Software Reliability

Allen P. Nikora
Autonomy and Control Section (345)
818-393-1104
Allen.P.Nikora@jpl.nasa.gov

Jet Propulsion Lab
Mail Code 125-209
4800 Oak Grove Dr.
Pasadena, CA 91101

Current Capabilities and Work

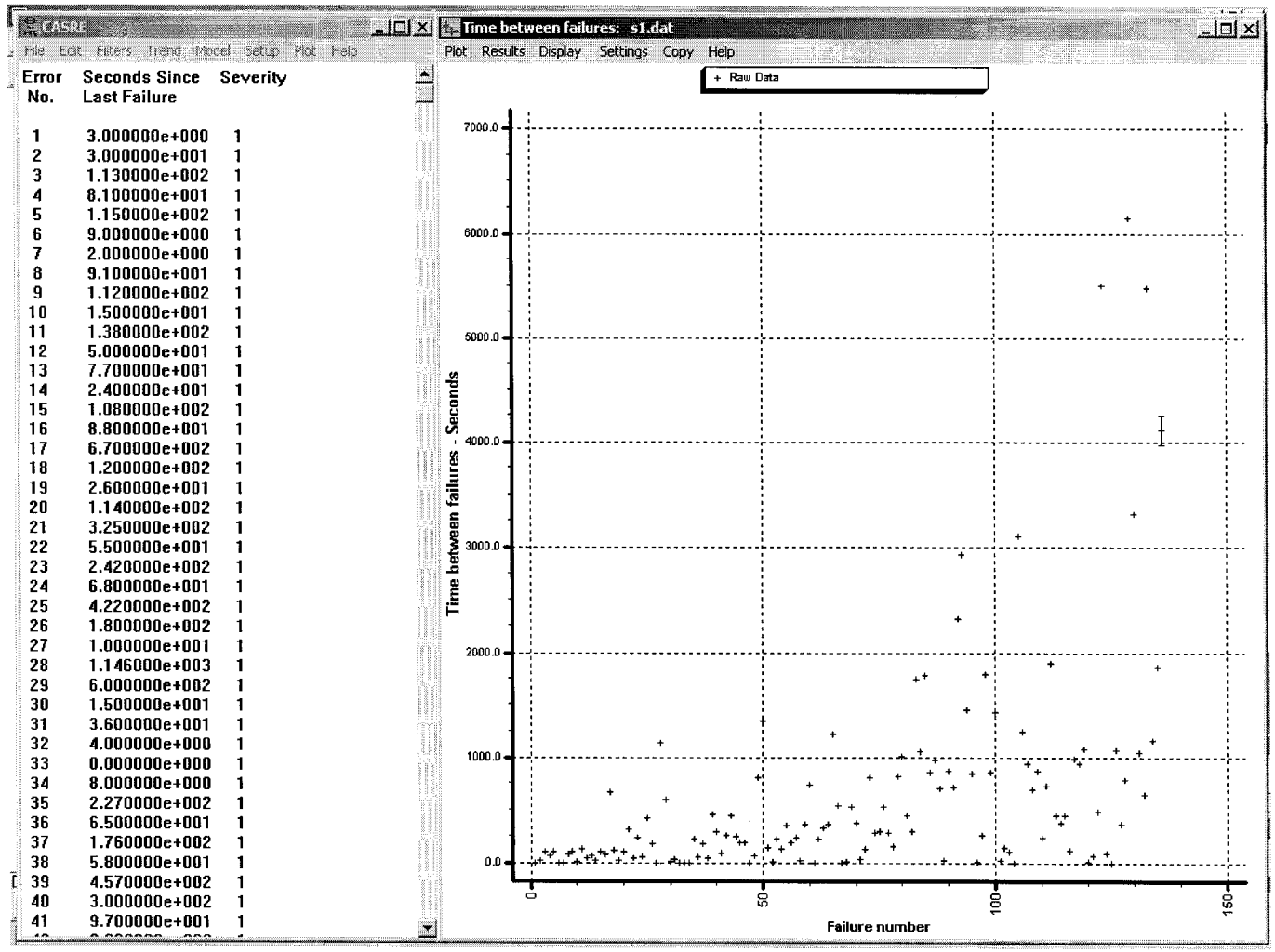
- Estimating/Forecasting software reliability during test
  - CASRE (Computer Aided Software Reliability Engineering) version 3.0 being readied for release
    - Statistical modeling of software reliability based on failure history of software during test
    - Determination of model applicability
    - Trend analysis of failure data – is application of software reliability models appropriate?
  - Development and maintenance funded since 1991 by AFOTEC.
- Estimating software fault content prior to test
  - On-going since 1996 – funded by NASA IV&V Facility, internal funding at JPL
    - Identify relationships between measurements of software structure and fault content.
    - Initial findings – measurements of structural evolution are related to number of faults inserted between subsequent SW versions.
    - Current work
      - Relate measurable characteristics of requirements changes and fault content of implemented changes
      - Relate structural change in different domains to different fault types
      - Develop practical measurement frameworks for software development efforts

