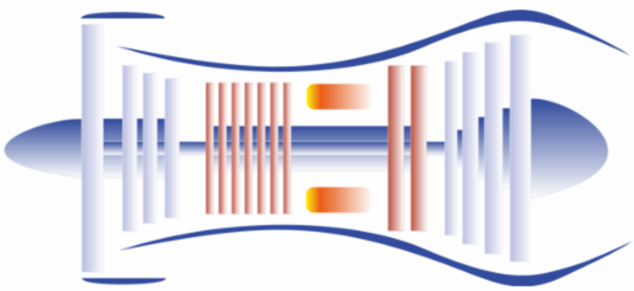
PROOSIS

Propulsion
Object
Oriented
Simulation
Software

EMPRESARIOS AGRUPADOS

Introduction to Gas Turbine Modelling with PROOSIS



A. Alexiou
T. Tsalavoutas

Laboratory of Thermal Turbomachines
National Technical University of Athens

EMPRESARIOS AGRUPADOS

Introduction to

Gas Turbine Modelling

with PROOSIS

Introduction to Gas Turbine Modelling with PROOSIS

First Edition

Alexis Alexiou

BEng, PhD, MIMechE, CEng (U.K.)
Senior Researcher
Laboratory of Thermal Turbomachines
National Technical University of Athens
a.alexiou@ltt.ntua.gr

Tasos Tsalavoutas

Mechanical Engineer, PhD
Senior Researcher
Laboratory of Thermal Turbomachines
National Technical University of Athens
ttsal@ltt.ntua.gr

Introduction to Gas Turbine Modelling with PROOSIS

First Edition

Alexis Alexiou, Tasos Tsalavoutas (LTT/NTUA)

Laboratory of Thermal Turbomachines (LTT)
National Technical University of Athens (NTUA)
Building 'O', 9 Iroon Polytechniou, Polytechnioupoli Zografou
15773, Athens, GREECE
www.ltt.ntua.gr
Phone: +302107722343 / +302107721467
Fax: +302107721658

Empresarios Agrupados Internacional (EAI) S.A.
Magallanes, 3 Madrid
28015 Spain
info@ecosimpro.com
www.ecosimpro.com
www.proosis.com
Phone: +34 913098142
Fax: +34 915912655

Revision: EcosimPro/PROOSIS Team (EAI)
Typesetting: by the authors
Cover design: EcosimPro/PROOSIS Team (EAI)

Preface

Nowadays the discipline of modelling and simulation has become a fundamental tool for the design of new products in the world of engineering. In the field of aerospace propulsion, engineers are currently working on new architectures for gas turbines that consume less, pollute less (a lower production of harmful gases such as CO_2 and NOx) and reduce current noise levels. To achieve these objectives advanced modelling tools are required to simulate these phenomena quickly and efficiently in order to optimize new designs. Simulation allows reproducing multiple engine configurations and can help in selecting the optimal ones. In the end, it can significantly reduce design times and the number of very expensive physical tests that are needed.

PROOSIS is a tool that allows the modelling of all these propulsion systems in 1D and assists greatly in the production of better engine designs. It is a simulation tool with several different user levels: from the basic user who creates component libraries to the end-user who simply runs closed models from a deck or from Excel, through users who graphically create new engine configurations, or users who create multiple experiments related to design, off-design, transients, parametric, optimization, etc. PROOSIS also has an object-oriented non-causal modelling language that is very easy to learn and that allows the modelling of algebraic-differential equation systems and discrete events.

PROOSIS allows the multi-disciplinary modelling of systems, i.e., it is possible to model both the propulsion system as well as any auxiliary system, such as control, fuel supply, secondary air system, etc. This means that a single tool can be used for the integral design of the entire propulsion system.

PROOSIS is based on the EcosimPro tool originally developed by Empresarios Agrupados Internacional for the European Space Agency. PROOSIS includes additional capacities such as: handling and viewing of performance maps, tools for the multi-point design of engines, handling of inequations, generation of decks in accordance with international standards, a complete library aerospace gas turbine components, etc.

We believe that this introductory book to the modelling of gas turbines with PROOSIS, written by Alex and Tasos of the Thermal Turbomachines Laboratory at the National Technical University of Athens, is a magnificent guide both for new users of PROOSIS and for experienced users. They have done an excellent job in describing the most important capabilities of the tool, using examples that provide a clear understanding of all the concepts covered in the book.

All the examples given in the book can be reproduced by the reader using the PROOSIS libraries included in the software installation. We believe that the examples provided cover the most typical cases both on design and off-design levels for a wide range of engine configurations. For all these reasons, the book will be very useful for learning to use the tool both on an academic level and on a company professional level.

We are convinced that it will be a great asset of all PROOSIS users and that it will enable them to use PROOSIS more efficiently.

Pedro Cobas Herrero, Head of PROOSIS & EcosimPro Development Team

Empresarios Agrupados Internacional S.A.

July 2011

Page intentionally left blank

Contents

Chapter 1

The PROOSIS Simulation Environment

Page intentionally left blank

1 The PROOSIS Simulation Environment

1.1 Introduction – A Quick Overview of PROOSIS

What is PROOSIS?

PROOSIS[1] [1] is a stand-alone, flexible and extensible object-oriented simulation environment for gas turbine engine performance modelling developed within the integrated European project **VIVACE** [2] by a consortium of European universities, research institutes and corporate companies. It is based on **EcosimPro** [3], a simulation tool developed by **Empresarios Agrupados Internacional S. A.**, for modelling any physical process expressed in terms of differential-algebraic equations or ordinary-differential equations and discrete events.

What does it do?

PROOSIS can perform all kinds of gas turbine engine simulations as well as generic system simulations (e.g. control, thermal, hydraulic, mechanical, etc.). It is capable of steady state and transient simulations as well as customer deck generation. Different calculation types can be performed (single or multi-point design, off-design, test analysis, sensitivity, parametric and optimisation studies, mission analysis, diagnostics, etc). It is also capable of performing multi-fidelity, multi-disciplinary and distributed simulations [4, 5]. These are greatly facilitated by its open architecture, which allows it to connect to external tools (commercial or company specific, e.g., CFD, FEA, life, etc.). These features make PROOSIS a useful tool for all phases of the engine life cycle, from preliminary and detailed design to post-certification and in-service support, and allow it to serve as a common framework in multi-partner collaborative engine projects providing common standards and methodologies.

