# Frestimate Users Manual Version 3.80

## 1.0 Essentials

**Prior to using FREstimate, it is essential that you have an understanding of the purpose and steps for doing a software reliability prediction! The "Big Picture" illustrates the major steps involved in doing a prediction. It is also strongly recommended that you understand the terms used in the prediction process. SoftRel provides training in these two areas (see www.softrel.com). This training is strongly recommended for users who have never preformed a software reliability prediction.**

### *1.1 Software Reliability Big Picture*

**There are two types of software reliability measurements.**

**Prediction models**- regardless of whether they are for software reliability or any other application - are developed by collecting trained data and observing relationships in that features and some outcome. In the case of software reliability the outcome is delivered defects normalized by code size. The features vary from model to model and are generally related to development practices. Some models have only one feature. Some models have many features. The model is the mathematical expression that determines some outcome given some set of features.

Predictors are used early in the development lifecycle to:

- Determine whether the current capabilities/development practices are suitable for meeting a system reliability objective
- Select the development practices that would allow the system reliability objective to be met
- Determine whether vendor supplied software will meet a system objective
- Determine suitable quality and reliability objectives for the software
- Determine staffing requirements for maintenance and testing
- Predict the inherent number of defects in the software at the start and end of testing

**Estimation models** - are models that project the future based on what has happened in the immediate past - on this project. Estimators do not use trained data like predictors; they use data collecting only from the project in which we are interested in measuring. Estimators have a variety of purposes including:

- Projecting how many more hours of testing are needed to reach some reliability objective
- Projecting how many more defects must be detected and then fixed to reach some reliability objective.
- Validating a reliability prediction

*1.2 The Prediction Big Picture.*

If you select *Help->Get prediction started* you will see the Big Picture literally speaking.

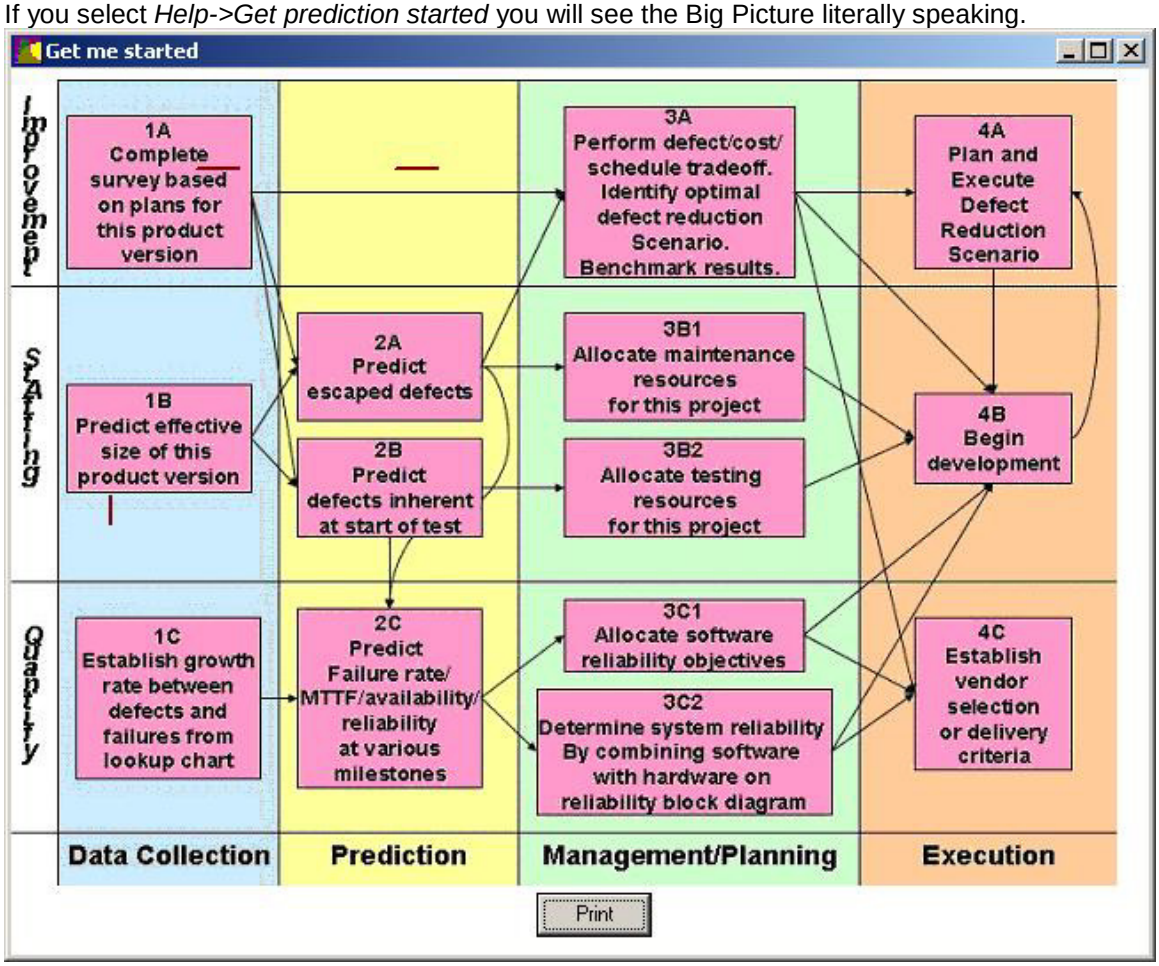



**Figure 1- The Big Picture**

There are three general objectives:

- Improve
- Staff
- Quantify

There are 4 steps to executing each of these objectives:

- Data collection (orange nodes)
- Prediction (blue nodes)
- Management/planning (green nodes)

The wizards as well as the HTML file which is launched from the main dialog are designed to walk you through each of these components and help you get your project set up.

**Table 1 - Tasks required**

| Goal | Tasks required |
|---|---|
| *Improvement goals* | |
| Reduce defects that are visible to your customer(s) by a specific predefined percentage | Predict deployed defects |
| Assist a vendor to reduce defects that are delivered to you from a vendor | Predict deployed defects |
| *Staffing goals* | |
| Predict defects that will be found by customer so as to plan for warranty and maintenance | Predict staffing profile |
| Predict defects that will be found during testing so as to plan resources required to find those defects | |
| *Quantify goals* | |
| Identify the system reliability for hardware and software (usually to provide to one or more of your customers) | 1A, 1B, 1C, 2A, 2C, 3C2 |
| Establish vendor selection or delivery criteria for vendors supplying only software | Defect driven - 1A, 3A, 4C OR<br>Reliability driven -1A, 1B, 1C, 2C, 3C1, 4C<br>Performed on each vendor version one at a time |
| Establish vendor selection or delivery criteria for vendors supplying hardware and software | 1A, 1B, 1C, 2A, 2C, 3C2, 4C performed on each vendor system one vendor/system at a time |

*1.4 Terms and Definitions*

**Defect density** – Defects per unit of size.

The number of defects can be determined by multiplying the defect density by the predicted size either EKSLOC or effective function points. If the effective KSLOC or effective function points is blank, then the KSLOC or function points is used. . The unit of measure (whether KSLOC or function points).

**Defect density confidence –** The confidence on the defect density prediction is determined by the model used to predict defect density. Some models, such as the SEI CMMi or industry models have a lot of variation so the confidence bounds are higher than with the Shortcut or Full-scale models. The confidence is used to establish the upper and lower bounds on the defect predictions.

**Failure rate –** Failure rate is computed using a general exponential model. It is a function of defects predicted, duty cycle and growth period**.**

The end of test failure is measured at delivery time.

**$\lambda_{del}$ = End of Test Failure rate = $N_{del}$ (1- exp(-$Q_{del}$)) / T**

The operational failure rate is measured at the end of the defined growth period. If you have input 0 for growth period then these two predictions will have the same value.

**T** = Number of hours operating per month or the duration in hours of that time interval. If the time interval is a month and there is one system running the software full time then T = 730 hours for example.

**TF** = Growth period

**$N_{del}$** = number of inherent defects delivered. This is determined by using any prediction technique and multiplying by the KSLOC or function points (depending on what you set the unit of measure to).

**$N_0$** = how many defects are predicted to be found during testing.

$\lambda_0$ = initial failure rate prediction - failure rate at the start of testing. The start of testing is when the failure rate is at its largest.

**$MTTF_i$ = T / ($N_{del}$ ((exp (-Q/TF)*(i-1))-exp(-Q/TF)*i))**

i = month after delivery for which MTTF is being solved for

**MTBI = Mean Time Between Interruptions**

**Interruptions –** Some defects can be resolved without a change to the source code. For example, defects that have a viable workaround. These are called interruptions.

**$N_{del}$** = number of inherent defects delivered. This is determined by using any prediction technique and multiplying by the KSLOC.

**Operational Failure $rate_i$ = failure rate after i months of growth after delivery = $N_{del}$ ((exp (-$Q_{del}$/TF)*(i-1))-exp(-$Q_{del}$/TF)*i) / T**

**$MTTF_i$ –** Since an exponential model is used to calculate failure rate, the Mean Time To Failure is **= 1/ failure rate.** MTTF is predicted based on all defects predicted to require a corrective action.

**$MTBI_i$ = $MTTF_i$ * Ratio of interruptions to defects that require a corrective action**

**MTTCF or Critical MTTF** - MTTF computed using only those defects that impact availability and have no workaround.

**Percentage of defects that are critical –** This field is used to filter for outputs for only defects that will impact mission availability. If you ignore this field then no filtering is done.

**Critical defect density** – If you enter a value for percentage of defects that are critical, this value will be multiplied by the defect density for only the critical

**KSLOC** and the result is a defect density for only serious defects.
The number of serious defects can be determined by multiplying the severe defect density by the critical EKSLOC or critical function points. If the critical

**EKSLOC** or critical function points is left blank then the EKSLOC or effective function points is used. If those are blank then the KSLOC or function points is used. The unit of measure (whether KSLOC or function points) is determined by the General Inputs dialog.

**Critical failure rate** – If you entered a value for percentage of defects that are severe, a serious failure rate is computed using the serious defect density.

The end of test severe failure is measured at delivery time. The operational severe failure rate is measured at the end of the defined growth period. If you have input 0 for growth period then these two predictions will have the same value.

