FUNWAVE-TVD Code Analysis Using SciTools *Understand*

Daniel Topa
HPCMP PETTT On Site

October 20, 2016

Abstract

How can a static analysis tool enhance development of the open source code FUNWAVE? Here we select the tool *Understand* and display different aspects of the code analysis; the results are invaluable. Should more time be devoted to a closer examination with this tool?

1 Conclusions

The reports from *Understand* paint a picture of a healthy, disciplined development project and suggest guidelines to keep the project healthy moving forward.

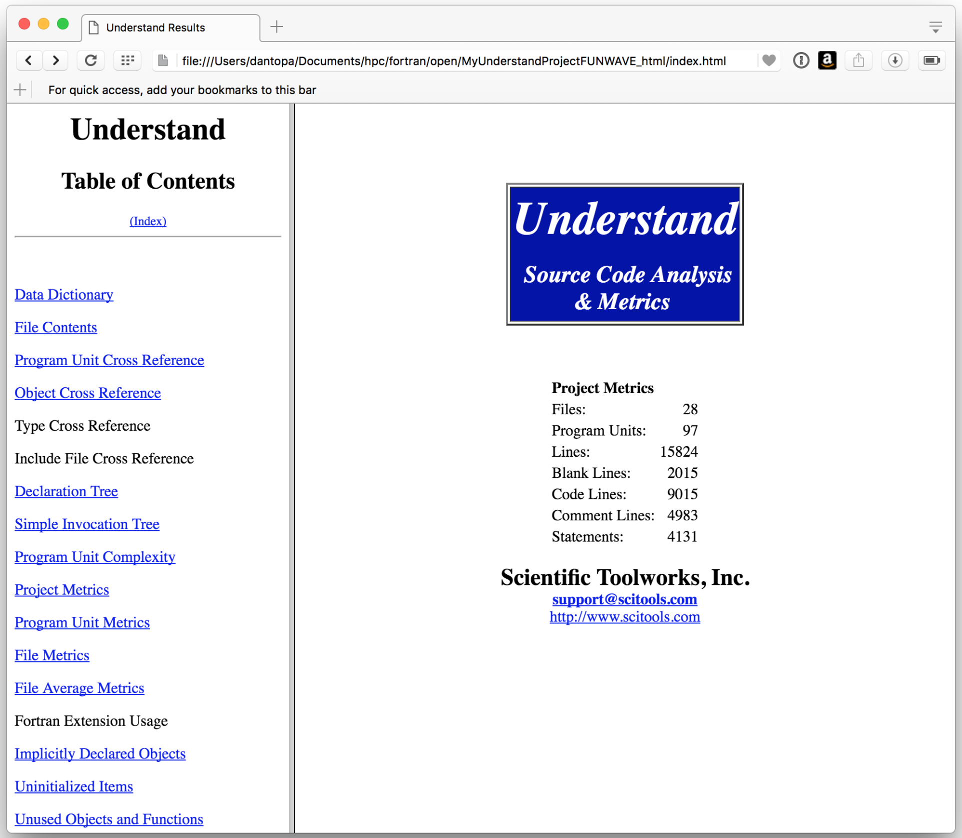
The information generated by the static analysis can greatly streamline debugging and accelerate development, and is an essential set of tools. Certainly this substantially aids development. The question is whether to select *Understand* or a comparable package to provide the toolkit.

2 Summary Report

The application generates a summary report as shown in figure 1. Below we pictorially survey the html form; a text file version is included in the zip file.

The subreports linked in the summary report provide a host of insights. In particular, it presents a set of maps and inverse maps. For example, if we encounter a variable and wish to check it's definition, we would consult the data dictionary. If we want to jump to a subroutine, we find a hyperlink in the data dictionary. If we encounter an object and wish to check it's where it is defined, set, and used, consult the object cross reference.

The following pages survey some of the 777 diagrams generated in the analysis of the code.



Understand Results

file:///Users/dantopa/Documents/hpc/fortran/open/MyUnderstandProjectFUNWAVE_html/index.html

For quick access, add your bookmarks to this bar

Understand

Table of Contents

(Index)

Data Dictionary
File Contents
Program Unit Cross Reference
Object Cross Reference
Type Cross Reference
Include File Cross Reference
Declaration Tree
Simple Invocation Tree
Program Unit Complexity
Project Metrics
Program Unit Metrics
File Metrics
File Average Metrics
Fortran Extension Usage
Implicitly Declared Objects
Uninitialized Items
Unused Objects and Functions

Understand

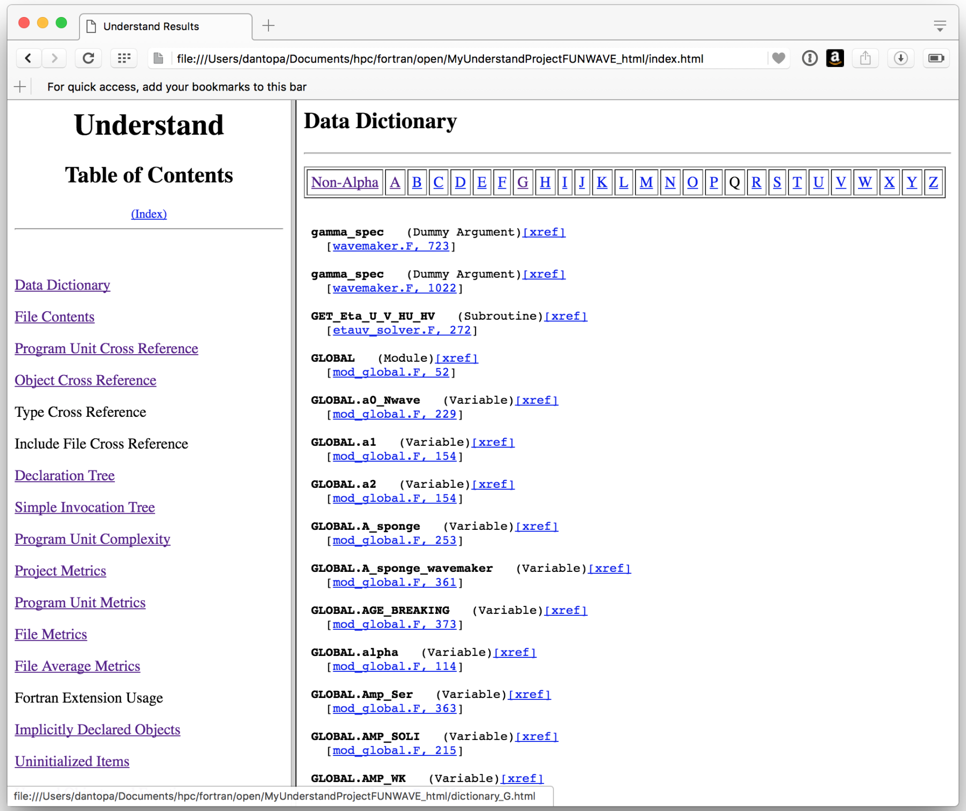
Source Code Analysis & Metrics

Project Metrics

Files:	28
Program Units:	97
Lines:	15824
Blank Lines:	2015
Code Lines:	9015
Comment Lines:	4983
Statements:	4131

Scientific Toolworks, Inc.
support@scitools.com
http://www.scitools.com

Figure 1: The splash page of the summary report.