How does it work?

PROOSIS uses a high-level object-oriented language (**EL**) for modelling engine systems. EL offers all the benefits of this type of programming, among these inheritance, polymorphism and encapsulation. Inheritance is the ability to create new components based on existing ones. Polymorphism is the capability of having methods and properties in different components that can be used interchangeably even though they have the same name. Encapsulation, meanwhile, offers the ability to conceal the actual implementation of properties and methods in the component.

Component is the most important concept in EL; it contains a mathematical description of the corresponding real-world component (e.g., compressor, turbine, burner, nozzle, etc.). Components communicate with each other through their **Ports**. Ports define the set of variables to be interchanged between connected components (e.g., mass flow rate, pressure, and temperature in a Fluid Port or rotational speed, torque and inertia in a Mechanical Port, etc.).

Components and ports are stored in reusable **Libraries.** Gas turbine engine components and ports are stored in library **TURBO**, which will be discussed in more detail in the following chapter. The creation of these libraries requires an expert user (**Level 1**). Such a user is an expert on the physical and mathematical formulation of component models and has deep knowledge of the PROOSIS modelling language EL in order to express the equations, interact with files, impose variables such as boundary, algebraic, data, etc. The main elements of EL are described in Appendix A using examples related to TURBO library.

[1] **PR**opulsion **O**bject **O**riented **SI**mulation **S**oftware.

Using existing libraries, a user (**Level 2**) can construct a model graphically by 'dragging-and-dropping' the required component icons from one or more library palettes to a **schematic** window, connecting the components through the appropriate ports and editing their attributes. A model, for example, could be a single component, a sub-assembly or a complete engine. Chapters 2 and 3 give examples of how to create new models graphically at the TURBO library component and engine levels, respectively. Components from different libraries can be combined in constructing a model as long as connected components share the same communication interface (e.g., ports).

The user must be aware of the mathematical implications of connecting components and must be able to create the associated mathematical model. This is called **Partition** in PROOSIS and it is set with the help of wizards. Built-in mathematical algorithms process the equations symbolically, resolve high index problems, solve algebraic loops, suggest boundary conditions and finally sort the equations for efficient calculation since PROOSIS allows for non-causal modelling; the order and form of equations does not matter. The procedure for generating partitions is presented in Appendix B while appropriate partitions for simulating individual TURBO components and typical engine configurations are described in the following two chapters.

For a given Partition, a user (**Level 3**) can define different simulation cases (**Experiments**). Using wizards or the object-oriented language EL, one can initialise variables, set the values of boundary condition variables and component data, run single and multiple steady state simulations, integrate the model over time (transient operation) and generate reports (write results to file or screen). With the help of internal (EL) or external (C, C++, FORTRAN) **Functions** it is possible to create very complicated simulations like those described in Chapter 4 (e.g., multi-point design, optimisation, test analysis, etc.).

Experiments can run either in batch mode or graphically using the Experiments **Monitor**. A **log** file with user-defined level of information is produced for debugging. A **postprocess** file can also be generated for an Experiment so that its results can be viewed in a postprocess Monitor. Experiment creation and simulation is described in Appendix C while many examples are presented in the subsequent chapters (2-4).

From a working experiment and through a wizard, different types of **Decks** can be generated for delivering a model to a customer (**Level 4** user). No knowledge of PROOSIS is required to use a deck. PROOSIS can generate decks with or without a graphical user interface, as executable programs. In either case a dynamic link library (dll) file is produced that implements subsets of SAE ARP4868 [6] and SAE AS4191 [7] API standards that can be accessed through a main file. It is also possible to run PROOSIS models from applications like EXCEL or Matlab. Deck generation and its use from other applications are described in Chapter 5.

The steps from component definition to deck generation are shown schematically in Figure 1.1, where the relevance of each step to the different levels of users in PROOSIS is also identified.