State of Practice

- Estimation/Forecasting of reliability during test
  - Use failure history during test as input to statistical models
  - Estimate current reliability/failure intensity; predict future reliability/failure intensity
  - Improves insight into testing effort
    - What is current reliability?
    - Does the current reliability of the software meet the reliability requirement?
    - How much more effort (and $) is required to attain the required reliability?
    - What is the impact on reliability if sufficient testing resources are not available?
  - Disadvantages
    - Can only be used during test – measurements made too late during development to use in controlling design/specification quality.
    - System may not exhibit reliability growth during test
    - Can't be used to determine whether ultra-reliable systems (e.g., 10^{-9} failures per hour) meet requirements

Software Reliability Estimation with CASRE - Example

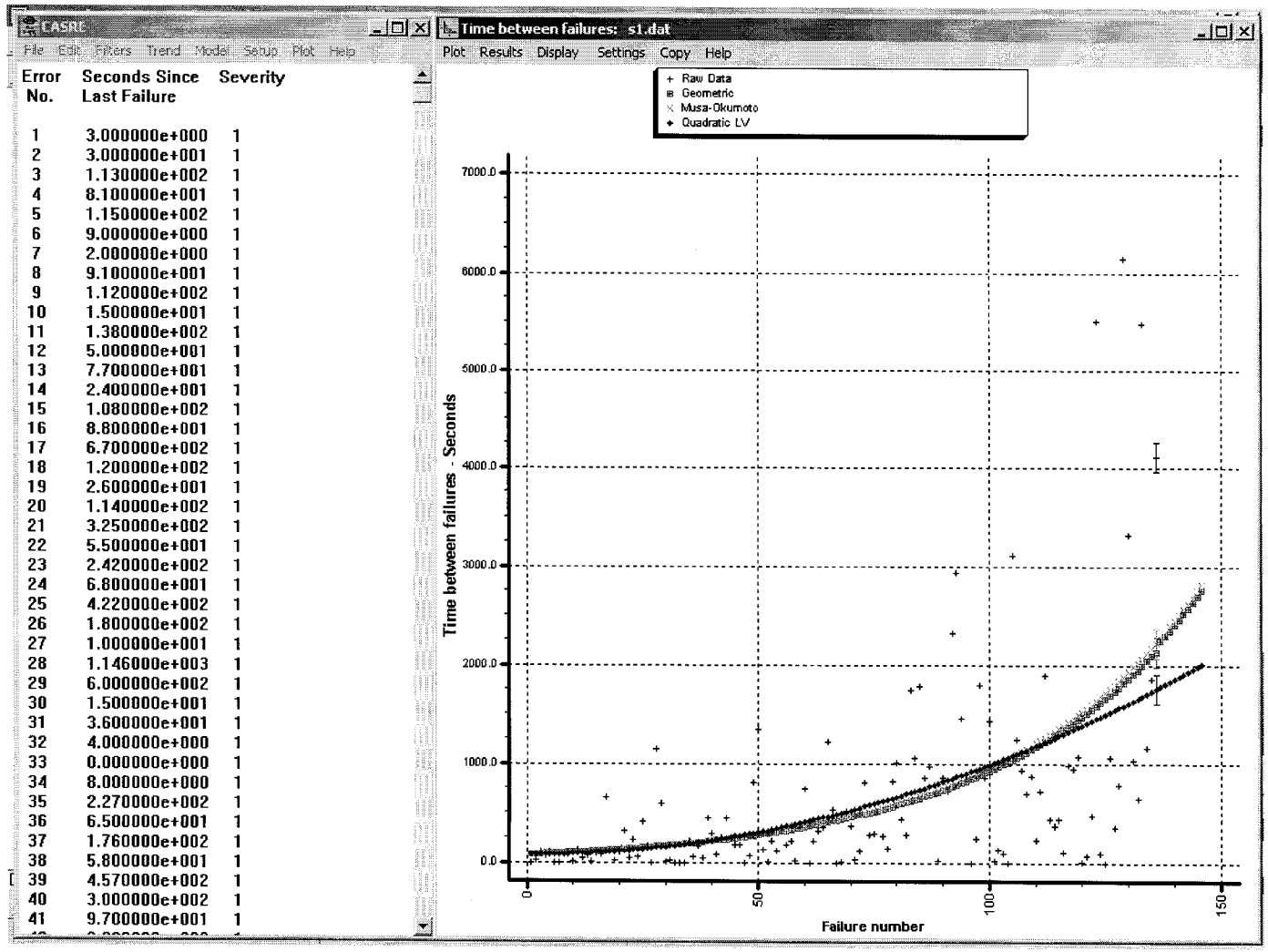
Failure History Data



Error No.	Seconds Since Last Failure	Severity
1	3.000000e+000	1
2	3.000000e+001	1
3	1.130000e+002	1
4	8.100000e+001	1
5	1.150000e+002	1
6	9.000000e+000	1
7	2.000000e+000	1
8	9.100000e+001	1
9	1.120000e+002	1
10	1.500000e+001	1
11	1.380000e+002	1
12	5.000000e+001	1
13	7.700000e+001	1
14	2.400000e+001	1
15	1.080000e+002	1
16	8.800000e+001	1
17	6.700000e+002	1
18	1.200000e+002	1
19	2.600000e+001	1
20	1.140000e+002	1
21	3.250000e+002	1
22	5.500000e+001	1
23	2.420000e+002	1
24	6.800000e+001	1
25	4.220000e+002	1
26	1.800000e+002	1
27	1.000000e+001	1
28	1.146000e+003	1
29	6.000000e+002	1
30	1.500000e+001	1
31	3.600000e+001	1
32	4.000000e+000	1
33	0.000000e+000	1
34	8.000000e+000	1
35	2.270000e+002	1
36	6.500000e+001	1
37	1.760000e+002	1
38	5.800000e+001	1
39	4.570000e+002	1
40	3.000000e+002	1
41	9.700000e+001	1

