CASRE – A Computer-Aided Software Reliability Estimation Tool

Michael R. Lyu
ECE Department
The University of Iowa

Allen Nikora
Jet Propulsion Laboratory
California Institute of Technology

Abstract

In this paper, we describe the construction of a CASE tool for a systematic and automatic application of software reliability modeling for real-world projects. Instead of proposing more new models, we focus on the practical project applications of existing software reliability models for better software reliability estimations. We build this CASE tool on top of a number of existing software reliability models, called component models, as the baseline for software reliability measurement. The main advancement of this tool over other similar tools is highlighted by its capability in constructing various reliability estimations under a new paradigm to linearly combine the component models. Moreover, this tool features its enhanced graphical user-interface which greatly facilitates the potentially tedious application procedure for software reliability estimation.

1: Introduction

Software reliability is defined as the probability of successful software operation without encountering failures during a period of time under a specified operational environment. This basic definition applies to all software systems, subsystems, and individual units. Since the complexity and size of software systems are growing dramatically, the capability to estimate software reliability becomes one of the major challenges for software engineers. Such an estimation presents important indicators for the quality of software products, and provides insight into the software design process so that areas for improvement could be identified for software reliability engineering.

The capability to measure software reliability has been the prerequisite for software reliability engineering. Consequently, software reliability modeling and estimation have drawn researchers' and practitioners' attentions over the past twenty years. Traditionally, software reliability modeling is a set of techniques that apply probability theory and statistical analysis to assess the achieved reliability of work products, both quantitatively and objectively. A software reliability model specifies the general form of the dependence of the failure process on the principal factors that affect it: fault introduction, fault manifestation, failure detection and recovery, fault removal, and operational environment[1].

Originally, people focused on the definition and description of new software reliability models, hoping that a unified model could emerge to serve as the best estimator for software reliability. A major difficulty in software reliability engineering practice is to analyze the particular context in which reliability measurement is to take place so as to decide *a priori* which model is likely to be trustworthy. Due to the intricacy of human activities involved in software development and operation process, as well as the uncertain nature of software failure patterns, such a priori has never been conclusive. It has been shown that there is no best software reliability model for every case under all circumstances [2], [3]. As a result, practitioners are left in a dilemma as to which software reliability models to choose, which procedures to apply, and which prediction results to trust, while contending with varying software development and operation practices.

Since the selection and application of the software reliability models as well as the search for the best estimates may involve tedious computation-intensive tasks, a computer-aided approach is inevitable. For this purpose, This paper presents a CASE tool, called Computer-Aided Software Reliability Estimation (CASRE) system, for an automatic and systematic approach in estimating and engineering software reliability. In the remainder of the paper, Section 2 describes the overall architecture of CASRE. Section 3 introduces the new software reliability approach by linear combination models featured in CASRE. Some CASRE project applications are presented in Section 4. Finally, Section 5 addresses conclusions and future work.

2: The CASRE tool – high-level structure and functionality

Currently available software reliability tools allow the user to apply any one of the more well-known software reliability models to a development effort. In addition to allowing the user to make reliability estimates, some of these tools also allow the practitioner to determine the applicability of a particular model to a set of failure data. However, there is additional functionality that would be useful if implemented. Namely, results from different models may be combined in various ways to yield reliability estimates whose predictive quality is better than the individual models themselves[4].

Secondly, the graphics capabilities of most currently-available tools are rather limited in that the variety of graphs produced by the tool is small. For instance, several tools [5] produce plots of actual and estimated failure frequencies or interfailure times, but do not allow the user to produce plots of actual and estimated cumulative number of failures directly. Others produce u-plots and y-plots[6] which help the user in determining the applicability of a particular model to a specific set of failure data, but do not produce plots that would be of use to a software manager in determining the status of a development effort (actual and estimated failure rates, cumulative failures, reliability growth, etc.).

