Appendix

A

Software Reliability Tools

George Stark
Mitre Corporation

A.1 Introduction

In studying the SRE estimation examples in Parts 1 and 2 of this book, you will notice several features needed to execute most applications of SRE. They include the following:

1. Collecting failure and test time information
2. Calculating estimates of model parameters using the information
3. Testing the fit of a model against the collected information
4. Selecting a model to make predictions of remaining faults, time to test, or other items of interest
5. Applying the model

It is the commonality of these and other features that leads to the development of special-purpose SRE measurement tools. This appendix provides a summary description of several of these tools. It does not consider early prediction tools, as they are generally less mature and not as readily available to the software engineering community.

Section A.2 discusses the relative merits of using an SRE tool rather than a general-purpose application like a spreadsheet or statistical package for conducting an SRE analysis. Section A.3 lists criteria for consideration when deciding which tool to purchase or use for a particular SRE project. These criteria should be kept in mind while reading Secs. A.4 through A.9, which summarize some widely used tools with an example execution of each tool. Finally, Sec. A.10 provides additional details on the features of the highlighted tools as well as some additional tools in tabular format.

A.2 Comparison of Commercially Available Tools with General-Purpose Languages and Applications

One of the many important decisions a reliability modeler or analyst must make in performing an SRE study is the choice of a tool. An inappropriate choice may in itself kill an SRE project if it cannot be completed in time, does not handle the type of data collected for the project, or does not have a robust set of models that may fit the project to make accurate predictions of important information. Engineers may choose to implement one of the myriad of software reliability models using a general-purpose application program such as a spreadsheet or a statistical package such as SAS or SPSS. They may also choose to develop their own models using a general-purpose programming language such FORTRAN or C. The following are some advantages of using a commercially available tool rather than a general-purpose application or developing your own program.

1. Commercially available tools provide most (if not all) of the features needed in executing a software reliability analysis, resulting in a decrease of programming time that can often be significant.
2. Comparing multiple models on the same failure data and changing the analysis to use a different model's predictions is generally easier to accomplish using a commercially available tool.
3. The tools provide better error detection because many potential types of errors have been identified and are checked for automatically. Since no code must be written, the chance of making a mistake in a calculation or formula is very small.
4. The tools provide a general framework for reliability estimation and prediction. Their basic structure is from the theories developed by researchers and uses the terminology of those models.

On the other hand, programming your own tool or using a general-purpose application such as a spreadsheet or statistical package to analyze the failure data offers several advantages. For example:

1. Most modelers already know a general-purpose application or language and have access to these items on a computer that the analyst wants to use. This is often not the case with a commercially available tool.
2. An efficiently written program in a general-purpose language may take less time than using a commercially available tool. This is

because commercial tools are designed to handle a wide variety of situations within their expected input types, whereas a FORTRAN program can be tailored to a particular environment.

3. General-purpose tools and languages allow greater flexibility than other tools. For example, using a general-purpose package the analyst is able to use models that are not part of a commercial tool. Also, tailoring a program to a particular project may allow the analyst to accommodate factors (e.g., workload, complexity, fault tolerance) that are not relevant to some models implemented in a tool. Using a spreadsheet or statistical package allows the analyst to look at the reliability data in a number of ways that may not be available in a commercial tool.

Although there are clearly advantages to using either approach, in general an analyst should give serious consideration to the use of a commercially available tool. If such a decision is made, the criteria discussed in Sec. A.3 may be helpful in deciding which particular tool to use.

A.3 Criteria for Selecting a Software Reliability Estimation Tool

Almost all software reliability tools use one of two basic types of input data: time-domain data (i.e., time-between-failures data) or interval-domain data (i.e., failure-count data). These types of data, defined in Chap. 1, are also used by modelers using general-purpose applications or languages. We assume in this section that a decision has been made to use a software reliability tool rather than a general-purpose program or programming language, and we present criteria which may be useful in selecting a tool. There are two levels at which a decision with regard to reliability tools must be made. At the highest level, an organization must decide which tools to purchase or lease for its general use. You should not necessarily feel that a single tool must be chosen, but it does help management if results are displayed consistently across projects for comparisons of processes and software engineering technologies. At the lower level, an analyst must decide which tool to use for a particular project or study.

The following are some of the criteria that should be considered in selecting a tool for an organization [Sarg79]:

1. Availability of the tool for the company's computer system(s)
2. Cost of installing and maintaining the program
3. Number of studies likely to be done

4. Types of systems to be studied
5. Quality of the tool documentation and support
6. Ease of learning the tool
7. Flexibility and power of the tool

When an analyst must pick a tool for a particular application, many of the above criteria are still relevant. In addition, however, the following factors are important:

1. Availability of tools, either in-house or on a network
2. Goals and questions to be answered by the study
3. Models and statistical techniques understood by the analyst
4. Schedule for the project and type of data collected
5. Tool's ability to communicate the nature of the model and the results to a person other than the analyst (e.g., the end user or a manager).

The following sections overview several widely used tools and present a sample execution of each tool using failure data from an operational system. Section A.10 provides additional details on the tools in tabular format for easy comparison.