Software Reliability Estimation with CASRE - Example

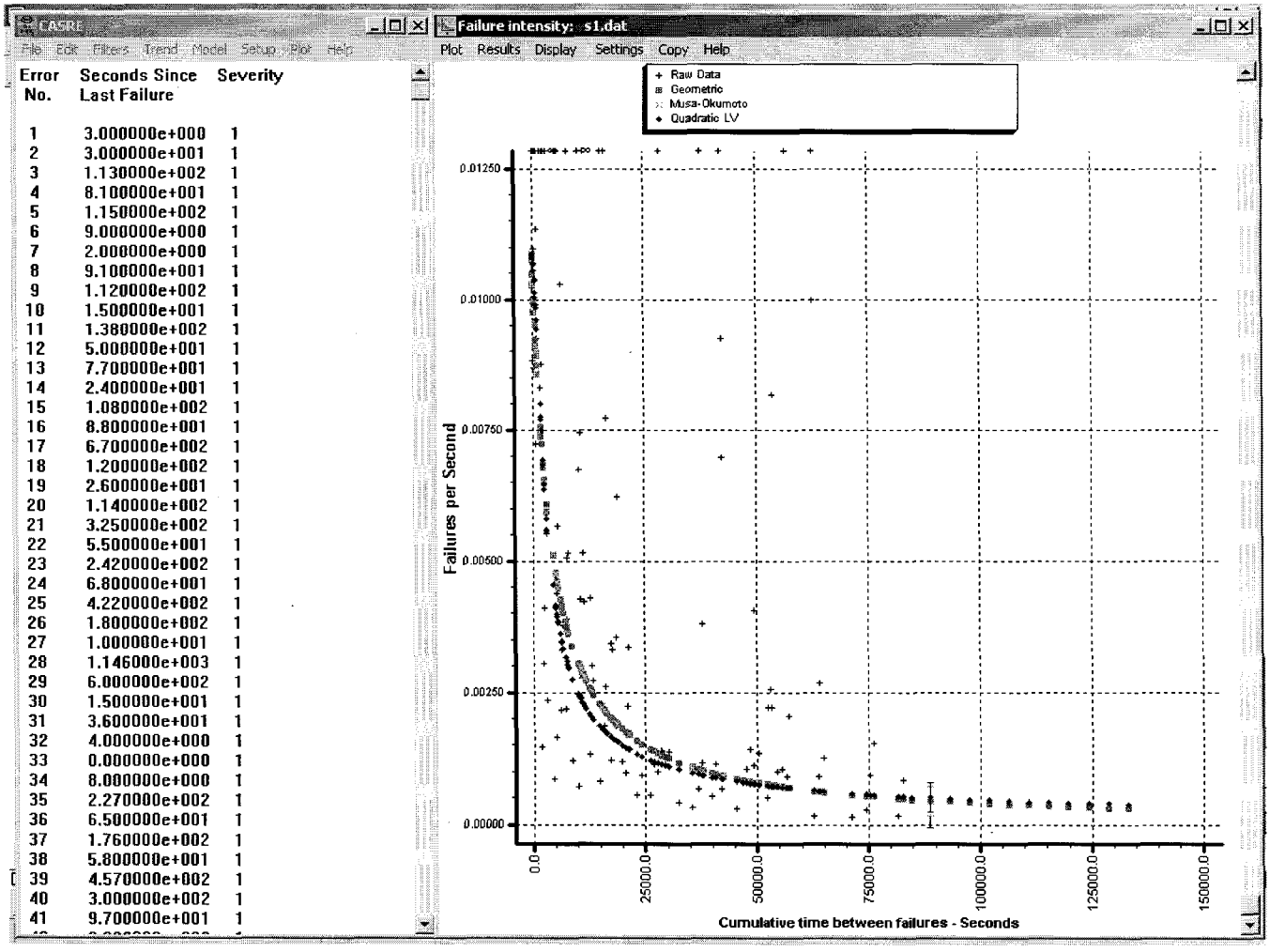
Show Model Results



Error No.	Seconds Since Last Failure	Severity
1	3.000000e+000	1
2	3.000000e+001	1
3	1.130000e+002	1
4	8.100000e+001	1
5	1.150000e+002	1
6	9.000000e+000	1
7	2.000000e+000	1
8	9.100000e+001	1
9	1.120000e+002	1
10	1.500000e+001	1
11	1.380000e+002	1
12	5.000000e+001	1
13	7.700000e+001	1
14	2.400000e+001	1
15	1.080000e+002	1
16	8.800000e+001	1
17	6.700000e+002	1
18	1.200000e+002	1
19	2.600000e+001	1
20	1.140000e+002	1
21	3.250000e+002	1
22	5.500000e+001	1
23	2.420000e+002	1
24	6.800000e+001	1
25	4.220000e+002	1
26	1.800000e+002	1
27	1.000000e+001	1
28	1.146000e+003	1
29	6.000000e+002	1
30	1.500000e+001	1
31	3.600000e+001	1
32	4.000000e+000	1
33	0.000000e+000	1
34	8.000000e+000	1
35	2.270000e+002	1
36	6.500000e+001	1
37	1.760000e+002	1
38	5.800000e+001	1
39	4.570000e+002	1
40	3.000000e+002	1
41	9.700000e+001	1

Software Reliability Estimation with CASRE - Example

Change Display to Failure Intensity



Error No.	Seconds Since Last Failure	Severity
1	3.000000e+000	1
2	3.000000e+001	1
3	1.130000e+002	1
4	8.100000e+001	1
5	1.150000e+002	1
6	9.000000e+000	1
7	2.000000e+000	1
8	9.100000e+001	1
9	1.120000e+002	1
10	1.500000e+001	1
11	1.380000e+002	1
12	5.000000e+001	1
13	7.700000e+001	1
14	2.400000e+001	1
15	1.080000e+002	1
16	8.800000e+001	1
17	6.700000e+002	1
18	1.200000e+002	1
19	2.600000e+001	1
20	1.140000e+002	1
21	3.250000e+002	1
22	5.500000e+001	1
23	2.420000e+002	1
24	6.800000e+001	1
25	4.220000e+002	1
26	1.800000e+002	1
27	1.000000e+001	1
28	1.146000e+003	1
29	6.000000e+002	1
30	1.500000e+001	1
31	3.600000e+001	1
32	4.000000e+000	1
33	0.000000e+000	1
34	8.000000e+000	1
35	2.270000e+002	1
36	6.500000e+001	1
37	1.760000e+002	1
38	5.800000e+001	1
39	4.570000e+002	1
40	3.000000e+002	1
41	9.700000e+001	1