Understand Results
file:///Users/dantopa/Documents/hpc/fortran/open/MyUnderstandProjectFUNWAVE_html/index.html
For quick access, add your bookmarks to this bar

Understand

Table of Contents

(Index)

Data Dictionary
File Contents
Program Unit Cross Reference
Object Cross Reference
Type Cross Reference
Include File Cross Reference
Declaration Tree
Simple Invocation Tree
Program Unit Complexity
Project Metrics
Program Unit Metrics
File Metrics
File Average Metrics
Fortran Extension Usage
Implicitly Declared Objects
Uninitialized Items

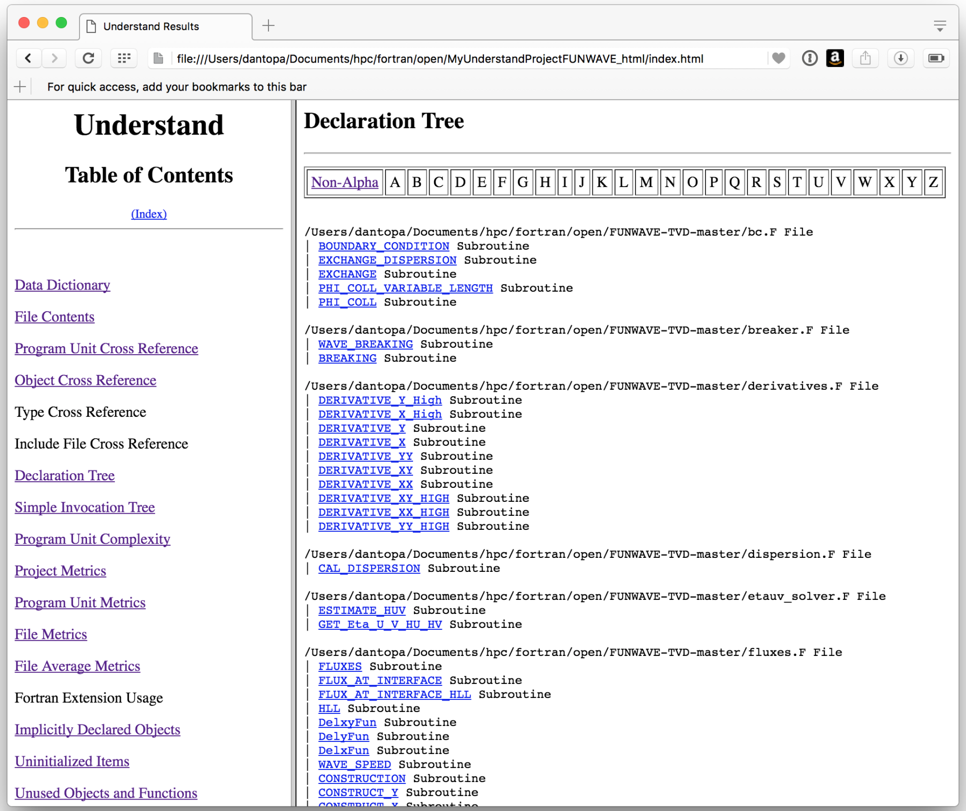
Data Dictionary

Non-Alpha | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z

```
gamma_spec   (Dummy Argument)[xref]
  [wavemaker.F, 723]

gamma_spec   (Dummy Argument)[xref]
  [wavemaker.F, 1022]

GET_Eta_U_V_HU_HV   (Subroutine)[xref]
  [etauv_solver.F, 272]

GLOBAL   (Module)[xref]
  [mod_global.F, 52]

GLOBAL.a0_Nwave   (Variable)[xref]
  [mod_global.F, 229]

GLOBAL.a1   (Variable)[xref]
  [mod_global.F, 154]

GLOBAL.a2   (Variable)[xref]
  [mod_global.F, 154]

GLOBAL.A_sponge   (Variable)[xref]
  [mod_global.F, 253]

GLOBAL.A_sponge_wavemaker   (Variable)[xref]
  [mod_global.F, 361]

GLOBAL.AGE_BREAKING   (Variable)[xref]
  [mod_global.F, 373]

GLOBAL.alpha   (Variable)[xref]
  [mod_global.F, 114]

GLOBAL.Amp_Ser   (Variable)[xref]
  [mod_global.F, 363]

GLOBAL.AMP_SOLI   (Variable)[xref]
  [mod_global.F, 215]

GLOBAL.AMP_WK   (Variable)[xref]
```

file:///Users/dantopa/Documents/hpc/fortran/open/MyUnderstandProjectFUNWAVE_html/dictionary_G.html

Figure 2: The data dictionary characterizes the variables by showing the type and pointing to the declaration.


Understand Results
file:///Users/dantopa/Documents/hpc/fortran/open/MyUnderstandProjectFUNWAVE_html/index.html
For quick access, add your bookmarks to this bar

Understand

Table of Contents

(Index)

