) for each BP:

This BP cannot be repaired if TOP is running	(a_j=1)
This BP is not working if TOP is not running	(b_j=1)
TOP will not be started if this BP is still failed	(c_j=1)

The strategies *a, b, c* are in principle independent of the fault logic, but they will be applied only if the logic gives room. Further, the strategies imply two new states, the (secondary) *waiting states.* The complete set of possible states for an entity is thus:

0	non-failed and running
0.5	non-failed and waiting for start
1	failed and repair is going on
1.5	failed and waiting for repair

The simulation of the current tree can now be performed in the TOP age interval $t \in (0, t_d]$. All BPs of the current tree have dependability functions – either from the current allocation step, or from some earlier allocation step. The dependability functions form the basis for variate generation of time to failure and time to repair. (For principles of simulation we refer to [3].) Each strategy is checked every time the state of TOP or a BP changes.

Here is a description of the first phase of simulation. All BPs are working (state 0), and random time to failure is generated for each BP. When the first BP fails, the state of the current tree is generated up to TOP according to Section 3.1. If TOP is still working (0) and *a*=1, the failed BP takes the "waiting-for-repair" state 1.5, otherwise (TOP failed or *a*=0) the failed BP takes the state 1, and repair time can be generated immediately. Besides, if TOP failed, then those working (non-failed) BP, whose *b*=1, must go into the "waiting-for-start" state (0.5). Etc.

5.3 *Checking Requirements and Continuing Allocation*

This is the final part of the allocation step (16). The simulated dependability of TOP is now compared to the required dependability. Comparisons are made with the simulated and required versions of the following figures:

Rel, *s*	(section 2.1)
μ, $T(Q)$	(section 2.2)
A_b, A_{bd}, A_0, A_m	(section 2.3).

The simulated and the required versions of failure tendency $\Lambda(t)$ and availability $A(t)$ are also compared as *entire* functions. Figure 2 shows the comparison concerning $\Lambda(t)$.

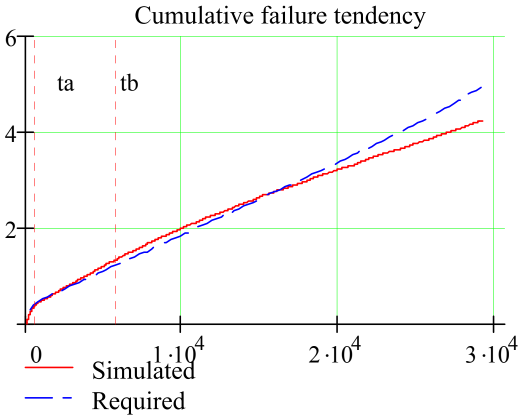

Figure 2. Comparison for $\Lambda(t)$

The values of the level parameters, α, β, must usually be iterated a few times, until a satisfactory situation is achieved (if possible at all). The allocation of the current gate is now complete, and a new allocation step can begin by selecting a new gate to be allocated (Figure 1), and returning to section 4.

6. *FINAL RESULTS*

The last simulation confirms the dependability of TOP and the BPs of the last current tree. The requirements for each BP are now summarized in two dependability functions (Λ, T). The simulation produces also a complete list of events, states of entities, their duration, etc. Various and quite detailed supplemental calculations and conclusions can of course be extracted from this "logbook". At this moment, the RAMAlloc software provides for example the following results:

(a) For TOP and BPs:

- Total operation time
- Total waiting-for-start time
- Total repair time
- Total waiting-for-repair time
- Total number of failures
- Availability
- MTTF
- MTTR
- TTR(95%)

(b) For gates:

- Total repair time
- Total number of failures

(c) For TOP or any BP, in a specified age interval (t_1, t_2]:

- Number of failures
- Number of failures, 95% quantile
- Reliability
- Availability

These results are especially significant for certain BPs, the "design entities", since attention will be paid to these in a later design process concerning the technical solution. (For this purpose, software has been developed, too.)

REFERENCES

1. S. Virtanen, P-E. Hagmark, "Allocation of Dependability Requirements in Power Plant Design", Chapter 4 in *Case Studies in Reliability and Maintenance* (Eds. W. R. Blischke, D. N. P. Murthy), Wiley, 2003, pp 85 – 107.
2. I. Gertsbakh, *Reliability Theory With Applications to Preventive Maintenance*, Springer, 2000, pp 187-190.
3. R. Y Rubinstein, B. Melamed, *Modern Simulation and Modeling*, Wiley, 1998, chapter 2.

BIOGRAPHIES

Per-Erik Hagmark, PhD
Machine Design and Operation Laboratory
Tampere University of Technology
Korkeakoulunkatu 6
FI-33101 Tampere, Finland

e-mail: per-erik.hagmark@tut.fi

Per-Erik Hagmark serves on the Machine Design and Operation Laboratory at the Tampere University of Technology. Previously he has held research and teacher positions at Helsinki University of Technology, The University of Helsinki and Helsinki Polytechnic. He earned his PhD in Mathematics, Applied Mathematics and Theoretical Physics at Helsinki University of Technology in 1983 with a dissertation on generalizations of Walsh functions and fast algorithms. His recent research activities have been around statistics, simulation, reliability theory, and programming,

Seppo Virtanen, PhD
Machine Design and Operation Laboratory
Tampere University of Technology
Korkeakoulunkatu 6
FI-33101 Tampere, Finland

e-mail: seppo.virtanen@tut.fi

Seppo Virtanen received his B.Sc., M.Sc. and PhD. degrees from Helsinki University of Technology, Finland. He is currently a Professor in the Machine Design and Operation Laboratory at the Tampere University of Technology. His research and teaching interest includes reliability and maintainability engineering and risk management within a product and system design process. Professor Virtanen has over 15 year's industry experience in the field of reliability engineering and maintenance, which includes three years in energy, pulp and paper industry in USA and two years offshore industry in Norway.