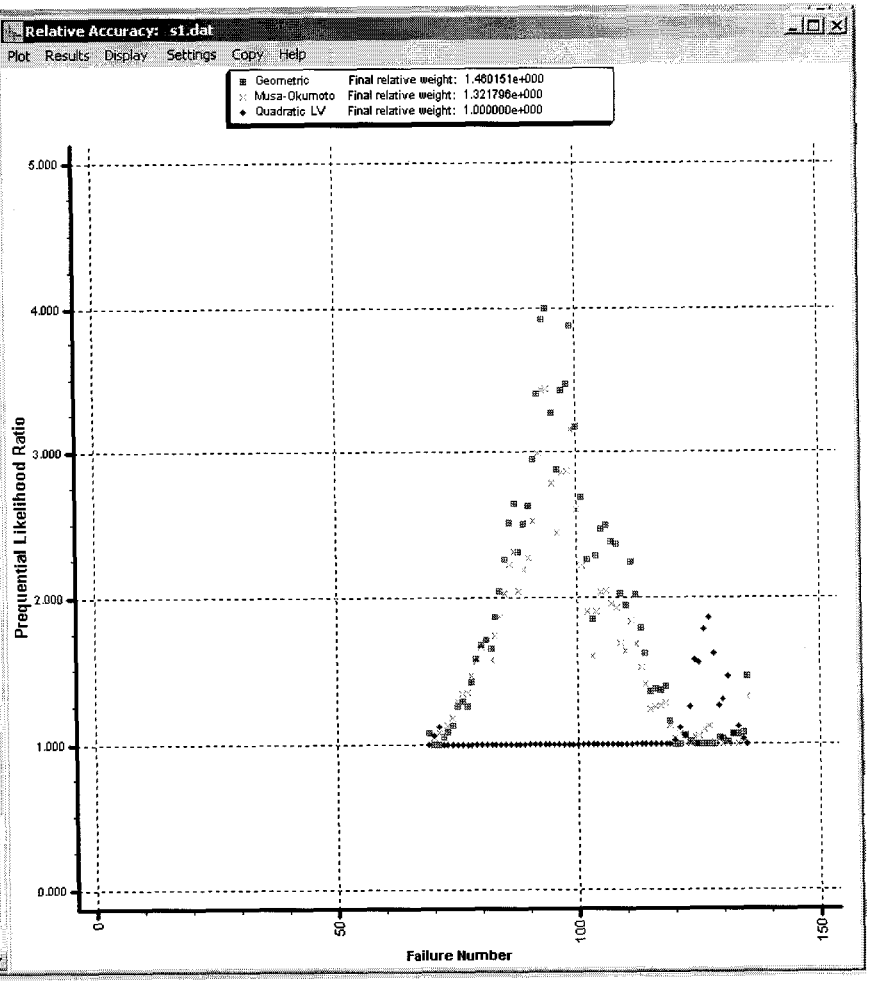
Software Reliability Estimation with CASRE - Example

Model Applicability – Prequential