A.4 AT&T Software Reliability Engineering Toolkit

The AT&T Software Reliability Engineering Toolkit is a command-line-driven system that executes the Musa basic and Musa/Okumoto logarithmic Poisson execution time software reliability models. These models are described in Chap. 3. After having been developed by AT&T in 1977, the toolkit evolved through a couple of versions; the latest and most comprehensive version is available through a three-day AT&T software reliability training course offered throughout the year, and its MS/DOS version is included in the Data and Tool Disk. The toolkit is available for any platform running MS/DOS or UNIX® operating systems that support the C language. References for the AT&T Toolkit include the Program Reference Guide [ATT90, Musa77a, Musa77b, Star91].

The simplicity and intuitive nature of the Musa model parameters coupled with the toolkit support for both time domain and interval domain failure data as input has wide appeal. The outputs from this

SOFTWARE RELIABILITY ESTIMATION
EXPONENTIAL (BASIC) MODEL
HANDBOOK DATA SET

BASED ON A SAMPLE OF 129 TEST FAILURES
TEST EXECUTION TIME IS 20 CPU-HR
FAILURE INTENSITY OBJECTIVE IS 2.4 FAILURES/1000-CPU-HR
CURRENT DATE IN TEST 900419
TIME FROM START OF TEST IS 37 DAYS

	CONF. LIMITS				MOST		CONF. LIMITS		
	95%	90%	75%	50%	LIKELY	50%	75%	90%	95%
TOTAL FAILURES	134	134	135	137	139	142	144	148	151
********** FAILURE INTENSITIES (FAILURES/1000-CPU-HR) **********									
INITIAL1463	1521	1614	1705	1837	1972	2070	2172	2238	
PRESENT78.69	85.49	97.25	109.9	130.3	154.0	172.8	194.5	209.4	
*** ADDITIONAL REQUIREMENTS TO MEET FAILURE INTENSITY OBJECTIVE ***									
FAILURES	5	5	6	7	10	13	15	19	21
TEST EXEC. TIME	20.85	22.09	24.21	26.49	30.19	34.61	38.29	42.73	45.98
WORK DAYS	6.52	7.05	8.02	9.13	11.08	13.60	15.85	18.73	20.91
COMPLETION DATE	900430	900501	900502	900503	900507	900509	900511	900516	900518

Figure A.1 Tabular output from AT&T Toolkit.

tool are easy to understand. Summary estimates for such reliability measures as total failures (for the basic model), the initial failure rate (the toolkit and Musa use the term *failure intensity* instead of *failure rate*), and the present failure rate of the program under test are shown in both tabular and graphical fashion. Figure A.1 shows a sample output from the AT&T Toolkit. Because the toolkit is estimating the parameters of the model from collected data, the tool also reports confidence intervals around the estimated (or most likely) value. Thus, in our example session, although total failures are reported as 139 (the "most likely" value), we can say with 90 percent confidence that total failures are between 134 and 148.

Other plots from the toolkit help you to see how well the model fits the actual data, the trend in the initial and current failure rate for the program, and predict the completion date of the project. In addition to the estimation model (est and plot), the AT&T SRE Toolkit also includes (1) a prediction model that calculates model-based predictions of cost and schedule for a range of failure-rate objectives and (2) a reliability demonstration model that helps an engineer develop a reliability demonstration chart to demonstrate the reliability of acquired software for a given operational profile.

A.5 Statistical Modeling and Estimation of Reliability Functions for Software (SMERFS)

SMERFS is a menu-driven tool that is suited particularly well for a newcomer to software reliability modeling. Originally developed in 1983 by the U.S. Naval Surface Warfare Center in Dahlgren, Virginia,

this public domain tool has evolved through several versions, the latest being SMERFS V. Historically, SMERFS has been the most popular tool because it offers flexibility in data collection and provides multiple time-domain (i.e., Littlewood and Verrall, Musa basic execution time, Musa/Okumoto logarithmic Poisson, Jelinski-Moranda, geometric, and nonhomogeneous Poisson) and interval-domain (i.e., Schneidewind, Brooks and Motley, nonhomogeneous Poisson, Yamada S-shaped, and generalized Poisson) models. This makes it useful for modelers who are not ready to settle on a single model for their software debugging process.

The program is written in a subset of ANSI FORTRAN '77, making it portable to a wide range of hardware platforms. SMERFS is designed as a driver program and a series of model libraries. Because the source code and software design documentation are provided with SMERFS, users can add new SRE models or delete models that are no longer in favor as they wish. The design also allows users to construct their own driver and develop a tailored user interface. A more detailed description of SMERFS, with examples, can be found in the SMERFS User's Guide [Farr93a] and in [Farr88].

SMERFS V both implements a database concept to data entry and allows ASCII file input of the data. In the database input, the user enters the failure data using the SMERFS data input module. Thus, the user does not need to know or understand the format of the input file, although it can be time-consuming. The ASCII file format is similar to that of the AT&T Toolkit. SMERFS allows the analyst to choose the model parameter estimation method: either maximum likelihood or least squares. SMERFS also provides basic statistics calculated from the data and the ability to analyze the residuals from the model analysis.

