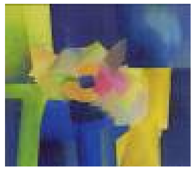
EA International



EcosimPro/Proosis Workshop 2014

Empresarios Agrupados

Phone: 34 – 91 444 08 80

http: www.ecosimpro.com

KopooS Consulting Ind.

Official distributor of Proosis/EcosimPro worldwide

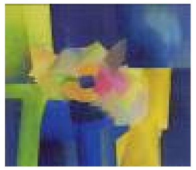
Commercial agency of Empresarios Agrupados worldwide

Phone: 33 – 6 08 60 26 73

http: www.kopoos.com

KopooS Consulting Ind.

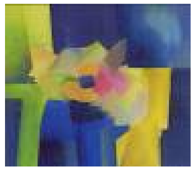
EA International



INDEX

KopooS Consulting Ind.

EA International



EcosimPro Proosis Overview

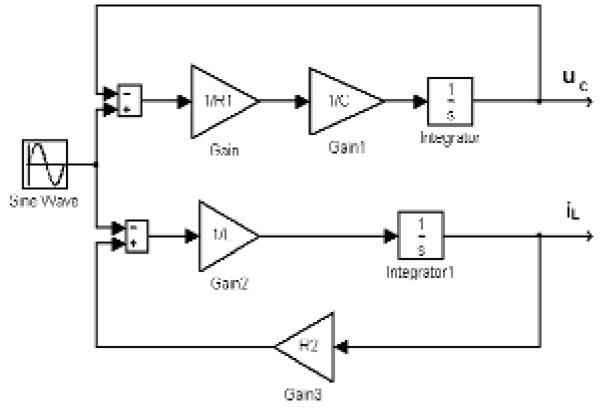
•**Proosis** is a dedicated **EcosimPro** release for **turbojets**

•The following presentation is related to **EcosimPro** but the capabilities of the two tools are the same

KopooS
Consulting Ind.

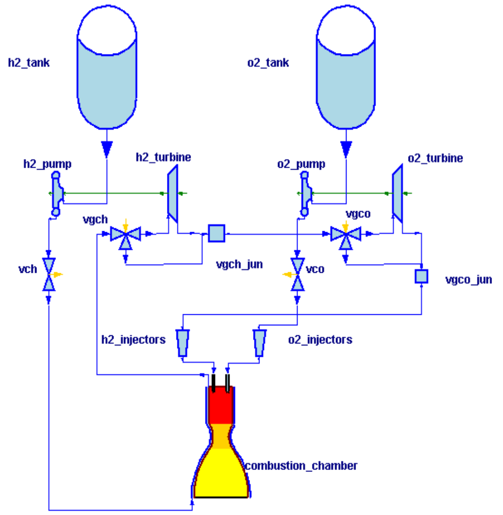
EcosimPro Overview-Object oriented tool

- What is EcosimPro?
  - EcosimPro (1989) is a general SYSTEM simulation tool which is object oriented instead of block diagram oriented (like Simulink)
  - Transient and steady states can be solved



