

**Instructions:**

**Search for < and >. These contain tailoring instructions which must be replaced as instructed**

**Keep all references for sections of technical material**

**<PROJECT NAME>**

**SOFTWARE RELIABILITY PROGRAM PLAN**

**<DATE>**  
**<AUTHORS>**

## Revision History

<b>Revision History</b> .....	<b>1</b>
<b>1.0 Definitions</b> .....	<b>6</b>
<b>2.0 The software reliability process</b> .....	<b>8</b>
Table 2-1 Software Reliability Process .....	8
Table 2-1 Software Reliability Process - Continued .....	9
<b>3.0 Prediction during Requirements, Design, Code, Unit Test and Integration testing</b> .....	<b>10</b>
Table 3-1 Summary of prediction models .....	10
<b>3.1 SEI CMMi and Industry lookup tables</b> .....	<b>11</b>
3.1.1 Collect Data.....	11
Table 3-2 Summary of simple prediction models.....	11
3.1.2 Predict.....	11
Table 3-3 SEI CMMi defect density lookup table .....	11
Table 3-4 Industry type defect density lookup table .....	11
Table 3-5 Application type defect density lookup table.....	12
3.1.3 Manage/Improve .....	12
3.1.4 Execute .....	12
3.1.5 Feedback.....	12
<b>3.2 SoftRel Shortcut model</b> .....	<b>13</b>
Figure 3-1 Qualitative differences between SoftRel model percentile groups .....	13
3.2.1 Collect Data.....	13
Table 3-6 – The Shortcut Survey[2] .....	14
3.2.2 Predict.....	14
Table 3-7 – The Shortcut Survey Algorithm.....	14
Table 3-8 – Predict Defect Density from Predicted Percentile Group ...	15
3.2.3 Manage/Improve .....	15
3.2.4 Execute .....	15
3.2.5 Feedback.....	15
<b>3.3 SoftRel Full-scale model</b> .....	<b>16</b>
3.3.1 Collect Data.....	16
Table 3-9 Determine which Full-scale surveys to complete.....	16
Table 3-10 Filter for Survey A .....	17
Table 3-11 – Full-scale Survey A .....	18
Table 3-11 – Full-scale Survey A - continued.....	19
Table 3-12 Filter for Survey B .....	20
Table 3-13 – Full-scale Survey B .....	20
3.3.2 Predict.....	21
3.3.3 Manage/Improve .....	21

Table 3-14 - How the percentile groups compare with respect to defects and on time delivery.....	21
Table 3-15 Definition of time, cost and index .....	23
Table 3-16 Cost/Time effectiveness suggestions for transitioning to the next percentile .....	23
3.3.4 Execute .....	23
3.3.5 Feedback.....	23
<b>3.4 Rome Laboratory Prediction Model .....</b>	<b>24</b>
Table 3-17 Overview of Rome Laboratory Model Factors.....	24
3.4.1 Collect Data.....	25
Table 3-18 Development (D) Factor Survey.....	25
3.4.2 Predict.....	28
3.4.2.1 Application Factor .....	28
Table 3-19 Application Factor .....	28
3.4.2.2 Development Factor.....	29
3.4.3 Manage/Improve .....	30
3.4.4 Execute .....	30
3.4.5 Feedback.....	30
<b>3.5 Our own historical data .....</b>	<b>31</b>
3.5.1 Collect Data.....	31
3.5.2 Predict/Estimate .....	31
3.5.3 Manage/Improve .....	32
3.5.4 Execute .....	32
3.5.5 Feedback.....	32
<b>3.6 Predictions on multiple components and contractors .....</b>	<b>32</b>
<b>4.0 Prediction during systems level testing.....</b>	<b>33</b>
Table 4-1 Summary of Reliability Growth Models.....	33
<b>4.1 Collect data .....</b>	<b>33</b>
<b>4.2 Estimate .....</b>	<b>34</b>
Table 4-2 Estimating Parameters for Reliability Growth Models.....	34
Table 4-3 Failure Rate/MTTF Formulas for Reliability Growth Models .	34
<b>4.3 Manage/Improve .....</b>	<b>35</b>
4.3.1 Determine which reliability growth model is currently estimating the best.....	35
4.3.2 How to determine how much more testing time is needed to meet an objective .....	35
Table 4-4 Estimating remaining testing time.....	36
4.3.3. Determine how many more defects must be discovered to meet an objective .....	36
Table 4-5 Estimating Remaining Defects.....	36
4.3.4 Make improvements to reach the reliability objective.....	36