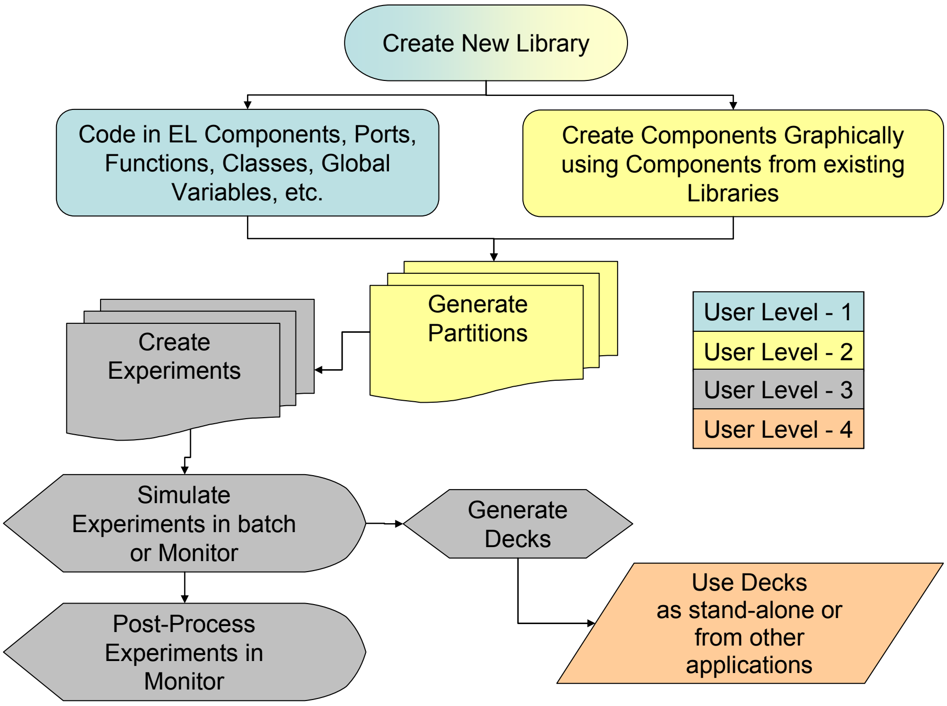


Figure 1.1: PROOSIS model creation steps

<u>Who should read this manual?</u>

This manual is mainly relevant to Level 2 and Level 3 users as it demonstrates how to perform a wide range of calculations at the component, sub-assembly or engine levels, using the existing TURBO library of engine components and the built-in wizards for generating partitions, defining experiments and running simulations. More advanced calculations may require experiments to be defined programmatically using EL. The structure of such experiments is described at the end of Appendix C while detailed examples are included in all the chapters as well as in the relevant libraries accompanying this manual. Level 1 users wishing to modify existing TURBO library components or add new ones should refer to Appendix A where the main elements of EL are presented using examples from the TURBO library. Finally, useful information for generating and using decks (Level 4 users) is included in the last chapter.

For more information on EL, graphical modelling, partitions, experiments and decks the reader is referred to the PROOSIS User and EL manuals [8, 9]. These can be accessed through the PROOSIS Help Menu or by pressing F1 or F2, respectively. The construction of libraries and the mathematical handling of equations are also discussed in the book by Vázquez et al. [10] using specific modelling and simulation examples developed in EcosimPro.

The following sections give a short overview of the PROOSIS graphical user interface and describe the steps for creating and simulating an engine model using TURBO library components.

1.2 PROOSIS Graphical User Interface

PROOSIS has an advanced graphical user interface that can be easily configured according to the modelling task performed by switching between the three available views: Code, Schematic and Simulation.
The **Code View** is relevant to Level 1 users and allows them to manipulate EL files, partitions and experiments. Level-2 users work in **Schematic View** to create components using symbols from existing libraries as well as create partitions and experiments. Finally, the **Simulation View** is the starting point for Level 3 users in order to create experiments on already existing models. In all three views, users can simulate their experiments either in batch or in the **Simulation Monitor** and generate the corresponding decks. Figure 1.2 shows a screen shot of the PROOSIS window in Schematic View.

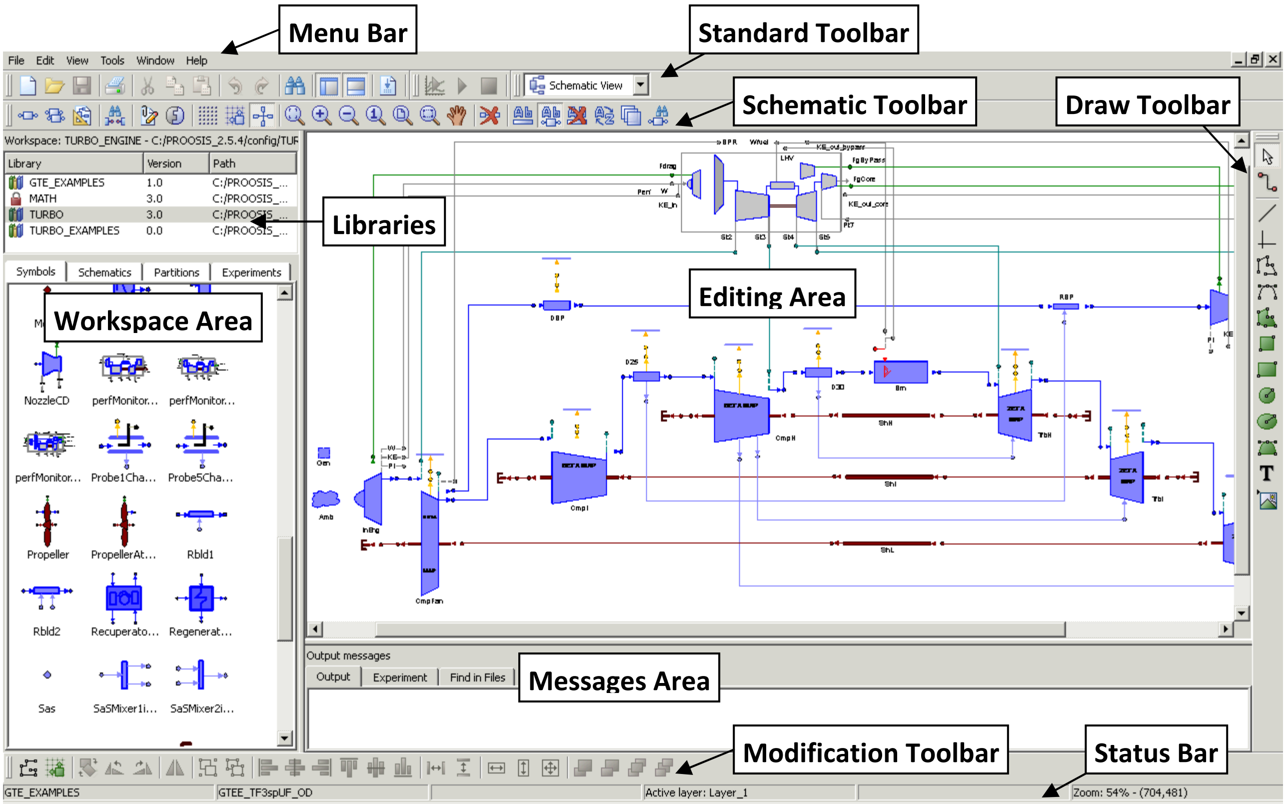


Figure 1.2: PROOSIS Graphical User Interface in Schematic View

The interface consists of a menu bar at the top (File, Edit, View, Tools, Window and Help menus) followed below by the **Standard** toolbar. There are three more toolbars: the **Schematic** toolbar underneath the Standard one, the **Draw** toolbar on the right-hand side of the screen and the **Modification** toolbar at the bottom. The toolbars can be moved around if needed. A description of what each toolbar button does is displayed if the mouse pointer is placed over it. The Schematic, Draw and Modification toolbars are specific to **Schematic View**. The Standard toolbar is available in all three views. Users can switch between the three views from the Standard Toolbar or the View Menu.

