ELMER User's Guide

ELMER User's Guide

CSC – Scientific Computing Ltd.

© CSC – Scientific Computing Ltd.
2000

1st edition

ISBN 952-XXXX-YY-Z

Printed at
FooBar
Helsinki 2000

http://www.csc.fi/elmer

No part of ELMER may be used, reproduced, stored or transmitted in any form or by any means without the written permission of CSC.

CSC assumes no responsibility or liability on any errors or inaccuracies in ELMER program or documentation. Any kind of material damages or other indirect consequences resulting from any ELMER part, documentation discrepancies and errors or non-anticipated program behavior are limited to the value of appropriate products purchased from CSC.

ELMER products are delivered to the customer at the 'as it is' state and revision level they are on the moment of purchasing, and are declared in more detail in the ELMER license agreement. User accepts and is obliged to follow the guidelines stated in the separate license agreement required in using any parts of ELMER.

This document is for informational use only. All information and specifications given have been carefully prepared by the best efforts of CSC, and are believed to be true and accurate as of time publishing. CSC reserves the right to modify ELMER and its documents without notice.

Additional software product copyrights included in ELMER: Sparse library copyright notice, Copyright (c) 1985,86,87,88 by Kenneth S. Kundert and the University of California, Department of Electrical Engineering and Computer Sciences University of California, Berkeley, California 94720.

About this Manual

The intended audience for ELMER User's Guide are engineers and scientists in universities, research institutes and industry. The document aims at offering a detailed approach to the use of ELMER software. For the first-time user, having basic knowledge on mathematical modeling, numerical methods and computer science, it acts as a start-up. Experienced users can utilize the User's Guide in problem solving, teaching and training.

ELMER User's Guide provides information on ELMER graphical user interface and mathematical background of ELMER. It leads the user to create mathematical models, and control existing numerical and visualization tools.

The present ELMER User's Guide corresponds to ELMER software version 1.4 for Windows NT and Unix platforms.

ELMER User's Guide will be delivered via the World Wide Web. Latest documentations and program versions of ELMER are available at http://www.csc.fi/elmer.

ELMER Installation

ELMER software is delivered on CD-ROM with the installation instructions. After inserting the CD-ROM into the computer, the ELMER software will be automatically installed by the auto install program.

As user receives a new ELMER version update, all user applications must be carefully checked against any discrepancies between old and new versions. The software provider does not guarantee congruent results between different versions.

The operating system can be either Windows NT 4.0 or Unix. It is recommended that the user has a computer system containing 64 Mb memory at minimum. Installed ELMER version requires approximately 40Mb disk space.

Contents

1 Introduction **1**

- 1.1 ELMER Capabilities 1
- 1.2 ELMER Structure 2
- 1.3 ELMER Work flow 7

2 ELMER Front **8**

- 2.1 Input to ELMER Front 8
- 2.2 ELMER Front main window 12
- 2.3 General user interface features 14
- 2.4 Starting ELMER Front 18
- 2.5 ELMER Front settings file 19
- 2.6 File menu 19
- 2.7 Edit menu 22
- 2.8 Display menu 31
- 2.9 Problem menu 33
- 2.10 Model menu 48
- 2.11 Mesh menu 59
- 2.12 Solver menu 69
- 2.13 Run menu 72
- 2.14 Window menu 77
- 2.15 Help menu 78
- 2.16 Model window toolbar 80
- 2.17 Model Window 81
- 2.18 Control toolbar 82
- 2.19 Message area 83
- 2.20 Status area 83

3 ELMER Solver **85**

- 3.1 Overview 85
- 3.2 Capabilities 86
- 3.3 Structure of a Simulation 89
- 3.4 Structure of the Base FEM Code 91

4 The Post processor **94**

4.1 General Overview 94
4.2 Performance: 95
4.3 Element Types 96
4.4 Input Format 96

5 Walk-through example 97
5.1 Boiler: heating water in a kettle 97
5.2 Visualization with ELMER Post 116

Appendix 129

A Mathematical Background 129
A.1 The Governing Equations 129
A.2 The Boundary Conditions 133
A.3 Numerical Methods 136

B ELMER Geometry File format 138
B.1 Introduction 138
B.2 The format 140
B.3 An example file 143

C ELMER Front settings file format 145

D ELMER Mesh Generator input file format 148
D.1 Model description 148
D.2 Mesh density control 148
D.3 File structure 149
D.4 The header 150
D.5 The vertices 150
D.6 The edges 151
D.7 The bodies 152

E ELMER Mesh File Format 156
E.1 The header file format 156
E.2 The node file format 157
E.3 The element file format 157
E.4 The boundary element file format 158

F ELMER Solver Input File Format 159
F.1 Array Variables 159
F.2 Parameters Depending on Field Variables or Time . 160
F.3 User Defined Functions 161
F.4 The Format 162
F.5 Using the Advection-Diffusion Equation Solver . . . 179
F.6 An Example of Using the Input File 181

G ELMER Solver Output File Format 191

G.1 The Header 191
G.2 The Output Data 193

H ELMER Post File Format 194
H.1 The Header 195
H.2 The Mesh 195
H.3 The Solution Data 196
H.4 An example file 196

I ELMER Solver Element Types 197

Index 200

List of Figures

1.1 ELMER structure 3
1.2 ELMER Front 4
1.3 ELMER Solve 5
1.4 ELMER Post 6
1.5 ELMER workflow 7

2.1 Model files 12
2.2 ELMER Front main window 13
2.3 ELMER startup command under Microsoft Windows NT 18
2.4 File menu 20
2.5 Edit menu 22
2.6 Bodies window 23
2.7 Boundaries window 24
2.8 Settings window 28
2.9 Problem menu 34
2.10 Model name and directories window 34
2.11 Datafiles window 36
2.12 Coordinate settings window 37
2.13 Timestep settings window 39
2.14 Equations window 43
2.15 Advection-diffusion equation 46
2.16 Model menu 48
2.17 Body forces window 50
2.18 Materials window 51
2.19 Initial conditions window 52
2.20 Parameter selection example 53
2.21 Argument variable list 55
2.22 Boundary conditions window 57
2.23 Mesh element types 60
2.24 Mesh example 61
2.25 Define mesh window 62
2.26 Mesh structure window 64

2.27 Mesh density window 67
2.28 Mesh logfile example 68
2.29 Solver settings window 69
2.30 Advection-diffusion solver example 70
2.31 Processor settings window 72
2.32 Run menu 73
2.33 Process table 76
2.34 Window list example 77
2.35 Help menu 78
2.36 Graphics info window 78
2.37 System info window 79
2.38 Model window toolbar 80
2.39 Model window example 82
2.40 Status area 84

5.1 Boiler: model geometry 99
5.2 Boiler: boundary names 100
5.3 Boiler: model name and directories 101
5.4 Boiler: coordinate system 102
5.5 Boiler: timestep settings 103
5.6 Boiler: equations 105
5.7 Boiler: body force settings 108
5.8 Boiler: temperature table 110
5.9 Boiler: mesh definition 111
5.10 Boiler: mesh density settings 112
5.11 Boiler: mesh 113
5.12 Boiler: solver settings 114
5.13 Boiler: solver process table 115
5.14 ELMER Post User Interface 116
5.15 ELMER Post main window areas 117
5.16 Read Model File -window 119
5.17 Color Mesh Edit -window in default mode 122
5.18 ColorScale definition 124
5.19 Time Step Control window 125
5.20 Kettle Model at timestep 80 127
5.21 Graphics window background setup 128

F.1 Geometry of the Marangoni convection example. . . 182
F.2 Results of the Marangoni convection example. Isocontours of the temperature field and the velocity vectors. 190

I.1 The 3 (303) and 6 (306) node triangular elements. . 197
I.2 The 4 (404), and 8 (408) node quadrilateral elements. 198

I.3 The 4 (504) and 10 (510) node tetrahedron elements. 198
I.4 The 8 (808) and 20 (820) node brick elements. . . . 199

Chapter 1

Introduction

ELMER software development is a joint project between Finnish universities, research laboratories and industrial companies. The main objective is to create a finite element method based program frame, which can be applied and further developed by academic and industrial communities in research and development, and in education.

ELMER development was initialized in the Finnish national computational fluid dynamics (CFD) technology programme in 1995-1999, funded by Tekes, the Finnish technology agency. At the first stage, the most active collaborators were CSC (responsible organization in ELMER development), Helsinki University of Technology, Jyväskylä University, Okmetic Ltd., and Technical Research Centre of Finland, and the main application areas were closely related to semiconductor and paper machine industries.

Currently, CSC continues its coordinative role in ELMER development. Development is in active phase, and new application areas are continuously founded. These include for instance micro electro mechanical systems (MEMS), simulation of blood flow, casting problems and crystal growth in magnetic fields. Some parts of ELMER have even utilized in CAVE environment.

1.1 ELMER Capabilities

Since its creation, ELMER has progress from the ordinary CFD program towards a computational tool which can tackle multi-physics problems. ELMER solves problems like

- compressible and incompressible fluid flows
- non-Newtonian flows
- laminar and turbulent flows
- heat transfer by conduction, convection and radiation
- phase changes
- free surfaces
- stresses and displacements in elastic bodies
- vibration
- magnetohydrodynamics
- fluid-structure interactions

The open design of ELMER allows user to work on a simulation on a desktop on small scale before performing the full simulation on an appropriate machine. ELMER has

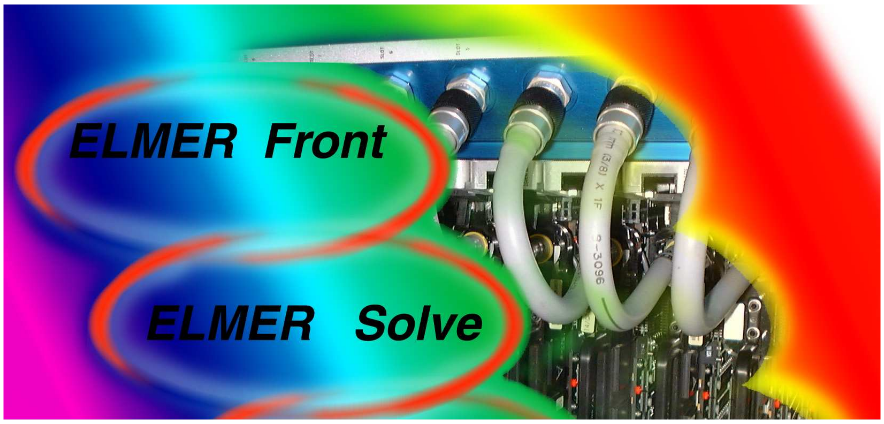
- open and flexible structure for user application development
- scalability to various computing platforms and operating system
- flexible and fast re-simulation with temporary values

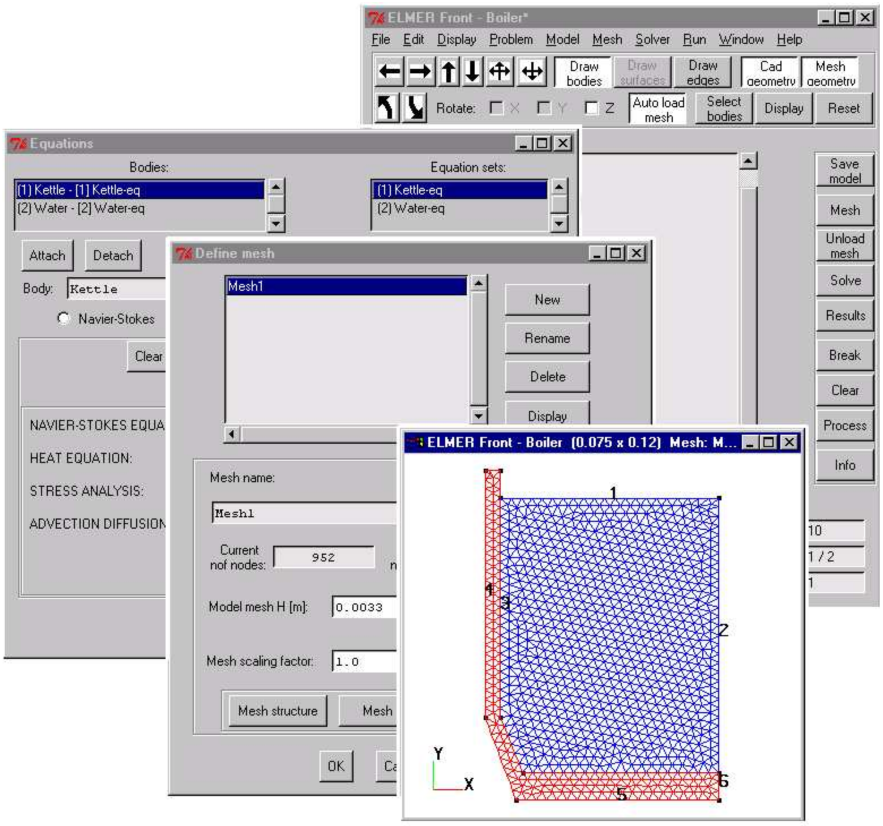
1.2 ELMER Structure

ELMER is a collection of programs rather than a single monolithic application. It consists of

- ELMER Front (preprocessor)
- ELMER Solver (solver based on finite elements)
- ELMER Post (postprocessor)

ELMER modules can be used as an integrated process or independently. Figure 1.1 depicts ELMER structure divided into three components.


ELMER Front
ELMER Solve



Bodies:
(1) Kettle - [1] Kettle-eq
(2) Water - [2] Water-eq

Equation sets:
(1) Kettle-eq
(2) Water-eq

Body: Kettle

Mesh1

Mesh name: Mesh1

Current nof nodes: 952

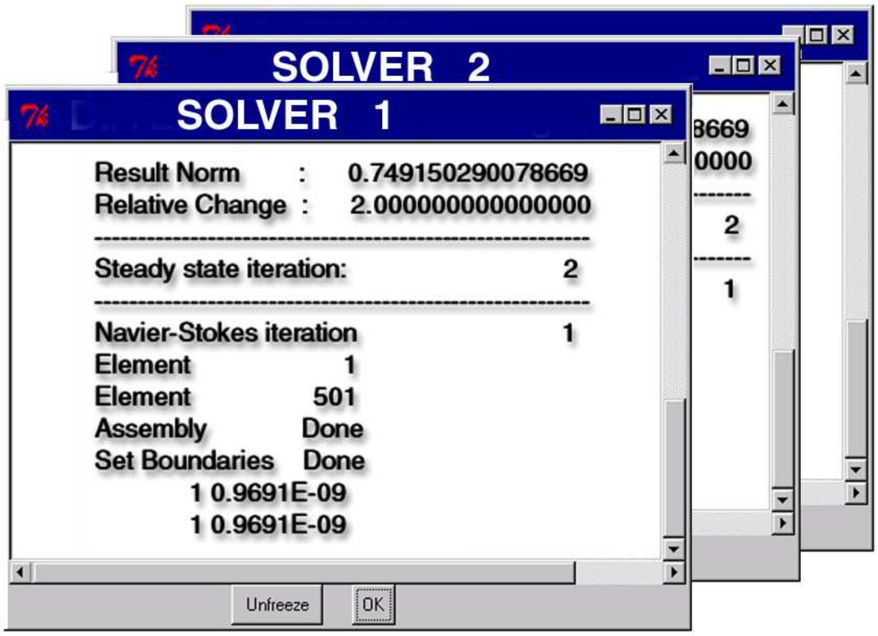
Model mesh H [m]: 0.0033

Mesh scaling factor: 1.0

10
1 / 2
1

Figure 1.2: *ELMER Front*

ELMER Solver processes mathematical models into discrete form, handles coupled systems, nonlinearities and time-dependencies, and provides output data for ELMER Post.



```
Result Norm     :    0.749150290078669
Relative Change :    2.000000000000000
-----------------------------------------------------
Steady state iteration:                            2
-----------------------------------------------------
Navier-Stokes iteration                            1
Element              1
Element            501
Assembly          Done
Set Boundaries    Done
           1 0.9691E-09
           1 0.9691E-09
```

Figure 1.3: *ELMER Solve*

The function of ELMER Post is to visualize the numerical results produced by the solver. ELMER Post operates with the unknown variables and shows contours, vector fields, animations etc. on the screen. It can also manipulate computed data into another form (for instance heat fluxes from temperature distributions).

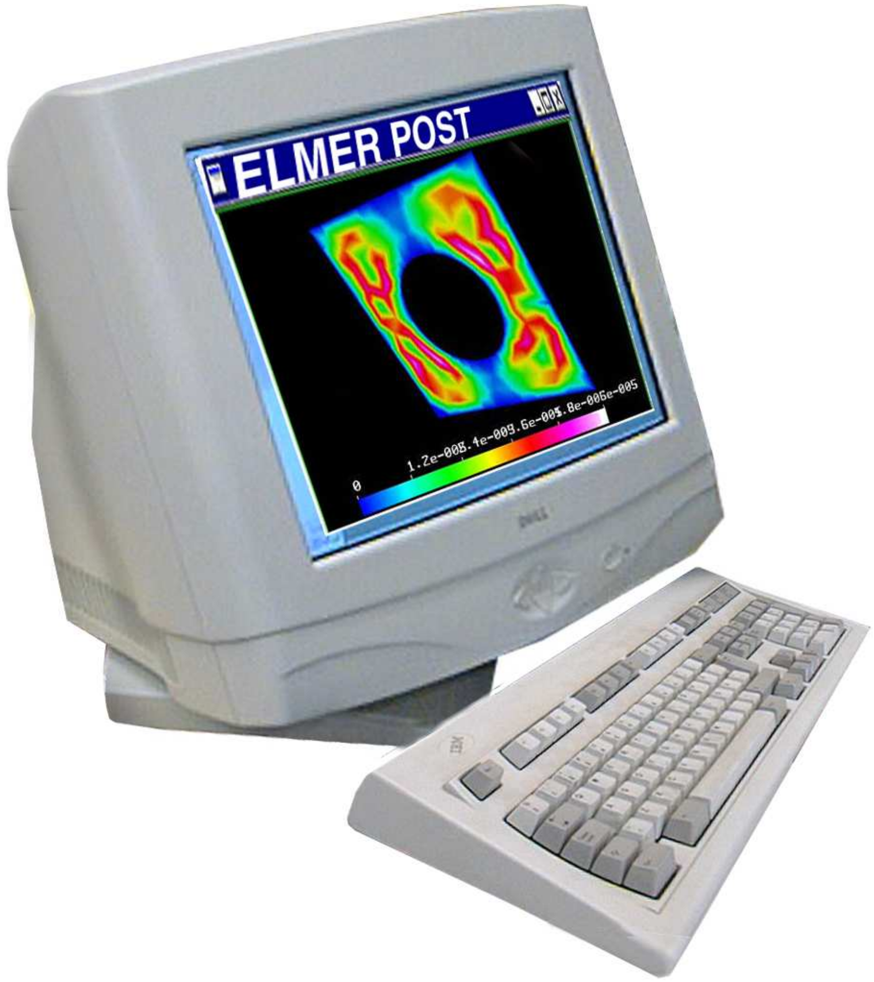


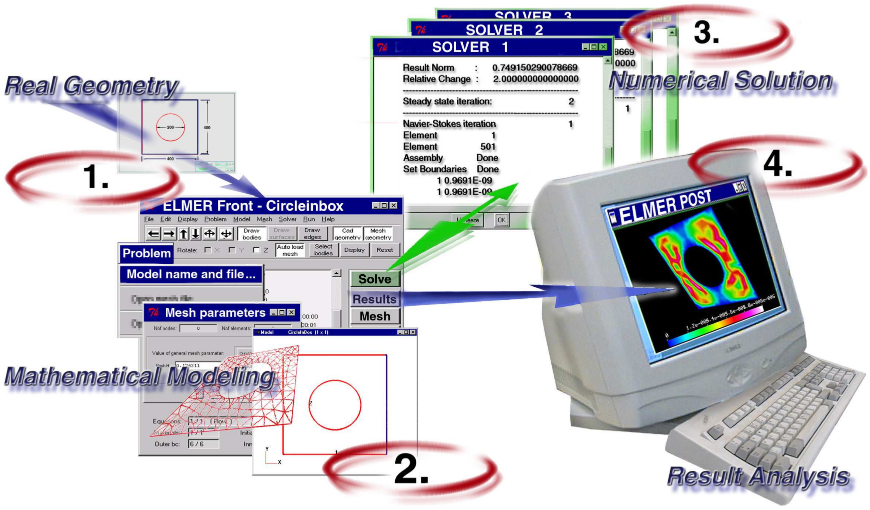
Figure 1.4: *ELMER Post*

Besides large variety of modeling, computing and visualization tools, ELMER also offers some special features, which may help in research work. These include, among others,

- interfaces to I-DEAS, ABAQUS and FLUENT
- MathC language for analyzing numerical results in ELMER Post
- computation of view factors and Gebhardt factors (modeling of radiation)

1.3 ELMER Work flow

The user can start ELMER software by clicking the ELMER icon on the computer's desktop, or writing the starting command on the command line. Then, ELMER work flow proceeds as shown in Figure 1.5.



Result Norm : 0.749150290078669
Relative Change : 2.000000000000000

Steady state iteration: 2

Navier-Stokes iteration 1

Figure 1.5: *ELMER workflow*

- As ELMER has been started, ELMER Front opens. User can now read in the file containing the model geometry **(1.)**, and start operating with the model shown on a separate Model window **(2.)**
- After all definitions (mathematical equations, mesh and material parameters,...) have been made, ELMER Solver can be started **(3.)**
- Solver output is used in ELMER Post visualization and results analysis **(4.)**

Workflow from **(1.)** to **(4.)** represents necessary/optional steps during a typical user session. Definitions in ELMER Front create a basis for numerical solution. The model can be solved in ELMER Solver after saving the model file, then by clicking the Solve button in ELMER Front. ELMER Post is started from the Results button.

NOTE! Notation from **(1.)** to **(4.)** in ELMER workflow is used as references later on in this manual.

Chapter 2

ELMER Front

ELMER Front is the Graphical User Interface module for ELMER. All essential problem solving definitions needed with other ELMER modules can be set in ELMER Front. First part of this chapter (sections 1.1 - 1.4) gives a general overview of ELMER Front. The rest of the chapter explains in detail all menus and commands needed in a typical ELMER Front session.

2.1 Input to ELMER Front

When a new model is to be built, in ELMER this starts by reading in the model geometry. This geometry must be first designed with some external program.

ELMER Front can read the geometry from three different sources:

- CAD geometry: a CAD file which has been created with an external CAD program or by other applications
- Mesh geometry: a mesh file which has been generated either by ELMER Mesh generator or by other applications
- Model file: a file produced and saved during a previous ELMER Front session

2.1.1 CAD file

A CAD file is an external file describing the model geometry in 2D wireframe format.

Supported CAD file formats are:

- *.unv (I-deas neutral CAD file format)

- *.igs (Iges file format)

- *.egf (Elmer Geometry File format). A text file format for creating basic 2D geometries. For more detailed description, see Appendix B

Overview:

After a CAD file has been read in, ELMER Front creates internally a boundary representation from the input geometry.

A CAD model contains typically multiple parts, called bodies. The geometry input file should describe all the bodies (called also objects or materials) which are needed in the problem.

In 2D bodies are defined by edge loops. These loops should be closed and not intersecting with themselves or with any other edge loops. In 3D, bodies are defined by surfaces (surface patches) and restrictions are similar as in 2D. Surfaces are described by edge loops and again equivalent restrictions hold as for bodies in 2D. However, currently only 2D CAD geometries are supported. This is because current ELMER Mesh generator is restricted to 2D geometry.

Geometry definitions:

When defining a CAD geometry, the following restrictions should be taken into account:

- No change in the geometry is allowed after the external CAD geometry is read in. This means that bodies or boundaries cannot be split or combined

- Because boundaries cannot be split, a boundary segment defined by corner vertices is the lowest level where boundary conditions can be set

- Because boundaries cannot be recombined, a large number of boundary segments results in a long list of boundaries. This happens even though from the problem viewpoint the boundary segments could form a single physical boundary. To simplify the problem it is advisable to simplify the CAD geometry to contain as few boundary segments as possible

2.1.2 Mesh file

A mesh file (or a set of files) is an external file where bodies are described by volume elements and the surfaces by boundary elements. If no boundaries are given, they are formed at body surfaces. The mesh input file can be a 2D or 3D mesh file.

Supported input formats are:

- Abaqus mesh files (*.inp)
- Fidap mesh files (*.fdneut)
- I-deas mesh files (*.unv)
- ELMER mesh files (mesh.header).

Overview:

In ELMER external mesh files are saved in an ELMER specific format and the result files are named using the following scheme:

- *mesh directory*/mesh.header
- *mesh directory*/mesh.nodes
- *mesh directory*/mesh.elements
- *mesh directory*/mesh.boundaries

Here *mesh directory* refers to the directory where ELMER mesh files are stored. This directory is created automatically and it is formed from the model directory and the mesh name using the following scheme:

mesh directory = *model directory*/MESHDIR/*mesh name*

Geometry definitions:

Contrary to a CAD geometry, a mesh geometry can be edited. Mesh boundaries can be split and combined by regrouping mesh elements that define the boundaries. This procedure is explained later in the context of the Edit menu.

When using an external mesh file, a typical problem is that too few boundaries are available. Without any specific boundary information, the only way to define boundaries is to divide them into internal boundaries (boundaries between bodies) and outer boundaries based on body (material) information. With only one body this would mean that there is only one (outer) boundary for the whole model.

2.1.3 Model file

Model file is an internal ELMER file containing all the model information, including the model geometry, in ELMER Front specific format.

For ELMER Front the model file is the most essential file. If the model file is deleted, all model definitions are also lost.

At the very first stage, an external geometry is read into ELMER Front. When the user defines the model, these definitions are stored in a model file. Default name for this file is *modelname*.emf. (The extension .emf is an abbreviation for ELMER Model File).

When model file is saved, all the model information which is relevant for solving the problem is automatically stored also in a separate ELMER Solver control file. Default name for this file is *modelname*.sif (The extension .sif is an abbreviation for Solver Input File).

Both these file are stored in the model directory. This directory can be set using the Model name and directories command in the Problem menu.

2.1.4 Structural overview

Figure 2.1 gives an overview of all files created or used in ELMER during a model building and simulation session.

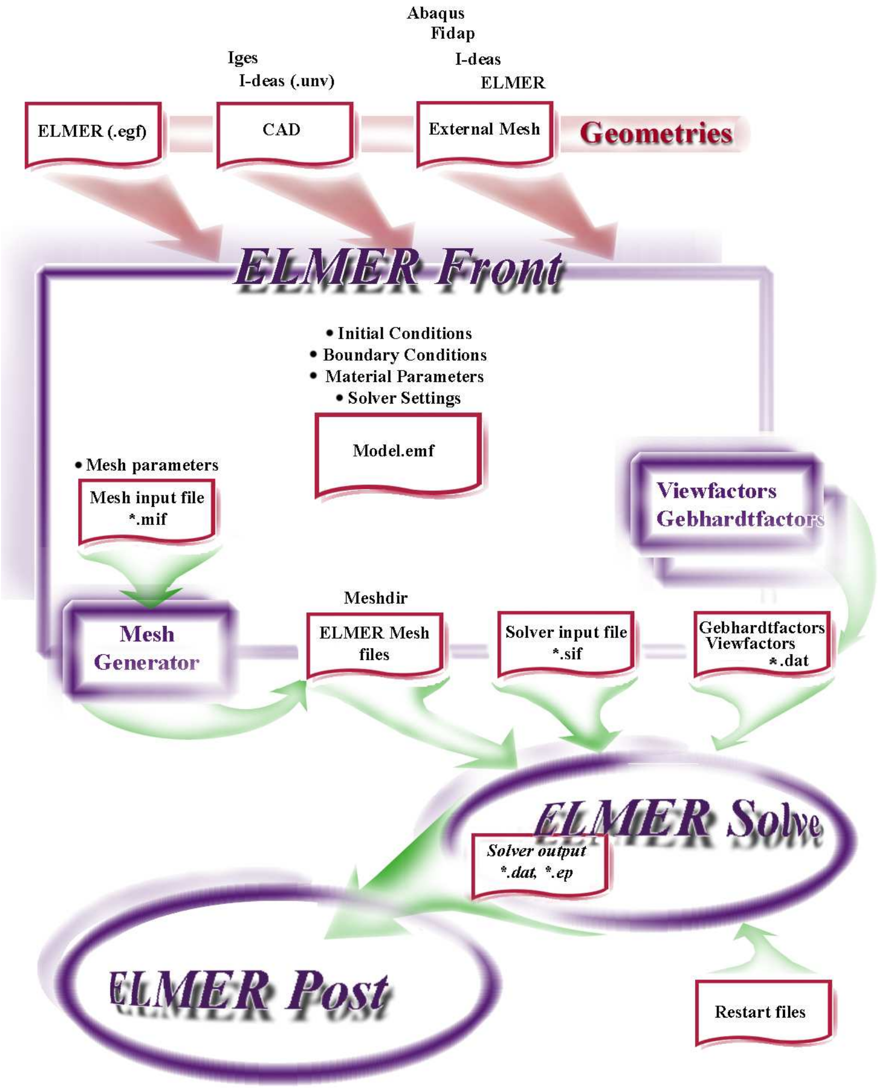


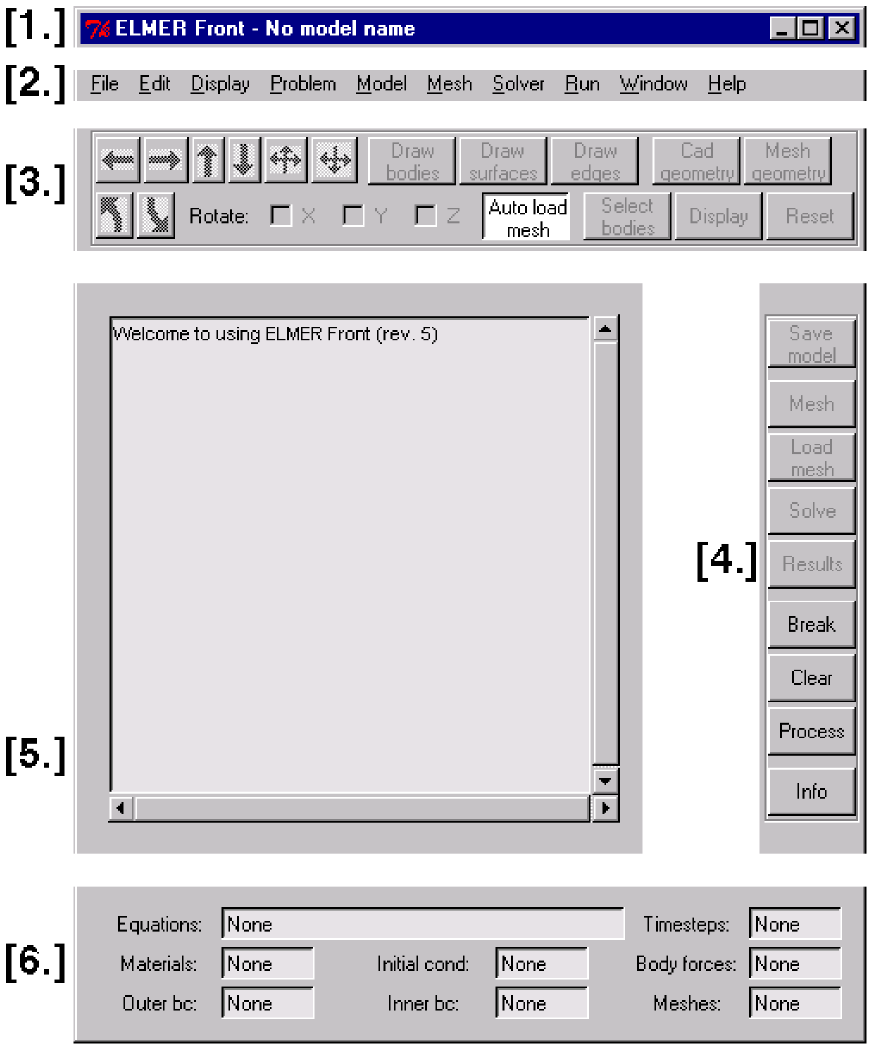
Figure 2.1: *Model files*

2.2 ELMER Front main window

ELMER Front main window is shown in Figure 2.2. The numbers shown in brackets refer to different areas in the main window. Window areas are the following:

- [1.] Titlebar
- [2.] Menubar

- [3.] Model window toolbar
- [4.] Prcoess control toolbar
- [5.] Message area
- [6.] Status area



Welcome to using ELMER Front (rev. 5)

Equations: None
Timesteps: None
Materials: None
Initial cond: None
Body forces: None
Outer bc: None
Inner bc: None
Meshes: None

Figure 2.2: *ELMER Front main window*

The Model window toolbar [3.] contains command buttons which control how the model geometry is displayed in the Model window. This window is a separate graphics window for the model geometry and it is not shown in Figure 2.2. The Process control toolbar [4.] can be used to start solver, postprocessor and other processes. The

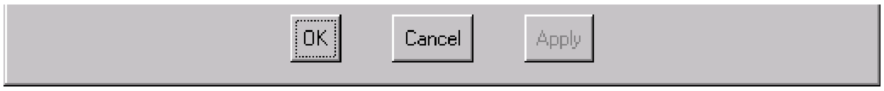
Message area [5.] in the middle of the window is used for displaying short messages related to commands and actions made during the session. The Status area [6.] at the bottom of the window tells the number of currently defined parameters and other status information for the model. In the example model parameters have not yet been defined as the 'None' values indicate.

2.3 General user interface features

This section gives an overview of the features which are common for all ELMER Front windows.

2.3.1 Standard window buttons

The window control buttons located at the bottom of each ELMER Front window have the following functions:



- Ok: accepts (if syntax is correct) and saves all possible changes made in the window. Note however that you still have to save the model file separately before changes are stored permanently. The window is closed
- Cancel: abandons all possible changes made in the window and closes the window
- Apply: same as the Ok button, but the window is left open

In general, windows will remain open until closed by the user with the Ok or Cancel buttons. Using the commands in the Window menu all open ELMER Front windows can be controlled in groups. For details, see the Window menu section in this chapter.

Modified data in entry fields is displayed using a red font until the modifications are accepted by pressing the Apply button.

In a window header, an asterisk displayed after the window title indicates that the data in the window has been modified. Pressing the

Apply button resets also this indicator.

2.3.2 Windows with body and boundary tables

A common factor to these windows is the Bodies listbox on the top left under the 'Bodies:' header. On the right side of the window is a listbox which contains the parameter sets which are entered in the panel. When these parameter sets are atteched directly to bodies, like the equations or material properties, only these two listboxes are visible. In some windows data is attached to the boundaries, and in these windows there is also a Boundaries listbox, which displays all the boundaries for the currently selected body.

In the following list the Boundaries, Boundary conditions and Mesh density window are boundary level windows, other windows are body level windows.

- *Edit menu:*
    - Bodies
    - Boundaries
- *Problem menu:*
    - Equations
- *Model menu:*
    - Initial conditions
    - Body forces
    - Materials
    - Boundary conditions
- *Mesh menu / Define Mesh:*
    - Mesh structure
    - Mesh density

Note: These windows are designed to be used in conjunction with the Model window, because it helps to identify bodies and boundaries in the model. Refer to the Model window section in the end of this chapter.

When boundaries are displayed in the Model window they are numbered using the same id numbers that are used in boundary tables.

All these windows have common rules and features for the data entry, as explained below.

The first step in the data entry is to select the target body or boundary for the data. After that definitions can be entered using the data entry fields in the window. Some of the fields may however be disabled, because they are not applicable for the target body or boundary. For example, if the user is entering a material definition for a body, and the body does not have a flow equation defined, it would not be possible to enter any flow related material data for the body.

In many windows data fields are devided into subsets by equation type. These subsets can be activated and displayed using the equation type radio buttons in the area above the data fields.

Data in the currently active entry field can be reset to the previous value by pressing the Esc key on the keyboard. If a field supports entering multiple values, a space is used as a separator.

2.3.3 Attaching parameter sets to bodies and boundaries

When all necessary data for the new parameter set has been entered, it is added to the list of parameter sets using the Add button under the parameter listbox. If an existing parameter set is being modified, the update is activated only after the Update button is pressed. The Delete button removes the currently selected entry from the listbox.

Each paramter must have a unique name. In general it is better to use descriptive names for parameter sets. However, it is also possible to use the deafult automatic naming scheme where parameters are named like: 'BodyForce1', 'BodyForce2', . . . If no parameter name is given, or the name is given is this basic form, the next available name in similar form is created automatically when the Add button is pressed. This practice is not recommended, but it be useful for quick experiments where the names are not essential.

If a target body or boundary does not currently have any parameter set attached to it, a new parameter set is automatically attached to the target body or boundary when the parameter is added to the listbox. Otherwise the user has to use the Attach button to attach the currently selected parameter to the currently selected body or boundary.

Only one parameter set can be selected at a time, but in some cases it is possible to select multiple target bodies or boundaries for the parameter. Multiple simultaneous selections are done by pressing the control key when selecting the targets with the left mouse button. Pressing the shift key and the left mouse button while dragging with the mouse selects a continuous group of rows from the listbox.

When a parameter set is added to the listbox, it inherits certain properties from the active target body or boundary. In the Model menu windows these properties are related to the equation type of the active body. In the Mesh menu windows this property is the mesh structure of the body. This property inheritance creates some restircitions on how parameters can be attached to target bodies or boundaries. An attachment is not possible if the parameter and the target have conflicting properties. For example it is not possible to attach a material property for a body which has only heat equation defined, if this material definition was originally created for a flow body.

To identify this matching, an asterisk ('*') is used to mark parameter sets which match the selected body or boundary. This asterisk is shown in the beginning of the parameter row.

When a body or an boundary is selected, all those existing parameter sets which can be attach to the selected target are marked with an asterisk.

It is possible to select a parameter which is not marked with an asterisk. This parameter can be edited and updated, but it cannot be attached to the current target (the Attach button is not active), because it does not match this target.

When a parameter set is double-clicked, the first target, to which this parameter is possibly attached, is activated. The next double-click will activate the next target etc.

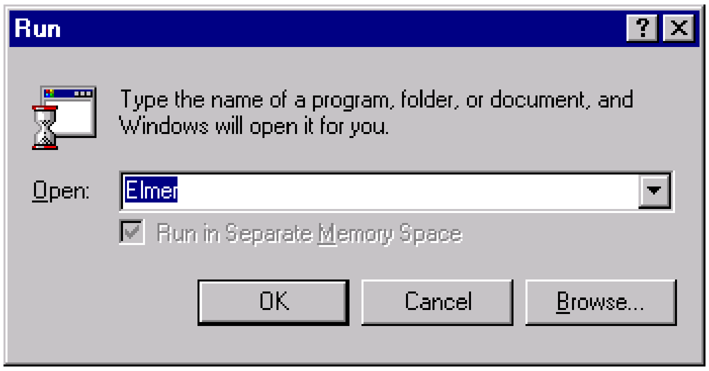
The buttons which are used to the handling of the paramter sets have the following functions:

- Add: adds a new parameter defined with the values in the data entry fields to the list of parameter sets
- Update: updates the currently active parameter with the values in the data entry fields

- Delete: deletes the selected parameter from the list and detaches the parameter from all the bodies or boundaries it is possibly attached to
- Attach: attaches the currently selected parameter to the currently selected body or boundary
- Detach: removes the current attachment from the currently selected body

2.4 Starting ELMER Front

ELMER Front starts by the 'ELMER' command. This command can be given in the command line or by using desktop menus or icons if they are available in the platform where ELMER is used.



Type the name of a program, folder, or document, and Windows will open it for you.

Open: Elmer

☑ Run in Separate Memory Space

Figure 2.3: *ELMER startup command under Microsoft Windows NT*

The following optional parameters may be used when the program is started:

- --model-directory=*model directory name*
- --model-name=*model name*
- --problem-name=*problem name*
- --settings-file=*settings file name*
- --results-directory=*results directory name*
- --include-path=*include path definition*
- --log-directory=*log directory name*

For example if ELMER Front is started using the command:

```
ELMER --model-directory=/ELMER/MODELS/stepflow
```

then the directory /ELMER/MODELS/stepflow would be the dedault directory for model files. If a model file stepflow.emf were stored in that directory, this command would automatically load this file into ELMER Front.

2.5 ELMER Front settings file

When an ELMER Front session is started, values for various control variables can be initialized from an input file. This file is loaded automatically if it is found in the current directory (i.e. in the directory where ELMER Front was started) or in a directory which is given by a user defined environment variable named 'ELMER_USER_HOME'.

The file is searched using the following locations, names and search order:

- Current directory/ElmerFrontSettings_*username*.txt
- Current directory/ElmerFrontSettings.txt
- ELMER_USER_HOME/ElmerFrontSettings_*username*.txt
- ELMER_USER_HOME/ElmerFrontSettings.txt

Here 'username' refers to the value of the system environment variable which stores the name for the current user. In MS Windows NT platform this variable is called 'USERNAME', in Unix systems it is typically called 'user'. During an ELEMR Front session this value can be seen using the System Info command in the Help menu.

The name for the settings file can be specified also in the command line when ELMER Front is started as explained in the previous section.

The structure and the contents of this file is described in Appendix C.

2.6 File menu

File menu commands are used for opening, saving and browsing the model data and related files. These commands are shown in Figure

2.6.

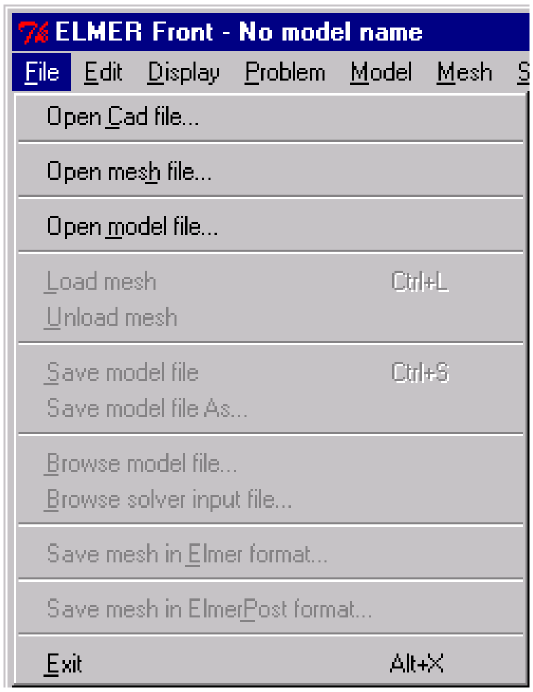


Figure 2.4: *File menu*

2.6.1 Opening commands

A new or an existing model is opened using the following commands:

- Open CAD file: read model geometry from an external CAD file
- Open mesh file: read model geometry and mesh from an external mesh file
- Open model file: read a previously saved model file (*.emf file)

If a model was previously opened and there is some unsaved data, an option is given to save the old model before loading a new one.

2.6.2 Load/Unload Mesh

Mesh can be loaded or unloaded during a ELMER Front session as follows:

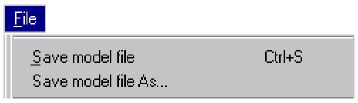
- Load mesh: if mesh file is available but currently not loaded, this commands reads the mesh into ELMER Front

- Unload mesh: this commands removes the mesh from the memory. Sometimes it may be useful to remove a large mesh from the memory before starting other ELMER modules

Note: If mesh is very large, it could be useful not to read in the mesh automatically. This autoload behaviour can be controlled using the Settings command in the Edit menu or through the Autoload mesh button in the main window.

2.6.3 Save model

The commands for storing model files and related data are the following:



- Save model file: saves model file (.emf file) in the model directory

- Save model file As: same as previous, but the name and location for the model file is asked instead of using the default name and location

Note: When model definitions have been modified model data should be saved. Otherwise updated model data is not available for other ELMER modules. The model file is saved automatically when ELMER Solver is started (if Auto save model file setting is turned on, as it is by default). This concerns also all other processes started from the Run menu (excluding ELMER Post).

2.6.4 Browser windows

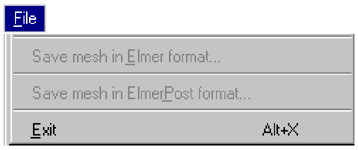
These windows are meant for browsing the files, file editing is not possible.

- Browse model file: displays the model file in a browser window

- Browse solver input file: displays the solver input file in a browser window

Note: Although model file is a normal text file, it is not meant to be edited 'off line'. The internal file structure is strictly predefined and any outside changes could make it unreadable for ELMER Front.

2.6.5 Save mesh



By default an external mesh is saved as an ELMER mesh when the model is saved.

Save mesh in ELMER format command can be used to save the external mesh explicitly if this default behavior is turned off.

Save mesh in ELMER Post format command saves the mesh in the ELMER postprocessor format

2.6.6 Exit

Exit command ends the current ELMER session. If there is any unsaved data, the user is given an option to save the model before closing the program.

2.7 Edit menu

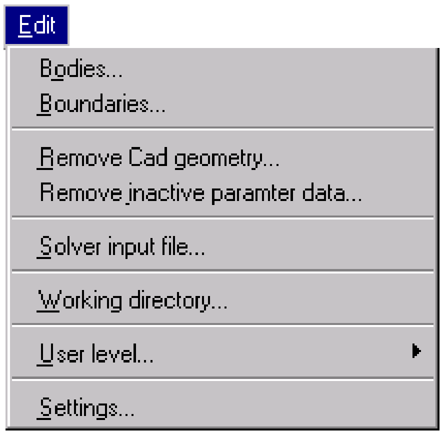


Figure 2.5: *Edit menu*

2.7.1 Bodies

Bodies command opens a window where body names and colors can be defined. This window is shown in Figure 2.6.

Default body names are in the form BodyN, where N is a sequence number. Because body names are used in the windows where data related to the body will be entered, they should be descriptive to make the identifying the bodies easier.

