SOFTWARE SUMMARY FORM

01.Summary Date: 07/25/94	02. Summary prepared by(Name and Phone) T.J. Ratchford 522-3083	03. Summary Action: New
04. Software Date: 7/25/94	05. Short Title: SEISMO	
06. Software Title: SEISMO - Simulation of Seismo-Mechanical Scenarios and Model Description.		07. Internal Software ID: NONE

08. Software Type:	09.Processing Mode:	10. APPLICATION AREA
☐ Automated Data System	☐ Interactive	A. General:
☐ Computer Program	☐ Batch	☐ Scientific/Engineering ☐ Auxiliary Analyses
■ Subroutine/Module	■ Combination	☐ Total System PA
		■ Subsystem PA ☐ Other
		b. Specific:

11. Submitting Organization and Address:	12. Technical Contact(s) and Phone:
CNWRA, SwRI, San Antonio, Texas	N. Eisenberg NRC

13. Narrative:

SEISMO - SEISMO Module is to provide a computational algorithm for estimating the consequences to the HLW repository due to a seismic event, in the TPA simulations.

14. Computer Platform	15. Computer Operating System:	16. Programming Language(s):	17. Number of Source Program Statements:
CRAY/XMP	UNIX	FORTRAN	7,624 lines of code
18. Computer Memory Requirements: UNKNOWN	19. Tape Drives: NONE	20. Disk/Drum Units: N/A	21. Graphics: UNKNOWN

22. Other Operational Requirements

NONE

23. Software Availability:	24. Documentation Availability:
■ Available ☐ Limited ☐ In-House ONLY	■ Available ☐ Inadequate ☐ In-House ONLY

25. Submission Package Status:

Acceptance Criteria: Met ■ Not Met ☐ Software QA Assessment: Successful ■ Unsuccessful ☐

Code Custodian: T.J. Ratchford Date: 7/25/94

CNWRA Form TOP-4-1 (08/93)

```
gemstone.2 ~/tpa/SEISMO/VCS => ls -l
total 5251
-rwxrwx---   1 tjr1     tjr1         1281 Jul  8  1993 s.Makefile*
-rwxrwx---   1 tjr1     tjr1         1152 Jul  8  1993 s.accel.F*
-rwxrwx---   1 tjr1     tjr1         3747 Jul  8  1993 s.analysis.F*
-rwxrwx---   1 tjr1     tjr1         1187 Jul  8  1993 s.case1.F*
-rwxrwx---   1 tjr1     tjr1         1078 Jul  8  1993 s.case2.F*
-rwxrwx---   1 tjr1     tjr1         1357 Jul  8  1993 s.common.H*
-rwxrwx---   1 tjr1     tjr1         3062 Jul  8  1993 s.echo1.F*
-rwxrwx---   1 tjr1     tjr1         1240 Jul  8  1993 s.echo2.F*
-rwxrwx---   1 tjr1     tjr1       344379 Jul  8  1993 s.lhs.out*
-rwxrwx---   1 tjr1     tjr1        88449 Jul  8  1993 s.lhs.out.Z*
-rwxrwx---   1 tjr1     tjr1         9502 Jul  8  1993 s.opnfil.F*
-rwxrwx---   1 tjr1     tjr1         2410 Jul  8  1993 s.rdmor.F*
-rwxrwx---   1 tjr1     tjr1         2496 Jul  8  1993 s.rdrun.F*
-rwxrwx---   1 tjr1     tjr1         2580 Jul  8  1993 s.rdsot.F*
-rwxrwx---   1 tjr1     tjr1         2219 Jul  8  1993 s.seismo.F*
-rwxrwx---   1 tjr1     tjr1        36902 Jul  8  1993 s.seismo.cpp*
-rwxrwx---   1 tjr1     tjr1          818 Jul  8  1993 s.seismo.in*
-rwxrwx---   1 tjr1     tjr1        33982 Jul  8  1993 s.seismo.pre*
-rwxrwx---   1 tjr1     tjr1         1797 Jul  8  1993 s.sotout.F*
-rwxrwx---   1 tjr1     tjr1     10459328 Jul  8  1993 s.sotsei.da*
-rwxrwx---   1 tjr1     tjr1     10459328 Jul  8  1993 s.sotsei.dat*
gemstone.3 ~/tpa/SEISMO/VCS =>
```