In the centre of the screen is the **Editing Area** where the user can edit symbols, schematic diagrams, EL files and experiments depending on the active view. Below it, there is a **Messages** area where the messages appearing relate to component compilation (Output tab), simulation (Experiment tab) or file search results (Find in Files), depending on which tab is selected.

At the very bottom of the screen there is a **Status** bar giving information depending on current user action (active library name, zoom level, cursor position, line/column number, etc.).

The left-hand side of the screen is the **Workspace Area**. The **Workspace** (TURBO_ENGINE in Figure 1.2) contains the **Libraries** the user works with. In general, a library is a collection of items (components, ports, functions, classes, etc.) relevant to a particular simulation discipline (e.g. thermal, control, hydraulic, gas turbines, etc.). The **Libraries Area** is on the upper left-hand side while the lower left-hand side contains information (files, items, symbols, schematics, partitions, experiments) for the selected library (Figure 1.2 shows the TURBO library symbols). The type of information displayed in this part of the screen also depends on the active View.

The Workspace Area, Messages Area and the Status Bar can be turned on and off from the **View** menu. The Workspace Area and Messages Area can also be turned on/off from the Standard Toolbar. In all three views, users can manipulate Workspaces from the File Menu (new, open, save, etc) as well as create new libraries or open existing ones. Right clicking on a library brings up a menu that allows various operations to be performed on it (compile, close, delete, etc.).

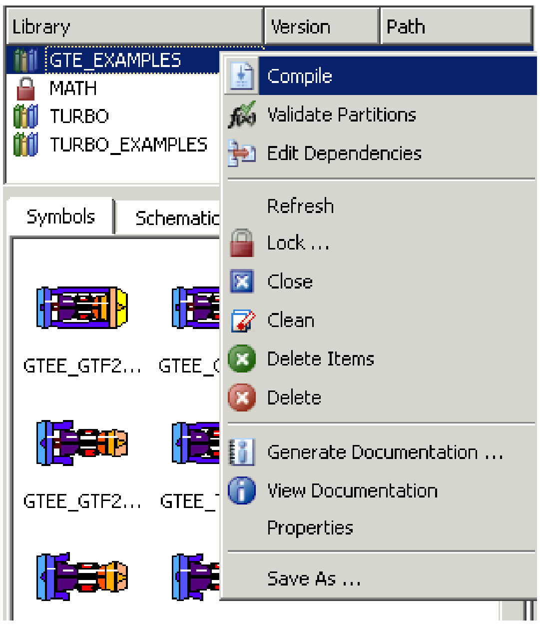


Figure 1.3: Library Menu

The relevant workspace for gas turbine performance simulation purposes is the TURBO_ENGINE one (to open it, press Ctrl+K and select file TURBO_ENGINE.wsp.xml). This includes four libraries: **MATH**, **TURBO**, **TURBO_EXAMPLES** and **GTE_EXAMPLES** (to open a library press Ctrl+L and select the corresponding .elb file, e.g., TURBO.elb). The TURBO library comprises a collection of gas turbine engine components and functions including fluid and thermodynamic functions. Examples of TURBO library component simulation are included in TURBO_EXAMPLES (presented in Chapter 2). The GTE_EXAMPLES library contains different engine performance models using TURBO library components and a wide range of simulation cases (see Chapters 3 and 4). These libraries use various items (global constants, functions, etc.) from the MATH library. Existing components can be modified or new ones can be added in these libraries to allow a particular model to be created while other libraries can also be included in the workspace to enable more detailed simulations (control, secondary air system, etc.).

The following section presents the procedure for creating and simulating an engine model.

1.3 Model Creation & Simulation Procedure

The main steps for creating a model (which can be a single component, a subassembly or an entire engine) and running a graphical simulation are the following (see also Figure 1.1):

I. Build the required model schematic using appropriate TURBO library symbols. Symbols from other libraries can also be included if necessary (e.g. CONTROL)
II. Generate a suitable partition (mathematical model)
III. Define one or more experiments (using wizards or EL)
IV. Customise the monitor and simulate the experiment

The detailed actions that must be performed within each one of these steps will now be described. It is assumed that the schematic view is active during this procedure. The turbojet model included in the GTE_EXAMPLES library (GTEE_TJ_OD) is used as an example. The schematic diagram, partition and various experiments for this configuration are described in Section 3.4.2.

I. Build model schematic

1. Create a new schematic from the File menu (File -> New -> Schematic …) or using the relevant button in the standard toolbar.

2. A dialogue appears for specifying the library this new schematic will be added to e.g. GTE_EXAMPLES, and the name of the schematic.

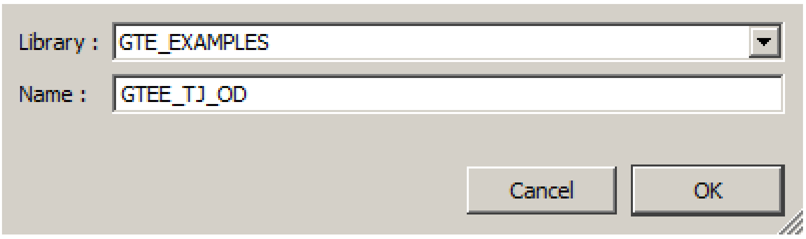


Figure 1.4: Create New Schematic

3. A blank window appears in the editing area. Drag-and-drop the appropriate symbols from the workspace area (e.g. TURBO library -> Symbols tab) to the schematic window in the editing area (see Figure 1.2).

4. Connect the components through their ports using the relevant button from the draw toolbar or press shift, place the mouse pointer near a port and left-click when the anchor symbol appears.