Simulink Model

Block Diagram Model



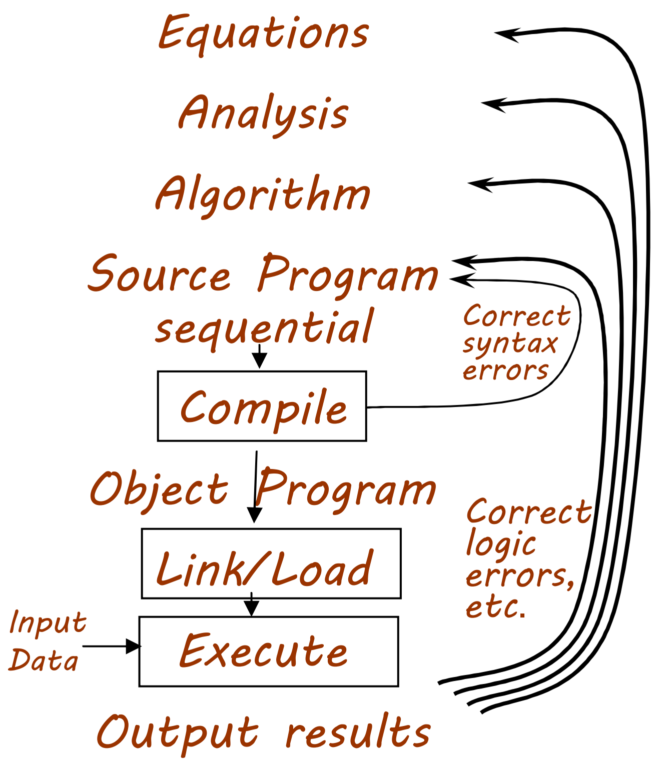
EcosimPro Model

Object Oriented Model

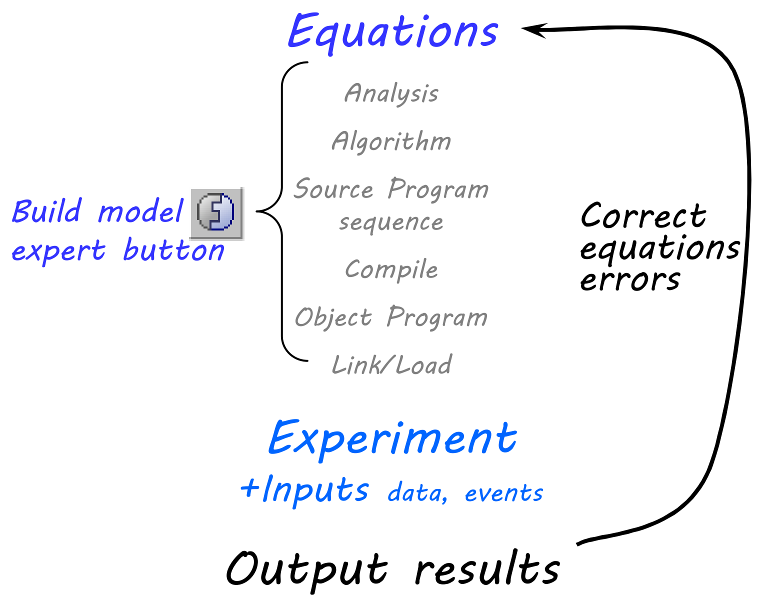
EcosimPro Overview - Acausal tool

- What is Proosis EcosimPro?
  - EcosimPro (1989) is a general SYSTEM simulation tool which is acausal*

*acausal: the written equations are not sorted, not solved



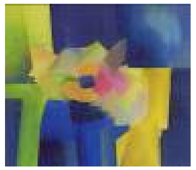
Standard software



EcosimPro acausal equations system Model

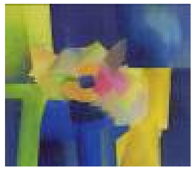
EA International

EcosimPro Overview - Users



- **LEVEL 1: Users who develop libraries of components.**
  need to have a profound knowledge of the physics and mathematics of the problem to model and simulate it.
  need to create new components using EcosimPro's modelling language EL.
  For example, a creator of a basic ELECTRIC library with capacitors, resistors, inductances, etc.
- **LEVEL 2: Users who create models based on existing libraries.**
  no need to have the extent of knowledge LEVEL 1
  but need to know what is under the components.
  to generate and simulate the schematics that represent the different physical systems.
  For example a user creating an electric circuit based on the ELECTRICAL library.
- **LEVEL 3. Users who just run simulations from existing models.**
  need only be capable of changing the input data and running the simulations to obtain the results.
  not require any special math knowledge;
  just need background knowledge of the final application.
  For example, an operator in a plant needs to know what happens in a scenario if some of the plant parameters change. He can run the simulation with the new parameters to obtain the results.
- **LEVEL 4. This level does not require any knowledge of the tool, just how to connect to EcosimPro from other tools.**
  Because Models of EcosimPro can be exported to other tools such as Excel, Matlab, Simulink, etc.
  Also the tool can export models as black boxes that can be reused as a standalone programme.

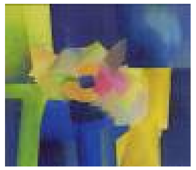
KopooS
Consulting Ind.



EcosimPro Overview - Main concepts

- OBJECT TOOL
  - PORT **connection type: It defines the connective points of the components with several variables managed by the equations. A different port type is** required **for every discipline (e.g. Electrical, Fluid, Chemical,etc.)**
  - COMPONENT**: It represents a model (with ports) by means of variables, Differential-Algebraic equations (e.g. between ports variables), topology and event-based behaviours (e.g. Resistor, Pump, Valve ON or OFF or transient opening, Pipe, Tank, etc.)**
  - Organisation into LIBRARIES**: for encapsulating components, ports, enumerative types and global variables in a hierarchical place (e.g.. CONTROL, ELECTRICAL, THERMAL libraries, etc.)**
- BUILD MODEL
  - PARTITION**: To simulate a component, it is first needed to create a solved mathematical model. The additional information needed to generate the mathematical model is the partition (a component can have several partitions)**
- EXPERIMENT**: Simulation cases with the input data and process events (commands valves ON, OFF, transient ON, etc.)**

