

Modelling and Simulation Software

EcosimPro/PROOSIS · Newsletter Nº 14 · March 2018

FROM THE EDITORS

In this new issue of our newsletter, we're excited to present some of the new features we added to the latest versions of our products at the end on 2017. We're sure our users will be pleased to see features they've specifically requested us to develop. For example, the new ability to edit all the data in a schematic diagram in a new spreadsheet-type editor. At a glance, the user can now have access to all the data of all the objects and can change them easily. This represents a great improvement for editing simulation schematic diagrams.

More and more of our users are making intensive use of classes in EL (which are similar to C++ but much more simplified). Every new version adds enhancements in this area to improve its use. For instance, this new version adds a number of new predefined classes to allow matrix calculation and linear algebra in EL language. We've opted to use the well-known Eigen3 libraries underneath, but only after making matrices and vectors much easier to use. I invite you to see some examples in this newsletter.

We've finished adding the ability to import FMI/FMU models into EcosimPro/PROOSIS. A year ago, we implemented the ability to export models using this standard for co-simulation. Now, with this new feature, EcosimPro users can easily reuse FMI/FMU models inside a model in EcosimPro/PROOSIS. This opens the door for many interesting applications such as the co-simulation, etc.

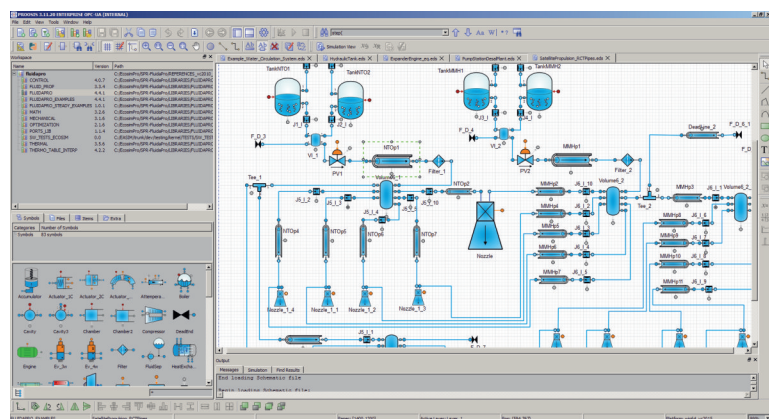
We're also putting the finishing touches on an aeronautical

simulation toolkit that we believe it will have a big impact. It will not only model the engine that already existed in the TURBO toolkit, but also the air systems (eg. environmental control systems), thermal-fluids phenomena (fuels & oil subsystems, etc.) and steam cycles (eg. refrigeration system, etc.). With this new toolkit, users can make an integrated design of aircraft systems unified into a single tool.

In this newsletter we also present a number of interesting (and complex) applications in the world of cryogenics and performance axial turbines. We hope they will be of interest to our readers.

Pedro Cobas (pce@ecosimpro.com)

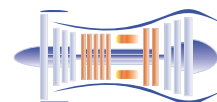
Head of the Development Team of EcosimPro/PROOSIS
EA Internacional



Propulsion System EcosimPro/FluidaPro

CONTENTS

PROOSIS: Aircraft Systems Simulation Toolkit	2	New Equations Solvers	8
ITER Cryogenic System Simulator	2	New Clases for Matrix Operations	8
Modeling ITER Cryogenic Pumps	3	Linear Equation Systems	10
Axial Turbomachinery Performance-Aerodynamics Integration in PROOSIS	4	Obtaining Eigenvalues and Eigenvectors From a Matrix	11
Exporting Models Using FMI Standard	5	Improve File Comparing Tools	11
New Attribute Editor	7	Automation Visual Library Documentation	12
		Graphic View of Dependencies in a Workspace	14



Modelling and Simulation Software

EcosimPro/PROOSIS · Newsletter Nº 14 · March 2018

1. PROOSIS: AIRCRAFT SYSTEMS SIMULATION TOOLKIT

FERNANDO RODRÍGUEZ, ECOSIMPRO/PROOSIS

A beta version of the Aircraft Systems Simulation Toolkit for PROOSIS is ready, and will soon be commercially available to simulate the thermo-fluid systems of aircraft, such as: the pneumatic system (environmental control ECS), hydraulics (fuel, oil) and steam cycles (cooling, bottoming cycles, etc.).

The libraries included in the toolkit are compatible with the TURBO library for simulating gas turbines, therefore making it possible to simulate the engine along with the systems it interacts with: fuel feed systems, oil coolant loop, air bleeding system and the ECS, etc. and of course, the thermal interactions between them.

The toolkit shares the TURBO library's philosophy in terms of modelling and scope. That makes it a useful tool for analyzing the thermodynamic cycle by making it possible to run design point studies (including all the normal calculations in these analysis: parametric, optimization, Monte Carlo, etc.) as well as off-design simulations of a particular system (static calculations and dynamic simulations). For dynamic simulations, the libraries include the most significant transient effects (slow dynamics).

The modelling flexibility (any thermal-fluid system) along with the variety of possible calculations make PROOSIS-Aircraft Systems a very useful design tool to integrate the different aircraft systems (thermal management).

Although useful for any fluid system of the aircraft, the toolkit is especially useful for simulating the ECS. It can be used to simulate any conventional or innovative configuration (more electric aircraft) based on air cycles (ACM) or vapor cycles (VCM). Not only can the toolkit be used to simulate the cooling cycle, but also the air bleeding subsystem and the cabin air distribution subsystem as well. Do not hesitate to contact us for more information (info@ecosimpro.com)

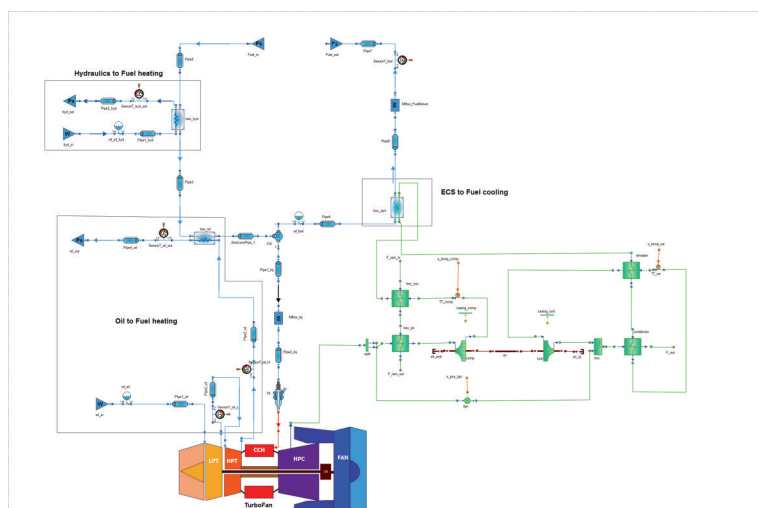


Figure 1-1 PROOSIS Aero Toolkit

2. ITER CRYOGENIC SYSTEM SIMULATOR

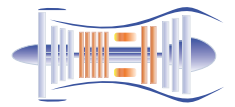
ANA MARÍA VELEIRO, ECOSIMPRO/PROOSIS

One of ITER Organization's goals is to develop an integrated simulator of the different systems making up the ITER experimental reactor under construction in Cadarache (France). The simulator is meant to bring together the individual simulators developed in the different systems and integrate them. The final purpose of the integrated simulator is to support the commissioning of ITER and the training of operators.