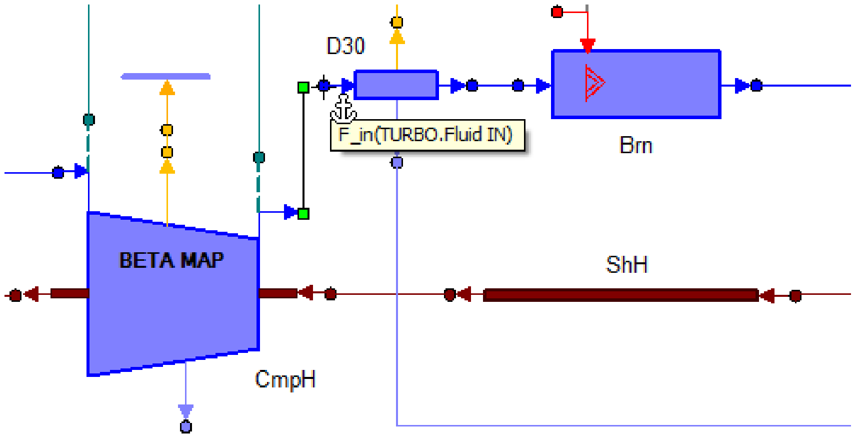
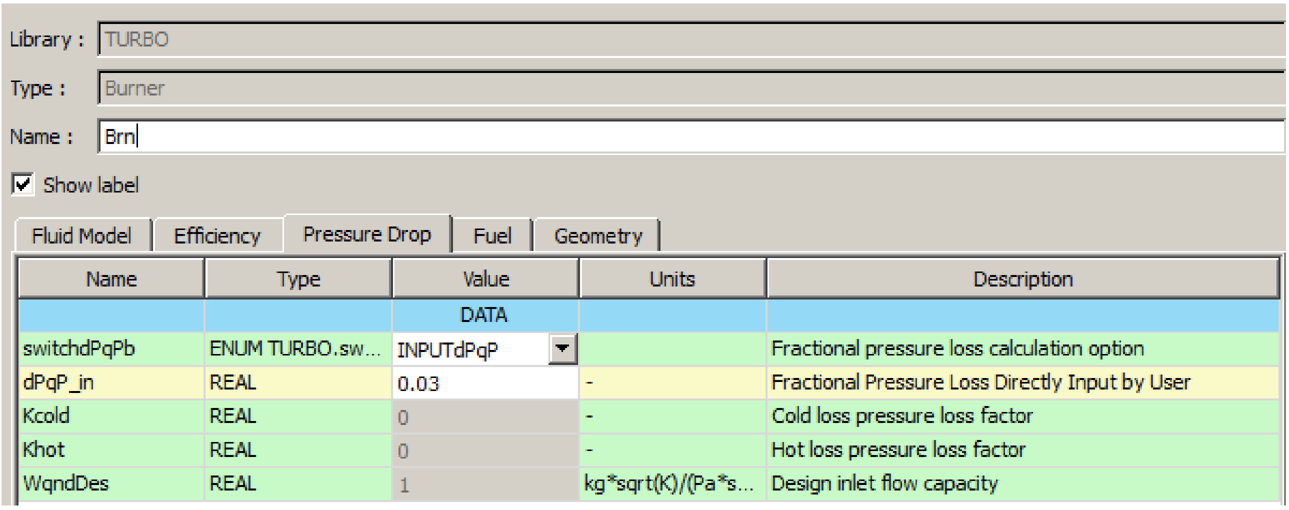


Figure 1.5: Connecting components through their ports

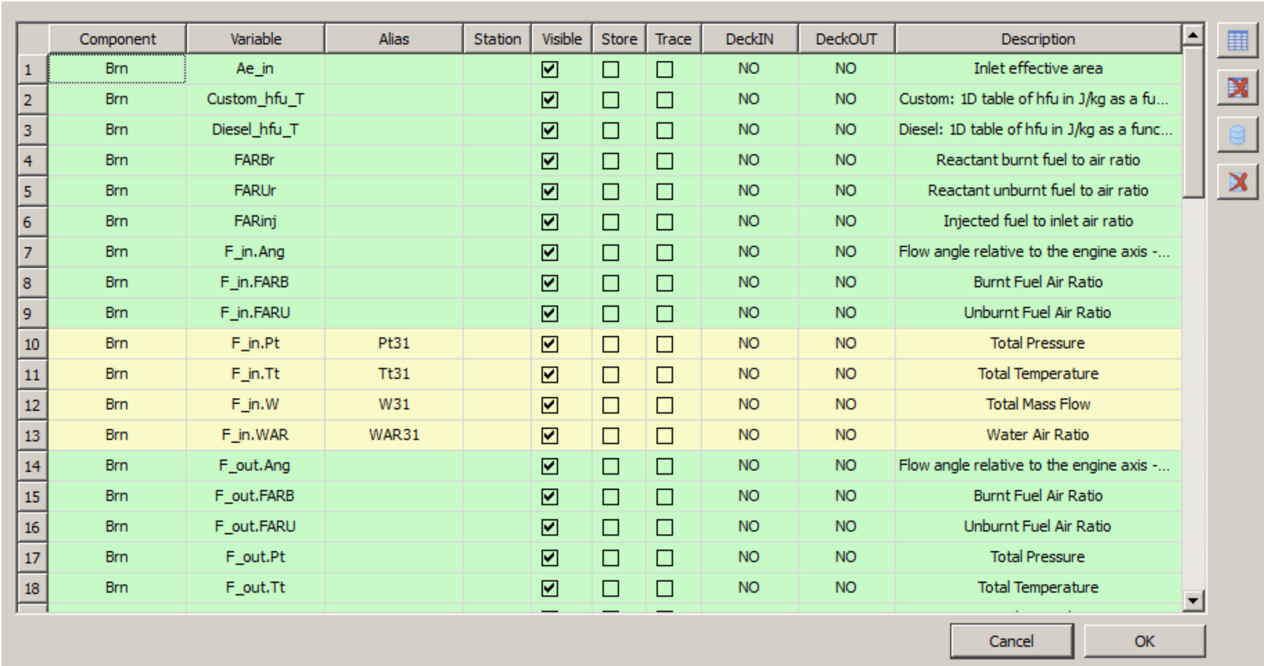
5. Right-click on a component and select **Edit Attributes** or left double-click on it to modify (if required) the default values of any of its attributes (such as bleed flows, efficiencies, scalars, pressure losses, etc). These values will be used in the final mathematical model of the component.