Likelihood Ratio ("Relative Accuracy")

CASRE

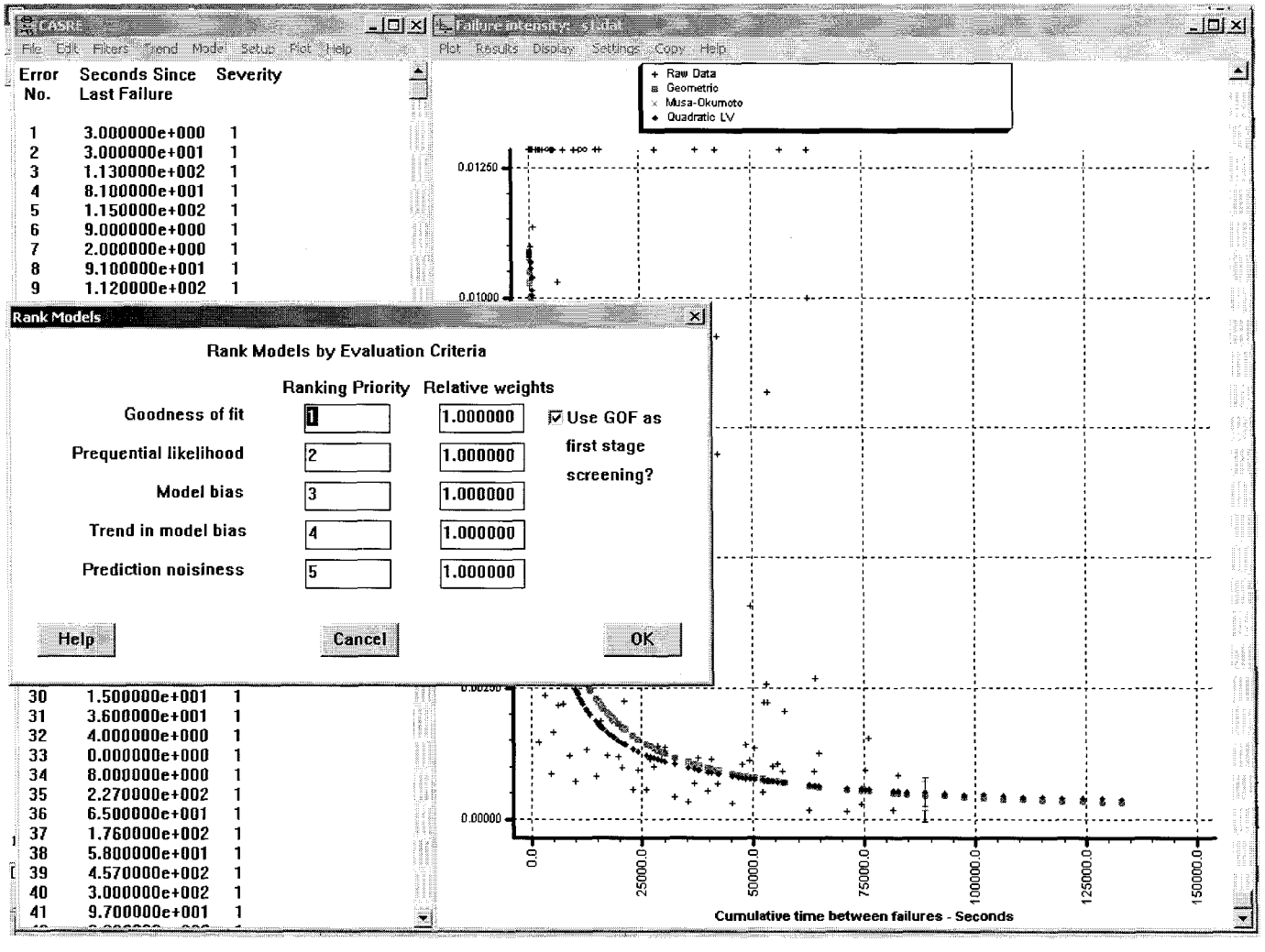
File Edit Filters Trend Model Setup Plot Help