Finally, many of the currently-available tools were implemented at a time when high-resolution displays and windowing systems such as X-Windows were not as widely available as they are today. This affects the quality of the graphics displayed (two of the better-known tools display character-based graphics) and may have influenced the tools' ease-of-use. For example, the menu-driven interface for one widely-used public domain tool [5] is implemented such that a user may have to "back out" of several layers of submenus to access another top-level menu. Current technology makes it a fairly routine matter to implement a menu-driven application in which the user can directly select a top-level menu item without having to back out of the current sub-menu.

Figure 1 shows the proposed high-level architecture for CASRE, whose major functional areas are: (1) Data Modification, (2) Failure Data Analysis, (3) Modeling and Measurement, and (4) Modeling/Measurement Results Display.

Figure 1: High-level architecture for CASRE

2.1: Data modification

CASRE allows users to create new failure data files, modify existing files, and perform global operations on files.

- Editing

CASRE allows users to create or alter failure history data files. A simplified spreadsheet-like user interface allows users to enter time between failures or test interval lengths and failure counts from the keyboard. Users are also allowed to invoke a preferred editor (e.g. emacs or vi).

- Smoothing

Since input data to the models is often fairly noisy, the following smoothing techniques are proposed:

  - Sliding rectangular window
  - Hann window
  - Polynomial fit
  - Specific cubic-polynomial fits (e.g. B-Spline, Bezier Curve)

Users select smoothing techniques appropriate to the failure data being analyzed. The smoothed input data can be plotted, used as input to a reliability model, or written out to a new file for later use. Summary statistics for the smoothed data can also be displayed (see "Failure Data Analysis" below).

- Data Transformation

In some situations, logarithmic, exponential, or linear transformations of the failure data produce better or more understandable results. The following operations, currently available in some tools, allow users to transform an entire set of failure data in this manner.

  - log(a * x(i)) + b); x(i) represents a failure data item, and a and b are user-selectable scale factors
  - exp(a * x(i) + b)
  - x(i) ** a
  - x(i) + a
  - x(i) * a
  - user-specified transformation

As with smoothing, users select a specific transformation. Users are able to manipulate transformed data as they would smoothed data.

2.2: Failure data analysis

The "Summary Statistics" block in Figure 1 allows users to display the failure data's summary statistics, including the mean and median of the failure data, 25% and 75% hinge points, skewness, and kurtosis.

2.3: Modeling and measurement

Figure 1 shows two modeling functions. The "Models" block executes single software reliability models on a set of failure data. The "Model Combination" block allows users to execute several models on the failure data and combine the results of those models. We include this capability because our experience in combining the results of more than one model indicates that such "combination models" may provide more accurate reliability predictions than single models[4]. The block labeled "Model Evaluation" allows users to determine the applicability of a model to a set of failure data.

- Single Model Execution

Based on our experience in applying software reliability models, we include the following models in CASRE:

(1) Bayesian Jelinski-Moranda Model (BJM)[7], [8], [9], [10]
(2) Brooks and Motley Model (BM)[11]
(3) Duane Model (DU) [12], [13]
(4) Geometric Model (GM) [11]
(5) Goel-Okumoto (GO) [14]
(6) Jelinski-Moranda (JM) [15], [16]
(7) Keiller-Littlewood Model (KL)[17], [18]
(8) Littlewood Model (LM) [19]
(9) Littlewood non-homogeneous Poisson Process Model (LNHPP) [2]
(10)Littlewood-Verrall (LV) [20]
(11)Musa-Okumoto (MO) [21]
(12)Generalized Poisson Model (PM)[11]
(13)Schneidewind Model (SM)[22]
(14)Yamada Delayed S-Shape Model (YM) [23]

The models should be implemented to allow input to be in the form of interfailure times or failure frequencies.