**Critical MTTF** – This is equal to 1/ severe failure rate since an exponential model is used to compute failure rate. The severe end of MTTF is measured at delivery time. The severe operational MTTF is measured at the end of the defined growth period. If you have input 0 for growth period then these two predictions will have the same value**.**

**Critical MTBI** – Critical MTTF * Ratio of interruptions to defects

**Duty cycle –** This is the percentage of time that the entire software system is operational per month**.**

**Component utilization –** Some components within a software system may be utilized more then others. Some components may be COTS (commercial off the shelf software) and may have different defect densities then other components. Some components may be reused and therefore also have a different defect density then others. If you have this data available, you can override the size inputs in the interim section of the All General Inputs page and input size and utilization estimates for each component of the software system. This method allows you to select different models for different components in your system as opposed to using one model for all components**.**

**Growth period –** This is the expected number of months after delivery in which the software will be repaired and therefore the reliability will grow. You can also think of this as how long it will take for the very last defect caused by this version to be found. Usually this is 2-3 major release cycles and is usually 36-48 months.

**Next release -** The time to the next release is the number of months until a major release (new

features). This is used to extrapolate the failure rate, MTTF, reliability and availability until the point in which the current version is replaced with a new version. The predictions at this milestone represent the best that the software will exhibit as long as new features continue to be added to it. The failure rate, MTTF, reliability and availability predictions at the end of growth milestone represent the best that the software will exhibit assuming that no new features are added.

**Average during release** – The failure rate, MTTF, reliability and availability predictions are computed as an average during the first release cycle. This is because this is the most useful metric for a user of your software. The predictions at the end of the release represent only a best case while the predictions of the average during the release are a more useful indicator.

**KSLOC** – The unit of measure for size is either KSLOC (1000 lines of source code) or function points. If you select KSLOC, then KSLOC from the general inputs page is used to compute defect density. If you select function points then function points from general inputs page is used.

**EKSLOC** is the effective KSLOC. Sometimes code is reused. That reused code has theoretically already been debugged. EKSLOC is KSLOC minus reused (tested!) KSLOC.

**Critical EKSLOC** is the EKSLOC that is associated with functions that may/will cause a mission critical failure. Not all KSLOC has the potential for causing a critical failure. Sometimes there may be auxiliary or optional functions that are useful but do not cause a system failure when are not available. If there is code in your system that will never contribute to a critical failure, then remove that EKSLOC from the count and enter the result in this field.

**Function points** - The unit of measure for size is either KSLOC (1000 lines of source code) or function points. If you select KSLOC, then KSLOC from the All General Inputs page is used to compute defect density. If you select function points then function points from the interim section of the All General Inputs page is used**.**

Sometimes code is reused. That reused code has theoretically already been debugged. Effective function points are function points minus reused (tested!) function points.

**Critical effective function points** is the effective function points that is associated with functions that may/will cause a mission critical failure. Not all function points have the potential for causing a critical failure. Sometimes there may be auxiliary or optional functions that are useful but do not cause a system failure when are not available. If there is code in your system that will never contribute to a critical failure, then remove that effective function points from the count and enter the result in this field.

**Executable size –** This is the number of bytes in the executable. This may be used for size if the neither function points nor KSLOC is known.
Software components – a software component is a component of the software system, which has functional characteristics. A software component may be a third party component. The top of the software architecture is a CSCI (Computer Software Configuration Item). This is either an executable or a DLL (Dynamic Link Library). Components are the next layer of architecture. Several components may constitute a CSCI. Several CSCI's may constitute a system. Within the CSC, there are usually several if not many CSU's (Computer Software Units). The Unit is the lowest level of architecture and is a function or procedure if a procedural language is used or a method if an object oriented language is used.

**Upper and lower bounds on size –** Since size is a prediction, you should supply the upper and lower bounds on the size prediction as these are used to compute the confidence of the size prediction. The relative error of the size prediction generally decreases with each phase of development until the code is completed and tested and at that time the size can be counted and is therefore known. At the requirements review milestone (the point in which the requirements

are signed off) the size prediction may be off by 50% or more. At the design review milestone 25% or more. During coding the size prediction may be within 10%. Finally when testing is complete the size prediction is no longer a prediction.

**Growth rate or Conversion ratio** – the growth rate or the conversion ratio converts defects to failure rate. It is computed either by historical data from projects at your company or by using the Rome Labs lookup chart.

**Growth rate confidence –** Since the growth rate is a prediction, we need to have a confidence value for it to establish the upper and lower bounds on the failure rate and MTTF predictions. When you select the industry type, the average confidence from our database of growth rates is also selected at the same time that the growth rate is selected.

**Expansion ratio** – This is the density of your language compared to assembler. FREstimate calculates all defect densities in terms of assembler. This allows you to multiply the prediction based on the expansion ratio that is appropriate for the language. You can override this value if you wish. FREstimate uses default values for each language.

**Life cycle phase**
The phases of the lifecycle range from concept to delivery. The life cycle phase is used to establish the default size prediction confidence bounds. Rome Laboratory model uses this field as well. Some of the factors in the Rome Laboratory model are phase specific.

**Application type**
The application type is used by the Industry model and the Rome Laboratory model. It is also used by the wizards to help you pick the model that works the best for your application type. When you select the application type, the Industry model computes an average defect density for that application type.

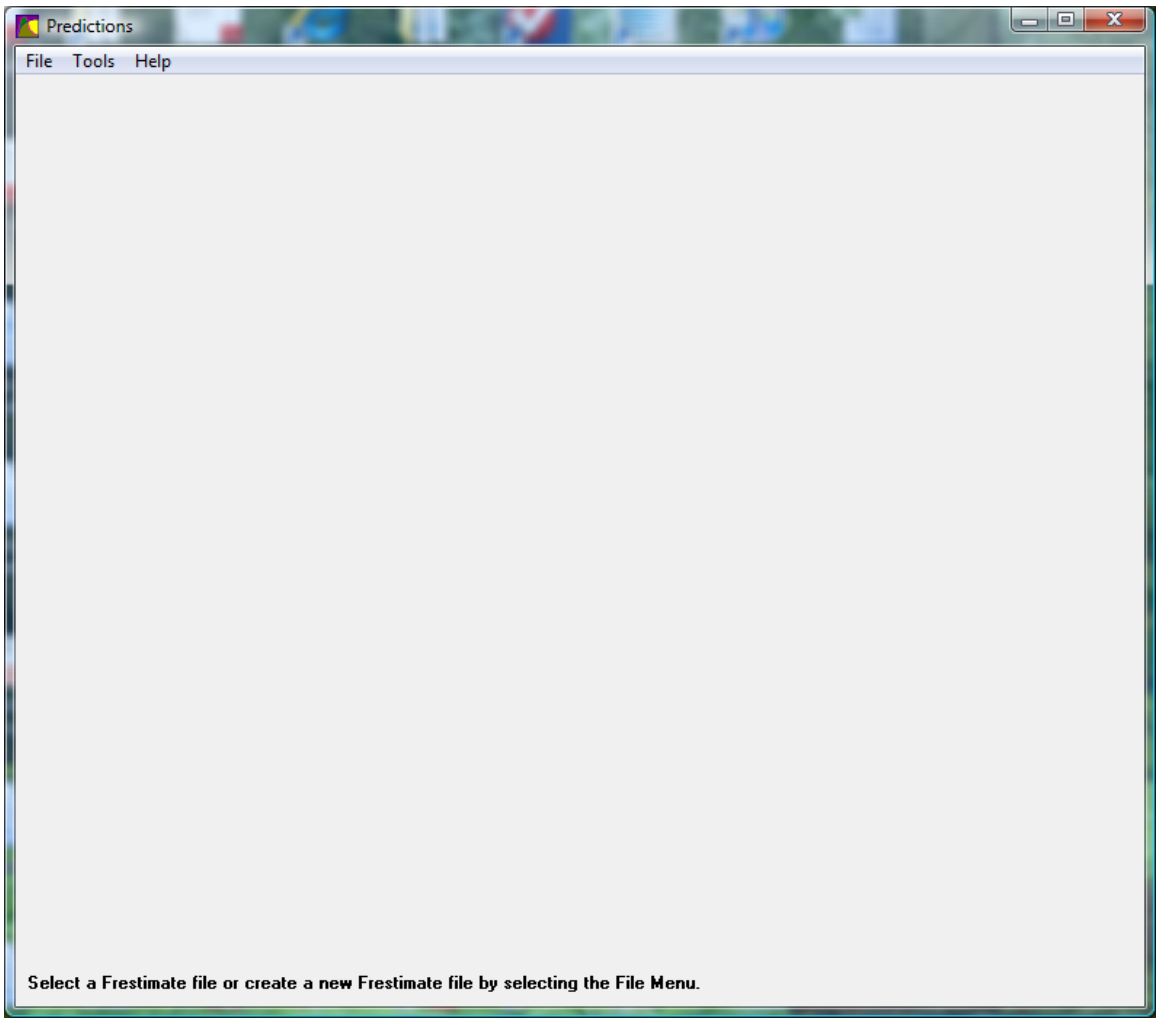
The application type is also used to select a default growth rate and the growth rate confidence bounds.

When you select the application type, the Rome Laboratory model uses that application to determine the “A Factor” which is a baseline defect density.