Error No.	Seconds Since Last Failure	Severity
1	3.000000e+000	1
2	3.000000e+001	1
3	1.130000e+002	1
4	8.100000e+001	1
5	1.150000e+002	1
6	9.000000e+000	1
7	2.000000e+000	1
8	9.100000e+001	1
9	1.120000e+002	1
10	1.500000e+001	1
11	1.380000e+002	1
12	5.000000e+001	1
13	7.700000e+001	1
14	2.400000e+001	1
15	1.080000e+002	1
16	8.800000e+001	1
17	6.700000e+002	1
18	1.200000e+002	1
19	2.600000e+001	1
20	1.140000e+002	1
21	3.250000e+002	1
22	5.500000e+001	1
23	2.420000e+002	1
24	6.800000e+001	1
25	4.220000e+002	1
26	1.800000e+002	1
27	1.000000e+001	1
28	1.146000e+003	1
29	6.000000e+002	1
30	1.500000e+001	1
31	3.600000e+001	1
32	4.000000e+000	1
33	0.000000e+000	1
34	8.000000e+000	1
35	2.270000e+002	1
36	6.500000e+001	1
37	1.760000e+002	1
38	5.800000e+001	1
39	4.570000e+002	1
40	3.000000e+002	1
41	9.700000e+001	1