Library : TURBO
Type : Burner
Name : Brn
Show label
Fluid Model | Efficiency | Pressure Drop | Fuel | Geometry

Name	Type	Value	Units	Description
		DATA		
switchdPqPb	ENUM TURBO.sw...	INPUTdPqP		Fractional pressure loss calculation option
dPqP_in	REAL	0.03	-	Fractional Pressure Loss Directly Input by User
Kcold	REAL	0	-	Cold loss pressure loss factor
Khot	REAL	0	-	Hot loss pressure loss factor
WqndDes	REAL	1	kg*sqrt(K)/(Pa*s...	Design inlet flow capacity

Figure 1.6: Component Attributes Editor

6. Right-click on a component and select **Edit Properties** to modify the properties of its variables or right-click in an empty part of the schematic to modify the properties of all the model variables and data, including global ones. For each variable, it is possible to define an **Alias** that can then be used instead of the original variable name in partitions and experiments. A **Station** can also be associated with each variable. Both Station and Alias are empty by default. This step is not compulsory.



	Component	Variable	Alias	Station	Visible	Store	Trace	DeckIN	DeckOUT	Description
1	Brn	Ae_in			☑	☐	☐	NO	NO	Inlet effective area
2	Brn	Custom_hfu_T			☑	☐	☐	NO	NO	Custom: 1D table of hfu in J/kg as a fu...
3	Brn	Diesel_hfu_T			☑	☐	☐	NO	NO	Diesel: 1D table of hfu in J/kg as a func...
4	Brn	FARBr			☑	☐	☐	NO	NO	Reactant burnt fuel to air ratio
5	Brn	FARUr			☑	☐	☐	NO	NO	Reactant unburnt fuel to air ratio
6	Brn	FARinj			☑	☐	☐	NO	NO	Injected fuel to inlet air ratio
7	Brn	F_in.Ang			☑	☐	☐	NO	NO	Flow angle relative to the engine axis -...
8	Brn	F_in.FARB			☑	☐	☐	NO	NO	Burnt Fuel Air Ratio
9	Brn	F_in.FARU			☑	☐	☐	NO	NO	Unburnt Fuel Air Ratio
10	Brn	F_in.Pt	Pt31		☑	☐	☐	NO	NO	Total Pressure
11	Brn	F_in.Tt	Tt31		☑	☐	☐	NO	NO	Total Temperature
12	Brn	F_in.W	W31		☑	☐	☐	NO	NO	Total Mass Flow
13	Brn	F_in.WAR	WAR31		☑	☐	☐	NO	NO	Water Air Ratio
14	Brn	F_out.Ang			☑	☐	☐	NO	NO	Flow angle relative to the engine axis -...
15	Brn	F_out.FARB			☑	☐	☐	NO	NO	Burnt Fuel Air Ratio
16	Brn	F_out.FARU			☑	☐	☐	NO	NO	Unburnt Fuel Air Ratio
17	Brn	F_out.Pt			☑	☐	☐	NO	NO	Total Pressure
18	Brn	F_out.Tt			☑	☐	☐	NO	NO	Total Temperature

Figure 1.7: Component Properties Editor

7. Finally, compile the schematic by pressing F7 or the corresponding button in the standard toolbar. It is now possible to create a partition for this component. The default symbol (a rectangle) will be created in the **Symbols** tab for the new component.

II. Generate Partition

8. In the **Partitions tab** right-click on the newly created component (e.g. GTEE_TJ_OD) and select the type of partition to be generated (New, Default, Design or Use).

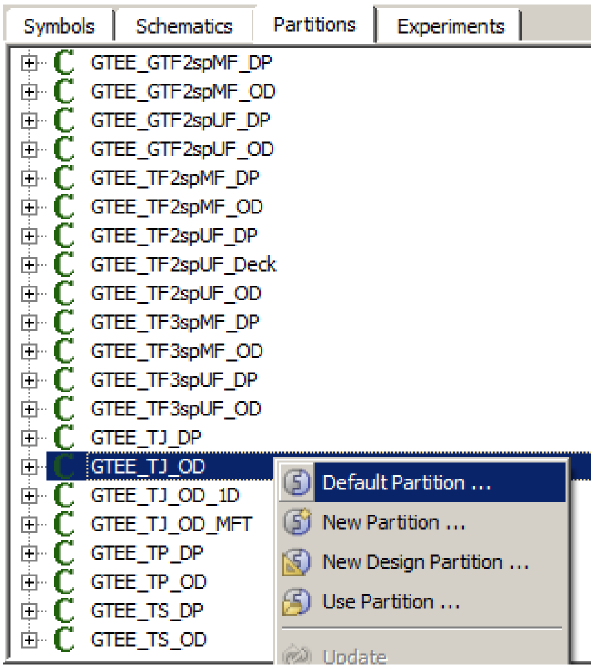


Figure 1.8: Generate Partition

9. Depending on the type of the partition specified, various wizards appear to guide you through the selection of variables for defining a valid mathematical model (design, boundary, algebraic, derivatives, high-index, etc.). These are described in Appendix B. If the default partition is selected only its name needs to be specified. Different experiments can now be defined for this partition.