Data Dictionary

File Contents

Program Unit Cross Reference

Object Cross Reference

Type Cross Reference

Include File Cross Reference

Declaration Tree

Simple Invocation Tree

Program Unit Complexity

Project Metrics

Program Unit Metrics

File Metrics

File Average Metrics

Fortran Extension Usage

Implicitly Declared Objects

Uninitialized Items

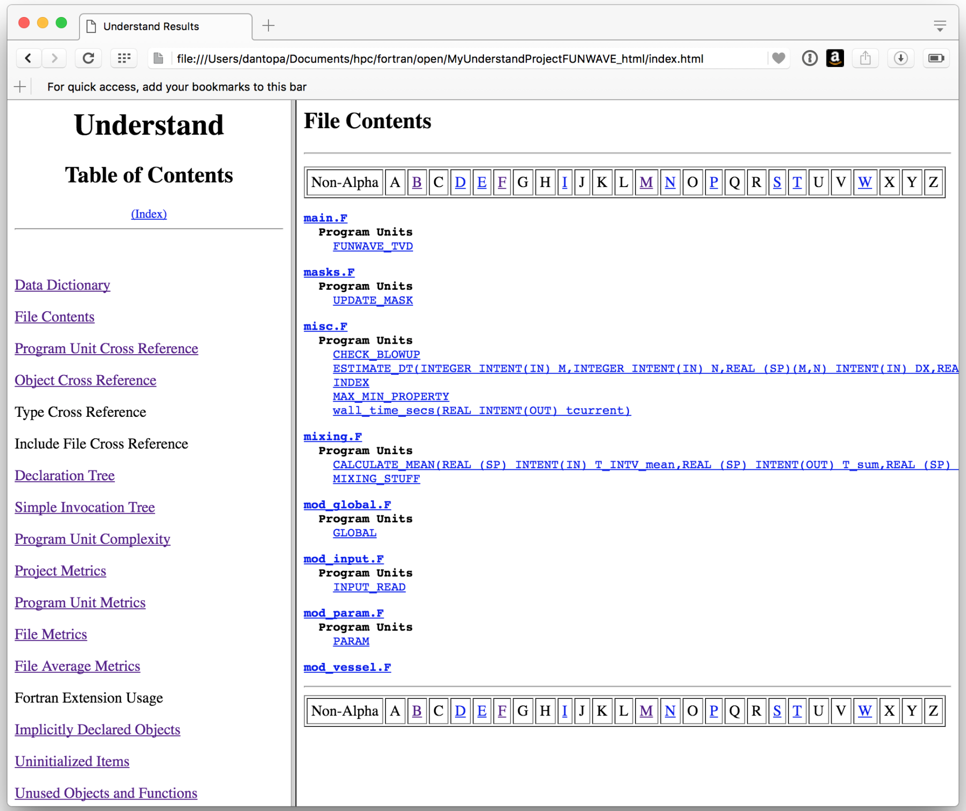
Unused Objects and Functions

Declaration Tree

Non-Alpha | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z

```
/Users/dantopa/Documents/hpc/fortran/open/FUNWAVE-TVD-master/bc.F File
| BOUNDARY_CONDITION Subroutine
| EXCHANGE_DISPERSION Subroutine
| EXCHANGE Subroutine
| PHI_COLL_VARIABLE_LENGTH Subroutine
| PHI_COLL Subroutine

/Users/dantopa/Documents/hpc/fortran/open/FUNWAVE-TVD-master/breaker.F File
| WAVE_BREAKING Subroutine
| BREAKING Subroutine

/Users/dantopa/Documents/hpc/fortran/open/FUNWAVE-TVD-master/derivatives.F File
| DERIVATIVE_Y_High Subroutine
| DERIVATIVE_X_High Subroutine
| DERIVATIVE_Y Subroutine
| DERIVATIVE_X Subroutine
| DERIVATIVE_YY Subroutine
| DERIVATIVE_XY Subroutine
| DERIVATIVE_XX Subroutine
| DERIVATIVE_XY_HIGH Subroutine
| DERIVATIVE_XX_HIGH Subroutine
| DERIVATIVE_YY_HIGH Subroutine

/Users/dantopa/Documents/hpc/fortran/open/FUNWAVE-TVD-master/dispersion.F File
| CAL_DISPERSION Subroutine

/Users/dantopa/Documents/hpc/fortran/open/FUNWAVE-TVD-master/etauv_solver.F File
| ESTIMATE_HUV Subroutine
| GET_Eta_U_V_HU_HV Subroutine

/Users/dantopa/Documents/hpc/fortran/open/FUNWAVE-TVD-master/fluxes.F File
| FLUXES Subroutine
| FLUX_AT_INTERFACE Subroutine
| FLUX_AT_INTERFACE_HLL Subroutine
| HLL Subroutine
| DelxyFun Subroutine
| DelyFun Subroutine
| DelxFun Subroutine
| WAVE_SPEED Subroutine
| CONSTRUCTION Subroutine
| CONSTRUCT_Y Subroutine
```

Figure 3: The declaration tree sweeps through each file and lists its methods: subroutines and functions.



Understand Results

file:///Users/dantopa/Documents/hpc/fortran/open/MyUnderstandProjectFUNWAVE_html/index.html

For quick access, add your bookmarks to this bar

Understand

Table of Contents

(Index)

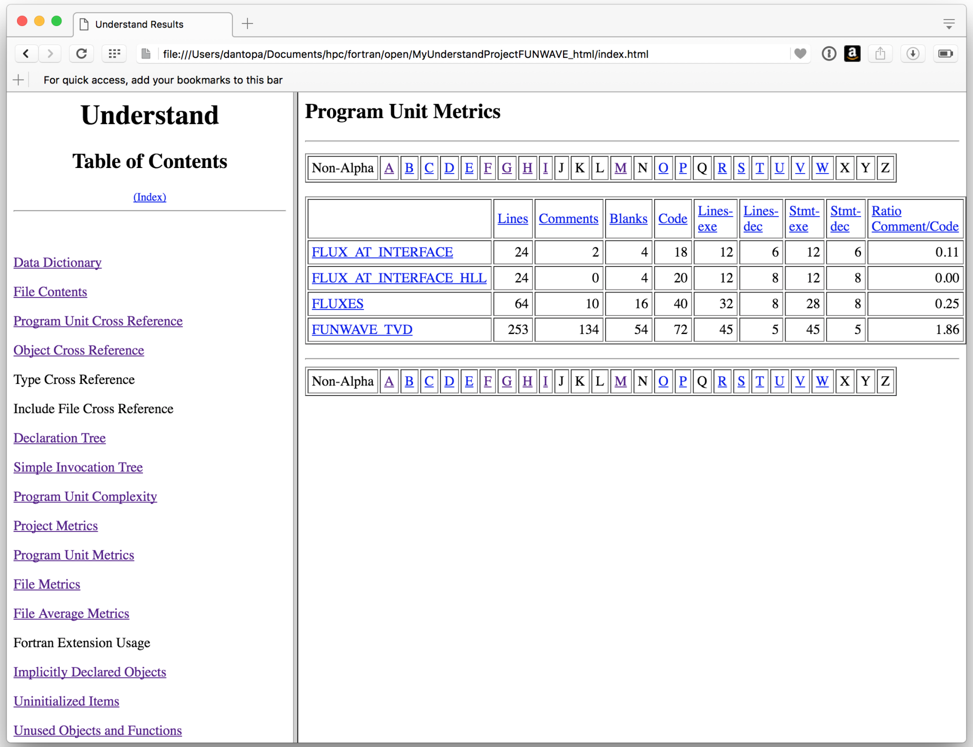
File Contents

Non-Alpha	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z

```
main.F
  Program Units
    FUNWAVE_TVD

masks.F
  Program Units
    UPDATE_MASK

misc.F
  Program Units
    CHECK_BLOWUP
    ESTIMATE_DT(INTEGER INTENT(IN) M,INTEGER INTENT(IN) N,REAL (SP)(M,N) INTENT(IN) DX,REA
    INDEX
    MAX_MIN_PROPERTY
    wall_time_secs(REAL INTENT(OUT) tcurrent)

mixing.F
  Program Units
    CALCULATE_MEAN(REAL (SP) INTENT(IN) T_INTV_mean,REAL (SP) INTENT(OUT) T_sum,REAL (SP)
    MIXING_STUFF

mod_global.F
  Program Units
    GLOBAL

mod_input.F
  Program Units
    INPUT_READ

mod_param.F
  Program Units
    PARAM

mod_vessel.F
```

Non-Alpha	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z

Figure 4: The file contents page allows the user to access each file and view the program units contained within.