CASRE allows users to choose the parameter estimation method (maximum likelihood, least squares, or method of moments). Model outputs include:

– Current estimates of failure rate/interfailure time
– Current estimates of reliability
– Model parameter values, including high and low parameter values for a user-selectable confidence bound
– Current values of the pdf and cdf
– The probability integral transform u_i [2]
– The normalized logarithmic transform of u_i, y_i [2]

Users can display these quantities on-screen or write them to disk.

• Combination Models

CASRE allows users to combine the results of several models according to various combination schemes discussed in Section 3. Users may also be allowed to define their own combination schemes. The resulting combination models could be further used as the component models to form another combination model.

• Model Evaluation

CASRE includes the following statistical methods to help users determine the applicability of a model (including "combination models") to a specific failure data set:

– Computation of prequential likelihood (PL) function (the "Accuracy" criterion).
– Determination of the probability integral transform u_i, (plotted as the u-plot - the "Bias" criterion).
– Computation of y_i to produce the y-plot (the "Trend" criterion).
– Noisiness of model predictions (the "Noise" criterion).

The Akaike Information Criterion (AIC)[24], similar in concept to prequential likelihood, could also be implemented. This model evaluation function would also compute goodness-of-fit measures (e.g. Chi-Square test). The PL and AIC outputs are used as input to "Model Combination" to determine the relative contribution of individual models if the user has specified a combination model.

2.4: Display of results

CASRE graphically displays model results in the following forms:

– Interfailure times/failure frequencies, actual and estimated
– Cumulative failures, actual and estimated
– Reliability growth, actual and estimated

Actual and estimated quantities are available on the same plot. Plots include user-specified confidence limits. Users are able to control the range of data to be plotted as well as the usual cosmetic aspects of the plot (e.g. X and Y scaling, titles). In a windowing environment, multiple plots could be simultaneously displayed. CASRE allows users to save plots displayed on-screen as a disk file or to print them. One public-domain tool, SMERFS [5] version 4, can write the data used to produce a plot to a file that can be imported by a spreadsheet, a DBMS, or a statistics package for further analysis. CASRE also includes this capability.

The plotting function also produces u-plots and y-plots from Model Evaluation's u_i and y_i outputs. These plots indicate the degree and direction of model bias and the way in which the bias changes over time.

2.5: On-screen appearances

Figures 2(a) − 2(f) show a series of screen dumps for the described CASRE tool. It can been seen that the application of models to failure data is a straightforward process. The user is also given a considerable amount of choice in the models to be applied. This combination of simple operation and variety in the available models makes it easy for the user to identify an appropriate model for a particular development effort or investigate a family of models.

• Screen 1 – initial failure data display

The screen is shown in Figure 2(a). After opening a failure history file from the "File" menu, the contents of the file are displayed in tabular and graphic forms. The tabular representation resembles a spreadsheet, and the user can perform similar types of operations (e.g. selecting a range of data, deleting one or more rows of data). All of the fields can be changed by the user except for the "Interval Number" field (or "Error Number" field if the data is interfailure times). In this example, the selected data set is in the form of test interval lengths and number of failures per test interval. The user can scroll up and down through this tabular representation and resize it as per the MOTIF conventions.

The large graphics window displays the same data as the worksheet. If the failure data set is interfailure times, the initial graphical display is interfailure times. If, as in this example, the failure data set is test interval lengths and failure counts, the initial graphical display is the number of failures per test interval. The display type can be changed by selecting one of the items from the "Display Type" menu associated with the graphics window. The user can move forward and backward through the data set by pressing the right arrow or left arrow buttons at the bottom of the graphics window.