III. Define Experiments

10. In the **Experiments tab** right-click on the model (componentName.PartitionName e.g. GTEE_TJ_OD.default) and select New Experiment Wizard …

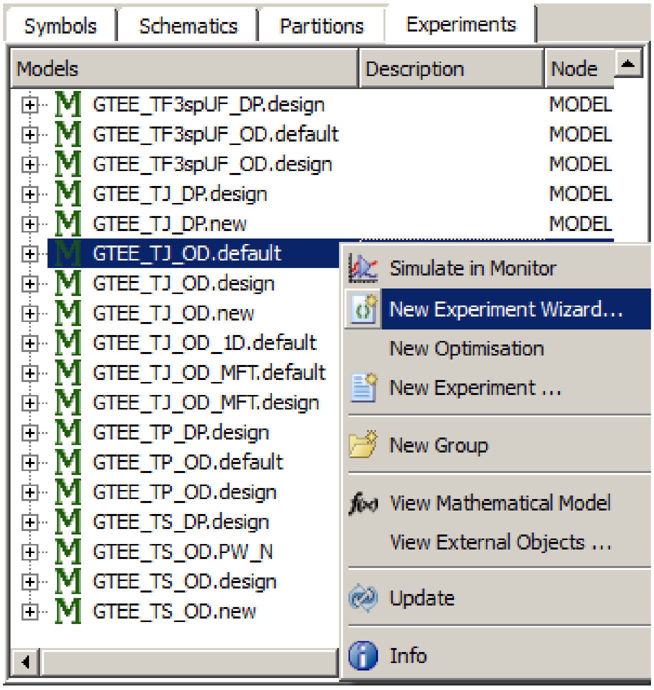


Figure 1.9: Create Experiment

11. Right-click on a case and add the required calculation (steady, design, parametric, etc). The default names of the experiment (exp1), case (case_0) and calculation (cal_steady_0 for a steady calculation) can be changed (e.g. steady calculation *runDP* in *case1* on experiment **multiCalc**).

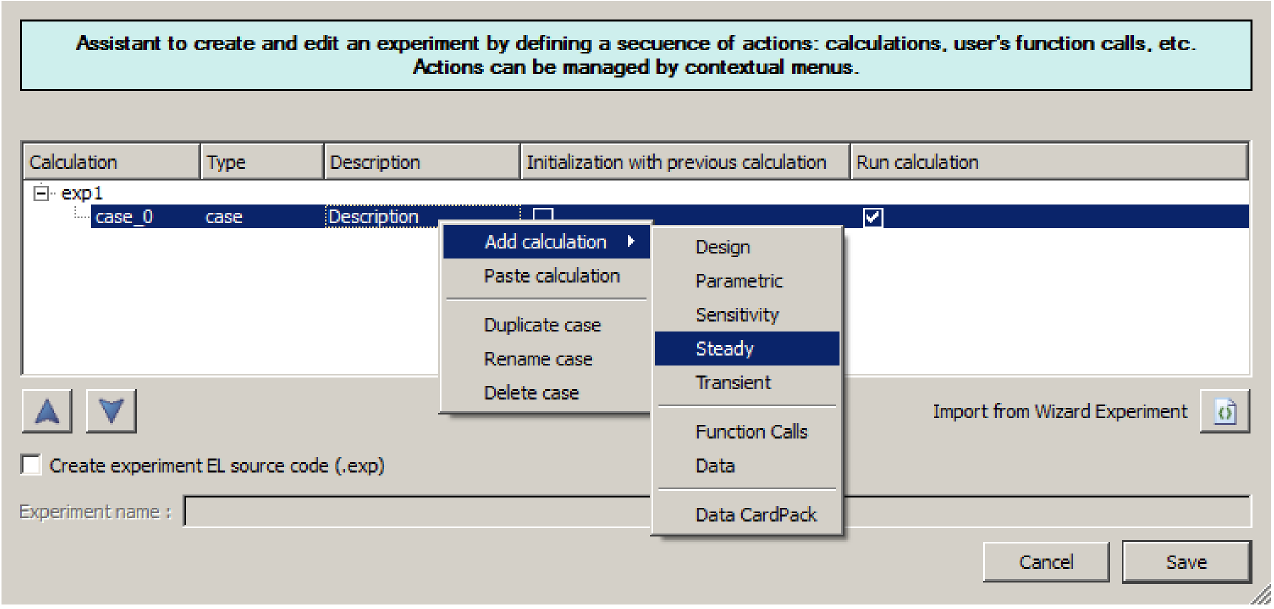


Figure 1.10: Experiment Wizard

12. Right-click on the calculation and edit it accordingly (specify solver settings, report generation, values for design, boundary, algebraic variables, etc.)

13. Finally, save the experiment. It is now ready to be simulated. More details on defining various experiments using wizards or EL are included in Appendix C.

IV. Simulate in Monitor

14. Right-click on the experiment (e.g. multiCalc) and select Simulate in Monitor

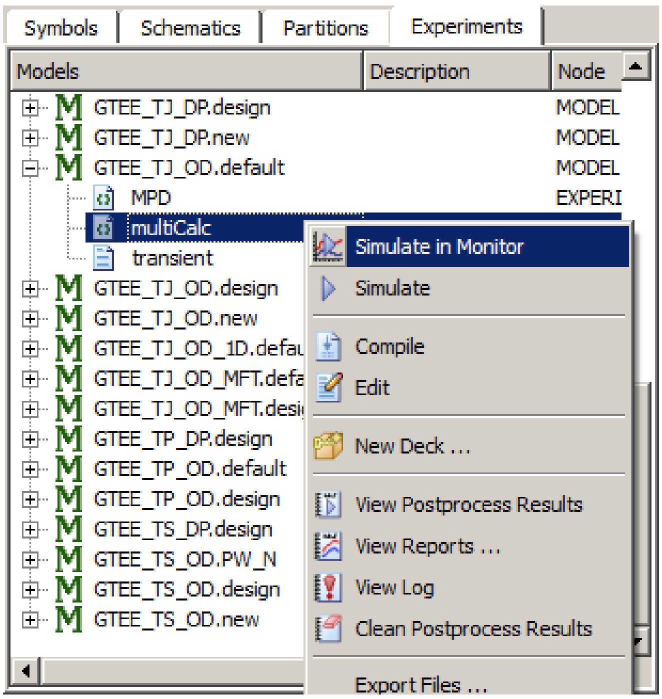


Figure 1.11: Experiment Menu

15. This opens the simulation monitor in a separate window

16. Define how the results will be viewed (component maps, plots, tables, histograms, alarms, GSE). The results can be organised in different tabs for clarity.