The ITER team responsible for the cryogenics system has been working for some time on developing models that can verify the design, design advanced control algorithms and test the control with hardware simulations in the loop.

With this aim, the simulation department at EAI has developed dynamic models of the circuits that cool the ITER magnets and initial models for the cryogenic pumps and their distribution.

Because of the complexity of the system, IO has proposed creating a distributed simulation platform that can integrate



Modelling and Simulation Software

EcosimPro/PROOSIS · Newsletter Nº 14 · March 2018

models from the different subsystems and simulate them jointly. To achieve this goal, EAI developed some particular features in EcosimPro in 2017 to allow distributed simulation of these highly complex models.

In the framework of this project, EAI endowed EcosimPro with the capability of generating OPC UA servers from the tool itself. This lets models developed in EcosimPro connect to other tools that have an OPC US interface and create a powerful distributed simulation platform. Similarly, a mechanism for synchronizing models has been developed that links up many different systems so that they can be simulated all together as a whole, with the results displayed in EcosimPro in a unified way.

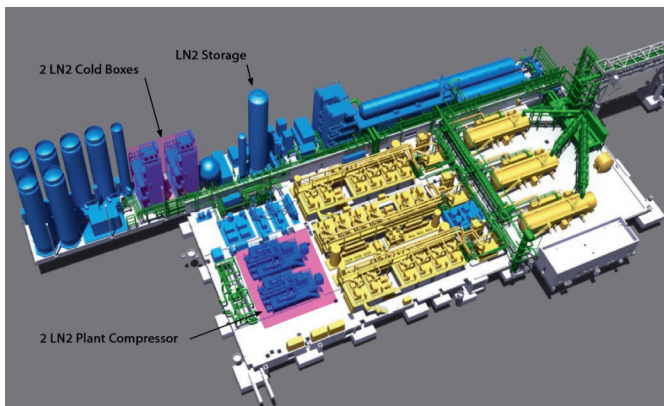


Figure 2-1 ITER Criogenic Plant

3. MODELING ITER CRYOGENIC PUMPS

ANA MARÍA VELEIRO, ECOSIMPRO/PROOSIS

The simulation team here at Empresarios Agrupados has been doing more work for the ITER experimental fusion reactor. Over this period of time, a prototype of the cryogenic distribution system in ITER (FEDCS) was developed. It includes the auxiliary cold box (ACB) of the cryogenic pumps, the distribution lines that carry the coolant from the ACB to the cold valve box (CVB), the CVB, including its control, and one of the cryogenic pumps of the torus. The final model, which will include all the cryogenic pumps, and has a twofold purpose:

- Verification of the ITER distribution system (FEDCS) that supplies coolant to the cryogenic pumps.
- Assistance in the development of the control system, providing information on the system dynamics.

The cryogenic pumps are used in high-vacuum applications and consist of an internal surface (cryogenic panels) cooled to low temperatures where the gases and vapors condense. The gas molecules are immobilized on these surfaces, thus reducing the pressure within the system. In the specific case of ITER, the cryogenic pumps serve two purposes: firstly, they are used downstream from the mechanical pumps to reduce the pressure in the vacuum vessel down to adequate pressure conditions and, secondly, they adsorb the gases from the plasma during reactor operation.

Because of the intrinsic difficulty involved in trapping helium particles, the surfaces need to be cooled to a very low temperature (4.5 K). In addition, every so often deuterium and tritium particles are adsorbed with the helium particles. Those particles will subsequently be recovered and treated in the tritium plant so they can be put back into the system. Therefore, the pumps operate in a constant cooling-pumping-heating cycle and are within a temperature range of between 4.5K during the pumping, and 470K during one of the regeneration scenarios.

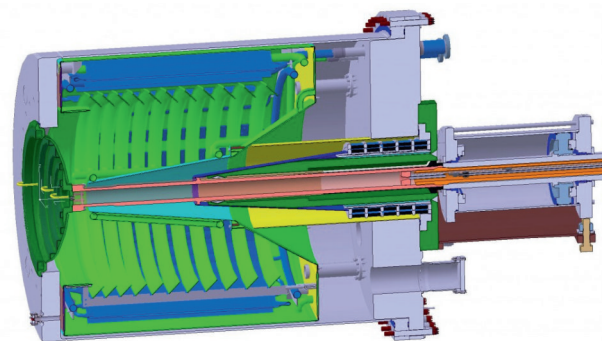
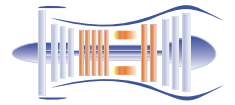


Figure 3-1 Cryogenic Pump

The FEDCS control system has to operate the clients dynamically in such a way that they comply with the operating requirements of the plasma, taking the correct regeneration of the pumps into account at the same time. Having a dynamic model available has two main advantages: first, it can be used to verify the behavior of the FEDCS elements that operate together and second, it can be used to design and verify the complex control system required for correct operation.



Modelling and Simulation Software

EcosimPro/PROOSIS · Newsletter Nº 14 · March 2018

4. AXIAL TURBOMACHINERY PERFORMANCE-AERODYNAMICS INTEGRATION IN PROOSIS

I. KOLIAS & A. ALEXIOU, NTUA (NATIONAL TECHNICAL UNIVERSITY OF ATHENS)

The preliminary design phase of a gas turbine engine involves a variety of engineering disciplines such as thermodynamics and aerodynamics (among others) and is a highly iterative procedure. The engine cycle analysis typically begins by assuming (among other parameters) the turbomachinery components efficiency. Based on the design choices made, the aerodynamic design calculations that follow may affect the assumed turbomachinery efficiencies. This way, an iterative procedure is established between the two disciplines that concludes when the efficiencies assumed at the outset of the cycle analysis match the values calculated during the aerodynamic design calculations. Basic component (and consequently engine) dimensions and weights can then be obtained.

In the context of the EU's Horizon 2020 Clean Sky 2 research and innovation project DEMOS (Developing advanced Engine Multi-disciplinary Optimization Simulations), a consistent, single-step modelling process was developed by the Laboratory of Thermal Turbomachines at the National Technical University of Athens. The method integrates, at the same modelling level, 1D aerodynamic design, flowpath geometry generation, and weight estimation capabilities into the OD performance models for axial-flow, multistage compressors and turbines.

The new turbomachinery components that integrate aerodynamic design, flowpath generation, and weight estimation functionalities inherit directly the entire mathematical model (attributes, variables, ports, equations) of the PROOSIS TURBO library ones. Iterative procedures that may be employed in order to estimate the stage pressure ratio and isentropic efficiency are executed within dedicated functions and are limited at the stage level; thus, the mathematical models at the component and engine level remain unchanged. This way, a consistent, single-step design procedure is established where there is no need for continuous data interchange between the different disciplines at component level, while the existing mathematical formulations (number of non-linear equations systems and/or algebraic variables) for performance simulations of components and the whole engine remain unaffected and at

the same level of robustness and speed of execution as for typical performance calculations.

The aerodynamic design of both compressor and turbine components is performed through a mean-line, stage-by-stage approach where the stagewise isentropic efficiency is estimated employing either semi-empirical or loss correlations. At the end of this procedure, the overall design point efficiency of the turbomachinery components is also obtained. This avoids the need of costly component iterations by assuming the component efficiency at the outset of the cycle analysis while ensuring consistency between performance and aerodynamic design calculations.