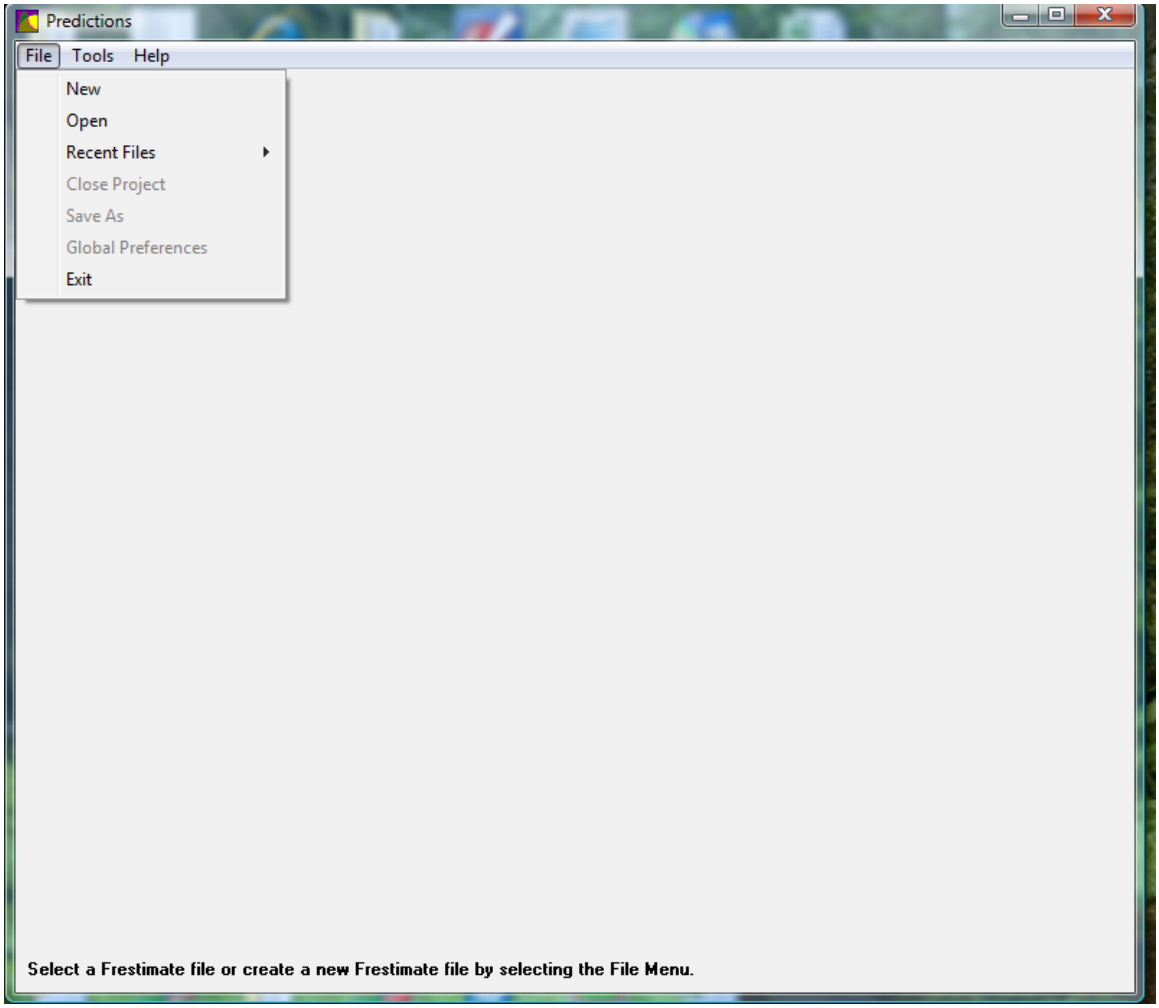
**Language**
Language is an input because it is used to convert the predicted defects per KSLOC of assembler to whatever language is applicable for your project. If you leave this as assembler and your language is C++ the predicted defects could be off by a measurable amount as Assembler is denser then C++. The language field and the Code Expansion field are related. When you change the language field, you change the default for the code expansion field. You can override this default, however.

## 2.0 File Menu

When you first launch Frestimate, there are no open project files. You will see the below screen.




The first thing that you need to do is open an existing project file or create a new project. Select the File Menu.



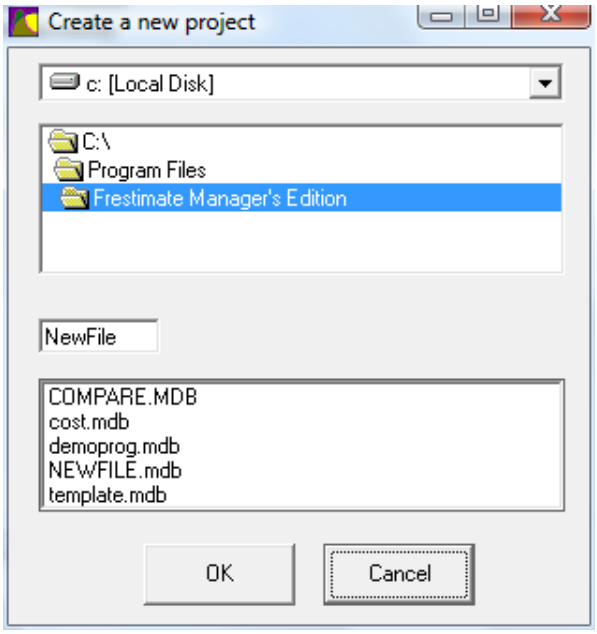

The file menu contains the project file related functions. You must have a project open to perform a software reliability prediction.

- New - Creates a new project file and then opens that file
- Open - Opens an existing project file
- Recent files – Allows you to select from a list of the 4 most recent project files opened
- Close - Closes the project file you are currently editing.
- Save As - Allows you to save this project to another name and then opens that file.
- Global Preferences- allows you to toggle the units of measure for time (i.e. hours, millions of hours or billions of hours).

- Exit - Exits FREstimate.

*2.1 New File*

The new file option allows you to create a new prediction file. One file is created for both the prediction and estimation models. You can create this file on any drive and in any folder**.**




**To create a project:**

1. Click on “File” from the main menu with your mouse and then click on “New”. The “New” function is not enabled when a project file is already open.
2. Decide which drive and which folder you want your new project to reside on.
3. Enter the name of the new file. Do NOT include the extension “.mdb”.
4. Click on “OK” to create the new project, or click on “CANCEL” to cancel the creation of the new project
5. If that selected project name already exists you will not be able to create it.
6. After the project is successfully created, you may open it and do all other “File” operations on it.

Name your project accordingly so that everyone using the software will put the correct information in the correct project. One good practice is to create a new project when going to a new release, keeping all of the closed features/corrective actions in one database and creating a new project with the features/corrective actions that are still active in a new database. If you have software products at your organization that are not related in any way then you should also create separate projects for each product.

Projects are database files that contain all of the information needed for predicting and estimating reliability. How you define projects is up to you. Generally a project is created for every release of a particular piece of software. You may also want to create one project for just the severe defects that are detected during testing.

Another good practice is to create a new database when there are allot of records in the current project. The more records there are, the slower it will get to manipulate that database. If you do create a new database for this reason, make sure that you group the features/corrective actions so that the older/obsolete ones are in one file and the newer records in another file.

*2.2 Open File*

You may open only one file at a time. You can select from any drive on your computer. Only valid FREstimate files will be displayed. If you open a file that is not valid, you will be notified appropriately.

Opening a project will open the database file selected. When a project is not open, only the New, Open and Exit commands are enabled. Once a project is open then the other menu items will be appropriately enabled.

Version 3.7 will upgrade FREstimate files created from version 3.6, 3.5, 3.4 and 3.3. However, it will not upgrade FREstimate files that were created from versions earlier than 3.3.

NOTE: The Full-scale and Shortcut models changed dramatically between version 3.6 and any earlier version of FREstimate. Even if FREstimate upgrades your files to version 3.6, you do need complete the Full-scale and Short-cut surveys with version 3.6.

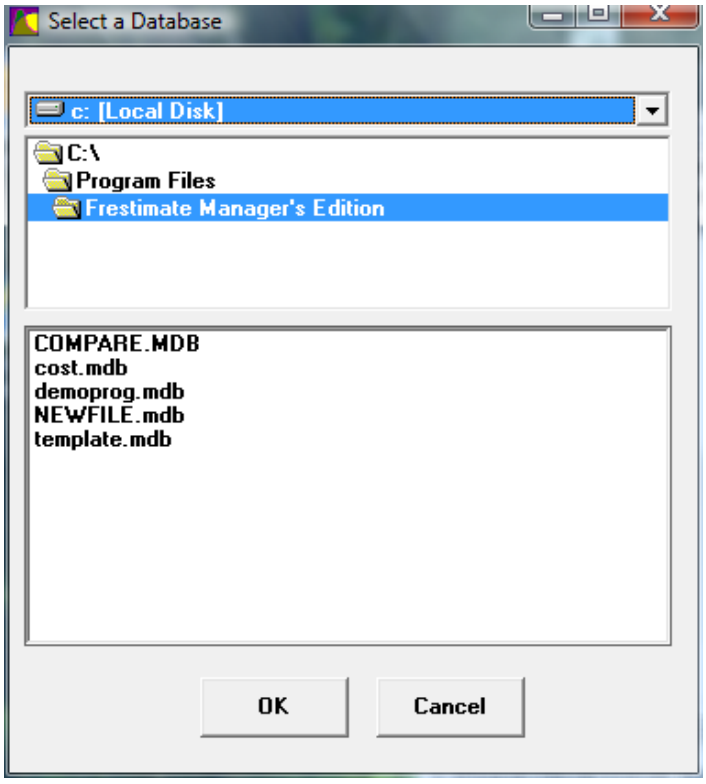
**To open a project:**

1. Click on “File” from the main menu with your mouse and then click on “Open”.
.
2. Select the file you would like to open using the scroll window or the up and down arrow keys.

3. Click on “OK” to open the project, or click on “CANCEL” to cancel the open.

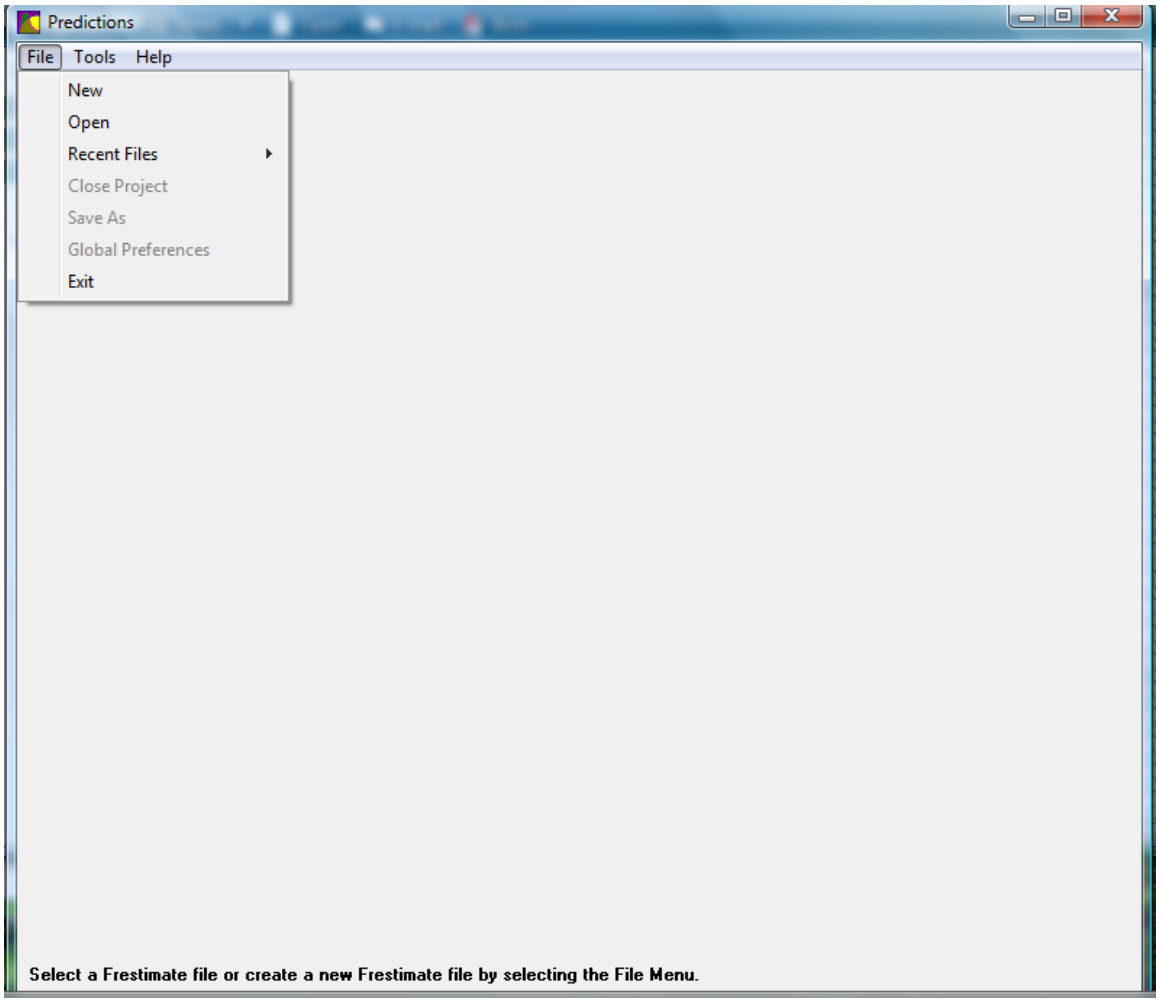
4. Once a file is opened, you will be able to select from the Prediction, Estimation and Fielded MTTF metrics menus.

