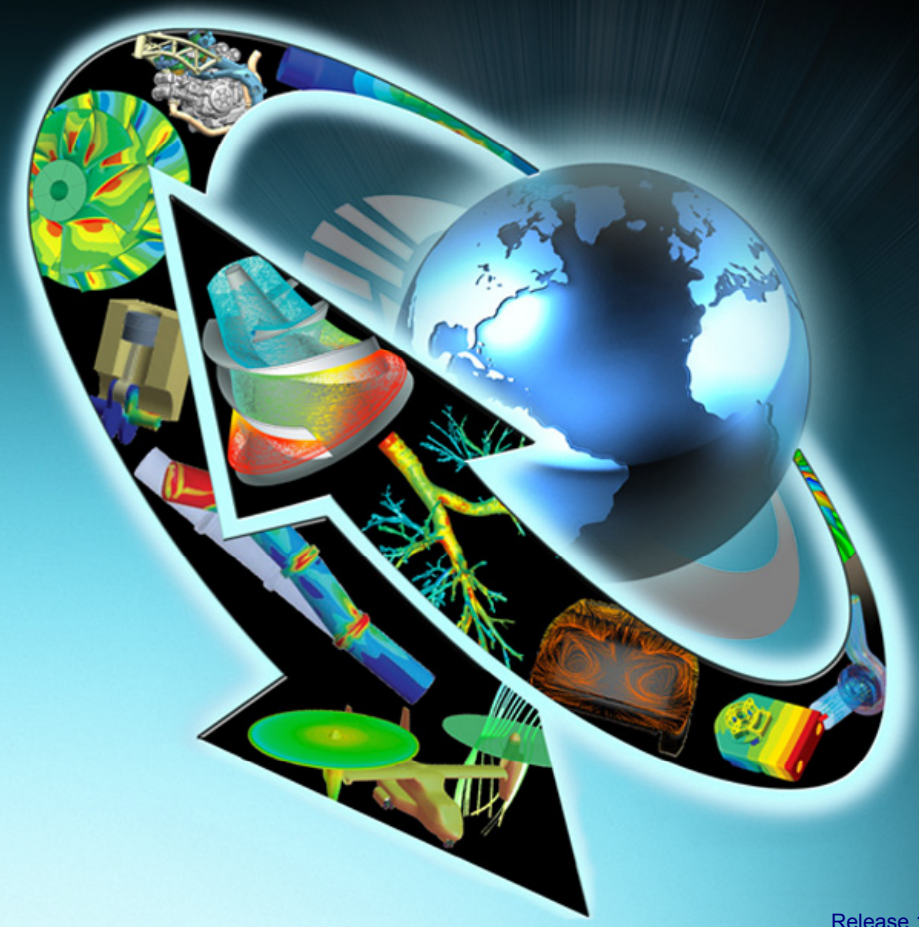
ANSYS®

Chapter 2

Compilation and Interpretation



Advanced FLUENT User-Defined Functions

How to use the UDFs

- First, we need to write and save the C-source code. The name need to have a ".c" extension (you can also supply your own header file(s))
- To use this file, the steps are:
  1: Compile or interpret the UDF
  2: Start the solver (FLUENT) and read in your case/data files
  3: Hook up the UDFs to the FLUENT case
  4: Run the calculation as usual

- Note:

  Values obtained from and returned to the solver by UDFs must be in SI units only

Interpreted Vs. Compiled Code

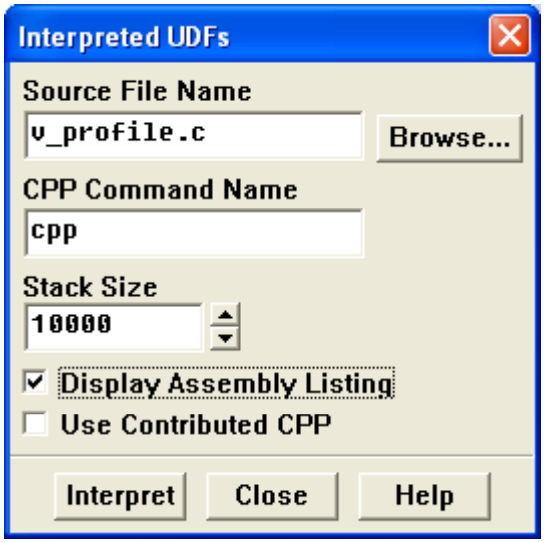
- UDFs can be 'interpreted' on-the-fly using the standard 'GUI'
  - does not need a separate compiler and are architecture-independent
  - It translates the C-source to assembly language
  - Executes the code on line-by-line instantaneously
    - performs slower than compiled UDFs
  - The interpreter resides in the computer's memory
    - involves extra memory usage
- UDFs can be precompiled before invoking in FLUENT
  - Needs a compiler (gcc or microsoft's .NET C++ compiler)
  - It translates the C-source to machine language (object modules)
  - Creates 'shared libraries' linked with the rest of the solver

Interpreted UDFs

- Interpreter limitations:
  - mixed mode arithmetic
  - structure references etc.
  - cannot be linked to compiled system or user libraries
  - less powerful than compiled UDFs due to limitations in the C language supported by the interpreter
- In particular, interpreted UDFs cannot contain:
  - non ANSI-C prototypes for syntax
  - declarations of local structures, unions, pointers to functions, and arrays of functions
  - direct structure references
- Interpreted UDFs can indirectly access data stored in a FLUENT structure only via a set of macros
- Users are strongly encouraged to use compilation only

Interpreting the UDFs

- Define/User Defined/ Functions/Interpreted…



  - Click Interpret
  - The assembly language code will scroll past window

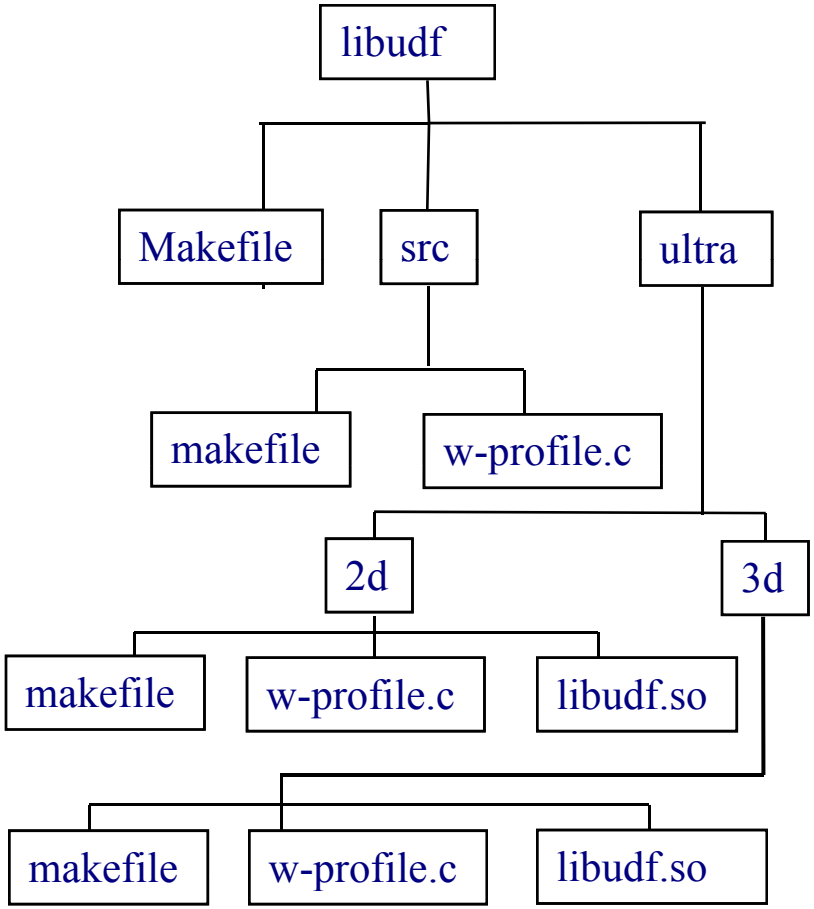
Listing appearing on Fluent windows:

```
w_profile:
    .local.pointer thread (r0)
    .local.int position (r1)
0   .local.end
0    save
    .local.int f (r6)
8    push.int 0
10   save
    .local.int.
    .
    .
    .
.L1:
132 restore
133 restore
134 ret.v
```

