SOFTWARE RELEASE NOTICE

01. SRN Number: EBS-SRN-035

02. Project Title:

SCCEX - Substantially Complete Containment Example Analysis.

Project No.

20-5702-524

03. SRN Title: SCCEX

04. Originator/Requester:

T. Ratchford

Date:

09/20/94

05. Summary of Actions

- ■ Release of new code admitted to CM System (P. Nair)
- ☐ Release of modified code:
  - ☐ Enhancements made
  - ☐ Corrections made
- ■ Change of access code (P. Lichtner)

06. Persons Authorized Access

Name	RO/RW	A/C/D

07. Element Manager Approval: Date:

08. Remarks:

A copy of the software package SCCEX, Ver. 1.1 was retained by the Principle Investigator for use in the CNWRA work center; therefore, a new release may not be necessary.

SOFTWARE SUMMARY FORM

01.Summary Date: 09/20/94	02. Summary prepared by(Name and Phone) T.J. Ratchford 522-3083	03. Summary Action: New
04. Software Date: 9/20/94	05. Short Title: SCCEX	
06. Software Title: SCCEX - Substantially Complete Containment Example Analysis.		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	P. Lichtner 210-522-6084

13. Narrative:

SCCEX - SCCEX computer code is designed to demonstrate an approach for evaluating containment by a waste package and to identify those areas that require more detailed analysis.

14. Computer Platform	15. Computer Operating System:	16. Programming Language(s):	17. Number of Source Program Statements:
CRAY/XMP	UNIX	FORTRAN	8863 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: 9/20/94

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

```
   8863 total
sn423(13) ls -l
total 100
]-rwxrwx---   1 tjr1     tjr1       21926 Jul 27 06:26 brine.f
-rwxrwx---    1 tjr1     tjr1        4696 Jul 27 06:26 bulk.f
-rwxrwx---    1 tjr1     tjr1        1317 Jul 27 06:26 ecorr.f
-rwxrwx---    1 tjr1     tjr1        2147 Jul 27 06:26 hist.f
-rwxrwx---    1 tjr1     tjr1        5013 Jul 27 06:26 input.f
-rwxrwx---    1 tjr1     tjr1        4777 Jul 27 06:26 inside.out
-rwxrwx---    1 tjr1     tjr1      225569 Jul 27 06:26 lsodes.f
-rwxrwx---    1 tjr1     tjr1       11292 Jul 27 06:26 main.out
-rwxrwx---    1 tjr1     tjr1       14907 Jul 27 06:26 mech.f
-rwxrwx---    1 tjr1     tjr1       13252 Jul 27 06:26 mechanic.out
-rwxrwx---    1 tjr1     tjr1        7361 Jul 27 06:26 ode.f
-rwxrwx---    1 tjr1     tjr1        6161 Jul 27 06:26 release.out
-rwxrwx---    1 tjr1     tjr1        1802 Jul 27 06:26 sccex_co.inp
-rwxrwx---    1 tjr1     tjr1         118 Jul 27 06:26 sccsml.dat
-rwxrwx---    1 tjr1     tjr1        9286 Jul 27 06:26 simple.out
-rwxrwx---    1 tjr1     tjr1         768 Jul 27 06:26 solve.f
-rwxrwx---    1 tjr1     tjr1       24354 Jul 27 06:26 subrouti.f
-rwxrwx---    1 tjr1     tjr1         128 Jul 27 06:26 summary.out
-rwxrwx---    1 tjr1     tjr1         509 Jul 27 06:26 vapor.f
-rwxrwx---    1 tjr1     tjr1         306 Jul 27 06:26 watt.f
sn423(14)
```

9/20/94

134.20.1.1 08:39:05

SCCEX Fortran Program
Static and Dynamic Analysis

DRAFT

August 31, 1994

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 used 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.

The program was found to abort when loaded with a core preset of indefinite. A core preset of zero was used for the dynamic analysis runs.

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

The SCCEX program contains 58 Fortran routines.

There are no alternate entry points.

Module "rkqc" is declared as an external but not used in module "sccex".

There are 4 extraneous subroutines:

```
ratint, srcms, xsetf, xsetun.
```

4. Common Block Irregularities

```
572  SCCEX    Argument #16 to LSODES must be an external
              routine name
 20  SRCMS    Common block /LS0001/ differs in layout from
              previous definitions (at RLS)
 21  SRCMS    Common block /LSS001/ differs in layout from
              previous definitions (at ILSS)
 22  SRCMS    Common block /EH0001/ differs in layout from
              previous definitions (at IEH)
 18  STODE    Common block /LS0001/ differs in layout from
              previous definitions (at CONIT)
```

6. Local Variable Irregularities

There are some instances of a parameter not being used in a module in which it is declared:

Parameter	Modules not using
rgas	drip
two	odeint

The following parameters are assigned inconsistent values:

nmax, rgas.