The “File Open” function is disabled when a project file is already open.

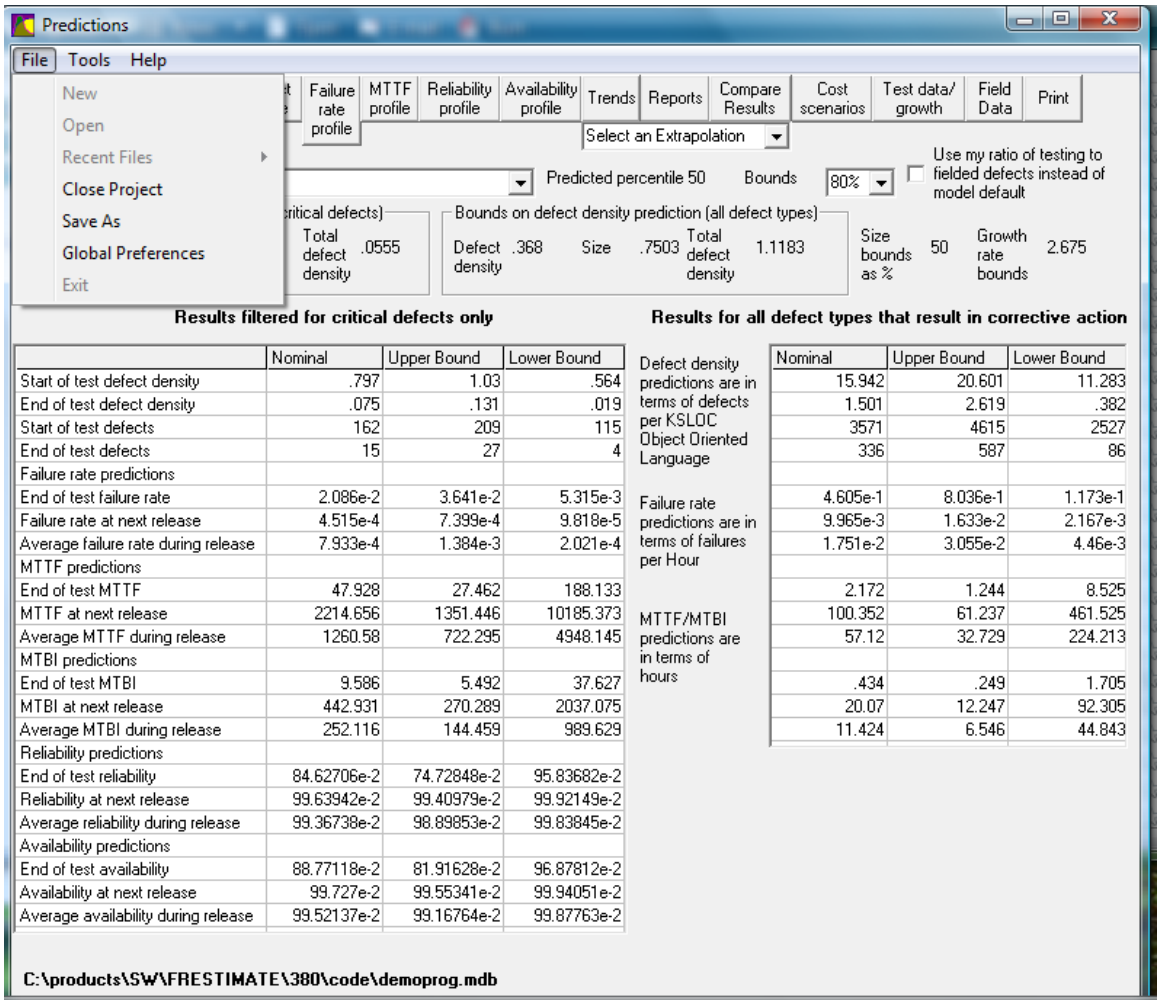


*2.3 Recent Files*

You can also open a file by using the Recent Files option. When you select "Recent Files" from the "File" menu you will see the 4 most recent files that you have opened/created. The first time that you use Frestimate you won't see any file names in this list. "Recent Files" is only enabled when there is no project file open.




Once you have a project file open, the below file menu is displayed. You can see that now the other menu options are enabled. Regardless of whether you use the "New", "Open" or "Recent Files" option to open a project file, the "File" Menu will now look like the below.

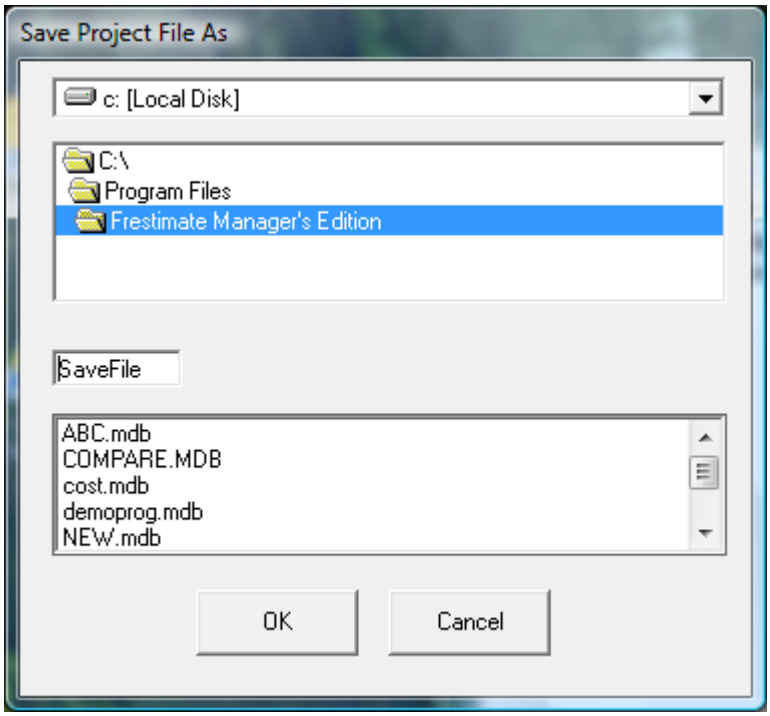

Predictions
File
Tools
Help
New
Open
Recent Files
Close Project
Save As
Global Preferences
Exit
Failure rate profile
MTTF profile
Reliability profile
Availability profile
Trends
Reports
Compare Results
Cost scenarios
Test data/ growth
Field Data
Print
Select an Extrapolation
Predicted percentile 50
Bounds
80%
Use my ratio of testing to fielded defects instead of model default
ritical defects)
Total defect density .0555
Bounds on defect density prediction (all defect types)
Defect density .368
Size .7503
Total defect density 1.1183
Size bounds as % 50
Growth rate bounds 2.675
Results filtered for critical defects only
Results for all defect types that result in corrective action
Nominal
Upper Bound
Lower Bound
Start of test defect density .797 1.03 .564
End of test defect density .075 .131 .019
Start of test defects 162 209 115
End of test defects 15 27 4
Failure rate predictions
End of test failure rate 2.086e-2 3.641e-2 5.315e-3
Failure rate at next release 4.515e-4 7.399e-4 9.818e-5
Average failure rate during release 7.933e-4 1.384e-3 2.021e-4
MTTF predictions
End of test MTTF 47.928 27.462 188.133
MTTF at next release 2214.656 1351.446 10185.373
Average MTTF during release 1260.58 722.295 4948.145
MTBI predictions
End of test MTBI 9.586 5.492 37.627
MTBI at next release 442.931 270.289 2037.075
Average MTBI during release 252.116 144.459 989.629
Reliability predictions
End of test reliability 84.62706e-2 74.72848e-2 95.83682e-2
Reliability at next release 99.63942e-2 99.40979e-2 99.92149e-2
Average reliability during release 99.36738e-2 98.89853e-2 99.83845e-2
Availability predictions
End of test availability 88.77118e-2 81.91628e-2 96.87812e-2
Availability at next release 99.727e-2 99.55341e-2 99.94051e-2
Average availability during release 99.52137e-2 99.16764e-2 99.87763e-2
Defect density predictions are in terms of defects per KSLOC Object Oriented Language
Failure rate predictions are in terms of failures per Hour
MTTF/MTBI predictions are in terms of hours
Nominal
Upper Bound
Lower Bound
15.942 20.601 11.283
1.501 2.619 .382
3571 4615 2527
336 587 86
4.605e-1 8.036e-1 1.173e-1
9.965e-3 1.633e-2 2.167e-3
1.751e-2 3.055e-2 4.46e-3
2.172 1.244 8.525
100.352 61.237 461.525
57.12 32.729 224.213
.434 .249 1.705
20.07 12.247 92.305
11.424 6.546 44.843
C:\products\SW\FRESTIMATE\380\code\demoprog.mdb

*2.4 Close File*

When you close a file it will automatically be saved. Demonstration software packages will not save the file that you worked on.