SMERFS allows the user to enter data, edit and/or transform the data if necessary, plot the data, select an appropriate model to fit the data, determine the fit of the model using both statistical and graphical techniques, make various reliability predictions based upon the fitted model, and try different models if the initial model proves inadequate. For model selection SMERFS provides the prequential likelihood statistic (see Chap. 4) as well as u-plots, y-plots, and measure of noise. Among the various reliability indicators provided by the different models are the following: expected time-to-next-failure occurrence, an estimate of the reliability for a specified operational time, the number of faults remaining in the software, an estimate of the time it will take to trigger the remaining failures, and the expected number of failures in the next session of a given duration. Figure A.2 shows SMERFS menus and sample model executions.

```
 THE AVAILABLE MAIN MODULE OPTIONS ARE:
     1 DATA INPUT              6 PLOT(S) OF THE RAW DATA
     2 DATA EDIT               7 MODEL APPLICABILITY ANALYSES
     3 UNIT CONVERSIONS        8 EXECUTIONS OF THE MODELS
     4 DATA TRANSFORMATIONS    9 STOP EXECUTION OF SMERFS
     5 DATA STATISTICS
    ENTER MAIN MODULE OPTION.
            5
                          INTERVAL DATA WITH EQUAL LENGTHS
                           WITH FAULT COUNTS TOTALING TO          129
                          ******************************************
    MEDIAN OF THE DATA    *                .50000000E+01              *
    LOWER & UPPER HINGES  *      .20000000E+01      .85000000E+01      *
    MINIMUM AND MAXIMUM   *      .00000000E+00      .27000000E+02      *
    NUMBER OF ENTRIES     *                    20                     *
    AVERAGE OF THE DATA   *                .64500000E+01              *
    STD. DEV. & VARIANCE  *      .63451183E+01      .40260526E+02      *
    SKEWNESS & KURTOSIS   *      .18568731E+01      .35976645E+01      *
                          ******************************************

    ENTER MAIN MODULE OPTION, OR ZERO FOR A LIST.
            0
    THE AVAILABLE MAIN MODULE OPTIONS ARE:
     1 DATA INPUT              6 PLOT(S) OF THE RAW DATA
     2 DATA EDIT               7 MODEL APPLICABILITY ANALYSES
     3 UNIT CONVERSIONS        8 EXECUTIONS OF THE MODELS
     4 DATA TRANSFORMATIONS    9 STOP EXECUTION OF SMERFS
     5 DATA STATISTICS
    ENTER MAIN MODULE OPTION.
8
ENTER COUNT MODEL OPTION, OR ZERO FOR A LIST.
            0
    THE AVAILABLE FAULT COUNT MODELS ARE:
     1 THE BROOKS AND MOTLEY MODEL
     2 THE GENERALIZED POISSON MODEL
     3 THE NON-HOMOGENEOUS POISSON MODEL
     4 THE SCHNEIDEWIND MODEL
     5 THE S-SHAPED RELIABILITY GROWTH MODEL
     6 RETURN TO THE MAIN PROGRAM
    ENTER MODEL OPTION.
            3
    ENTER ONE FOR NHPP MODEL DESCRIPTION; ELSE ZERO.
            0
    ENTER ONE FOR  MAXIMUM LIKELIHOOD METHOD,  TWO FOR LEAST SQUARES
    METHOD, OR THREE TO TERMINATE MODEL EXECUTION.
            1
    ML MODEL ESTIMATES ARE:
    (THE APPROXIMATE 95% CONFIDENCE INTERVALS APPEAR IN PARENTHESES)
     PROPORTIONALITY CONSTANT    .13230E+00 (  .9738E-01,  .1672E+00)
     TOTAL NUMBER OF FAULTS      .13885E+03 (  .1290E+03,  .1638E+03)
     # OF FAULTS REMAINING       .98495E+01 (  .0000E+00,  .3481E+02)

    THE AVAILABLE FUTURE PREDICTIONS ARE:
     1) THE NUMBER OF FAULTS EXPECTED IN THE NEXT TESTING PERIOD
     2) THE NUMBER OF PERIODS NEEDED TO DISCOVER THE NEXT M FAULTS
    ENTER PREDICTION OPTION, OR ZERO TO END PREDICTIONS.
            1
     ENTER PROJECTED LENGTH OF THE PERIOD, OR ZERO TO END.
         1.000000000000000
     # OF FAULTS EXPECTED        .12206E+01
```

Figure A.2 SMERFS main menu and sample model executions.