Understand Results

file:///Users/dantopa/Documents/hpc/fortran/open/MyUnderstandProjectFUNWAVE_html/index.html

For quick access, add your bookmarks to this bar

Understand

Table of Contents

(Index)

Data Dictionary
File Contents
Program Unit Cross Reference
Object Cross Reference
Type Cross Reference
Include File Cross Reference
Declaration Tree
Simple Invocation Tree
Program Unit Complexity
Project Metrics
Program Unit Metrics
File Metrics
File Average Metrics
Fortran Extension Usage
Implicitly Declared Objects
Uninitialized Items
Unused Objects and Functions

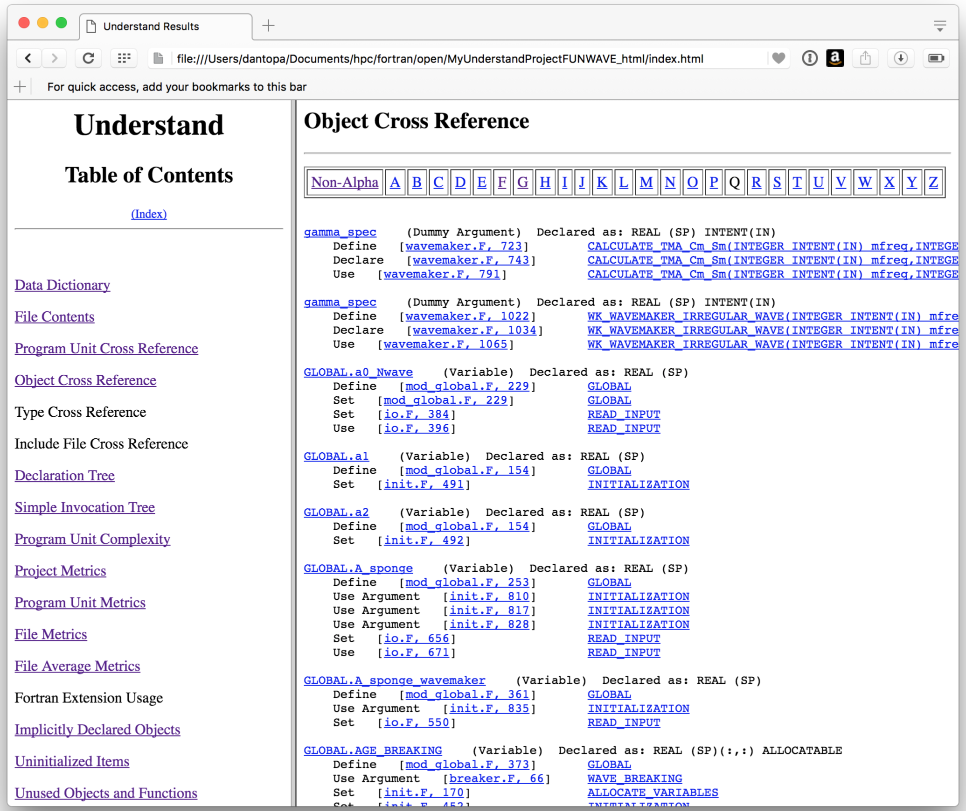
Program Unit Metrics

Non-Alpha A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

	Lines	Comments	Blanks	Code	Lines-exe	Lines-dec	Stmt-exe	Stmt-dec	Ratio Comment/Code
FLUX_AT_INTERFACE	24	2	4	18	12	6	12	6	0.11
FLUX_AT_INTERFACE_HLL	24	0	4	20	12	8	12	8	0.00
FLUXES	64	10	16	40	32	8	28	8	0.25
FUNWAVE_TVD	253	134	54	72	45	5	45	5	1.86

Non-Alpha A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Figure 5: The program unit metrics provides line counts for code and comments for each routine.



Understand Results

file:///Users/dantopa/Documents/hpc/fortran/open/MyUnderstandProjectFUNWAVE_html/index.html

For quick access, add your bookmarks to this bar

Understand

Table of Contents

(Index)

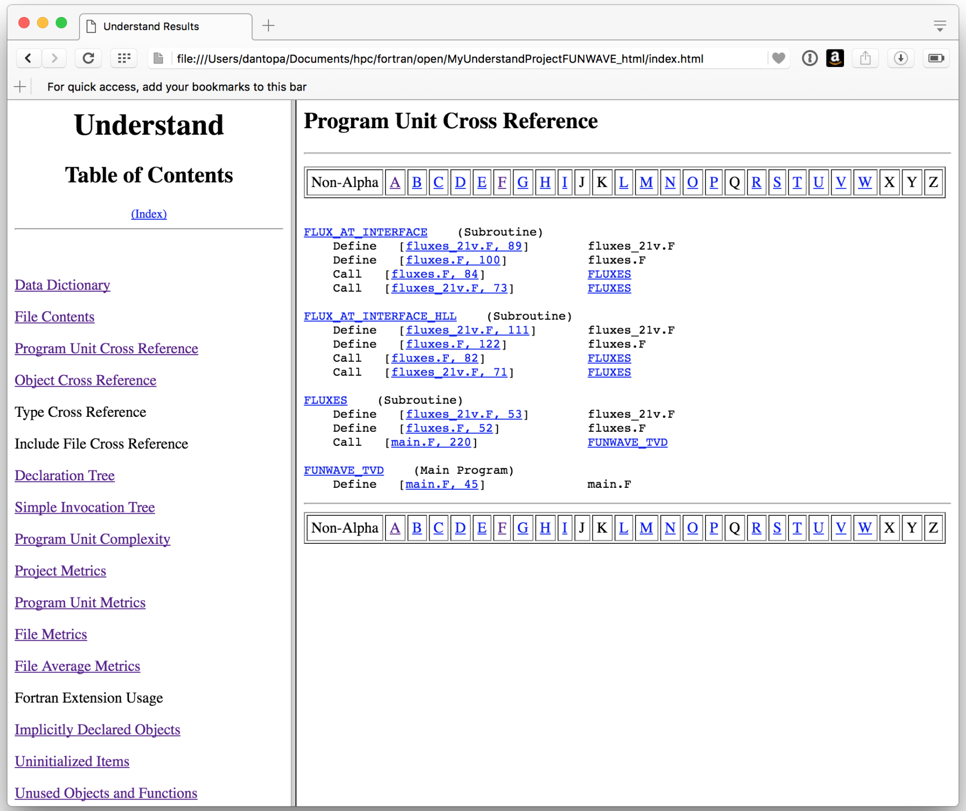
Data Dictionary
File Contents
Program Unit Cross Reference
Object Cross Reference
Type Cross Reference
Include File Cross Reference
Declaration Tree
Simple Invocation Tree
Program Unit Complexity
Project Metrics
Program Unit Metrics
File Metrics
File Average Metrics
Fortran Extension Usage
Implicitly Declared Objects
Uninitialized Items
Unused Objects and Functions