Local variable exceptions are noted as follows:

Module	Variable	Exception
bsstep	errc	unused
bsstep	ieq	unused
bsstep	yseq	possibly undefined
corrode	henry	undefined, unused
corrode	ph	defined, unused
corrode	time	unused
d1mach	idum	unused
derivs	yy	unused
drip	tin	possibly undefined
lsodes	ho	defined, unused
md	tail	possibly undefined
mdp	ilp	defined, unused
mech	afail	defined, unused
mech	btype	defined, unused
odrv	next	defined, unused
rate	dum	defined, unused
rate	emax	possibly undefined
rate	tmpvol	defined, unused
sccex	dvdt	undefined, unused
sccex	dydt	undefined, unused
sccex	nbad	possibly undefined
sccex	nok	possibly undefined
sccex	outttt	undefined, unused

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.

The performance data show that a high percentage of the overall execution time (83.972%) is spent in the first 5 routines listed. This is due primarily to the following (applies to some or all of the 5 routines):

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

2) a high overhead factor for calls to the routines (IFact > 1)

3) a high rate of instruction buffer fetches (IBFR > 1).

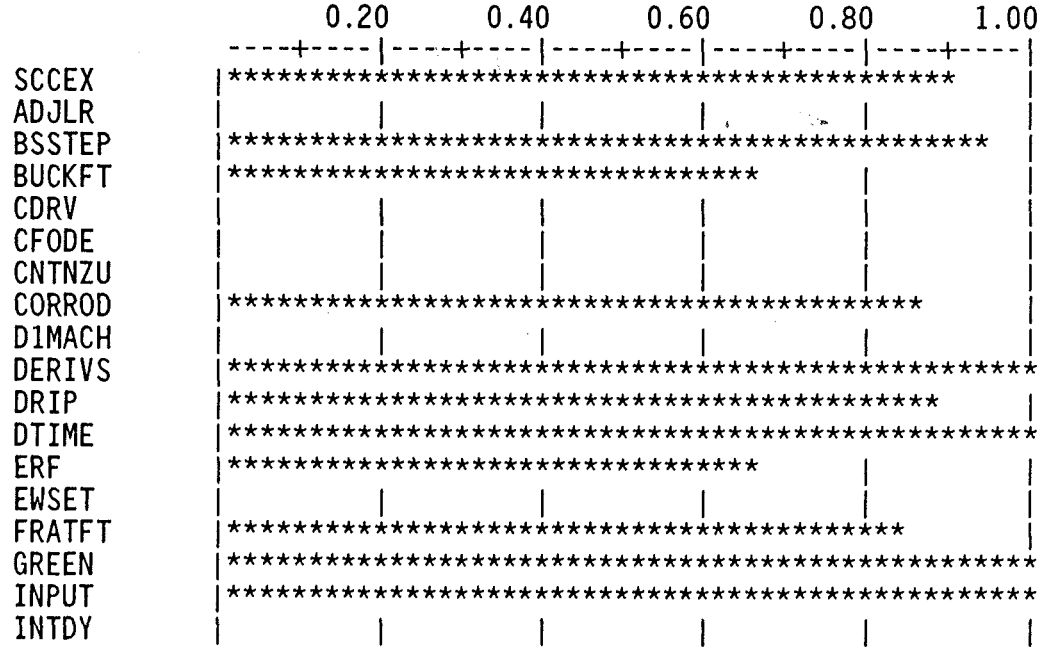
A detailed optimization analysis effort should focus on these 3 areas.

PERFORMANCE DATA FOR SCCEX

ROUTINE NAME	Time	%ExTime	%AccumT	%Vflops	IFact	MC/MR	IBFR
ERF	2.739	25.619	25.619	0.00000	1071.61	0.983	0.644
LINT	1.986	18.574	44.193	0.00000	505.33	0.965	0.519
GREEN	1.927	18.029	62.222	0.00022	42.30	0.249	1.373
RATE	1.302	12.179	74.401	0.82926	0.93	0.491	1.051
POWER	1.023	9.571	83.972	21.90881	20.08	0.120	0.240
DERIVS	0.513	4.800	88.772	0.00612	39.72	0.323	1.337
MMID	0.287	2.688	91.460	76.05446	7.15	0.510	1.347
SCCEX	0.224	2.093	93.554	10.66701	0.00	0.137	1.084
BSSTEP	0.217	2.031	95.585	79.22197	1.97	0.183	0.859
DRIP	0.118	1.101	96.686	0.00000	10.44	0.633	1.328
CORRODE	0.108	1.012	97.698	0.01120	0.14	0.073	0.100
RZEXTR	0.097	0.905	98.602	3.51440	21.23	0.892	0.569
RELEASE	0.074	0.690	99.293	0.00000	8.56	0.093	1.226
ODEINT	0.068	0.633	99.925	39.08563	0.01	0.477	1.756
INPUT	0.004	0.042	99.967	82.27850	0.00	0.118	1.015
MECH	0.002	0.020	99.987	12.29648	0.00	0.153	1.270
FRATFT	0.001	0.005	99.992	0.63694	0.14	0.085	1.008
YFUNC	0.000	0.004	99.997	62.99214	0.18	0.498	0.766
BUCKFT	0.000	0.002	99.998	0.00000	0.48	0.358	0.000
YILDFT	0.000	0.001	100.000	0.00000	0.55	0.861	0.000
DTIME	0.000	0.000	100.000	83.11689	0.02	0.389	0.806
Totals (All Traced Routines)	10.691	100.000	100.000	7.43120	1145.00	0.475	0.842