A.6 Statistical Modeling and Reliability Program (SRMP)

The Statistical Modeling and Reliability Program (SRMP) was developed by the Reliability and Statistical Consultants, Limited, of the United Kingdom in 1988. SRMP is a command-line-oriented tool developed for an IBM PC/AT with 500K of memory, and requires a math coprocessor to be installed. It also runs on UNIX®-based workstations. SRMP contains nine models: Musa/Okumoto, Duane, Jelinski-Moranda, Goel/Okumoto, Bayesian Jelinski-Moranda, Littlewood/Verrall, Littlewood, Keiller/Littlewood, and Littlewood nonhomogeneous Poisson. SRMP uses the maximum likelihood estimation technique to compute the model parameters, and provides the following reliability indicators: reliability function, failure rate, mean time to failure, median time to failure (as well as the 25 and 75 percent values), and the model parameters for each model. It expects time domain input data and does not execute on interval domain data. This tool was the first to provide a user with the ability to analyze model goodness-of-fit using the analytical techniques of prequential likelihood, u-plots, y-plots, and several measures of noise. These techniques have become a standard part of the analyst's toolkit and are considered necessary for most tools today.

SRMP requires an ASCII data file as input. The file contains the name (or other identification) of the project for which reliability calculations are being performed, the number of failures involved in the reliability analysis, and the interfailure times of all the failures. The input file also specifies the initial sample size (the initial portion of the total number of failures) chosen by the analyst and used by SRMP for the initial fitting of each reliability model to the data. The remaining failures are used by SRMP to calculate the prequential likelihood and other measures for assessing a reliability model's prediction accuracy. Furthermore, the input file contains certain mathematical parameters, chosen by the analyst, which are needed to initiate and control the SRMP algorithm's search for a convergent solution. Obviously, analysts must be knowledgeable in setting up the data file, as many parameters are at their discretion and the user manual does not provide an example file that can be copied. The input file allows flexibility in the analysis, such as partitioning the data set or scaling the data, and gives the analyst a choice of executing one model at a time or all models on a particular data set. To format the results from an SRMP run for review, the analyst must execute a second program, called OUTPUT. If the results are sent to the screen, the analyst must control the scrolling from the keyboard. The results may be printed to a file for later viewing or hard-copy printing.

More detail on SRMP is available in the User's Guide [SRMP88]. A sample SRMP output file is shown in Fig. A.3.

*** JELINSKI & MORANDA MODEL ***

PARAMETER ESTIMATION

I	T(I+1)	NHAT	FI	LOGF	FAIL	NUM
35	65.	47	0.251152D-03	-.208315D+03	0	11
36	176.	51	0.222643D-03	-.214248D+03	0	10
37	58.	54	0.204387D-03	-.220518D+03	0	9

133	1160.	140	0.361335D-04	-.946668D+03	0	9
134	1864.	141	0.355285D-04	-.955210D+03	0	8
135	4116.	142	0.348906D-04	-.963945d+03	0	8

RELIABILITY PREDICTIONS

I	T(I+1)	MEAN	LOWQ	MEDIAN	UPQ	ROCOF
35	65.	0.33180D+03	0.95454D+02	0.22999D+03	0.45998D+03	0.30138D-02
36	176.	0.29943D+03	0.86142D+02	0.20755D+03	0.41510D+03	0.33396D-02
37	58.	0.28780D+03	0.82796D+02	0.19949D+03	0.39898D+03	0.34746D-02

133	1160.	0.39536D+04	0.11374D+04	0.27404D+04	0.54808D+04	0.25293D-03
134	1864.	0.40209D+04	0.11567D+04	0.27871D+04	0.55742D+04	0.24870D-03
1354116.		0.40944D+04	0.11779D+04	0.28380D+04	0.56761D+04	0.24423D-03

ANALYSIS OF PREDICTIVE QUALITY

I	T(I+1)	U(I+1)	P(I+1)	SLOGP
35	65.	0.177904D+00	0.247765D-02	-.600045D+01
36	176.	0.444439D+00	0.185537D-02	-.122901D+02
37	58.	0.182517D+00	0.284041D-02	-.181539D+02

133	1160.	0.254279D+00	0.188618D-03	-.752167D+03
134	1864.	0.370970D+00	0.156439D-03	-.760930D+03
135	4116.	0.634054D+00	0.893767D-04	-.770253D+03

SUM OF DEVIANCE OF

	ROCOF	LOWQ	MEDIAN	UPQ
ABSOLUTE :	0.118622D-01	0.337083D+04	0.812175D+04	0.162435D+05
NORMALISED :	0.823370D+01	0.940515D+01	0.940515D+01	0.940515D+01

Figure A.3 SRMP tabular output.

A.7 Software Reliability Program (SoRel)

SoRel is a Macintosh-based software reliability measurement tool that was developed by LAAS, a lab of the National Center for Scientific Research in Toulouse, France, in 1991. It runs on a Macintosh II or later computer with a math coprocessor. The program was written in Pascal and requires about 200K of memory. SoRel is composed of two parts. The first part allows several reliability trend tests: the arithmetical test, the Laplace test, Kendall test, and Spearman test. These tests allow an analyst to identify whether the reliability function is increasing or decreasing so that an appropriate model can be

applied (see Chap. 10). The second part allows reliability growth model application and contains four models. The Goel-Okumoto model and the Yamada S-shaped model is available for interval domain data, the Littlewood/Verrall model for time domain, whereas the hyperexponential model operates on both time and interval domain data. The chosen model can then be validated using SoRel's three criteria: Kolmogorov-Smirnov distance, prequential likelihood, and *residue.* Residue is the residual value (observed – expected) from the model fit.