Object Cross Reference

Non-Alpha | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z

```
gamma_spec    (Dummy Argument)  Declared as: REAL (SP) INTENT(IN)
    Define   [wavemaker.F, 723]         CALCULATE_TMA_Cm_Sm(INTEGER INTENT(IN) mfreq,INTEGE
    Declare   [wavemaker.F, 743]        CALCULATE_TMA_Cm_Sm(INTEGER INTENT(IN) mfreq,INTEGE
    Use   [wavemaker.F, 791]            CALCULATE_TMA_Cm_Sm(INTEGER INTENT(IN) mfreq,INTEGE

gamma_spec    (Dummy Argument)  Declared as: REAL (SP) INTENT(IN)
    Define   [wavemaker.F, 1022]        WK_WAVEMAKER_IRREGULAR_WAVE(INTEGER INTENT(IN) mfre
    Declare   [wavemaker.F, 1034]       WK_WAVEMAKER_IRREGULAR_WAVE(INTEGER INTENT(IN) mfre
    Use   [wavemaker.F, 1065]           WK_WAVEMAKER_IRREGULAR_WAVE(INTEGER INTENT(IN) mfre

GLOBAL.a0_Nwave    (Variable)  Declared as: REAL (SP)
    Define   [mod_global.F, 229]        GLOBAL
    Set   [mod_global.F, 229]           GLOBAL
    Set   [io.F, 384]                   READ_INPUT
    Use   [io.F, 396]                   READ_INPUT

GLOBAL.a1    (Variable)  Declared as: REAL (SP)
    Define   [mod_global.F, 154]        GLOBAL
    Set   [init.F, 491]                 INITIALIZATION

GLOBAL.a2    (Variable)  Declared as: REAL (SP)
    Define   [mod_global.F, 154]        GLOBAL
    Set   [init.F, 492]                 INITIALIZATION

GLOBAL.A_sponge    (Variable)  Declared as: REAL (SP)
    Define   [mod_global.F, 253]        GLOBAL
    Use Argument   [init.F, 810]        INITIALIZATION
    Use Argument   [init.F, 817]        INITIALIZATION
    Use Argument   [init.F, 828]        INITIALIZATION
    Set   [io.F, 656]                   READ_INPUT
    Use   [io.F, 671]                   READ_INPUT

GLOBAL.A_sponge_wavemaker    (Variable)  Declared as: REAL (SP)
    Define   [mod_global.F, 361]        GLOBAL
    Use Argument   [init.F, 835]        INITIALIZATION
    Set   [io.F, 550]                   READ_INPUT

GLOBAL.AGE_BREAKING    (Variable)  Declared as: REAL (SP)(:,:) ALLOCATABLE
    Define   [mod_global.F, 373]        GLOBAL
    Use Argument   [breaker.F, 66]      WAVE_BREAKING
    Set   [init.F, 170]                 ALLOCATE_VARIABLES
```

Figure 6: The object cross reference page displays where variables are declared, defined and used.


Understand Results
file:///Users/dantopa/Documents/hpc/fortran/open/MyUnderstandProjectFUNWAVE_html/index.html
For quick access, add your bookmarks to this bar

Understand

Table of Contents

(Index)

Program Unit Cross Reference

Non-Alpha	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z

```
FLUX_AT_INTERFACE    (Subroutine)
    Define   [fluxes_21v.F, 89]        fluxes_21v.F
    Define   [fluxes.F, 100]           fluxes.F
    Call   [fluxes.F, 84]              FLUXES
    Call   [fluxes_21v.F, 73]          FLUXES

FLUX_AT_INTERFACE_HLL    (Subroutine)
    Define   [fluxes_21v.F, 111]       fluxes_21v.F
    Define   [fluxes.F, 122]           fluxes.F
    Call   [fluxes.F, 82]              FLUXES
    Call   [fluxes_21v.F, 71]          FLUXES

FLUXES    (Subroutine)
    Define   [fluxes_21v.F, 53]        fluxes_21v.F
    Define   [fluxes.F, 52]            fluxes.F
    Call   [main.F, 220]               FUNWAVE_TVD

FUNWAVE_TVD    (Main Program)
    Define   [main.F, 45]              main.F
```

Non-Alpha	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z

Figure 7: The program unit cross reference sweeps through all function and subroutine calls, listing the line number and the call arguments.

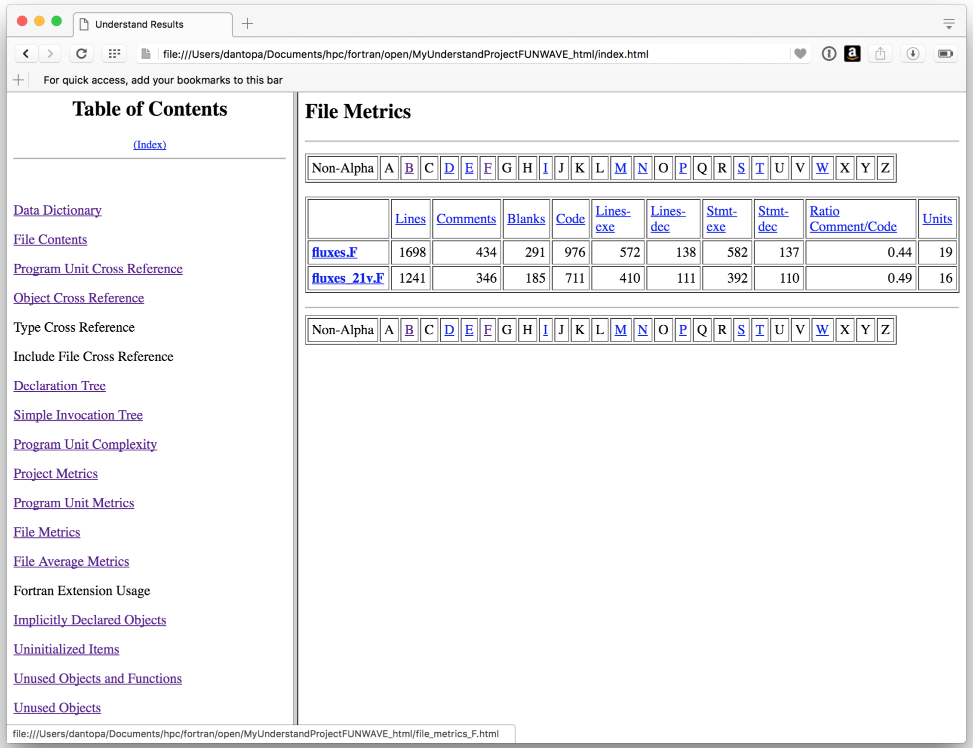

Understand Results
file:///Users/dantopa/Documents/hpc/fortran/open/MyUnderstandProjectFUNWAVE_html/index.html
For quick access, add your bookmarks to this bar