From the aerodynamic design calculations, the stagewise flow annulus radii are also computed and are used to axially size the component stages and to produce the component overall flowpath geometry. Empirical correlations utilizing both cycle and geometry parameters are used to estimate the weight of the turbomachinery and other engine related components.

The developed capability is used to conduct parametric and constrained optimization multi-point design (MPD) studies for an unmixed Ultra-High Bypass Ratio Geared Turbofan engine with Variable Pitch Fan (VPF) and/or bypass Variable Area Nozzle (VAN) and an entry into service (EIS) of 2025, while taking into account performance, aerodynamic, structural, and thermal considerations for the three main operating points of a commercial aircraft flight envelope, namely take-off, top-of-climb, and cruise.

For the VPF modelling, the fan component of the PROOSIS TURBO library was also extended so that the variation of the blade pitch angle and its effect on fan performance can be simulated, while for the VAN, the nozzle component of the PROOSIS TURBO library was extended to allow both variable and fixed area nozzles to be simulated.

Figure 4-1 depicts the schematic diagram of an UHBRGT engine that incorporates the newly developed components.

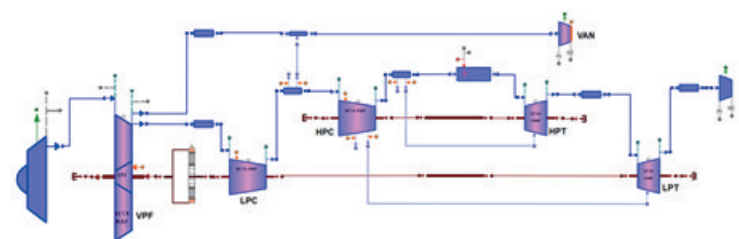
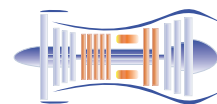


Figure 4-1 UHBRGT engine schematic model in PROOSIS



Modelling and Simulation Software

EcosimPro/PROOSIS · Newsletter Nº 14 · March 2018

Engine's installed performance is calculated by correcting the net thrust for nacelle drag and engine weight using a simplified approach. Figure 4-2 is an example of a 3D design space as obtained when using three different methods to establish the turbomachinery components efficiency at cruise conditions:

- Using the newly developed aerodynamic design approach (Method-1);
- Using correlations interrelating turbomachinery component cruise polytropic efficiency, technology level parameters, EIS, and component size;
- Using constant efficiency values (Method-3) representative of a geared turbofan engine with an EIS of 2020.

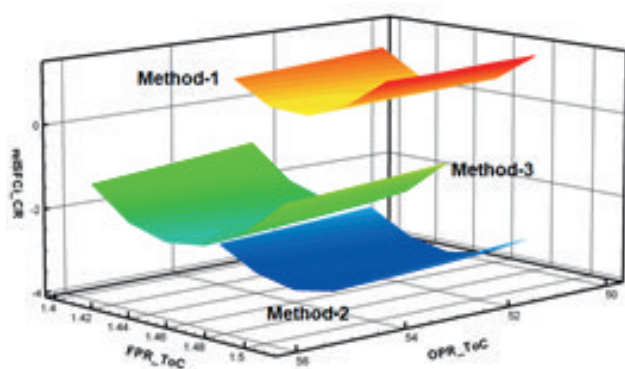


Figure 4-2. Design Space from Parametric MPD runs

Finally, through Method-1 it is possible to visualize in PROOSIS Monitor the engine flowpath. Figure 4-3 compares the flowpaths produced using the optimum values of design parameters obtained from Method-1 and Method-3, respectively (both use Method-1 to establish the component efficiencies at cruise).

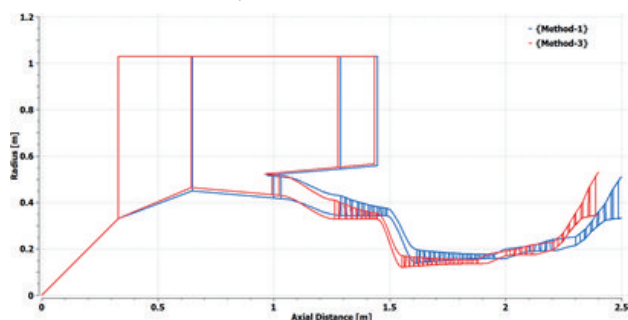


Figure 4-3. Engine Flowpath Comparison

5. EXPORTING MODELS USING FMI STANDARD

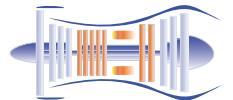
FERNANDO PUECH, ECOSIMPRO/PROOSIS

Functional Mockup Interface (FMI) is a standard for exchanging dynamic models between simulation tools. In this standard, an FMU or Functional Mockup Unit is the unit of exchange, i.e., the model that implements the FMI interface, its binaries or source code and a file that describes the capabilities of the FMI standard as well as information on the model such as the input and output variables. Version 2.0 of the standard defines two ways of exporting models to be used by tools that can understand the FMI interface:

- **Model Exchange:** equations and events are exported so that a third party can solve them using a global integrator. This part of the standard is not supported by EcosimPro or PROOSIS.
- **Co-simulation:** The model is exported as a black box that has a series of inputs and outputs and its own integrator able to synch with a master in charge of managing the simulation and exchanging data among the different FMUs. This part of the standard is the one supported by both EcosimPro and PROOSIS.

EcosimPro 5.6 and PROOSIS 3.8 introduced the ability to export models following the "FMI 2.0 for Co-simulation" standard, which lets any use model in EcosimPro and send it for use by third-party tools.

What's new to EcosimPro 5.10 and PROOSIS 3.10 is the ability to use FMI 2.0 models for co-simulation that are already generated using EcosimPro 5.6 or greater, PROOSIS 3.8 or greater or any other tool able to export models with the FMI 2.0 interface for co-simulation. This new feature can be used to complete a co-simulation diagram much like the one below using PROOSIS or EcosimPro as master simulators:



Modelling and Simulation Software

EcosimPro/PROOSIS · Newsletter Nº 14 · March 2018

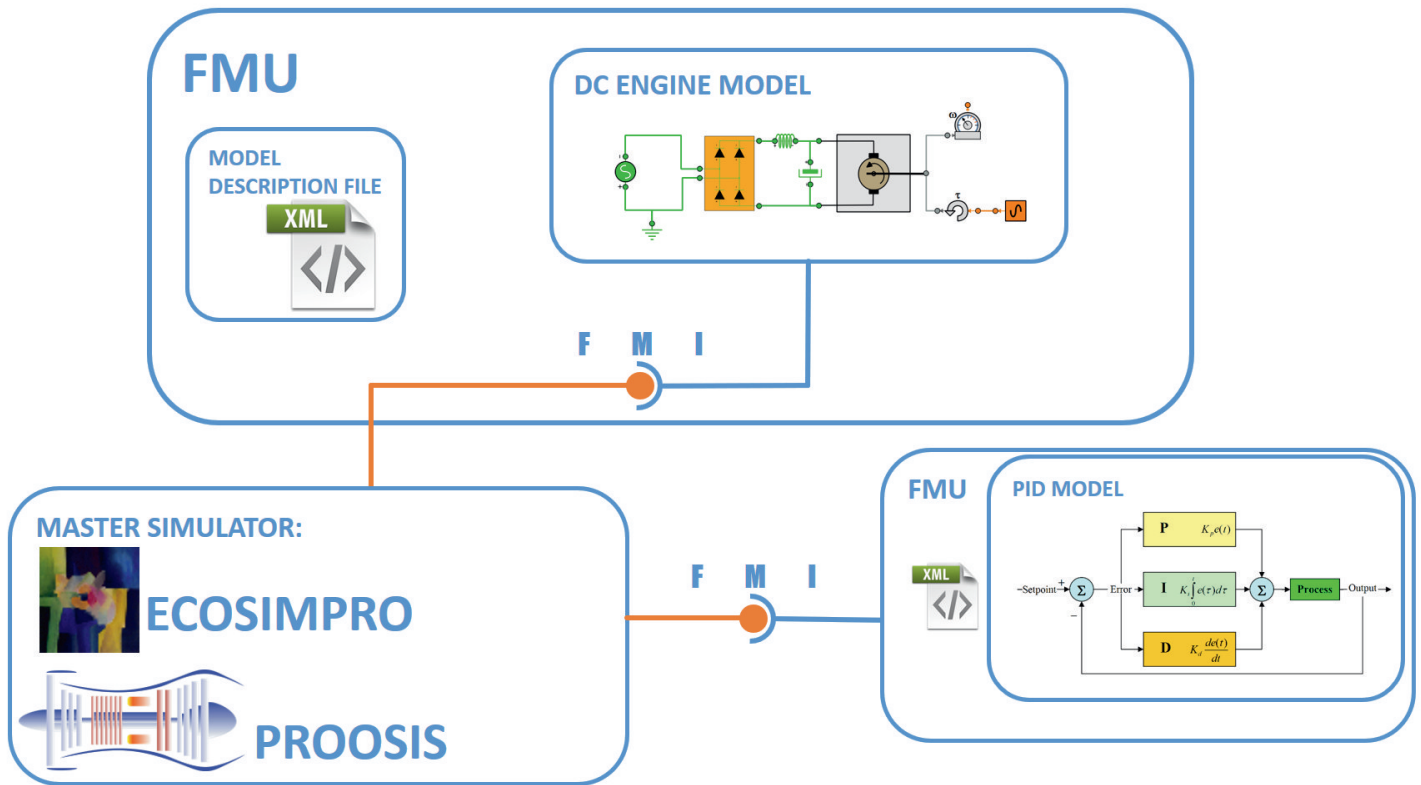


Figure 5-1 EcosimPro Co-simulation with FMI

Generating an FMU in EcosimPro or PROOSIS entails creating a model, generating a simple experiment in which to integrate until the next communication interval (CINT), and then generating a deck with the FMI 2.0 co-simulation interface, as shown in the screen shot below:

The deck creation wizard also lets you choose the variables that will be FMU inputs and outputs. Once the wizard is done, it will have created an FMU ready to send to a third party:

To control an FMU from EcosimPro and PROOSIS, they both include the COMM_FMI library to create components that interact with the FMU and to use them subsequently in other, more complex models or create power co-simulation experiments written in EL, such as the following experiment that load four FMUs containing a turbojet model and are simulated in parallel:

```
BODY
// Define starting integration time and time step
TIME = 0
CINT = 0.1
TSTOP = 100

// Load engine models and specify TSTOP and communication interval
FOR(eId = 1; eId <= nEngines; eId = eId + 1)
    hModel[eId] = cfmLoadModel("engine.fmu", TRUE)
    cfmSetCINT(hModel[eId], CINT)
    cfmSetStopTime(hModel[eId], TSTOP)
END FOR

// Integration loop
WHILE((TIME + CINT) < TSTOP)

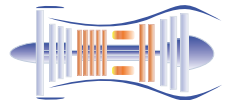
    // Send to INPUTS the engines
    FOR(eId = 1; eId <= nEngines; eId = eId + 1)
        cfmSetValueReal(hModel[eId], "FN", FN[eId]) != 0)
    END FOR

    // Run a step
    FOR(eId = 1; eId <= nEngines; eId = eId + 1)
        cfmRun(hModel[eId])
    END FOR

    // Wait for the steps to complete
    FOR(eId = 1; eId <= nEngines; eId = eId + 1)
        cfmWait(hModel[eId])
    END FOR

    // Read engine outputs
    WRITE("TIME: %g\n", TIME)
    FOR(eId = 1; eId <= nEngines; eId = eId + 1)
        FN[eId] = computeFNBasedOnWF(cfmGetValueReal(hModel[eId], "WF"))
    END FOR

    // Advance time step
    TIME = TIME + CINT
END WHILE
```

Modelling and Simulation Software

EcosimPro/PROOSIS · Newsletter Nº 14 · March 2018

For more details on how to generate FMUs or use FMUs with EcosimPro or PROOSIS, we encourage you to read the chapter on FMI in the EcosimPro and PROOSIS manual.

6. NEW ATTRIBUTE EDITOR

ANTONIO JOSÉ RIVERO, ECOSIMPRO/PROOSIS

One of the main new features in the new version of EcosimPro 5.10 and PROOSIS 3.10 is the new Attribute Editor included in the schematic diagrams.

This new tool replaces the old attribute editor and gives the end user much more flexibility and speed in handling data on the components, since it now works much like the standards followed by modern spreadsheets.

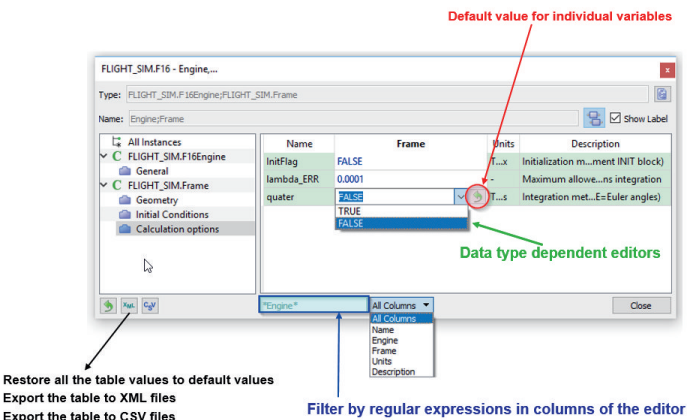


Figure 6.1. A View of the New Editor

Listing all the features of the new editor is a task that involves detail, so here we'll focus on the three main ones:

- The ability to view one or more editors at the same time
- Non-blocking editors that can work in the diagram or in the application.
- Editing multiple components of multiple types

So, with the new EcosimPro/PROOSIS, you can work with several editors open at once, including in different schematic diagrams. The end user can easily compare data, copy them from one editor to another, export them, and all while performing operations such as compiling a schematic diagram,

editing a symbol, or launching the monitor to simulate an experiment.