SoRel uses ASCII input files that are created using a spreadsheet. Numerical results are displayed on the screen during execution; the user can also request plots of the data. SoRel uses the maximum likelihood parameter estimation technique and provides the following reliability indicators: mean time to failure, cumulative number of failures, failure intensity, as well as the model parameters to evaluate other reliability functions. While only one model is executed at a time, the results are automatically saved to ASCII files which can be imported into spreadsheets or other applications for model comparisons. SoRel also allows the analyst to partition the data set for anal-

RESULTS DISPLAY

```
ESTIMATION PARAMETERS :

   NO =    1    MO   =  20  MT =  20

VALIDATION CRITERIA :

   Binf =   1   Bsup =  20

THE MODEL HAS BEEN APPLIED IN A RETRODICTIVE WAY


        No window has been used to calibrate the parameters of the model

        The parameters of the model have been re-calibrated each Step =   1

OPTIMISATION PROCEDURE  :     NEWTON-RAPHSON METHOD

         INITIAL VALUE OF (b)..................... =     1.0e-1
         CONVERGENCE SPECIFIED TOLERANCE ......... =     1.0e-4
         MAXIMUM NUMBER OF ITERATIONS ............ =     10

      Estimated number of latent faults in the program ( a ) =  130.668686
      Fault activation rate ( b ) =  0.317808

Time unit(i)    N(i)          H(i)          n(i)          h(i)       Residue


       1      2.700e+1      5.354e+0      2.700e+1      9.605e+0    -2.165e+1
       2      4.300e+1      1.748e+1      1.600e+1      1.398e+1    -2.552e+1
       3      5.400e+1      3.229e+1      1.100e+1      1.526e+1    -2.171e+1
       4      6.400e+1      4.743e+1      1.000e+1      1.481e+1    -1.657e+1
       5      7.500e+1      6.161e+1      1.100e+1      1.347e+1    -1.339e+1
       6      8.200e+1      7.425e+1      7.000e+0      1.176e+1    -7.755e+0
       7      8.400e+1      8.512e+1      2.000e+0      9.987e+0     1.118e+0
       8      8.900e+1      9.425e+1      5.000e+0      8.306e+0     5.252e+0
       9      9.200e+1      1.018e+2      3.000e+0      6.801e+0     9.789e+0
      10      9.300e+1      1.079e+2      1.000e+0      5.499e+0     1.492e+1
      11      9.700e+1      1.129e+2      4.000e+0      4.402e+0     1.586e+1
      12      1.040e+2      1.168e+2      7.000e+0      3.495e+0     1.279e+1
      13      1.060e+2      1.199e+2      2.000e+0      2.755e+0     1.390e+1
      14      1.110e+2      1.223e+2      5.000e+0      2.159e+0     1.135e+1
      15      1.160e+2      1.243e+2      5.000e+0      1.684e+0     8.260e+0
      16      1.220e+2      1.257e+2      6.000e+0      1.307e+0     3.748e+0
      17      1.220e+2      1.269e+2      0.000e+0      1.011e+0     4.900e+0
      18      1.270e+2      1.278e+2      5.000e+0      7.787e-1     7.901e-1
      19      1.280e+2      1.285e+2      1.000e+0      5.982e-1     4.748e-1
      20      1.290e+2      1.290e+2      1.000e+0      4.582e-1     0.000e+0


Sum of the residues  : R  =   -3.446
Sum in absolute values of the residues :  Ra = 209.738
Mean residue : Rm = 10.487
```

Figure A.4 Sorel output: S-shaped model application result.

ysis and establish intervals for validation criteria evaluation. More detail on SoRel can be found in [Kano92]. An example SoRel session is shown in Fig. A.4.

A.8 Computer-Aided Software Reliability Estimation (CASRE) Tool

CASRE is a PC-based tool that was developed in 1993 by the Jet Propulsion Laboratories for the Air Force to address the ease-of-use issues of other tools. CASRE requires the WINDOWS™ operating environment. It has a pull-down, menu-driven user interface and uses the same model library as the SMERFS tool (see Sec. A.5) with the additional feature of allowing linear combinations of models to create new ones at the user's discretion. Four combined models are permanently available in CASRE. These are constructed using the nonhomogeneous Poisson process, the Musa/Okumoto, and the Littlewood/Verrall models for time domain data. A more detailed description of CASRE, with examples, can be found in the CASRE User's Guide [Niko93] and in [Lyu92d, Lyu93b].

CASRE allows an analyst to invoke a preferred text editor, word processor, or other application from within CASRE to create the ASCII input data set. The input data set contains fields for the test interval number, the number of failures observed in the interval, the length of the test interval, the fraction of the program tested, and the severity of the failure. Once the data is entered, CASRE automatically provides the analyst with a raw data plot.

CASRE provides the analyst with the ability to convert from time-domain data to interval-domain data and vice versa. Model parameters can be estimated using either maximum likelihood or least squares as determined by the analyst. After the application of several models to a data set, multiple model results can be displayed in the graphical display window for analysis. CASRE uses the SMERFS library routines to calculate the prequential likelihood statistic for interval-domain models as well as u-plots, y-plots, and measures of noise for time-domain models. Figure A.5 shows sample CASRE models' execution results.