Table of Contents

(Index)

File Metrics

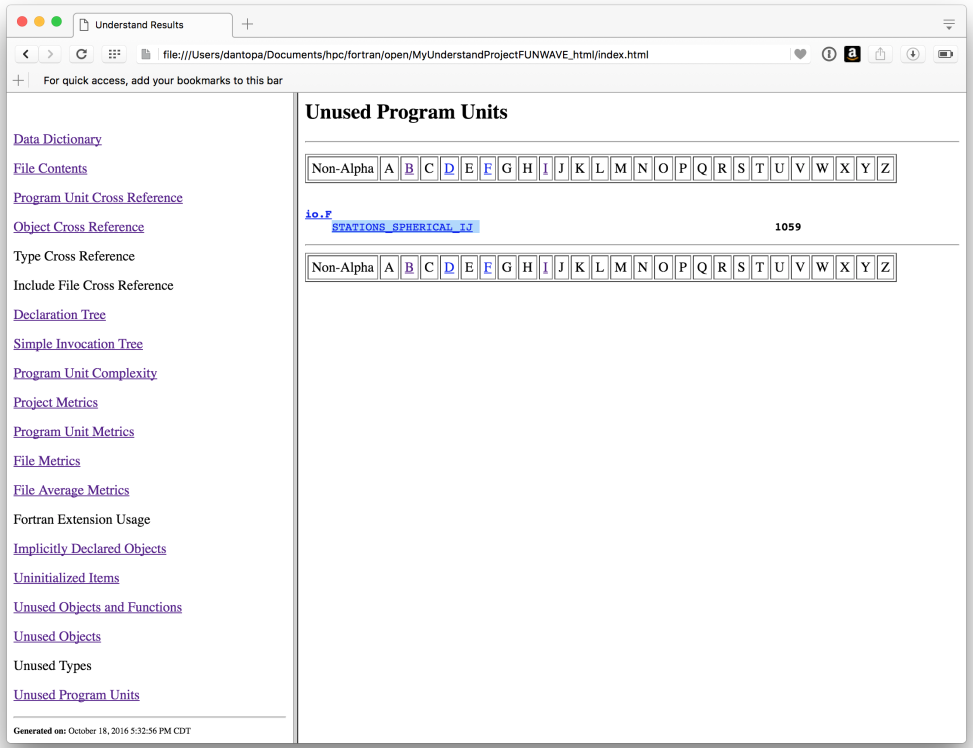
Non-Alpha	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z

	Lines	Comments	Blanks	Code	Lines-exe	Lines-dec	Stmt-exe	Stmt-dec	Ratio Comment/Code	Units
fluxes.F	1698	434	291	976	572	138	582	137	0.44	19
fluxes_21v.F	1241	346	185	711	410	111	392	110	0.49	16

Non-Alpha	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z

file:///Users/dantopa/Documents/hpc/fortran/open/MyUnderstandProjectFUNWAVE_html/file_metrics_F.html

Figure 8: The file metrics page summarizes the lines count in terms of code and comments.


Understand Results

file:///Users/dantopa/Documents/hpc/fortran/open/MyUnderstandProjectFUNWAVE_html/index.html

For quick access, add your bookmarks to this bar

Data Dictionary
File Contents
Program Unit Cross Reference
Object Cross Reference
Type Cross Reference
Include File Cross Reference
Declaration Tree
Simple Invocation Tree
Program Unit Complexity
Project Metrics
Program Unit Metrics
File Metrics
File Average Metrics
Fortran Extension Usage
Implicitly Declared Objects
Uninitialized Items
Unused Objects and Functions
Unused Objects
Unused Types
Unused Program Units

Generated on: October 18, 2016 5:32:56 PM CDT

Unused Program Units

Non-Alpha	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z

```
io.F
    STATIONS_SPHERICAL_IJ                              1059
```

Non-Alpha	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z

Figure 9: The unused program units page display code units which are not called.

3 Diagrams

Analysis of the FUNWAVE code produced over 770 diagrams. A brief survey follows.

The primary chart, figure 10, displays the main dependencies showing the plumbing connecting the files.

1. Butterfly plot: traces call chain to a function 11
2. Called by plot: shows immediate call to a function 12
3. Calls by plot: shows immediate calls by a routine 13
4. Cluster calls by plot: shows where a program unit is defined and which routines have access 14
5. Cluster control flow by plot: enhanced flow chart 15
6. Control flow by plot: classic flow chart 16
7. Declaration plot: upstream and downstream dependencies 17
8. Declaration file plot: dependent sub units 18

