SOFTWARE RELEASE NOTICE

01. SRN Number: GHGC-SRN-119	
02. Project Title: UNSAT-H -Unsaturated Soil Water and Heat Flow Model	Project No. 20-5702-723
03. SRN Title: UNSAT-H	
04. Originator/Requestor: Budhi Sagar	Date: 01/22/96

05. Summary of Actions

- ☐ Release of new software
- ☐ Release of modified software:
  - ☐ Enhancements made
  - ☐ Corrections made
- ☐ Change of access software
- ■ Software Retirement

06. Persons Authorized Access

Name	RO/RW	A/C/D
N/A		

07. Element Manager Approval: E. C. Pe[illegible] Date: 1/26/96

08. Remarks:

Not considered important to regulatory reviews in revised FY96 OPS Plans.

CNWRA Form TOP-6 (06/95)

SOFTWARE SUMMARY FORM

01.Summary Date: 07/29/94	02. Summary prepared by(Name and Phone) T.J. Ratchford 522-3083	03. Summary Action: New
04. Software Date: 7/29/94	05. Short Title: UNSAT-H	
06. Software Title: UNSAT-H - Unsaturated Soil Water and HeAT fLOW Model.		07. Internal Software ID: NONE

08. Software Type: ☐ Automated Data System ■ Computer Program ☐ Subroutine/Module	09.Processing Mode: ☐ Interactive ☐ Batch ■ Combination	10. APPLICATION AREA A. General: ■ Scientific/Engineering ☐ Auxiliary Analyses ☐ Total System PA ☐ Subsystem PA ☐ Other b. Specific:
11. Submitting Organization and Address: CNWRA, SwRI, San Antonio, Texas		12. Technical Contact(s) and Phone: S. Stothoff 210-522-5208

13. Narrative:

UNSAT-H - UNSAT-H simulates the dynamic processes of infiltration, dranage, redistribution, evaporation, and uptake of water from soil by plants. In addition this version has options to calculate soil heat flow, surface-energy balance, and actual evaporation (without resorting to the concept of potential evapotranspiration).

14. Computer Platform CRAY/XMP	15. Computer Operating System: UNIX	16. Programming Language(s): FORTRAN	17. Number of Source Program Statements: 8,072 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: ■ Available ☐ Limited ☐ In-House ONLY	24. Documentation Availability: ■ 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: 7/29/94

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

07/29/94 14:38 Directory C:\SHELL\TEMP\UNSATH*.*
Free Disk Space:117645056

.	<CURRENT>	<DIR>			..	<PARENT>	<DIR>		
BALL	.INC	2777	07/29/94	09:41	BARRAY	.INC	622	07/29/94	09:41
BMET	.INC	746	07/29/94	09:41	BOUT	.INC	1248	07/29/94	09:41
BSOIL	.INC	1231	07/29/94	09:42	BTIME	.INC	368	07/29/94	09:42
BVAR	.INC	1015	07/29/94	09:42	B_L_R	.F	1371	07/29/94	09:42
CALPET	.F	1927	07/29/94	09:42	CLAY	.INP	9057	07/29/94	09:42
CLAY	.OUT	21087	07/29/94	09:42	DATAINH	.F	44377	07/29/94	09:42
DATAOUT	.F	3841	07/29/94	09:42	DDSUM	.F	4007	07/29/94	09:42
DELCHK	.F	1961	07/29/94	09:42	HARDCOPY	.F	3036	07/29/94	09:42
HDRYCALC	.F	979	07/29/94	09:42	HEATFLOW	.F	11476	07/29/94	09:42
INITSUM	.F	1680	07/29/94	09:42	LISTDATA	.F	8310	07/29/94	09:42
MANUAL	.WP5	43631	07/29/94	09:42	MYHRLY	.F	760	07/29/94	09:42
N62NP	.INP	14377	07/29/94	09:43	N62NP	.OUT	7012	07/29/94	09:43
N62NPH	.INP	28560	07/29/94	09:43	N62NPH	.OUT	9632	07/29/94	09:43
NETRAD	.F	2167	07/29/94	09:43	NODEZ	.F	801	07/29/94	09:43
POLYCH	.F	3481	07/29/94	09:43	POLYKH	.F	2936	07/29/94	09:43
POLYTH	.F	2708	07/29/94	09:43	PROFILER	.F	6245	07/29/94	09:43
READ	.ME	15984	07/29/94	09:43	READREC	.F	1702	07/29/94	09:43
REINIT	.F	3538	07/29/94	09:43	RLD	.F	534	07/29/94	09:44
SCAN	.F	3083	07/29/94	09:44	SUMMARY	.F	4064	07/29/94	09:44
THERME	.F	756	07/29/94	09:44	THERMK	.F	1119	07/29/94	09:44
TIMEX	.F	2202	07/29/94	09:44	TRIDAG	.F	801	07/29/94	09:44
UNSATH	.F	37413	07/29/94	10:06	V20TEST4	.INP	5755	07/29/94	09:44
V20TEST4	.OUT	15144	07/29/94	09:44					

7/29/94

UNSATH Fortran Program Static and Dynamic Analysis

March 10, 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).