CASRE also provides operations to transform or smooth the failure data to remove noise from the data or change the shape and position of the data; the user can select and/or define models for application to the data and make various reliability predictions based upon the best model. Because multiple models may be executed, compared, and combined at the same time, the CASRE tool offers modeling flexibility not offered in the other tools.

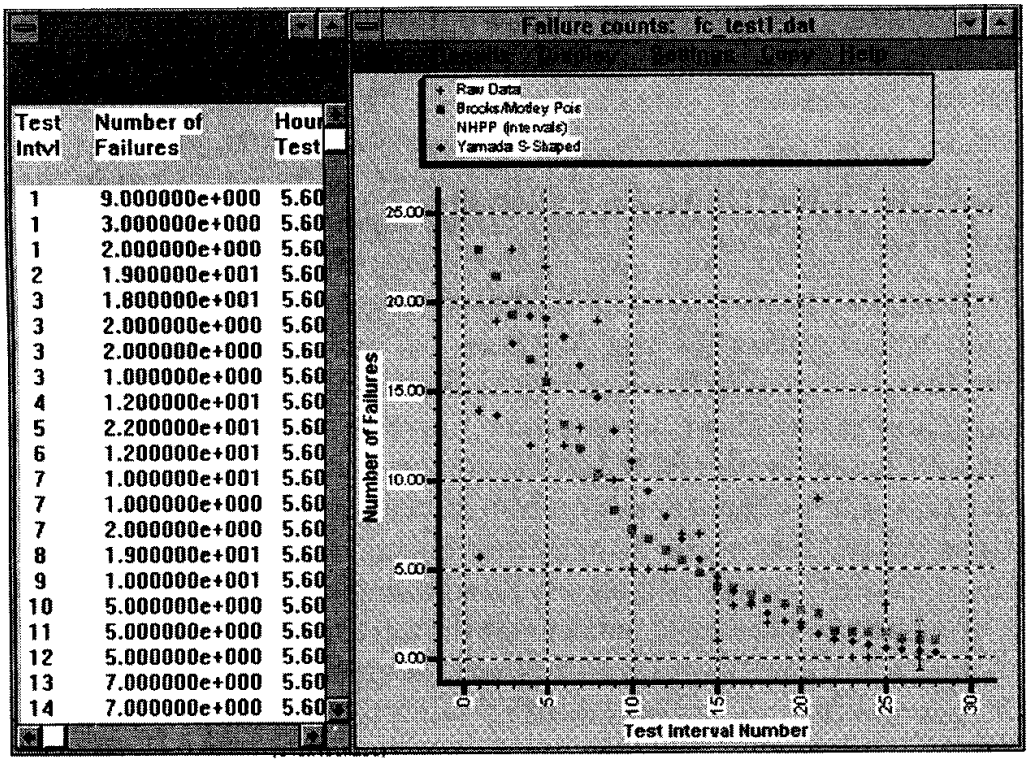

Figure A.5 CASRE model results display.

A.9 Economic Stop Testing Model (ESTM) Tool

The Bellcore ESTM System is a command-line-driven system that can be used to help decide when to stop testing a large software system. The package determines the optimal stopping time using a birth-death model for the introduction of faults, and an economic model to measure the trade-off between shipping and further testing. Unlike other packages, the ESTM System assumes that the system under test changes over time. A measure of this change, usually the number of NCNCSL (new or changed noncommentary source lines) is part of the input to the model.

The methodology implemented by the ESTM System is described in [Dala90, Dala92, Dala94]. The package runs on any platform that runs the UNIX® operating system and supports the C and PostScript™ languages.

The authors of the ESTM System suggest that a time metric other than calendar days be used to measure testing effort; their recommendation is to use the number of staff days spent by the testing team. Some (Bourne) shell scripts and an X-Window System™ program are provided to assist in collecting the extra data (staff time and NCNCSL). (The ESTM System has been used with other metrics measuring effort like the number of transactions processed by the system.)