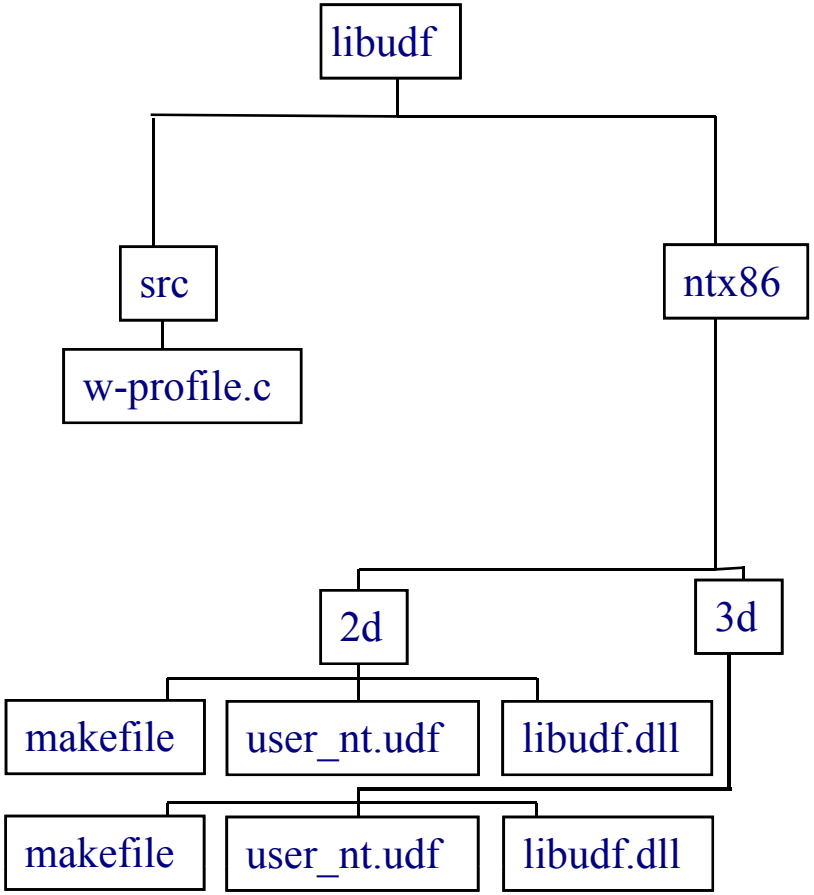
} Skipping display here

Compiled UDF Directory Structure

Unix Tree

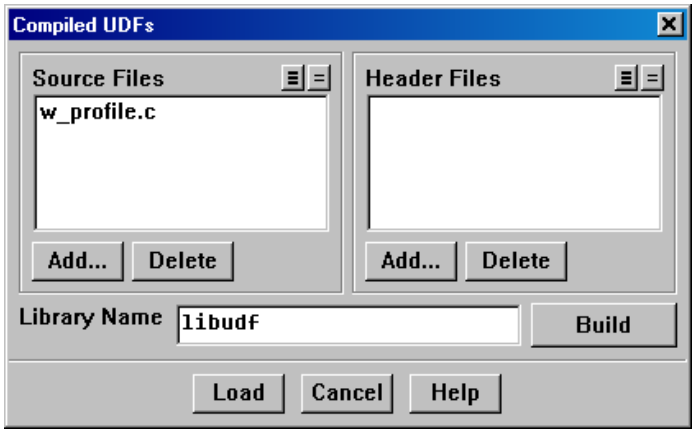


Windows Tree



UDF Compilation

- To compile UDFs from within Fluent, use:
  - Define/User_Defined/Functions/ Compile...
- Placing source routines in your working directory
- This GUI creates the directory structure below your working directory (where you have your case and data files)
- This GUI identifies the architecture as well as the version of fluent running and compiles only for the appropriate UDF version (2d/2ddp/3d/3ddp/or any parallel version)
- The library (directory) name can be modified by the user



- To unload a compiled UDF library, use
- `Define/User_Defined /Functions/Manage`

select the library, then click Unload button