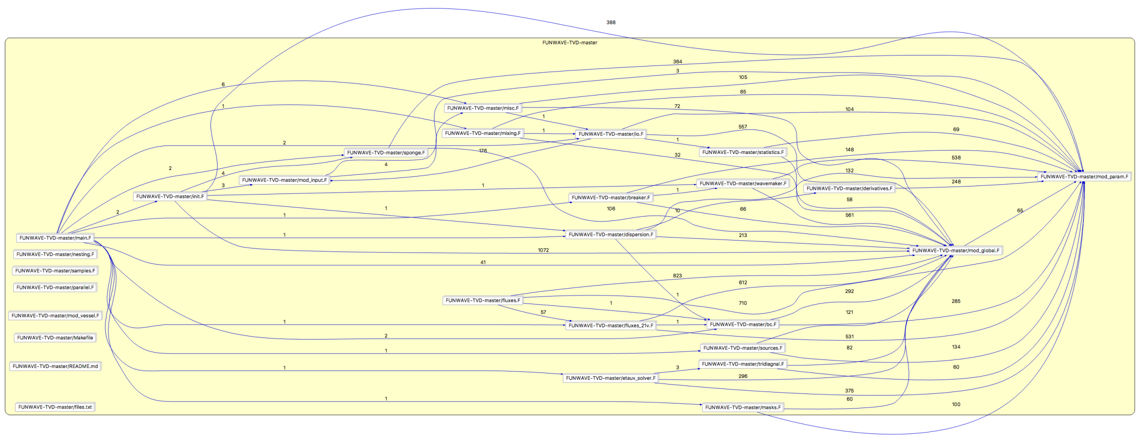

FUNWAVE-TVD-master
FUNWAVE-TVD-master/main.F
FUNWAVE-TVD-master/nesting.F
FUNWAVE-TVD-master/samples.F
FUNWAVE-TVD-master/parallel.F
FUNWAVE-TVD-master/mod_vessel.F
FUNWAVE-TVD-master/Makefile
FUNWAVE-TVD-master/README.md
FUNWAVE-TVD-master/files.txt
FUNWAVE-TVD-master/init.F
FUNWAVE-TVD-master/mod_input.F
FUNWAVE-TVD-master/sponge.F
FUNWAVE-TVD-master/misc.F
FUNWAVE-TVD-master/mixing.F
FUNWAVE-TVD-master/io.F
FUNWAVE-TVD-master/statistics.F
FUNWAVE-TVD-master/wavemaker.F
FUNWAVE-TVD-master/breaker.F
FUNWAVE-TVD-master/derivatives.F
FUNWAVE-TVD-master/dispersion.F
FUNWAVE-TVD-master/mod_global.F
FUNWAVE-TVD-master/mod_param.F
FUNWAVE-TVD-master/fluxes.F
FUNWAVE-TVD-master/fluxes_21v.F
FUNWAVE-TVD-master/bc.F
FUNWAVE-TVD-master/sources.F
FUNWAVE-TVD-master/tridiagnal.F
FUNWAVE-TVD-master/etauv_solver.F
FUNWAVE-TVD-master/masks.F
388
364
3
105
85
72
104
557
69
148
538
132
248
58
561
66
106
10
213
1072
41
823
612
292
710
285
121
531
82
134
296
60
375
60
100
65
176
32
57
6
2
4
3
1

Figure 10: Internal dependencies between the files.

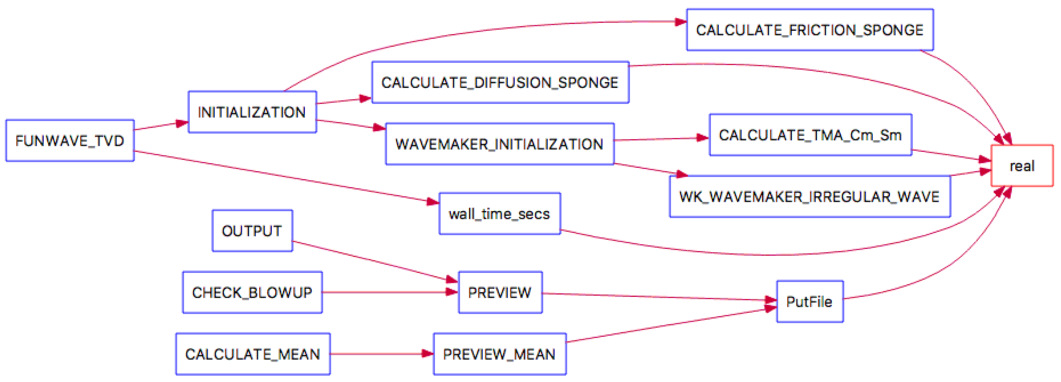
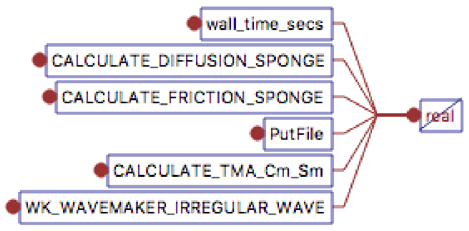


Figure 11: The butterfly plot traces call chain to a function, here the function REAL.



wall_time_secs
CALCULATE_DIFFUSION_SPONGE
CALCULATE_FRICTION_SPONGE
PutFile
CALCULATE_TMA_Cm_Sm
WK_WAVEMAKER_IRREGULAR_WAVE
real

Figure 12: The called by plot shows the immediate routines to call a function, here the function REAL.

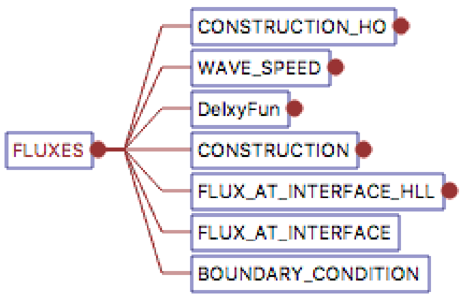


Figure 13: The calls by plot shows functions and subroutines called by a program unit, here the function FLUXES.

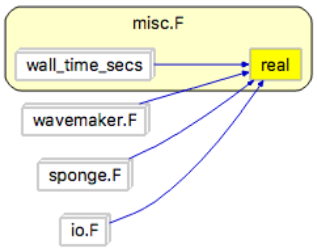


Figure 14: The cluster calls by plot shows where a program unit is defined and which routines have access, here the function REAL.

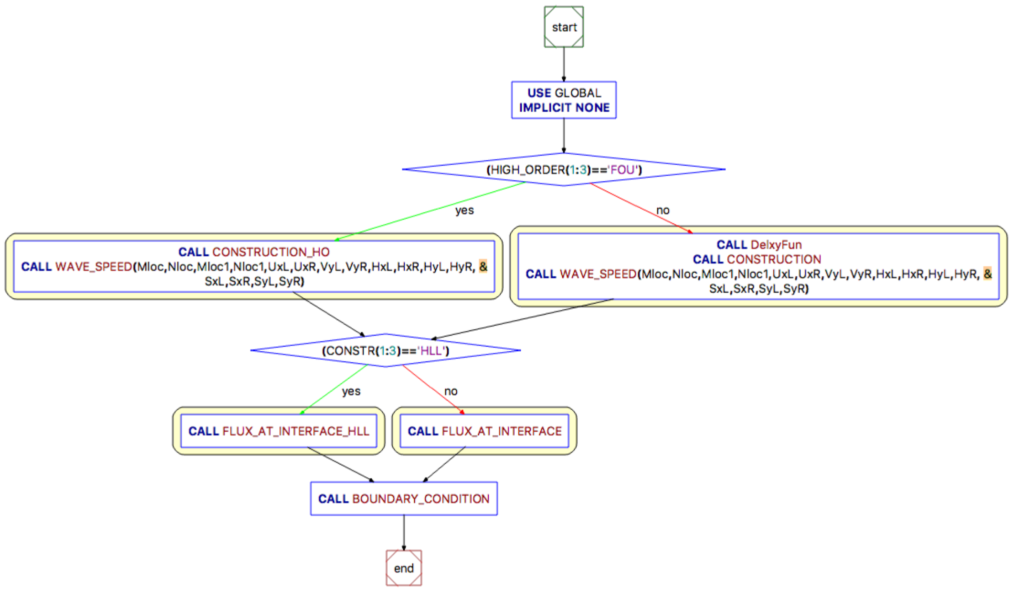


Figure 15: The cluster control flow plot presents an enhanced flow chart for each program unit, here FLUXES.