The name of the selected body is displayed in the name entry field where it can be edited.

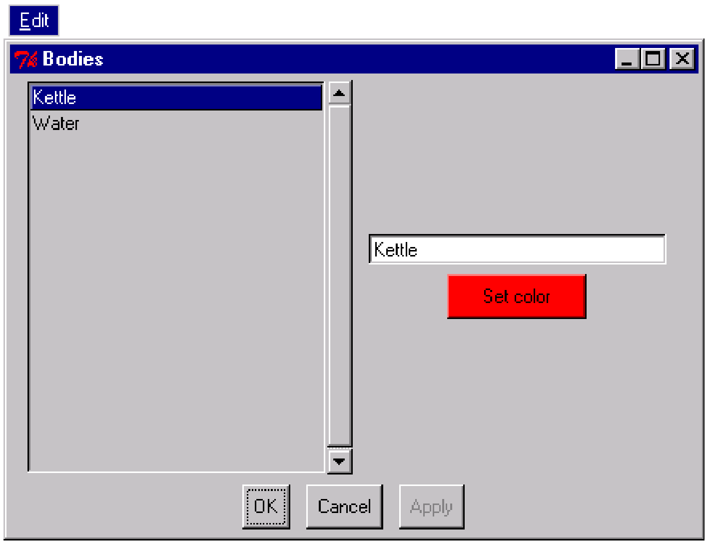
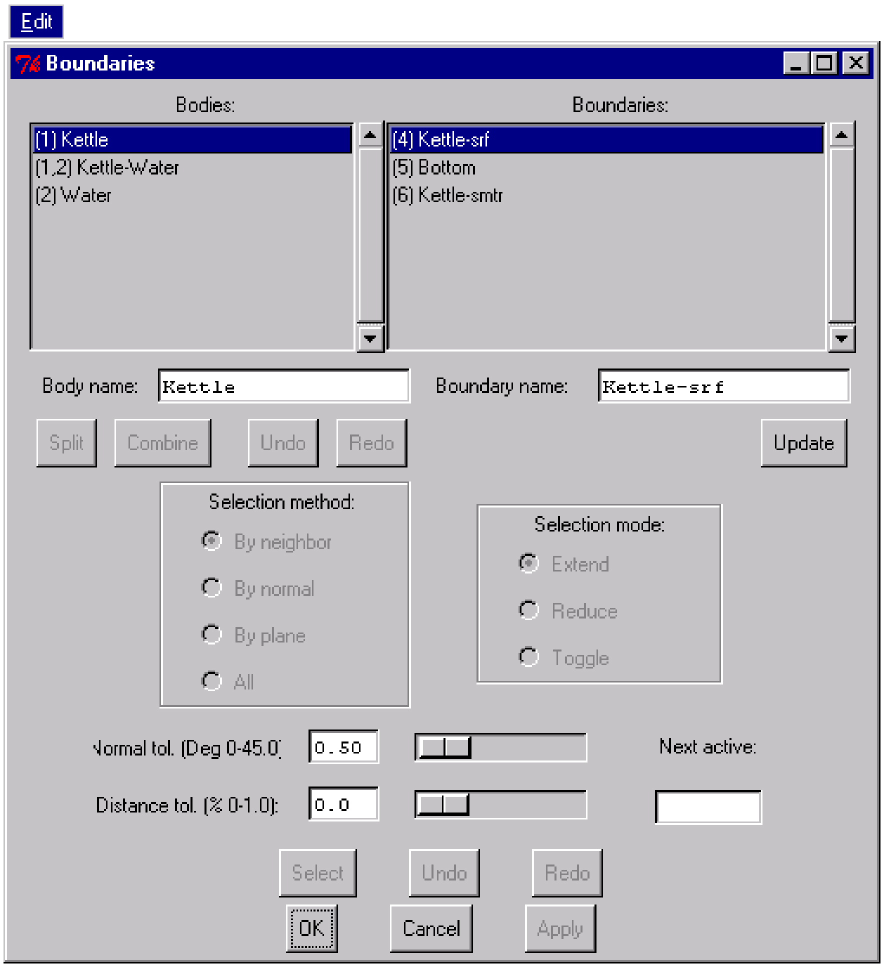


Figure 2.6: *Bodies window*

The color of the selected body color is shown on the Set color button. This color is used when the body is displayed in the Model window. (If the Model window is not currently visible, it can be opened by the Display model command in the Display menu or using the Display button in the main window). Pressing the Set color button opens a color palette where the color for the body can be defined.

2.7.2 Boundaries


Body name: Kettle

Boundary name: Kettle-srf

Selection method: ● By neighbor

Selection mode: ● Extend

Normal tol. (Deg 0-45.0): 0.50

Distance tol. (% 0-1.0): 0.0

Figure 2.7: *Boundaries window*

Boundaries command opens a window where boundary names can are defined. If geometry is read from an external mesh file (i.e. model does not contain a CAD geometry), boundaries can also be edited by splitting and combining them.

Boundaries (edges or surfaces) of separate bodies can be adjacent. Such boundaries are called inner boundaries and the adjacent bodies are called body pairs. Other boundaries are called outer boundaries. When a body is selected, all its outer boundaries are displayed in the boudaries listbox. Inner boundaries can be displayed by selecting the corresponding body pair.

Inner and outer boundaries are sometimes called as boundary elements. They should not be confused with mesh boundary elements.

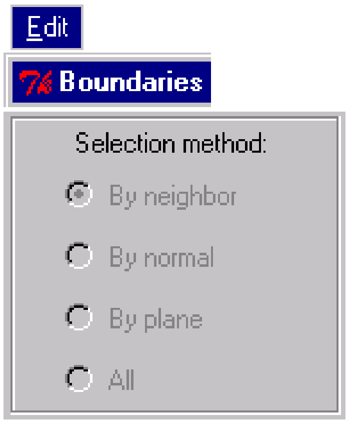
Boundary names are by default in the form BoundaryN, where N is a sequence number. A boundary can be activated by selecting it from the boundary listbox or from the Model window. The name of the selected boundary is displayed in the boundary name entry field, where it can be edited.

Editing boundaries can be done only for one body or bodypair at a time. Only one boundary can be split at a time. Only complete boundaries can be combined. Split and combine operations can be applied in any order and both operations can be cancelled using the undo command.

Multiple boundaries can be selected by pressing the Control key during a selection.

Note: See also the Model Window section at the end of this chapter.

2.7.3 Element Selection method



Selection method:

● By neighbor

○ By normal

○ By plane

○ All

A split operation for a boundary is done by selecting a group of mesh elements in the boundary and by forming a new boundary with this group. The remaining mesh boundary elements stay with the original boundary.

Selecting the mesh elements can be done by two methods:

1. First method: elements can be selected from the graphics window (Model window) by ‘painting’ them directly. This is done by pressing the shift key, and painting with the mouse cursor while pressing the left mouse button. If control key is pressed instead of the shift key, mesh elements can be selected by clicking them with the left mouse button. Using the right mouse button cancels the current selections.

2. Second method: at first a reference mesh element is selected using the mouse. This base selection can then be extended by using the Select button in the window. The extended selection is controlled by the criteria set in the Selection radio button group. These criteria are the following:

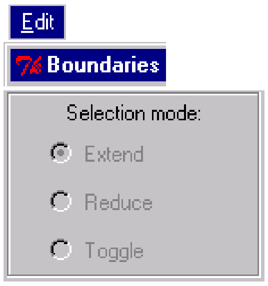
- By neighbor: selection is extended by adding neighbor elements for currently selected elements. Neighbor elements are added only if their normal vector changes less than what is set by the Normal tolerance slider.

- By normal: select all elements whose normal vectors differs less than the given tolerance from the normal vector of the reference element

- By plane: select all elements which are on the same plane as the reference element. Plane distance can differ within the tolerance which is set by the Distance tolerance slider.

- All: select all mesh elements in the boundary

2.7.4 Selection mode



Selection mode controls how the element selection actually behaves:

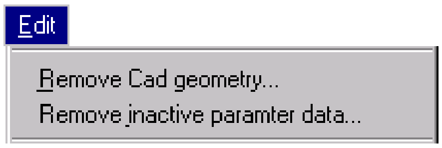
- Extend: always extends the current selection

- Reduce: always reduces (deselects) the current selection

- Toggle: toggle the selection state for the elements being effected

Value displayed in the Next active field tells the next tolerance value which would select new mesh elements.

2.7.5 Remove CAD geometry



Mesh boundaries cannot be modified if model geometry is defined by a CAD geometry. This limitation is implemented to prevent conflicts between the CAD and mesh geometries. However, if the original CAD geometry is removed from the model, the existing mesh can be modified as it were a normal external mesh.

The original CAD geometry is removed from the model using the Remove Cad geometry command. Note that remeshing is not possible if the CAD geometry is removed. It is also not possible to restore the CAD geometry, so it is recommended to save the original model and save the mesh only version in a different model file.

2.7.6 Remove inactive parameter data

An equation may be inactivated in the model. However, the parameters related to this equation are not erased by this. They just become idle and they will be reactivated if the equation is again turned on. This command removes these nonactive parameters permanently from the model file.

2.7.7 Solver input file

An existing solver input file is opened for editing with the Solver input file command. However, two requirements has first to be met:

1. User level has to be set to Power user. (See User level menu below)

2. The Auto save solver input file selection in the Settings window has to be checked off. By default all changes in the solver input file will be overwritten when the model is saved for the next time. This may be prevented by setting the Auto save solver input file checkbox off.

2.7.8 Working directory

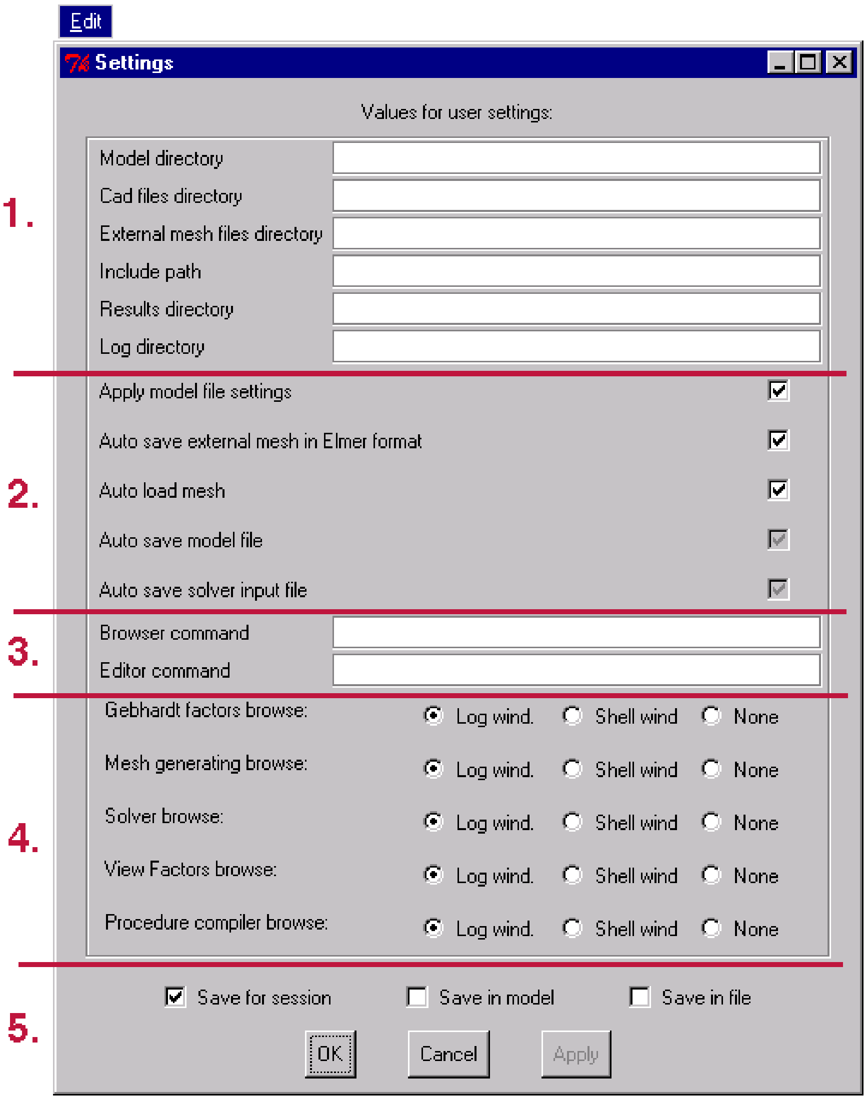
The current working directory can be set using this command. This directory is used as the default model directory.

2.7.9 User level

The user level setting controls the amount of commands displayed in menus and the amount of warning messages displayed when working with the panels. There are three levels available. New users should first use the Novice level. This level simplifies menus by displaying only the most essential commands. It also protects againts unintentional loss of data by giving warning messages when modified panels are closed without saving the data etc. The Power user level in contrast practically bypasses all warnings. The Advanced and Power user levels speed up the work for experienced users.

Note: The default level in the beginning of each session is the Advanced level.

2.7.10 Settings



Values for user settings:

Model directory

Cad files directory

External mesh files directory

Include path

Results directory

Log directory

Apply model file settings ☑

Auto save external mesh in Elmer format ☑

Auto load mesh ☑

Auto save model file ☑

Auto save solver input file ☑

Browser command

Editor command

Gebhardt factors browse: ● Log wind. ○ Shell wind ○ None

Mesh generating browse: ● Log wind. ○ Shell wind ○ None

Solver browse: ● Log wind. ○ Shell wind ○ None

View Factors browse: ● Log wind. ○ Shell wind ○ None

Procedure compiler browse: ● Log wind. ○ Shell wind ○ None

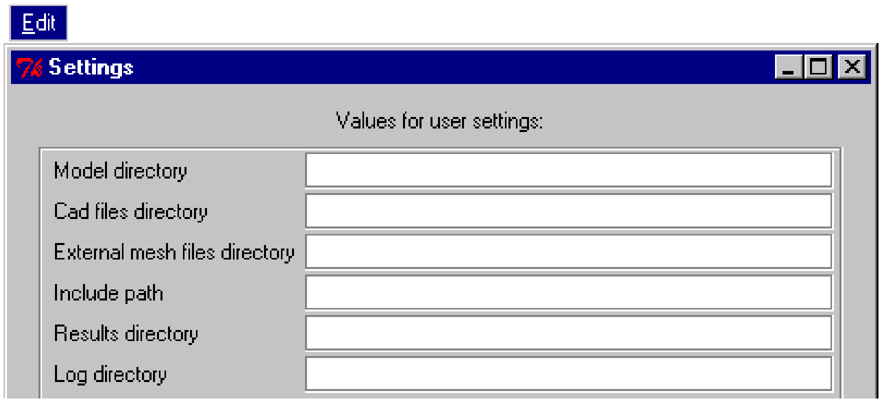
☑ Save for session ☐ Save in model ☐ Save in file

Figure 2.8: *Settings window*

Settings command opens a window where current problem settings can be modified.

The setting parameters can be devided into five groups. These groups are shown by numbers 1...5 in Figure 2.8 and they explained below in this order.

1. Directory definitions:



Values for user settings:

Model directory
Cad files directory
External mesh files directory
Include path
Results directory
Log directory

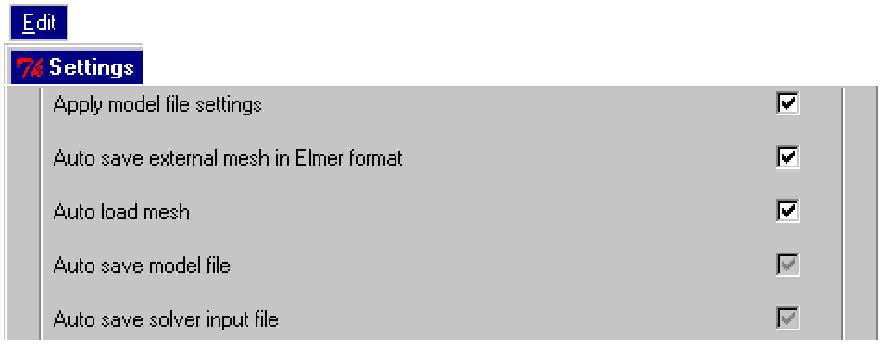
The default values for various model directories can be set and modified using these entry fields.

2. Save/Load control:

These parameters control various actions associated to the loading and saving of a model file.

- Apply model file settings forces the settings saved in a model file to be applied when the model file is loaded. Otherwise these settings are ignored

- Auto save external mesh in Elmer format forces the external mesh to be saved automatically in the Elmer mesh format when the corresponding model file is saved.



Apply model file settings ☑

Auto save external mesh in Elmer format ☑

Auto load mesh ☑

Auto save model file ☑

Auto save solver input file ☑

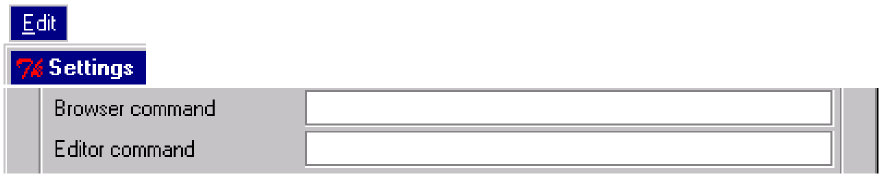
- Auto load mesh forces the possible mesh to be loaded always when a model file is loaded.

- Auto save model file forces the model file to be saved automatically when processes in the Run menu are started (excluding ELMER Post). This option is active only in the Power user level

- Auto save solver input file controls whether the solver input file will be saved when a model file is saved. This option is active only in the Power user level

Note: If the model contains a very large mesh and displaying the mesh is not necessary, it may be useful to turn off temporarily the mesh autoloading. This can be done by unchecking the first and third checkboxes seen in the figure above, and then selecting the Save for session command checkbox at bottom of this window and finally pressing the Ok button.

3. Browser and Editor:



Browser command

Editor command

- Browser command field can be used to define an external program for file browsing

- Editor command field can be used to define an external program for file editing.

4. Data output methods:

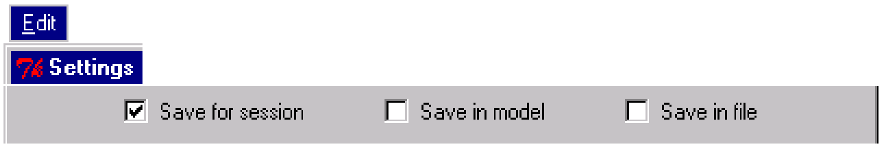
These radio buttons are used to control where the output from other ELMER modules or the procedure compiler is directed or stored to:

Gebhardt factors browse: ● Log wind. ○ Shell wind ○ None

- Log file: save output into a log file
- Shell window: direct output to the operating system shell window
- None: Nothing is displayed during the process

Note: Possible logfiles are stored in the Log directory This directory can be defined using the Model name and directories command in the Problem menu. Logfiles can be viewed in the process browser window (see the Process table command in the Run menu). Logfiles are automatically deleted when an Elmer Front session ends.

5. Save options:



☑ Save for session ☐ Save in model ☐ Save in file

The following options are used to control how the parameter values set in this window are used:

- Save for session: current settings apply only in this session
- Save in model: settings are saved (when applicable) in the current model file
- Save in file: settings are saved in the current settings file

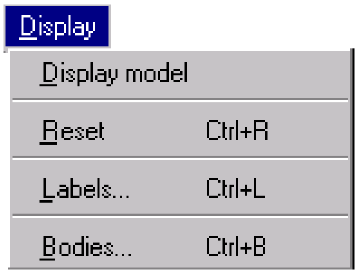
2.8 Display menu

Display menu is used to control the Model graphics window.

2.8.1 Display model

The Display model command can be used to open the Model window if it is not visible. This command is also attached to the Display button in the main window.

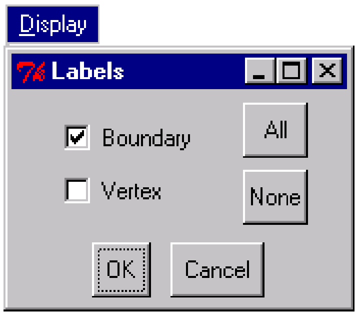
The model is displayed in a separate graphics window. For details see the Model window section in this chapter.



2.8.2 Reset

The Reset command sets the Model window into its initial state.

2.8.3 Labels



☑ Boundary
☐ Vertex

All body elements (inner and outer boundaries, edges and vertices) are uniquely numbered. These numbers can be displayed as element labels in the Model window. These same numbers are also used to identify the elements in the data entry windows.

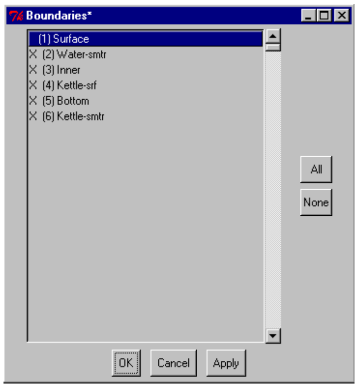
The Labels command opens a window where the lable display can be controlled.

In a complex model it is sometimes desirable to turn the labels off by unchecking the corresponding checkboxes. The All and None buttons control these labels in groups.

2.8.4 Bodies

Bodies to be displayed can be selected by using the Bodies menu command or the Select bodies button in the main window.

2.8.5 Boundaries



Boundaries to be displayed can be selected by using the Boundaries command in the Display menu. When a boundary is double-clicked in the listbox, its draw mode is turned off or on. The 'All' and 'None' settings can be activated by pressing the Apply or Ok buttons.

2.9 Problem menu

In this menu the actual problem definition is started. Most of the definitions here concern the whole model and are not specific to bodies or body elements.

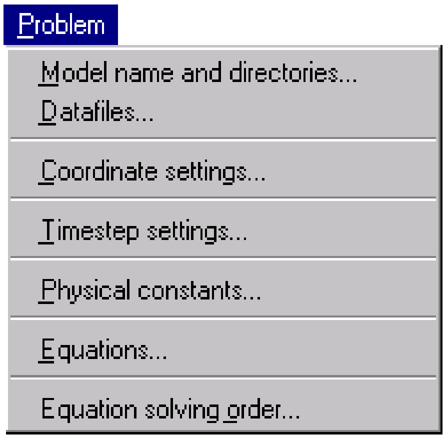
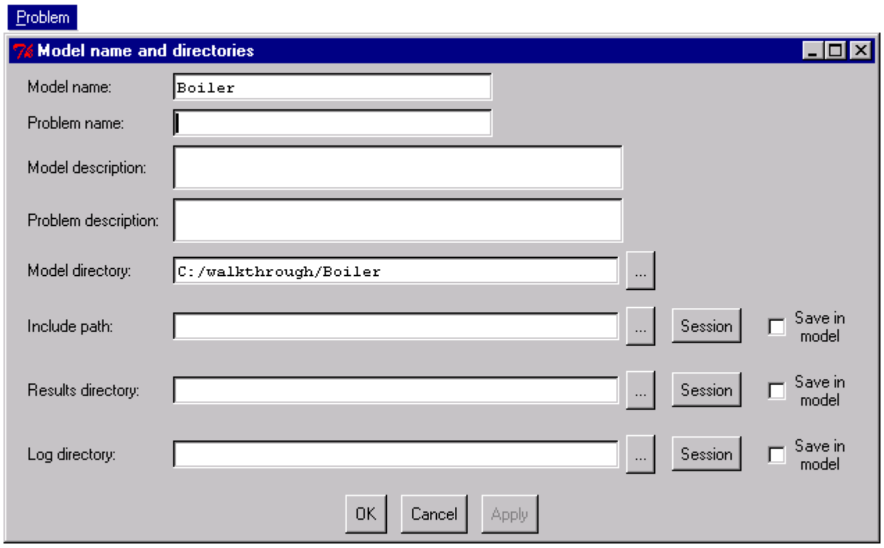


Figure 2.9: *Problem menu*

2.9.1 Model Name and directories

The Model name and directories command opens a window where the model name and the file directories can be entered. These directories are used when reading, saving or searching data files related to the model.



Model name: Boiler

Model directory: C:/walkthrough/Boiler

Figure 2.10: *Model name and directories window*

Note: it is possible to use spaces in the model and problem name, but because these names are used also in filenames, it may be safer to avoid spaces. Spaces can create problems especially when filenames are used as arguments when calling external programs.

- Model name: this name must be given because it is used as a part of the model file name whose default value is: *modelname*.emf

- Problem name: this optional name can be used for example to identify different versions of the basic model. If given, it will be part of the model file name: *modelname.problemname*.emf

- Model description: this comment field is optional. It is displayed only in this window

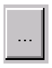
- Problem description: this comment field is optional. It can be used to describe a specific version of the basic model. It is displayed only in this window

- Model directory: the root directory for the model files

- Include path: A semicolon separated list of paths which are used as search directories when looking for parameter files. Parameter file can be used in the following windows: Initial conditions, Body forces, Material parameters and Boundary conditions

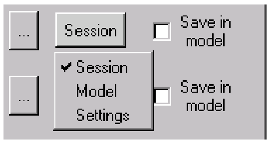
- Results directory: the root directory for the solver output files. These simulation result files are stored using the following directory structure: *result directory*/MESHDIR/*meshname*. The default value for this directory is the model directory.

- Log directory: The location for the log files created during the session, the defaul value is *model directory*/LOGDIR.



The directory browser button opens a browser window for selecting directories.

Note: Directory selection is done by single-clicking the directory name in the listbox and pressing ok button. Double-clicking the directory name opens a subdirectory list.



When this button is pressed a selection list opens. Currently active type is displayed as the button label. Available options are:

- Session: the value defined for the session is used

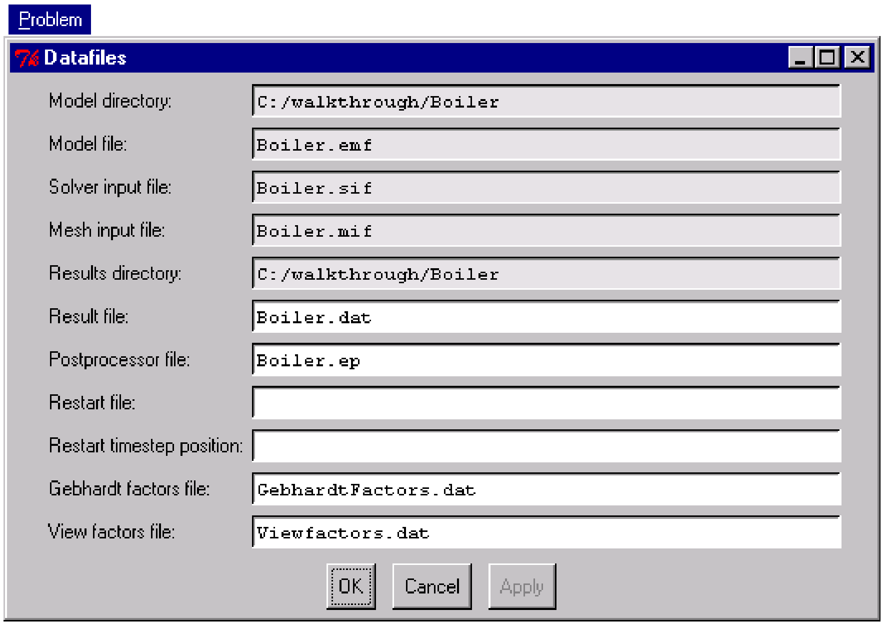
- Model: the value defined in the model file is used

- Settings: the value defined in the settings files is used

When the Save in model checkbox is checked, the active value is saved in the model file.

2.9.2 Datafiles

The names for different input and output file can be defined in this window which is opened by the Datafiles command. This window is shown in Figure 2.11.



Model directory: C:/walkthrough/Boiler
Model file: Boiler.emf
Solver input file: Boiler.sif
Mesh input file: Boiler.mif
Results directory: C:/walkthrough/Boiler
Result file: Boiler.dat
Postprocessor file: Boiler.ep
Restart file:
Restart timestep position:
Gebhardt factors file: GebhardtFactors.dat
View factors file: Viewfactors.dat

Figure 2.11: *Datafiles window*

- Model directory: (information field, not editable)
- Model file: (information field, not editable)
- Solver input file: (information field, not editable)
- Mesh input file: (information field, not editable)
- Results directory: (information field, not editable)
- Result file: solver output file, the default value is: *modelname*.dat
- Postprocessor file: this file stores solver results in the ELMER Post format, the default value is: *modelname*.ep.
- Restart file: if this file is given ELMER Solver uses the variable values in this file as initial values. This file should be in the output file format

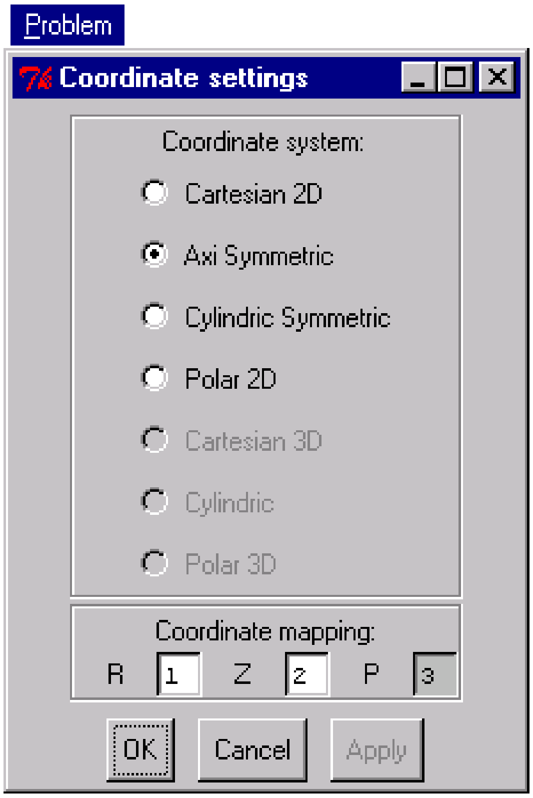
- Restart timestep position: the timestep in the restart file which is used for the restart values. A zero value in this field means that the last time step is used.

- Gebhardt factors file: this file is used to store Gebhardt factors when they are calculated using the Gebhardt factors command in the Run menu. Default value is: GebhardtFactors.dat

- View factors file: this file is used to store view factors when they are calculated using the View factors command in the Run menu. Default value is: ViewFactors.dat

Note: Only simple file names can be entered in the file name fields, because directory structure is already predefined by the result directory.

2.9.3 Coordinate settings



Coordinate system:
○ Cartesian 2D
● Axi Symmetric
○ Cylindric Symmetric
○ Polar 2D
○ Cartesian 3D
○ Cylindric
○ Polar 3D

Coordinate mapping:
R 1 Z 2 P 3

Figure 2.12: *Coordinate settings window*

Model's coordinate system is specified in this window which is opened with the Coordinate settings command.

Available coordinate systems depend on the dimension of the model geometry.

In 2D possible options are:

- *Cartesian 2D*
- *Axisymmetric*
- *Cylindric symmetric*
- *Polar 2D*

In 3D options are:

- *Cartesian 3D*
- *Cylindric*
- *Polar 3D*

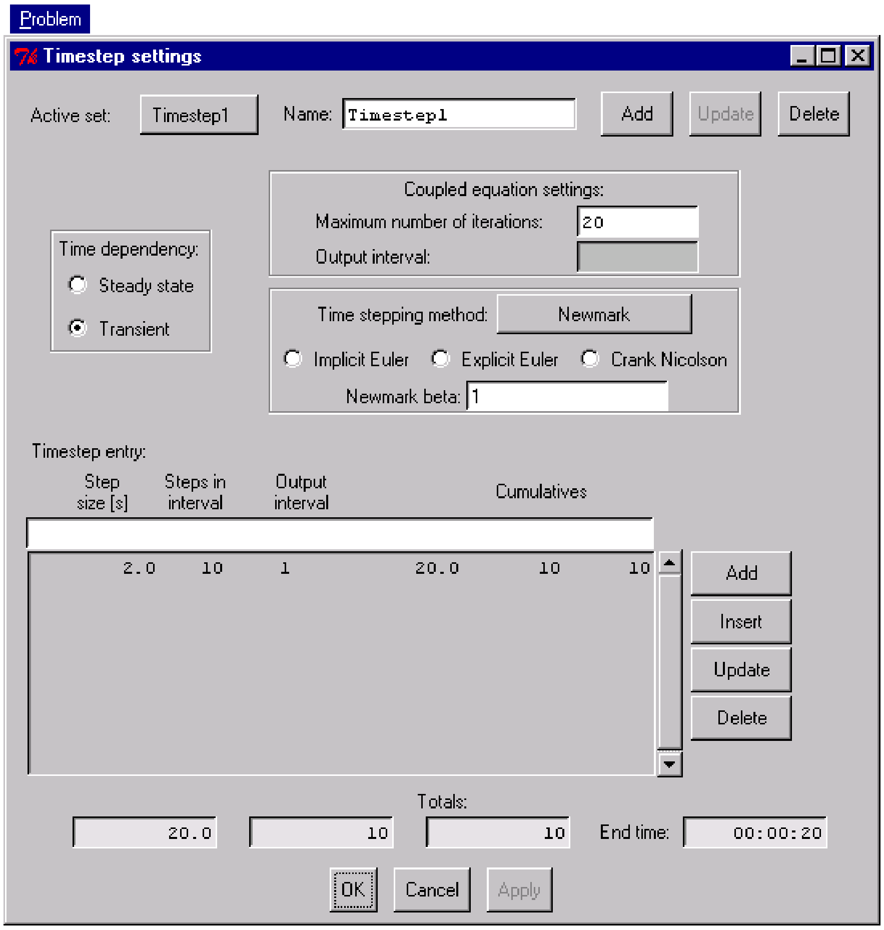
Coordinate Mapping: Coordinate dependent variables (like velocity) are entered in data windows by components (like velocity-X, velocity-Y, velocity-Z). The default mapping between the components and the coordinate axes is X = 1, Y = 2 and Z = 3. This mapping can be changed entering a new index set in the Coordinate mapping fields.

The field labels for the entries are also altered when coordinate system is changed:

- *Cartesian: X, Y, Z*
- *Cylindric: R(ho), Z, P(hi)*
- *Polar: R(ho), T(heta), P(hi)*

These labels are also used in data entry windows to illustrate the coordinate system and the mapping currently in use.

2.9.4 Timestep settings



Active set: Timestep1

Name: Timestep1

Maximum number of iterations: 20

Output interval:

Time dependency:
○ Steady state
● Transient

Time stepping method: Newmark

○ Implicit Euler ○ Explicit Euler ○ Crank Nicolson

Newmark beta: 1

Step size [s]	Steps in interval	Output interval	Cumulatives		
2.0	10	1	20.0	10	10

Totals: 20.0 10 10 End time: 00:00:20

Figure 2.13: *Timestep settings window*

The Timestep settings window is used to set the time dependency and timestep settings for the simulation.

Active set button: selection list of the currently stored timestep sets. The name of the currently active set is displayed as the button label

Name: a user given name for the current timestep set

The Add, Update, and Delete buttons are used to add, update and delete timestep set entries.

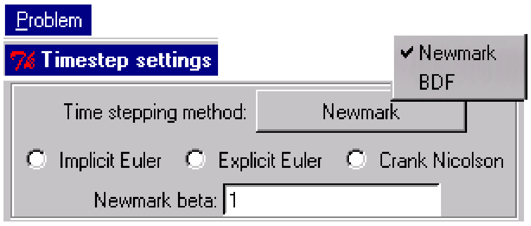
The time dependency type of the current simulation problem is set in the Time dependency box. Options are Steady state and Transient.

In the Steady state case, total iteration amount and the output frequency are controlled by the following fields:

- Maximum number of iterations: Maximum limit for the steady state iterations (iteration is stopped even if the steady state convergence criterion is not yet met, if this limit is met first)
- Output interval: Output frequency for steady state iterations

For transient problems two basic timestepping schemes are available: Newmark and BDF.

In the Newmark method the beta parameter can be freely selected, but the three most common methods can be simply selected using the radio buttons for the Impilict Euler, Explicit Euler and the Crank Nicolson methods. These methods correspond the beta values 1.0, 0.0 and 0.5. Default value is 1.0 (Implicit Euler).

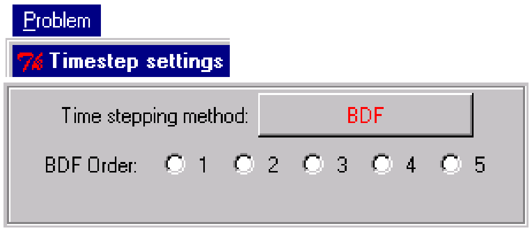


Time stepping method: Newmark

○ Implicit Euler ○ Explicit Euler ○ Crank Nicolson

Newmark beta: 1

For the BDF method only one of the predefined values 1..5 can be selected.



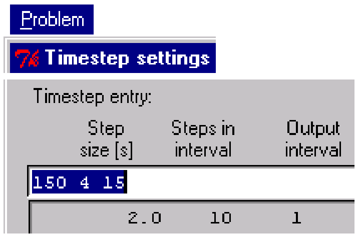
Time stepping method: BDF

BDF Order: ○ 1 ○ 2 ○ 3 ○ 4 ○ 5

Timesteps are defined by entering three numbers in the Timesteps field. They have the following meaning (values must be entered in this order):

- Step size: the step size in seconds in the interval
- Steps in interval: the number of steps in the interval
- Output interval: the output frequency in the interval

A new timestep interval can be added by entering three values separated with a space and then pressing the Add button. In the example step size is 150, steps in interval 4, and output interval 15 (because every fifteenth timestep should be output, this setting would not create any output from this interval!).



Timestep entry: 150 4 15

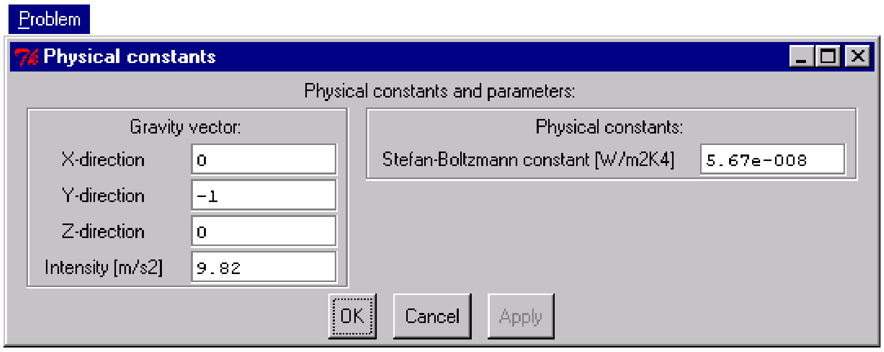
Step size [s]	Steps in interval	Output interval
2.0	10	1

New entries are added at the end of the table when the Add button or the Enter key is pressed. The Insert button inserts a new entry above the currently selected row in the table. The currently active row may be updated by updating the values in the Timestep field and then pressing the Update button. The currently selected row may be deleted from the table by pressing the Delete button.

For each row in the list, running cumulatives are displayed as the last three numbers. Total cumulatives are shown in the Totals fields at the bottom of the window.

2.9.5 Physical constants

The Physical constants window is used to modify the default values for physical constants. Modifying these values may be necesary when for example input values are in nonstandard units. The constants are the gravity vector and the Stefan-Boltzmann constant.



Physical constants and parameters:

Gravity vector:

X-direction: 0

Y-direction: -1

Z-direction: 0

Intensity [m/s2]: 9.82

Physical constants:

Stefan-Boltzmann constant [W/m2K4]: 5.67e-008

The default gravity value is 9.82 m/s^2 in the direction of the negative y-axis.

2.9.6 Equations

Assigning equations for the bodies is perhaps the most important task in the problem setup in ELMER Front. Each body must have an equation attached to it before any equation related data (like material property sets, body force sets, initial and boundary condition sets) can be entered for the body or its boundaries.

A typical equation definition sequence could be the following:

- A body or bodies are selected

- An equation is defined;

  a descriptive equation name is entered

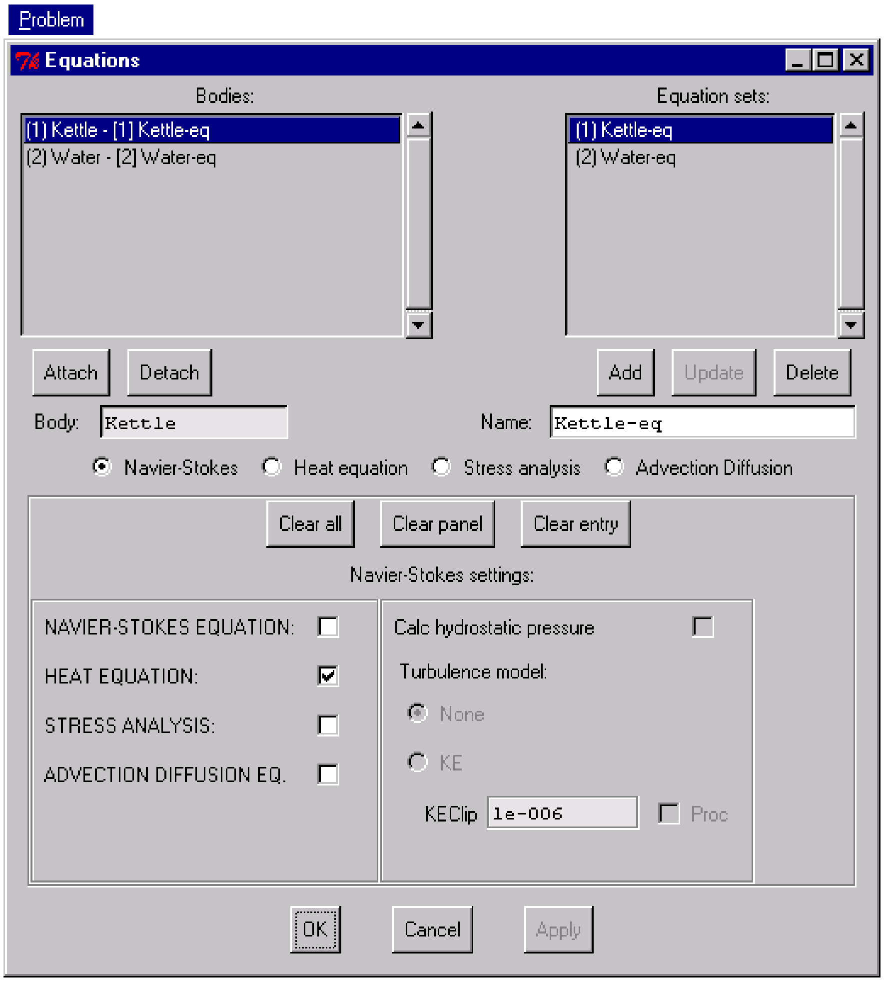
  an equation type is selected

  parameters for the equation are entered.

- When all data is entered, the new equation set is added to the list using the Add button

- The defined equation set is attached to the selected body (or bodies) using the Attach button

Equations window is shown in Figure 2.14.



Bodies:

(1) Kettle - [1] Kettle-eq
(2) Water - [2] Water-eq

Equation sets:

(1) Kettle-eq
(2) Water-eq

Body: Kettle

Name: Kettle-eq

● Navier-Stokes ○ Heat equation ○ Stress analysis ○ Advection Diffusion

Navier-Stokes settings:

NAVIER-STOKES EQUATION: ☐
HEAT EQUATION: ☑
STRESS ANALYSIS: ☐
ADVECTION DIFFUSION EQ. ☐

Calc hydrostatic pressure ☐
Turbulence model:
● None
○ KE
KEClip 1e-006 ☐ Proc

Figure 2.14: *Equations window*

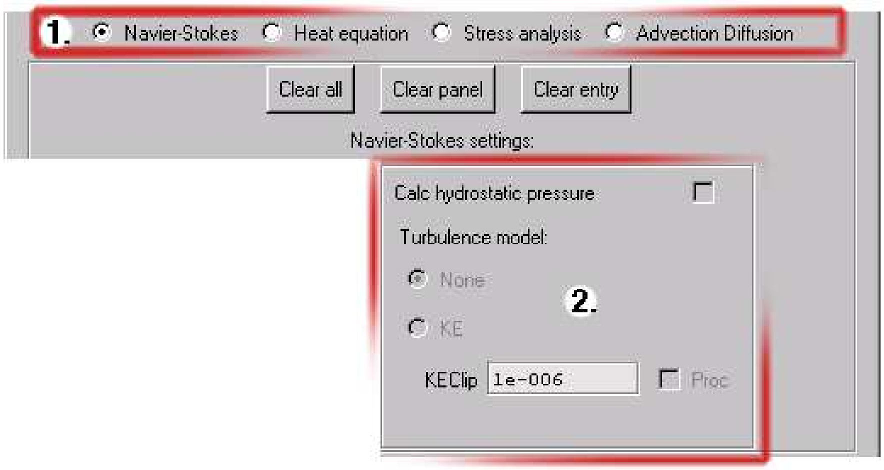
The name of the currently selected body is shown in the Body field This field is not editable.

The text for the selected body in the figure ((1)Kettle - [1]Kettle-eq), shows that the equation number 1 is attached to the body number 1. It also shows the body name (Kettle), and the equation name (Kettle-eq).

Name: in this field the user can enter or modify the name of an equation set. Pressing the Update button stores the name.

The currently defined equations are displayed in the Equation sets listbox. An automatically generated equation sequence number is shown at the beginning of each line. When an equation is attached to a body, this number is also displayed after the body name.

When parameters for a specific equation are entered, this equation must be first selected using the equation type radio buttons. Pressing a radio button on area 1 opens the appropriate input panel on area 2 where the equation is defined in detail. The Navier-Stokes equation has been checked in the example below.