Three sample problems were 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.

One sample problem ("mario") was used for the analysis. The companion programs DATAINH and DATAOUT, for input preprocessing and output postprocessing, respectively, were not analyzed.

UNSATH was found to abort when loaded with a core preset of indefinites.

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 UNSATH program contains 13 Fortran routines.

There are no alternate entry points.

There are no extraneous subroutines.

4. Common Block Irregularities

There are 7 common blocks in the UNSATH program.

All common block declarations are consistent.

Variable exceptions are noted as follows:

Block name	Variable	Exception
/met/	pet	undefined and unused
/met/	daymet	undefined and unused
/met/	alt	undefined and unused
/met/	pmb	undefined and unused

There are several instances of common blocks not being used by modules in which they are declared:

Block name	Modules not using
/array/	b_l_r
/ihy/	hdrycalc
/ipara/	b_l_r, delchk, rld, tridag
/logic/	b_l_r, delchk, hdrycalc, heatflow, netrad, polych, polykh, polyth, rld, therme, thermk, tridag
/rpara/	b_l_r, netrad, polyth, therme, tridag

5. Interface Irregularities

Module "hdrycalc", which has 1 argument, is called with no arguments twice from "unsath".

6. Local Variable Irregularities

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

Parameter	Modules not using
lub	b_l_r, delchk, hdrycalc, heatflow, netrad, polych, polykh, polyth, rld, therme, thermk, tridag
lui	b_l_r, delchk, hdrycalc, heatflow, netrad, polych, polykh, polyth, rld, therme, thermk, tridag
lur	b_l_r, delchk, hdrycalc, heatflow, netrad, polych, polykh, polyth, rld, therme, thermk, tridag
lus	hdrycalc, heatflow, netrad, polykh, polyth, rld, therme, thermk, tridag
m1	delchk, rld
m2	b_l_r, delchk, netrad, rld, tridag
m3	b_l_r, delchk, netrad, rld, tridag
m4	b_l_r, delchk, netrad, rld, tridag

Variable exceptions are noted as follows:

Module	Variable	Exception
b_l_r	daybeg	undefined, unused
	dayend	undefined, unused
	finish	undefined, unused
	kl	undefined, unused
	le	undefined, unused

delchk	daybeg	undefined, unused
	dayend	undefined, unused
	finish	undefined, unused
	rsave	defined, unused
hdrycalc	amount	undefined, unused
	daybeg	undefined, unused
	dayend	undefined, unused
	finish	undefined, unused
	le	undefined, unused
heatflow	amount	undefined, unused
	daybeg	undefined, unused
	dayend	undefined, unused
	finish	undefined, unused
	kl	undefined, unused
netrad	daybeg	undefined, unused
	dayend	undefined, unused
	finish	undefined, unused
	kl	undefined, unused
	le	undefined, unused
polych	amount	undefined, unused
	daybeg	undefined, unused
	dayend	undefined, unused
	finish	undefined, unused
polykh	amount	undefined, unused
	daybeg	undefined, unused
	dayend	undefined, unused
	finish	undefined, unused
polyth	amount	undefined, unused
	daybeg	undefined, unused
	dayend	undefined, unused
	finish	undefined, unused
rld	daybeg	undefined, unused
	dayend	undefined, unused
	finish	undefined, unused
therme	amount	undefined, unused
	daybeg	undefined, unused
	dayend	undefined, unused
	finish	undefined, unused
	kl	undefined, unused
thermk	amount	undefined, unused
	daybeg	undefined, unused
	dayend	undefined, unused
	finish	undefined, unused
	kl	undefined, unused
	le	undefined, unused
tridag	daybeg	undefined, unused
	dayend	undefined, unused
	finish	undefined, unused
unsath	ihpond	defined, unused
	le	undefined, unused
	psflux	defined, unused

7. Fortran Extensions

A data statement preceeds a specification statement in module "unsath".

A potential overlap in a character assignment statement occurs in each program module.

All program modules contain some lower case alphabetic characters in their active Fortran.

The following routines contain entity names which are longer than 6 characters:

```
unsath, b_l_r, hdrycalc, heatflow, netrad, therme, thermk.
```

Module "unsath" contains references to a common block and a subroutine which both have the same name ("time").

Format statement 10 in "unsath" has no comma to separate the string ' Time: ' from the format specifier A8.

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 more typical (possibly longer) runs.

The performance data show that a high percentage of the overall execution time (74.115%) is spent in the first 3 routines listed. This is primarily due to a lack of vectorization in the 3 routines (%Vflops = 0.00000). A detailed optimization analysis effort should focus on this area.