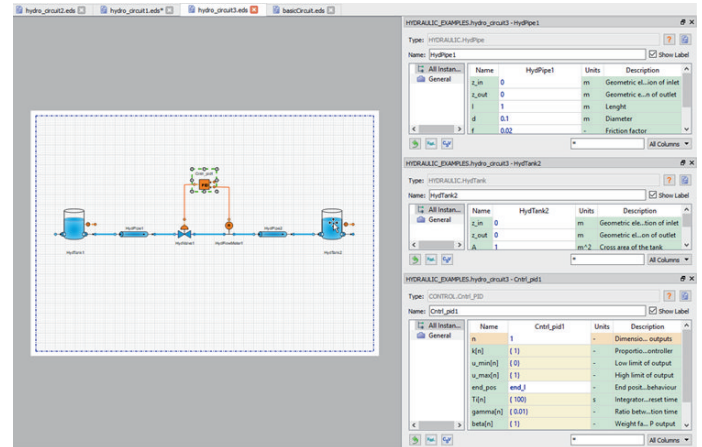


Figure 6.2. Multiple editors in the same schematic diagram

However, the main feature of the new editor is the ability to manage different components of the same schematic diagram all in one single table. In other words, users can choose whatever components they want and edit all of them at the same time. The system automatically manages the information and groups the data by name, type and unit. The end result is a spreadsheet that makes it easy to find, compare, and edit the values of dozens of components.

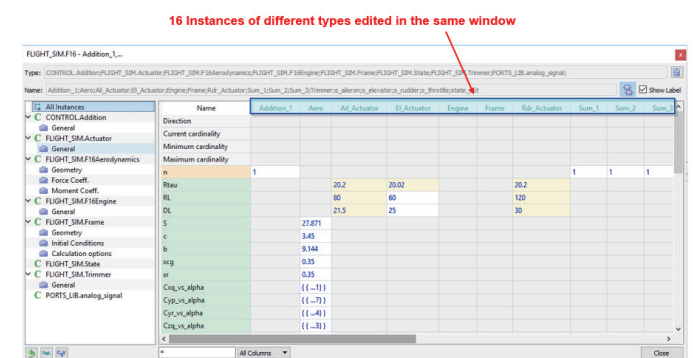
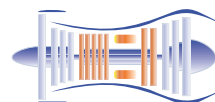


Figure 6.3. Multiple components in one single editor

And all with multiple aids in editing and filtering the data to make it easier and quicker to find variables based on their category, value, component type, name, etc.



Modelling and Simulation Software

EcosimPro/PROOSIS · Newsletter Nº 14 · March 2018

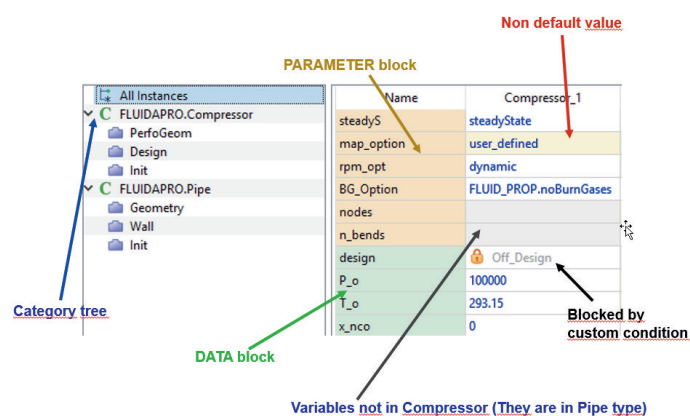


Figure 6.4

In short, the new Attribute Editor is a completely new tool, extraordinarily versatile and powerful, designed and built in answer to our users' requests. And we will keep enhancing it.

7. NEW EQUATIONS SOLVERS

FERNANDO PUECH, ECOSIMPRO/PROOSIS

EcosimPro and PROOSIS are in constant development and take special care that their numerical solvers work as efficiently and robustly as possible. The latest versions of EcosimPro and PROOSIS include several performance enhancements, one of which is the linear equations solver. The graph below shows that the new linear box solver is up to 50% faster than in previous versions while maintaining the same numerical results. Of course, not all the model has linear boxes, and depending on the number and size of linear boxes, the influence of this improvement will be more noticeable in some cases than others in cutting down on simulation time. Electrical models usually benefit the most from this improvement.

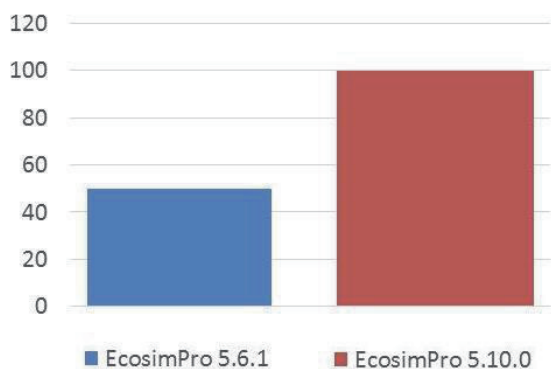


Figure 7.1. Linear box solver performance (higher is better)

In addition to the improved linear box solver, there are improvements to how memory is managed in the CVODE and IDAS solvers included in EcosimPro and PROOSIS as well as optimizing the Jacobian calculation during the solving process, leading to substantial improvements in some simulations.

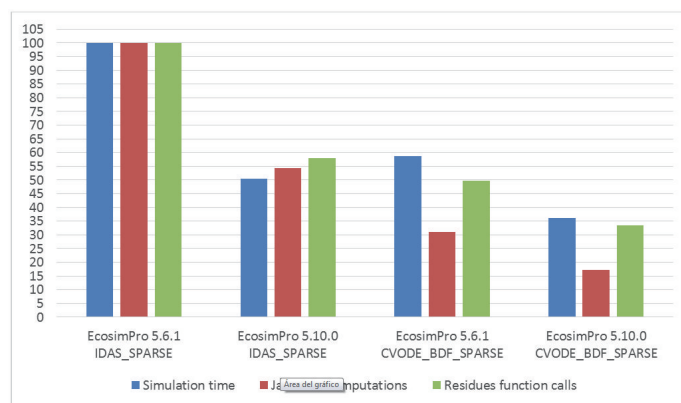


Figure 7.2. Performance of the Priming_complex ESPSS experiment using IDAS and CVODE.

EcosimPro 5.6.1 vs EcosimPro 5.10.0 (Lower relative values are better)

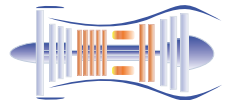
EcosimPro and PROOSIS have not only improved the existing solvers, but have also added the following to give the user even more flexibility when simulating:

- **RK45:** ODE solver, explicit, variable-step, equivalent to Matlab's ode45. It converges faster than RK4.
- **BACKEULER:** ODE solver, implicit and fixed-step. Useful to solve "stiff" problems in situations needing a fixed step.
- **BACKEULER_SPARSE:** ODE solver, implicit and fixed-step. Useful to solve "stiff" problems in situations needing a fixed step. This solver is more efficient than BACKEULER and is meant to make use of the structure of the model by reducing the number of times the residues function is called.

8. NEW CLASSES FOR MATRIX OPERATIONS

PEDRO COBAS, ECOSIMPRO/PROOSIS

The EigenMatrix, EigenVector and EigenRowVector predefined classes allow users to handle matrices and vectors of real numbers in a simple and intuitive way. These classes can be used to perform typical linear algebra operations such as



Modelling and Simulation Software

EcosimPro/PROOSIS · Newsletter Nº 14 · March 2018

matrix arithmetic (additions, multiplications, etc), determinants, calculation of eigenvalues, etc.