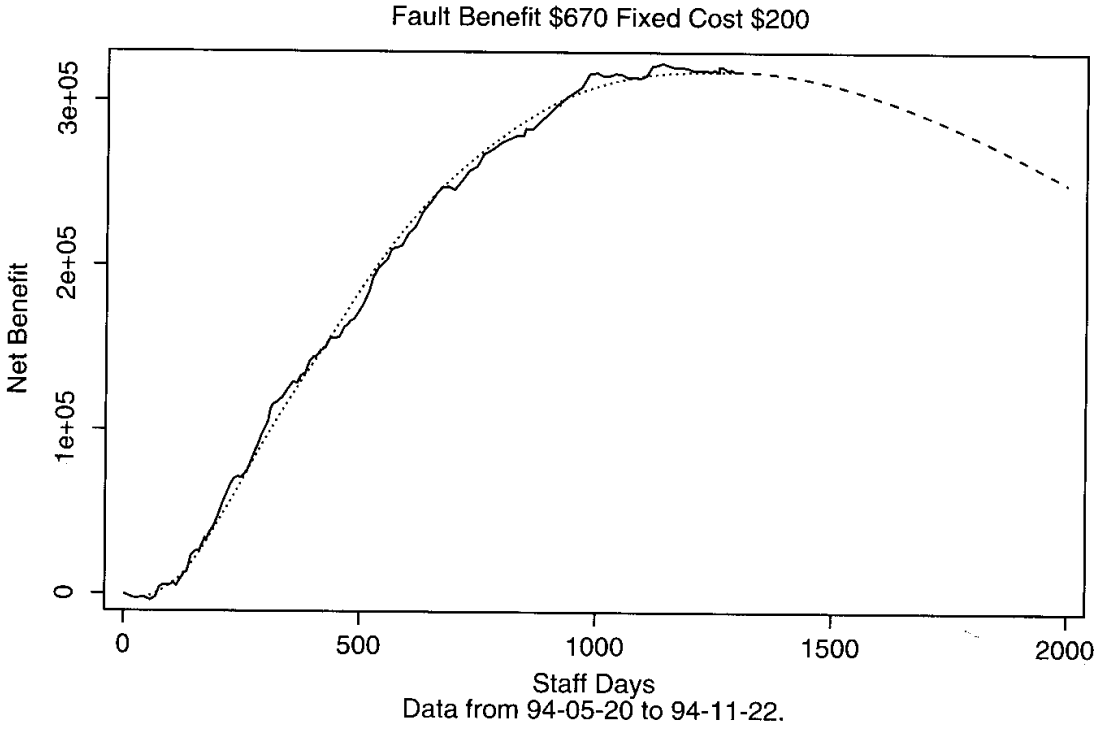

Figure A.6 Sample ESTM output.

The program that fits the model requires four columns of interval-domain data as input: the date, the number of staff days spent testing on that date, the number of faults found on that date, and the cumulative number of NCNCSL added to the system as of that date. Provision is made for adding historical data from other relevant systems (usually previous releases of the same system) to improve the accuracy of predictions.

The main output of the ESTM System is a series of plots, written in the PostScript language. The plots are intended to help assess the fit of the model, as well as help decide when to stop testing. A sample plot of ESTM output is shown in Fig. A.6.

A.10 Tool Comparisons

Several additional tools are available to the software reliability engineer, although most can be replaced by the tools highlighted in this chapter. For example, the GOEL tool [Vinn87] which is distributed through the U.S. Data and Analysis Center for Software (DACS) provides only parameter estimation for the Goel-Okumoto model, which is a basic NHPP model that is implemented in all of the above tools.

Besides the tools highlighted in this chapter, other tools are available to the software reliability engineer and new ones are becoming available every year. Tables A.1 and A.2 provide an overall comparison

TABLE A.1 Information About Current Software Reliability Estimation Tools

Tool name	Supplier	Contact	Models	Hardware	Minimum operating system
Statistical modeling and estimation of reliability functions for software (SMERFS)	Naval Surface Warfare Center (NSWC)	Dr. William Farr NSWCDD-B10 17320 Dahlgren Rd., Dahlgren, VA 22448 (540) 653-8388	Littlewood/Verrall Musa Basic Musa/Okumoto Jelinski-Moranda Geometric Execution Time NHPP Generalized Poisson NHPP Brooks/Motley Schneidewind S-Shaped	Cyber 170/760, DEC VAX, IBM PC	DEC VMS, MS DOS 3.0, Cyber Operating System
Software Reliability Modeling Programs (SRMP)	Reliability and Statistical Consultants, Ltd.	Dr. Bev Littlewood Center for Software Reliability Northampton Square London EC1V0HB, England (+44)-71-477-8420	Musa/Okumoto Duane Jelinski/Moranda (JM) Goel/Okumoto Bayesian JM Littlewood/Verrall Littlewood Keiller/Littlewood Littlewood NHPP	IBM PC	MS DOS 3.0
GOEL	Data & Analysis Center for Software (DACS)	DACS P.O. Box 120 Utica, NY 13503 (315) 734-3696	Goel/Okumoto	IBM PC	MS DOS 2.11
ESTM	Bell Communications Research	Dr. Sid Dalal Bellcore 445 South Street Morristown, NJ 07960 (201) 829-4292	Goel/Okumoto with economic testing criteria	Sun, HP, Dec Workstations	UNIX® System
AT&T SRE Toolkit	AT&T Bell Laboratories	Dr. Michael Lyu AT&T Bell Laboratories 600 Mountain Ave. Murray Hill, NJ 07974 (908) 582-5366	Musa Basic Musa/Okumoto	Any platform running UNIX® System V or MS/DOS	see above
SoRel	LAAS-CNRS	Dr. Karama Kanoun LAAS-CNRS 7, avenue du Colonel Roche 31077 Toulouse Cedex France (+33) 61 33 6235	Goel/Okumoto Littlewood/Verrall Hyperexponential Yamada S-shaped	Macintosh II with a math coprocessor	Macintosh
CASRE	NASA COSMIC	Ms. Karen Newcomb COSMIC The University of Georgia 382 East Broad Street Athens, GA 30602 (706) 542-7265	Littlewood/Verrall Musa Basic Musa/Okumoto Jelinski-Moranda Geometric Execution Time NHPP Generalized Poisson NHPP Brooks/Motley Schneidewind S-Shaped	IBM PC	MS-DOS 5.0 or higher with Windows 3.1 Windows NT Windows 95