Finally, the iconified window at the lower left corner of the screen lists the summary statistics for the data. To open this window, the user clicks on the icon. The following information is then displayed in a separate window:

- Number of observations in this data set
- Type of observations made (interfailure times or test interval lengths and failure counts)
- Mean value of the observations
- Minimum and maximum values
- Median
- 25% and 75% hinges
- Standard deviation and variance
- Skewness and Kurtosis

• Screen 2 – selecting failure data range

The screen is shown in Figure 2(b). The user will frequently use only a portion of the data set to estimate the current reliability of the software. This is because testing methods may change during the testing effort, or different portions of the data set may represent failures in different portions of the software. To use only a subset of the selected data set, the user may simply "click and drag" on the tabular representation of the data set to highlight a specific range of observations. The user may also select previously-defined data ranges. To do this, the user chooses the "Select Range" option of the Edit menu. This brings up a dialogue box containing a scrolling text window in which the names of previously-defined data ranges and the points they represent are listed. To select a particular range, the user highlights the name of the range in the scrolling text window and presses the "OK" button. Pressing the "Cancel" button removes the dialogue box and the Edit menu from the screen.

Once a range has been selected, all future modeling operations will be only for that range. The selected data range is highlighted in the tabular representation. The graphics display will change to include only the highlighted data range. All other observations will be removed from the graphics display.

• Screen 3 – applying software reliability models

The screen is shown in Figure 2(c). After the user has opened a file, selected a data range, and done any smoothing or other transformation of the data, a software reliability model can be run on the data. In the Model menu, the user has the choice of 13 individual models or a set of models which combine the results of two or more of the individual models. The user may also choose the method of parameter estimation (maximum likelihood, least squares, or method of moments), the confidence bounds that will be calculated for the selected model, and the interval of time over which predictions of future failure behavior will be made.

• Screen 4 – prioritization of model selection criteria

The screen is shown in Figure 2(d). There are many models from which to choose in this tool. The user may not know which model is most appropriate for the data set being analyzed. Using CASRE, the user can request, "display the results of the individual model which best meets the four prioritized criteria of accuracy (based on prequential likelihood), biasedness, trend, and noisiness of prediction." To do this, the user first selects the "Individual" option of the Model menu. A submenu then appears, on which 13 individual models are listed, as well as a "Choose Best" option. The user selects the "Choose Best" option, which results in a "Selection Criteria" dialogue box being displayed. The user moves the four sliders in this dialogue box back and forth to establish the relative priorities of the four criteria. Numerical values of the priorities are displayed in the text boxes on the right side of the dialogue box. Once the priorities have been established, the user presses the "OK" button. CASRE then proceeds to run all of the individual models against the data set, first warning the user that this is a time-consuming operation and allowing cancellation of the operation. If the user continues, CASRE provides the opportunity for cancellation at any time if the user decides that the operation is taking too much time.

• Screen 5 – display of model results

The screen is shown in Figure 2(e). Once a model has been run on the failure data, the results are graphically displayed. Actual and predicted data points are shown, as are confidence bounds. The model is identified in the window’s title bar; the percent confidence bounds are given at the right side of the graphics window. This concludes one round of software reliability measurement with CASRE.

- Screen 6 – determination of model bias

The screen in which one of the result of model evaluation can be displayed is shown in Figure 2(f). Statistical methods can be applied to determine the applicability of the model to the failure data set on which it was executed. To display the evaluation results, the user select the "Evaluations" pull-down menu in the graphics display window's main menu bar. Several model evaluation methods, including u-plots and y-plots, are available to the user. In this example, the user has chosen the "U-Plot" menu item. The u-plot, which indicates biases in the model, is displayed on screen. CASRE also indicates whether the model has an optimistic bias (predictions of time to the next failure tend to be greater than observed inter-failure times) or a pessimistic bias. To return to the display shown in the figure, the user may select the "Display Type" pull-down menu and choose the desired type of reliability-related display.

3: Introduction of the combination models

The major feature of CASRE is the introduction of a new and practical approach toward software reliability measurement which tends to produce better reliability estimations. This general combination modeling approach is formalized as follows:

1. Identify a basic set of models (called *component* models).
2. Select models that tend to cancel out in their biased (if any) predictions.
3. Keep track of the software failure data with all the component models.
4. Apply certain criteria to weigh the selected component models and form one or several *linear combination models* for final predictions. The weights could be either constants or variables which are dynamically changed with time.