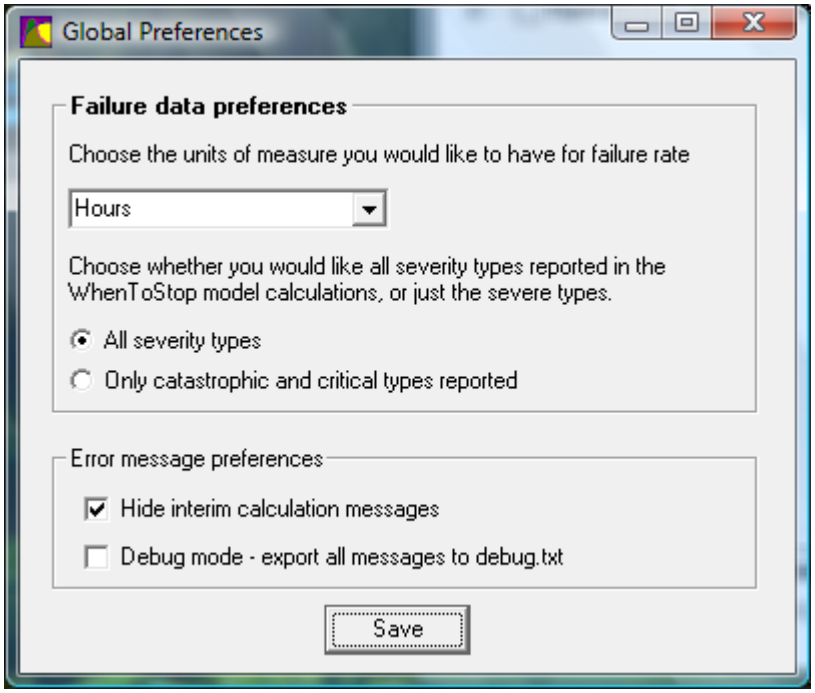
*2.5 "Save As" File*

This function allows you to save this file as another file. This is useful when you have two predictions that have similar data inputs. After this function is complete, the project that you create is now in loaded and not the original project. In the below example, SaveFile will be in memory after the "OK" button is pressed assuming that this project name does not already exist.



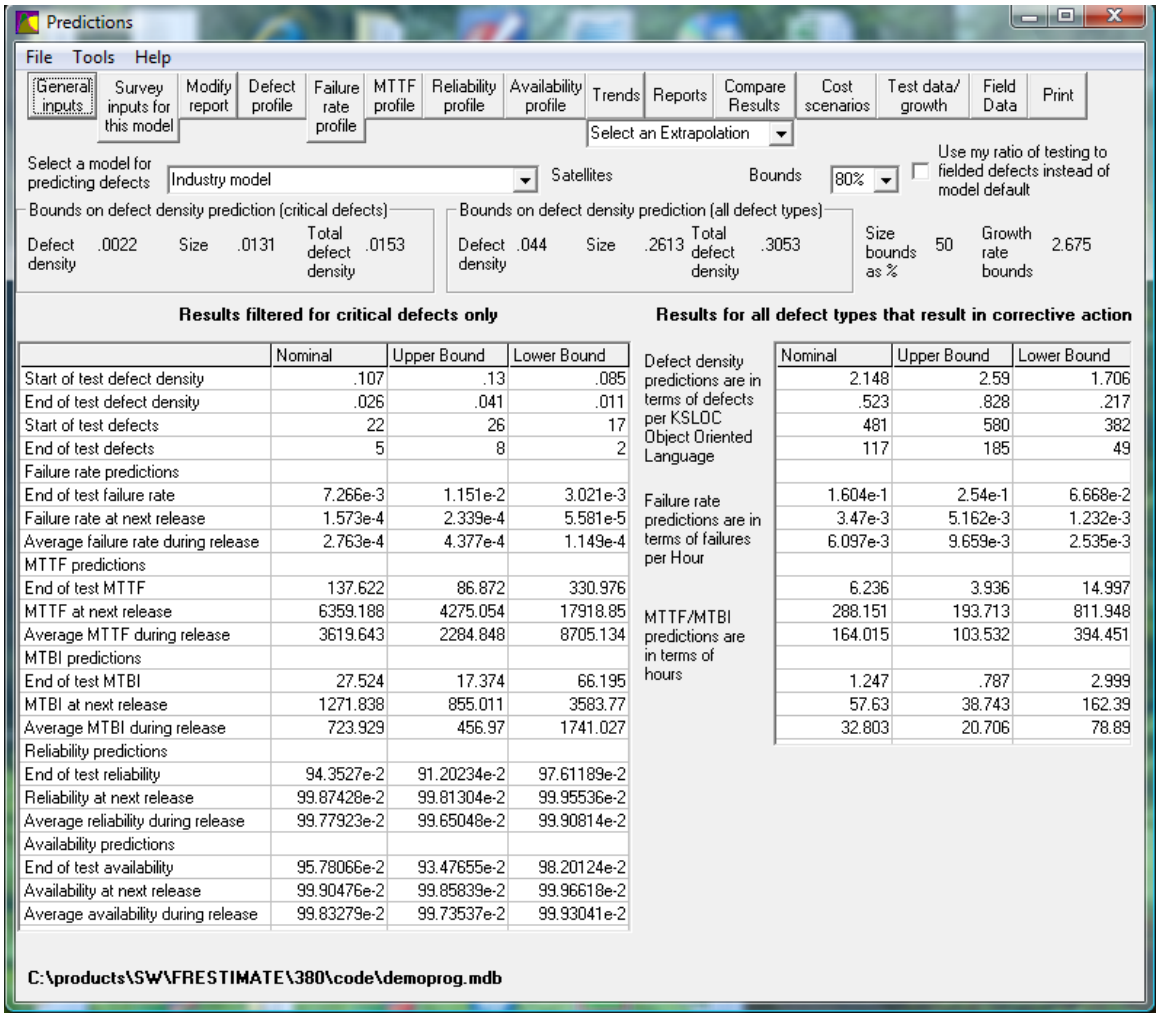
*2.6 Global preferences*

This dialog allows you to select which unit of measure for time will be displayed in both the prediction and estimation results. Your choices are hours, millions of hours and billions of hours. Failure rate will be displayed in terms of whichever of these you select from this dialog. This preference is a global preference in that all files that you create or modify will use this preference. You can change this preference anytime that a project file is open. After the change, all future projects will be automatically converted to this unit of measure. This is a global preference because it is usually not changed often within an organization. If you have the WhenToStop estimation module, a message will be displayed indicating that the estimation computations are being updated**.**



## 3.0 Prediction Results

The prediction results show predictions before code is even written. See the Prediction user's manual for more information on section 3.0.



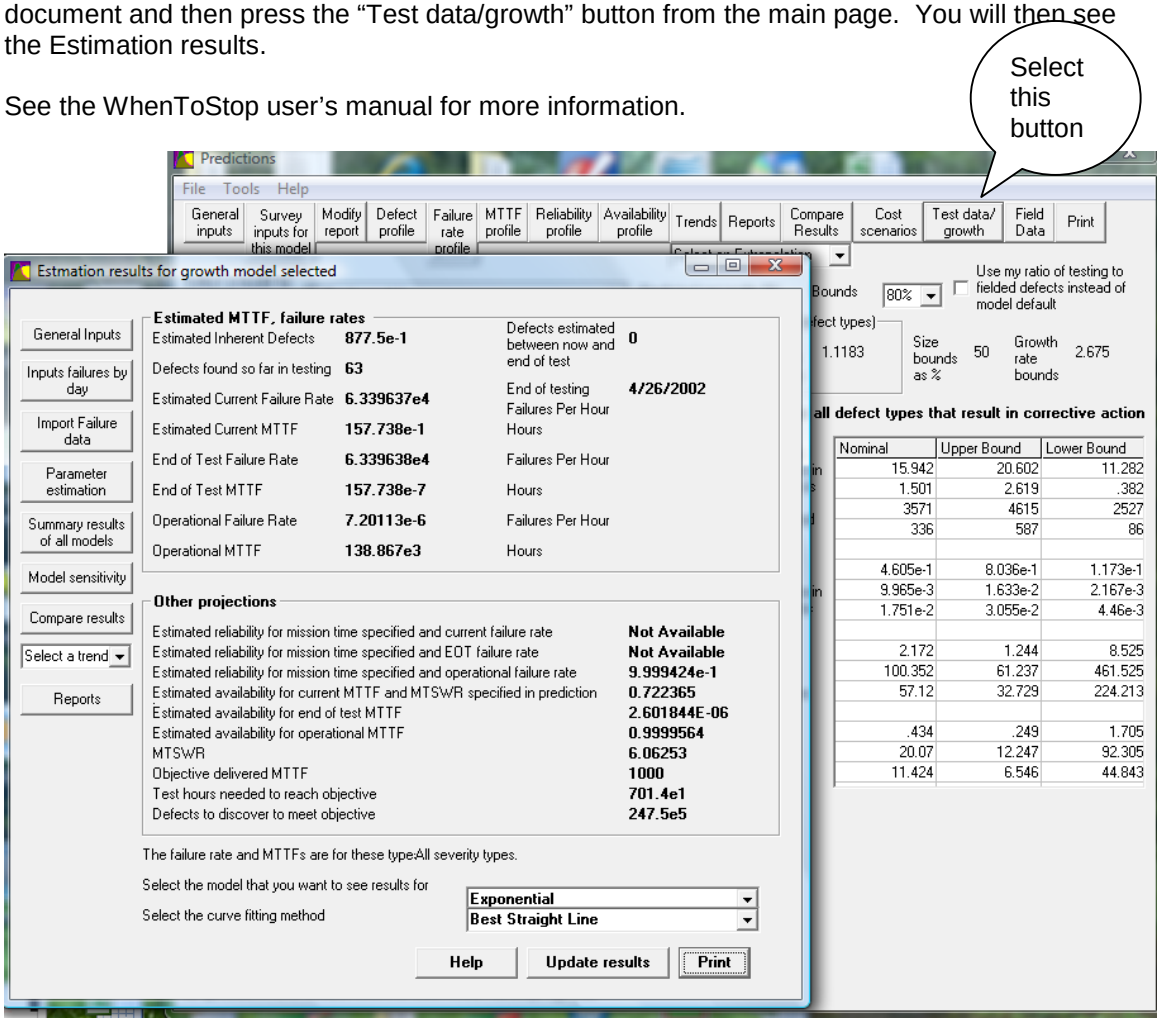

| | Nominal | Upper Bound | Lower Bound |
|---|---|---|---|
| Start of test defect density | .107 | .13 | .085 |
| End of test defect density | .026 | .041 | .011 |
| Start of test defects | 22 | 26 | 17 |
| End of test defects | 5 | 8 | 2 |
| Failure rate predictions | | | |
| End of test failure rate | 7.266e-3 | 1.151e-2 | 3.021e-3 |
| Failure rate at next release | 1.573e-4 | 2.339e-4 | 5.581e-5 |
| Average failure rate during release | 2.763e-4 | 4.377e-4 | 1.149e-4 |
| MTTF predictions | | | |
| End of test MTTF | 137.622 | 86.872 | 330.976 |
| MTTF at next release | 6359.188 | 4275.054 | 17918.85 |
| Average MTTF during release | 3619.643 | 2284.848 | 8705.134 |
| MTBI predictions | | | |
| End of test MTBI | 27.524 | 17.374 | 66.195 |
| MTBI at next release | 1271.838 | 855.011 | 3583.77 |
| Average MTBI during release | 723.929 | 456.97 | 1741.027 |
| Reliability predictions | | | |
| End of test reliability | 94.3527e-2 | 91.20234e-2 | 97.61189e-2 |
| Reliability at next release | 99.87428e-2 | 99.81304e-2 | 99.95536e-2 |
| Average reliability during release | 99.77923e-2 | 99.65048e-2 | 99.90814e-2 |
| Availability predictions | | | |
| End of test availability | 95.78066e-2 | 93.47655e-2 | 98.20124e-2 |
| Availability at next release | 99.90476e-2 | 99.85839e-2 | 99.96618e-2 |
| Average availability during release | 99.83279e-2 | 99.73537e-2 | 99.93041e-2 |

| Nominal | Upper Bound | Lower Bound |
|---|---|---|
| 2.148 | 2.59 | 1.706 |
| .523 | .828 | .217 |
| 481 | 580 | 382 |
| 117 | 185 | 49 |
| | | |
| 1.604e-1 | 2.54e-1 | 6.668e-2 |
| 3.47e-3 | 5.162e-3 | 1.232e-3 |
| 6.097e-3 | 9.659e-3 | 2.535e-3 |
| | | |
| 6.236 | 3.936 | 14.997 |
| 288.151 | 193.713 | 811.948 |
| 164.015 | 103.532 | 394.451 |
| | | |
| 1.247 | .787 | 2.999 |
| 57.63 | 38.743 | 162.39 |
| 32.803 | 20.706 | 78.89 |

