

THE RIAC 217PLUS™ SOFTWARE FAILURE RATE MODEL

David Nicholls, RIAC (Quanterion Solutions Incorporated)

In this edition of the Journal, we present the 217Plus™ software model in its entirety. The model form is different than those for the hardware component models previously described [References 1 through 7]. Its development is discussed in more detail in Reference 8. A brief example will be provided at the end of the article.

Development of the 217Plus™ Software Failure Rate Model

Modern systems (and systems of systems) typically depend on significant amounts of software. Therefore, for a reliability assessment tool to be complete, it must include provisions for the estimation of software reliability. Many of the existing software reliability models are estimation models that require empirical test data. In many cases, data is simply not available at a point in time when a reliability estimate of a system is needed. Therefore, it was necessary to develop a predictive software reliability model that does not require empirical data. Like the 217Plus™ component models, the technique must be based on readily accessible data and information.

Like the hardware component models presented previously (and included in their entirety in Reference 9), the premise of the software model is that the inherent fault density of software can be estimated as a function of the development processes. However, in the software model, a separate process grading criteria is not included. Rather, due to its acceptance within the industry, the SEI (Software Engineering Institute) Capability Maturity Model (CMM) was used for this purpose. Once the inherent fault density is estimated as a function of the achieved CMM level, it is converted to a failure rate based on the defined operational profile of the software.

The 217Plus™ software reliability growth characteristics are modeled in a manner similar to that of hardware. For example, the potential for software reliability growth is assessed and the likely failure rate impact as a function of time is estimated. Both the growth rate and the stabilization time are estimated for this purpose. The default time for items to plateau and their residual fault content to stabilize is typically 48 months following its initial release. Subsequent item releases, such as a

new software version, typically take 24 months to stabilize. In the case of software, this reliability growth is a function of the organization that will perform the field maintenance, which may be different than the development organization.

The user must assess the SEI Capability Maturity Model Level of the process developing the code. This assessment should be based upon the actual SEI assessment, if that exists. Lacking a SEI assessment of the development facility, one can use the Safety Level of the Software Level that the software is being developed to meet, or the ISO 9000 facility rating. If none of the cited process ratings exist, then review the SEI CMM Level requirements to determine and apply the CMM Level that most reasonably fits this item. (It should be noted that the SEI has since updated their CMM, defining it as CMM-Integrated (or CMMI). The structured version of the CMMI retains the same five-level format as its predecessor, so the basic methodology for using the 217Plus™ software reliability prediction model remains valid.)

The “Defect Stabilization Level” values should only be used if the organization that is maintaining the software has processes in place that will improve the reliability of the delivered code after faults are identified. If such growth processes are not in place, then the stabilization level that should be used is 100%. Also, the CMM level used for determining the defect stabilization level should be that of the maintaining organization, which could be different from that of the development organization.

Determining the Reliability Growth Coefficient

The software reliability grows as the fault content decreases exponentially over time, which results from the number of faults experienced and removed in the code, and is proportional to the total number of faults in the code. The empirical evidence shows program faults are discovered and removed exponentially over time as follows:

$$F(t) = F_0 e^{-kt}$$

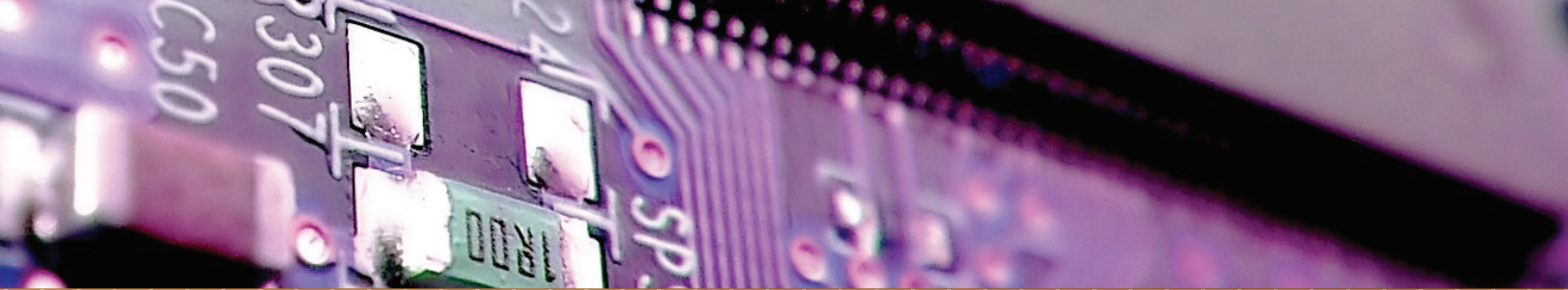
where,

$F(t)$ = Current number of faults remaining in the code after calendar time, t

F_0 = Initial number of faults in the code at delivery

k = Reliability growth constant

t = Calendar time in months



Taking the natural log of both sides of this equation yields:

$$k = \left(\frac{1}{t}\right) \ln\left(\frac{F_0}{F(t)}\right)$$

The reliability growth constant, k , is determined such that the latent fault content drops to its stabilization level or proportion of the initial faults as determined by the defect stabilization level, after the stabilization time. For example, for CMM Level 1 code, the constant “ k ” is determined by solving this equation for $F(t)$ dropping to 10% at $t = 48$ months. Then $(F_0/F(t)) = 10$, and $k = 0.048$ per month.