YNR 7/25/94

SEISMO Fortran Program
Static and Dynamic Analysis

June 29, 1993

Earl S. Marwil
John E. Tolli
Scientific Computing Unit
Idaho National Engineering Laboratory

1. Introduction

This analysis was performed on the Cray version of the software as provided by Southwest Research Institute (SwRI).

One sample problem was supplied along with the source code. The program was analyzed using the Craft (Cross Reference Analysis of Fortran) tool, FORWARN, the Fortran 77 analyzer, and PC-Metric. These tools provide static analysis, coverage analysis, and complexity analysis.

2. References

[1] N.H. Marshall and E.S. Marwil, Cross Reference Analysis of Fortran (CRAFT), EG&G-CATT-9198, EG&G Idaho, Inc., July 1991.
[2] Fortran 77 Analyzer User's Manual, National Bureau of Standards, NBS GCR 81-359, 1981
[3] FORWARN User's Guide, Quibus Enterprises, Inc., July 1991.
[4] PC-Metric User's Guide, SET Laboratories, Inc., 1987.

3. Functions

There are 12 entry points with no alternate entry points.

There are no unreferenced subroutines or functions.

4. Common Block Irregularities

There are 3 common blocks. All common block declarations are consistent. Common block "canth" is declared but unreferenced in "echo2", "rdmor", "rdrun", and "sotout". Common block "faild" is declared but unreferenced in "echo1", "rdmor", "rdrun", and "rdsot"

5. Interface Irregularities

Argument usage is consistent.

6. Local Variable Irregularities

Parameter usage is consistent.

The local variable "nz" is read from input data in "rdsot", but is not used.

7. Fortran Extensions

All routines except "accel" contain some lower case alphabetic characters in their active Fortran. Routines "analysis", "case1", "case2", "echo1", "echo2", "rdmor", "rdrun", "rdsot", "seismo", and "sotout" have some entity names which are longer than 6 characters. All such usage is non-ANSI standard.

8. Optimization

The following table summarizes the performance data gathered from execution of the sample problem. Only those routines exercised by the sample problem are shown (see "Coverage Analysis" for a list of routines not exercised by the sample problem, i.e., coverage = 0%). The table lists all program modules in descending order according to CPU time. To optimize code execution time, emphasis should be placed on those modules which appear highest in the listing.

In order to obtain meaningful statistics for performance evaluation, the program should execute for a reasonable amount of time. Note that the execution time for this sample problem is short (<< 10 sec) and that the resulting statistics may therefore not accurately reflect program performance for longer runs.

The performance data show that a high percentage of the overall execution time (91.815%) is spent in the first 2 routines listed (RDSOT, ECHO1). This is due primarily to the following:

1) a low percentage of floating point operations which are performed in vector mode (%Vflops is small)

2) a high rate of instruction buffer fetches in ECHO1 (IBFR > 1).

A detailed optimization analysis effort should focus on these 2 areas.