In general, this model could be expressed as follows:

$$\tilde{f}_i(t) = \sum_{j=1}^{n} \omega_j(t)\, \tilde{f}_i^j(t)$$

Where $\tilde{f}_i^j(t)$ is the predictive probability density function of the jth component model, given that $i-1$ data of times between successive failures have been observed. Note that $\sum_j \omega_j(t) = 1$ for all t's.

As an example to illustrate this combination approach, we chose GO, MO, and LV in CASRE as the three component models to form a set of linear combination models. Reasons for choosing these three component models are:

1. Their predictive validity has been observed in our recent investigation[3]. In fact, they are judged to perform well by many practitioners [26], [27], and they have been widely used.
2. They represent different categories of models: GO (similar to JM and SM) represents the exponential shape non-homogeneous Poisson process (NHPP), MO represents the logarithmic shape NHPP model, and LV represents the inverse-polynomial shape Bayesian model.
3. Their predictive biases tend to cancel: GO tends to be optimistic, LV tends to be pessimistic, and MO might go either way.

As a result, we formulated a set of four combination models as follows:

1. *ELC - Equally-Weighted Linear Combination Model*
   This model is formed by assigning the three component models a constant, equal weight[28]. The arithmetic average of all component models' predictions is taken as the ELC model prediction, namely, $ELC = \frac{1}{3}GO + \frac{1}{3}MO + \frac{1}{3}LV$. These weightings remain constant and unchanged throughout the modeling process. The motivation of this approach is to reduce the risks of relying on any particular model, while preserving the simplicity of the prediction process.

2. *MLC - Median-Oriented Linear Combination Model*
   Instead of choosing the arithmetic mean for the prediction in ELC, the median value is used. In other words, each time when a prediction is called for, the component model whose predicted value is the median is selected as the output of this model. The justification for this approach is that median might be a more moderate choice than the mean in some cases, since it can better tolerate an erroneous prediction which is far away from the others.

3. *ULC - Unequally-Weighted Linear Combination Model*
   This model is similar to MLC except that instead of

being dominated by the median value, the optimistic and pessimistic predictions will get certain weightings in the outcome of the final prediction. Here we use the weightings similar to *Program Evaluation and Review Technique* (PERT), i.e., the formulation of this model is $\frac{1}{6}O + \frac{4}{6}M + \frac{1}{6}P$, where O represents an optimistic prediction, P represents a pessimistic prediction, and M represents the median prediction.

4. *DLC - Dynamically-Weighted Linear Combination Model*
   In this model, we use a *meta-predictor* to form a linear combination of several predictions with the weightings chosen in some optimal way (e.g., posterior probabilities)[2]. A Bayesian interpretation of "prequential likelihood" as *a posteriori* could be dynamically calculated in a long run or in a short time window to determine the weight assignments. Here we pick one time frame prior to each prediction as the reference in assigning weights. For this model, the weighting function for each of the component models varies with time.

To catch the local trend of the prediction accuracy, the described DLC model uses only one time frame as the reference to determine the weights on the component models. This "one observation window" approach might lose the global trend in the measurement effort. It is natural to extend the window size to a larger number, say, N, as the reference. This leads to two types of DLC models:

(1) *DLC/F/N* ("DLC with Fixed N-size-window") model: Make a weight assignment by observing the predictive accuracy on N predictions. The weight assignment remain fixed for the next N predictions, at the end of which the weightings will be recomputed according to that N predictions. So on and so forth.

(2) *DLC/S/N* ("DLC with Sliding N-size-window") model: Make a weight assignment by observing the prediction accuracy on N predictions. The weight assignment is recomputed each time for a new prediction, using the observed predictive accuracy of the most recent N predictions.

The difference of these two models is illustrated by Figures 3(a) and 3(b).