The classic operators for linear algebra operations can be used in the Eigen classes (matrices and vectors), making it highly powerful. For instance, the following function adds two 2x3 matrices and prints the result:

```
FUNCTION NO_TYPE testInitial()
DECLS
  REAL aInit[2,3]= {{1,4,1},{2,2,2}}
  REAL bInit[2,3]= {{4,6,8},{2,5,2}}
OBJECTS
  EigenMatrix A, B, Result
BODY
  A.setFromArray2D(aInit,2,3)
  B.setFromArray2D(bInit,2,3)
  Result = A + B
  WRITE("A=\n%s \n", A.asString())
  WRITE("B=\n%s \n", B.asString())
  WRITE("Result= A + B =\n%s \n",
    Result.asString())
END FUNCTION
```

The output of this function is:

```
A=
[1, 4, 1]
[2, 2, 2]

B=
[4, 6, 8]
[2, 5, 2]

Result= A + B =
[ 5, 10, 9]
[ 4, 7, 4]
```

The addition has been completed using only the '+' operator. In other words, these objects can be used as if they were conventional mathematical operators.

The user can copy the arrays values from an EL array and, the opposite, he can copy back the eigen array to an EL object, for example:

```
FUNCTION NO_TYPE testCopyFromArray2D()
DECLS
  REAL mat1[3,3] = {{1,2,3},{4,5,6},
    {7,8,9}}
  REAL mat2[3,3]
OBJECTS
  EigenMatrix a, b, c -- dynamic matrix
BODY
  a.setFromArray2D(mat1,3,3)
```

```
b.setFromArray2D(mat1,3,3,1,2,3,1)
WRITE("a=\n%s\n",a.asString())
WRITE("b=\n%s\n",b.asString())
a.copyToArray2D(mat2,3,3)
FOR(i IN 1,3)
  FOR(j IN 1,3)
    WRITE("mat2[%d,%d]=
      %g\n",i,j,mat2[i,j])
  END FOR
END FOR
END FUNCTION
```

The output of this function is:

```
a=
[1, 2, 3]
[4, 5, 6]
[7, 8, 9]

b=
[2]
[5]
[8]

mat2[1,1]= 1
mat2[1,2]= 2
mat2[1,3]= 3
mat2[2,1]= 4
mat2[2,2]= 5
mat2[2,3]= 6
mat2[3,1]= 7
mat2[3,2]= 8
mat2[3,3]= 9
```

Matrix a is a copy of array mat1, and matrix b is created from the second column of mat1. Lastly, the values of the matrix are copied in array mat2 and then printed on the screen.

Also the initial array can be extracted from a string using the row separator ";" and column separator ",", for instance:

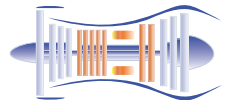
```
a.setFromStr("1,2,3;4,5,6;7,8,9") --
dimension 3x3
```

Initialise the matrix.

```
[1, 2, 3]
[4, 5, 6]
[7, 8, 9]
```

The user can change partially any matrix with other submatrix with the setChunk() method, for instance:

```
FUNCTION NO_TYPE testSetChunck()
OBJECTS
  EigenMatrix a, b, c
```



Modelling and Simulation Software

EcosimPro/PROOSIS · Newsletter Nº 14 · March 2018

BODY

```
a.resizeInit(5,5,0)
b.resizeInit(2,2,8)
WRITE("a=\n%s\n",a.asString())
WRITE("b=\n%s\n",b.asString())
a.setChunk(3,3,2,2,b)
WRITE("a after a.setChunk(3,3,2,2,b) =
      \n%s\n",a.asString())
```

END FUNCTION

It produces the output:

```
a=
[0, 0, 0, 0, 0]
[0, 0, 0, 0, 0]
[0, 0, 0, 0, 0]
[0, 0, 0, 0, 0]
[0, 0, 0, 0, 0]

b=
[8, 8]
[8, 8]

a after a.setChunk(3,3,2,2,b) =
[0, 0, 0, 0, 0]
[0, 0, 0, 0, 0]
[0, 0, 8, 8, 0]
[0, 0, 8, 8, 0]
[0, 0, 0, 0, 0]
```

The submatrix b has been inserted at position (3,3) in a matrix.

Also the user can calculate some typical operations in matrices such as inverse, determinant, transpose, etc. For instance:

```
FUNCTION NO_TYPE testGetTransf()
DECLS
  REAL v1[3,3]= {{0,1,1},{1,0,0},{0,0,1}}
OBJECTS
  EigenMatrix p1,plinv,plad,pltrans
BODY
  p1.setFromArray2D(v1,3,3)
  WRITE("p1=\n%s\n", p1.asString())
  WRITE("p1.determinant()= %g\n",
        p1.determinant())
  plinv= p1.inverse()
  pltrans= p1.transpose()
  plad= p1.adjoint()
  WRITE("p1.inverse()=\n%s\n",
        plinv.asString())
  WRITE("p1.transpose()=\n%s\n",
        pltrans.asString())
  WRITE("p1.adjoint()=\n%s\n",
        plad.asString())
END FUNCTION
```

The output is:

```
p1=
[0, 1, 1]
[1, 0, 0]
[0, 0, 1]

p1.determinant()= -1

p1.inverse()=
[ 0, 1, 0]
[ 1, 0, -1]
[ 0, 0, 1]

p1.transpose()=
[0, 1, 0]
[1, 0, 0]
[1, 0, 1]

p1.adjoint()=
[0, 1, 0]
[1, 0, 0]
[1, 0, 1]
```

9. LINEAR EQUATION SYSTEMS

PEDRO COBAS, ECOSIMPRO/PROOSIS

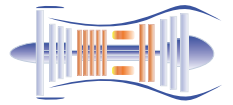
The new class EigenHandler provides a solve() method used to solve a linear system of equations. It uses the method with the Householder rank-revealing QR decomposition of a matrix with column-pivoting. The required parameters are:

A: is a square matrix of size NxN
B: is a column vector of size N
sol: matrix with the solution to the problem

The method returns TRUE if there has been no problem in the resolution of the system of equations and returns FALSE if any problems have been encountered. For instance, to solve the following system of equations:

$$\begin{aligned} 4x + 5y - 12z &= 98 \\ 36x - 8y + 3z &= 23 \\ -3x + 5y - 2z &= 45 \end{aligned}$$

A matrix of 3x3 size shall be created with the coefficients of the equation in x,y,z (matrix A) and column vector of size 3 to insert the independent values (matrix B). The following function solves this equation:



Modelling and Simulation Software

EcosimPro/PROOSIS · Newsletter Nº 14 · March 2018

```

FUNCTION NO_TYPE testSolve()
DECLS
  REAL relative_error
OBJECTS
  EigenMatrix A, sol
  EigenVector B
  EigenHandler eh
BODY
  A.setFromStr("4,5,-12;36,-8,3;-3,5,-2")
  B.setFromStr(" 98; 23; 45")
  WRITE("A=\n%s\n",A.asString())
  WRITE("B=\n%s\n",B.asString())
  eh.solve(A, B, sol)
  WRITE("Solution\n%s\n",sol.asString())
END FUNCTION

```

The output is:

```

A=
[ 4, 5, -12]
[ 36, -8, 3]
[ -3, 5, -2]

```

```

B=
[98]
[23]
[45]

```

```

Solution
[ 3.02447]
[ 9.54153]
[-3.18287]

```

10. OBTAINING EIGENVALUES AND EIGENVECTORS FROM A MATRIX

PEDRO COBAS, ECOSIMPRO/PROOSIS

The class EigenHandler provides two methods for calculating the eigenvalues() and eigenvectors() of a matrix. methods return the eigenvalues and eigenvectors of a matrix.