● Navier-Stokes ○ Heat equation ○ Stress analysis ○ Advection Diffusion

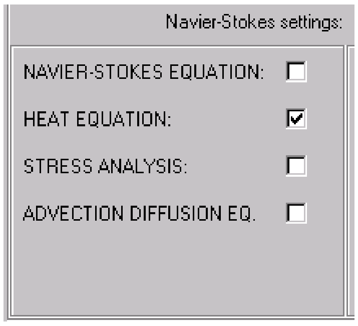
Calc hydrostatic pressure ☐

● None

○ KE

KEClip 1e-006 ☐ Proc

Separate equation types in the currently active equation set can be turned on and off by using the checkboxes in the lower left part of the window. In the example, only the heat equation is activated, others are turned off.



NAVIER-STOKES EQUATION: ☐

HEAT EQUATION: ☑

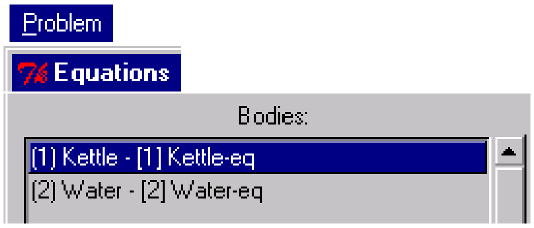
STRESS ANALYSIS: ☐

ADVECTION DIFFUSION EQ. ☐

When solving an equation attached to a body, only those equation types are taken into account whose checkboxes are turned on. When an inactive equation type is made again active, all previously defined parameter values (in the input area 2) are automatically taken into use. Making an equation inactive may also affect other equations. For example, if the flow equation is turned off, all flow related fields (like convection) are made inactive in other equations. So, it is a good idea to check all active equation types by using the equation type radio buttons, whenever an equation is made active/inactive by the checkboxes.

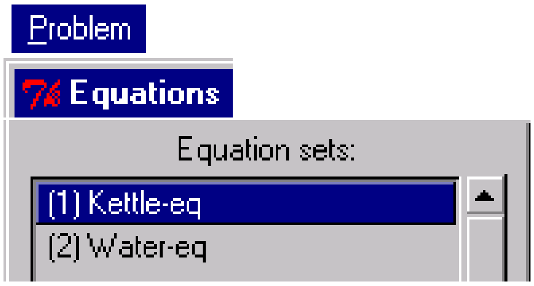
When the new equation set has been completely defined it can be added to the list of equation sets by pressing the Add button. Pressing the Update button updates the currently active equation set definition. The Delete button removes the currently active equation set from the listbox and also removes any attachments to bodies for this equation set.

2.9.7 Attachment



(1) Kettle - [1] Kettle-eq
(2) Water - [2] Water-eq

1. An equation is attached to a body by first selecting the body in the bodies list.



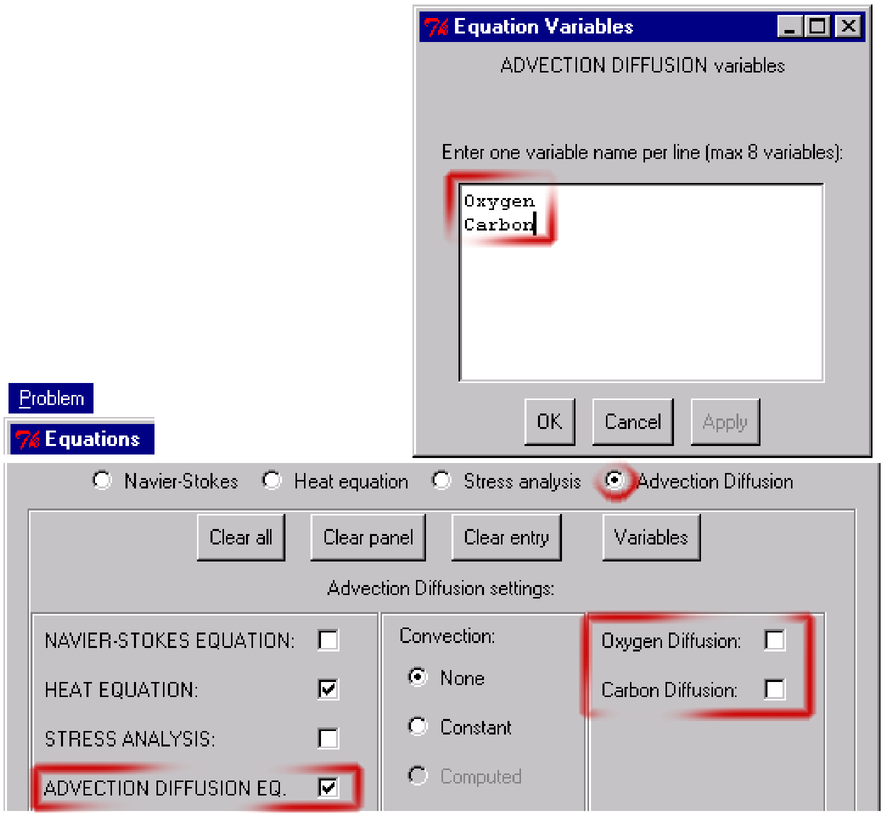
(1) Kettle-eq
(2) Water-eq

2. Next the equation is chosen.

3. Finally the attachment is done using the Attach button under the bodies listbox.

An equation may be attached to a single body or multiple bodies. There are no restrictions how bodies and equations are attached to each other, but when the attachement is done, the body inherits a kind of equation mask from the equation. This mask is transparent to the user, but it controls how other properties can be attached to this body or its boundaries. For example, if no flow equation is defined for a body, it is not possible to attach any flow related material properties for this body in the material properties panel.

2.9.8 Advection-diffusion equation



Equation Variables

ADVECTION DIFFUSION variables

Enter one variable name per line (max 8 variables):

```
Oxygen
Carbon
```

○ Navier-Stokes ○ Heat equation ○ Stress analysis ● Advection Diffusion

Advection Diffusion settings:

NAVIER-STOKES EQUATION: ☐

HEAT EQUATION: ☑

STRESS ANALYSIS: ☐

ADVECTION DIFFUSION EQ. ☑

Convection:

● None

○ Constant

○ Computed

Oxygen Diffusion: ☐

Carbon Diffusion: ☐

Figure 2.15: *Advection-diffusion equation*

When the advection diffusion equation is activated by checking the corresponding checkbox, a separate window for advection-diffusion variables is opened. In this window the user may enter or update the variable names for which the advection-diffusion equation should be solved. These variable names are displayed in the input panel area 2. for the adevection-diffusion equation and the checkboxes in this area control if these variables are active or not in the current equation set. Each active variable activates also new entry fields in other panels where advection-diffusion equation related data can be entered. Because each variable is solved by a separate solver, each active variable also creates its own entry in the Solver settings window.

This window can be reopened using the Variables button which is visible only in the advetion-diffusion equation entry panel.

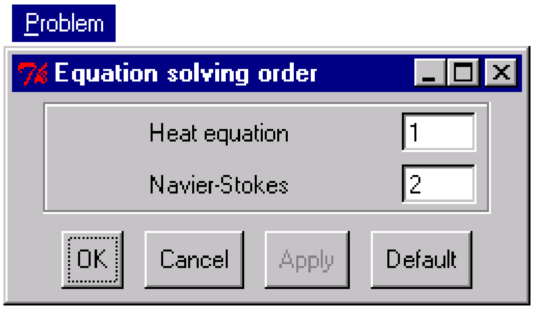
2.9.9 Data checking

As the user activates an equation, certain data integrity checks are automatically done. They are the following:

- Heat equation: heat equation includes automatically heat conduction and constant convection can be added (note: constant convection values are given in the material parameters window). The latent heat release can be selected only if some of the phase change models is activated.

- Navier-Stokes equation: when this equation is active, it is possible to select computed convection for the heat and the advection-diffusion equations.

2.9.10 Equation solving order



Heat equation 1

Navier-Stokes 2

The Equation solving order is defined in this window. The order can be changed by entering new order numbers, These values can be any numbers, only their relative order is important. Only the currently active equation types are displayed in this window.

The default solving order is the following:

1. Heat equation

2. Navier-Stokes
3. Stress analysis
4. KE Turbulence
5. Advection-diffusion equations (a separate equation for each advection-diffusion variable)

2.10 Model menu

After all body equations and other general problem data have been entered, the model parameters are entered. This is done using the model menu commands.

Figure 2.16 shows the Model menu where also the body list submenu has been opened. Clicking the dotted line on the top of this submenu opens a separate body list window on the desktop.

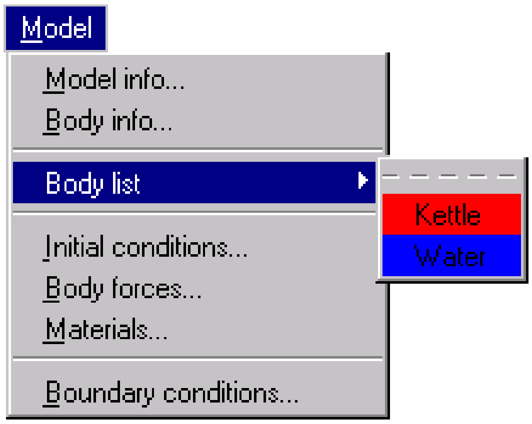


Figure 2.16: *Model menu*

2.10.1 Model info

The Model info command opens a window displaying general model information. Most of the fields are non-editable information fields but the following fields are editable:

- CAD input file: the name of the original CAD source file
- Mesh input file: the name of the external mesh input file
- Model created: model creation info

Values for these fields are set automatically when a model is created. They can be updated to reflect possible changes in the external file names, but it is not necessary because these values are not used for any actual processing of the model.

Other fields are either collected from other panels for reference or are simple model object counters, but Minimum edge size field needs some explanation. It is the smallest edge size which is used when a (linearized) CAD geometry is transmitted for the ELEMR mesh generator. This information may be useful when setting the mesh density parameters for the model.

Note: Nof is an abbreviation for 'Number of.'

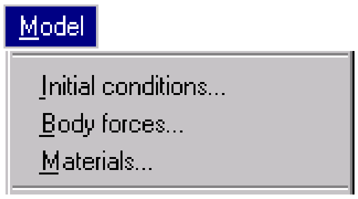
2.10.2 Body info

Body info command opens an information table where the physical dimensions and the number of mesh elements for each body can be checked. Data in this window cannot be edited.

2.10.3 Body list

Body list command opens an information table displaying a list of all model bodies. Button colors are the same as the body colors and body names are used as button labels. This list may be used to match the bodies in the data entry lists with the bodies in the Model window.

2.10.4 Initial conditions/Body forces/Materials

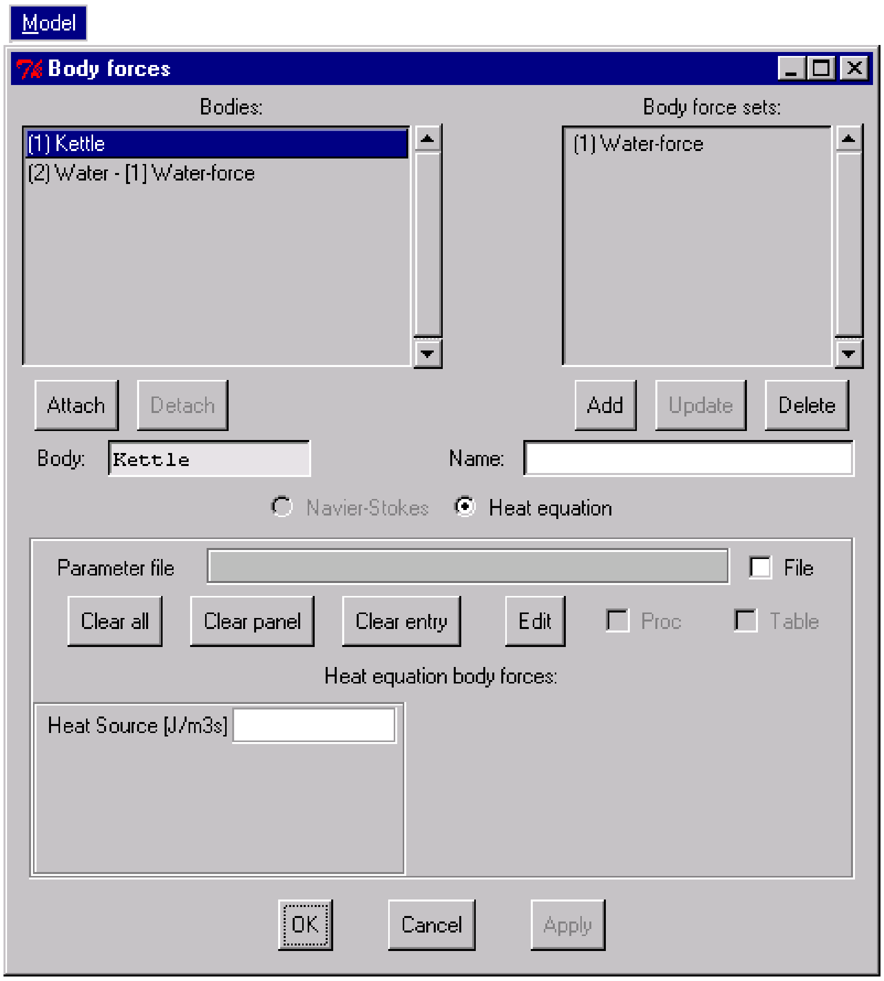


In the Initial conditions window the user can set initial values for model variables.

In the Body forces window the user can set the external forces affecting to model bodies.

In the Materials window the user can set material parameters for the bodies.

Next figures display the layout for these windows. The workflow in these windows is essentially the same and in the following the Initial conditions window is used as an example.



Bodies:
[1] Kettle
[2] Water - [1] Water-force

Body force sets:
[1] Water-force

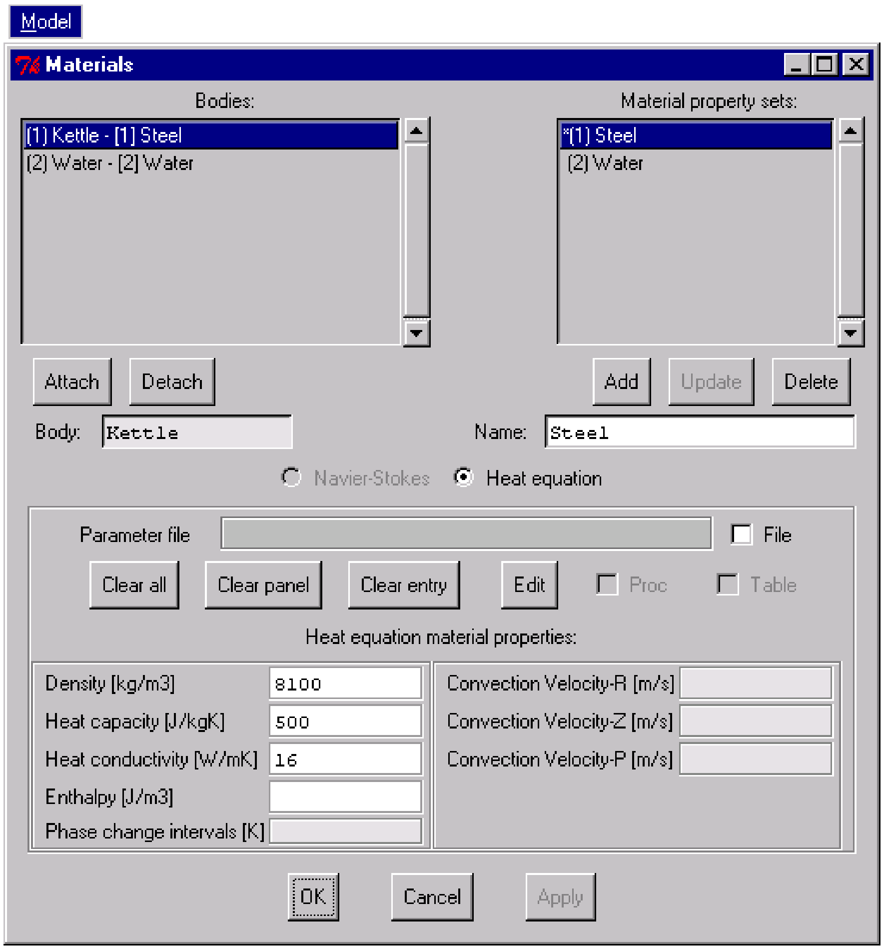
Body: Kettle

Name:

○ Navier-Stokes ● Heat equation

Heat Source [J/m3s]

Figure 2.17: *Body forces window*



Bodies:
[1] Kettle - [1] Steel
[2] Water - [2] Water

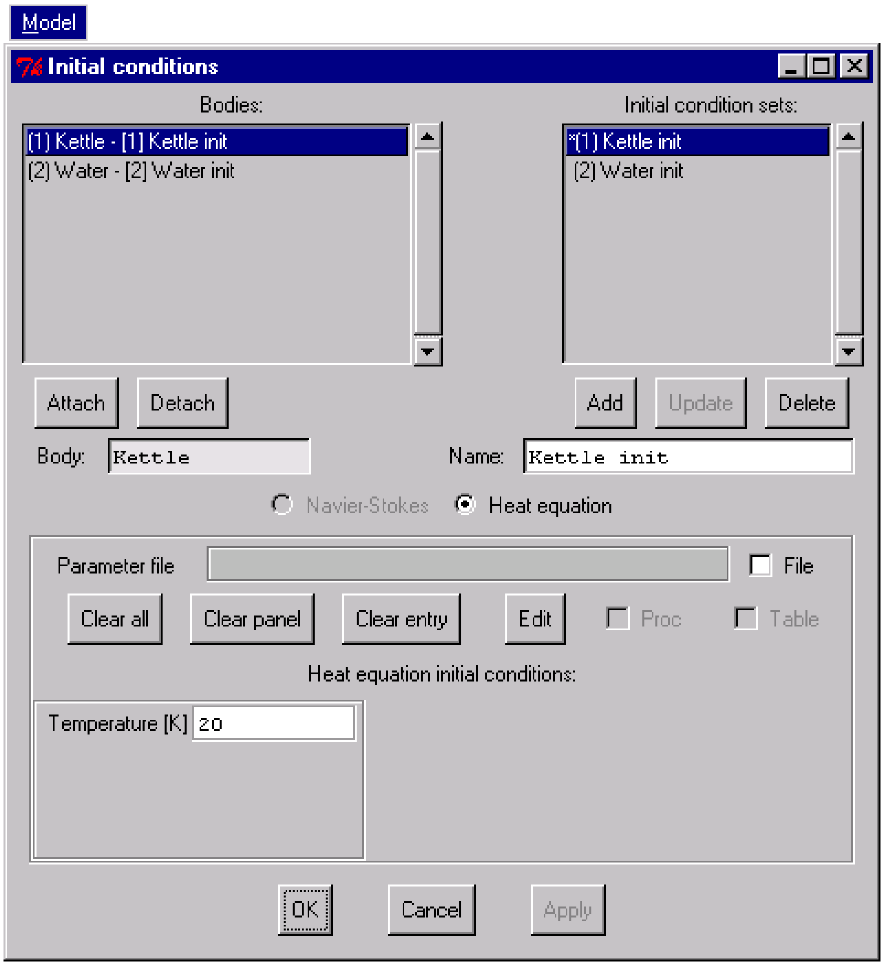
Material property sets:
*[1] Steel
[2] Water

Body: Kettle
Name: Steel

● Heat equation

Density [kg/m3] 8100
Heat capacity [J/kgK] 500
Heat conductivity [W/mK] 16

Figure 2.18: *Materials window*



Initial conditions

Bodies:

[1] Kettle - [1] Kettle init
[2] Water - [2] Water init

Initial condition sets:

*[1] Kettle init
[2] Water init

Body: `Kettle`

Name: `Kettle init`

○ Navier-Stokes ● Heat equation

Parameter file ☐ File

☐ Proc ☐ Table

Heat equation initial conditions:

Temperature [K] `20`

Figure 2.19: *Initial conditions window*

In all these windows parameter sets are attached to the bodies. The general workflow and the button properties were already described in the Equations window section.

Entry fields displayed on the window depend on the selected equation type. When a body is selected only those equation type radio buttons are active which are relevant for the body. And only those equation type buttons are visible which are relevant for the current problem. For example, if no flow equation on any of the bodies has been defined, the Navier-Stokes radio button would not be visible on these windows.

Initial conditions:

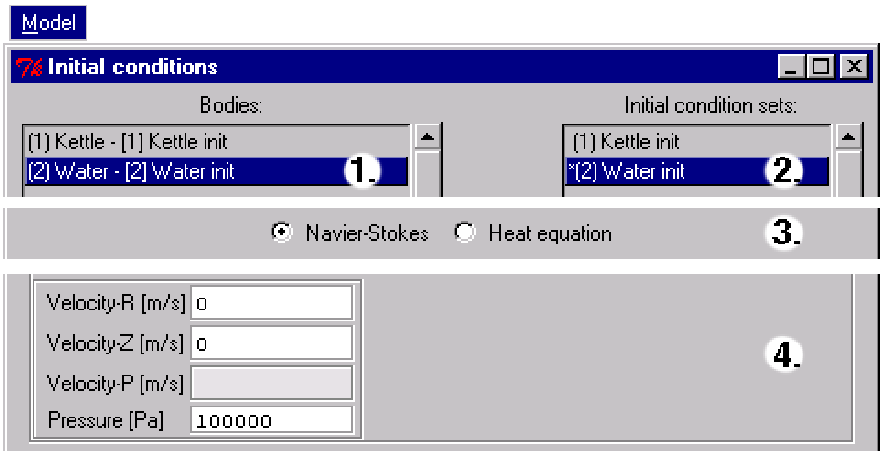
In Figure 2.19 Body1 with the initial condition set “Kettle init” has been selected. The Heat Equation radio button is shown selected.

Now the user clicks on the Body2:

1. The ‘Water’ named body is clicked on.

2. The ‘Water’ body row and the initial condition set ‘Water init’ become highlighted.

3. The Navier Stokes radio button becomes enabled, and when the user clicks it entry fields for the flow equation are displayed.



Bodies:

[1] Kettle - [1] Kettle init
[2] Water - [2] Water init

Initial condition sets:

[1] Kettle init
*[2] Water init

● Navier-Stokes ○ Heat equation

Velocity-R [m/s] 0
Velocity-Z [m/s] 0
Velocity-P [m/s]
Pressure [Pa] 100000

Figure 2.20: *Parameter selection example*

Bodies can be selected also in the Model window by double-clicking on the body area.

As a body is selected all those initial condition sets which can be attached to the body are marked with an asterisk (*). Any of the conditions marked with an asterisk matches the equation type of the currently selected body and consequently they can be attched to the body. In the example the ‘Water init’ condition set matches the ‘Water’ body.

Parameter file field:

An initial condition set can also contain a parameter file name. Name for this file is entered in the Parameter file field. This field can be activated by checking the File checkbox at the right side of the field. Unchecking the checkbox makes this field inactive and its value is not taken into acoount.

When ELMER Solver sees a parameter file name entry in the solver input file, it reads input values for this parameter set from the given file. The structure of this file should be the same as in the standard solver input file.

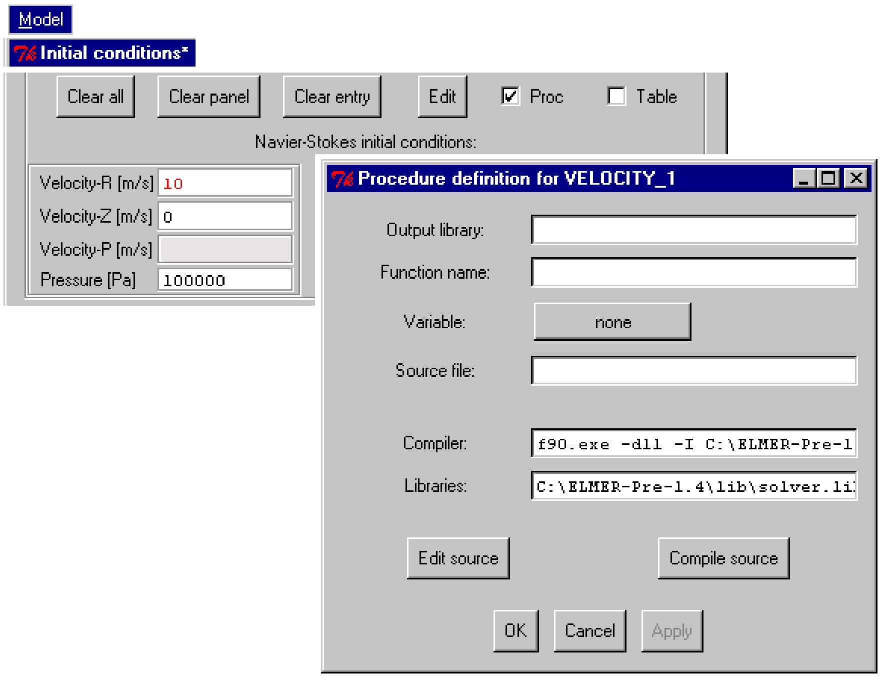
If the parameter file name does not contain any path information (i.e. when it is a simple file name), the file is searched using the Include path value given in the Model name and directories window. Otherwise the path given in the entry field is used when searching the file. (Note that the existence of the file is not checked when the data is entered in ELMER Front.)

If the initial condition set contains also other data fields, values in these fields will always overwrite corresponding values in the parameter file data.

Paramter files can be used also for body forces, material properties and boundary conditions.

Procedure entry:

- Proc checkbox: clicking this checkbox opens a separete window for procedure definition. This window is shown in Figure 2.10.4 and the fields in the window are the following:



☑ Proc
☐ Table

Velocity-R [m/s] 10
Velocity-Z [m/s] 0
Velocity-P [m/s]
Pressure [Pa] 100000

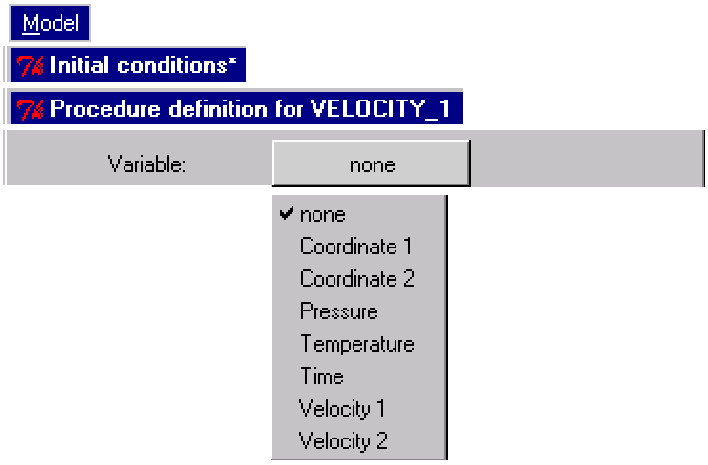
Variable: none
Compiler: f90.exe -dll -I C:\ELMER-Pre-1
Libraries: C:\ELMER-Pre-1.4\lib\solver.li

- Output library: the library where the procedure being defined is stored

- Function name: the call name for the procedure in the library

- Variable: the possible argument variable can be selected by pressing this button. This variable can be any of the variables solved in the model, any of the model coordinates and for a transient problem it can also be time.

Figure 2.21 shows the variables available for a transient 2D problem with heat and flow equations.



Variable: none

Figure 2.21: *Argument variable list*

- Source file: the source file for the procedure

- Libraries: possible external libraries needed by the compiler

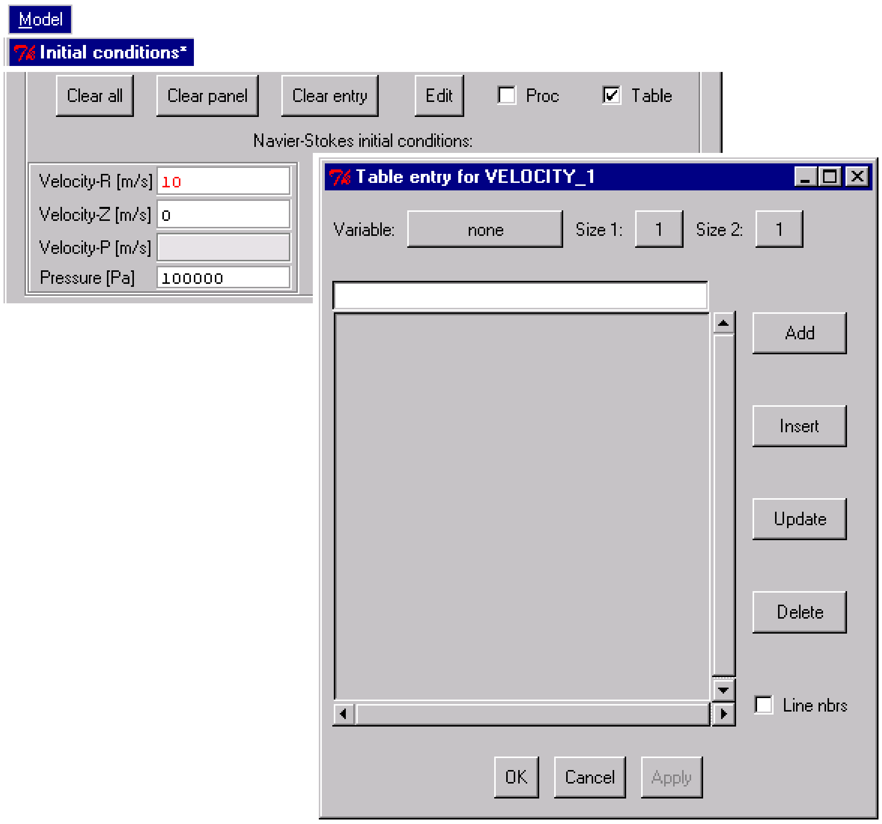
- Compiler: the compiler command script

- Edit source button: opens the procedure source file for editing.

- Compile source button: compiles the source file and append the procedure to the libary

Table entry:

- Table checkbox: clicking this checkbox opens a separete window for entering a data table. This window is shown in Figure 2.10.4 and the fields are the following:



☐ Proc ☑ Table

Velocity-R [m/s] 10
Velocity-Z [m/s] 0
Velocity-P [m/s]
Pressure [Pa] 100000

Variable: none Size 1: 1 Size 2: 1

☐ Line nbrs

- Variable: a possible argument variable for the table

- Size 1: if the parameter is not a scalar, this is the first dimension of the data

- Size 2: if the parameter is not a scalar, this is the second dimension of the data

For a scalar field (like density) these values are always one. Argument variables are always scalars.

- Entry field: the data in the table is entered using this field. First value should be always for the argument variable if it is defined. After that come the data values. The number of data values should be $Size1 \times Size2$ and these values should be separated by spaces

- Add: a new row is added at the end of the table

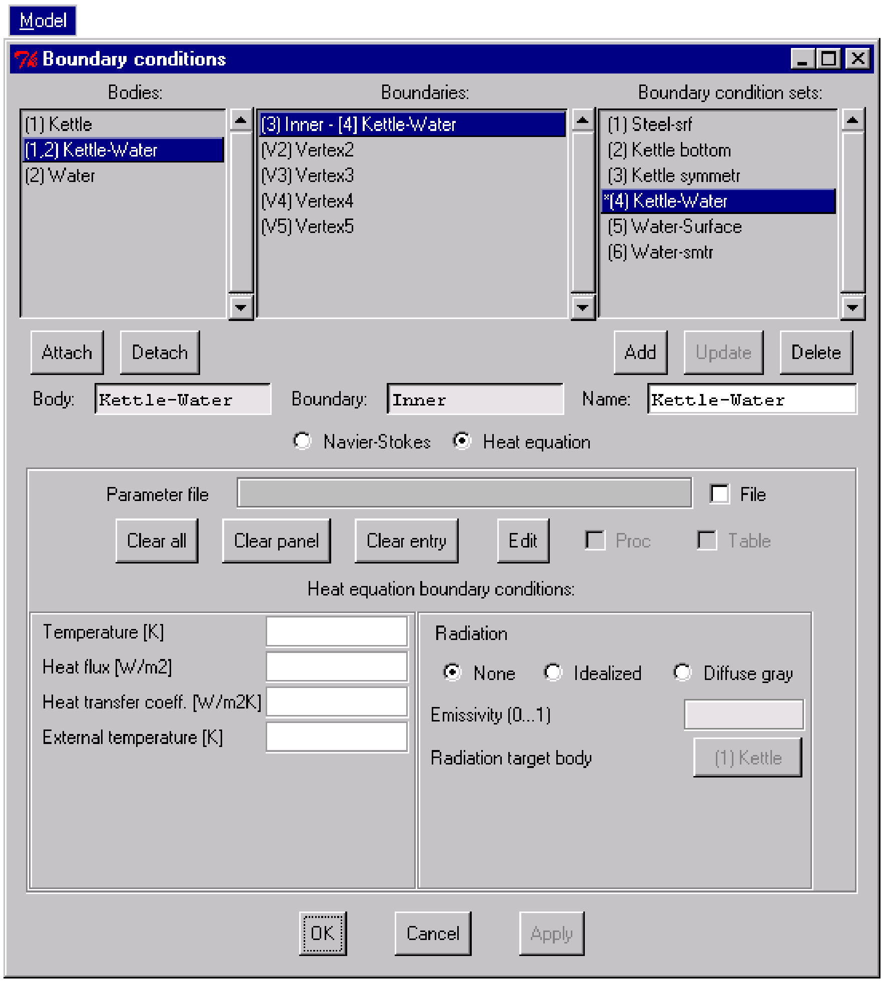
- Insert: a new row is added above the currently selected row

- Update: updates the currently selected row

- Delete: deletes the currently selected rows
- Line nbrs: when this checkbutton is checked a line number is displayed at left side of each row

2.10.5 Boundary conditions

The Boundary conditions window differs from the previous windows as boundary conditions are applied to boundaries and other elements defining the bodies, like edges and vertices. Thus, in addition to the bodies and boundary conditions listboxes, this window contains a third listbox, the boundaries listbox.



Bodies:
[1] Kettle
[1,2] Kettle-Water
[2] Water

Boundaries:
[3] Inner - [4] Kettle-Water
[V2] Vertex2
[V3] Vertex3
[V4] Vertex4
[V5] Vertex5

Boundary condition sets:
[1] Steel-srf
[2] Kettle bottom
[3] Kettle symmetr
*[4] Kettle-Water
[5] Water-Surface
[6] Water-smtr

Body: Kettle-Water
Boundary: Inner
Name: Kettle-Water

○ Navier-Stokes ● Heat equation

Parameter file ☐ File
☐ Proc ☐ Table

Heat equation boundary conditions:

Temperature [K]
Heat flux [W/m2]
Heat transfer coeff. [W/m2K]
External temperature [K]

Radiation
● None ○ Idealized ○ Diffuse gray
Emissivity (0...1)
Radiation target body: (1) Kettle

Figure 2.22: *Boundary conditions window*

Entry fields are grouped by the equation type also in this window. If the problem contains more than one equation type the radio buttons in the middle of the window are used to select the equation.

When a body is selected in the bodies listbox, all boundary elements (boundaries, edges and vertices) for the body are displayed in the boundaries listbox. An asterix is again used to mark the boundary conditions which can be attached to the boundaries of the selected body.

Selecting a body displays only outer boundaries for this body. These are boundaries which are not shared between bodies. Boundaries bewteen two bodies are called inner boundaries and they are grouped under body pairs. A body pair inherits equation types from both parent bodies and body pairs are listed just like normal bodies in the bodies listbox.

Boundary elements can be grouped into boundaries (faces in 3D and edges in 2D) and into subelements like edges (in 2D) and vertices. Normally boundary conditions are attached directly to boundaries, but this can create conflicts at common vertices (and at common edges in 3D) if Dirichlet-type conditions are applied. For example, if a temperature constraint has different values in two boundaries which have a common vertex, the final value at this vertex is not unique. In ELMER Solver boundary conditions are applied in the order they are created in ELMER Front. So, this offers a simple method to control these conflicting values. The last value will be used.

However, it is sometimes important to control constraints at edges and vertices more exactly, and for these cases the option to define boundary condtions at that specific level is needed. It should be also noted that it is not possible to attach a boundary level condition to edges or vertices and vice versa, because this conditions could contain values which are not applicable at these lower geometrical level. For example, it would not be meaningful to apply a flux constraint on a vertex.

Radiation boundaries

For a diffuse-gray radiation boundary, one has to define a radiation target body. For an outer boundary the target can be either the body itself or the 'Outside' environment. For an inner boundary, the target body must be one of the parent bodies. It is possible to apply a constraint defined at a boundary to boundaries in other bodies having similar equations, as long as the target body is the same. However, if a constraint is applied to multiple bodies, it is not any more possible to change the target body, because this would create contradictions. The only way to change the target body in these cases is to detach attachements until the constraint is again attached to the boundaries of only one body or body pair.

2.11 Mesh menu

2.11.1 Background

ELMER Mesh supports triangular and quadrilateral 2D element meshes. Elements can be linear or parabolic and different bodies can have different mesh types. However, element types cannot be mixed within a body.

The structure of a triangular mesh is free. This means that element size can change freely over the body. On the other hand, quadrilateral meshes are always structured. This means that a quadrilateral mesh can be defined only for those bodies which are quadrilaterals, ie. for bodies which have four boundaries. These boundaries can be polylines which consit of multiple segments, but topologically they must define a quadrilateral geometry.

Triangular meshing is controlled by the mesh density values which are defined at body vertices. During the triangulation process these densities are interpolated over the body and these values control the mesh density in different parts of the body. The final structure of the mesh may depend on the actual meshing method, but the basic principle is the same.

When defining the mesh, it is possible to enter these density values for each vertex in the model. However, for convenience data entry is organized so that density values can be entered either at model, body, boundary or vertex level, whatever is suitable for the problem. Whenever a lower level value is not defined, it is inherited from the upper level. So, it is possible to enter just one model level density parameter and create a constant density mesh for the whole model.

In a structured mesh opposing boundaries are always devided into same number of mesh segments (mesh boundary elements). The structure is thus defined by two numbers which tell the number of elements per boundary pairs. It is possible to define a structured mesh for a body and a triangular mesh for a body which is adjacent to this body. However, in these cases the structured mesh defines automatically the mesh density at the common boundary and the resulting triangular mesh may become quite artificial. So, mixing element types in one model is possible and in many cases quite useful, but it must be done carefully.

These concepts are highlighted in the example shown in Figure 2.24.

Labels

In the Model window normal boundaries are labeled by sequence numbers and vertices with the sequence number preceded by the letter V.

Mesh element types

ELMER Mesh2D supports the four mesh element types shown in Figure 2.23. `A` is a linear quadrilateral element, `B` is a parabolic quadrilateral element, `C` is a is linear triangular element and `D` is a parabolic triangular element.

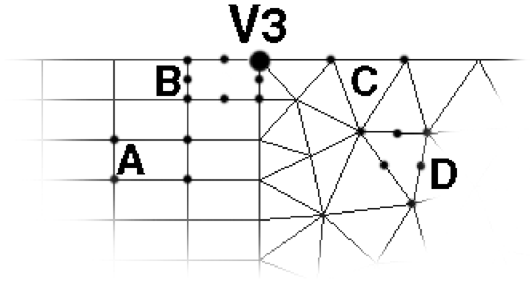


Figure 2.23: *Mesh element types*

2.11.2 The three mesh definition levels:

- The Model Level (L1)
- The Body Level (L2)
- The Boundary Level (L3) - including also edges and vertices

If no definitions has been made on (L2) and (L3) levels, the model level (L1) parameters are set as default to all meshes included in the model.

2.11.3 Mesh example

Figure 2.24 shows an example where each body of the model has a different mesh type. Body1 has a quadrialteral (structured) mesh and

Body2 has a triangular (free) mesh.

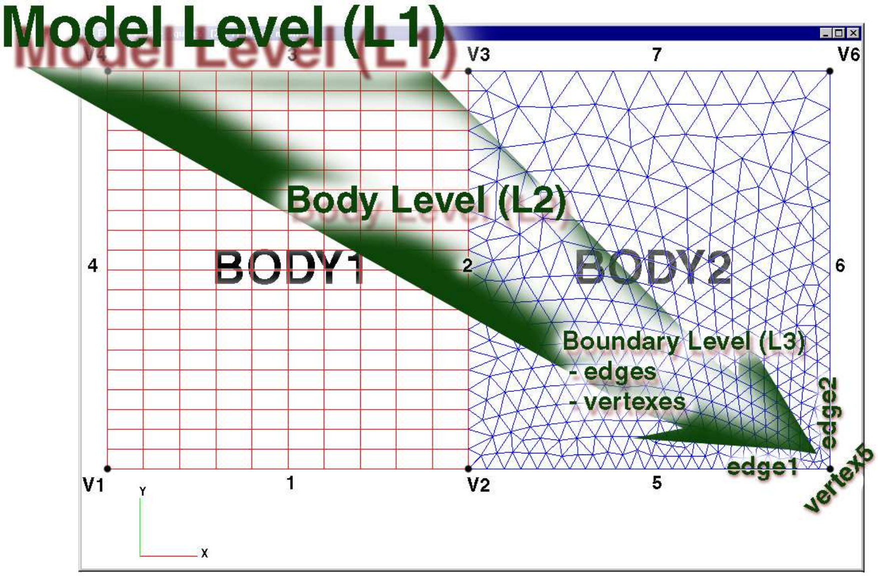


Figure 2.24: *Mesh example*

The arrow from the top left down to the right corner highlights the relation between different levels. The boundary level (L3) (edges and vertices) inherits the appropriate body level (L2) mesh density value if no definitions for them has been made. Similarily the body level inherits the model level (L1) density value.

BODY 1:

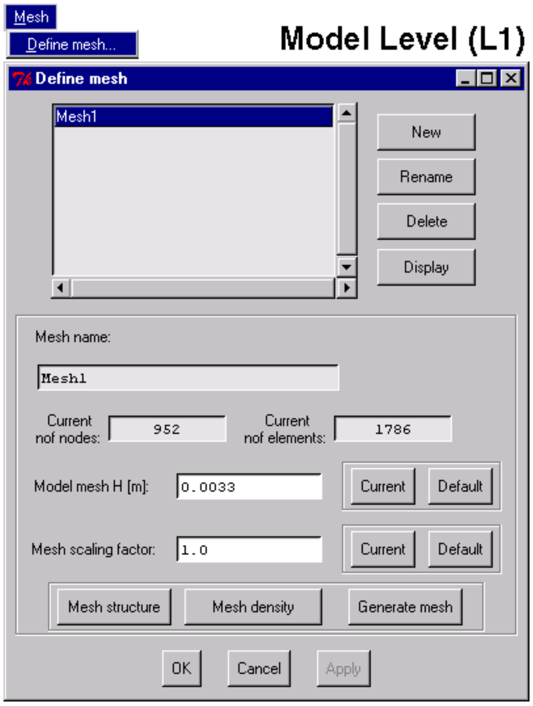
For Body1 a quadrilateral mesh has been defined. Edges 1 and 3 are devided into 10 segment and edges 2 and 4 into 20 segments and the total number of elements is 200.

BODY 2:

For Body2, the user has defined a triangular mesh. Vertex5 has its own mesh density value and the mesh is denser around this vertex.

If the user would not have defined any value for Vertex5, it would have inherited an average value derived from the density values at the edges 5 and 6.

2.11.4 Define mesh



Mesh1

Mesh name: Mesh1

Current nof nodes: 952

Current nof elements: 1786

Model mesh H [m]: 0.0033

Mesh scaling factor: 1.0

Figure 2.25: *Define mesh window*

ELMER Solver has a multimesh capability. This means that different equations can be solved using different meshes. If variable values from one equation are needed in other equations during the simulation, they are interpolated from one mesh to an other. This is completely transparent to the user, it is only necessary to define the mesh for each equation.

The Define mesh command in the Mesh menu opens a window where one can define all the meshes needed in the problem. These meshes must have unique names. These names are used also in the Solver settings window where the mesh for each equation is defined.

In the upper part of the window is a listbox where all currently defined meshes are listed. When a row in this listbox is selected, the corresponding mesh is made active and its parameter definition can be checked or updated and the mesh can be generated or regenerated using the Generate mesh button at the bottom of the window. When a new mesh is being defined, the user has to first press the New mesh button. This activates the mesh name field where one can now enter a new unique mesh name. This new mesh is added to the mesh list when the mesh is generated.

The button and fields in the window are the following:

- New: a name for a new mesh can be entered

- Rename: a new name for the currently selected mesh can be entered

- Delete: the currently selected mesh is deleted

- Display: the currently selected mesh is displayed in the Model window

- Mesh name: the name for the selected mesh is displayed in this field and it is used for entering and renaming the mesh name as explained above

- Current nof nodes: the total number of nodes in the selected mesh

- Current nof elements: the total number of elements in the selected mesh

- MeshH value [m]: the default mesh element size for the whole model is defined with this value. Values defined at lower level will overwrite this value. Default value is derived from the average dimension of the model.