PERFORMANCE DATA FOR UNSATH

ROUTINE NAME	Time	%ExTime	%AccumT	%Vflops	IFact	MC/MR	IBFR
POLYKH	0.924	27.714	27.714	0.00000	0.00	0.806	0.467
POLYCH	0.896	26.867	54.582	0.00000	0.00	0.498	0.362
POLYTH	0.651	19.534	74.115	0.00000	0.01	0.520	0.510
UNSATH	0.288	8.647	82.762	92.60432	0.00	0.406	0.510
THERMK	0.196	5.873	88.635	99.96344	0.07	2.027	0.145
TRIDAG	0.169	5.081	93.716	0.00000	0.12	0.930	0.037
HEATFLOW	0.149	4.468	98.184	93.57477	0.01	0.307	0.207
THERME	0.059	1.770	99.954	100.00000	0.06	3.266	0.150
DELCHK	0.002	0.046	100.000	0.00000	0.58	0.728	0.840
Totals (All Traced Routines)							

```
                3.334 100.000 100.000  60.56352     0.09    0.733   0.393

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

A coverage analysis shows that the sample problem yielded a 51% segment coverage of UNSATH. Sample problems provided with simulation programs typically achieve only 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.

Note that 4 routines have 0% coverage. These routines are not tested with the supplied sample problem.

Three routines achieve 20%-39% coverage, 1 routine achieves 40%-59% coverage, 1 routine achieves 60%-79% coverage, 1 routine achieves 80%-99% coverage, and 3 routines achieve 100% coverage.

Module Name	Number of Segments in module	Number of Segments Executed	Percent Segment Coverage
UNSATH	475	239	50.3
B_L_R	11	0	0.0
DELCHK	23	20	87.0
HDRYCALC	4	0	0.0
HEATFLOW	90	62	68.9
NETRAD	18	0	0.0
POLYCH	29	10	34.5
POLYKH	25	9	36.0
POLYTH	22	7	31.8
RLD	3	0	0.0
THERME	3	3	100.0
THERMK	6	6	100.0
TRIDAG	5	5	100.0
Totals	714	361	50.6

```
                        0.20      0.40      0.60      0.80      1.00
                  ----+----|----+----|----+----|----+----|----+----|
      UNSATH      |************************    |         |         |
      B_L_R       |         |         |         |         |         |
      DELCHK      |*******************************************      |
```

```
HDRYCALC   |         |         |         |         |         |
HEATFLOW   |*********************************      |         |
NETRAD     |         |         |         |         |         |
POLYCH     |*****************  |         |         |         |
POLYKH     |******************* |         |         |         |
POLYTH     |****************   |         |         |         |
RLD        |         |         |         |         |         |
THERME     |*************************************************
THERMK     |*************************************************
TRIDAG     |*************************************************
            ----+----|----+----|----+----|----+----|----+----|
```

coverage = 0.	B_L_R	HDRYCALC	NETRAD	RLD
0.20 <= coverage < 0.40	POLYCH	POLYKH	POLYTH	
0.40 <= coverage < 0.60	UNSATH			
0.60 <= coverage < 0.80	HEATFLOW			
0.85 <= coverage < 0.90	DELCHK			
coverage = 1.00	THERME	THERMK	TRIDAG	

Program coverage for this run =0.51

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 few unconditional GO TO statements and logical IFs for most program modules. This code appears to be fairly well structured.

Most routines have a good ratio of non-blank comments to source code. Note, however, that HDRYCALC and HEATFLOW have no non-blank comments.

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.

Complexity Report by Subprogram for UNSATH

Name	loc	sloc	cmnt	ncomt	ncomt /sloc	vg2 /sloc	cgoto	cgoto /sloc	ugoto	ugoto /sloc	bIF	bif /sloc	lIF	lif /sloc	Bhat
UNSATH	1383	983	305	199	20.2	24.3	0	0.0	30	3.1	102	10.4	67	6.8	24
B_L_R	141	22	77	64	290.9	22.7	0	0.0	1	4.5	1	4.5	2	9.1	1
DELCHK	131	39	70	57	146.2	28.2	0	0.0	2	5.1	4	10.3	5	12.8	0
HDRYCALC	157	130	43	0	0.0	1.5	0	0.0	0	0.0	0	0.0	0	0.0	0
HEATFLOW	452	58	27	0	0.0	77.6	0	0.0	0	0.0	0	0.0	0	0.0	6
NETRAD	161	38	86	69	181.6	28.9	0	0.0	0	0.0	3	7.9	5	13.2	1
POLYCH	198	67	82	65	97.0	17.9	1	1.5	6	9.0	5	7.5	1	1.5	1
POLYKH	173	57	73	60	105.3	17.5	1	1.8	5	8.8	4	7.0	0	0.0	1
POLYTH	167	50	73	60	120.0	18.0	1	2.0	5	10.0	3	6.0	0	0.0	1
RLD	80	6	54	41	683.3	33.3	0	0.0	0	0.0	0	0.0	1	16.7	0
THERME	131	13	68	55	423.1	15.4	0	0.0	0	0.0	0	0.0	0	0.0	0
THERMK	155	21	75	62	295.2	14.3	0	0.0	0	0.0	1	4.8	0	0.0	0
TRIDAG	91	17	52	39	229.4	58.8	0	0.0	0	0.0	0	0.0	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