Software Reliability Estimation with CASRE - Example

Rank Models



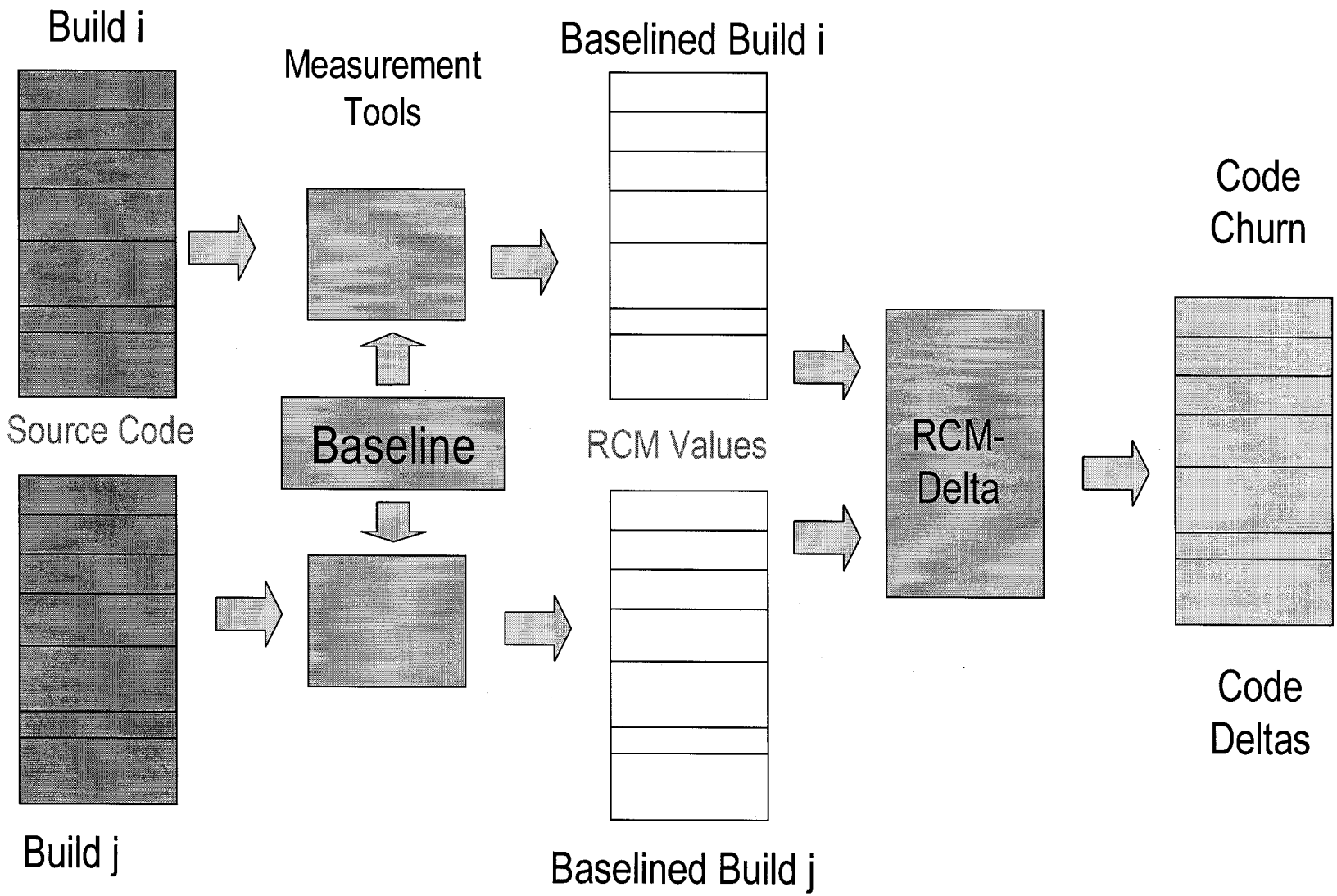
Organizations Using Software Reliability Estimation Techniques

- AFOTEC
- Cisco
- Raytheon (WAAS)
- Motorola
- United Space Alliance (STS Primary Avionics System Software)