Converting Fault Density to An Operational Failure Rate

The fault removal curve equation represents reliability growth. For purposes of estimating the operational failure rate of the fielded code, one can apply a linear, piece-wise approximation to the exponential curve. This simplifies calculation of an average failure rate over the time interval. The number of faults found and removed is indicative of the failure rate of the software. That is, it usually takes a failure to uncover an underlying fault in the software. The more failures that occur, the more underlying faults there are revealed in the code. The software failure rate, λ , is related to the number of underlying faults as follows:

$$\lambda(t) = \frac{F(t_2) - F(t_1)}{t_2 - t_1}$$

Adding a constant of proportionality, FER,

$$\lambda(t) = FER \times \frac{F(t_2) - F(t_1)}{t_2 - t_1}$$

Table 1: Elements of the Fault Expansion Ratio (FER)

Parameter Symbol	Name	Description	Default
FL	Fault Latency	Average number of times a failure is expected to reoccur before its underlying fault is corrected	2.0 (dimensionless number)
FA	Fault Activation	Fraction (in decimal form) of population exhibiting fault activation	1.0 (100%)
AS	Average % Severity	Fraction (in decimal form) of faults that are disruptive, or critical, to the customer	0.5 (50%)
DC	Duty cycle	Fraction of calendar time the software is in operation (in decimal form)	Operational profile duty cycle

where “FER” is the Fault Expansion Ratio, a factor which accounts for the elements summarized in Table 1.

The faults contained in the FER version of the software failure rate equation are generally classified into four severity classes (1, 2, 3, 4). Severity 1 usually implies a failure that is disabling to the application. Severity 2 implies a serious, but not catastrophic, disruption of work. Severity 3 and 4 are much less severe failures that would usually be considered just annoyances. A value between 0.0 and 1.0 would represent the percentage of all faults that would be disruptive and considered serious by the customer.

217Plus™ Software Failure Rate Model

The basic form of the 217Plus™ Software Model [Reference 9] is:

$$\lambda_{SW} = \left(\frac{F_{t_{i-1}} - F_{t_i}}{730}\right) (DC \times FL \times FA \times AS) \times 10^6$$

where,

λ_{SW} = Predicted software failure rate at month t_i (in failures per million calendar hours)

F_{t_i} = Number of faults remaining at time t_i

$$F_{t_i} = F_0 e^{-kt_i}$$

F_0 is the initial defect density, and is calculated as:

$$F_0 = KSLOC \times FD$$

KSLOC = Lines of source code (in thousands)

FD = Fault Density (default from Table 3)

continued on next page >>>

k is the growth rate, and is:

$$k = \frac{\ln\left(\frac{1}{DSL}\right)}{t_s}$$

t_i = Time (in months) after deployment

t_s = Time (in months) to software stabilization
(default from Table 2)

DSL = Defect Stabilization Level (default from
Table 3)

F_{i-1} = Number of faults remaining at time, t_i-1

$$F_{i-1} = F_0 e^{-k(t_i-1)}$$

The model parameters, and their symbol, description, and default value, are summarized in Table 2. Table 3

Whenever actual company experience exists, it should be substituted for the heuristic 217Plus™ reliability profiles summarized in Table 4. Use your own recent item history if your company has determined the latent defect density of its systems or products.

Example Calculation

A new release of a software program for an entertainment system used on a commercial aircraft for international and trans-continental travel has been used in the field for thirty-six (36) months. The software program is implemented using 245,500

lines of source code (i.e., comment lines and other non-executable code are not counted). The organization that developed the software code, and the organization that supports/maintains it, have each been certified as being SSEI CMM Level 3 compliant. Through the collection and analysis of its own data, a number of measurements and metrics have been established. An analysis of the software operational profile has determined that the software program is exercised approximately 70% of the calendar time each year (the software is non-operational during taxi, takeoff and landing, and when the aircraft is out of service for maintenance or normal downtime). The company has also determined that, on average, a failure reoccurs 2.6 times before the underlying fault is correctly identified and corrected. To date, 5% of the fielded systems containing this new software have experienced faults resulting in intermittent system interruptions lasting between 5 and 30 seconds. The fraction of these faults that are considered annoying to the customer is around 10%, as they occur primarily during routine “canned” safety instructions. At the time the new software was “shipped”, the company measured the fault density as 2.1 remaining defects per thousand lines of source code. Finally, based on its SEI CMM Level 3 certification, the company accepts a default value of 0.05 as its Defect Stabilization Level.