- Mesh scaling factor: all mesh density values given, derived or inherited in the model are finally multiplied with this factor. Default value is 1.0

- Current: this button sets the field value into the currently active value

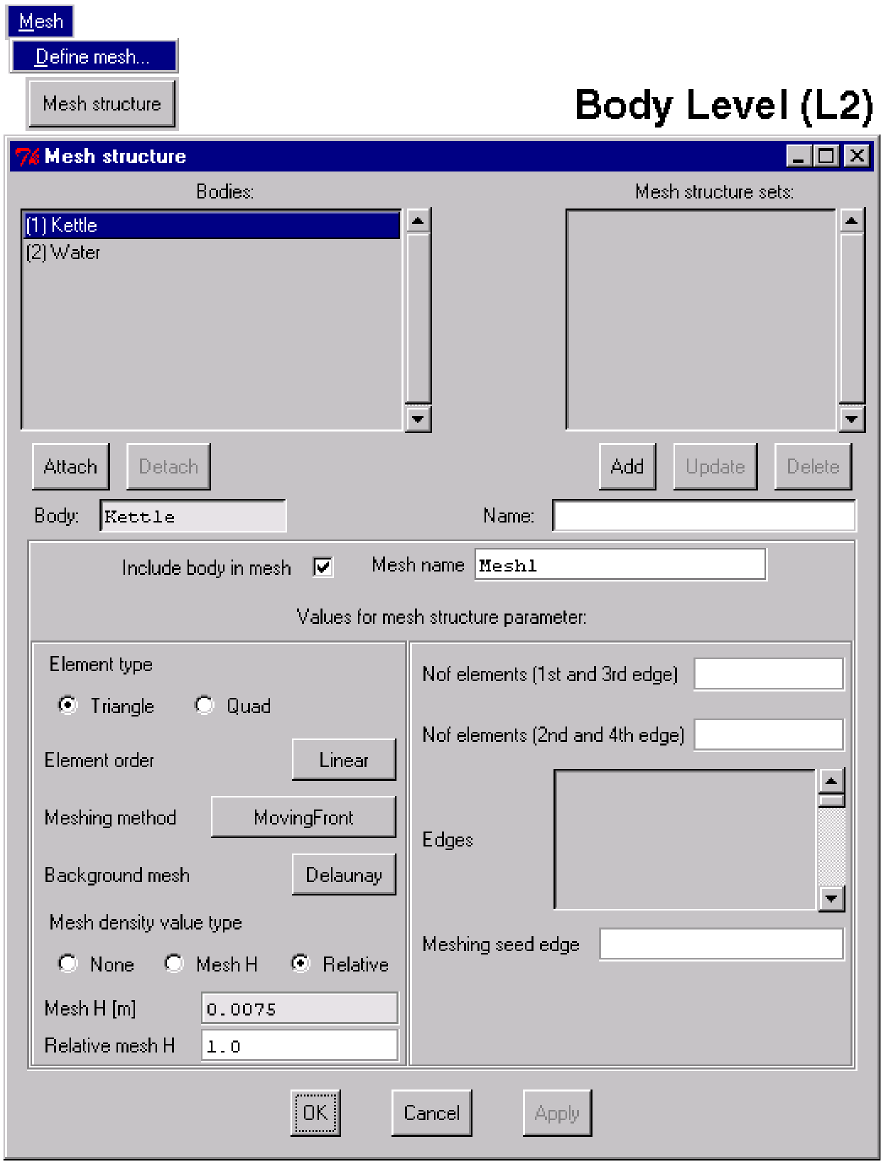
- Default: this button sets the field value into the default value

- Mesh structure: this button opens a window for entering body level mesh definitions (L2)

- Mesh density: this button opens a window for entering boundary level mesh definitions (L3)

- Generate mesh: this button will generate the mesh using the current mesh definition values. Note that a new mesh is not added to the list of meshes until this button is pressed

2.11.5 Mesh structure



Bodies: (1) Kettle, (2) Water

Body: Kettle

Name:

Include body in mesh ☑ Mesh name Mesh1

Element type ● Triangle ○ Quad

Element order Linear

Meshing method MovingFront

Background mesh Delaunay

Mesh density value type ○ None ○ Mesh H ● Relative

Mesh H [m] 0.0075

Relative mesh H 1.0

Nof elements (1st and 3rd edge)

Nof elements (2nd and 4th edge)

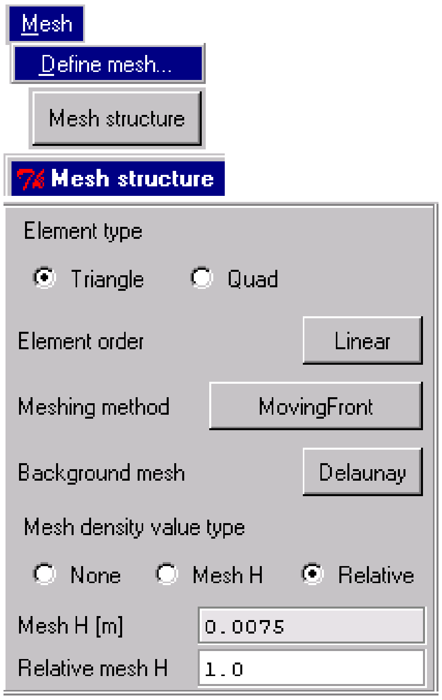
Meshing seed edge

Figure 2.26: *Mesh structure window*

The Mesh structure window opens by clicking on the Mesh structure button in the Define mesh window. Parameters entered in this window define the mesh at body level (L2).

On top left is the list of all bodies and the mesh structure sets attached to them. The name of the currently active mesh is displayed in the

Mesh name field. This field is not editable. The selected body can be included or excluded from the mesh by using the Include body in mesh checkbox.



Element type

● Triangle ○ Quad

Element order: Linear

Meshing method: MovingFront

Background mesh: Delaunay

Mesh density value type

○ None ○ Mesh H ● Relative

Mesh H [m]: 0.0075

Relative mesh H: 1.0

The mesh element type can be selected using the Triangle and Quad radio buttons. If the quadrilateral mesh type is selected the mesh size parameter entries on the right side are activated as shown in Figure 2.11.5.

- Element order: Linear setting creates three nodes triangular and four nodes quarilateral elements. Parabolic setting creates six nodes triangular and eight nodes quadrilateral elements

- Meshing method: Basic meshing strtegies for free (triangular) meshes are moving front methods and the Voronoi vertex method. Default value is the standard MovingFront method. Other moving front options are SSSFMovingFront (single seed, single front) and SSMFMovingFront (single seed, multi front) methods. These methods need a starting edge as seed. The boundary id for this seed can be entered in the Meshing seed edge field.

- Background mesh: Free (triangular) meshing starts from an automatically created background mesh. Delaunay is the default method. Other options are regular background grids: Grid, SparseGrid and DenseGrid. A dense background grid creates better quality meshes, but it is also a slower method. SparseGrid is faster, but the mesh quality may be lower.

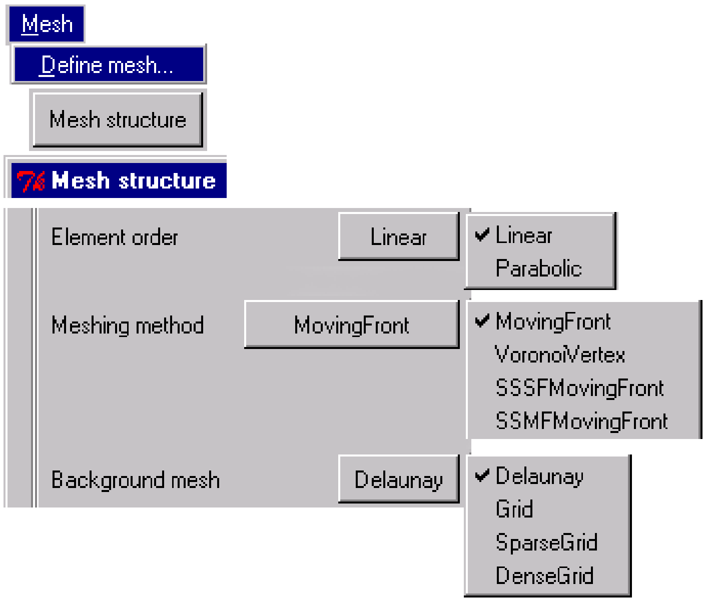
Options for these buttons are displayed in Figure 2.11.5.

Mesh density value type radio buttons:

- None: deactivates values on this level

- Mesh H: activates the Mesh H field to be used for entering an absolute mesh density value in this level

- Relative: activates the Relative mesh H field to be used for entering a scaling factor for mesh density in this level. The actual mesh density value for this level is the product of the scaling factor and the model level density value.

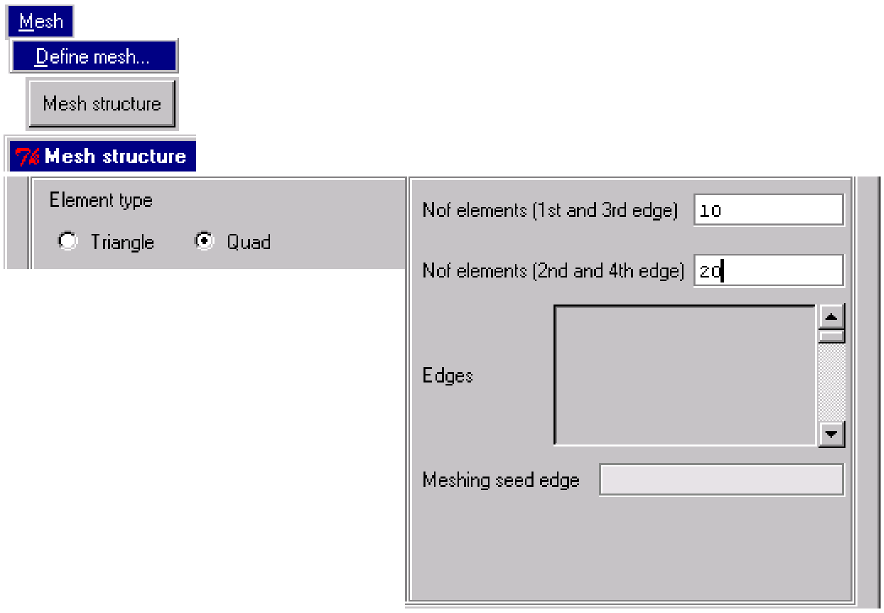


Element order Linear

Meshing method MovingFront

Background mesh Delaunay

When the Quad mesh element type is selected, the fields at right are activated for input.



Element type

○ Triangle ● Quad

Nof elements (1st and 3rd edge) 10

Nof elements (2nd and 4th edge) 20

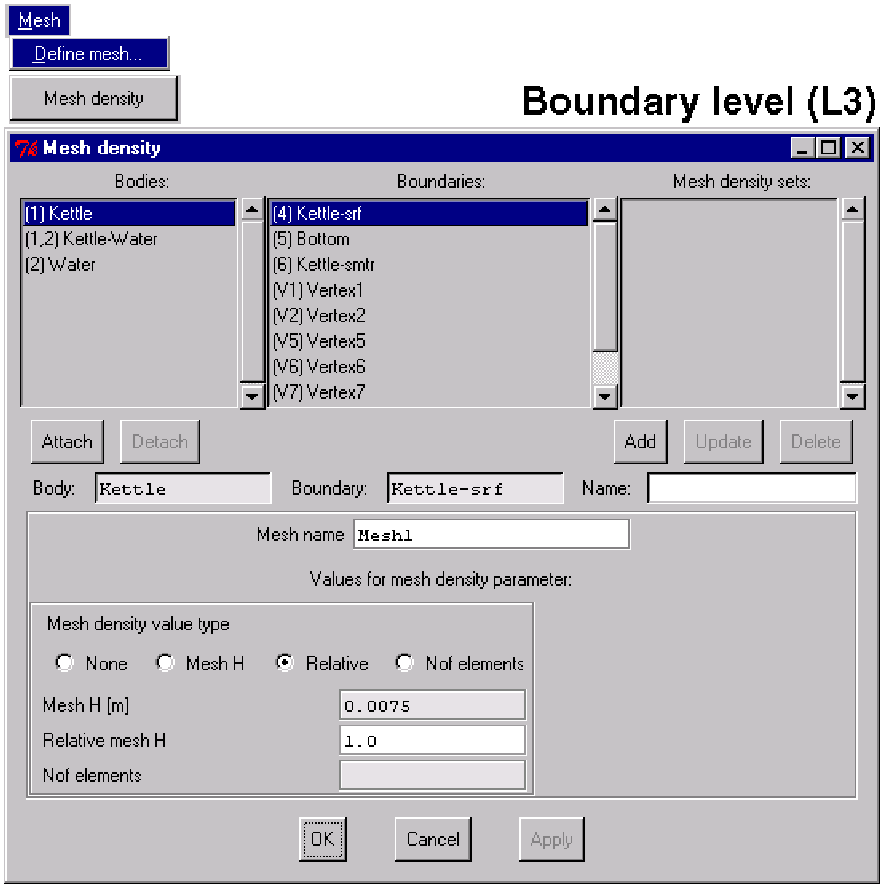
Meshing seed edge

- Nof elements (1st and 3rd edge): the number of elements (segements) for the first and third edges in the Edges listbox

- Nof elements (2nd and 4th edge): the number of elements (segements) for the second and fourth edges in the Edges listbox

2.11.6 Mesh Density

The Mesh density window opens by pressing the Mesh density button in the Define Mesh window. These definitions affect the mesh at the boundary level (L3). Mesh density values can be set at edges or at vertices. However, a parameter set defined for an edge cannot be applied to a vertex and vice versa. They have different 'masks'. This is mainly because a 'nof elements'-type definition does not have any meaning at vertex level.



Bodies:
- [1] Kettle
- [1,2] Kettle-Water
- [2] Water

Boundaries:
- [4] Kettle-srf
- [5] Bottom
- [6] Kettle-smtr
- [V1] Vertex1
- [V2] Vertex2
- [V5] Vertex5
- [V6] Vertex6
- [V7] Vertex7

Body: Kettle
Boundary: Kettle-srf
Name:

Mesh name Mesh1

Values for mesh density parameter:

Mesh density value type
○ None ○ Mesh H ● Relative ○ Nof elements

Mesh H [m] 0.0075
Relative mesh H 1.0
Nof elements

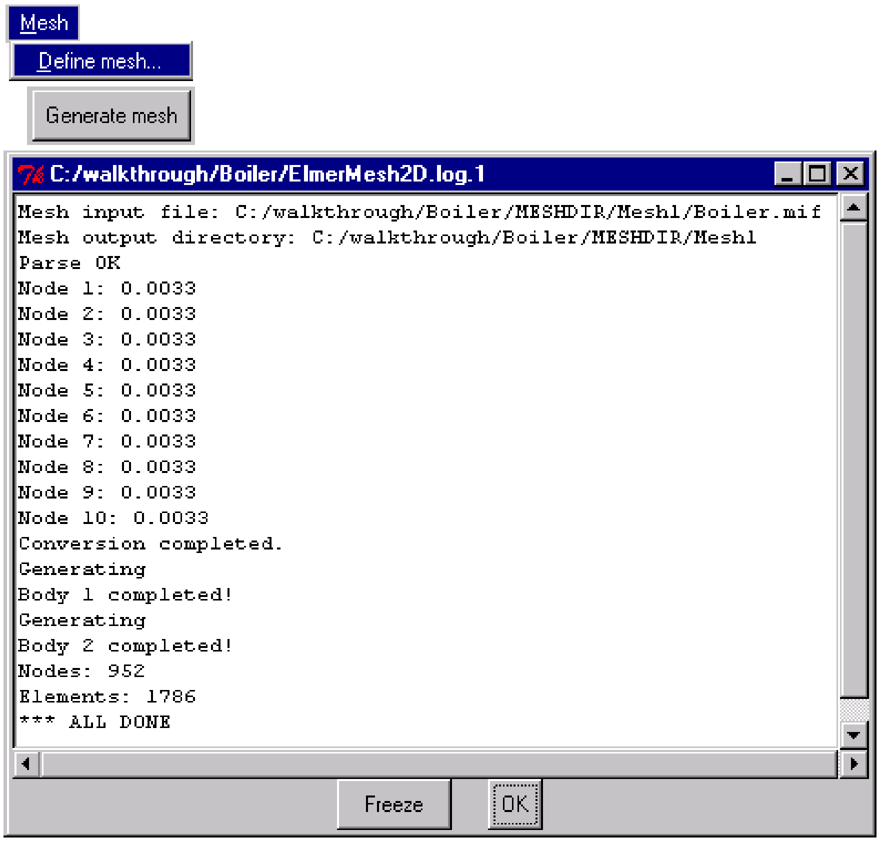
Figure 2.27: *Mesh density window*

Mesh density value type radio buttons are used to select between three different input options:

- None: deactivates values on this level
- Mesh H: activates the Mesh H field to be used for enetering an absolute mesh density value in this level
- Relative: activates the Relative mesh H field to be used for enetering a scaling factor for mesh density in this level. The actual mesh density value for this level is the product of the scaling factor and the body (or model) level density value.
- Nof elements: the fixed number of elements (segments) at the edge

Mesh log file

When the mesh is generated a log file is created (if this process output setting is activated). An example of this file is shown in Figure 2.28. This and other logfiles may be viewed using the Process table command in the Run menu.



C:/walkthrough/Boiler/ElmerMesh2D.log.1

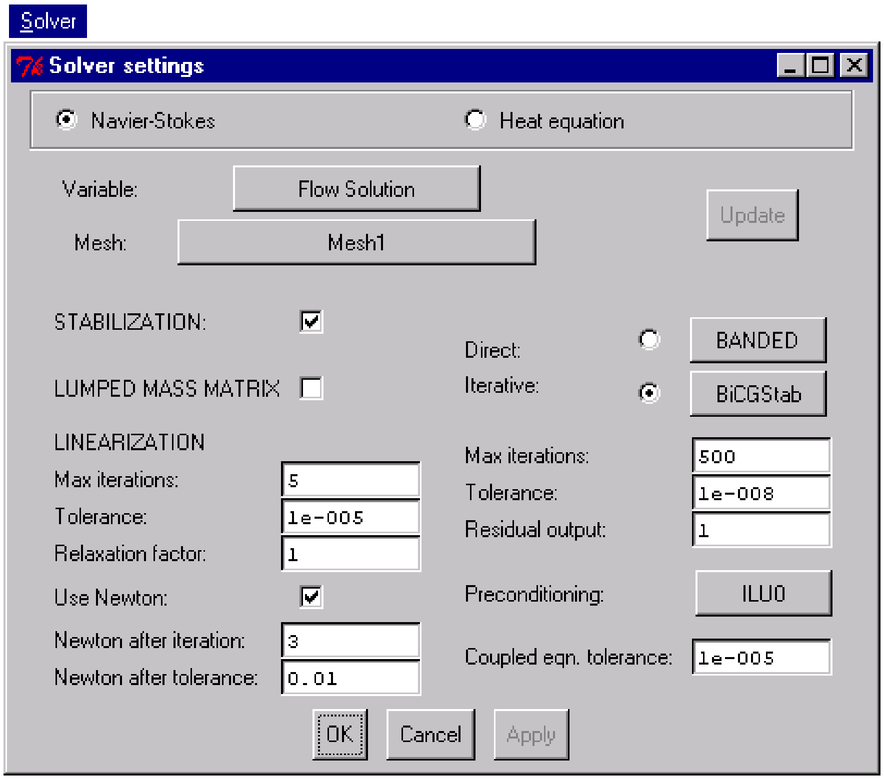
```
Mesh input file: C:/walkthrough/Boiler/MESHDIR/Mesh1/Boiler.mif
Mesh output directory: C:/walkthrough/Boiler/MESHDIR/Mesh1
Parse OK
Node 1: 0.0033
Node 2: 0.0033
Node 3: 0.0033
Node 4: 0.0033
Node 5: 0.0033
Node 6: 0.0033
Node 7: 0.0033
Node 8: 0.0033
Node 9: 0.0033
Node 10: 0.0033
Conversion completed.
Generating
Body 1 completed!
Generating
Body 2 completed!
Nodes: 952
Elements: 1786
*** ALL DONE
```

Figure 2.28: *Mesh logfile example*

2.12 Solver menu

2.12.1 Solver settings

The Solver settings window is shown in Figure 2.29. Settings for the Navier-Stokes equation are displayed. In this example all fields have their default values.



● Navier-Stokes ○ Heat equation

Variable: Flow Solution

Mesh: Mesh1

STABILIZATION: ☑

LUMPED MASS MATRIX ☐

LINEARIZATION

Max iterations: 5

Tolerance: 1e-005

Relaxation factor: 1

Use Newton: ☑

Newton after iteration: 3

Newton after tolerance: 0.01

Direct: ○ BANDED

Iterative: ● BiCGStab

Max iterations: 500

Tolerance: 1e-008

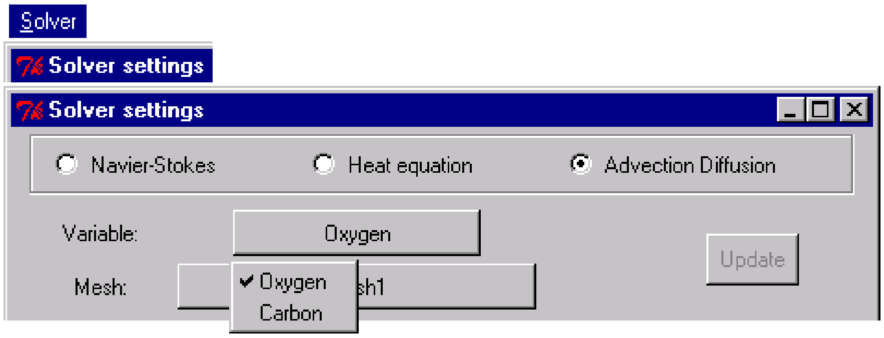
Residual output: 1

Preconditioning: ILU0

Coupled eqn. tolerance: 1e-005

Figure 2.29: *Solver settings window*

Each equation system in the model is solved by a separate solver. These solvers can selected using the radio buttons on the top of the window. The advection-diffusion equation differs from other equations because it can solved for multiple variables. Each of these variables is solved by a separate solver. These solvers can be selected by pressing the Variable button and selecting the corresponding variable from the list.



○ Navier-Stokes ○ Heat equation ● Advection Diffusion

Variable: Oxygen

Mesh: sh1

Figure 2.30: *Advection-diffusion solver example*

When all data fields for the solver are entered or modified, the settings for the solver are stored by the Update button. All solvers have initial values which are adequate for standard cases. However, it may be that the solution for an equation does not converge or converges slowly, and in these cases it is a good idea to test with different parameter values.

Fields in the window are grouped so that fields related to the general form and linearization methods of the equation are in the left side of the window. Fields related to the linear system and its solving methods are on the right side of the window.

STABILIZATION: when selected, solver uses stabilization. It is a recommended option for the Navier-Stokes equation

LUMPED MASS MATRIX: the matrix corresponding to the mass matrix is lumped in the equation

LINEARIZATION:

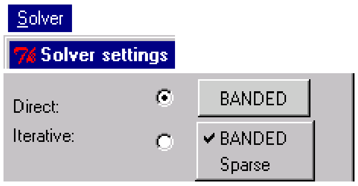
- Max iterations: linearization is based on subiterations and this number sets the maximum number for these iterations
- Tolerance: if the linearization error norm is smaller than this number, linearization iteration is stopped
- Relaxation factor: default value is 1.0. Linearization could be speeded up in some cases using somewhat larger value. On the other hand, sometimes a value less than unity is needed to get any solution

Use Newton: when selected Newton's linearization method is used. It is faster than the default method (Picard), but is useful only when the actual solution is close enough to the exact one. The following fields control when Newton's method is started:

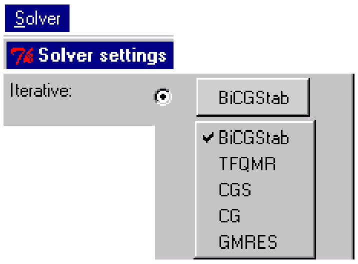
- Newton after iteration

- Newton after tolerance: Newton's method is not started until at least one of these criteria is fulfilled

- Direct: this radio button selects a direct solving method. For relatively small problems a direct solver is normally faster. For any larger problem it is usually better to use an iterative solver. The version for the direct solver can be selected pressing the button at the right side of the radio button.



- Iterative: this radio button selects an iterative solving method. Current options are the following:



- BiCGStab: BiConjugate Gradient Stabilized
- TFQMR: Transpose Free Quasi-Minimal Residual
- CGS: Conjugate Gradient Squared
- CG: Conjugate Gradient
- GMRES: Generalized Minimal Residual

- Max iterations: the maximum number of iterations for the iterative solver

- Tolerance: an error norm threshold. If the error norm is smaller than this number, iteration is stopped

- Residual output: an integer value; linear system residual is output for each Nth iteration (zero value disables the output)

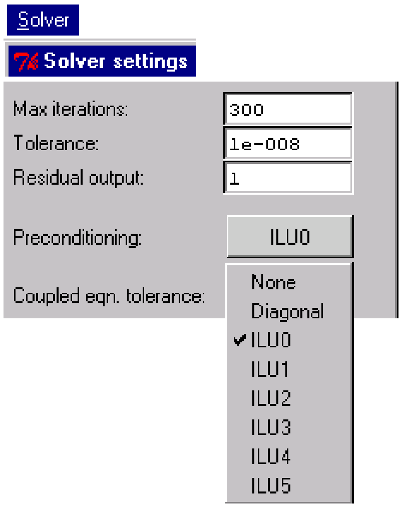
- Preconditioning: using preconditioning normally speeds up the iteration, however, resulting in some overhead. Options here are:

- None: no preconditioning

- Diagonal: Diagonal preconditioning matrix. Fast construction, but not very effective

- ILU: Different level of Incomplete LU decompositions. ILU0 is the recommended choice

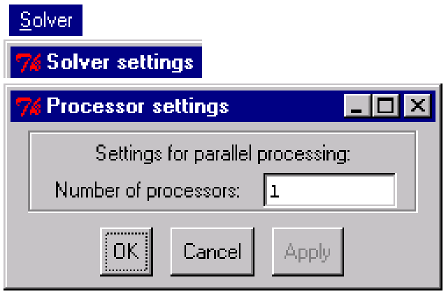
- Coupled eq. tolerance (or Steady state tolerance): for transient problems this field sets the tolerance for stopping the iteration when multiple equations are coupled to each other. For steady state problems it sets the tolerance for stopping the steady state iteration



Max iterations: 300
Tolerance: 1e-008
Residual output: 1
Preconditioning: ILU0
Coupled eqn. tolerance:

2.12.2 Processor settings

This command opens a window for setting the number of processors reserved for the problem. This parameter has meaning only in a parallel processing environment.



Settings for parallel processing:
Number of processors: 1

Figure 2.31: *Processor settings window*

2.13 Run menu

The Run menu commands are used to start other ELMER modules. Before a module can be started, all the necessary definitions and settings should be completed. If this not the case, an error message box is displayed. If the missing data is essential for the module, the process cannot continue.

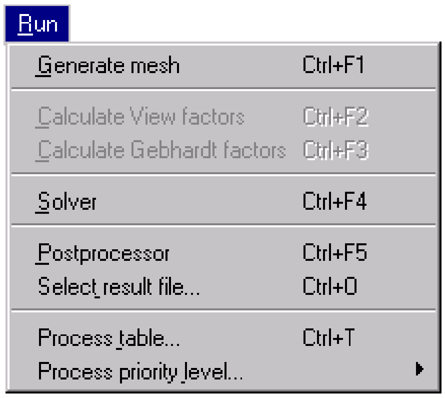


Figure 2.32: *Run menu*

2.13.1 Generate mesh:

Generate mesh command starts the ELMER mesh generator. it creates a browsable log file.

2.13.2 Calculate View factors:

Calculate View factors command starts ELMER View factors program. It stores the result data in the View factors file defined in the Problem / Datafiles window. **Note: for large meshes this can be a time consuming process.** It creates a browsable logfile.

2.13.3 Calculate Gebhardt factors:

Calculate Gebhart factors command starts ELMER Gebhardt factors program. It stores the data in the Gebhardt factors file defined in the Problem / Datafiles window. **Note: for large meshes this can be a time consuming process.** It creates a browsable logfile.

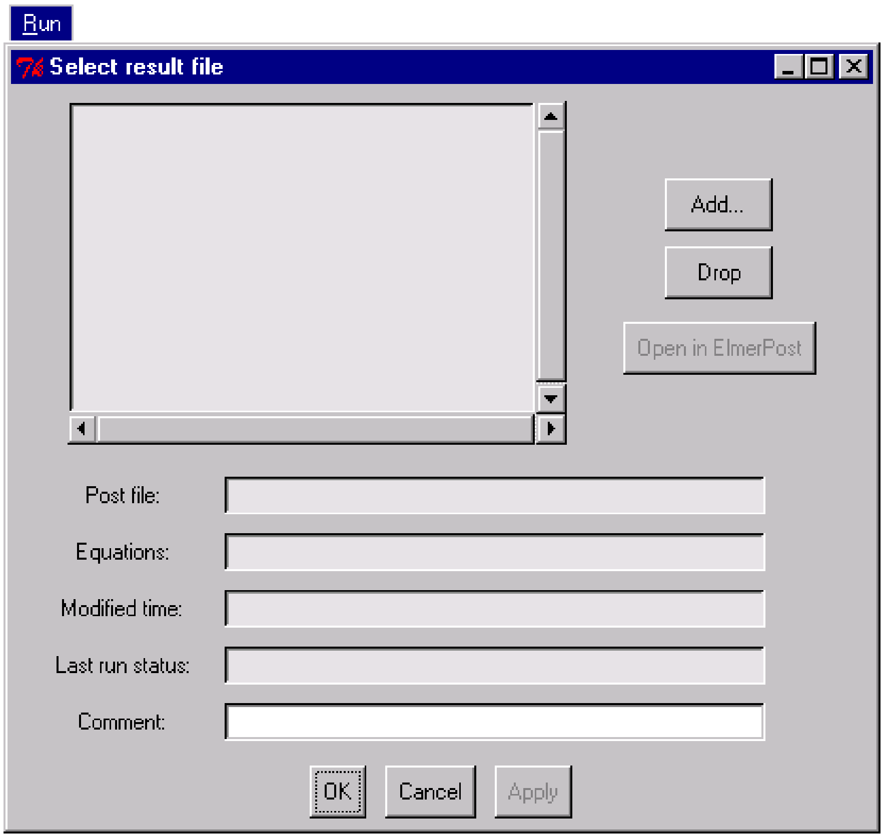
2.13.4 Solver:

This command starts ELMER Solver. It creates a browsable logfile.

2.13.5 Postprocessor:

Postprocessor command starts ELMER Post in a separate window.

2.13.6 Select result file



This window is needed when multiple meshes are in use. Because all result files are stored in the same directory where the mesh for an equation is stored, it is necessary to specify the result for postprocessing in a multimesh case.

All postprocessor output files produced during the session are listed in the listbox. Any of the files may be selected as the input for the postprocessor.

- Add opens a file browser for adding new files (created in other sessions) to the list

- Drop removes a file from this list. The file itself is not affected
- Open in ELMER Post opens the selected file in ELMER Post

The following fields are non-editable information fields:

- Post file: the full path for the postprocessor file
- Equations: the equations solved in the result file
- Modified time: the last update time for the result file
- Last run status: status of the process which updated last the output file
- Comment: this comment field is only for this session and it is not saved in the model file

2.13.7 Process table

When other ELEMR modules are started in ELMER Front, information on these processes is collected into a process table. This table can be opened using the Process table command or the Process button in the main window. This table displays information on the current status of a process and the user can also control the process using the command buttons in the process table window.

Process table columns:

- Process: the name of the process
- Nbr: an internal id number for the process
- PID: the operation system process id (if available)
- Prior: the process priority level
- State: the current process state. Value are:

  ALL DONE = process ended succesfully

  FAILED = process failed during the processing

  KILLED = process was ended by the user

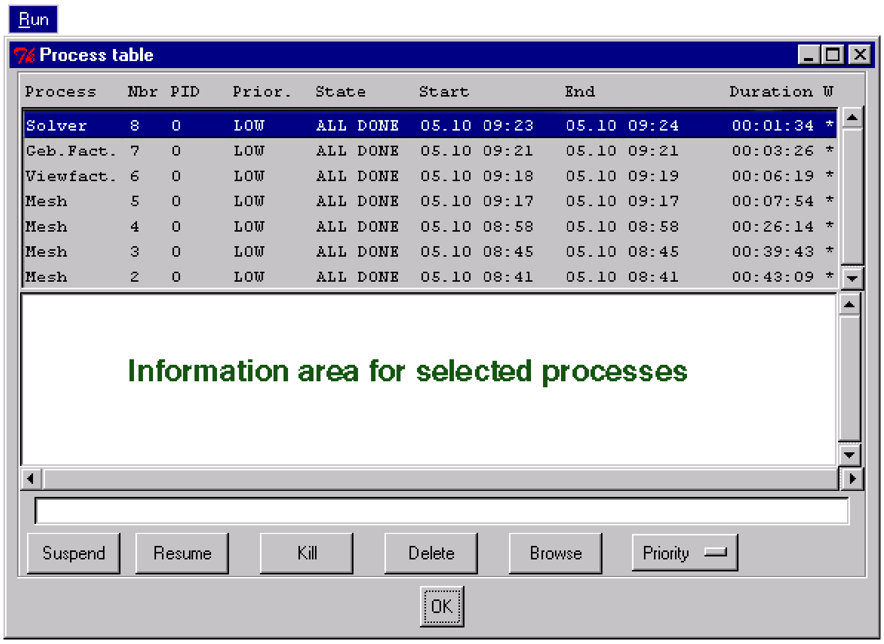
  SUSPENDED = process is suspended and is waiting to be activated

  RUNNING = an active and currently running process

- Start / End: process start and end times (wall clock times). For a running solver process the End field displays the number of currently processed timesteps

- Duration: total run time

- W: the rightmost (W) column is marked with an asterisk if the process created any error messages or warnings.



Process	Nbr	PID	Prior.	State	Start	End	Duration	W
Solver	8	0	LOW	ALL DONE	05.10 09:23	05.10 09:24	00:01:34	*
Geb.Fact.	7	0	LOW	ALL DONE	05.10 09:21	05.10 09:21	00:03:26	*
Viewfact.	6	0	LOW	ALL DONE	05.10 09:18	05.10 09:19	00:06:19	*
Mesh	5	0	LOW	ALL DONE	05.10 09:17	05.10 09:17	00:07:54	*
Mesh	4	0	LOW	ALL DONE	05.10 08:58	05.10 08:58	00:26:14	*
Mesh	3	0	LOW	ALL DONE	05.10 08:45	05.10 08:45	00:39:43	*
Mesh	2	0	LOW	ALL DONE	05.10 08:41	05.10 08:41	00:43:09	*

Figure 2.33: *Process table*

Processes are listed in the order they were started. The most recent process is on the top of the list. All fields are info fields. Below the process lists is an area where information on the currently selected process in displayed. The actual contents depends on the process type. Possible error and warning messages are also displayed in this area.

Process table buttons:

- Suspend: the selected process is suspended (paused)

- Resume: resumes (continues) the process

- Kill: stops the process (after confirmation). Note: the process name is not erased from the list

- Delete: kills the process and removes the process from the list (after confirmation)

- Browse: opens the log file (if available) for the selected process

- Priority: the process priority level can be set using this button. Options are Favor GUI or Favor process. These are descriptive values and the actual priority level transmitted for the operation system is platform specific. Favor GUI is the default and recommended value. Giving more priority for running processes can make the user interface almost unresponsive.

2.14 Window menu

Commands in the Window menu can be used to control currently active ELMER Front windows:

- Close all: closes all windows (except the main window and the Model window)

- Close unmodified: closes all windows where data has not been modified (except the main window and the Model window)

- Window list: displays a list of currently open windows. This list can be opened as a separate window by clicking on the dotted line as shown in Figure 2.34. Clicking a window name in this list activates the window.

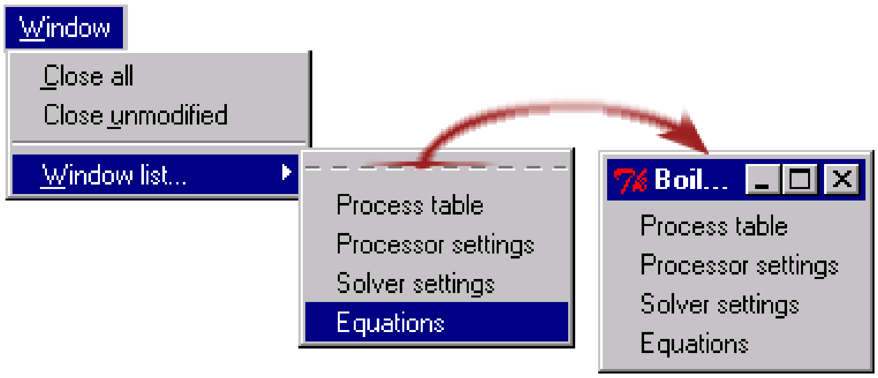
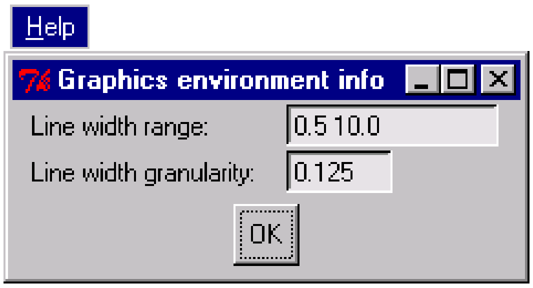


Figure 2.34: *Window list example*

2.15 Help menu

Figure 2.35: *Help menu*

2.15.1 Graphics info



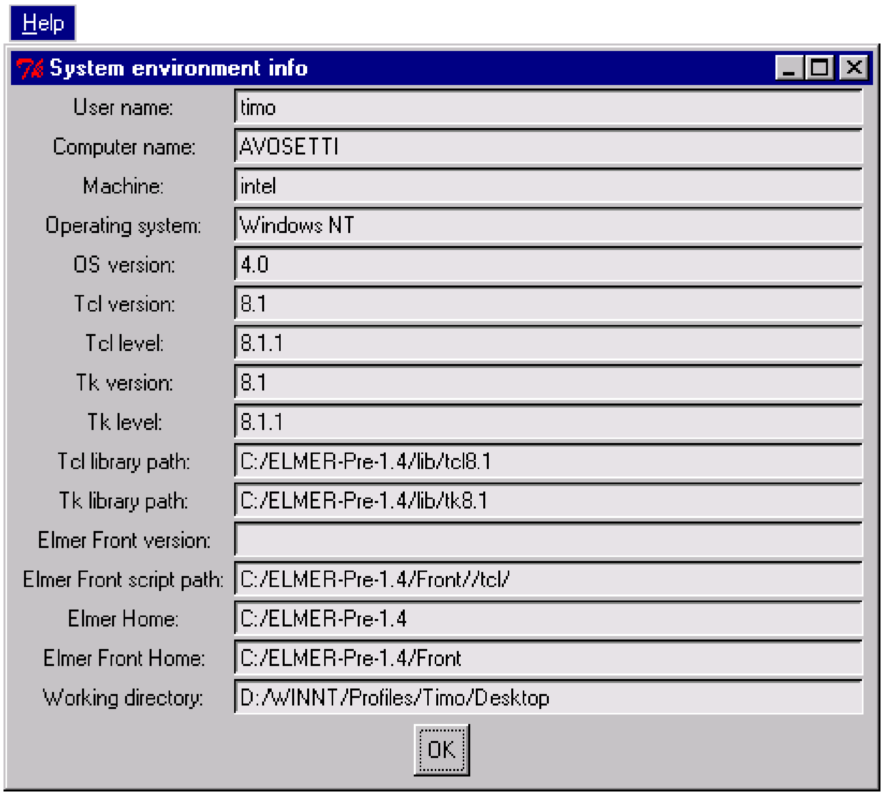
Line width range: 0.5 10.0

Line width granularity: 0.125

Figure 2.36: *Graphics info window*

The Graphics info window displays some OpenGL related graphics hardware information. These are non-editable information fields.

2.15.2 System info



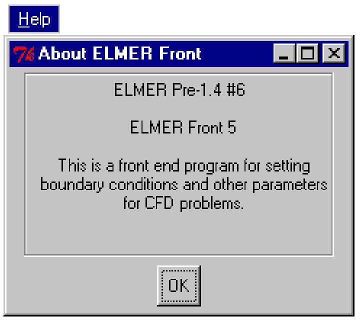
User name: timo
Computer name: AVOSETTI
Machine: intel
Operating system: Windows NT
OS version: 4.0
Tcl version: 8.1
Tcl level: 8.1.1
Tk version: 8.1
Tk level: 8.1.1
Tcl library path: C:/ELMER-Pre-1.4/lib/tcl8.1
Tk library path: C:/ELMER-Pre-1.4/lib/tk8.1
Elmer Front version:
Elmer Front script path: C:/ELMER-Pre-1.4/Front//tcl/
Elmer Home: C:/ELMER-Pre-1.4
Elmer Front Home: C:/ELMER-Pre-1.4/Front
Working directory: D:/WINNT/Profiles/Timo/Desktop

Figure 2.37: *System info window*

The System info window displays some information on operation system and on the external libraries which are needed for ELMER. These are non-editable information fields.

2.15.3 About

The About window displays information on the current ELMER version.



ELMER Pre-1.4 #6

ELMER Front 5

This is a front end program for setting boundary conditions and other parameters for CFD problems.

2.16 Model window toolbar

Figure 2.38: *Model window toolbar*

The following toolbar buttons in the ELMER Front main window are used to control how the model geometry is displayed in the Model window:

- Arrows on the left side are used for moving, rotating and zooming the model
- Rotate X, Y, Z checkboxes: model rotation axis is locked to the selected axis.
- Draw bodies: a CAD geometry (2D only) is drawn in solid format. For a mesh geometry this drawing mode displays the volume mesh by drawing the element edges
- Draw surfaces: mesh geometry is drawn using filled boundary elements (only for 3D models)
- Draw edges: CAD geometry is drawn using edges, mesh geometry is drawn using boundary element edges
- CAD geometry: activates/deactivates the display of the CAD geometry (if CAD geometry is available)
- Mesh geometry: activates/deactivates the display of the mesh geometry (if mesh is geometry available)

- Auto load mesh: when selected a mesh is automatically loaded whenever a model file is loaded. If the mesh is very large it may be useful to deactivate the autoload behaviour

- Select bodies: the bodies to be displayed are can be selected here

- Display: redisplays the Model window on the desktop if it has been closed or minimized

- Reset: resets the the graphics displyed in the Model window to its initial state

2.17 Model Window

The Model window is opened automatically when a geometry input file is read into ELMER Front. This window is used to display the model geometry and to offer a flexible interface for the model. When model objects are selected in the Model window, these selections are reflected in the object listboxes in data entry panels. On the other hand, selections made in these panels are highlighted in the Model window.

In the following example a boundary has been selected in the Model Window by double-clicking on it. As are result, the correponding boundary, the parent body and the attached boundary condition set rows are displayed as selected in the Boundary conditions window.

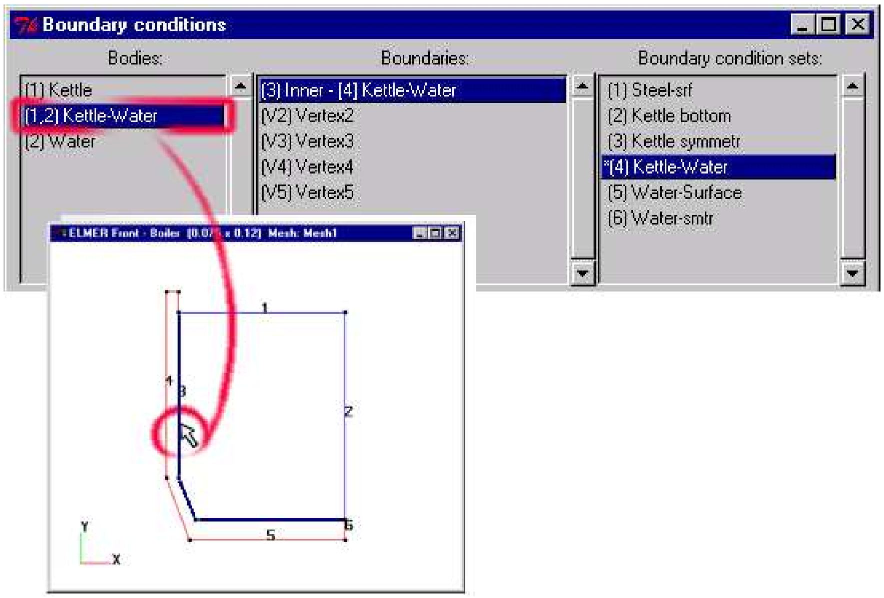


Figure 2.39: *Model window example*

The model can be manipulated in the window either by dragging it with the mouse or using the arrow buttons in the main window. Dragging with the left mouse button moves the model in the window. Dragging with the right mouse button rotates the model. The model can be scaled (zoomed) with the middle mouse button (or pressing left and right buttons simultaneously when using a 2-button mouse).

Bodies are displayed using different colors. These colors can be defined using the the Bodies command in the Edit menu.

All body elements (inner and outer boundaries, edges and vertices) are uniquely numbered. Same numbers are also used to identify these elements in the data entry windows. The Labels command in the Display menu can be used to switch theses labels on and off in the Model window.

2.18 Control toolbar

Most buttons in this toolbar have an equivalent command in menus.

- Save model: save model file

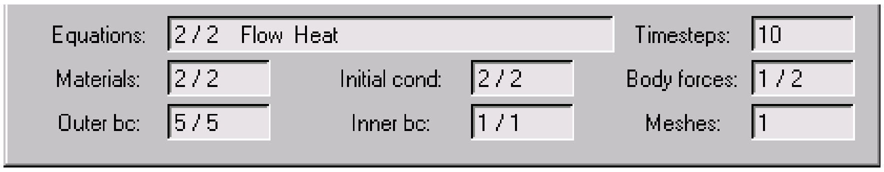
- Mesh: opens the Define Mesh window
- Load/Unload mesh: a two state button for loading and unloading the mesh
- Solve: starts the solver
- Results: starts the postprocessor
- Break: interrupts current ELMER Front processing
- Clear: deletes all current messages in the message area
- Process: opens the process table
- Info: opens the model info window

2.19 Message area

The window area in the middle of the main window is used to display information of various session events. This is a cumulative list of warnings, error messages and other session related incidents. This list can be cleared using the Clear button in the main window.

2.20 Status area

Fields at the bottom of the main window display information on the current status of the model. Red colored fields are meant to warn the user that the model may be incompletely defined concerning the parameters displayed in the status field. However, it is up to the user to decide if this really is the situation.