PERFORMANCE DATA FOR SEISMO

ROUTINE NAME	Time	%ExTime	%AccumT	%Vflops	IFact	MC/MR	IBFR
RDSOT	0.246	56.226	56.226	0.00000	0.00	0.093	0.938
ECHO1	0.156	35.589	91.815	0.00000	0.00	0.290	1.004
ANALYSIS	0.021	4.709	96.524	0.00000	0.00	0.097	1.209
ACCEL	0.005	1.123	97.647	0.00000	2.42	1.493	0.965
OPNFIL	0.004	0.854	98.501	0.00000	0.00	0.409	0.513
CASE1	0.002	0.452	98.953	0.00000	6.02	1.159	0.600
CASE2	0.002	0.363	99.316	0.00000	7.50	1.656	0.748
ECHO2	0.001	0.201	99.517	0.00000	0.00	0.280	0.891
RDMOR	0.001	0.164	99.681	0.00000	0.00	0.247	0.764
SOTOUT	0.001	0.160	99.841	0.00000	0.00	0.278	1.038
RDRUN	0.001	0.134	99.975	0.00000	0.00	0.116	0.984
SEISMO	0.000	0.025	100.000	0.00000	0.00	0.724	1.118
Totals (All Traced Routines)	0.437	100.000	100.000	0.00000	0.25	0.165	0.969

Key:

- %AccumT = accumulated percentage of total CPU time
- %ExTime = percentage of total CPU time
- %Vflops = percentage of floating point operations due to vector floating point operations
- IBFR = Instruction Buffer Fetch Rate (megafetches/sec)
- IFact = Inline Factor (total calls to routine / average time spent in routine for each call)
- MC = number of memory conflicts
- MR = number of memory references
- Time = total CPU time (sec)

9. Coverage Analysis

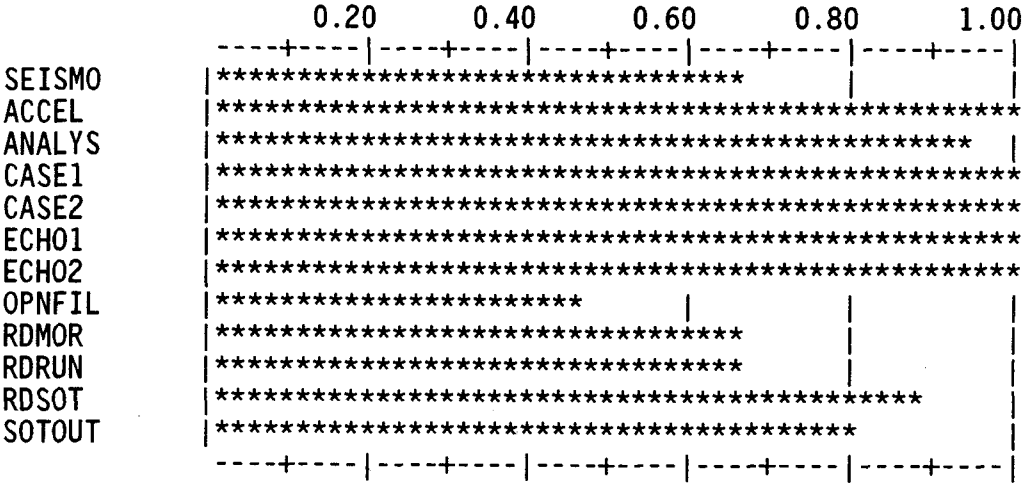
One sample problem was supplied. A coverage analysis shows that this problem yielded a 75% segment coverage of SEISMO. Sample problems provided with simulation programs typically achieve 35% to 50% coverage. A statement of software quality cannot be made for routines that have low coverage, i.e., large portions of the code are untested.

All routines have at least 40% coverage.

One routine achieves 40%-59% coverage, 4 routines achieve 60%-79% coverage, 2 routines achieve 80%-99% coverage, and 5 routines achieve 100% coverage.

The following table shows the percent coverage for each routine.