## 4.0 Estimation Results

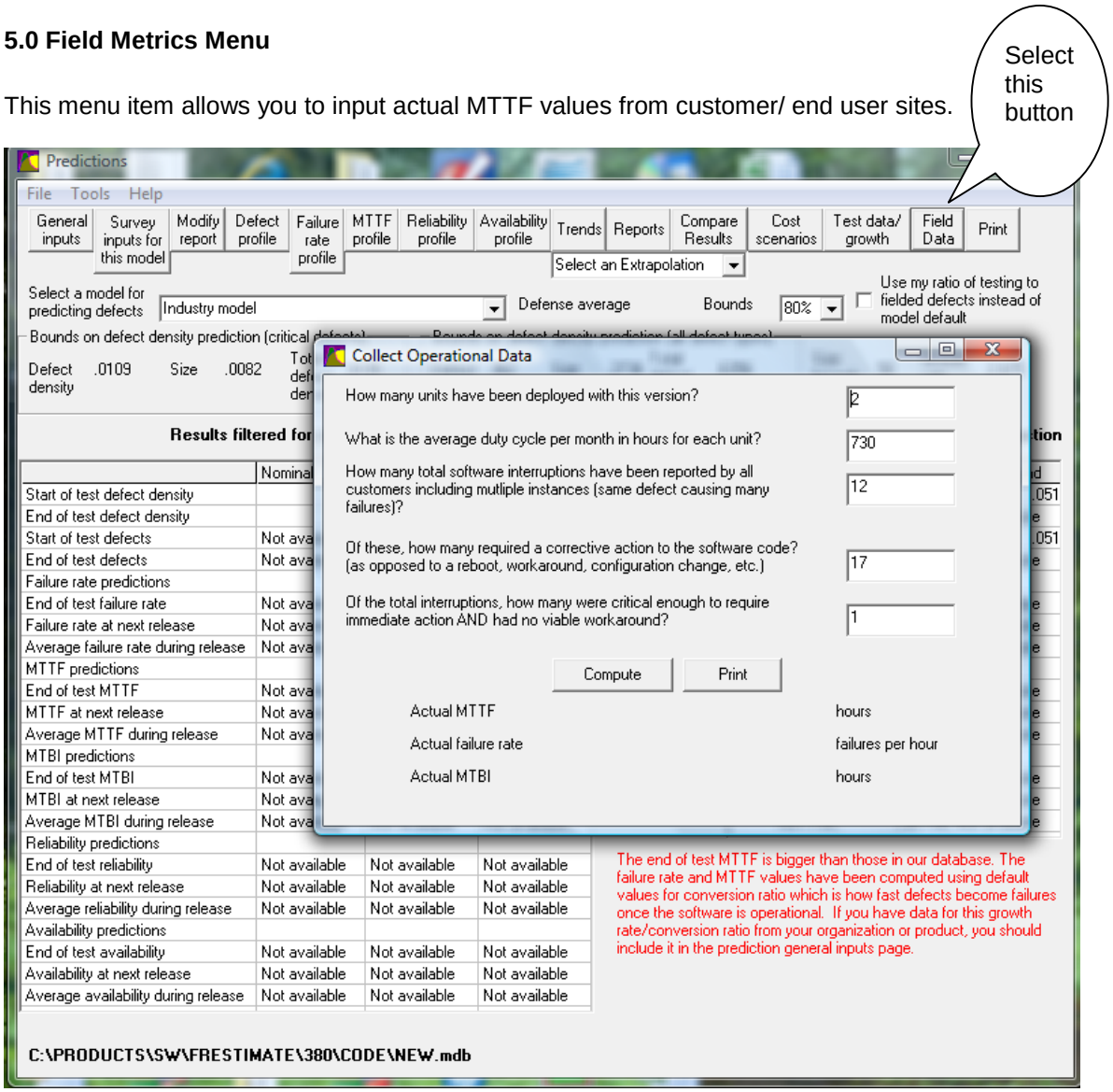
The Systems Testing Estimation menu is enabled in the Frestimate Manager's edition. These functions do estimations based on failure data collected during testing. To access the Estimation Results you must have the Manager's Edition installed. Open a file as per section 1 of this document and then press the "Test data/growth" button from the main page. You will then see the Estimation results.

See the WhenToStop user's manual for more information.



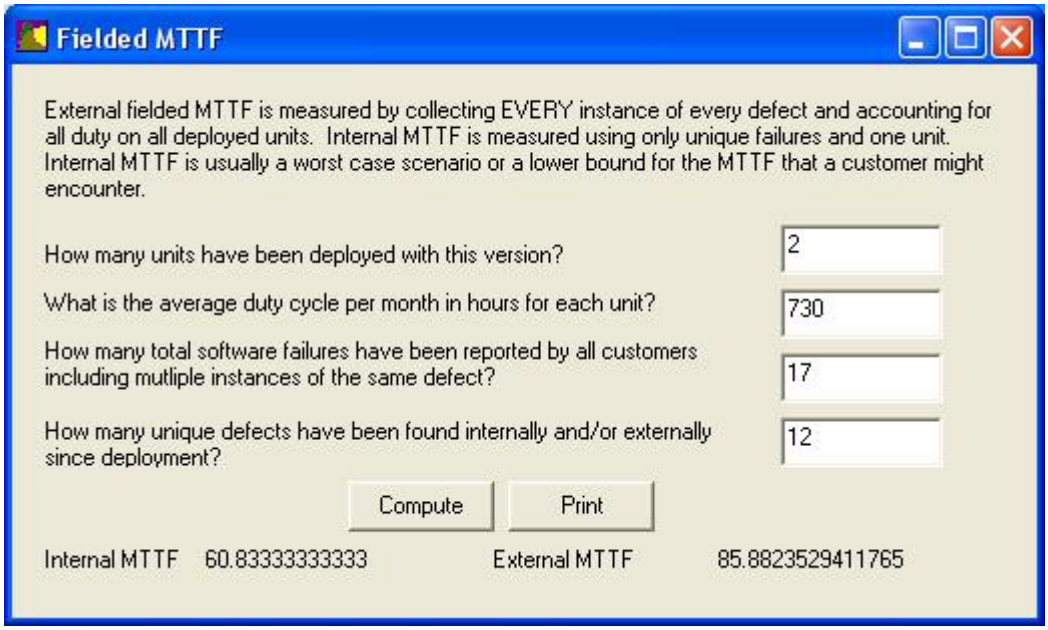
## 5.0 Field Metrics Menu

This menu item allows you to input actual MTTF values from customer/ end user sites.



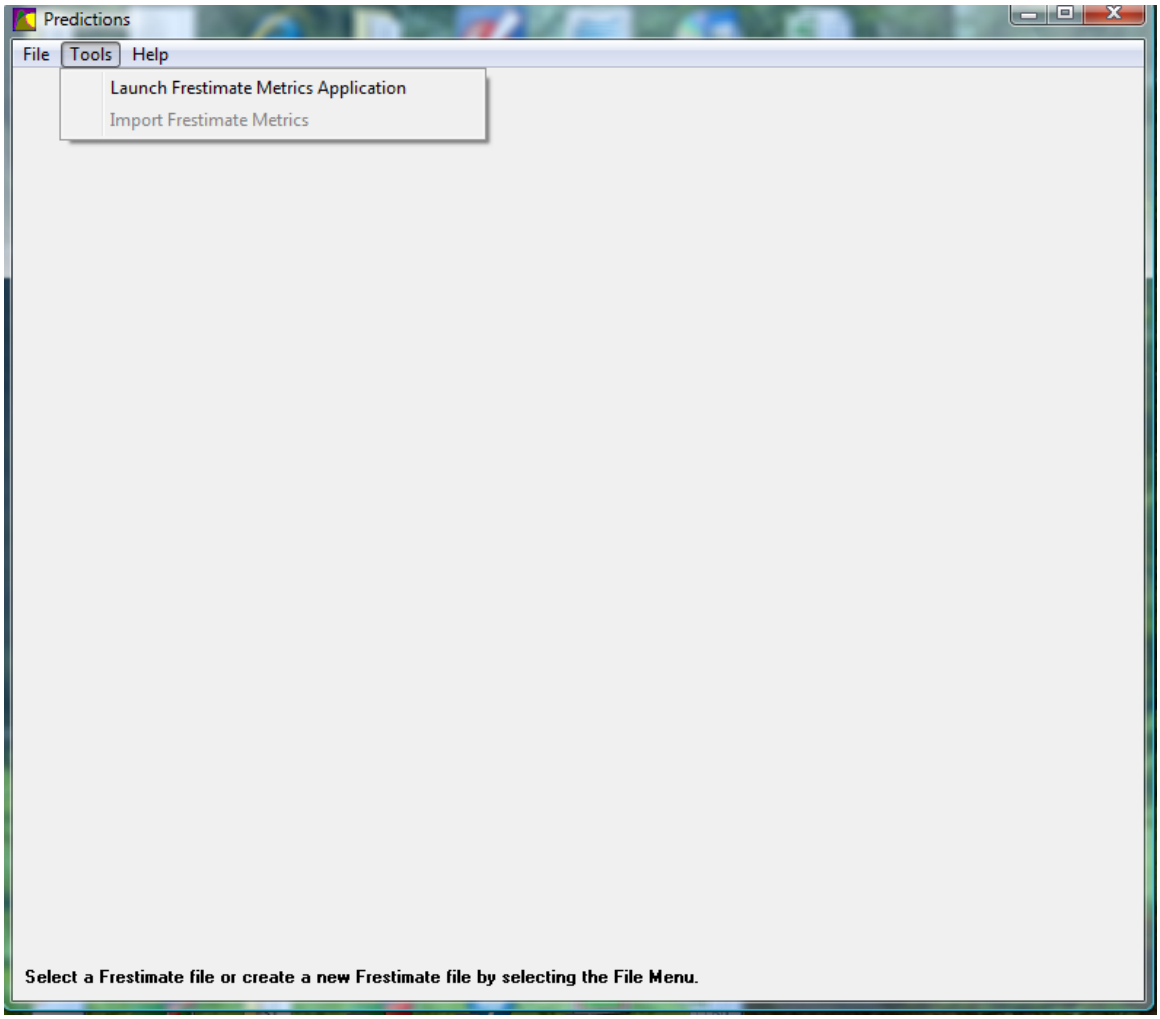

Internal Fielded MTTF is simply the total number of hours in operation on one average system/customer site divided by the total number of unique failures. This is almost always the worst case scenario for the fielded MTTF as it is the MTTF if all failures were to be visible to all customers. However, it also presumes that a particular failure is observed only once.