Equations: 2 / 2 Flow Heat
Timesteps: 10
Materials: 2 / 2
Initial cond: 2 / 2
Body forces: 1 / 2
Outer bc: 5 / 5
Inner bc: 1 / 1
Meshes: 1

Figure 2.40: *Status area*

- Equations: nof bodies having an equation / total nof bodies in the model and names for the defined equations
- Timesteps: total nof timestesp defined for the problem. In a steady state case the maximum nof iterations
- Materials: nof bodies having a material definition / total nof bodies
- Initial cond: nof bodies having an initial condition / total nof bodies
- Body forces: nof bodies having a body force definition / total nof bodies
- Outer bc: nof outer boundaries having a boundary condition / total nof outer boundries
- Inner bc: nof inner boundaries having a boundary condition / total nof inner boundries
- Meshes: nof currently active meshes

Chapter 3

ELMER Solver

In finite element (FE) context, the term *solver* refers to the part of the software package that forms the discretized equations corresponding to the mathematical model of the physical problem, and solves the resulting system(s) of equations. ELMER Solver is a general FE solver applicable for many types of partial differential equations and physical models.

This chapter gives an overview on ELMER Solver capabilities and usage, and explains the main structures and utilities available for the user.

A detailed description of all Solver options and how they are selected from ELMER Front is found in Chapter 2. A complete walk-through example of defining and solving a problem and visualizing the results is described in Chapter **??**. The mathematical equations and numerical methods are explained in Appendix A, and the `.sif` file (Solver Input File) format and keywords in Appendix F.

3.1 Overview

ELMER Solver is a separate program which may be started either from ELMER Front, or by running the executable `Solver` from the directory where the model's `.sif` file and the file `ELMERSOLVER_STARTINFO` are located.

Solver reads the user-defined model data, including initial and boundary conditions, material parameters, equation definitions, numerical methods and the mesh. Solver then forms the system of equations which it solves (several times, if the problem is non-linear, coupled or time-dependent). Solution is obtained for a vector of values representing the field variables such as velocity, displacement and temperature.

ELMER Front is used to define the model data for Solver. The mesh and `.sif` files required by Solver may also be created by other means, as discussed in Chapter 2.

Solver can currently handle, among others, the following problem types: incompressible and compressible fluid flow, heat transfer and radiation problems, structural analysis problems and the induction equation for magnetic field. These separate equation solvers may be coupled. It is also flexible to add one's own equation solvers to extend ELMER Solver's functionality.

Solver is written mainly in Fortran 90 and works in Unix (SGI, DEC and Linux) and Windows NT environments. The source code is available to the user, so one may add new features to Solver. In many cases, these may also be added by writing dynamically linkable procedures without touching the original code.

3.2 Capabilities

Solver can be applied to steady state or time dependent problems. Time integration may be done by the general Euler's method (Newmark Beta) or the backward differentiation formulae (BDF, orders 1–5).

Solver is capable of handling the implemented equations in several different coordinate systems, e.g., Cartesian 2D and 3D coordinates, polar 2D and 3D coordinates and cylindrical coordinates. The cylindrical coordinate system may include cylindrical symmetry or the axial symmetry of a vector field (such as velocity). In cylindrical symmetry, all variables are independent of the azimuthal coordinate. In axial symmetry, the azimuthal vector component is, in addition, identically zero.

The most important problem types included in Solver are fluid dynamics and heat transfer, typically solved as a coupled problem, advection-diffusion problems, elastic stress analysis, possibly coupled to the fluid flow (fluid-structure interaction) or temperature (thermal stress analysis), and magnetohydrodynamics (coupled induction equation and Navier-Stokes equations).

In this section we will discuss the heat transfer, Navier-Stokes, advection-diffusion and structural analysis problems. These problem types are offered an interface from ELMER Front version 1.4.

Heat transfer is described by the energy equation which is a scalar diffusion-convection equation. In other words, heat is transferred in a moving fluid (liquid or gas) by conduction and convection. In a solid material heat is transferred by conduction.

If the heat equation is coupled to the fluid flow, the velocity field may be obtained from the Navier-Stokes solution. The heat equation can be used to describe both an incompressible and compressible fluid.

Heat transfer mechanisms may also include radiation. In ELMER Solver, the *radiative heat transfer* may be by the following mechanisms:

- In idealized radiation, a physical body loses or gains energy on a boundary depending on the temperature difference of the boundary and a given external temperature.
- In diffuse-gray radiation, heat exchange between radiation surface elements is modeled.

In diffuse-gray radiation, *view factors* need to be computed. The view factors define the angle in which radiation elements see each other, taking into account possible obstruction by other object between them. The view factors are determined by different algorithms in cylindrically symmetric and Cartesian geometries. In large 3D problems, the view factor calculation may require a lot of computer resources, although an efficient ray tracing algorithm is used. From view factors, one must compute the *Gebhardt factors* that take into account the different emissivities of the radiation surfaces.

Thus, in diffuse-gray radiation, one must take the following three steps to solve the problem:

- Calculate the view factors.
- Compute the Gebhardt factors.
- Execute the solver.

All these tasks may be done from ELMER Front or run as separate programs from the command line. View factors and Gebhardt Factors are saved in separate files, so they don't need to recalculated every time.

The Navier-Stokes equations (continuity and momentum equations) describe the fluid flow. These equations always form a strongly coupled system in ELMER Solver, i.e., they are solved simultaneously. Navier-Stokes equations can be solved either in the incompressible or compressible form.

In the latter case, the equation of state is the ideal gas law.

Linearization of the various terms (radiation in heat equation, convection in Navier-Stokes, and compressible continuity equation) can be done either by Newton or Picard type methods. It is often a good idea to compute the first few nonlinear iterations with the more stable Picard linearization and then change to Newton linearization which has a faster rate of convergence. Nonlinearities resulting from nonlinear materials are solved with Picard iteration.

Advection-diffusion equation is formally identical to the heat equation. It is used to describe the advection and diffusion of a chemical subspecies in a flow field. It may naturally also be used to describe pure diffusion of an impurity or a dopant in a solid.

In many convection dominated cases, the Navier-Stokes, heat and advection-diffusion equations must be solved by the stabilized FE formulation, available in ELMER.

In **structural analysis**, the elastic stress equations are implemented in Solver. They are solved for the displacement field under an external load and/or thermal stress. The elastic stress equations are available for small and large displacement formulation. ELMER Front version 1.4 offers an interface to the small displacement formulation only.

The stress tensor may be calculated from the displacement field as an postprocessing step in ELMER Post.

Available element types in Solver are the Lagrangian element family: line, triangle, quadrilateral, tetrahedral and brick elements with linear, quadratic and cubic basis functions. All the elements are isoparametric, i.e., the same basis functions are used for the field variables and coordinates.

The linear systems of equations may be solved by direct or iterative methods. The direct solvers, based on LU decomposition use either LAPACK routines for band matrices with bandwidth optimization by the Cuthill-McKee method, or sparse matrix routines from SPARSE.

The iterative solver library HUTITER includes a few iterative methods (CG, BiCGStab, TFQMR and GMRES) with diagonal and ILU(n) preconditioners. A good choice of an iterative method is in many cases the bi-conjugate gradient stabilized (BiCGStab) method with incomplete LU factorization, ILU(0).

3.3 Structure of a Simulation

A simulation run has a set of common properties for all equations and bodies. These include the coordinate system, control parameters for time integration and steady-state iteration, physical constants, and result and mesh file information. It is also possible to use different meshes for solving different equations in a same body.

In addition to the definitions mentioned above, the model input data for Solver consists of bodies, boundaries, equations, body forces, boundary conditions and materials.

Model geometry is made of simulation *bodies*. A body is a geometrical entity which has boundaries, governing equations, initial conditions, body forces and material properties mapped to it. A physical body may consist of several simulation bodies.

Equations for a given body may be any set of the equations implemented in Solver. The various control parameters for each specific *equation solver* are described in Section 3.3.1.

Body forces define force or source terms for different equations. Typical body force terms are a heat source, gravitational force and Lorentz force.

Boundary conditions are organized as a set for all equations on a single boundary or a set of boundaries. If a boundary condition is not specified, this enforces the natural FEM boundary condition for that degree of freedom. Note that the natural boundary condition always corresponds to a physical boundary condition such as zero heat flux on the boundary.

Initial conditions give starting values for the field variables. They must be given in a transient problem, and can also be given for a steady-state analysis. The initial conditions may also be read from a given time step of a restart file.

Material properties for the body contain material parameter values needed to solve the equations.

All input data for the model that depends on space, time or solution—material properties, boundary conditions and body forces—may be given as constants, tabulated and interpolated tables or user-defined functions. This is discussed in Appendix F.

All Solver input data may be set from ELMER Front or in the `.sif` file.

3.3.1 Equation Solvers

In ELMER, equation solver refers to a solver for specific equation (such as the heat equation) or for a set of equations that is always solved strongly coupled (such as the Navier-Stokes equations). Equation solvers define the control parameters for linearization methods, solution of the system of linear equations, iteration strategy and convergence criteria. These equation solvers may be coupled to yield the governing equations in each body.

We have already discussed in Section 3.2 the following equations solvers included in ELMER Solver:

- Navier-Stokes equation
- heat equation
- advection-diffusion equation
- elastic stress equation.

The following coupling mechanisms between these equation solvers, available in ELMER, are often useful:

- Density variation due to temperature dependency in the Navier-Stokes equations, causing gravitational force and natural (Grashof) convection. For incompressible flow, the Boussinesq approximation is available.
- Surface tension coefficient variation due to temperature dependency in the Navier-Stokes equations, causing thermocapillary force (Marangoni convection).
- Temperature dependency of other material parameters. The variable viscosity method is available in the solidification phase change model.

- The convection term in the heat and advection-diffusion equations uses the solution for the velocity field.
- Viscous dissipation due to velocity field may produce heat.
- Thermal stress analysis uses the solution for the temperature field.

Solver goes through all the equation solvers for the governing equations sequentially. Each solves its system of (linearized) equations, possibly a few times, if an inner iteration for the nonlinear system is selected. After all equation solvers have been called, the outer iteration for the nonlinear problem within one time step or the steady-state problem is repeated.

The convergence criteria for different equation solvers and the maximum inner and outer iteration counts are set by the user. The convergence criteria must be satisfied by all equation solvers.

In time-dependent problems, the procedure is repeated for the specified number of time steps.

3.4 Structure of the Base FEM Code

The base FEM code is a general purpose partial differential equation (PDE) solver. Solver provides the element level discretizations of the linearized PDEs. The mesh describing the computational domain is divided into bulk elements and boundary elements. Contributions to the global matrix are added from each element in turn (local matrix generation).

All the material parameters can be provided to these routines at element nodes, enabling them to depend on coordinates or solution. For the anisotropic materials, the parameters may also be given as tensors instead of scalar values.

Solver provides, among others, the following FEM utilities to the programmer:

- Gauss integration points for different element types and desired accuracy
- Value of a quantity at the nodes of a given element

- Basis functions and their first and second derivatives with respect to global coordinate system
- Metric tensor of the element coordinate system, and the square root of the determinant of the Jacobian of the element's coordinate mapping
- Metric tensor of the global coordinate system and the square root of its determinant
- Christoffel symbols of the global coordinate system and their spatial derivatives
- Length, area or volume of a line, surface or volume element, respectively
- Normal and tangential vectors at a given point on the surface or boundary of an element
- Routines to determine in which element a given point in space belongs to
- Line, surface and volume integration routines.

These utilities may be used, e.g., to calculate the first and second derivatives with respect to global coordinate system or element coordinate system, and integrate new contributions added to the global stiffness matrix.

The base FEM code is written in such a way that the actual element types used in a simulation are transparent to the programmer in all routines and modules. By reading in different meshes, different elements are used. Different element types may also be combined in a model.

What element type to use in a given simulation depends on the application, personal taste and the mesh generators available. The mesh generator in ELMER version 1.4 produces only triangular elements. Using the quadratic basis functions instead of the linear ones will increase both the matrix assembly time and the time spent in solving the linear systems. On the other hand, the computation results are more accurate, and by using less elements the same solution accuracy can be achieved. In fact, some problems cannot be solved by the linear elements without using much denser mesh than with quadratic elements.

3.4.1 Additional Utilities

Besides the FEM utilities and equation solvers already discussed, Solver includes some additional utilities such as:

- Bandwidth optimization, employed before the direct LAPACK solver in band matrix format. The bandwidth optimization also benefits the iterative solvers by reducing the potential fill-in in ILU(n) preconditioning.
- LU decomposition
- Utility routines to handle sparse matrices in Compressed Row Storage (CRS) format, and to perform the FEM utilities in CRS format
- Storage of material parameters, boundary and initial conditions, field variables and solver control variables in list structures and manipulation of these lists
- A ray tracer module for the view factor computations
- Utilities for manipulation of bilinear, biquadratic and bicubic Bezier surfaces. This module is also used in the view factor computations.
- File handling utility routines
- Sort and search functions
- Loading and calling of the dynamically linkable user routines.

The usage of these utilities is beoynd the scope of this manual. You should contact the authors at CSC for advice and material on more advanced programming.

Chapter 4

The Post processor

This Chapter contains a brief overview of ELMER Post capabilities. **A separate ELMER Post manual** gives more specific information about this software. See also Appendix A.

4.1 General Overview

The visual post processing part of ELMER is called ELMER Post. ELMER Post is a program for displaying and analyzing results from time dependent finite element method simulations. ELMER Post can handle both 2D- and 3D- element models. ELMER Post is only loosely connected to the rest of the ELMER package, and might be used in totally different contexts.

The basic program building blocks are:

- TCL/TK for user interface
- OpenGL graphics library
- C language

A software package called Mesa, a free implementation of OpenGL, is used if platform dependent OpenGL implementation is not available.

These are all tools that are portable to various platforms including all Unix versions that we have tried, Windows NT, and potentially also Macintosh computers.

In the following a brief description of the software is given.

ELMER Post is started either from ELMER Front application or from the command prompt using `elmerpost` command.

4.2 Performance:

The program provides several options for displaying vector and scalar field variables:

- contour line display
- color coded meshes
- contour surfaces
- vector display
- sphere display
- particle and stream line display
- selecting arbitrary cuts from the model

In addition to being able to display the data there is more to ELMER Post. Several additional features include

- Basic matrix and vector operations
- Dynamically sizeable variables
- A programming language with functions, flow control, loops, I/O, etc.
- Derivative operators for the finite element model: gradient, divergence, and curl
- Graphical and a command line user interface
- The TCL/TK environment can be used to full extent, providing programmable user interface to extend the program to specific needs
- Colormap editor, surface material editor, background color editor
- Unlimited number of user controllable model views
- A versatile clipping ability
- Element grouping
- On-line help in HTML format (a simple basic HTML browser is also included)

- A matrix language MATC, used to manage data. It could be used to compute the stress tensor from a displacement field. Field variables from the simulation are automatically seen as MATC variables, after the FEM model and simulation data are read in to ELMER Post

Two programming languages, TCL and MATC, within a single application adds versatility to the program. The scope of the two languages are also quite different, TCL is a command/script language, and MATC a matrix programming language. Thus they complement each other.

ELMER Post has also a built-in ability to execute user provided extension programs. These extension programs may be used to draw to the ELMER Post graphics window using OpenGL graphics calls. Also MATC variables could be directly manipulated etc.

4.3 Element Types

The element types ELMER Post are the same type as in the solver. These have been described in Appendix **??**.

4.4 Input Format

The input file format of ELMER Post has been described in Appendix **??**. In addition to this format, ELMER Post is able to read the FIDAP neutral file format. The model mesh must be given to ELMER Post using either of these formats.

Data might be read in separately. Read in could utilize the MATC input routines:

- via the load command
- a set of routines which resemble the C language std I/O routines

Chapter 5

Walk-through example

In this chapter the usage of ELMER software is demonstrated with an example. All basic steps needed when using ELMER are covered: preparing the input geometry, defining the problem in ELMER Front, generating the mesh using ELMER Mesh2D, solving the problem using ELMER Solver and visualizing the results with ELMER Post.

It is assumed that the user is familiar with contents of the ELMER User's Guide and uses it as a reference during this exercise.

It is also suppoused that ELMER Front is opened before the walk-through of this example is started. ELMER Front can be started by entering the command 'ELMER' in the command line. For details, see ELMER User's Guide Chapter 2.

After ELMER Front has been opened, it is advised that a first-time user selects the 'Novice' user level by using the User level command in the Edit menu. This user level setting protects a new user from loosing updates by giving warning messages if modified panels are closed without saving them.

5.1 Boiler: heating water in a kettle

In this example water is heated in a steel kettle. The kettle is heated at the bottom and heat conduction and the resulting flow in the water is modelled.

In modelling terms this a coupled problem. Velocities calculated in the Navier-Stokes equation are needed in the convection term of the heat equation. On the other hand temperatures calculated in the heat equation are needed in the force term of the Navier-Stokes equation.

The geometry of the model consists of two bodies, a steel kettle and the water in this kettle. Because of rotational symmetry this problem can be modelled using an axi-symmetric 2D geometry.

5.1.1 Input geometry

The geometry for this example is defined in the following file delivered with ELMER software:

`.../dist/examples/Boiler.egf`.

This file can be opened in ELMER Front by using the Open Cad file command in the File menu. The user should use the file browser to locate the above path and then open the `Boiler.egf` file.

When the geometry file is loaded into ELEMR Front, the Model graphics window is opened automatically. It should look similar as in Figure 5.1.

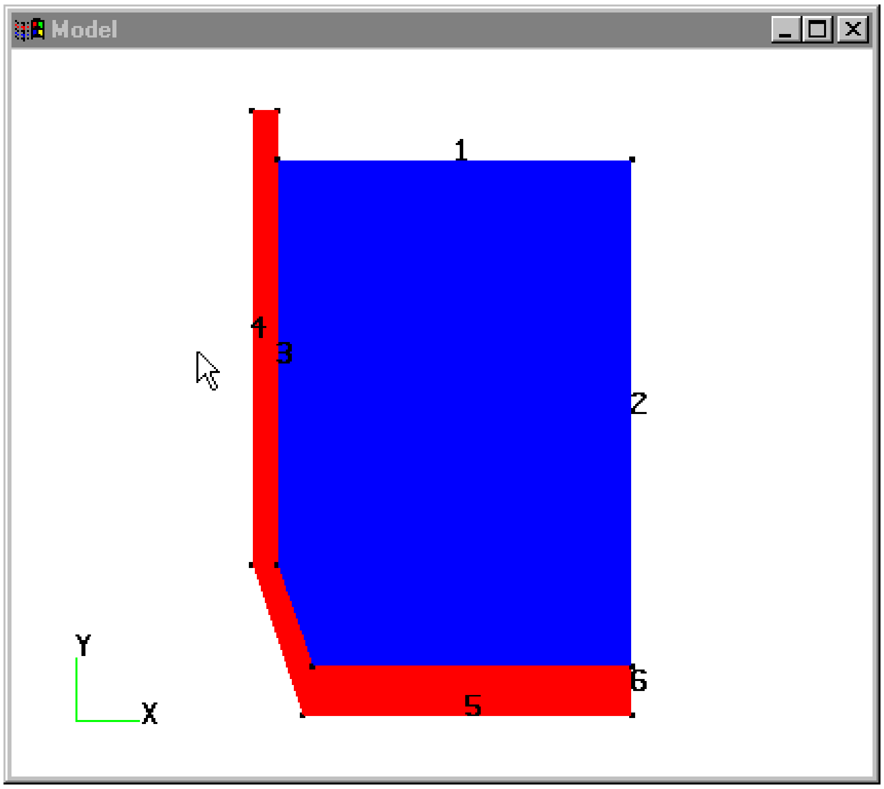


Figure 5.1: *Boiler: model geometry*

Note: because this an axi-symmetric problem, only the left half of the 2D crosssection is defined in the file.

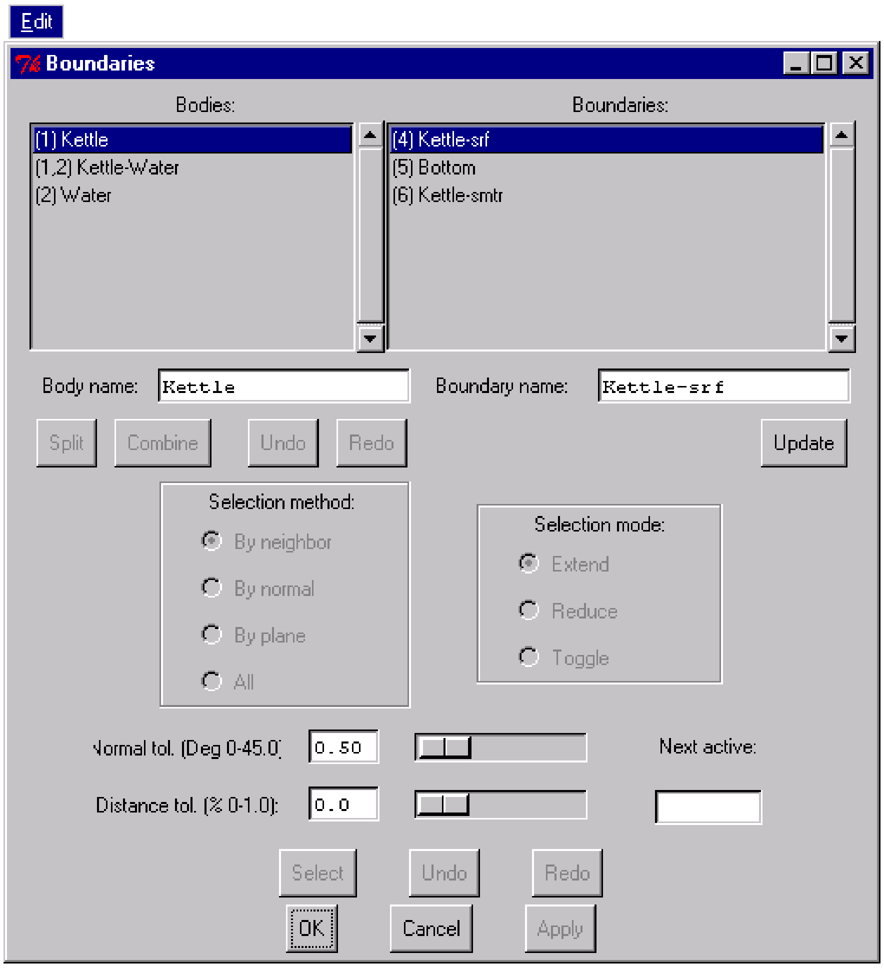
In Figure 5.1 bodies are displayed using the colors defined in the input file. If the Draw edges button in the main window is pressed, bodies are drawn unfilled using only the boundary edges. The Draw bodies button can be used to return to the original mode.

5.1.2 Defining names and directories

Usually the very first step in ELMER Front model definition is to give names for bodies and boundaries. These names are displyed in data entry panels and it is easier to identify different bodies and boundaries if they have meaningul names.

However, this step is not needed in this example because the names were defined already in the input file. The user can check the names using the Bodies and the Boundaries commands in the Edit menu.

The latter command opens the Boundaries window which is shown in Figure 5.2.



Bodies:
[1] Kettle
[1,2] Kettle-Water
[2] Water

Boundaries:
[4] Kettle-srf
[5] Bottom
[6] Kettle-smtr

Body name: Kettle

Boundary name: Kettle-srf

Selection method:
● By neighbor
○ By normal
○ By plane
○ All

Selection mode:
● Extend
○ Reduce
○ Toggle

Normal tol. (Deg 0-45.0): 0.50

Distance tol. (% 0-1.0): 0.0

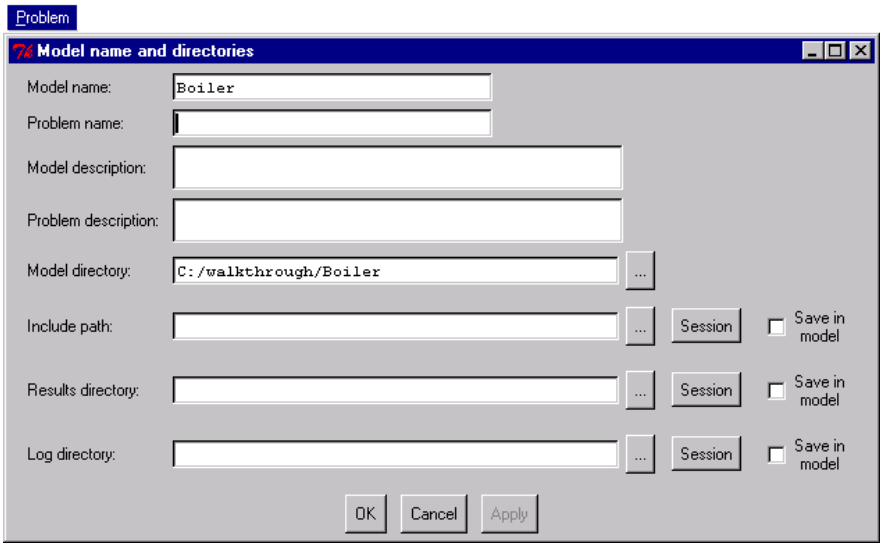
Next active:

Figure 5.2: *Boiler: boundary names*

The only editable field in this window is the Boundary name entry where the user can edit the boundary name. All other fields are inactive. They are used when boundaries are split and combined, but these operations are not allowed for a CAD geometry as explained in Chapter 2.

It is a good idea to save now the first version of the model file. For this, the model name and directory must be defined. This data is entered in the Model name and directories window which is opened using the corresponding command in the Problem menu. Again the model name was defined already in the input file, but model directory is not yet defined.

If the directory does not yet exists, the user must enter the directory name in the Model directory entry field. An existing directory can be selected using the directory browser which can be opened pressing the browser button on the right side of the entry field. Figure 5.3 shows the situation after the user selected the `C:`-drive in the directory browser, then opened the `walkthrough` subdirectory by double-clicking it, then selected the `Boiler` directory by single-clicking it in the list of directories and finally pressed the Ok button in the browser window.



Model name: Boiler

Model directory: C:/walkthrough/Boiler

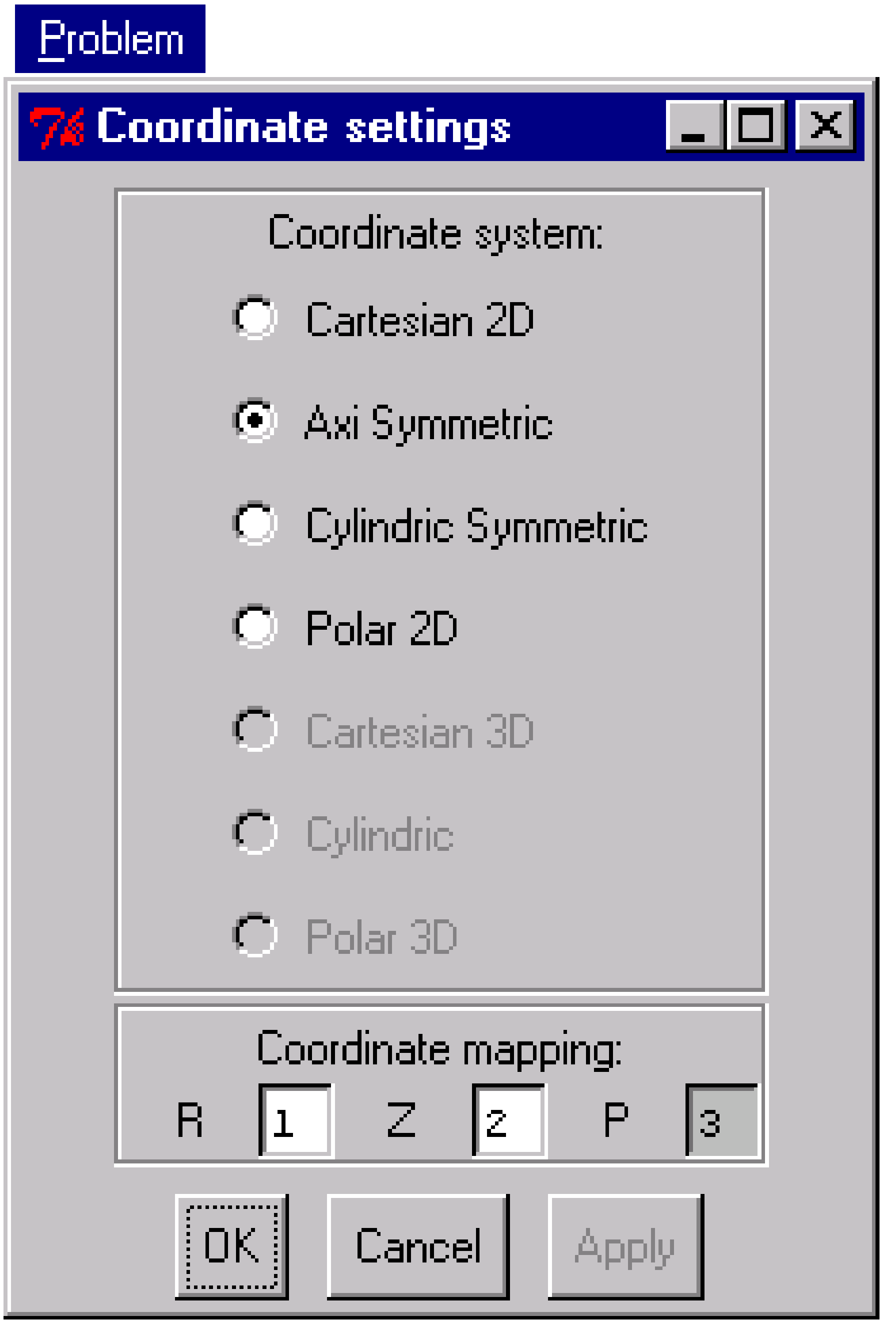
Figure 5.3: *Boiler: model name and directories*

Pressing the Ok or the Apply button updates the model. The model file can be now saved by using either the Save model button in the main window or the Save model command in the File menu. This model would be saved in the following file: `C:/walkthrough/Boiler/Boiler.emf`.

5.1.3 Problem definition

This section covers the settings for the model coordinate system, timestepping scheme and the equations to be solved.

Model coordinate system can be set using the Coorinate settings command in the Problem menu. The user should select the ‘Axi Symmetric’ setting as shown in Figure 5.4. The default coordinate mapping is suitable for this model.



Coordinate system:

○ Cartesian 2D
● Axi Symmetric
○ Cylindric Symmetric
○ Polar 2D
○ Cartesian 3D
○ Cylindric
○ Polar 3D

Coordinate mapping:

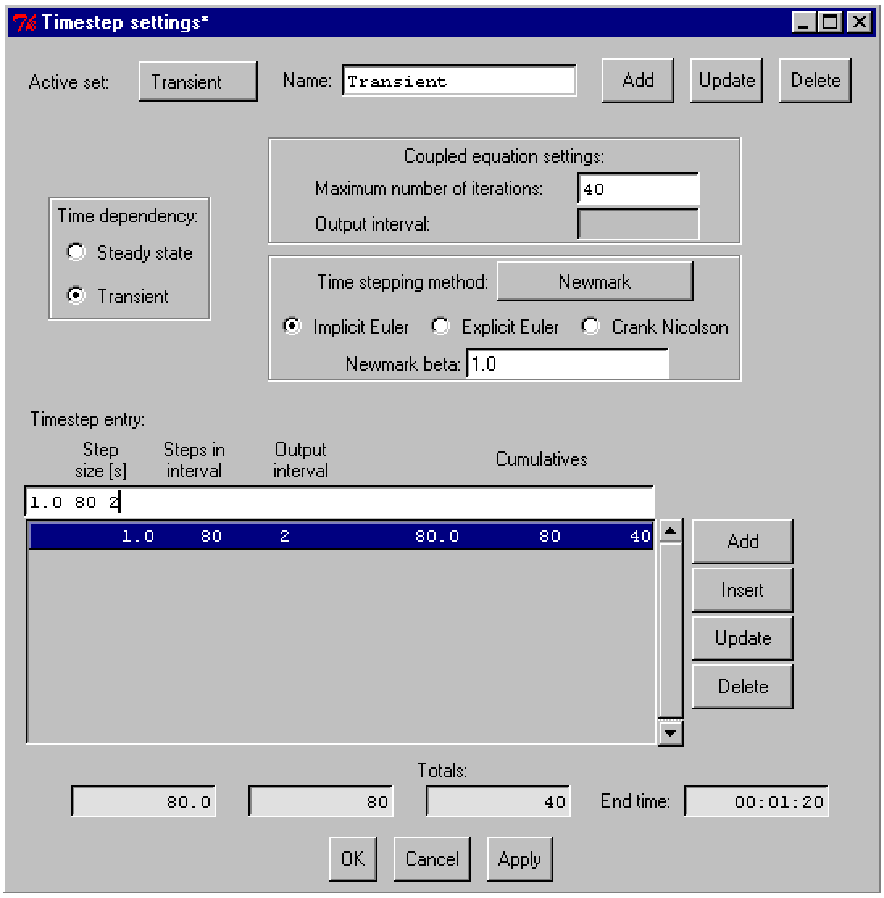
R 1 Z 2 P 3

Figure 5.4: *Boiler: coordinate system*

A timestepping scheme for the model is defined in the Timestep settings window. To open this window, the user should select Timestep settings in the Problem menu.

A scheme for a time-dependent simulation is defined. This timestep set is named by default as 'Timestep1', but in the example a new name is entered in the Name field. When the Transient radio button is selected, the user can enter timestep data in the Timestep entry field. Three numbers are needed: the timestep size in seconds, the number of timesteps in the interval and the ouput interval. When the Add button is pressed (at the righ side of the listbox) a new interval is added to the timestep set which is being defined.

In Figure 5.5 one interval has been added. The timestep size is one second, the number of timesteps is 80 and every second timestep will be ouput. So, the simulation is ended at 80 seconds. It is of course possible use a longer simulation time, but it could be a lengthy process and owing to the inherently chaotic nature of the problem, numerical problems may become an obstacle for the simulation when the flow in water is fully developed.



Active set: Transient

Name: Transient

Coupled equation settings:

Maximum number of iterations: 40

Output interval:

Time dependency: ○ Steady state ● Transient

Time stepping method: Newmark

● Implicit Euler ○ Explicit Euler ○ Crank Nicolson

Newmark beta: 1.0

Timestep entry: 1.0 80 2

Step size [s]	Steps in interval	Output interval	Cumulatives		
1.0	80	2	80.0	80	40

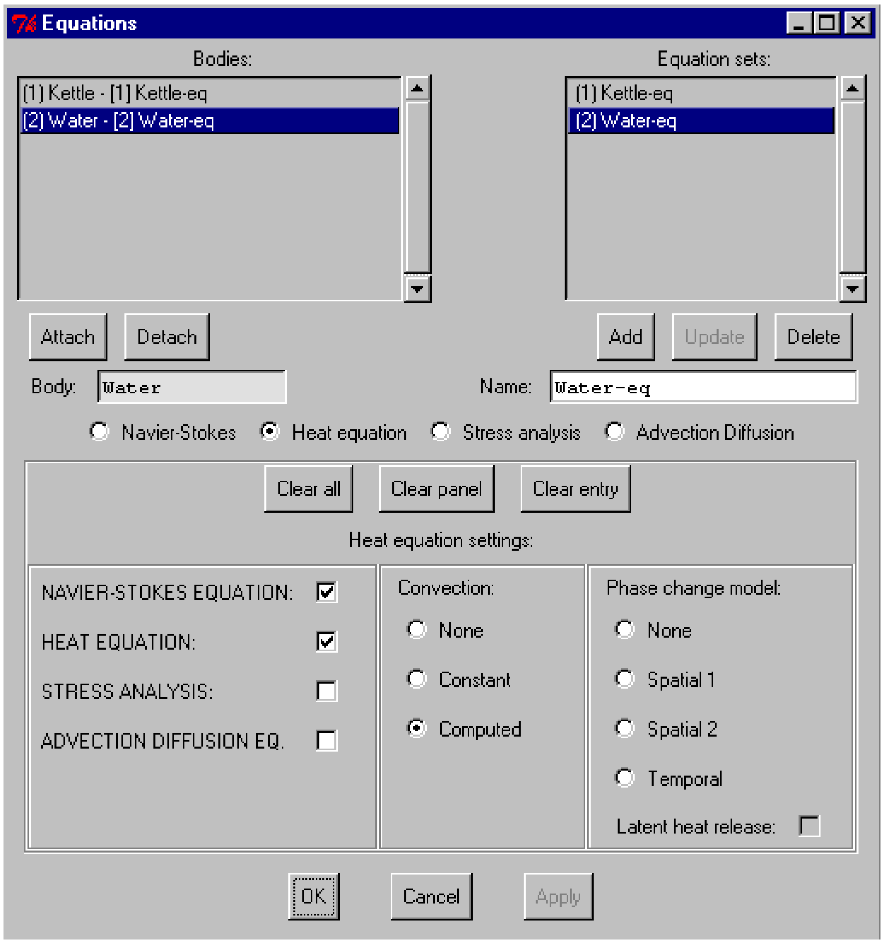
Totals: 80.0 80 40 End time: 00:01:20

Figure 5.5: *Boiler: timestep settings*

Equations for the problem are defined next. The Equations window is opened using the Equations command in the Problem menu. This window is shown in Figure 5.6.

In this problem each body needs a separate equation definition. In the 'Kettle' body only heat equation is calculated. For the 'Water' body the user has to define an equation set where both heat and flow equations are included. In the example equation sets are named according to the body names.

The equation 'Kettle-eq' is a simple heat conduction equation and it is attached to the 'Kettel' body. When defining the 'Water-eq' it is best first to check the HEAT EQUATION and NAVIER-STOKES EQUATION checkboxes in the lower left side of the window. Activating the flow equation allows entering flow related parameters in the heat equation panel. This panel can be activated by pressing the Heat equation radio button in the middle area of the window. It is now possible to select computed convection for the heat equation. The 'Water-eq' can now be added to the list of equation sets by pressing the Add button under the listbox. It should be attached to the 'Water' body as shown in Figure 5.6.



Bodies:
(1) Kettle - [1] Kettle-eq
(2) Water - [2] Water-eq

Equation sets:
(1) Kettle-eq
(2) Water-eq

Body: `Water`
Name: `Water-eq`

○ Navier-Stokes ● Heat equation ○ Stress analysis ○ Advection Diffusion

NAVIER-STOKES EQUATION: ☑
HEAT EQUATION: ☑
STRESS ANALYSIS: ☐
ADVECTION DIFFUSION EQ. ☐

Convection:
○ None
○ Constant
● Computed

Phase change model:
○ None
○ Spatial 1
○ Spatial 2
○ Temporal
Latent heat release: ☐

Figure 5.6: *Boiler: equations*

All necessary steps in the Problem menu are now done and other model parameters can be entered.

5.1.4 Model parameters

The commands in the Model menu are used to enter initial conditions, body forces, material properties and boundary conditions.

Because this a time-dependent case, both initial and boundary conditions are needed for a well defined problem.

Flow in the 'Water' body is created by natural convection. In ELMER this is modelled by a body force where the so called Boussinesq approximation is used to simulate a buoyant force.

The initial conditions are the following:

$$\begin{aligned} T &= 293 \quad \text{on} \quad \text{Kettle}, \\ \vec{u} = 0,\ T &= 293 \quad \text{on} \quad \text{Water}, \end{aligned}$$

The boundary conditions are as follows:

$$\begin{aligned} \vec{u}_Z = 0,\ \ -k\frac{\partial T}{\partial n} &= 40.0 \cdot (T - 293.0) \quad \text{on} \ \ \text{Water-srf (b1)}, \\ \vec{u}_R = 0,\ \ -k\frac{\partial T}{\partial n} &= 0 \quad \text{on} \ \ \text{Water-smtr (b2)}, \\ \vec{u} &= 0, \quad \text{on} \ \ \text{Kettle-Water (b3)}, \\ -k\frac{\partial T}{\partial n} &= 10.0 \cdot (T - 293.0) \quad \text{on} \ \ \text{Kettle-srf (b4)}, \\ T(0) = 293.0,\ \ T(40) &= 373.0 \quad \text{on} \ \ \text{Bottom (b5)}, \\ -k\frac{\partial T}{\partial n} &= 0 \quad \text{on} \ \ \text{Kettle-smtr (b6)}. \end{aligned}$$

At the water surface no vertical velocities are allowed and the heat transfer coefficient between water and the environment at room temperature is $40.0\frac{\mathrm{W}}{\mathrm{m^2K}}$. The corresponding coefficient for the kettle surface is $10.0\frac{\mathrm{W}}{\mathrm{m^2K}}$.

At the symmetry line no heat flux or horizontal velocities are allowed.

The boundary between the kettle and water is a no-slip, heat conduction boundary.

The boundary condition for the bottom of the kettle means that the temperature is increased from room temperature to 100 centigrades in 40 seconds and then kept at this level.

The material parameters for the (steel) kettle are the following:

$$\begin{aligned} \rho &= 8100.0 \quad \frac{\mathrm{kg}}{\mathrm{m}^3}, \\ c_p &= 500.0 \quad \frac{\mathrm{J}}{\mathrm{kgK}}, \end{aligned}$$

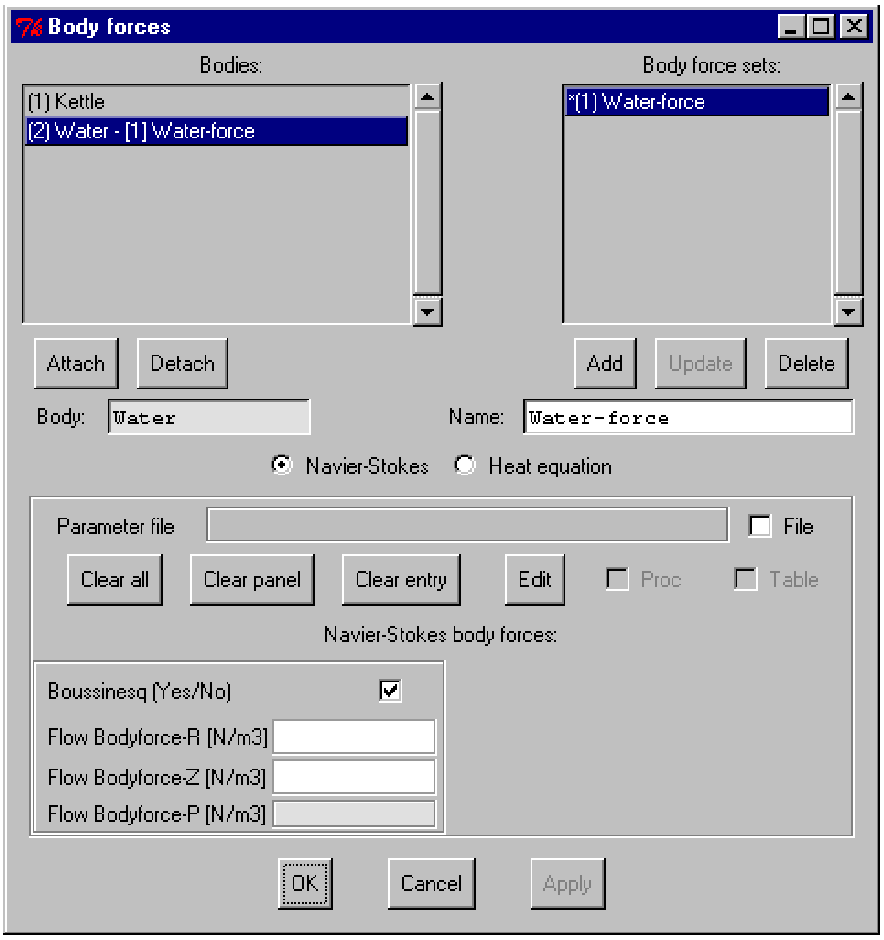
$$k = 16.0 \quad \frac{\mathrm{W}}{\mathrm{mK}},$$

The material parameters for water at room temperature are the following:

$$\begin{aligned}
\rho &= 998.2 && \frac{\mathrm{kg}}{\mathrm{m}^3},\\
\mu &= 9.93 \cdot 10^{-4} && \frac{\mathrm{kg}}{\mathrm{ms}},\\
c_p &= 4182.5 && \frac{\mathrm{J}}{\mathrm{kgK}},\\
k &= 0.597 && \frac{\mathrm{W}}{\mathrm{mK}},\\
\beta &= 2.1 \cdot 10^{-4} && \frac{1}{\mathrm{K}}.
\end{aligned}$$

A material property set and an initial condition set must be defined for each body. These definitions can be entered using the Materials and Initial conditions commands in the Model menu. When entering the data for the 'Water' body, the user must use the equation-type radio buttons in the middle of the window to display the equation-type dependent entry fields.

The body force definition needed for the 'Water' body is shown in Figure 5.7.