Key:
%AccumT = accumulated percentage of total CPU time

MDU	34	0	0.0
MECH	46	30	65.2
MMID	10	10	100.0
NNFC	60	0	0.0
NNSC	16	0	0.0
NNTC	16	0	0.0
NROC	17	0	0.0
NSFC	163	0	0.0
ODEINT	25	19	76.0
ODRV	17	0	0.0
POWER	6	6	100.0
PREP	104	0	0.0
PRJS	72	0	0.0
RATE	34	26	76.5
RATINT	19	0	0.0
RELEAS	8	7	87.5
RK4	9	0	0.0
RKQC	19	0	0.0
RZEXTR	15	15	100.0
SLSS	18	0	0.0
SRCMS	20	0	0.0
SRO	27	0	0.0
STODE	168	0	0.0
VNORM	3	0	0.0
XERRWV	13	0	0.0
XSETF	3	0	0.0
XSETUN	3	0	0.0
YFUNC	9	8	88.9
YILDFT	3	2	66.7
Totals	1716	263	15.3



0.80 <= coverage < 0.85	FRATFT				
0.85 <= coverage < 0.90	CORROD	DRIP	RELEAS	YFUNC	
0.90 <= coverage < 0.95	SCCEX	BSSTEP			
coverage = 1.00	DERIVS POWER	DTIME RZEXTR	GREEN	INPUT	MMID

Program coverage for this run =0.15

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 a fairly heavy reliance on unconditional GO TO statements and logical IFs for several program modules (e.g., CDRV, LSODES).

Many routines show a low ratio of non-blank comments to source code (e.g., BSSTEP, ERF).

McCabe's extended cyclomatic complexity (vg2), normalized per 100 lines of source code, indicates 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.

POWER	132	15	102	90	600.0	26.7	0	0.0	0	0.0	1	6.7	0	0.0	0
PREP	271	192	75	59	30.7	24.5	2	1.0	28	14.6	0	0.0	28	14.6	3
PRJS	227	131	86	77	58.8	26.7	1	0.8	9	6.9	0	0.0	17	13.0	2
RATE	445	108	304	248	229.6	14.8	0	0.0	0	0.0	7	6.5	4	3.7	2
RATINT	42	39	3	1	2.6	20.5	0	0.0	0	0.0	2	5.1	1	2.6	1
RELEASE	185	22	141	122	554.5	18.2	0	0.0	0	0.0	1	4.5	1	4.5	0
RK4	26	21	0	0	0.0	23.8	0	0.0	0	0.0	0	0.0	0	0.0	0
RKQC	56	45	5	3	6.7	17.8	0	0.0	1	2.2	3	6.7	1	2.2	0
RZEXTR	41	37	1	0	0.0	18.9	0	0.0	0	0.0	2	5.4	0	0.0	0
SLSS	80	23	34	31	134.8	34.8	1	4.3	2	8.7	0	0.0	3	13.0	0
SRCMS	54	26	20	12	46.2	38.5	0	0.0	1	3.8	0	0.0	1	3.8	0
SRO	106	49	58	33	67.3	28.6	0	0.0	5	10.2	0	0.0	5	10.2	1
STODE	509	300	237	235	78.3	25.7	1	0.3	57	19.0	0	0.0	45	15.0	4
VNORM	21	6	12	11	183.3	33.3	0	0.0	0	0.0	0	0.0	0	0.0	0
XERRWV	118	16	98	91	568.7	43.7	0	0.0	1	6.2	0	0.0	6	37.5	0
XSETF	11	4	6	2	50.0	75.0	0	0.0	0	0.0	0	0.0	1	25.0	0
XSETUN	11	4	14	9	225.0	50.0	0	0.0	0	0.0	0	0.0	1	25.0	0
YFUNC	66	57	14	6	10.5	8.8	0	0.0	1	1.8	2	3.5	1	1.8	0
YILDFT	25	6	14	6	100.0	33.3	0	0.0	0	0.0	0	0.0	1	16.7	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