Table 4-6 Improvement Options during Testing..... 37

**4.4 Execute ..... 37**

**4.5 Feedback ..... 37**

**5.0 Prediction during operation..... 38**

**5.1 Collect the following data from actual operation of the software/system..... 38**

**5.2 Compute ..... 38**

**5.3 Manage ..... 39**

Figure 5-1 Estimating inherent defects and growth rate..... 40

**5.4 Execute ..... 40**

**5.5 Feedback/Archive..... 40**

Table 5-1 Archiving Operational Data..... 40

**6.0 Predicting MTTF, MTTCF, MTBI, interruption rate and failure rate from predicted defects..... 41**

Table 6-1 Predicting Growth Rate ..... 41

**7.0 Predicting Reliability and Availability from Predicted MTTF and failure rate ..... 43**

**7.1 Reliability ..... 43**

**7.2 Availability..... 43**

**7.2.1 MTSWR ..... 43**

**8.0 Predicting Effective Normalized Size..... 44**

Table 8-1 Converting to normalized KSLOC [6]..... 44

**9.0 Predicting Maintenance staffing required..... 45**

**10.0 Predict testing resources ..... 46**

Table 10-1 Testing Growth Rates..... 46

**11.0 Allocate reliability objectives..... 47**

**11.1 Allocate between hardware and software ..... 47**

11.1.1 Using an availability objective ..... 47

11.1.2 Using a reliability objective ..... 47

**11.2 Models to combine hardware and software reliability..... 48**

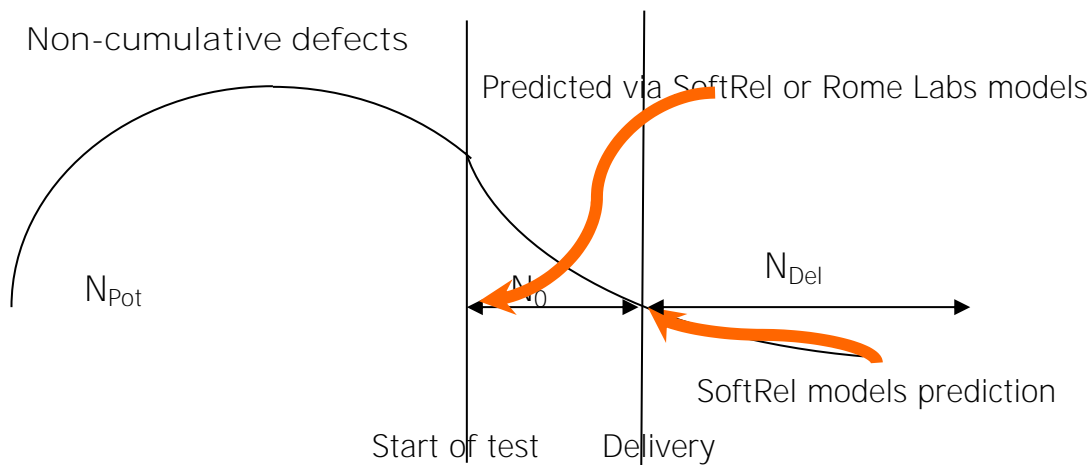
Table 11-1 Summary of Models used to Combine Hardware and Software Reliability ..... 48

11.2.1 Split the failure rate evenly .....	48
11.2.2 By number of CSCIs and HWCIs .....	49
11.2.3 By Relative Complexity .....	49
11.2.4 By Past History .....	49
11.2.5 By Achievable Failure Rates .....	49
11.2.6 Mission Model.....	49
11.2.7 Markov Model .....	50
11.2.8 Operational Profile Model .....	50
Figure 11-1 Operational Profile Model.....	51
<b>11.3 Reliability block diagrams .....</b>	<b>52</b>
Figure 11-2 Reliability Block Diagrams when software and hardware are not redundant.....	52
Figure 11-3 Reliability Block Diagrams when software is not redundant but hardware is redundant.....	52
Figure 11-4 Reliability Block Diagrams when hardware is redundant and software is not redundant but is not expected to have common cause or repeatable failures.....	53
<b>12.0 References.....</b>	<b>54</b>

## 1.0 Definitions

Software reliability is the probability (a number between 0 and 100%) of success of the software over some specified mission time. It is a function of inherent defects introduced during requirements translation, design, code, and corrective action as well as the operational profile[1].

Generally, the defects found of the life of the software project will exhibit a Rayleigh curve as shown below. Note that for incremental or large projects there may be multiple peaks. Our goal is to predict the defects under this curve at milestones of interest. The two key milestones shown below are during testing and after testing is completed (delivery).



### Defects, faults, failures and errors

Errors - human mistakes

Defects and faults - manifestation of a human mistake in the product which includes requirements, design and code

Failures - these are always events - what happens when the operational profile is such that one of these defects causes an event

### Escapes

These are defects that are not found during systems testing and are observed in the field and result in a corrective action.

Escapes do not include:

- defects found after delivery that are intentionally not fixed due to being negligible
- defects found after delivery that are intentionally not fixed for any other reason

Defect density - Defects per size. This measure is useful because it allows the defects from projects of all sizes to be compared against each other.

KSLOC - 1000 Executable non commented, non blank lines of code

EKSLOC - Effective KSLOC - new or modified code

### **Categories of defects -**

Critical - Those that impact availability **and** have no workaround. These may include level 1 or level 2 defects.

Interruptions - Those defects that impact available and have a viable known workaround.

All others - These include defects that are serious enough to require a corrective action at some future point in time. These do NOT include defects which are negligible (don't require a corrective action) or new feature requests.

### **Reliability terms**

Failure rate - Failures per unit of time

MTTF - Mean Time To the next Failure

MTTCF - Mean Time to the next Critical Failure

Critical failure rate - The inverse of the MTTCF (assumes an exponential distribution)

MTBI - Mean Time Between Interruptions

Interruption rate - the inverse of the MTBI (assumes an exponential distribution)