Body forces

Bodies:

[1] Kettle

[2] Water - [1] Water-force

Body force sets:

*[1] Water-force

Body: Water

Name: Water-force

● Navier-Stokes ○ Heat equation

Parameter file ☐ File

☐ Proc ☐ Table

Navier-Stokes body forces:

Boussinesq (Yes/No) ☑

Flow Bodyforce-R [N/m3]

Flow Bodyforce-Z [N/m3]

Flow Bodyforce-P [N/m3]

Figure 5.7: *Boiler: body force settings*

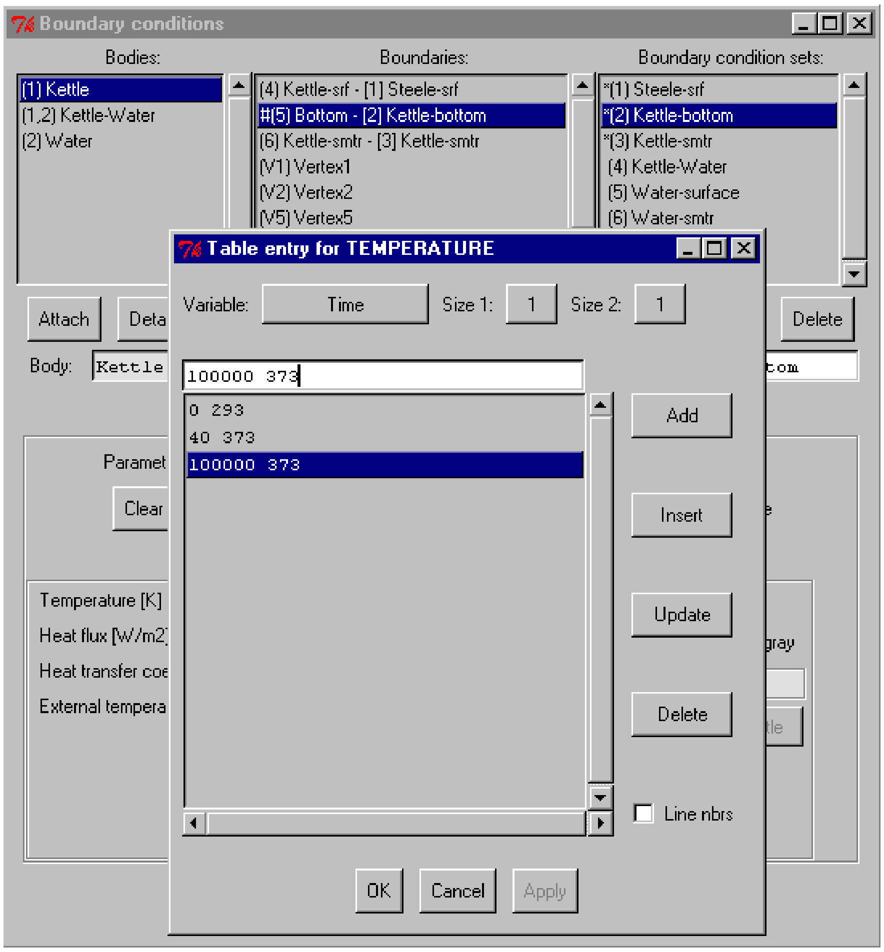
The workflow in the Boundary conditions window is basically similar as in the previous windows. The only difference is that boundary conditions are attached to boundaries instead of bodies. An outer boundary can be selected by first selecting the parent body in the bodies listbox and then selecting the target boundary in the boundaries list. An inner boundary, ie. a boundary between two bodies, can be selected by first selecting the body pair formed by the two parent bodies.

For example, the no-slip boundary condition for the boundary between the kettle and water can be defiend by first selecting the ‘Kettle-Water´ body pair and then attaching the proper condition for the ‘Ìnner’ boundary. The no-slip condition is defined by selecting the Navier-Stokes equation and entering zero values for both velocity components.

The temperature boundary condition for the bottom of the kettle can be entered as a temperature table where the simulation time is the argument variable. The user should first select the Temperature field in the heat equation panel and then check the Table checkbox. A table entry window is opened and the data shown in Figure 5.8 should be entered. First, the argument variable 'Time' is selected from the Variable button menu. Next the (time,temperature)-pairs are entered in the entry field and added to the table by pressing the Add button. The last, very large time-argument value guarantees that the constant end-temperature is kept during the rest of the simulation (otherwise the bottom temperature would be set to zero after 40 seconds).

This table entry can be reopened by double-clicking on the temperature field or by pressing the Edit button when the input cursor is on the Temperature field.

It should noted that the updated table data is not stored unless the Update button in the Boundary conditions window is pressed. This concerns in fact all fields, but it is far easier to re-enter a lost update for a normal scalar field than to re-enter a complicated table!



Bodies:
[1] Kettle
[1,2] Kettle-Water
[2] Water

Boundaries:
[4] Kettle-srf - [1] Steele-srf
#[5] Bottom - [2] Kettle-bottom
[6] Kettle-smtr - [3] Kettle-smtr
[V1] Vertex1
[V2] Vertex2
[V5] Vertex5

Boundary condition sets:
*[1] Steele-srf
*[2] Kettle-bottom
*[3] Kettle-smtr
[4] Kettle-Water
[5] Water-surface
[6] Water-smtr

Body: Kettle

Variable: Time
Size 1: 1
Size 2: 1

```
100000 373
```

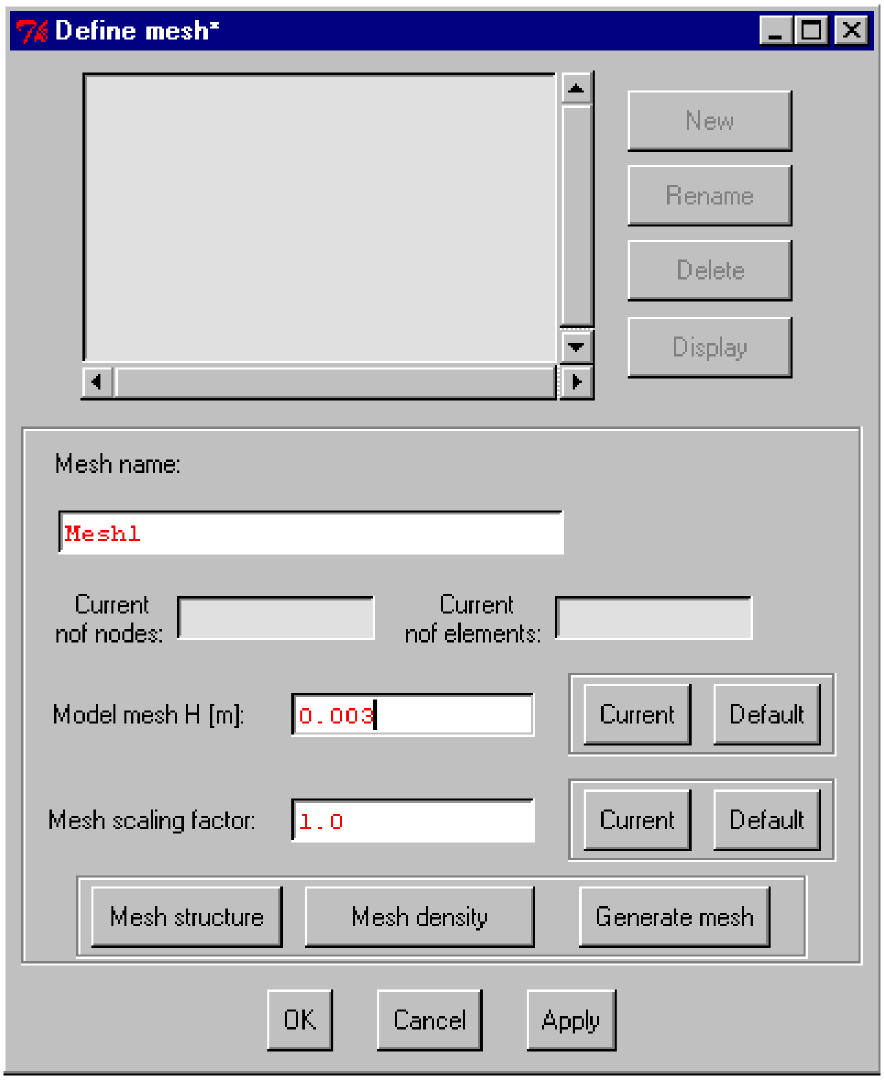
```
0 293
40 373
100000 373
```

Figure 5.8: *Boiler: temperature table*

Boundary conditions can be set also at vertices, but in this case there is no need to set conditions at vertex level.

5.1.5 Mesh generation

Mesh definition is started by the Define mesh command in the Mesh menu. This command opens the window shown in Figure 5.9.



Mesh name: Mesh1

Model mesh H [m]: 0.003

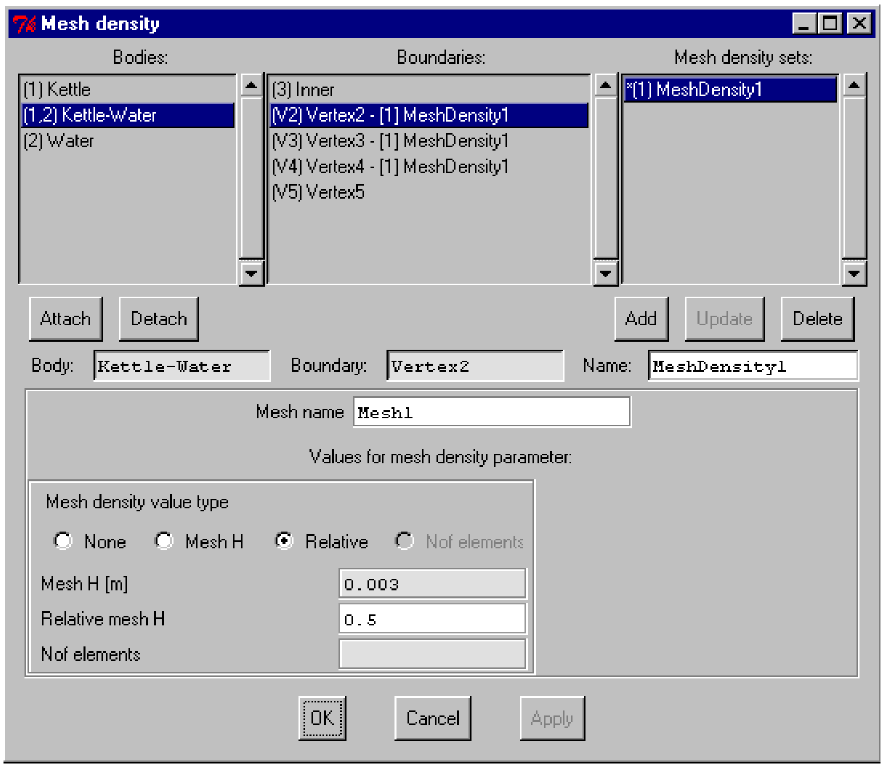
Mesh scaling factor: 1.0

Figure 5.9: *Boiler: mesh definition*

A new mesh called ‘Mesh1’ will be defined. Pressing the New button will activate the Mesh name entry where this nmae can be entered. The model level mesh parameter is set to the value 0.003 instead of the default value 0.0075.

Pressing the Mesh structure button opens a window where the mesh structure for each body can be defined. However, for this example the default mesh structure values are suitable (an unstructured linear triangular mesh) and there is no need to enter any body level mesh parameters.

However, the mesh should be denser at the kettle-water surface to improve the stability of the simulation. For this, the user should enter mesh density values for the three vertices `V2, V2 and V3` at this surface (vertex labels can be displayed in the Model graphics window by using the Labels command in the Display menu). This can be done in the Mesh density window which is opened by pressing the Mesh density button in the mesh definition window.



Bodies:
(1) Kettle
(1,2) Kettle-Water
(2) Water

Boundaries:
(3) Inner
(V2) Vertex2 - [1] MeshDensity1
(V3) Vertex3 - [1] MeshDensity1
(V4) Vertex4 - [1] MeshDensity1
(V5) Vertex5

Mesh density sets:
*(1) MeshDensity1

Body: Kettle-Water
Boundary: Vertex2
Name: MeshDensity1

Mesh name Mesh1

Values for mesh density parameter:

Mesh density value type
○ None ○ Mesh H ● Relative ○ Nof elements

Mesh H [m] 0.003
Relative mesh H 0.5
Nof elements

Figure 5.10: *Boiler: mesh density settings*

As shown in Figure 5.10, a mesh density parameter with the relative mesh H value of 0.5 (compared to the model level mesh density in this case) has been attached to the three vertices. Closing this window and using the Generate mesh button in the mesh definition window will create the mesh shown in Figure 5.11.

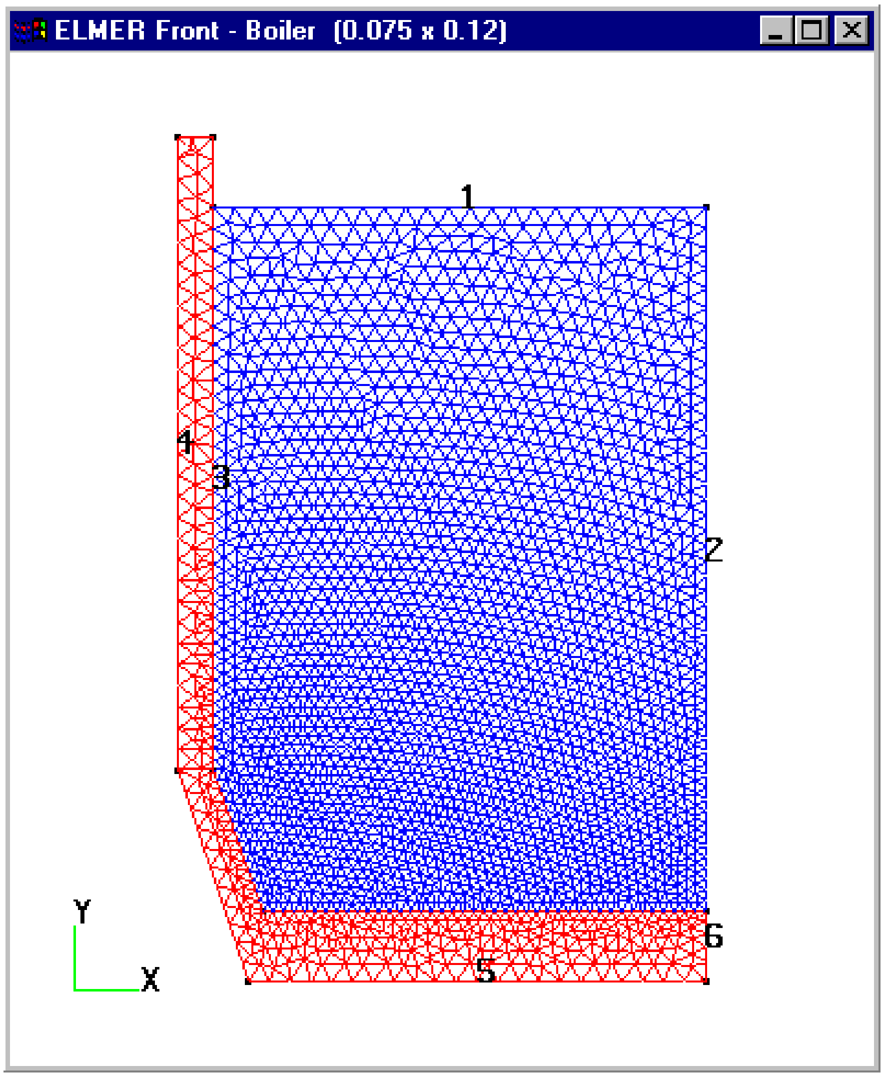
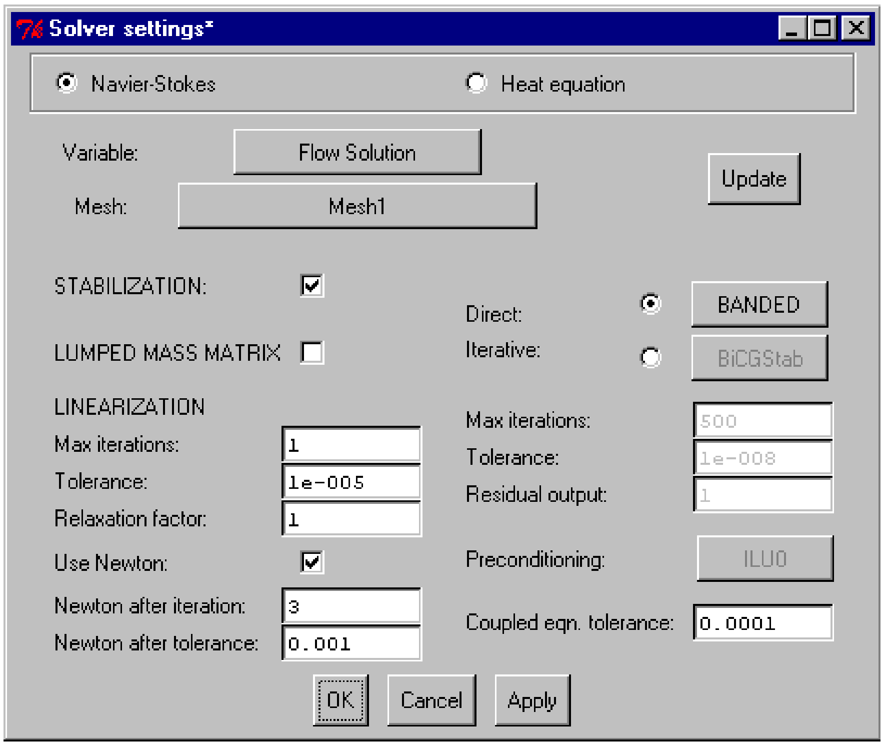


Figure 5.11: *Boiler: mesh*

5.1.6 Solving

The final step in the model preparation is to define solver parameters for each equation. Selecting Solver settings in the Solver menu opens a window where these parameters can be entered. At the top of this window is a set of radio buttons, one for each active equation in the model. Selecting a radio button opens solver entry fields for the corresponding equation.

The solver parameter settings for the Navier-Stokes equation are shown in Figure 5.12.



● Navier-Stokes ○ Heat equation

Variable: Flow Solution

Mesh: Mesh1

STABILIZATION: ☑

LUMPED MASS MATRIX ☐

Direct: ● BANDED

Iterative: ○ BiCGStab

LINEARIZATION

Max iterations: 1

Tolerance: 1e-005

Relaxation factor: 1

Use Newton: ☑

Newton after iteration: 3

Newton after tolerance: 0.001

Max iterations: 500

Tolerance: 1e-008

Residual output: 1

Preconditioning: ILU0

Coupled eqn. tolerance: 0.0001

Figure 5.12: *Boiler: solver settings*

Most of the fields can be left to their default values. However, a dircet linear system solver is used instead of the (default) iterative solver. Because the mesh is quite small, a direct solver is propably the best option for this problem. The user should press the radio button labelled 'Direct' and check that the text 'BANDED' is displayed on the button right to the radio button.

The Max iterations field in the 'LINEARIZATION' block should be set to one instead of the default value three. In practice this means that the solver does not actually enter into the linearization process. Instead of the normal 'internal' iterations, the solution for each timestep is searched by 'external' iterations which comes from the coupling between the Navier-Stokes equation and the heat equation. The maximum number for coupled equation iterations was already set to 40 in the Timesteps settings panel. Setting the Coupled eqn. tolerance solver parameter value to 0.0001 will quarantee that this 'external' iteration reaches a solution.

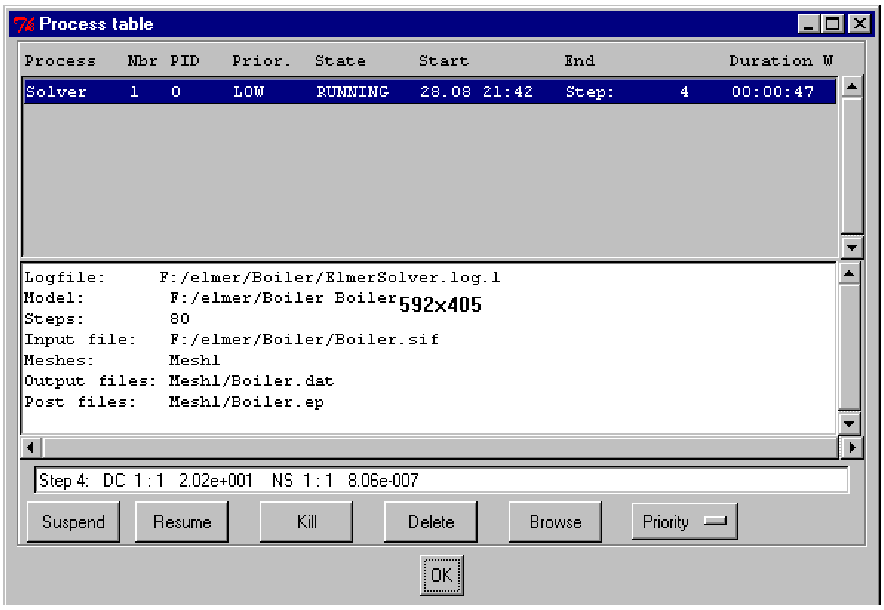
When the parameters have been entered, the user should press the

Update button to store the definitions for the Navier-Stokes solver. Similar values should be entered for the heat equation solver.

Pressing the Ok button will update the model data and the model can be solved. For this, the user should press the Solve command button in the main window or use the Solver command in the Run menu.

When ELMER Solver is started, its ouput is displayed in a browsable log window. The user can monitor the solver also in the Process table window which can be opened by using the Process command button in the main window or the Process table command in the Run menu.

This table displays information on processes started during the current ELEMR Front session. The user can also control the processes by using the command buttons at the bottom of the table. In Figure 5.13 the user has selected the currently active solver process and the information for this process is displayed.



```
Process   Nbr  PID   Prior.  State     Start          End            Duration W
Solver    1    0     LOW     RUNNING   28.08 21:42    Step:      4   00:00:47
```

```
Logfile:     F:/elmer/Boiler/ElmerSolver.log.1
Model:        F:/elmer/Boiler Boiler
Steps:        80
Input file:   F:/elmer/Boiler/Boiler.sif
Meshes:       Mesh1
Output files: Mesh1/Boiler.dat
Post files:   Mesh1/Boiler.ep
```

```
Step 4:  DC 1 : 1  2.02e+001   NS 1 : 1  8.06e-007
```

Figure 5.13: *Boiler: solver process table*

The information in the status row above the command buttons is updated whenever one of the equations is solved. The following status information is displayed: the number of the current timestep, a short-hand name for the equation, the number of the coupled iteration within the timestep, the number of the linearization iteration within the coupled iteration and the average norm value for the unknown variable being solved.

For example, the string:

```
Step 49: DC 2:1 3.03e+001  NS 1:1 4.00e-003
```

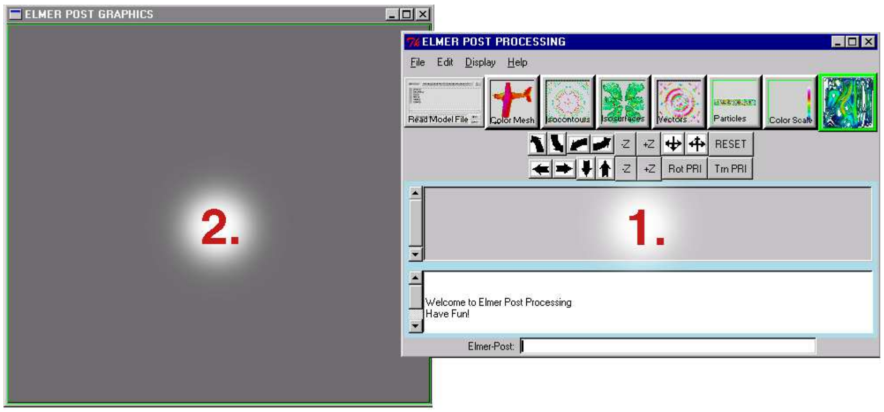
would tell that in the timestep 49 heat equation (Diffusion-Convection eq.) has been iterated once for the linearization during its second coupled iteration and the average temperature in the bodies where heat equation was solved was about 303 degrees of Kelvin. The average absolute flow velocity (from the previous coupled iteration) was about 4 mm per second.

When a solver process is succesfully at end, its state in the process table will change from 'RUNNING' to 'ALL DONE'. A similar message is displayed also at the end of the solver log. At this phase all timesteps are output to the postprocessor file.

5.2 Visualization with ELMER Post

ELEMR Post can be started by using the Results command button in the ELMER Front main window or by the Postprocessor command in the Run menu.

As ELMER Post is started, the ELMER Post main window is opened (1.) along with the Graphics Window (2.) as shown in Figure 5.14.

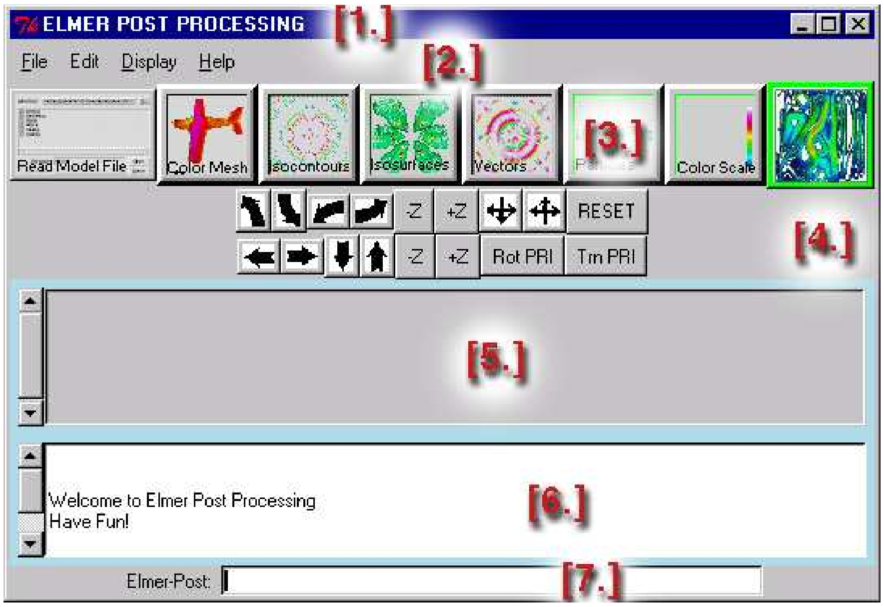


```
Welcome to Elmer Post Processing
Have Fun!
```

Figure 5.14: *ELMER Post User Interface*

5.2.1 ELMER Post User Interface

The main window is divided in seven horizontal sections, shown numbered in the next figure from [1.] to [7.]:



Welcome to Elmer Post Processing
Have Fun!

Figure 5.15: *ELMER Post main window areas*

Section definitions:

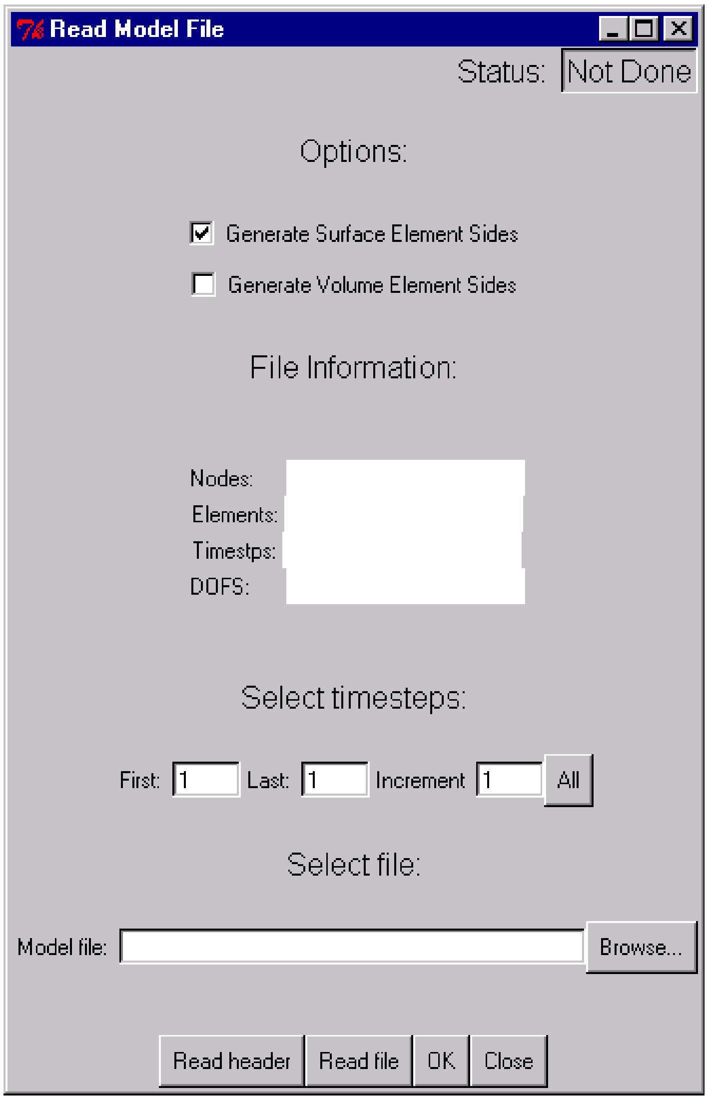
- [1.] Window header
- [2.] Menubar
- [3.] Menu command buttons
- [4.] Graphics window control buttons
- [5.] The log for Elmer-Post commands
- [6.] Message area
- [7.] An entry for Elmer-Post commands

This walk-through concentrates mainly on the standard ELEMR Post features which are available through the menus. For the special Elmer-Post commands and the programming capabilities in ELMER Post, the user should see the ELMER Post User Manual.

5.2.2 Reading the postprocessor file

First click on the File [2.] / Open -menu.

The Read Model File window opens. The next figure shows the Read Model File window opened in its initial state, before any user definitions has been made.



Read Model File

Status: Not Done

Options:

☑ Generate Surface Element Sides

☐ Generate Volume Element Sides

File Information:

Nodes:
Elements:
Timestps:
DOFS:

Select timesteps:

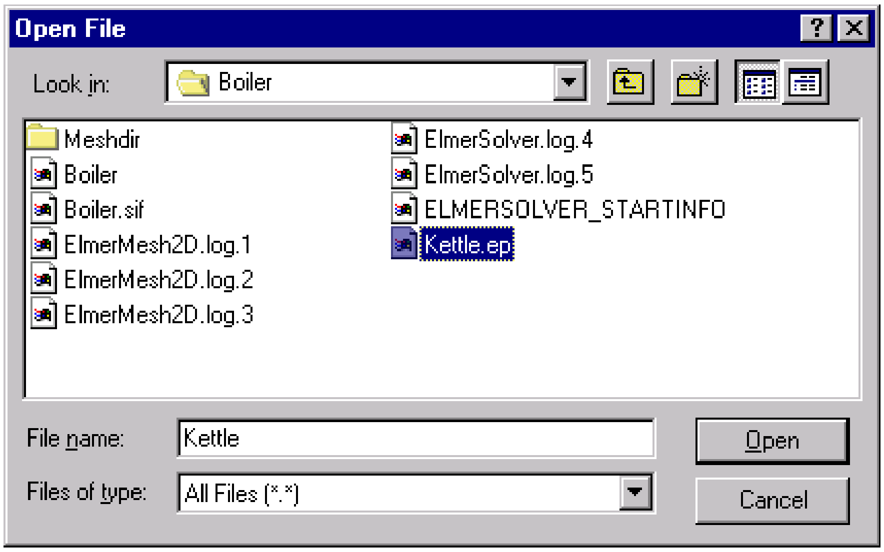
First: 1 Last: 1 Increment 1

Select file:

Model file:

Figure 5.16: *Read Model File -window*

Click Browse... button at low left part to open the file browser. As File browser opens, the desired file is located and selected by clicking on the Open button. The browser window closes.



Look in: Boiler

File name: Kettle

Files of type: All Files (*.*)

The Read Model File window has now the file selected. The filename is now seen defined in the Model File -field of the Read Model File window.

First thing to decide is how many timesteps are going to be displayed. Select All timesteps by clicking on the All button.

First: 1 Last: 160 Increment 1

Next Click on Read file button.

Model file: C:/walkthrough/Boiler/Kettle.ep

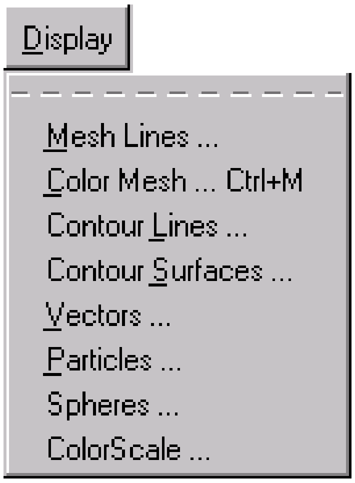
Status indicator at top right corner will now display a message: Reading.

Status: Read Done

ELMERPost starts to load the file in. When file reading is ready, Read Done message is shown. Read Model File window is closed with the Close button. The Model has now been successfully read into the ELMERPost, although at the moment there is no visible indication of this. This is because no default variable values exist for the Model.

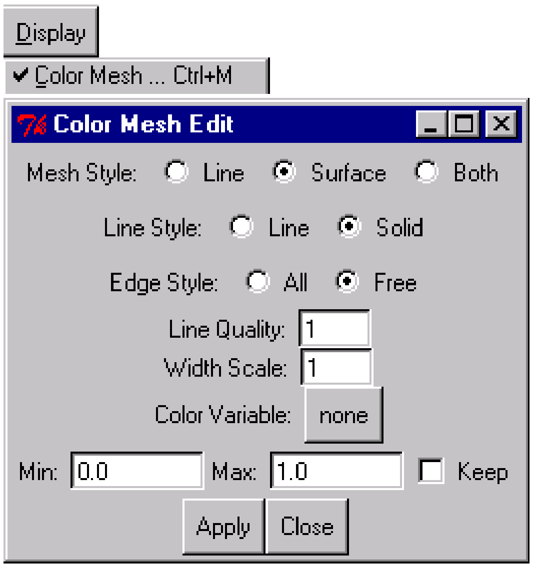
Note: These actions (like window closing after clicking on the Close button) might consume time, depending on the available computing resources and Model size.

5.2.3 Model Display properties



Next thing is to define the Model display properties. The adjacent window opens after clicking on the Display [2.] menu button.

Clicking on the Color Mesh menu item opens a Color Mesh Edit window.



Mesh Style: ○ Line ● Surface ○ Both

Line Style: ○ Line ● Solid

Edge Style: ○ All ● Free

Line Quality: 1

Width Scale: 1

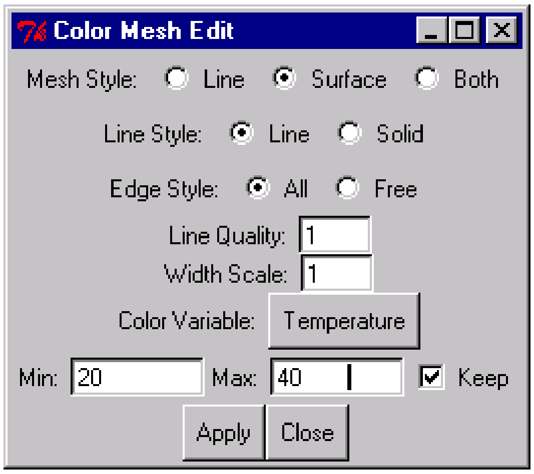
Color Variable: none

Min: 0.0 Max: 1.0 ☐ Keep

Figure 5.17: *Color Mesh Edit -window in default mode*

As the user has clicked on a menu item, this is indicated with a check mark. In the figure above this check mark is shown at top left.

As a default, no Color Variables have been defined for the model. Clicking on the Color Variable button, opens a variable list window called vlist. The desired variable to be viewed is selected here, and the window is then closed. In our example below we have selected the Temperature variable from the list. After closing the vlist window, this variable is shown selected on the button. As in this Model water temperature rises only slightly, we select the displayed temperatures between to be 20-40 degrees. The keep checkbox is checked, and Mesh style Surface radio-button selected. Refer to the next figure.



Color Mesh Edit

Mesh Style: ○ Line ● Surface ○ Both

Line Style: ● Line ○ Solid

Edge Style: ● All ○ Free

Line Quality: 1

Width Scale: 1

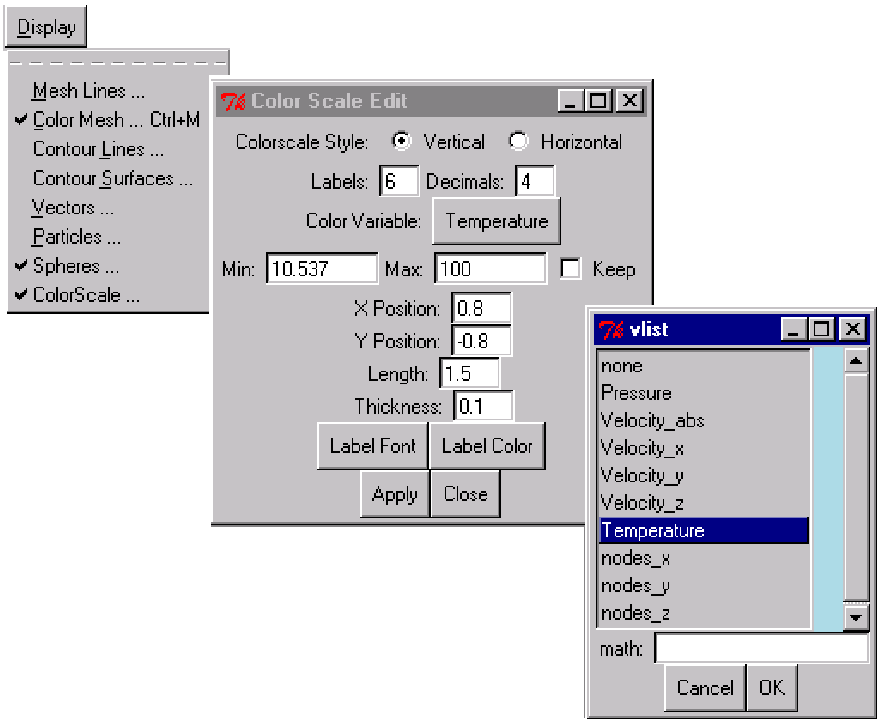
Color Variable: Temperature

Min: 20 Max: 40 ☑ Keep

Click on Apply and Close buttons. In the Graphics Display will be now viewable a solid blue model. This represents the Model in the first Timestep, when both model bodies, water and kettle are at 20 degrees celsius.

Keep checkbox is used to fix variable limits permanent for all used timesteps. If checkbox is not selected, the limits will be automatically updated by the currently selected timestep setting.

In the next figure is shown how this selection is done for ColorScale bar to be displayed in the graphics window.



Colorscale Style: ● Vertical ○ Horizontal

Labels: 6 Decimals: 4

Color Variable: Temperature

Min: 10.537 Max: 100 ☐ Keep

X Position: 0.8

Y Position: -0.8

Length: 1.5

Thickness: 0.1

math:

Figure 5.18: *ColorScale definition*

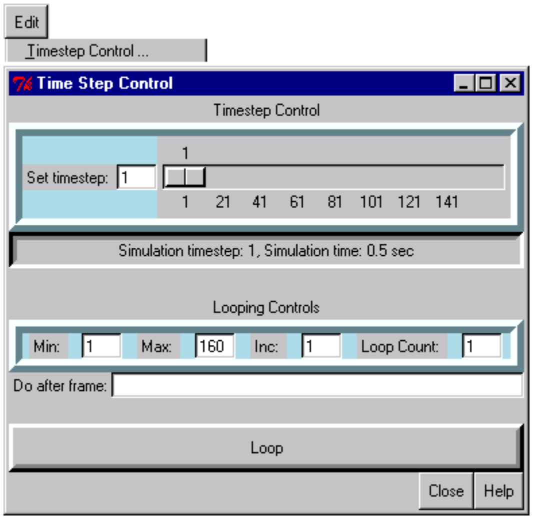
The vlist window (above, lower right corner) is a general variable window, used by multiple panels in variable selection. In case the vlist window is already open for other definitions, clicking on a variable selection button results in a conflict. An error message is then displayed.

5.2.4 Timestep control

Through the Edit [2.] menu is selected Timestep Control. This opens a Time Step Control window used for viewing those timesteps that have already been read in.

Timestep Control section in the next example figure (Time Step Control window) indicates that currently is displayed Timestep one. From Looping Controls section can be seen that maximum number of iterations possible for this model is 160.

By clicking on the Loop button, the Model timesteps start to advance.



Set timestep: 1

Simulation timestep: 1, Simulation time: 0.5 sec

Min: 1 Max: 160 Inc: 1 Loop Count: 1

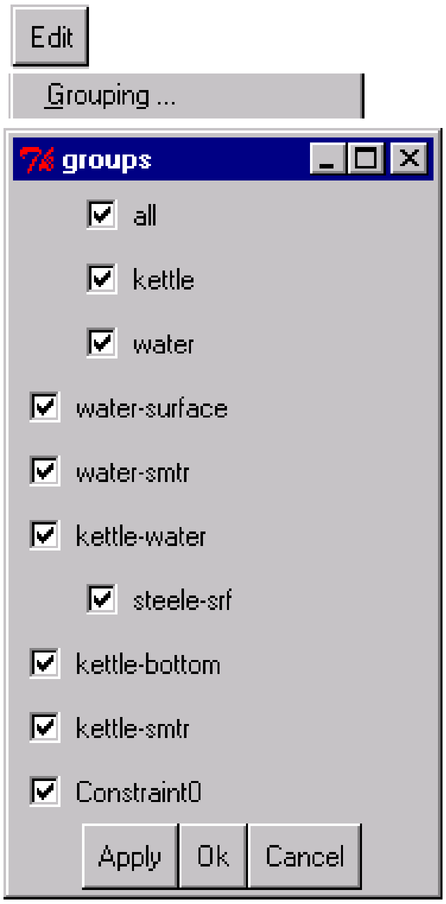
Do after frame:

Figure 5.19: *Time Step Control window*

In the Graphics window the temperature is seen gradually rise upwards in the kettle bottom area. The Timestep Control Set timestep field displays the current step and also the slider advances in parallel with the Graphics window. As the temperature continues to rise, around timestep 50 the water starts to warm up, and visible convection starts to form. Looping can be stopped by clicking on the same button (now the label on the button is: Stop).

Any timestep can be selected by typing the desired step number into the Set timestep field and then hitting enter.

The desired Model bodies can be selected through the Edit [2.] Grouping menu. Clicking on this, the following groups menu opens. In this Model top three checkboxes represent the available bodies that can be selected for viewing. Refer to the figure at left.



- ☑ all
- ☑ kettle
- ☑ water
- ☑ water-surface
- ☑ water-smtr
- ☑ kettle-water
- ☑ steele-srf
- ☑ kettle-bottom
- ☑ kettle-smtr
- ☑ Constraint0

Next figure displays the Graphics window showing the Model at timestep

80.

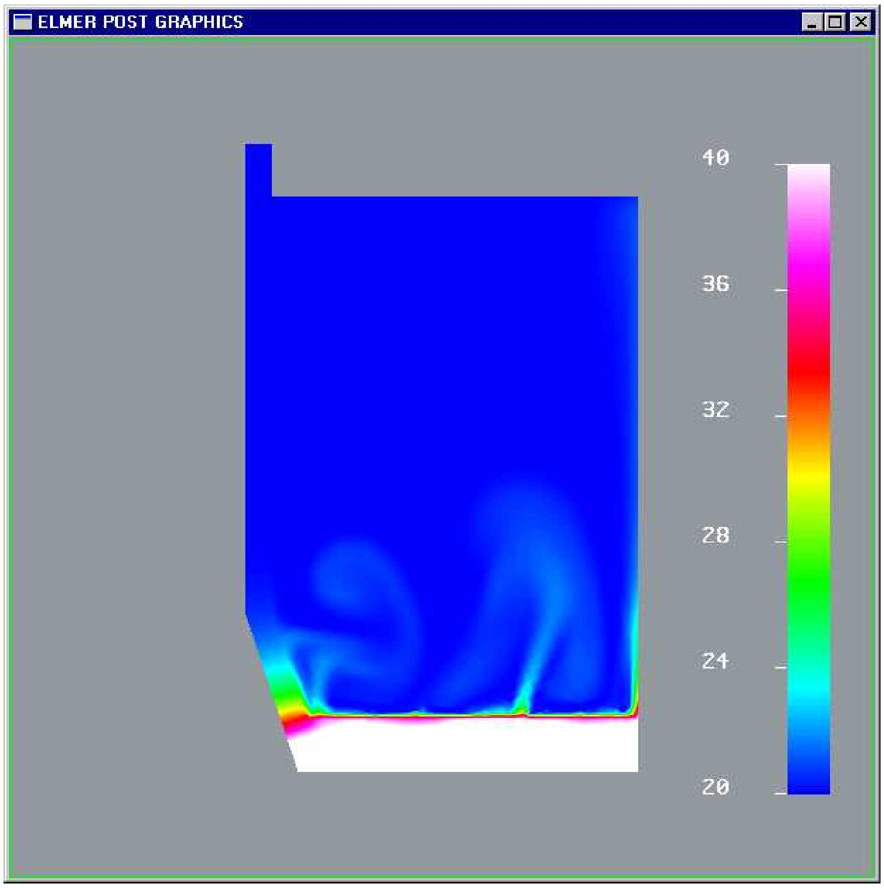


Figure 5.20: *Kettle Model at timestep 80*

In our example, Display [2.] ColorScale window has a value selected also to 20-40 degrees (keep checkbox selected). The Graphics window background has been also changed for illustration purposes.

The window for defining the Graphics background is opened through Edit [2.] Background menu refer to next figure at right.

For printouts it might be desirable to have a white background; this is achieved by moving the three slider at far right.

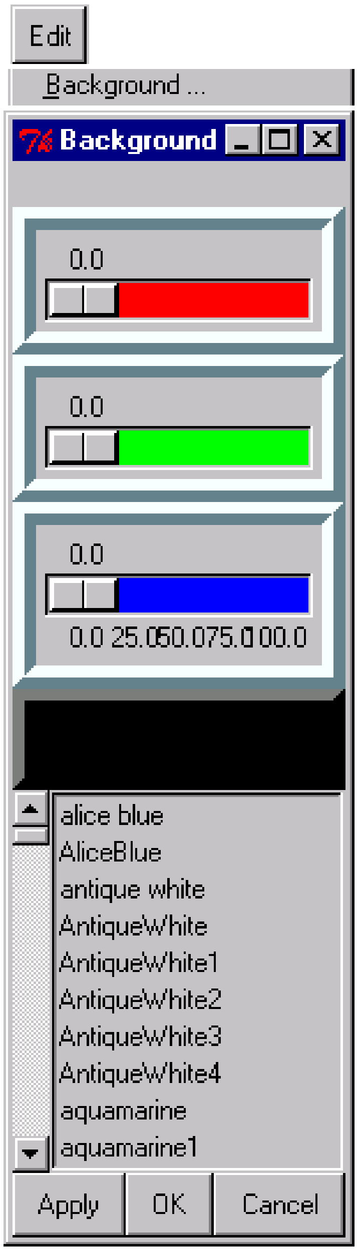


Figure 5.21: *Graphics window background setup*

Appendix A

Mathematical Background

In solid and liquid materials heat transfer and viscous fluid flow are governed by heat and Navier-Stokes equations, which can be derived from the basic principles of conservation of mass, momentum and energy. Fluid can be either Newtonian or non-Newtonian. In the latter case the consideration in ELMER is limited to purely viscous behaviour with the power-law model.