External Fielded MTTF is simply the total number of hours in operation by any customer/delivered system divided by the total number of failures regardless of whether they are unique or not. So, if the same defect causes the same failure at 10 systems then 10 failures are counted. The external MTTF also takes into consideration the possibility that the same failure may be observed over and over at the same customer site.


Fielded MTTF
External fielded MTTF is measured by collecting EVERY instance of every defect and accounting for all duty on all deployed units. Internal MTTF is measured using only unique failures and one unit. Internal MTTF is usually a worst case scenario or a lower bound for the MTTF that a customer might encounter.
How many units have been deployed with this version?
2
What is the average duty cycle per month in hours for each unit?
730
How many total software failures have been reported by all customers including mutliple instances of the same defect?
17
How many unique defects have been found internally and/or externally since deployment?
12
Compute
Print
Internal MTTF 60.83333333333
External MTTF 85.8823529411765

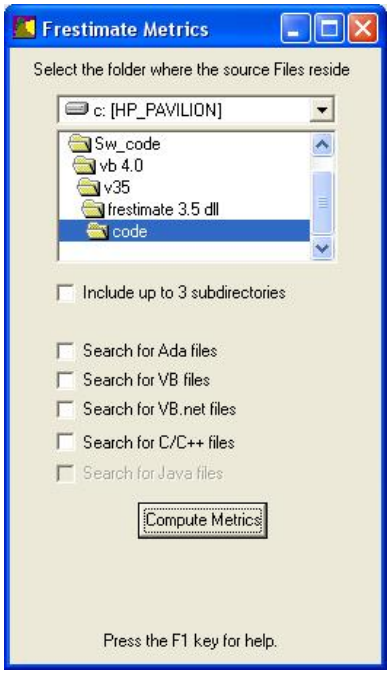
## 6.0 Tools Menu

The tools menu is new as of version 3.5. This menu executes the FREstimate Metrics software package.



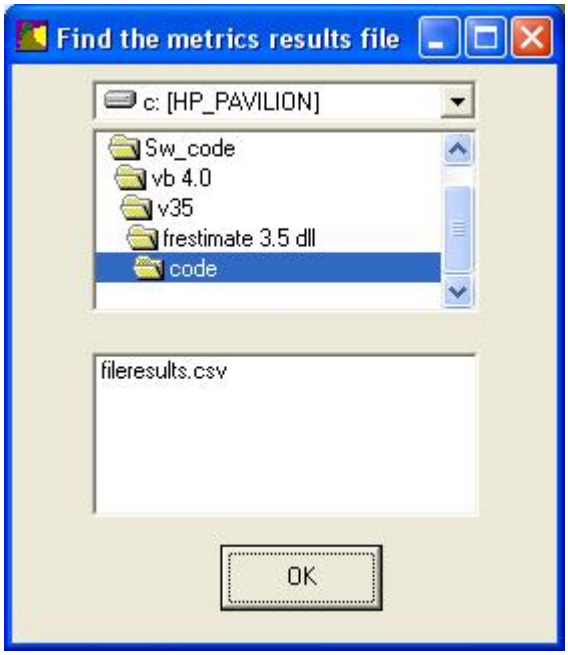

### *6.1 Launch FREstimate Metrics*

If you have the FREstimateMetrics.exe application installed in the same folder as the FREstimate software, this tool menu item will be enabled. When you select this menu item the following page is displayed. The FREstimate Metrics package has a user's manual and help file for more detailed instructions on how to use the metrics component. Refer to these documents for details. Note that only the language modules that you have purchased will be enabled in the below page. (The Java language module will be implemented at a future date)


Frestimate Metrics
Select the folder where the source Files reside
c: [HP_PAVILION]
Sw_code
vb 4.0
v35
frestimate 3.5 dll
code
Include up to 3 subdirectories
Search for Ada files
Search for VB files
Search for VB.net files
Search for C/C++ files
Search for Java files
Compute Metrics
Press the F1 key for help.

*6.2 Import FREstimate Metrics*

If you have a project file open, you can import the results of the FREstimateMetrics application directly into the project file. The below page is displayed when you select this menu item. You simply select the results file from the FREstimateMetrics application and then the size and complexity are imported into the FREstimate project file. You will be able to see the imported results by going to the "All general inputs page".



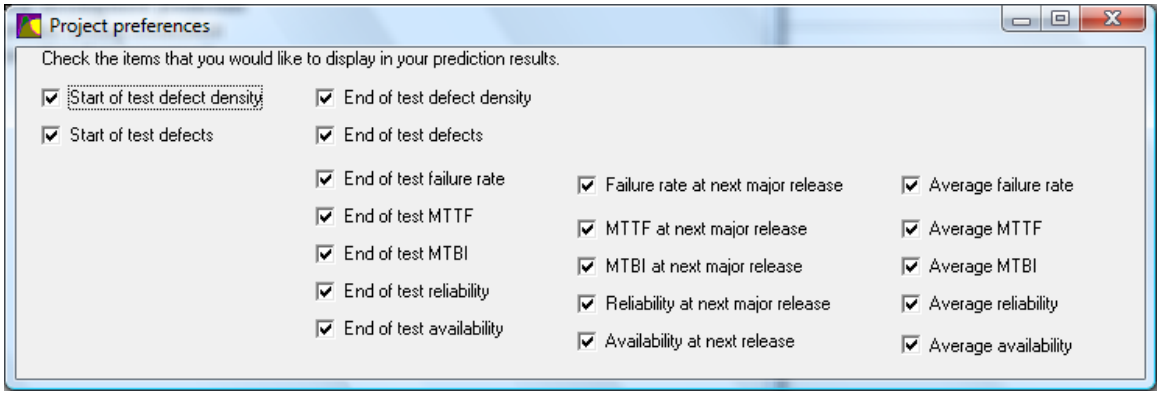

The size metrics import will convert all language types to assembler and multiply the stand code expansion ratios to this for a resulting KSLOC size. The standard code expansion ratios are:

| | |
|---|---|
| VB and VB .net | 6 |
| C | 3 |
| C++ | 6 |
| Ada | 6 |

Since the tool cannot determine what percentage of the code is effective, it will also set the EKSLOC and critical EKSLOC to the total size imported from the metrics tool. Please make sure that you modified the EKSLOC field so that only the new KSLOC is counted. Alternatively, you can perform the FREstimate metrics on only the effective KSLOC prior to importing the size and complexity. The imported complexity is currently used only by the Rome Labs module.

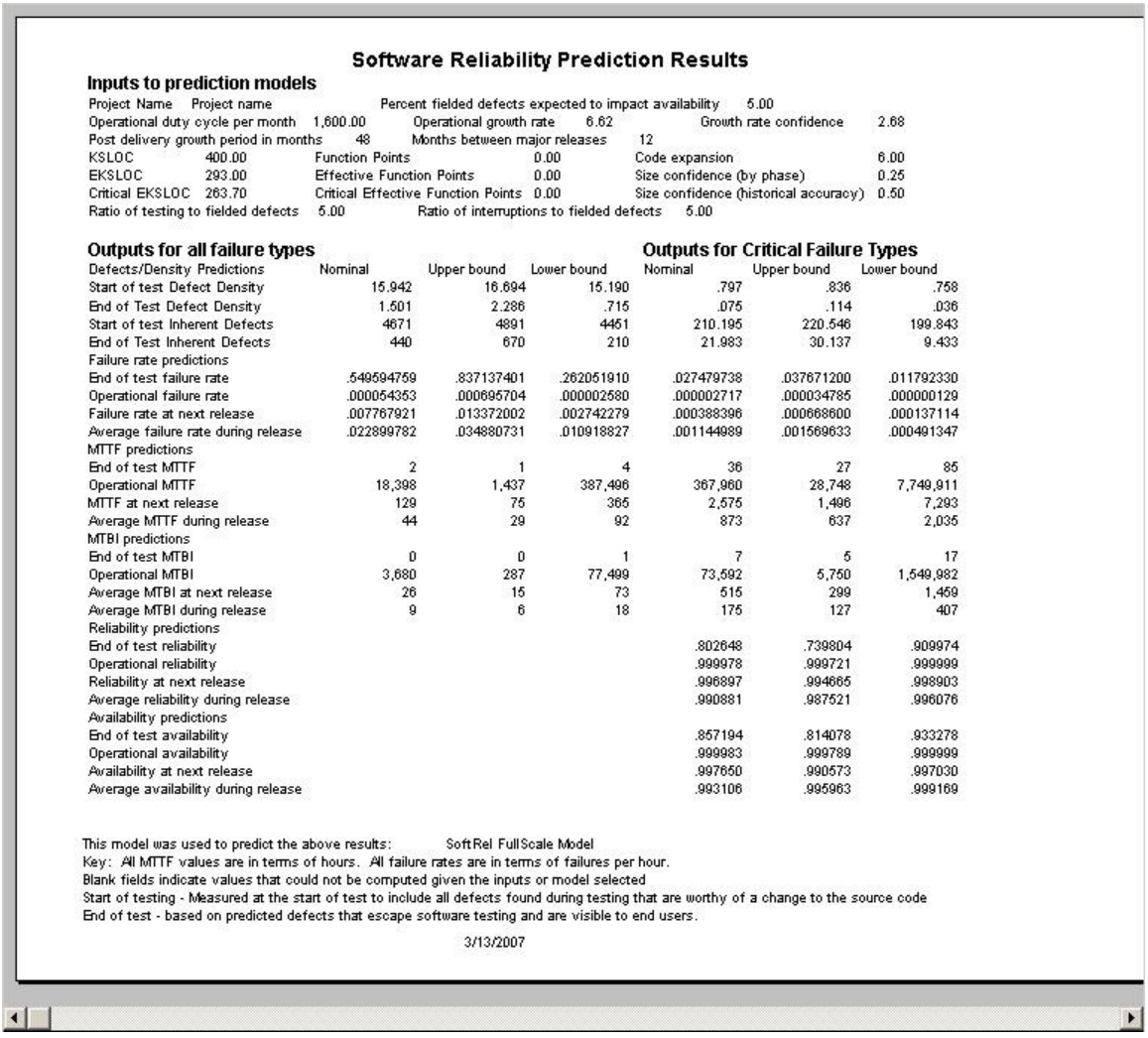
*2.7 Modify Report*

Frestimate computes many different kinds of results as shown in Figure 1. For a particular project some of these results may be more relevant than others. This page allows you to select which metrics are displayed in the results page.