The failure rate equation for software [Reference 9] is:

$$\lambda_{sw} = \left(\frac{F_{t_i-1} - F_{t_i}}{730} \right) (DC \times FL \times FA \times AS) \times 10^6$$

The initial defect density, F_0 , is calculated as $F_0 = 245.5 * 2.1 = 515.55$ defects based on 245,500 lines of source code and $FD = 2.1$ (given).

Table 2: Parameters Used in the 217Plus™ Software Model

Parameter Symbol	Name	Description	Default
KSLOC	Lines of Source Code (in thousands)	Lines of source code (in thousands), not including comments, i.e., size of the software	None
FD	Fault Density	Initial quality as measured by fault density at item shipment	Table 3
FL	Fault Latency	Average number of times a failure is expected to reoccur before its underlying fault is corrected	2.0 (dimensionless number)
FA	Fault Activation	Fraction (in decimal form) of population exhibiting fault activation	1.0 (100%)
AS	Average % Severity	Fraction (in decimal form) of faults that are disruptive, or critical, to the customer	0.5 (50%)
ts	Time to Stabilization		48 (months) for initial software release. However, this value should be changed to 24 (months) for subsequent software releases.
DSL	Defect Stabilization Level	The level at which the software failure rate stabilizes relative to F0	Table 3
DC	Duty Cycle	Fraction of calendar time the software is in operation (in decimal form)	Operational profile duty cycle

Table 3: Default Values of Defect Density (FD) and Defect Stabilization Level (DSL)

SEI'S CMM Level	Initial Design Defect Density (FD) (Defects per 1000 lines of source code, for all severities)	Defect Stabilization Level (DSL)
5	0.5	0.01
4	1.0	0.03
3	2.0	0.05
2	3.0	0.07
1	5.0	0.10
Unrated	6.0	Not Estimated

Table 4: Default Operating Profile Values

Equipment Type	Operating Profile	
	Duty Cycles	Cycling Rate (Cycles/yr)
Automotive	5	1000
Commercial Aircraft	25	2982
Computer	80	1491
Consumer	30	368
Emergency Power	10	50
Industrial	80	184
Military Aircraft	25	1008
Military Ground	45	263
Telecommunications	80	368

The growth rate factor, k , is calculated to be:

$$k = \frac{\ln\left(\frac{1}{0.05}\right)}{48} = 0.0624$$

where,

DSL = 0.05, based on SEI CMM Level 3 certification (from Table 3)

$t_s = 48$, based on an initial software release (from Table 2)

The number of faults remaining at 36 months, F_{36} is:

$$F_{36} = 515.55e^{-(0.0624)(36)} = 54.51$$

The number of faults remaining at 35 months, F_{35} is:

$$F_{35} = 515.55e^{-(0.0624)(35)} = 58.02$$

DC = 0.70 (given as 70%)

FL = 2.60 (given)

FA = 0.05 (given as 5%)

AS = 0.10 (given as 10%)

$$\lambda_{sw} = \left(\frac{58.02 - 54.51}{730}\right)(0.70)(2.60)(0.05)(0.10)(10^6) = 43.76$$

$\lambda_{sw} = 43.76 \text{ f}/10^6 \text{ calendar hours}$

References:

1. "An Introduction to the RIAC 217Plus™ Component Failure Rate Models", Journal of the Reliability Information Analysis Center, First Quarter 2007, available for PDF download from the RIAC at <http://theRIAC.org>
2. "The 217Plus™ Capacitor and Diode Failure Rate Models", Journal of the Reliability Information Analysis Center, Second Quarter 2007, available for PDF download from the RIAC at <http://theRIAC.org>
3. "The 217Plus™ Integrated Circuit and Inductor Failure Rate Models", Journal of the Reliability Information Analysis Center, Third Quarter 2007, available for PDF download from the RIAC at <http://theRIAC.org>
4. "The 217Plus™ Transformer and Optoelectronic Device Failure Rate Models", Journal of the Reliability Information Analysis Center, Fourth Quarter 2007, available for PDF download from the RIAC at <http://theRIAC.org>
5. "The 217Plus™ Switch and Relay Failure Rate Models", Journal of the Reliability Information Analysis Center, First Quarter 2008, available for PDF download from the RIAC at <http://theRIAC.org>
6. "The 217Plus™ Connector and Resistor Failure Rate Models", Journal of the Reliability Information Analysis Center, Second Quarter 2008, available for PDF download from the RIAC at <http://theRIAC.org>
7. "The 217Plus™ Transistor and Thyristor Failure Rate Models", Journal of the Reliability Information Analysis Center, Third Quarter 2008, available for PDF download from the RIAC at <http://theRIAC.org>
8. Denson, W.K. and Keene, S., "New System Reliability Assessment Methodology", Reliability Analysis Center, RAC Project #A06839, March 1997
9. Denson, W.K., "Handbook of 217Plus™ Reliability Prediction Models", Reliability Information Analysis Center (RIAC), 26 May 2006, ISBN 1-933904-02-X
10. "An Overview of the 217Plus™ System Reliability Assessment Methodology", Journal of the Reliability Information Analysis Center, Fourth Quarter 2006, available for