In the following we present the governing equations of fluid flow, heat transfer and stresses in elastic material applied in ELMER. Also the most usual boundary conditions applied in computations are described.

A.1 The Governing Equations

The heat equation is expressed as

$$\rho c_p \left(\frac{\partial T}{\partial t} + (\vec{u} \cdot \nabla) T \right) - \nabla \cdot (k \nabla T) = \rho h, \tag{A.1}$$

where ρ is the density, c_p the heat capacity at constant pressure, T the temperature, $\vec{u}$ the convection velocity, k the heat conductivity and h is source of heat. The equation is derived from the Fourier heat conduction law and the energy conservation equation.

The momentum and continuity equations can be written as

$$\rho \left(\frac{\partial \vec{u}}{\partial t} + (\vec{u} \cdot \nabla) \vec{u} \right) - \nabla \cdot \overline{\overline{\sigma}} = \rho \vec{f}, \tag{A.2}$$

and

$$\left(\frac{\partial \rho}{\partial t} + (\vec{u}\cdot\nabla)\rho\right) + \rho(\nabla\cdot\vec{u}) = 0, \tag{A.3}$$

where $\overline{\overline{\sigma}}$ is the stress tensor. For Newtonian fluids

$$\overline{\overline{\sigma}} = 2\mu\overline{\overline{\varepsilon}} - \frac{2}{3}\mu(\nabla\cdot\vec{u})\overline{\overline{I}} - p\overline{\overline{I}}, \tag{A.4}$$

where μ is the viscosity, p is the pressure, $\overline{\overline{I}}$ the unit tensor and $\overline{\overline{\varepsilon}}$ the linearized strain rate tensor, i.e.

$$\varepsilon_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right). \tag{A.5}$$

The density of an ideal gas depends on the pressure and temperature through the equation of state

$$\rho = \frac{p}{RT}, \tag{A.6}$$

where R is the gas constant:

$$R = \frac{\gamma - 1}{\gamma}c_p. \tag{A.7}$$

The specific heat ratio γ is defined as

$$\gamma = \frac{c_p}{c_v}, \tag{A.8}$$

where c_p and c_v are the heat capacities in constant pressure and volume, respectively. The value of γ depends solely on the internal molecular properties of the gas.

An imcompressibe flow is characterized by the condition ρ=constant, from which it follows that

$$\nabla\cdot\vec{u} = 0. \tag{A.9}$$

Enforcing the constraint (A.9) in (A.2), (A.3) and (A.4), the equations reduce to the Navier-Stokes equations

$$\begin{aligned}\rho\left(\frac{\partial\vec{u}}{\partial t} + (\vec{u}\cdot\nabla)\vec{u}+\right) - \nabla\cdot(2\mu\overline{\overline{\varepsilon}}) + \nabla p &= \rho\vec{f}, &\text{(A.10)}\\ \nabla\cdot\vec{u} &= 0. &\text{(A.11)}\end{aligned}$$

Compressible flows are modelled by the equations (A.2)-(A.8). Then, it is possible to replace the state equation (A.6) by

$$\rho = \frac{1}{c^2} p, \tag{A.12}$$

where $c = c(p, T, \ldots)$ is the speed of sound. The equation (A.12) can be used with liquid materials as well.

Most commonly the term $\rho \vec{f}$ represents a force due to gravity, in which case the vector $\vec{f}$ is the gravitational acceleration. It can also represent, for instance, the Lorenz force when magnetohydrodynamic effects are present.

In pure heat transfer problems the equation (A.1) is applied, whereas in isothermal flows only the equations presented in (A.10) and (A.11) are taken into account.

For coupled fluid flows we assume that the Boussinesq approximation is valid. This means that the density of the fluid is constant except in the body force term where the density depends linearly on temperature through the equation

$$\rho = \rho_0(1 - \beta(T - T_0)), \tag{A.13}$$

where β is the volume expansion coefficient and the subscript 0 refers to a reference state. Assuming that the gravitational acceleration $\vec{g}$ is the only external force, then the force $\rho_0 \vec{g}(1 - \beta(T - T_0))$ is caused in the fluid by temperature variations. This phenomenon is called Grashof convection or natural convection.

The solidification phase change model ELMER uses is an enthalpy based method. The enthalpy is defined to be

$$H(T) = \int_0^T \left(\rho c_p + \rho L \frac{\partial f}{\partial \lambda} \right) d\lambda, \tag{A.14}$$

where $f(T)$ is the fraction of solid material as a function of temperature, and L is the latent heat. The enthalpy-temperature curve is used to compute an effective heat capacity, whereupon the equations become identical to the heat equation. There are two ways of computing the effective heat capacity in ELMER:

$$c_{p,\text{eff}} = \frac{\partial H}{\partial T}, \tag{A.15}$$

and

$$c_{p,\mathrm{eff}} = \left(\frac{\nabla H \cdot \nabla H}{\nabla T \cdot \nabla T} \right)^{1/2}. \tag{A.16}$$

The former method is used only if the local temperature gradient is very small, while the latter is the preferred method. In transient simulations a third method is used, given by

$$c_{p,\mathrm{eff}} = \frac{\partial H / \partial t}{\partial T / \partial t}. \tag{A.17}$$

Frictional heating may be applied by introducing a velocity dependent heat source

$$h_f = 2\mu \overline{\overline{\varepsilon}} : \overline{\overline{\varepsilon}}. \tag{A.18}$$

The magnetic induction equation describes interaction of ionized plasma and applied and flow induced magnetic fields (refer to Appendix **??**). The magnetic induction equation is given by

$$\frac{\partial \vec{B}}{\partial t} - \frac{1}{\sigma \mu_0} \nabla \times \nabla \times \vec{B} - \nabla \times \vec{u} \times \vec{B} = 0, \tag{A.19}$$

where σ is the electrical conductivity of the material and μ_0 the magnetic permeability. The magnetic field induced force term for the flow momentum equations is defined as

$$\vec{f_m} = \vec{J} \times \vec{B}, \tag{A.20}$$

The Joule heating is given by

$$h_m = \frac{1}{\sigma} \vec{J} \cdot \vec{J}. \tag{A.21}$$

In the above equations, $\vec{B}$ and $\vec{E}$ are the magnetic and electric fields, respectively. The current density $\vec{J}$ is defined as

$$\vec{J} = \sigma(\vec{E} + \vec{u} \times \vec{B}). \tag{A.22}$$

ELMER has also a linear elastic stress analysis module, which is meant primarily for computing thermal stresses. The equation of stress for isotropic elastic media is

$$-\nabla(\lambda \nabla \cdot \vec{d}) - \nabla \cdot (2\mu \overline{\overline{\varepsilon}}) + \nabla((3\lambda + 2\mu)\beta(T - T_0)) = \vec{F}, \tag{A.23}$$

where $\vec{d}$ is the displacement vector, β the heat expansion coefficient, and $\vec{F}$ a volume force, λ and μ are the Lamé parameters

$$\lambda = \frac{E\nu}{(1+\nu)(1-2\nu)}, \quad \mu = \frac{E}{2(1+\nu)}, \tag{A.24}$$

E is Young's modulus, ν is the Poisson ratio, and $\overline{\overline{\varepsilon}}$ is the linearized strain tensor

$$\varepsilon_{ij} = \frac{1}{2}\left(\frac{\partial d_i}{\partial x_j} + \frac{\partial d_j}{\partial x_i}\right). \tag{A.25}$$

The stress tensor can be computed as a post processing step (one can actually do this inside the ELMER Post program) using the stress-strain relations for elastic media:

$$\overline{\overline{\sigma}} = \lambda(\nabla \cdot \vec{d})\overline{\overline{I}} + 2\mu\overline{\overline{\varepsilon}} - (3\lambda + 2\mu)\beta(T - T_0)\overline{\overline{I}}. \tag{A.26}$$

In ELMER one can choose between transient and steady state analysis. In transient analysis one has to set, besides boundary conditions, also initial values for the unknown variables.

A.2 The Boundary Conditions

For temperature one can apply boundary conditions and have either temperature or heat flux prescribed.

Dirichlet boundary condition (temperature is prescribed) reads as

$$T = T_b. \tag{A.27}$$

The value of T_b can be constant or a function of time, position or other variables.

Heat flux depending on heat transfer coefficient α and external temperature T_{ext} may be written as

$$-k\frac{\partial T}{\partial n} = \alpha(T - T_{ext}). \tag{A.28}$$

Both variables α and T_{ext} can be constant or functions of time, position or other variables. If the heat transfer coefficient α is equal to zero, it means that the heat flux on a boundary is identically zero. The Neumann boundary condition $-k\partial T/\partial n = 0$ is also used in a symmetry axis in 2D, axisymmetric or cylindrical problems.

Heat flux can consist of idealized radiation whereupon

$$-k\frac{\partial T}{\partial n} = \sigma\varepsilon(T^4 - T_{\mathrm{ext}}^4). \tag{A.29}$$

Above, σ is the Stefan-Boltzmann constant and ε the surface emissivity. The emissivity and the external temperature can again be constant or functions of time, position, or other variables.

If the surface k is receiving radiation from other surfaces in the system, then the heat flux reads as

$$-k_k\frac{\partial T_k}{\partial n_k} = \sigma\varepsilon_k(T_k^4 - \frac{1}{A_k\varepsilon_k}\sum_{i=1}^{N} G_{ik}\varepsilon_i T_i^4 A_i), \tag{A.30}$$

where the subscripts i and k refer to surfaces i and k, and the parameters A_i and A_k to the specific surface areas. The factors G_{ik} are Gebhardt factors, and N represents the total number of radiating surfaces present in the system. Emissivities are assumed to be constant on each surface.

One may also give an additionl heat flux term as

$$-k\frac{\partial T}{\partial n} = q. \tag{A.31}$$

Similarly, boundary conditions for velocity components and tangential or normal stresses can be defined.

In 2D or axisymmetric cases the Dirichlet boundary condition for velocity component u_i is simply

$$u_i = u_i^b. \tag{A.32}$$

A value u_i^b can be constant or a function of time, position or other variables. In cylindrical cases the Dirichlet boundary condition for angular velocity u_θ is

$$u_\theta = \omega, \tag{A.33}$$

where ω is the rotation rate.

In axisymmetric geometries one has to set $u_r = 0$ and $\partial u_z/\partial r = 0$ (in cylindrical problems also $u_\theta = 0$) on the symmetry axis.

If there is no flow across the surface, then

$$\vec{u}\cdot\vec{n} = 0 \tag{A.34}$$

where $\vec{n}$ is the outward unit normal to the boundary.

Surface stresses can be divided into normal and tangential stresses. Normal stress is usually written in the form

$$\sigma_n = \frac{\gamma}{R} - p_a \tag{A.35}$$

where γ is the surface tension coefficient, R the mean curvature and p_a the atmospheric (or external) pressure. Tangential stress has the form

$$\vec{\sigma}_\tau = \nabla_s \gamma, \tag{A.36}$$

where ∇_s is the surface gradient operator.

The coefficient γ is a thermophysical property depending on the temperature. Temperature differences on the surface influence the transport of momentum and heat near the surface. This phenomenon is called Marangoni convection or thermocapillary convection. The temperature dependence of the surface tension coefficient can be approximated by a linear relation:

$$\gamma = \gamma_0(1 - \vartheta(T - T_0)), \tag{A.37}$$

where ϑ is the temperature coefficient of the surface tension and the subscript 0 refers to a reference state. If a Boussinesq hypothesis is made, i.e., the surface tension coefficient is constant except in (A.36) due to (A.37), the boundary condition for tangential stress becomes

$$\vec{\sigma}_\tau = -\vartheta \gamma_0 \nabla_s \gamma. \tag{A.38}$$

In equation (A.35) it holds then that $\gamma = \gamma_0$. The linear temperature dependence of the surface tension coefficient is naturally only one way to present the dependence. In fact, the coefficient γ can be any user defined function in ELMER. One may also give the force vector on a boundary directly as in

$$\overline{\overline{\sigma}} \cdot \vec{n} = \vec{g}. \tag{A.39}$$

For stress analysis one may prescribe the displacement on a boundary as

$$d_i = d_i^b. \tag{A.40}$$

On the remaining boundaries it is possible to prescribe the traction as in (A.39).

A.3 Numerical Methods

As is well known, the convective transport term of the Navier-Stokes equations and the heat equation is a source of both physical and numerical instability. The numerical instability must be compensated somehow in order to solve the equations on a computer. For this reason the so called stabilized finite element method ([**?**],[**?**]) is used in ELMER to discretize these equations. The equations in the form implemented in ELMER and the discretization are described in more detail in Appendix **??**. In Appendix **??** a brief introduction to the methods used in computing the diffuse gray radiation boundary condition is given. In Appendix **??** the free surface problem is presented.

The convection term of the Navier-Stokes equations is nonlinear and has to be linearized for computer solution. There are two linearizations of the convection term in ELMER:

$$(\vec{u} \cdot \nabla)\vec{u} \approx (\vec{\mathcal{U}} \cdot \nabla)\vec{\mathcal{U}} \tag{A.41}$$

and

$$(\vec{u} \cdot \nabla)\vec{u} \approx (\vec{u} \cdot \nabla)\vec{\mathcal{U}} + (\vec{\mathcal{U}} \cdot \nabla)\vec{u} - (\vec{\mathcal{U}} \cdot \nabla)\vec{\mathcal{U}}, \tag{A.42}$$

where $\vec{\mathcal{U}}$ is the velocity vector from the previous iteration. The first of the methods is called Picard iteration or the method of the fixed point, while the latter is called Newton iteration. The convergence rate of the Picard iteration is of first order, and the convergence might at times be very slow. The convergence rate of the Newton method is of second order, but to succesfully use this method, a good initial guess for velocity and pressure fields is required. The solution to this problem is to first take a couple of Picard iterations, and switch to Newton iteration after the convergence has begun.

The heat equation is also nonlinear when radiation is modelled. The nonlinear term in the boundary condition (A.29) can be linearized as

$$T^4 - T_{\mathrm{ext}}^4 \approx (\mathcal{T}^3 + T_{\mathrm{ext}}\mathcal{T}^2 + T_{\mathrm{ext}}^2\mathcal{T} + T_{\mathrm{ext}}^3)(T - T_{\mathrm{ext}}), \tag{A.43}$$

where $\mathcal{T}$ is the temperature from the previous iteration.

When iterating the nonlinearity one may also use relaxation, where the current solution vector is replaced by a vector defined as

$$u_i^{'} = \lambda u_i + (1 - \lambda)u_{i-1}, \tag{A.44}$$

where λ is the relaxation factor. A factor below unity might help to stabilize the nonlinear solver. A factor above unity might speed up the convergence.

Time integeration of the equations is done with the implicit Euler method. Mathematically this reads

$$M\frac{\partial \mathbf{u}}{\partial t} + A\mathbf{u} = F \Rightarrow \frac{1}{\Delta t}M(\mathbf{u}_i - \mathbf{u}_{i-1}) + A\mathbf{u}_i \approx F_i. \tag{A.45}$$

ELMER has both a direct and iterative solver for solving the various linear systems. The direct solver uses a band matrix solver from the LAPACK set of linear algebra routines. This routine is based in LU decomposition. The iterative solver uses the package HUTIter described elsewhere in this document. HUTIter contains various iterative solution algorithms, including stabilized biconjugate gradient (BiCGStab) method, transpose free quasi minimal residual (TFQMR) method, conjugate gradient squared (CGS) method, and generalized minimal residual (GMRES) method. With a large base of experiments behind, we recommend the use of the stabilized biconjugate gradient method, at least with the ELMER Solver.

The iterative methods are almost always not usable for the linear systems in our context without preconditioning. The preconditioning method of choice in ELMER is the Incomplete LU (ILU) decomposition preconditioning. As the name implies, some part of the LU decomposition of the linear system coefficient matrix is computed. The decomposition is then used to stabilize the linear system. Instead of the original equation

$$Ax = b \tag{A.46}$$

we will try to solve the equation

$$(ILU)^{-1}Ax = (ILU)^{-1}b, \tag{A.47}$$

which will hopefully be more stable than the original equation. Experiments have shown this method to be effective. The equation (A.47) is basically the idea behind the method, not the actual algorithm used.

Appendix B

ELMER Geometry File format

B.1 Introduction

This appendix describes the ELMER Geometry File format. This geometry file is for implementing basic geometries which do not actually need the use of a full featured CAD program. The geometry primitives which can be defined in this file are limited to polylines and circular arcs, but in practice it is possible to define quite complicated geometries with these primitives.

The data in the file is devided into sections and these sections must come in a specific order in the input. However, some of the sections are optional. Each section starts with a keyword and ends with the keyword `End`. Data items within a section also start with a specific keyword.

Below is the list of the section keywords in the order they must be in the input. The letter N after the section name indicates that multiple section entries are allowed, but each section entry must be identifed by a unique id number. For example, there can be multiple edge sections in the input, but they must all be after the vertices and before the edge loops and bodies.

`Header`

`Vertices`

`Edge` N (optional)

`Edge Loop` N (optional)

`Body` N

This order means that the geometry is defined hierarchically. Lower lever elements must be given before the higer level elements which refer to them. First all vertices are defined, then the edges using theses vertices, then edge loops and finally the bodies using these edge loops.

However, it is possible to define a body using only edges or vertices. For example, if the geometry consists of one rectangular body, the four corner vertices are enough to define the geometry. On the other hand, if the geometry consists of multiple bodies which have common boundaries, it would be more effective to first define boundary edges which define the bodies, because same edges could be used in multiple bodies. Edge loops would be useful if some bodies are enclosed within other bodies, because then same edge loops could be used in multiple bodies.

Data values can be either integers, reals (decimal numbers) or strings. Corresponding data type identifiers are `Integer`, `Real` and `String`. Normally it is not necesary to specify the type of the data, because it can be concluded from the context,

It is not either necessary to specify the size of the data, because data values are read till the next keyword and this way the size can be concluded from the data itself. The size can be given using the `Size` keyword and one or two integers depending if data forms an array or a table.

If data forms a table, like the vertex points in the Vertices section, then the data type and size specifiers are always needed. The size keyword must be before the type keyword.

Accordingly, for example a list of edges could be given simply:

```
Edges 1 2 3 4
```

or

```
Edges
  Size 4
      1 2 3 4
```

or in the fully sepcified form

```
Edges
  Size 4
    Integer
      1 2 3 4
```

A string value could be given like:

```
  Name "First object"
```

or

```
  Name
    String
      "First object"
```

The layout of the input data is free as long as numeric values are separated with at least one space. Of course, using linefeeds and indenting makes data easier to check and read for users.

On each line, characters after '!' or '#' are interpreted as comments.

B.2 The format

In the following format description one of the letters (`i, r, s`) after a keyword means that some data value is needed in that place and the type of this data should be integer, real or string.

A string value must be given in quotes if it contains spaces or if it is given without a preceiding 'String' data type specifier. Otherwise quotes are not necessary, but it is safer to always quote the strings. Quotes within quotes are not allowed.

All coordinate walues and other geometrical parameters are given in meters.

B.2.1 The `Header` section

```
Header
  Model Name s          ! an optional string
  Model Directory s     ! an optional string
  Dimension i           ! currently only value 2 is accepted
End
```

B.2.2 The `Vertices` section

```
Vertices
```

```
  Points                ! keyword for the point table, must be given

    Variable Index      ! an optional keyword telling that the first
                        ! value  in each data row is and index (id)
                        ! for the vertex

    Size i1 i2          ! table size. If two integer values are given,
                        ! then i1 tells the number rows and i2 tells the
                        ! number of coordinate values per row. If only one
                        ! integer value is given, it tells the number
                        ! coordinate values per row. In this case the table
                        ! must be ended with the End keyword. See the
                        ! example file below

       Real             ! keyword for the data type in the table.

       (i1) r11 r12 ... ! coordinate values. Integers i1, i2, ...
       (i2) r21 r22 ... ! are given only if the keyword Variable Index is
                        ! given for an indexing argument variable (see above)

    (End)               ! this keyword which is needed when only one size
                        ! parameter is given.
End
```

Note: Vertices are always given as a table of points. Vertex coordinates can be either 2D or 3D coordinates. The number of coordinates must always be given (using the `Size` keyword). If an index variable is given in the beginning of each table row, this value is used as the vertex id. Index values should be unique and the index variable (an argument variable in general) is not included in the size parameter. The number of values per data row should refer only to the actual data values. If no index variable is used, vertices are implicitely numbered using integers starting from one.

B.2.3 The `Edge` section

```
Edge i                  ! the unique id number i must be given
   Name s               ! optional, a string

   Geometry s           ! optional, a string, values are: Linear, Circle

   Center r             ! optional, two or three real coordinate values
```

```
                        ! defining the center of a circle.

   Radius r             ! optional, a real value for the circle radius

   Vertices i           ! vertex indices defining either a line
                        ! (can be a polyline) or an circular arc
End
```

Note: a full circle can be defined by given the center and the radius. A circular arc can be defined by the center and two end vertices or by three vertices.

B.2.4 The `Edge Loop` section

```
Edge Loop i             ! the unique id number i must be given

   Edges i              ! the edge ids (in any order) defining the loop
End
```

Note: Although the order of edge ids in the input is not important, these edges should always create a closed loop which does not interect with itself or with any other edges.

B.2.5 The `Body` section

```
Body i                   ! the unique id number i must be given

   Name s                ! optional, a string

   Color i               ! optional, four integers 0...255 (RGB values
                         ! and an alpha value)

   Vertices i| Polygon i ! the vertex indices defining the body

   Edges i               ! the edge ids (in any order) defining the body

   Edge Loops i          ! the loop ids defining the body. The outermost loop
                         ! must always be given first, after that the inner
                         ! loops (if any) in any order
End
```

Note: The keywords Vertices and Polygon are equivalent. Again, the order of edge ids is not important, but these edges should create a well defined edge loop.

B.3 An example file

The following example geometry is taken from the `Boiler.egf` file which can be found in the directory `samples/cad_files` in the ELMER distribution. However, some clarifying comments are added here. This is also the same geometry which is used as an example in the ELMER User Guide. It defines two bodies, a kettle and the water in the kettle. Only the left half of this symmetric geometry is defined here.

```
Header
  Model Name "Boiler" ! model name is optional, it can be defined later
  Dimension 2           ! dimension must be given. Currently always 2
End                     ! each section ends with the keyword End

Vertices
  Points                ! a table of points starts
    Variable Index      ! an 'argument' variable is used to index the points
    Size 2              ! points as 2D coordinates
    Real                ! type must be given for a table
     1   0.000 0.000    ! note: one argument value (vertex id) and a 2D point
     2   0.000 0.010
     3  -0.063 0.010
     4  -0.070 0.030
     5  -0.070 0.110
     6  -0.070 0.120
     7  -0.075 0.120
     8  -0.075 0.030
     9  -0.065 0.000
     10  0.000 0.110
   End                  ! an End is needed to indicate the end of the table
End                     ! this is the section End

Edge 1                  ! edge number one
 Name "Surface"         ! water surface
 Vertices 5 10          ! 5 and 10 are indices in the vertex table
End
```

```
Edge 2
 Name "Water-smtr"      ! the water boundary at the symmetry line
 Vertices 10 2
End

Edge 3
 Name "Inner"           ! the inner boundary between the kettle and water
 Vertices 2 3 4 5       ! this is a polyline edge.
End

Edge 4
 Name "Kettle-srf"      ! the outer boundary of the kettle, excluding
                        ! the bottom
 Vertices 5 6 7 8 9     ! an other polyline edge
End

Edge 5
 Name "Bottom"          ! the bottom of the kettle
 Vertices 9 1
End

Edge 6
 Name "Kettle-smtr"     ! the kettle boundary at the symmetry line
 Vertices 1 2
End


Body 1                 ! body number one is the kettle
  Name "Kettle"
  Color 255 0 0 255    ! color by RGB codes and an alpha value (0...255)
  Edges 3 4 5 6        ! defined by four edges (3 and 4 are polylines!)
End

Body 2                 ! body number two is the water in the kettle
  Name "Water"
  Color 0 0 255 255
  Edges 1 2 3          ! three is the minimal number of edges in 2D!
End
```

Appendix C

ELMER Front settings file format

When an ELMER Front session is started, values for various control variables can be initialized from an input file. This appendix describes the structure of this file and the keywords which are available for setting initial values to these variables.

All keywords given in the file should be within a single section called ‘User Settings’. So, the basic structure for the file is the following:

```
User Settings
   Keyword value
   Keyword value
   ...
End
```

The order of the keywords is free. In the following, all keywords starting with ‘Default’ are keywords whose values can be stored only in a settings file. Other keyword values can be stored also in a model file and they will be used automatically when the model file is loaded. However, the user can prevent this by defining the settings parameter ‘Default Use Model Settings’ to be false. This can be done either in a settings file or by using the Settings command in the Edit menu.

On each line, characters after ‘!’ or ‘#’ are interpreted as comments.

A string value must be given in quotes if it contains spaces or if it is given without a preceiding ‘String’ data type specifier. Otherwise quotes are not necessary, but it is safer to always quote the strings. Quotes within quotes are not allowed.

A logical value can be either 1 (meaning true) or 0 (meaning false).

In the following keywords are in the same order as they are displayed in the Settings window. The meaning of each keyword is explained in the context of the Edit menu in Chapter 2.

The keyword values given below are just for an example. The user should of course enter own values for the keywords which he or she defines in the settings file.

```
Default Model Directory
   String "/elmer/Models"

Default Cad Files Directory
   String "/elmer/data/cad_files"

Default External Mesh Files Directory
   String "/elmer/data/mesh_files"

Default Include Path
   String "/elmer/input"

Default Log Directory
   String "/elmer/work"

Default Results Directory
   String "/elmer/work"

! Should the settings stored in a model file used when it is loaded
!
Default Use Model Settings
   Logical 0

Auto Load Mesh
   Logical 1

Auto Save Model
   Logical 1

Auto Save Solver Input
   Logical 1

! Command for an external logfile browser (here in read only mode)
!
Browser Command
```

```
   String "C:\Program Files\TextPad\TXTPAD32.EXE -r"

! Command for an external editor (for editing Solver input file etc.)
!
Editor Command
   String "C:\Program Files\TextPad\TXTPAD32.EXE"

! The following browse mode flags are available
! Logfile: ouput is directed to a logfile
! Shell:   output goes to system console
! None:    no output at all
!
Browse Mode Gebhardt Factors
   String None

Browse Mode Mesh
   String Shell

Browse Mode Procedure Compiler
   String Logfile

Browse Mode Solver
   String Logfile

Browse Mode View Factors
   String Logfile
```

Appendix D

ELMER Mesh Generator input file format

D.1 Model description

The input model consists of a number of bodies that are divided into layers. The mesh type can be selected separately for each layer. The layers are represented as collections of closed loops of edges: One outer boundary loop and possibly several inner cavity loops. The edges are defined by lists of nodes that define a broken line segment.

Each loop is a sequence of directed edges. The edges are directed so that the defined layer is on the left side of the edges. So the outer loop goes counterclockwise around the layer and the inner loops clockwise around the cavities. Adjacent edges in a loop must share a common node, so that the last node in an edge is the same as the first node in the next one. The first node of the first edge must be the same as the last node of the last edge.

The nodes corresponding to the geometry vertices will have the same ids that the vertices have in the input file. Predetermined inner nodes can also be specified.

D.2 Mesh density control

The mesh density is defined by a h-value, that is, the preferred element side length. The density values can be described at any level of the model hierarchy: A density value can be given for the whole model, for a body, for a layer, for an edge or for a vertex. The density values given for an entity always overrides those given for the higher level entities.

The density can be given as an absolute value or a value relative to the global density value. The actual density values in the middle of a layer are interpolated from the values on the border. Three types of background meshes can be used for the interpolation:

- An explicit background mesh can be given as a set of points with associated density values. Any mesh density values from the model are ignored.
- A delaunay triangulation can be made from the original geometry vertices.
- An orthogonal grid can be used to calculate the density values as a solution to a Laplacian boundary value problem. The grid density can be chosen from three values.

D.3 File structure

The file consists of a hierarchy of records that contain keywords and data. All keywords end with a colon. The file is divided into words separated by any kind of whitespace characters, so the file may be freely decorated by indentation or adding empty lines. The file is case sensitive.

Most of the fields in the records are mandatory and must be given in a fixed order. Several records may contain an optional mesh density field that has one of two keywords “H:” or “R:”. The keyword “H:” is followed by an absolute density value and the keyword “R:” specifies a value relative to the global density value.

The input file is divided into 4 major fields:

- A header
- A list of vertices
- A list of edges
- A list of bodies

There can be lines of comment between these fields. Comment lines begin with an "!" or a "#". The vertex list is only ment for the vertices used for defining edges. The additional inner nodes are listed in the layer records.

D.4 The header

The header starts with the keyword **Geometry2D:** followed by 5 mandatory fields:

- **H:** The global mesh density value.
- **MeshScalingFactor:** The mesh scaling factor. All mesh density values are multiplied by this value.
- **Nodes:** The number of vertex records.
- **Edges:** The number of edge records.
- **Bodies:** The number of body records.

Example:

```
Geometry2D:
H: 0.1
MeshScalingFactor: 1.5
Nodes: 3
Edges: 2
Bodies: 1
```

All the density values in this file will be multiplied with 1.5, including the 0.1 given in the header. The mesh scaling factor is an easy way to change the overall density of the entire mesh.

D.5 The vertices

Each vertex record has the fields:

- **NodeId:** Two positive integers: a unique vertex id and a boundary id. If a boundary element is not required at this vertex, a zero or negative boundary id may be given.

- An optional mesh density field. See above.
- Two vertex coordinates.

Example:

```
NodeId: 1 1
R: 2.0
0 0
NodeId: 2 -1
H: 0.2
1 1
NodeId: 3 1
0 1
NodeId: 4 -1
-1 0
NodeId: 5 -1
-1 1
```

Only vertices 1 and 3 will produce a boundary element. Both the first and second vertices will produce a mesh density of 0.3. The last three vertices do not specify a density of their own.

D.6 The edges

Each edge record has the fields:

- **EdgeId:** two positive integers: a unique edge id and a boundary id.
- An optional mesh density field with an additional keyword “N:” that may be used to specify a fixed number of boundary elements.
- The number of vertices that define the edge. (At least two)
- A list of vertex ids

Example:

```
EdgeId: 1 1
H: 0.05
```

```
3 1 2 3
EdgeId: 2 2
N: 10
2 1 3
EdgeId: 3 3
2 1 4
EdgeId: 4 3
2 4 5
EdgeId: 5 3
2 3 5
```

The first edge is defined by three vertices and specifies a mesh density of 0.075. The second edge is defined by two endpoints and will be divided into 10 equal-sized segments (so there will be 11 nodes on that edge). Because the node 3 didn't specify a mesh density of its own, it will have a density that is the average of the densities of these two edges. That is 0.155.

D.7 The bodies

Each body record has the fields:

- **BodyId:** A unique body identification (a positive integer).
- An optional mesh density field.
- **ElementOrder:** The order of the elements to be created for this body. Either Linear or Parabolic.
- **Layers:** The number of layers in this body.
- A list of layer records.

Each layer record has the fields:

- **LayerId:** A unique layer indentification (a positive integer).
- An optional mesh density field.
- **LayerType:** The layer mesh type. Described below.
- A type-dependent part. Described below.

- **Loops:** The number of loops
- A list of loop records:
  - **LoopId:** A positive integer
  - **Direction:** 1 or -1. If the direction is -1, the loop direction is inverted.
  - **Edges:** The number of edges
  - A list of signed edge ids (negative means backwards)

The layer type can be one of the following:

- **Connect**: Nodes are generated on the layer edges and connected with a delaunay triangulation.
- **VoronoiVertex**: Nodes are inserted one by one into the place that has the greatest distance to the nearest existing node.
- **MovingFront**: Nodes are inserted by taking an accepted mesh segment and inserting a third node so that a good triangle is produced. The entire boundary is used as a starting front.
- **SSMFMovingFront**: The same as MovingFront, but the process is started from a single seed position. Advances to the direction where the distance to the nearest node is greatest.
- **SSSFMovingFront**: The same as SSMFMovingFront, but advances more locally.
- **QuadGrid**: A grid of quadrilateral elements. The layer must have only 1 loop with exactly four edges. The user must also make sure that the opposite edges will be subdivided into equal numbers of elements. This can be done easily with the GridSize-specification described below.
- **TriangleNEGrid**, **TriangleNWGrid**, **TriangleUJNEGrid**, **TriangleUJNWGrid**, **TriangleFBNEGrid** or **TriangleFBNWGrid**: A QuadGrid that is split into triangles. NE/NW specifies the splitting direction of the first quadrilateral. UJ means an alternating direction and FB is a sceme that alternates only along the first and third edges.

The type-dependent fields in a layer of type VoronoiVertex, MovingFront, SSSFMovingFront or SSMFMovingFront:

- **FixedNodes:** The number of fixed mesh nodes and a list of vertex records.
- **BGMesh:** Background mesh type: One of Delaunay, Grid, SparseGrid, DenseGrid or Explicit.
- If background mesh is Explicit:
  Background mesh size and a list of nodes with two coordinates and a mesh density value. The other selections require no additional parameters.

If the type is SSSFMovingFront of SSMFMovingFront, the following additional fields:

- **Seed:** Seed type: Implicit or Explicit.
  If type is Explicit:
  - **Nodes:** Three node ids that define the initial front.

  If type is Implicit:
  - **Edge:** An edge id. A segment from the middle of the edge is used a the initial front.

Example:

```
BodyId: 1
ElementOrder: Linear
Layers: 1
  LayerId: 1
  R: 0.5
  LayerType: SSSFMovingFront
  FixedNodes: 1
    NodeId: 4 -1 0.5 0.5
  BGMesh: Grid
  Seed: Implicit
    Edge: 2
  Loops: 1
    LoopId: 1
    Direction: 1
    Edges: 2
      1 -2
```

This will create a mesh with one predetermined node at position (0.5, 0.5). The edge 2 has a negative sign, so the nodes of the edge are taken in a reverse order. Thus this body is a single triangle formed by nodes 1 2 and 3. The triangulation will start a the middle of edge 2. The mesh density is controlled by a cartesian grid.

The type-dependent fields in a layer of type QuadGrid, TriangleNEGrid, TriangleNWGrid, TriangleUJNEGrid, TriangleUJNWGrid, TriangleFBNEGrid or TriangleFBNWGrid are:

- **GridSize:** The number of elements in which to split the odd and even edges of the layer. (Two positive integers.) If this field is omitted the edges are split as defined by the density values. The values given in a GridSize-field override any density values given for the edges.

Example:

```
BodyId: 2
ElementOrder: Linear
Layers: 1
  LayerId: 1
  R: 0.5
  LayerType: QuadGrid
  GridSize: 5 10
  Loops: 1
    LoopId: 1
    Direction: -1
    Edges: 4
4 -5 2 3
```

This body will be a grid of 50 quadrilateral elements. The outer loop is given clockwise, but the direction field tells the program to interpret it backwards. Another way to specify the grid size could be to give “N: 5” to edges 3 and 5 and “N: 10” to edges 2 and 4.

Appendix E

ELMER Mesh File Format

In this appendix the ELMER mesh file format is described. The mesh is written by a mesh generator, and used within the ELMER Solver.

The mesh consists of four separate files: `mesh.header`, `mesh.nodes`, `mesh.elements`, and `mesh.boundary`. The contents of all the files will be described.

In the mesh files numeric element type codes are being used. The type codes along with the node numbering order of the elements have been described in appendix I.

E.1 The header file format

The contents of the `mesh.header` file is as follows

```
nodes elements boundary-elements
nof_types
type-code nof_elements
type-code nof_elements
...
```

In the first line of the file the number of nodes, elements, and boundary elements in the model are given. Second line gives the number of different element types used in the mesh followed by lines with type code, and number of elements of that type included in the mesh.

As an example, the following header file

```
300 261 76
2
404 261
202 76
```

tells us that the mesh is composed of 300 nodes, 261 bulk elements, 76 boundary elements. There are 2 different element types used in the mesh, bilinear quads (type code 404) and linear line elements (type code 202).

E.2 The node file format

In the `mesh.nodes` file each line contains two integer numbers and three real numbers:

```
1 p x y z
2 p x y z
 ...
n p x y z
```

First number in any given line is the order number of the node. Nodes are not necessarily (allthough usually) listed in ascending order. Second number is a partition index for parallel execution and is usually not needed for sequential runs. The partition number may be set to the value -1 (or 1), if not needed. Next the spatial coordinates of the nodal point are given. Three coordinates must allways be given, even if the simulation would be in 1D or 2D.

E.3 The element file format

The `mesh.elements` file consists of lines of type

```
1 body type n1 ... nn
2 body type n1 ... nn
...
n body type n1 ... nn
```

where first entry in a line is the order number of the element, `body` is the simulation body which this element is part of. Next in the element definition is the element type code followed by the nodal indices, defined in the `mesh.nodes` file, which make out the element geometry.

For example, the following lines might describe the first few elements in a mesh:

```
1 1 404 1 2 32 31
2 1 404 2 3 33 32
3 1 404 3 4 34 33
4 1 404 4 5 35 34
...
```

E.4 The boundary element file format

The `mesh.boundary` file is similiar to the `mesh.elements` file:

```
1 bndry p1 p2 type n1 ... nn
2 bndry p1 p2 type n1 ... nn
...
n bndry p1 p2 type n1 ... nn
```

the first entry in a line is again the order number of the boundary element. Next there is now the boundary identification number. The following two integer numbers `p1` and `p2` refer to the order number of the (parent) elements, defined in the `mesh.elements` file, this boundary element belongs to. If the boundary element is on an outer boundary, there is only one parent and either of the parent indices may be set to zero. Next in the boundary element definition is the element type code followed by the nodal indices, which make out the element geometry.

Appendix F

ELMER Solver Input File Format

In this appendix we describe the solver input file format in detail. This file is usually written by the ELMER Front, but can also be generated by other means or modified by the user to suit spesific needs.

For the programmers: the solver input file reader has no knowledge of the content of the different sections in this file except of the `Header` section. So in any section you can add name-value pairs whenever needed without breaking anything. These values may then be accessed by calling them by the name provided. The utility routines `ListGet*` may be used for this purpose.

Because of the generality of the format, the types of the parameters must be given with their values. Also there are a few generalizations of the format given below.

F.1 Array Variables

Any parameter typed `Real` or `Integer` may be given keyword `size n1 n2`, where `n2` is optional and when given, the statement defines a two dimensional array instead of a one dimensional. An example of an array variable is the gravity vector declared in the constants section:

```
Gravity
  Size 4
    Real 0 1 0 9.82
```

The above statement gives a real vector whose length is four. In this case the first three components give the direction vector of the gravity and the fourth component gives its intensity.

F.2 Parameters Depending on Field Variables or Time

Any parameter typed `Real` depending on some field variable or time, may be given as a piecewise linear spline as follows:

```
Parameter-Name
  Variable Variable-Name
    Real
      Variable-Value-1 Parameter-Value-1
      Variable-Value-2 Parameter-Value-2
                   ...
      Variable-Value-n Parameter-Value-n
    End
```

If the `size` keyword is also given, and differs from one, the parameter array values are given instead of the single parameter value. The variables on which a parameter may depend are, for example:

- `Time`,
- `Temperature`,
- `Pressure`,
- `Velocity 1, Velocity 2, Velocity 3`,
- `Coordinate 1, Coordinate 2, Coordinate 3`,
- `Displacement 1, Displacement 2, Displacement 3`,
- `Magnetic Field 1, Magnetic Field 2, Magnetic Field 3`,
- `Electric Current 1, Electric Current 2, Electric Current 3`.

The solution vectors are of course only available, if the simulation is such, that they are being computed. A programmer may add more variables using the utility routine `VariableAdd`.

In the Marangoni convection example later in this chapter, there is a temperature boundary condition depending on x-coordinate, which is defined as follows:

```
Temperature
  Variable Coordinate 1
    Real
       0  0.5
       1 -0.5
    End
```

This says, that the value of temperature is 0.5 degrees when x-coordinate is 0, the value of temperature is -0.5 degrees when x-coordinate is 1, and that the value of temperature will be linearily varying with the x-coordinate in between these values.

F.3 User Defined Functions

Any parameter typed `Real`, `Integer` or `Logical` may be given keyword `Procedure` instead of the value of the parameter. Then a user defined function will be executed when the parameter value is needed. The syntax of this statement is:

```
Parameter-Name
  Real Procedure "Filename" "Function-name"
```

The `Filename` is the name of the file of the dynamically loadable code (called a shareable image on Unix and a DLL on WINDOWS). As a very simple example, the following Fortran 90 function would return the value of the surface tension coefficient (as in the Marangoni convection example later in this chapter), when needed:

```
FUNCTION SurfTension(Model,n,T) RESULT(Tension)
  !DEC$ATTRIBUTES DLLEXPORT :: SurfTension
  USE Types
  IMPLICIT NONE

  TYPE(Model_t) :: Model
  INTEGER :: n
  REAL(KIND=dp) :: T
  REAL(KIND=dp) :: Tension

  Tension = 548*(1-T)
END FUNCTION SurfTension
```

If this function is in the file `Surf.f90`, the following command could be used to compile and link the program on a Silicon Graphics or DEC workstation (among others):

```
f90 -I$ELMER_HOME/include -o Surf -shared Surf.f90
```

On a Windows NT workstation, using the Digital (Compaq) Visual Fortran (the Fortran 90 comment line `!DEC$ATTRIBUTES...` is actually a compiler directive for this compiler), the command would be:

```
f90 -I%ELMER_HOME%\include -o Surf -dll Surf.f90
```

After the function has been compiled, it can be used to define the surface tension coefficient in the solver input file as

```
Surface Tension Coefficient
   Variable Temperature
     Real Procedure "Surf" SurfaceTension
```

The user function will receive as first argument the model structure holding pointers to all information about the model, solved field variables, materials, etc. It also holds a pointer to the element that is being assembled at the moment when the function is called. The second argument is the number of the node for which the function should return its value and if the parameter in question is defined to be varying with some field variable or time, the third argument will contain the value of that variable. The user functions may use the utility routines defined in the solver to get hold of other parameter values, field variable values, etc.

F.4 The Format

F.4.1 The `Header` Section

The first section in the input file must be the header section defining various dimensions: number of model bodies, number of body force definitions, number of separate boundaries, number of boundary condition definitions, number of equation set definitions, number of initial condition definitions and finally number of equation solver definitions. Also the path to the mesh database is given in this section. The syntax of the header section is as follows:

```
Header
  Mesh DB "Dir" "Subdir"
  Bodies                n
  Body Forces           n
  Boundaries            n
  Boundary Conditions   n
  Equations             n
  Initial Conditions    n
  Materials             n
  Solvers               n
End
```

The `Mesh DB` setting is relative to current model directory, and gives the directory where the `mesh.*` files (chapter E) are located. Also viewfactors, gebhardt factors, and the result files are written to this directory.

For every solver used in the model one may define its own mesh directory (and thus own meshes) in the `Solver` section described later. The mesh directory definition in the `Solver` section overrides the the mesh directory setting of the `Header` section. However, those `Solver` sections that don't use their own meshes will still use the one defined in the `Header` section.

Solver result files may also be redirected to different place by including an optional keyword `Results Directory "ResultsDir"`. The directory given with this keyword must also have the same directory hierarchy as the model directory.

Note that the `Header` section is the only section whose contents is known to the solver input reader.

F.4.2 The `Constants` Section

The constants section must always be present and hold at least direction and intensity of the gravity vector and value of the Stefan-Bolzmann constant.

```
Constants
  Gravity
    Size 4
      Real  x y z abs
```

```
   Stefan Bolzmann
      Real
End
```

F.4.3 The `Simulation` Section

The simulation section gives the case control data:

- `Simulation Type`: a character string containing either the keyword `Transient` or `Steady State`.
- `Coordinate Mapping`: mapping of the mesh coordinates to the order used in the solver: (x, y, z) in cartesian coordinates, (r, z, θ) in cylindrical coordinates and (r, θ, ϕ) in polar coordinates. The mapping is a triplet of integers giving the order number of the solver coordinate variables in the mesh coordinate arrays.
- `Coordinate System`: a character string defining the coordinate system to be used, one of: `Cartesian 1D`, `Cartesian 2D`, `Cartesian 3D`, `Polar 2D`, `Polar 3D`, `Cylindric`, `Cylindric Symmetric` and `Axi Symmetric`.
- `Gebhardt Factors`: If the model includes diffuse gray radiation, the file containing the Gebhardt factors must be given. This file is written by the program `GebhardtFactors` as a preprocessing step.
- `View Factors`: If the model includes diffuse gray radiation, the file containing the view factors must be given to the program computing the Gebhardt factors. This file is written by the program `Viewfactors` as a preprocessing step. The tasks of computing view factors and the Gebhardt factors have been divided, because the view factors depend only on geometry (and thus the mesh), while the Gebhardt factors also depend on the boundary emissivities.
- `Timestepping Method`: the possible values of this parameter are `Newmark` (an additional parameter `Newmark Beta` must be given), `BDF` (`BDF Order` must be given). Also as a shortcut to `Newmark`-method with values of `Beta=0.0,0.5,1.0` the keywords `Explicit Euler`, `Crank-Nicolson`, and `Implicit Euler` may be given respectively. The recommended choice for the first order time integration is the BDF method of order 2.