TABLE A.1 Information About Current Software Reliability Estimation Tools (*Continued*)

Minimum memory	Release data: Current version	Release data: Current release	Release data: Original release	Release data: Distributed copies	Development language	Program developed for: Commercial use	Program developed for: Project-specific use	Program structure: Menu-driven	Program structure: Command-driven	Program structure: Integrated system	Program structure: Stand-alone tool	Cost ($)
256K	5.0	Oct-93	Oct-83	More than 200	FORTRAN 77	X		X		X		$59.95 (free with this book)
500K	1.0	May-88	May-88	Unknown	FORTRAN	X			X	X		$5,000
256K	1.0	Nov-87	Nov-87	68	Unknown	X		X			X	$50
—	1.0	Jun-93	1987	Unknown	C	X				X	X	Call Bellcore
120K	1.0	May-91	May-91	> 200	C	X			X		X	Free with this book
200K	1.0	May-91	May-91	Unknown	Pascal	X (all documentation in French)		X	X		X (needs Excel™ for plotting)	Free
8 MB	1.3	1994	1995	Unknown	FORTRAN	X		X		X		$100 for software (free with this book) $36 for documentation

TABLE A.2 Comparisons Among Available Software Reliability Estimation Tools (Part 1)

	Available models			Parameter estimation		Inputs			Performance measures and confidence intervals		
Tool name	Time domain	Interval domain	Total	Max. likeli-hood	Least squares	Time between failure	Failure counts and interval length	Resource parameters	Current relia-bility or failure rate	Expect total failure	Expect faults remain-ing
AT&T	2	2	2	X		X	X	X	X	X	X
SMERFS	6	6	12	X	X	X	X	X	X	X	X
SRMP	9	0	9	X		X			X		
SoRel	3	3	4	X	X	X	X		X	X	
CASRE	10	6	16*	X	X	X	X	X	X	X	X
GOEL	1	1	1	X		X	X	X	X	X	X
ESTM	0	2	2	X			X	X	X	X	X

* NOTE: CASRE has the capability to combine models to create new ones

TABLE A.2 Comparisons Among Available Software Reliability Estimation Tools (Part 2)

	Data manipulation					User interface					
Tool name	Storage retrieval	Transfor-mations	Editing	Can use externally created data	Handles missing values	On-line help available	Off-line support	Interactive processing	Batch processing	Error messages	High-resolution graphics
AT&T				X			X		X	X	X
SMERFS	X	X	X	X		X	X	X		X	
SRMP				X					X	X	
SoRel		X	X	X		X	X	X		X	X
CASRE	X	X	X	X		X	X	X	X	X	X
GOEL				X		X		X		X	
ESTM	X			X					X	X	X

of the tools highlighted in this chapter as well as some additional tools. Table A.1 provides an organization-level look at the tools; Table A.2 provides the analyst view. The list of criteria used in the tables is not exhaustive. In particular, we have chosen not to include computer time efficiency, computer storage requirements, or ease of learning the tool, since we believe these depend heavily on the computer used, the analyst's skill, and personal bias.

At the publication of this book a new tool, M-elopee, is also made available. M-elopee is implemented for Windows™ environment, integrating software reliability models and relational databases. Interested readers may contact Mathix, 19 rue du Banquier, 75013 Paris France. Tel: 33(1)43-37-76-00, FAX 33(1)43-37-00-73.

Performance measures and confidence intervals			Model estimates and analytical tools								
Time-to-next-failure dist.	Time-to-*K*th-failure dist.	Schedule or cost estimate	Parameter estimate	Variance of estimates	Model goodness-of-fit	*u*-plot	*y*-plot	Noise	Residue	Basic statistics	Residual plots
		X	X								
X	X		X		X	X	X	X		X	X
X	X		X		X	X	X	X			
X			X	X	X	X			X		X
X	X		X		X	X	X	X		X	
X	X	X	X	X	X					X	
		X	X		X						X

Output formatting options selectable						Documentation					Product support		
Line printer graphs	Table output	Page size	Fonts	Titles	Color	Explain model theory	Explain tool usage	Sample run	Explain error messages	Aids result interpretation	Full support	No support	Support no commitment
	X			X			X	X			X		
X	X			X		X	X	X		X			X
X	X					X	X	X		X			X
	X					X	X	X	X	X			X
	X	X	X	X	X	X	X	X	X	X			X
X	X					X	X	X		X		X	
				X		X	X	X	X	X			X

A.11 Summary

We highlight several software reliability tools currently available in the market. For each tool we briefly describe its functionality, capability, and user interface. Sample execution or output of each tool is also provided. Note that AT&T SRE Toolkit, SMERFS, CASRE, and SoftRel (the software reliability simulation tool described in Chap. 16) are included in the Data and Tool Disk.