Below is an example to calculate the eigenvalues and eigenvectors of the following matrix:

```

[-3, 0, 2]
[ 1, -1, 0]
[-2, -1, 0]

```

This can be done by passing as arguments to matrices, one of them for storing the real part of the solution (eg var, ver) and the other one for the imaginary part of the solution (eg

vai,vei):

```

FUNCTION NO_TYPE testEigenValues()
DECLS
  REAL Ainitial[3,3]={{-3, 0, 2},{1, -1, 0},
                      {2, -1, 0}}
OBJECTS
  EigenMatrix A, var, vai,ver,vei
  EigenHandler eh
BODY
  A.setFromArray2D(Ainitial,3,3)
  WRITE("A=\n%s\n",A.asString())
  eh.eigenvalues(A,var,vai)
  WRITE("Eigenvalues:\n%s\n",
        eh.complexAsString(var,vai))
  eh.eigenvectors(A,ver,vei)
  WRITE("Eigenvectors matrix:\n%s\n",
        eh.complexAsString(ver,vei))
END FUNCTION

```

The output is:

```

A=
[-3, 0, 2]
[ 1, -1, 0]
[-2, -1, 0]

```

```

Eigenvalues:
[-1+1.41421i]
[-1-1.41421i]
[-2]

```

```

Eigenvectors matrix:
[-0.540449-0.203096i, -0.540449+0.203096i,
-0.666667]
[-0.143611+0.382155i, -0.143611-0.382155i,
0.666667]
[-0.396839-0.585251i, -0.396839+0.585251i,
-0.333333]

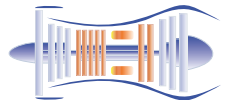
```

In this example both, the eigenvalues and eigen vectors are complex numbers

11. IMPROVED FILE COMPARING TOOLS

FERNANDO PUECH, ECOSIMPRO/PROOSIS

EcosimPro and PROOSIS have joined file comparing options to make it easier to spot differences between results from simulations, experiments, partitions, etc. This change shows the comparison options in the context menu of the different elements in the "Files" tab and the "Items" tab:



Modelling and Simulation Software

EcosimPro/PROOSIS · Newsletter Nº 14 · March 2018

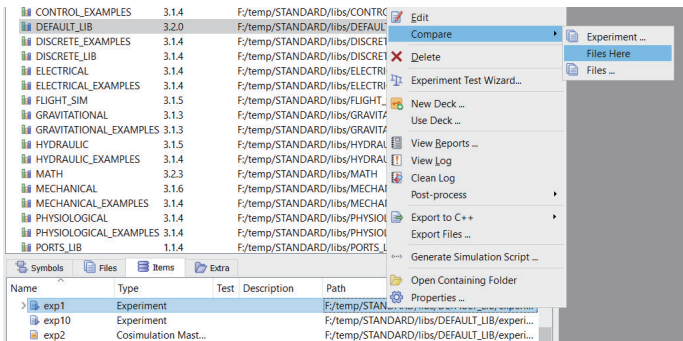


Figure 11.1. Select Compare

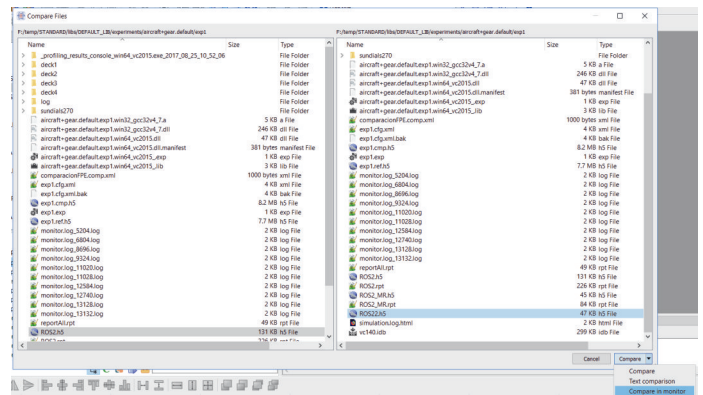


Figure 11.3. Navigation Window

how to compare them by dropping down the options from the "Compare" button:

When comparing files, EcosimPro and PROOSIS try to make the best comparison they know in terms of the file type. In other words, they compare files as if they were text, unless

If two post-process files are compared on the monitor, it

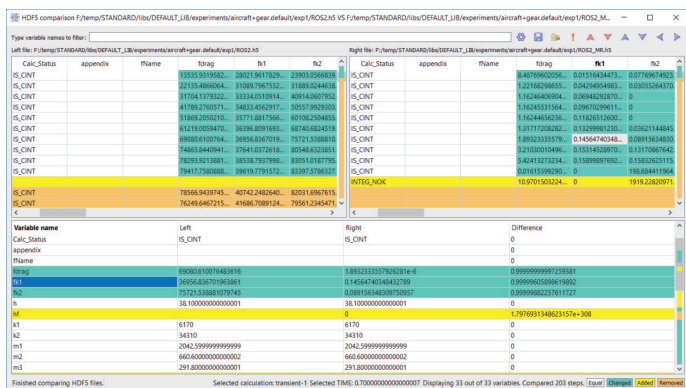


Figure 11.2. Compare HDF5 Files

they are post-process files with an h5 extension (HDF5 format) in which they make a detailed binary comparison:

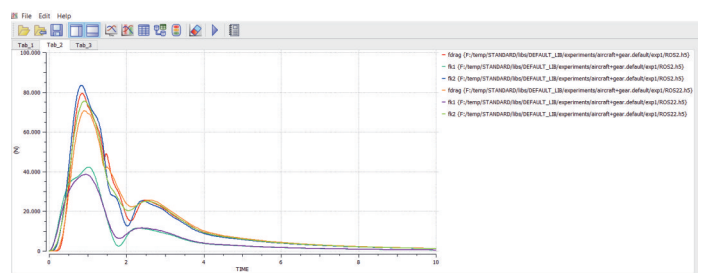


Figure 11.4. Post-Process Comparison

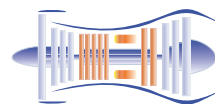
makes a graphic comparison of any variables that exist in both files and both have the same type, such as shown in the screen shot below:

12. AUTOMATION VISUAL LIBRARY DOCUMENTATION

EcosimPro/PROOSIS can generate automatic library documentation, so in this Newsletter we'll show the various different views we can have of the components. The library we'll use is TURBOJET from the DEFAULT workspace. We can generate automatic documentation from this library if we choose the library and the option: "Documentation->Generate

TURBOJET Library | Components | Ports | Classes | Functions | Enumeratives | Global Variables

TURBOJET Library Documentation



Modelling and Simulation Software

EcosimPro/PROOSIS · Newsletter Nº 14 · March 2018

Documentation", which opens a browser with the following header:

From here we can browse through the different components, ports, classes, functions, enumeratives and global variables in

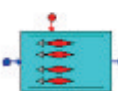

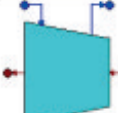
TURBOJET Library	Components	Ports	Classes	Functions	Enumeratives	Global Variables
	Alphabetical	Inheritance Tree	Inheritance Graph			

the TURBOJET library. Let's take a closer look at what's under the Components heading. If we click on Components, it takes us to another page with this header:

It gives us three different views to see the components: one in

Components

Note: click [+] for accessing the source code file

Symbol	Name	Description
	AfterBurner [+]	Afterburner
	Burner [+]	Burner
	Compressor [+]	Compressor

alphabetical order, another with a inheritance tree in text mode and the last with the inheritance tree in graph mode. In the first we see a table with each component, its icon and a description:

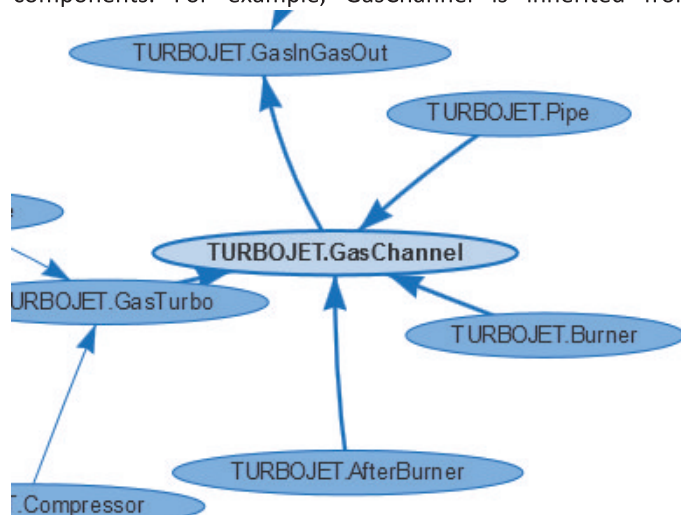
If we click on the [+] sign it takes us directly to the source code for this component (only if available). If we choose the "Inheritance Tree" option, we see:

Component Inheritance Tree

Note: click [+] for accessing the source code file

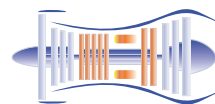
- [FuelTank \[+\]](#)
- [FuelValve \[+\]](#)
- [Fuelmeter \[+\]](#)
- [GasInGasOut \[+\]](#)
 - [GasChannel \[+\]](#)
 - [AfterBurner \[+\]](#)
 - [Burner \[+\]](#)
 - [GasTurbo \[+\]](#)
 - [Compressor \[+\]](#)
 - [Fan \[+\]](#)
 - [Turbine \[+\]](#)
 - [Pipe \[+\]](#)
 - [Plane \[+\]](#)

in which we see the inheritance hierarchy among components. For example, GasChannel is inherited from



GasInGasOut and Afterburner is in turn inherited from GasChannel. If we want to see this information in graph mode, we can choose the "Inheritance Graph" view, which displays a graph like this:

Each bubble represents a component. When a bubble has an



Modelling and Simulation Software

EcosimPro/PROOSIS · Newsletter Nº 14 · March 2018

arrow pointing to another bubble, it means it is Inherited from that component. For example, we see that GasInGasOut points to GasInGasOut, indicating that the former is inherited from the latter, and so on for the rest of the components. It's a useful view because it shows us all the inheritance relationships of all the components in the library at a glance.

Moreover, the user can interact with the graph by moving the bubbles around with the mouse into new arrangements; when one bubble is moved, the others move automatically to make way for it. We recommend users to reach this view and "play" with this high-level overview of the library.

13. GRAPHIC VIEW OF DEPENDENCIES IN A WORKSPACE

Often, when working in a workspace, we need to see the dependencies between libraries, i.e., when a library uses elements from other libraries. This information tells us that we need to have them all before we can compile the final library. Moreover, they have to be compiled in the right order

so that all the components, ports, classes and functions are compiled in chronological order of use.

The tool comes with a graphic utility that gives a high-level overview of these dependencies between libraries. We only have to choose the "Properties" option associated with a workspace, and on the "Information" tab we see a graph with the dependencies. For example, if we do it in the TURBO workspace, we the screenshot below.

In this case we see that the TURBO library uses the MATH, PORT_LIB and CONTROL libraries. Other libraries such as GTE_TURBOSHIFT and GTE_EXAMPLES in turn use the TURBO library.

The user can move the bubbles around with the mouse and rearrange them to give a different view of the graph, which lets each user create his own custom-made views.

We've had a quick look here at the dependency among libraries that can often be helpful to get a clear idea of their compilation order and their dependencies.

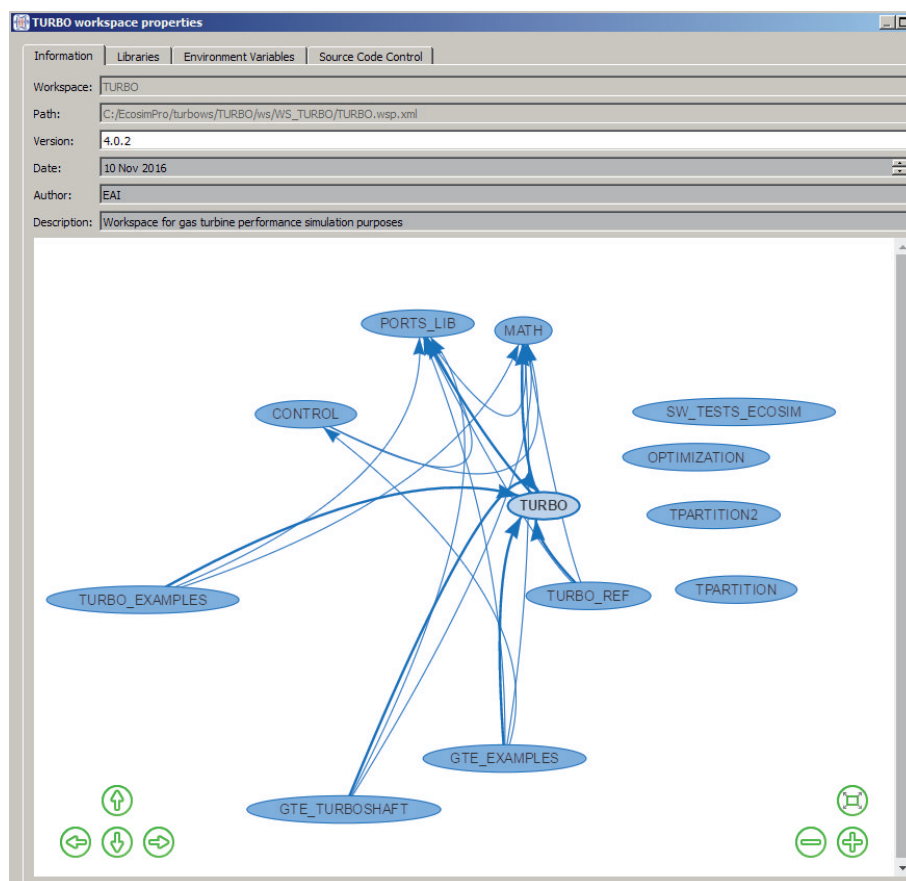


Figure 13.1. Workspace Dependencies

EA Internacional S.A.
Magallanes, 3 Madrid
28015 Spain
E-mail: info@ecosimpro.com
URL: <http://www.ecosimpro.com>
Phone: +34 91 309 81 42
Fax: +34 91 591 26 55



EDITED BY: INMACULADA REYES
REVIEWED BY: ÁNGEL BARRASA