Project preferences
Check the items that you would like to display in your prediction results.
Start of test defect density
Start of test defects
End of test defect density
End of test defects
End of test failure rate
End of test MTTF
End of test MTBI
End of test reliability
End of test availability
Failure rate at next major release
MTTF at next major release
MTBI at next major release
Reliability at next major release
Availability at next release
Average failure rate
Average MTTF
Average MTBI
Average reliability
Average availability

### *2.8 Print Prediction Project Summary*

This will print the results for each prediction model that you have enabled. You can print to the screen, to a file or to a printer. Printing is not enabled for the demonstration program. If you want to print the inputs for each of the models you can do so by going to that input screen and printing. Graphs can be printed as well as exported and saved to the clipboard.



**Software Reliability Prediction Results**

**Inputs to prediction models**

| | | | | | |
|---|---|---|---|---|---|
| Project Name | Project name | Percent fielded defects expected to impact availability | 5.00 | | |
| Operational duty cycle per month | 1,600.00 | Operational growth rate | 6.62 | Growth rate confidence | 2.68 |
| Post delivery growth period in months | 48 | Months between major releases | 12 | | |
| KSLOC | 400.00 | Function Points | 0.00 | Code expansion | 6.00 |
| EKSLOC | 293.00 | Effective Function Points | 0.00 | Size confidence (by phase) | 0.25 |
| Critical EKSLOC | 263.70 | Critical Effective Function Points | 0.00 | Size confidence (historical accuracy) | 0.50 |
| Ratio of testing to fielded defects | 5.00 | Ratio of interruptions to fielded defects | 5.00 | | |

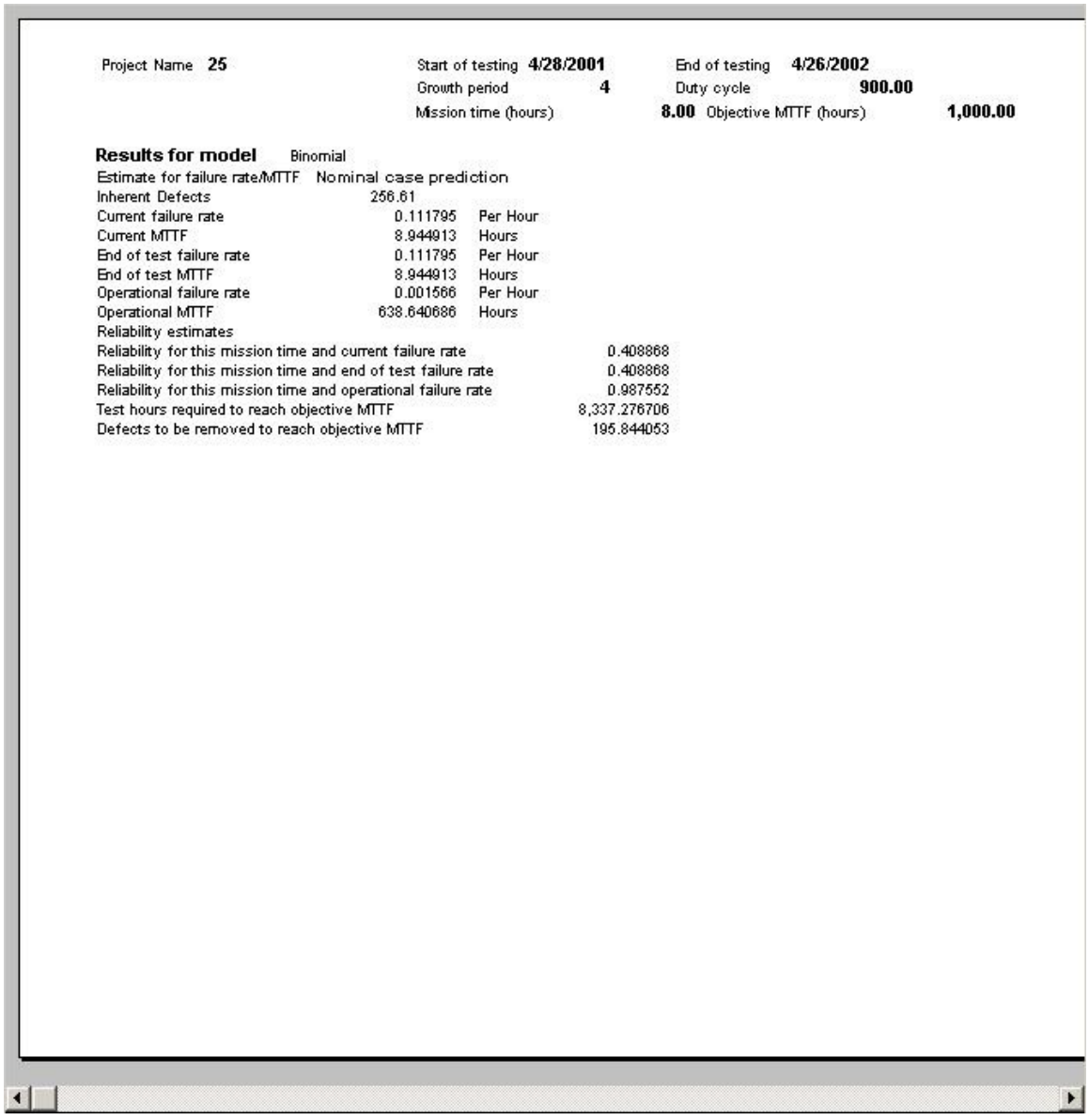
| | Outputs for all failure types | | | Outputs for Critical Failure Types | | |
|---|---|---|---|---|---|---|
| Defects/Density Predictions | Nominal | Upper bound | Lower bound | Nominal | Upper bound | Lower bound |
| Start of test Defect Density | 15.942 | 16.694 | 15.190 | .797 | .836 | .758 |
| End of Test Defect Density | 1.501 | 2.286 | .715 | .075 | .114 | .036 |
| Start of test Inherent Defects | 4671 | 4891 | 4451 | 210.195 | 220.546 | 199.843 |
| End of Test Inherent Defects | 440 | 670 | 210 | 21.983 | 30.137 | 9.433 |
| Failure rate predictions | | | | | | |
| End of test failure rate | .549594759 | .837137401 | .262051910 | .027479738 | .037671200 | .011792330 |
| Operational failure rate | .000054353 | .000695704 | .000002580 | .000002717 | .000034785 | .000000129 |
| Failure rate at next release | .007767921 | .013372002 | .002742279 | .000388396 | .000668600 | .000137114 |
| Average failure rate during release | .022899782 | .034880731 | .010918827 | .001144989 | .001569633 | .000491347 |
| MTTF predictions | | | | | | |
| End of test MTTF | 2 | 1 | 4 | 36 | 27 | 85 |
| Operational MTTF | 18,398 | 1,437 | 387,496 | 367,960 | 28,748 | 7,749,911 |
| MTTF at next release | 129 | 75 | 365 | 2,575 | 1,496 | 7,293 |
| Average MTTF during release | 44 | 29 | 92 | 873 | 637 | 2,035 |
| MTBI predictions | | | | | | |
| End of test MTBI | 0 | 0 | 1 | 7 | 5 | 17 |
| Operational MTBI | 3,680 | 287 | 77,499 | 73,592 | 5,750 | 1,549,982 |
| Average MTBI at next release | 26 | 15 | 73 | 515 | 299 | 1,459 |
| Average MTBI during release | 9 | 6 | 18 | 175 | 127 | 407 |
| Reliability predictions | | | | | | |
| End of test reliability | | | | .802648 | .739804 | .909974 |
| Operational reliability | | | | .999978 | .999721 | .999999 |
| Reliability at next release | | | | .996897 | .994665 | .998903 |
| Average reliability during release | | | | .990881 | .987521 | .996076 |
| Availability predictions | | | | | | |
| End of test availability | | | | .857194 | .814078 | .933278 |
| Operational availability | | | | .999983 | .999789 | .999999 |
| Availability at next release | | | | .997650 | .990573 | .997030 |
| Average availability during release | | | | .993106 | .995963 | .999169 |

This model was used to predict the above results: SoftRel FullScale Model
Key: All MTTF values are in terms of hours. All failure rates are in terms of failures per hour.
Blank fields indicate values that could not be computed given the inputs or model selected
Start of testing - Measured at the start of test to include all defects found during testing that are worthy of a change to the source code
End of test - based on predicted defects that escape software testing and are visible to end users.

3/13/2007

## *2.9 Print Estimation Project Summary*

This will print the estimation results for the model selected in Systems Testing Estimation->Results. You can print to the screen, to a file or to a printer. Printing is not enabled for the demonstration program.



Project Name **25**

Start of testing **4/28/2001** End of testing **4/26/2002**
Growth period **4** Duty cycle **900.00**
Mission time (hours) **8.00** Objective MTTF (hours) **1,000.00**

**Results for model** Binomial

Estimate for failure rate/MTTF Nominal case prediction

| | | |
|---|---|---|
| Inherent Defects | 256.61 | |
| Current failure rate | 0.111795 | Per Hour |
| Current MTTF | 8.944913 | Hours |
| End of test failure rate | 0.111795 | Per Hour |
| End of test MTTF | 8.944913 | Hours |
| Operational failure rate | 0.001566 | Per Hour |
| Operational MTTF | 638.640686 | Hours |

Reliability estimates

| | |
|---|---|
| Reliability for this mission time and current failure rate | 0.408868 |
| Reliability for this mission time and end of test failure rate | 0.408868 |
| Reliability for this mission time and operational failure rate | 0.987552 |
| Test hours required to reach objective MTTF | 8,337.276706 |
| Defects to be removed to reach objective MTTF | 195.844053 |