17. Finally, press the play button on the monitor toolbar to simulate the experiment.

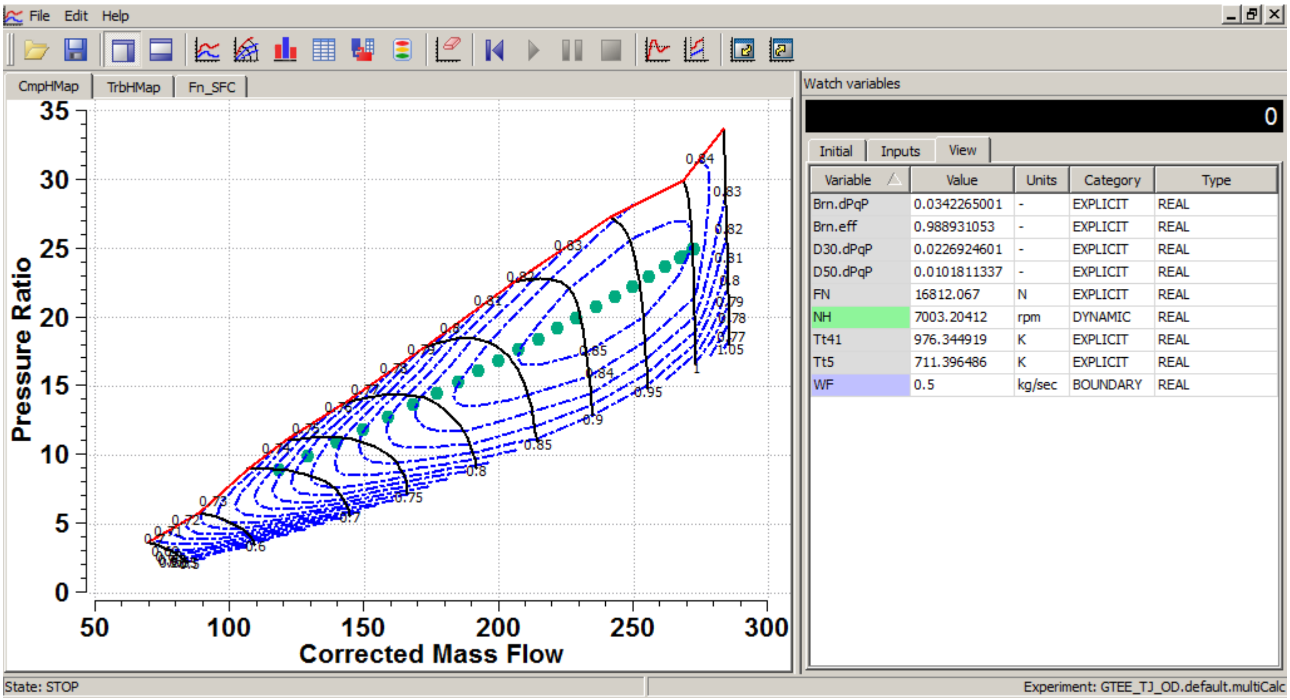


Figure 1.12: Simulation results in Monitor

Using this procedure, various examples have been prepared at engine component level in the TURBO_EXAMPLES library described in the next chapter and at engine level in the GTE_EXAMPLES library presented in Chapters 3 and 4. These could serve as a guide to become familiar with the model creation and simulation procedure as well as to use as templates to carry out simulations on similar components and engines by modifying component attributes in the schematic diagrams, selecting other boundary variables in the partitions or defining new calculations in the experiments, according to user needs.

1.4 Summary

PROOSIS is a powerful, flexible and extensible simulation tool for gas turbine performance calculations. It features an advanced graphical user interface for creating component libraries, models and simulations. It has a high-level, object-oriented, intuitive and non-causal modelling language (EL) capable of handling both continuous and discrete processes. It manages the complexity of sorting equations internally, solving linear and non-linear equation systems and optimising the numerical model, thus allowing the engineer to concentrate on the modelling aspects of the simulation.

This chapter gave a general overview of the tool's capabilities and interface and demonstrated the steps for constructing and simulating a model using components from the TURBO library. This library is described in the following chapter.

ABOUT THE AUTHORS

Dr. A. Alexiou (CEng, MIMechE) has obtained in 1994 a first class degree in Mechanical Engineering from the University of Sussex in the UK.

He then joined the Thermo-Fluid Mechanics Research Centre at the same University where he carried out experimental and numerical work on gas turbine internal air systems for various industry-relevant projects. He was awarded his PhD in 2000 for his work on fluid flow and heat transfer in compressor rotating cavities with axial throughflow.

In 2004, he joined the Laboratory of Thermal Turbomachines at the National Technical University of Athens to work on engine performance modelling and participated in the VIVACE integrated project that led to the release of PROOSIS v2.0 in 2007. Since then he continued to support the development of PROOSIS through its use in different research projects dealing with subjects such as advanced engine configurations, multi-disciplinary optimization, two-phase flow, etc. He has also written a number of PROOSIS based research articles (http://www.ltt.ntua.gr/index.php/publication/modelling).

Dr. T. Tsalavoutas has obtained in 1993 a degree in Mechanical Engineering from the National Technical University of Athens.

He then continued his research work in the field of gas turbine modelling and engine condition monitoring in the Laboratory of Thermal Turbomachines of National Technical University of Athens. He was awarded his PhD in 2004 for his work on the development of gas path analysis diagnostic techniques and the establishment of a condition monitoring system of industrial gas turbines.

Since then he participated in a number of research projects in which he was mainly involved in the development of performance models of current and future configurations of aero-engines, in the enhancement and implementation of diagnostic methods and finally in the establishment of methods that can be used to assess the environmental impact of aviation.

EA Internacional
Magallanes, 3
28015 Madrid. SPAIN
E-mail: info@proosis.com
Web: www.proosis.com

EMPRESARIOS AGRUPADOS