- `BDF Order`: integer value from 1 to 5.

- `Newmark Beta`: a real value in range from 0.0 to 1.0. The value 0.0 equals to the explicit Euler integration method and the value 1.0 equals to the implicit Euler method.

- `Timestep Intervals`: timesteps are given in intervals of fixed sizes, this integer array gives the number of timesteps within each of the intervals. The number of intervals must be given with the `size` specifier.

- `Timestep Sizes`: a real array of the same size as given with the `Timestep Intervals` keyword, giving size of the timesteps within the intervals.

- `Output File`: Name of the output file. Format of the output file is described in Section G. The output file will be written to the directory given with the `Mesh DB` keyword in the `header` section.

- `Output Intervals`: an integer array of the same size as given with the `Timestep Intervals` keyword, giving the update frequency of the output file within the timestep intervals. For steady state simulation, there is only one interval.

- `Post File`: If this keyword is given, mesh and results are written to the file in the format understood by Elmer Post. The post processing file will be written to the directory given with the `Mesh DB` keyword in the `header` section.

- `Restart File`: The format of the restart file is the same as that of the `Output File`. The restart file gives field variable values, which will replace the values of the variables inside the solver after initial conditions have been set. The format of the file allows only some of the field variables to be given. The restart file will be read from the directory given with the `Mesh DB` keyword in the `header` section.

- `Restart Position`: an integer giving the order number of the timestep or steady state iteration output within the restart file. If this number is larger than the number of timesteps in the file or is set to zero, last timestep found is used. If this keyword is not given, the first usable data from the file will be used.

- `Steady State Max Iterations`: Maximum number of steady state or coupled system iterations within one timestep for time dependent simulations. If equation specific convergence tolerances, set in the `Solver` sections, are achieved before the iteration count is exceeded, they will terminate the iteration instead.

```
Simulation
  BDF Order                      Integer [1-5]
  Coordinate Mapping
    Size 3
      Integer
  Coordinate System              String  [Cartesian 1D]
                                         [Cartesian 2D]
                                         [Cartesian 3D]
                                         [Cylindric Symmetric]
                                         [Axi Symmetric]
                                         [Cylindric]
                                         [Polar 2D]
                                         [Polar 3D]
  Gebhardt Factors               File
  Newmark Beta                   Real    [0-1]
  Output File                    File
  Output Intervals               Integer Array
  Post File                      File
  Restart File                   File
  Restart Position               File
  Simulation Type                String  [Transient]
                                         [Steady State]
  Steady State Max Iterations    Integer
  Timestep Intervals             Integer Array
  TImestepping Method            String  [BDF]
                                         [Newmark]
                                         [Implicit Euler]
                                         [Explicit Euler]
                                         [Crank-Nicolson]
  Timestep Sizes                 Real Array
  View Factors                   File
End
```

F.4.4 The `Solver` Section

The solver section defines equation solver control variables:

- `Equation`: a character string containing the name of the equation this solver is intended for, for example: `Navier-Stokes`, `Heat Equation`, `Stress Equations` or `KE Turbulence`. It is possible to add more equation solvers to the system, and thus this set of strings can not be enumerated here. The name set here should match at least one of the settings in one or more of the `Equation` sections included in the `.sif` file.

- `Linear System Solver`: a character string containing either `Iterative` or `Direct`. The specific algorithm to be used must be given with either `Linear System Direct Method` or `Linear System Iterative Method` keywords.

- `Linear System Direct Method`: either the string `BANDED` or the string `SPARSE`. The `BANDED` choice selects the *LAPACK* band matrix solver as the linear system solver. If the `SPARSE` keyword is given instead, the solver uses *Sparse*-library from the University of California as the linear system solver. This keyword must be given if the direct linear system solver is selected.

- `Linear System Iterative Method`: a character string containing name of the iterative method to be used: `BiCGStab`, `CGS`, `TFQMR`, `GMRES` or `CG`. Beware, that the last of the choices is for symmetrical linear systems, which is not the case for any of the default equations in our context. This keyword must be given, if the iterative linear system solver is selected.

- `Linear System Symmetric`: logical value. Symmetric linear systems may be flagged with this keyword. Some computer memory space is spared if this keyword is given, as well as CPU time.

- `Linear Solver Max Iterations`: the maximum number of iterations the iterative solver is allowed to do. If the residual of the linear system is small enough before the iteration count is met, that condition will terminate the iteration instead.

- `Linear Solver Preconditioning`: a character string containing the name of the preconditioning method for iterative solver, one of: `None`, `Diagonal`, `ILU`. The choice `ILU` (ILU stands for Incomplete LU decomposition) is the recommended choice, and usually the iterative solver won't be able to solve the equations without this preconditioning.

- `Linear System Convergence Tolerance`: a threshold value giving the value of the residual to terminate the iteration. More accurately, the termination criterion is

$$||Ax - b|| < \epsilon ||b||,$$

where ϵ is the value given with this keyword.

- `Nonlinear System Convergence Tolerance`: this keyword gives a criterion to terminate the nonlinear iteration after the relative change of the norm of the field variable between two consecutive iterations is small enough

  $$||u_i - u_{i-1}|| < \epsilon ||u_i||,$$

  where ϵ is the value given with this keyword.

- `Nonlinear System Max Iterations`: The maxmimum number of nonlinear iterations the solver is allowed to do.

- `Nonlinear System Newton After Iterations`: Change the nonlinear solver type to Newton iteration after a number of Picard iterations have been performed. If a given convergence tolerance between two iterations is met before the iteration count is met, it will switch the iteration type instead.

- `Nonlinear System Newton After Tolerance`: Change the nonlinear solver type to Newton iteration, if the relative change of the norm of the field variable meets a tolerance criterion:

  $$||u_i - u_{i-1}|| < \epsilon ||u_i||,$$

  where ϵ is the value given with this keyword.

- `Nonlinear System Relaxation Factor`: Giving this keyword triggers the use of relaxation in the nonlinear equation solver. Using a factor below unity is sometimes required to achive convergence of the nonlinear system. A factor above unity might speed up the convergence. Relaxed variable is defined as follows:

  $$u_i^{'} = \lambda u_i + (1 - \lambda) u_{i-1},$$

  where λ is the factor given with this keyword. The default value for the relaxation factor is unity.

- `Steady State Convergence Tolerance`: With this keyword a equation specific steady state or coupled system convergence tolerance is given. All the active equation solvers must meet their own tolerances before the whole system is deemed converged. The tolerance criterion is:

$$||u_i - u_{i-1}|| < \epsilon ||u_i||,$$

where ϵ is the value given with this keyword.

- `Stabilize`: If this flag is set true the solver will use stabilized finite element method when solving the Navier-Stokes equations, and heat equation with a convection term (among others). Usually stabilization of the equations must be done in order to succesfully solve the equations.

  If solving for the compressible Navier-Stokes equations, a bubble function formulation is used instead of the stabilized formulation regardless of the setting of this keyword. Also for the incompressible Navier-Stokes equations, the bubbles may be selected by setting this flag to `False`.

  This setting has no meaning for the stress analysis type, for example, at least not at the moment.

- `Mesh`: a character string giving solvers own mesh directory, which may not be the same as defined in the header section nor the same as any other solvers mesh directory. The mesh used by the equation solver will be read from this directory, and all the results from the equation solver will also be written to this directory.

```
Solver id
  Equation                                  String [Navier-Stokes]
                                                   [Heat Equation]
                                                   [KE Turbulence]
                                                   [Stress Equations]
                                                   [Magnetic Induction]
                                                   [...]
  Linear System Convergence Tolerance       Real
  Linear System Direct Method               String [BANDED][SPARSE]
  Linear System Iterative Method            String [BiCGStab][CGS]
                                                   [TFQMR][GMRES][CG]
  Linear System Max Iterations              Integer
  Linear System Preconditioning             String [None][Diagonal]
                                                       [ILU]
  Linear System Solver                      String [Iterative]
                                                   [Direct]
```

```
  Linear System Symmetric                      Logical
  Nonlinear System Convergence Tolerance       Real
  Nonlinear System Max Iterations              Integer
  Nonlinear System Newton After Iterations     Integer
  Nonlinear System Newton After Tolerance      Real
  Nonlinear System Relaxation Factor           Real
  Stabilize                                    Logical
  Mesh                                         String
End
```

An external equation solver is flagged with a few additional keywords:

- `Procedure`: with this keyword a file containing the precompiled external solver is given along with the name of the subroutine to call.
- `Variable DOFs`: integer value giving the number of components, the external solver is designed for.
- `Variable`: with this keyword the name of the variable to be solved by this solver is given.
- `Time Derivative Order`: if the equation discretized by this solver has second order time derivatives, it must be flagged with this keyword.

```
  Procedure
    File "Object File Name" "Procedure Name"

  Variable                String
  Variable DOFs           Integer
  Time Derivative Order   Integer
```

F.4.5 The `Body` Section

The body section defines which of the equations, initial conditions, materials, and body forces defined in the input file, should be used for the given simulation body. All these sections are described in detail below.

```
Body id
  Body Force             Integer
```

```
  Equation               Integer
  Material               Integer
  Initial Condition      Integer
End
```

F.4.6 The `Equation` Section

The equation section is used to define a set of equations for a body or set of bodies:

- `Navier-Stokes`: if set to `True`, solve the Navier-Stokes equations.
- `KE Turbulence`: if set to `True`, solve the k-epsilon turbulence model along with Navier-Stokes equations.
- `Heat Equation`: if set to `True`, solve the heat equation.
- `Magnetic Induction`: if set to `True`, solve the magnetic induction equation along with the Navier-Stokes equations.
- `Stress Analysis`: if set to `True`, solve the stress equations.

If an equation solver is added to ELMER Solver externally, this list expands accordingly. Note that the `Equation` setting in the `Solver` section must match the name of the corresponding variable in the `Equation` section.

Some controlling variables may also be set in the `Equation` section

- `Convection`: a character string giving the convection type to be used in the heat equation, one of: `None`, `Computed`, `Constant`.
- `Stress Model`: a character string containing either `Mechanical` or `Thermal`. If the latter choice is used heat equation must also be solved.
- `Phase Change Model`: a string value, one of: `None`, `Spatial 1`, `Spatial 2`, and `Temporal`. Note that when soldification is modelled, the enthalpy-temperature- and viscosity-temperature-curves must be defined in the material section.

```
Equation id
  Convection             String [None][Computed][Constant]
```

```
  Heat Equation          Logical
  KE Turbulence          Logical
  Magnetic Induction     Logical
  Navier-Stokes          Logical
  Phase Change Model     String  [None]
                                 [Spatial 1]
                                 [Spatial 2]
                                 [Temporal]
  Stress Analysis        Logical
  Stress Model           String [Mechanical][Thermal]
End
```

For the external solvers, for example the advection-diffusion solver in ELMER distribution or any user defined external solvers, the name of the parameter much match the `Equation` setting defined in the `Solver` section.

F.4.7 The `Body Force` Section

The body force section may be used to give additional force terms for the equations. The following keywords are recongnized by the base solver:

- `Bussinesq`: a logical value, if set true, sets the Bussinesq model on.
- `Flow BodyForce 1,2,3`: may be used to give additional body force for the flow momentum equations.
- `Heat Source`: an additional heat source for the heat equation may be given with this keyword.
- `Stress Bodyforce 1,2,3`: an additional force term for the stress equations may be given with these keywords.
- `Lorenz Force`: a logical value, if set true, triggers the magnetic field force for the flow mementum equations, and the Joule heating for the heat equation.
- `Friction Heat`: a logical value, if set true, triggers use of the frictional heating:

$$h_f = 2\mu\varepsilon^2. \tag{F.1}$$

```
Body force id
  Bussinesq                                         Logical
  Flow Bodyforce 1                                  Real
  Flow Bodyforce 2                                  Real
  Flow Bodyforce 3                                  Real
  Heat Source                                       Real
  Stress Bodyforce 1                                Real
  Stress Bodyforce 2                                Real
  Stress Bodyforce 3                                Real
  Lorenz Force                                      Logical
  Friction Heat                                     Logical
 End
```

F.4.8 The `Initial Condition` Section

The initial codition section may be used to set initial values for the field variables. The following variables are active:

- `Pressure`
- `Temperature`
- `Velocity 1, Velocity 2, Velocity 3`
- `Displacement 1, Displacement 2, Displacement 3`
- `Magnetic Field 1, Magnetic Field 2, Magnetic Field 3`
- `Electric Current 1, Electric Current 2, Electric Current 3`

```
Initial Condition id
  Kinetic Energy                                    Real
  Kinetic Energy Dissipation                        Real
  Pressure                                          Real
  Temperature                                       Real
  Velocity 1                                        Real
  Velocity 2                                        Real
  Velocity 3                                        Real
  Displacement 1                                    Real
  Displacement 2                                    Real
  Displacement 3                                    Real
  Magnetic Field 1                                  Real
```

```
  Magnetic Field 2                                 Real
  Magnetic Field 3                                 Real
End
```

When an external solver is added to the ELMER Solver a variable name must be given. This variable, and its components, are automatically added to the list of variables inside the solver and also their names may be used in the initial and boundary condition definitions.

F.4.9 The `Material` Section

The material section is used to give the material parameter values. The following material parameters may be set:

- `Density`
- `Enthalpy`: note that, when using the solidification modelling, an enthalpy-temperature curve must be given. The enthalpy is derived with respect to tempereraturе to get the value of the effective heat capacity.
- `Viscosity`: note that, when using the solidification modelling, a viscosity-temperature curve must be given. The viscosity must be set to high enough value in the temperature range for solid material to effectively set the velocity to zero.
- `Heat Capacity`
- `Heat Conductivity`
- `Heat Expansion Cofficient`
- `Reference Temperature`: This is the reference temperature for the Bussinesq model of temperature dependancy of density.
- `Poisson Ratio`
- `Youngs Modulus`
- `Magnetic Permeability`
- `Electric Conductivity`
- `Convection Velocity 1,2,3`: Convection velocity for the constant convection model.

- `Applied Magnetic Field 1,2,3`: An applied magnetic field may be given with these keywords.

- `Compressiblity Model`: this setting may be used to set the compressibilty model for the flow simulations. Currently the setting may be set to either `Incompressible` or `Perfect Gas Equation 1`. If the latter choice is selected the settings `Reference Pressure` and `Specific Heat Ratio` must also be given. The default value of this setting is `Incompressible`.

- `Reference Pressure`: with this keyword a reference level of pressure may be given. This setting applies only if the `Compressiblity Model` is set to the value `Perfect Gas Equation 1`.

- `Specific Heat Ratio`: the ratio of specfic heats (in constant pressure versus in constant volume) may be given with this keyword. This setting applies only if the `Compressiblity Model` is set to value `Perfect Gas Equation 1`. The default value of this setting is 5/3, which is the appropriate value for monoatomic ideal gas.

```
Material id
  Applied Magnetic Field 1     Real
  Applied Magnetic Field 2     Real
  Applied Magnetic Field 3     Real
  Convection Velocity 1        Real
  Convection Velocity 2        Real
  Convection Velocity 3        Real
  Compressibility Model        String [Incompressible]
                                      [Perfect Gas Equation 1]
  Density                      Real
  Electric Conductivity        Real
  Enthalpy                     Real
  Heat Capacity                Real
  Heat Conductivity            Real
  Heat Expansion Coefficient   Real
  Magnetic Permeability        Real
  Poisson Ratio                Real
  Reference Temperature        Real
  Reference Pressure           Real
  Specific Heat Ratio          Real
  Viscosity                    Real
  Youngs Modulus               Real
End
```

For the k-epsilon turbulence model the model parameters may also be given in the material section using the following keywords:

- `KESigmaK` (default 1.0)
- `KESigmaE` (1.3)
- `KEC1` (1.44)
- `KEC2` (1.92)
- `KECmu` (0.09)

F.4.10 The `Boundary` Section

This section is for user reference only and may be omitted entirely. In the boundary section the following keywords may be given

- `Body 1`: With this keyword the id of the first body which the boundary belongs is given.
- `Body 2`: With this keyword the id of the second body which the boundary belongs is given.
- `Name`: With this keyword a name may be given to boundary.

```
Boundary id
   Body 1              Integer
   Body 2              Integer
   Name                String
End
```

F.4.11 The `Boundary Condition` Section

The boundary condition section holds the parameter values for various boundary condition types. Dirichlet boundary conditions may be set for all the primary field variables: displacement (keywords `Displacement 1`, `Displacement 2`, `Displacement 3`), velocity (keywords `Velocity 1`, `Velocity 2`, and `Velocity 3`), temperature and pressure, and magnetic field (`Magnetic Field 1`, `Magnetic Field 2`, and `Magnetic Field 3`. The Dirichlet conditions for the vector variables may be given in normal-tangential coordinate system instead of the coordinate axis directed system using the keywords `Normal-Tangential`

`Velocity`, `Normal-Tangential Displacement`, and `Normal-Tangential Magnetic Field`.

For every variable added to the solver by introducing an external solver a corresponding set of Dirichlet conditions are automatically made available.

For the heat equation the heat flux boundary condition control variables are:

- `Heat Flux BC`: must be set to `true`, if heat flux boundary condition is present.
- `Heat Flux`: a user defined heat flux term.
- `Heat Transfer Coefficient, External Temperature`: a heat flux boundary condition of the type

  $$-k\frac{\partial T}{\partial n} = \alpha(T - T_{ext})$$

  is used. The variable α is the value given with the keyword `Heat Transfer Coefficient`, the variable T_{ext} is the value given with the keyword `External Temperature`.
- `Radiation, Emissivity`: with these keywords the radiation boundary condition is activated. With the keyword `Radiation` a character string is given, containing the type of the radiation model for this boundary, one of: `None`, `Idealized`, `Diffuse Gray`. Emissivity is the emissivity of the boundary surface. Note that, if using the diffuse gray radiation model, the file containing the Gebhardt factors must be given in the simulation section.
- `Radiation Target Body`: this flag may be used to set the direction of the outward pointing normal. This is used when computing viewfactors. A body identification number must be given. The default is that the normal points to less dense material or outward on outer boundaries.

For the Navier-Stokes equations, the following variables control the force boundary conditions:

- `Flow Force BC`: a logical value that must be set to `true`, if there is a force boundary condition for the Navier-Stokes equations.
- `Surface Tension Coefficient, Surface Tension Expansion Coefficient`: Giving these values triggers a tangetial stress

boundary condition to be used. If the keyword `Surface Tension Expansion Coefficient` is given, a linear dependancy of the surface tension coefficient on the temperature is assumed. Otherwise the value given with the `Surface Tension Coefficient` is assumed to hold the dependancy explicitely. Note that this boundary condition is the tangential derivative of the surface tension coefficient.

- `External Pressure`: a pressure boundary condition directed normal to the surface.
- `Pressure 1,2,3`: a pressure boundary condition vector.

The k-epsilon turbulence model also has its own set of boundary condition keywords (in addition to the Dirichlet settings):

- `Wall Law`: this logical flag activates the (Reichardts) law of the wall for the boundary specified.
- `Surface Roughness`: a real number measuring roughness of the surface, the default is 9.0.
- `Boundary Layer Thickness`: a real number giving the thickness of the laminar layer close to the surface. The element size neer the surface outside the laminar layer into the exponential layer should be close to this value.

A free surface is specified by giving the keyword `Free Surface` value `True`. The logical keyword `Free Moving` specifies whether the regeneration of mesh is free to move the nodes of a given boundary when remeshing after moving the free surface nodal points. The default is that the boundary nodes are fixed.

The force boundary condition for the stress equations is a force vector given with keywords `Force 1, Force 2, Force 3`.

```
Boundary Condition id
  Normal-Tangential Displacement            Logical
  Displacement 1                            Real
  Displacement 2                            Real
  Displacement 3                            Real
  Emissivity                                Real
  External Pressure                         Real
  External Temperature                      Real
```

```
  Flow Force BC                            Logical
  Force 1                                  Real
  Force 2                                  Real
  Force 3                                  Real
  Heat Flux                                Real
  Heat Flux BC                             Logical
  Heat Transfer Coefficient                Real
  Kinetic Energy                           Real
  Kinetic Energy Dissipation               Real
  Pressure 1                               Real
  Pressure 2                               Real
  Pressure 3                               Real
  Radiation                                String [None][Idealized]
                                                  [Diffuse gray]
  Radiation Target Body                    Integer
  Surface Tension Coefficient              Real
  Surface Tension Expansion Coefficient    Real
  Temperature                              Real
  Normal-Tangential Velocity               Logical
  Normal Target Body                       Integer
  Velocity 1                               Real
  Velocity 2                               Real
  Velocity 3                               Real
  Free Surface                             Logical
  Free Moving                              Logical
  Normal-Tangential Magnetic Field         Logical
  Magnetic Field 1                         Real
  Magnetic Field 2                         Real
  Magnetic Field 3                         Real
  Wall Law                                 Logical
  Surface Roughness                        Real
  Boundary Layer Thickness                 Real
End
```

F.5 Using the Advection-Diffusion Equation Solver

The advection-diffusion equation solver included in ELMER distribution has been implemented as an external solver. This implies, that the `Solver` section must include the keywords `Procedure`, `Variable`, and `Variable DOFs` when using the distributed advection-diffusion equation solver. In this case the procedure definition must be given as follows:

```
  Procedure
     File "AdvectionDiffusion" "AdvectionDiffusionSolver"
```

Variable name may be any user defined name and the `Variable DOFs` parameter must be set to unity.

Note that in any given simulation there may be several `Solver` sections using the advection-diffusion solver as long as the variable names they define differ.

The material parameters for the advection-diffusion equation solvers may be set using the keywords `Name Diffusivity`, where the `Name` is the user given name of the field variable. A volume source may be given on the same lines with the keywords `Name Source`. Dirichlet boundary conditions are set as usual using the name of the variable. A flux boundary condition may be given using the keywords `Name Flux`.

As an example, one might define in the `Solver` section

```
Solver n
   Equation
     String "Oxygen Concentration"

   Procedure
     File "AdvectionDiffusion" "AdvectionDiffusionSolver"

   Variable
     String Oxygen

   Variable DOFs
     Integer 1
...
End
```

in the `Equation` section

```
Equation n
   Oxygen Concentration
     Logical True
...
End
```

in the `Material` section

```
Material n
    Oxygen Diffusivity
      Real 1.0
...
End
```

in the `Body Force` section

```
Body Force n
    Oxygen Source
      Real 1.0
...
End
```

and finally in the `Boundary Condtion` section

```
Boundary Condition n
   Oxygen Flux
     Real 1.0
...
End
```

F.6 An Example of Using the Input File

In the following a sample solver input file, written by the ELMER Front program, will be described. The problem the solver input file applies is a steady-state coupled heat and flow simulation, where the coupling comes from the convection term in the heat equation side, and temperature dependancy of the surface tension coefficient (called Marangoni convection) in the flow solver side. The geometry used in this simple example is a unit square, so there are four boundaries in this model. The boundaries were assigned boundary conditions so that bottom, top, left and right boundaries had boundary conditions id:s 1,2,3 and 4 respectively (figure F.1).

Mathematically the equations to be solved are

$$\begin{aligned} -\nabla\cdot(2\mu\overline{\overline{\varepsilon}}) + \rho(\vec{u}\cdot\nabla)\vec{u} + \nabla p &= 0 \quad \text{in } \Omega \\ \nabla\cdot\vec{u} &= 0 \quad \text{in } \Omega \end{aligned}$$

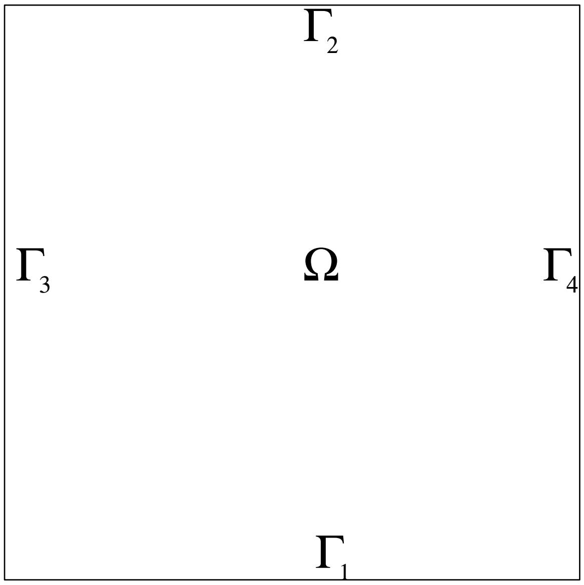


Figure F.1: *Geometry of the Marangoni convection example.*

$$
\begin{aligned}
\rho c_p \vec{u} \cdot \nabla T - \nabla \cdot (k \nabla T) &= 0 \quad \text{in } \Omega, \\
\vec{u} = 0, \quad T &= \frac{1}{2} - x \quad \text{on } \Gamma_1, \\
\sigma_\tau = \frac{\partial \gamma}{\partial \tau}, \quad \vec{u} \cdot \vec{n} = 0, \quad -k \frac{\partial T}{\partial n} &= 0 \quad \text{on } \Gamma_2, \\
\vec{u} = 0, \quad T &= \frac{1}{2} \quad \text{on } \Gamma_3, \\
\vec{u} = 0, \quad T &= -\frac{1}{2} \quad \text{on } \Gamma_4.
\end{aligned}
$$

The material parameters, in non-dimensional units, are as follows:

$$
\begin{aligned}
\rho &= 1.0, \\
\mu &= 1.0, \\
c_p &= 1.0, \\
k &= 1.37, \\
\gamma &= 548.0(1 - T).
\end{aligned}
$$

The Input File Header

First in the solver input file there is the header section. In the header section various dimensions of other sections are given. Also path to the mesh database is given here:

```
Header
  Mesh DB "Models" "Square"
  Bodies              1
  Materials           1
  Equations           1
  Solvers             2
  Body Forces         1
  Boundaries          4
  Boundary Conditions 4
End
```

Solver Control Data

Next we must give the solver the case control data

```
Simulation
  Coordinate System
    String Cartesian 2D

  Simulation Type
    String Steady State

  Steady State Max Iterations
    Integer 50

  Output Intervals
    Size 1
      Integer 1

  Output File
    File "Marangoni.dat"

  Post File
    File "Marangoni.ep"
End
```

The output file will contain the solved field variables for every iteration of the coupled system: temperature, velocity and pressure. The post processing file will also contain the mesh.

Body Definitions

This model has only one body which is assigned material and the equation set (to be defined later). There are no initial conditions or mass forces definition for the current case.

```
Body 1
  Equation
    Integer 1

  Material
    Integer 1
End
```

Equation Set Definitions

Next we define which equations to solve (there is only one set of equations in this example). The case where we would have more than one equation set is, for example, a case where temperature is solved in the whole model, but flow field is solved only in some parts of the model.

```
Equation 1
  Navier-Stokes
    Logical True

  Heat Equation
    Logical True

  Convection
    String Computed
End
```

Now we are ready to define solver parameters for the heat equation

```
Solver 1
  Equation
    String Heat Equation
```

```
  Stabilize
    Logical True

  Nonlinear System Max Iterations
    Integer 1

  Linear System Solver
    String Iterative

  Linear System Iterative Method
    String BiCGStab

  Linear System Preconditioning
    String ILU

  Linear System Max Iterations
    Integer 100

  Linear System Convergence Tolerance
    Real 1.0e-8
End
```

There is no nonlinearity present in the current case for the heat equation, so that we can set the number of nonlinear iterations to unity. The linear system is solved with an iterative method and system is precoditioned with an incomplete LU decomposition.

Solver Definitions

Define solver parameters for the Navier-Stokes equations

```
Solver 2
  Equation
    String Navier-Stokes

  Stabilize
    Logical True

  Nonlinear System Max Iterations
    Integer 10
```

```
  Nonlinar System Convergence Tolerance
    Real 1.0e-6

  Nonlinear System Newton After Iterations
    Integer 3

  Nonlinear System Newton After Tolerance
    Real 1.0e-2

  Nonlinear System Relaxation Factor
    Real 1.0

  Linear System Solver
    String Iterative

  Linear System Iterative Method
    String BiCGStab

  Linear System Preconditioning
    String ILU

  Linear System Max Iterations
    Integer 100

  Linear System Convergence Tolerance
    Real 1.0e-8

  Steady State Convergence Tolerance
    Real 1.0e-5
End
```

Material parameters

The material parameters are all constant in this example

```
Material 1
  Density
    Real 1.00

  Viscosity
    Real 1.00
```

```
  Heat Capacity
    Real 1.00

  Heat Conductivity
    Real 1.37
End
```

Boundaries and Boundary Conditions

Boundaries may be given in the input file section with the `Boundary n` keyword. These are for reference only.

```
  Boundary 1
    Name
     String "Bottom"
  End

  Boundary 2
    Name
     String "Top"
  End

  Boundary 3
    Name
     String "Left"
  End

  Boundary 4
    Name
     String "Right"
  End
```

The mapping from the boundaries to boundary conditions is given within the `Boundary Condition` section with the keyword `Target Boundaries`. If the boundary condition matches to several boundaries a list of boundary numbers may be given with this keyword. In this example each boundary has its own boundary condition, so the mapping is the trivial one.

The bottom boundary has a Dirichlet boundary condition for temperature, which varies linearily from 0.5 degrees on the left side to -0.5 degrees on the right side. This boundary condition has been given below as data points describing (generally) a piecewise linear spline. There are other methods to implement varying boundary conditions to be described later.

```
!****** Bottom BC: Temperature T(x) from T(0)= 0.5
!                 to T(1)=-0.5,Velocity=0
Boundary Condition 1
  Target Boundaries
    Size 1
      Integer 1

  Temperature
    Variable Coordinate 1
      Real
      ! x  T
        0  0.5
        1 -0.5
      End

  Velocity 1
    Real 0.0

  Velocity 2
    Real 0.0
End
```

The top boundary has a stress boundary condition. Note that the force boundary conditions must be flagged with the 'Flow Force BC' keyword in order to be recognized by the solver. The tangetial component of the force vector is the tangential derivative of the temperature dependent surface tension coefficient. The temperature dependancy of this parameter is assumed to be linear in this case. The normal component of the stress condition may be omitted as the curvature of the boundary is zero, and the level of the atmospheric pressure is immaterial and would only move the zero level of the pressure field. Instead the condition $u \cdot n = 0$ (no flow across the surface) is enforced.

```
!****** Top BC: Tangetial Stress, normal velocity zero
Boundary Condition 2
  Target Boundaries
    Size 1
      Integer 2
```

```
  Flow Force BC
    Logical True

  Surface Tension Coefficient
    Variable Temperature
      Real
       !  T    \gamma  (only the slope matters...)
        -1.0  548.0
         1.0 -548.0
      End

  Normal-Tangential Velocity
    Logical True

  Velocity 1
    Real 0.0
End
```

Right and left boundaries have Dirichlet boundary conditions for both the temperature and the velocity components.

```
!****** Right side BC: Temperature=-0.5, Velocity=0
Boundary Condition 3
  Target Boundaries
    Size 1
      Integer 3

  Temperature
    Real -0.5

  Velocity 1
    Real 0.0

  Velocity 2
    Real 0.0
End

!****** Left side BC: Temperature=0.5, Velocity=0
Boundary Condition 4
  Target Boundaries
    Size 1
      Integer 4
```

```
  Temperature
    Real 0.5

  Velocity 1
    Real 0.0

  Velocity 2
    Real 0.0
End
```

Results

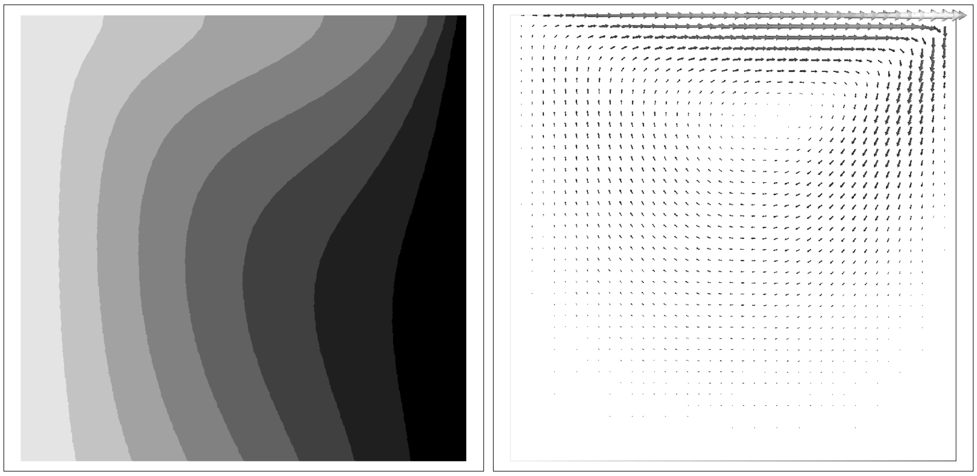


Figure F.2: *Results of the Marangoni convection example. Isocontours of the temperature field and the velocity vectors.*

The resulting temperature and velocity fields are shown in figure F.2.

Appendix G

ELMER Solver Output File Format

The ELMER solver output file format is very simple. The file consist of a header and the solution data. The solver output format is also used in the restart files.

G.1 The Header

First row of the file is the string

```
Degrees of freedom:
```

followed by names of the variables, which have been written to this file, along with the name of the equation solver they have been computed by. For example

```
 temperature :heat equation
 pressure :navier-stokes
 velocity 2 :navier-stokes
 velocity 1 :navier-stokes
```

The names must each be on their own rows. The default set variable names used by the solver is

- `Displacement 1`

- `Displacement 2`
- `Displacement 3`
- `Temperature`
- `Pressure`
- `Velocity 1`
- `Velocity 2`
- `Velocity 3`
- `Coordinate 1`
- `Coordinate 2`
- `Coordinate 3`
- `Magnetic Field 1`
- `Magnetic Field 2`
- `Magnetic Field 3`
- `Electric Current 1`
- `Electric Current 2`
- `Electric Current 3`
- `Velocity 1`
- `Velocity 2`
- `Velocity 3`

Following the names is the total number of degrees of freedom, for example:

```
 Total DOFs:           4
```

This ends the header section.

G.2 The Output Data

For every timestep or steady state iteration stored in output file, there is first a header row giving count of the timestep in this file, the timestep number and the current simulation time, for example

```
Time:     1    1      0.000000000000
```

Following is, for every variable stored in the file, first the name of the variable, followed by rows which contain the node number of the value, reordering of the nodes for bandwidth optimization, and the value itself:

```
temperature
           1   1   300.000000000000
           2   2   300.000000000000
           3   3   300.000000000000
           4   4   300.000000000000
           5   5   300.000000000000
           6   6   300.000000000000
               ....
```

Note that the reordering index also includes the information whether the node belongs to a body for which this specific variable has been computed. If the variable is not available for a given node, the reordering index has been set to zero.

Appendix H

ELMER Post File Format

The Elmerpost input file format is as follows

```
nn ne nf nt scalar: name vector: name ...
x0 y0 z0
...        ! node coordinates (nn) rows (x,y,z)
xn yn zn
group-name element-type i0 ... in
...        ! element decriptions (ne) rows
group-name element-type i0 ... in
#time 1 timestep1 time1
vx vy vz p ...
...        ! nn rows
vx vy vz p ...
#time 2 timestep2 time2
vx vy vz p
...        ! nn rows
vx vy vz p ...
 ....
#time n timestepn timen
vx vy vz p ...
...        ! nn rows
vx vy vz p ...
```

There is also another level of element grouping that can be employed as follows

```
#group group1
  element-definitions
```

```
            ...
#group group2
  element-definitions
            ...
#endgroup group2
  element-definitions
            ...
#endgroup group1
```

The number of element groups is not restricted in any way.

H.1 The Header

The following information must be given in the header

- `nn`: number of the mesh nodal points
- `ne`: the total number of the elements in the mesh including boundary elements
- `nf`: gives the total number of degrees of freedom (add three for any vector quantity and one for a scalar quantity).
- `nt`: number of timesteps stored in the file
- `scalar: name, vector: name`: list of variable types and names respectively.

H.2 The Mesh

First nodal coordinates are given, each triplet on its own row. There must be three coordinates even if the model would be 2D.

Element description:

- `group-name`: name of the element group (material, body or some such)
- `element-type`: numeric code giving the element type, refer to appendix I.

- The numbers `i0-in` in the element descriptions are indices to the nodal coordinate array at the beginning of the file. First coordinate in the file has the index zero.

H.3 The Solution Data

For each timestep the following information is written:

- `#time n t time`: `n` is the order number of the timestep written in this file, `t` is the simulation timestep number, and `time` is the current simulation time.
- Next the nodal values of the degrees of freedom are given, one nodal point at a time, and in the order given with the keywords `scalar:` and `vector:` in the header.

H.4 An example file

Following is a very simple mesh in ELMER Post format. There is only on element, three nodal points, one variable value given at the nodal points on a single time step.

```
3 1 1 1 scalar: Temperature
0 0 0
1 0 0
0 1 0
#group all
body1 303 0 1 2
#endgroup all
#time 1 1 0
1
2
3
```

Refer to Elmerpost documentation for more information.

Appendix I

ELMER Solver Element Types

The default element types of the ELMER Solver are the linear and quadratic elements of the Lagrange element family

- linear (element type code 202) and quadratic (203) 1D elements.
- linear (element type code 303) and quadratic (306) triangles (3 and 6 nodal points, Figure I.1),

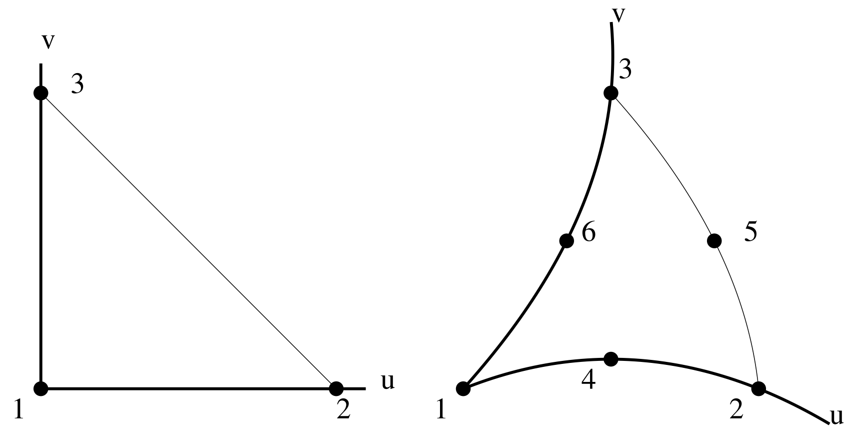


Figure I.1: *The 3 (303) and 6 (306) node triangular elements.*

- bilinear (404) and quadratic (408,409) quadrilaterals (4,8, and 9 nodal points, Figure I.2),
- linear (504) and quadratic (510) tetrahedrons (4 and 10 nodal points, Figure I.3),

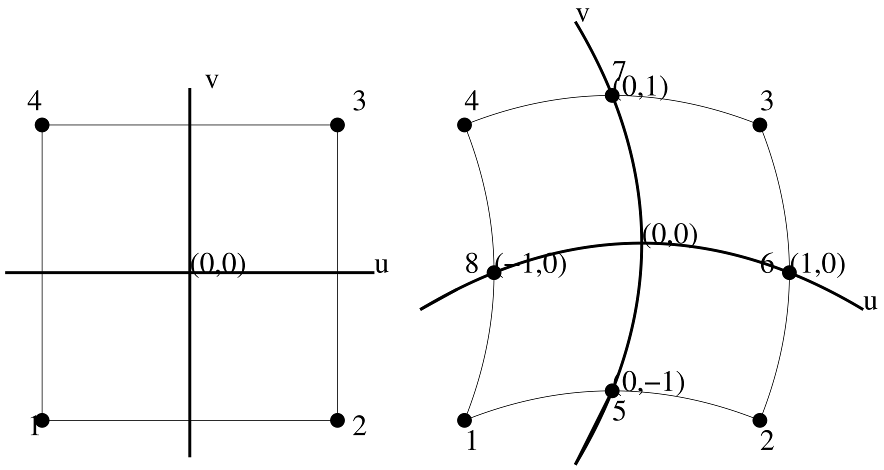


Figure I.2: *The 4 (404), and 8 (408) node quadrilateral elements.*

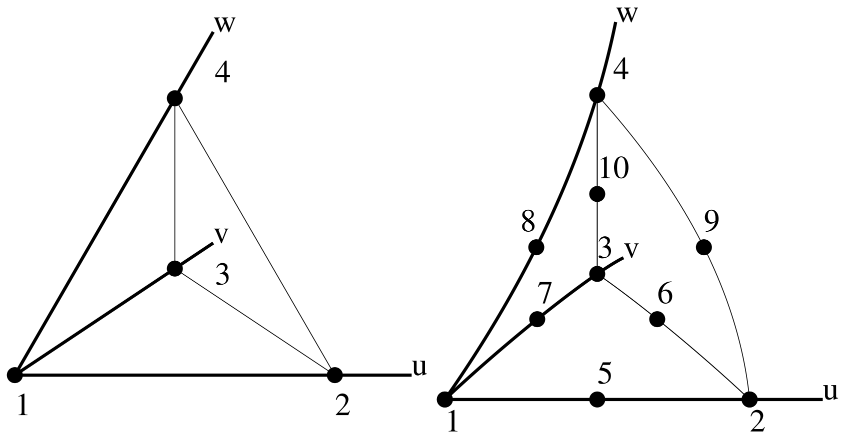


Figure I.3: *The 4 (504) and 10 (510) node tetrahedron elements.*

- trilinear (808) and quadratic (820,827) bricks (8, 20, and 27 nodal points, Figure I.4).

Further element types belonging to these basic topologies may be added to the program by editing element type definition file without touching the solver code nor compiling or linking the program. How to create a mesh using these element types is a separate question.

The definition file is read in when the solver code is initialized and a basis function matrix is formed for every element type defined. The rest of the FEM code (computing local and global derivates, integration of elements) is then basically independent of the element type.

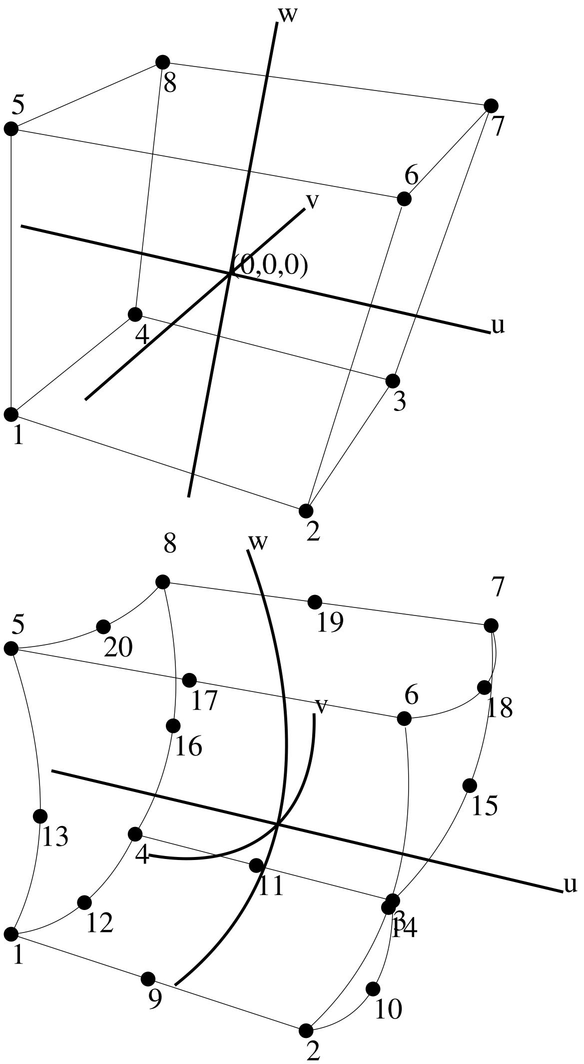


Figure I.4: *The 8 (808) and 20 (820) node brick elements.*

Index

A
advection-diffusion equation, 88, 179
axial symmetry, 86

B
body force, 89
boundary conditions, 89

C
continuity equation, 88
coordinate systems, 86
cylindrical symmetry, 86

E
elastic stress equations, 88
element types, 88, 197
`ELMERSOLVER_STARTINFO`, 85
energy equation, 87
equation solver, 90

F
Front, 8
functions
    user-defined, 90

G
Gebhardt factors, 87

H
heat transfer, 87

I
initial conditions, 89

L
linear solvers, 88
    direct methods, 88
    iterative methods, 88
linearization, 88
    Newton, 88
    Picard, 88

M
magnetohydrodynamics, 86
Marangoni convection, 181
material properties, 89
Mathbackground, 129
mesh file format, 156
`mesh.boundary`, 158
`mesh.elements`, 157
`mesh.header`, 156
`mesh.nodes`, 157
momentum equations, 88

N
Navier-Stokes equations, 88
    compressible, 88
    incompressible, 88

P
post, 94

R
radiation, 87
    diffuse-gray, 87
        Gebhardt factors, 87
        view factors, 87
    idealized, 87
restart file format, 191

S
`.sif` file, 85, 159
simulation body, 89
Solver, 85
    utilities, 91
Solver input, 159
    an example, 181
    array variables, 159

  - body defintions, 170
  - body forces, 172
  - boundaries, 176
  - boundary conditions, 176
  - equation sets, 171
  - header, 162
  - initial conditions, 173
  - materials, 174
  - simulation data, 164
  - solver defintions, 166
  - user functions, 161
  - variable parameters, 160
- Solver output format, 191
- structural analysis, 88

T

- time integration, 86

V

- view factors, 87