Module Name	Number of Segments in module	Number of Segments Executed	Percent Segment Coverage
SEISMO	3	2	66.7
ACCEL	1	1	100.0
ANALYS	15	14	93.3
CASE1	3	3	100.0
CASE2	3	3	100.0
ECHO1	5	5	100.0
ECHO2	3	3	100.0
OPNFIL	28	13	46.4
RDMOR	6	4	66.7
RDRUN	3	2	66.7
RDSOT	16	14	87.5
SOTOUT	5	4	80.0
Totals	91	68	74.7



0.40 <= coverage < 0.60	OPNFIL					
0.60 <= coverage < 0.80	SEISMO	RDMOR	RDRUN	SOTOUT		
0.85 <= coverage < 0.90	RDSOT					
0.90 <= coverage < 0.95	ANALYS					
coverage = 1.00	ACCEL	CASE1	CASE2	ECHO1	ECHO2	

Program coverage for this run =0.75

10. Complexity Analysis

Some key metrics are the number of executable statements (sloc), the number of non-blank comments (ncomt), McCabe's extended cyclomatic complexity (vg2), the number of branching statements (cgoto, ugoto, bIF, and lIF), and Halstead's predicted number of errors in (re)writing the code (bhat). Measures are normalized per 100 executable statements for ease of comparison and are listed in the table below.

The branching measures for this code indicate very few unconditional GO TO statements and logical IFs for most program modules. This code appears to be well structured.

Except for "analysis", all routines have a good ratio of non-blank comments to source code.

McCabe's extended cyclomatic complexity (vg2), normalized per 100 lines of source code, indicates moderate to high values. Generally, the routines with the highest complexity are those most likely to have defects. As a guideline, normalized measures of 15 or greater should be considered complex. A software maintenance program should focus on those routines with the highest measures.

Complexity Report by Subprogram for SEISMO

Name	loc	sloc	cmnt	ncomt	ncomt /sloc	vg2 /sloc	cgoto	cgoto /sloc	ugoto	ugoto /sloc	bIF	bif /sloc	lIF	lif /sloc	Bhat
SEISMO	64	19	45	36	189.5	10.5	0	0.0	0	0.0	1	5.3	0	0.0	0
ACCEL	23	6	17	15	250.0	16.7	0	0.0	0	0.0	0	0.0	0	0.0	0
ANALYSIS	151	7	11	0	0.0	114.3	0	0.0	0	0.0	0	0.0	0	0.0	1
CASE1	26	6	21	17	283.3	33.3	0	0.0	0	0.0	0	0.0	1	16.7	0
CASE2	21	5	18	15	300.0	40.0	0	0.0	0	0.0	0	0.0	1	20.0	0
ECHO1	103	29	44	38	131.0	10.3	0	0.0	0	0.0	0	0.0	0	0.0	0
ECHO2	55	9	38	32	355.6	22.2	0	0.0	0	0.0	0	0.0	0	0.0	0
opnfil	211	59	136	123	208.5	27.1	0	0.0	0	0.0	8	13.6	0	0.0	1
RDMOR	104	30	63	51	170.0	10.0	0	0.0	0	0.0	2	6.7	0	0.0	0
RDRUN	92	25	56	49	196.0	8.0	0	0.0	0	0.0	1	4.0	0	0.0	0
RDSOT	110	33	64	55	166.7	18.2	0	0.0	3	9.1	3	9.1	0	0.0	0
SOTOUT	78	18	46	38	211.1	16.7	0	0.0	0	0.0	1	5.6	0	0.0	0

Legend of Metrics in Report

loc -- lines of code
sloc -- number of executable statements
cmnt -- total number of commnts
ncomt -- number of non-blank COMMENT statements
100*ncomt/sloc -- percent, nonblank comments to number of executable statements
100*vg2/sloc -- percent, extended complexity of number of executable statements
cgoto -- number of COMPUTED GO TO statements
100*cgoto/sloc -- percent, computed GOTO's to number of executable statements
ugoto -- number of UNCONDITIONAL GO TO statements
100*ugoto/sloc -- percent, unconditional GOTO's to number of executable statements
bIF -- number of BLOCK IF statements
100*bif/sloc -- percent, Block IF statements to number of executable statements
lIF -- number of LOGICAL IF statements
100*lif/sloc -- percent, logical IF statements to number of executable statements
Bhat -- Halstead's predicted number of errors in writing code