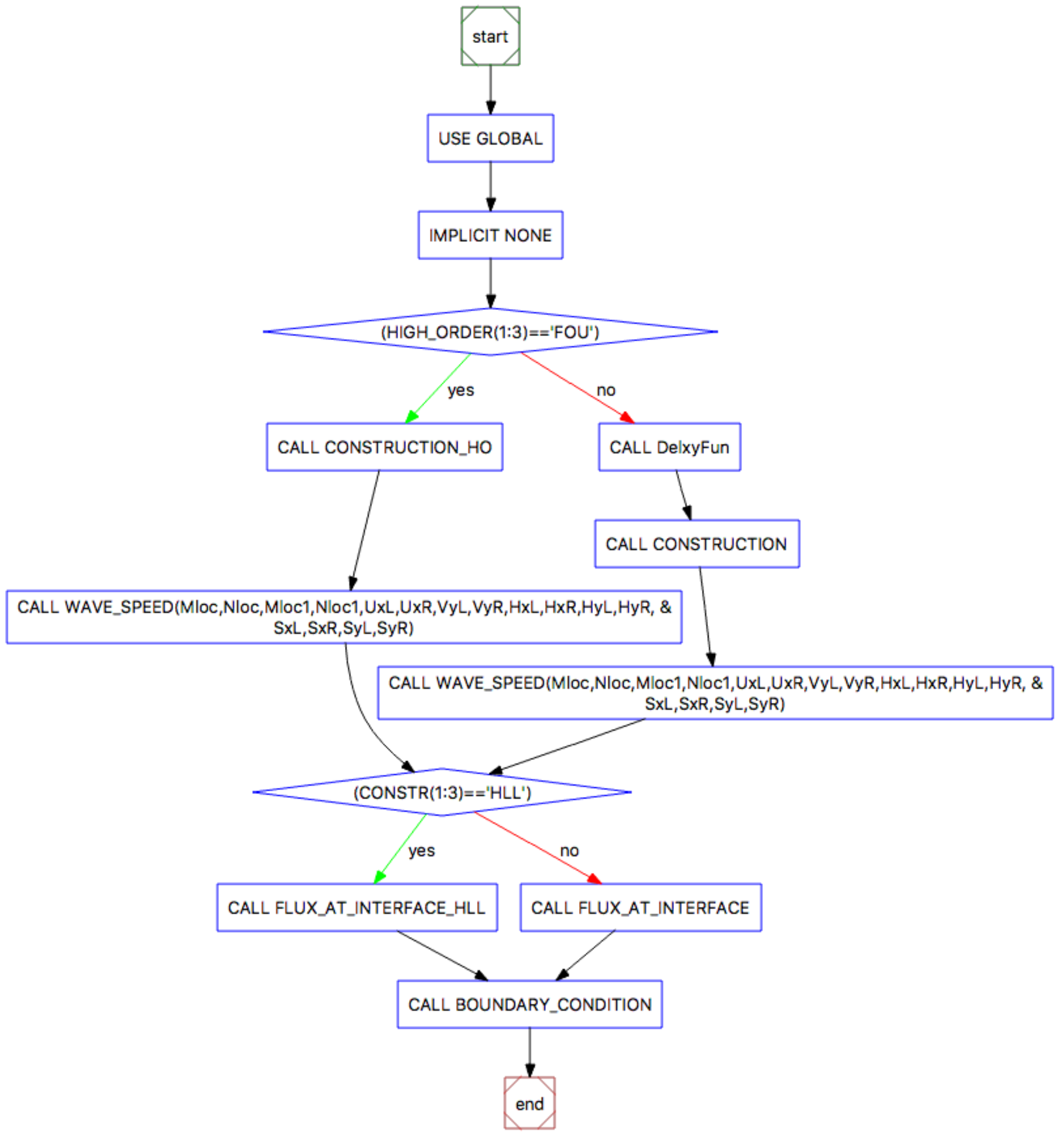

start
USE GLOBAL
IMPLICIT NONE
(HIGH_ORDER(1:3)=='FOU')
yes
no
CALL CONSTRUCTION_HO
CALL DelxyFun
CALL CONSTRUCTION
CALL WAVE_SPEED(Mloc,Nloc,Mloc1,Nloc1,UxL,UxR,VyL,VyR,HxL,HxR,HyL,HyR, & SxL,SxR,SyL,SyR)
CALL WAVE_SPEED(Mloc,Nloc,Mloc1,Nloc1,UxL,UxR,VyL,VyR,HxL,HxR,HyL,HyR, & SxL,SxR,SyL,SyR)
(CONSTR(1:3)=='HLL')
yes
no
CALL FLUX_AT_INTERFACE_HLL
CALL FLUX_AT_INTERFACE
CALL BOUNDARY_CONDITION
end

Figure 16: The control flow plot presents an enhanced flow chart for each program unit, here FLUXES.

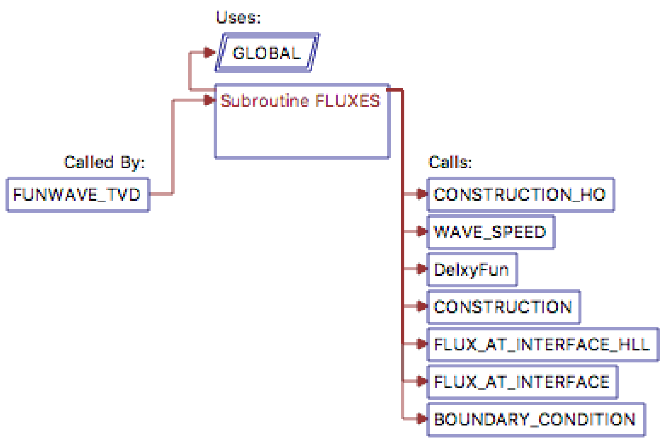


Figure 17: The declaration plot shows the upstream and downstream calls, here for FLUXES.

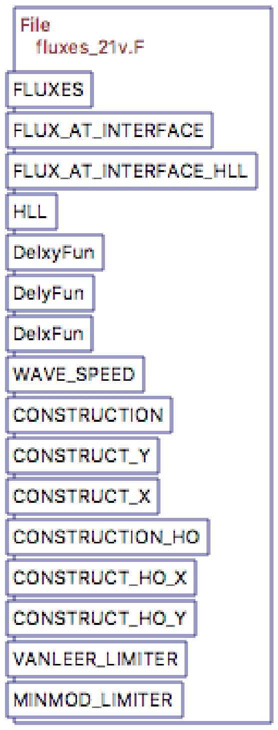


Figure 18: The declaration file plot shows the dependent program units, here for FLUXES.