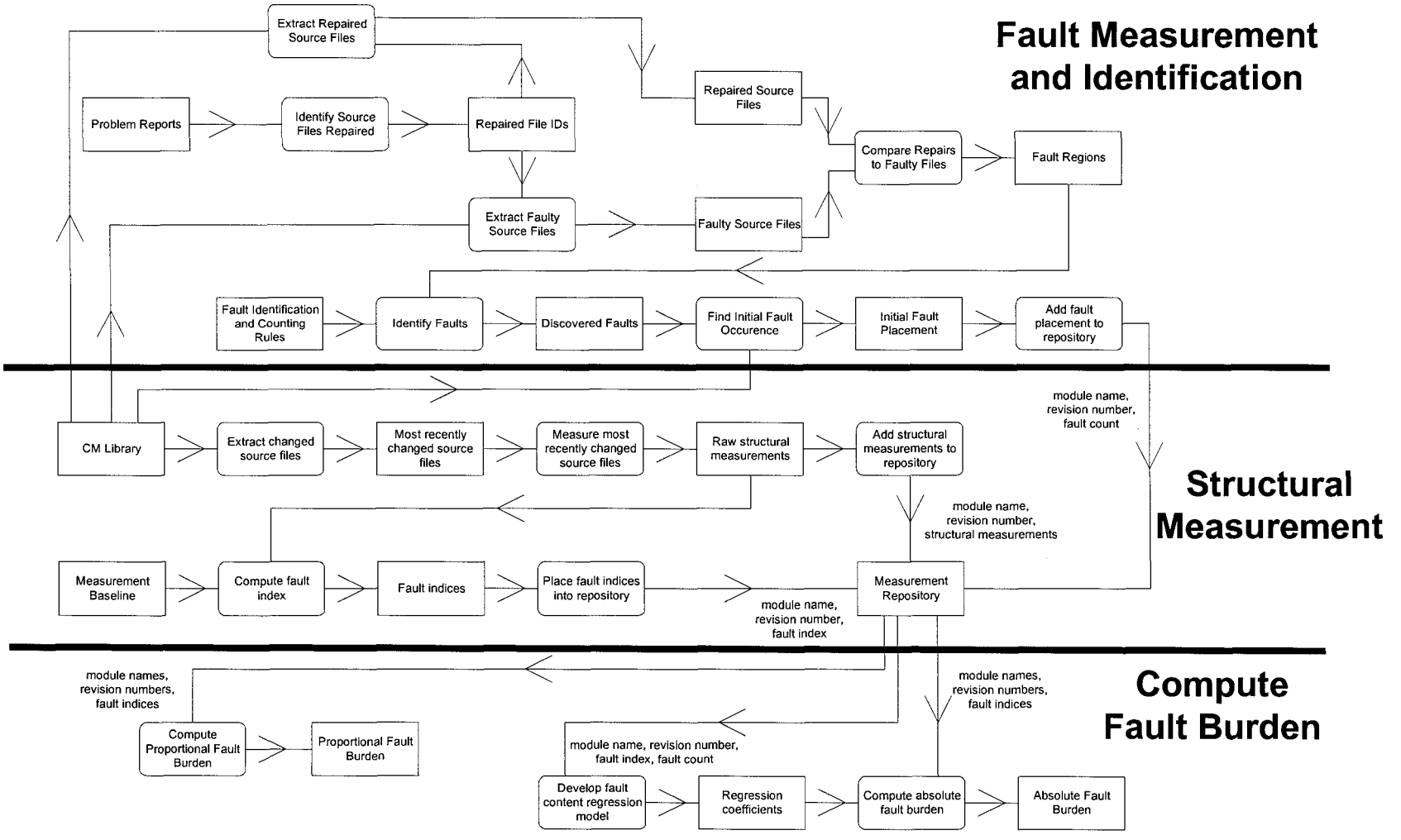
Estimating Software Fault Content

- Goal – Improve visibility into and control of software reliability during development
- How do we estimate reliability prior to test?
  - Failure data is usually not available
- Measures of software structure can be used to identify fault-prone components, estimate fault content
  - Identifying fault-prone components
    - Boolean discriminant functions (BDFs) use critical values of software structural measurements to identify fault-prone components
    - Relative Critical Value Deviation (RCVD) – difference between a module's software metric and its critical value indicates extent to which the module's quality differs from the norm.
  - Recent work indicates strong relationships between measurements of structural evolution and number of faults inserted
  - Structural measurements are easy to collect
  - Absolute or proportional fault burdens of individual functions/methods can be estimated prior to test.
- Currently collaborating with JPL development effort in fielding measurement framework

Measuring Structural Evolution - Comparing Two Builds



A Fault Measurement and Estimation Framework



Future Work

- Collect data from additional projects to:
  - Improve accuracy of estimated fault introduction rate
  - Determine fault removal rates
  - Determine effects of development characteristics to a greater extent
- Extend measurement methods to earlier development phases
- Refine and formalize fault identification and counting rules. Include ways of identifying faults in:
  - Object hierarchies
  - Real-time constructs

References

1	A. Nikora, N. Schneidewind, J. Munson, "IV&V Issues in Achieving High Reliability and Safety in Critical Control System Software," JPL D-15740, final report, January 19, 1998. In 3 volumes: NASA INTERNAL DISTRIBUTION ONLY – Volume 1 - "Measuring and Evaluating the Software Maintenance Process and Metrics-Based Quality Control" – Volume 2 – "Measuring Defect Insertion Rates and Risk of Exposure to Residual Defects in Evolving Software Systems" – Volume 3 – "Appendices"
2	J. Munson, A. Nikora, "Estimating Rates of Fault Insertion and Test Effectiveness in Software Systems", proceedings of the Fourth ISSAT International Conference on Quality and Reliability in Design, Seattle, WA, August 12-14, 1998
3	A. Nikora, J. Munson, "Determining Fault Insertion Rates For Evolving Software Systems", proceedings of the Ninth International Symposium on Software Reliability Engineering, Paderborn, Germany, November 4-7, 1998
4	Norman F. Schneidewind, "Reliability Modeling for Safety Critical Software", IEEE Transactions on Reliability, Vol. 46, No.1, March 1997, pp.88-98.
5	N. Schneidewind, A. Nikora, "Predicting Deviations in Software Quality by Using Relative Critical Value Deviation Metrics", proceedings of the 10th International Symposium on Software Reliability Engineering, Boca Raton, FL, Nov 1-4, 1999