EA International



EcosimPro Overview – General Structure

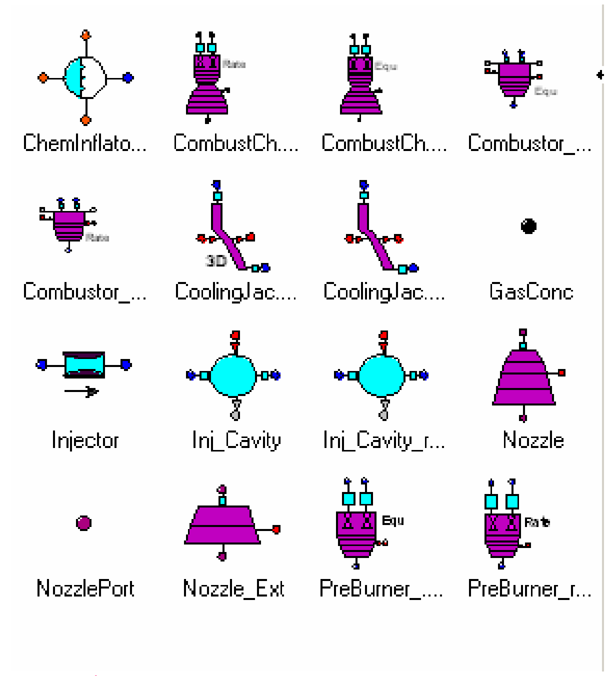
```
USE ...
COMPONENT GeneralStructureCompo
  PORTS
    ...
  DATA
    ...
  DECLS
    ...
  OBJECTS
    ...
  TOPOLOGY
    ...
  INIT
    ...
  DISCRETE
    ...
   CONTINUOUS
       ... Equations
       ... Equations
       ... Equations

END COMPONENT
```

The structure can include several parts in addition to the continuous equations

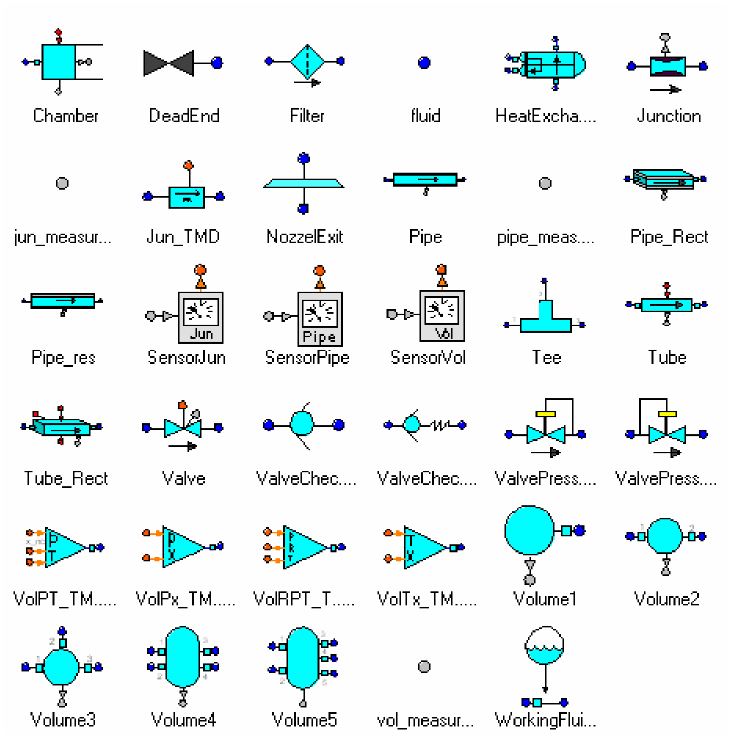
EA International

EcosimPro Overview - Libraries

- Examples of developed Libraries are :
  - Control, Electrical & Mechanical circuits
  - Two-Phase Fluid flow in Pipe Networks
  - Rockets engines simulation



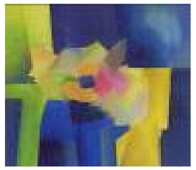
ESPSS * COMB_CHAMBERS palette of symbols
*available only under ESA agreement



ESPSS * FLUID_FLOW_1D palette of symbols
*available only under ESA agreement

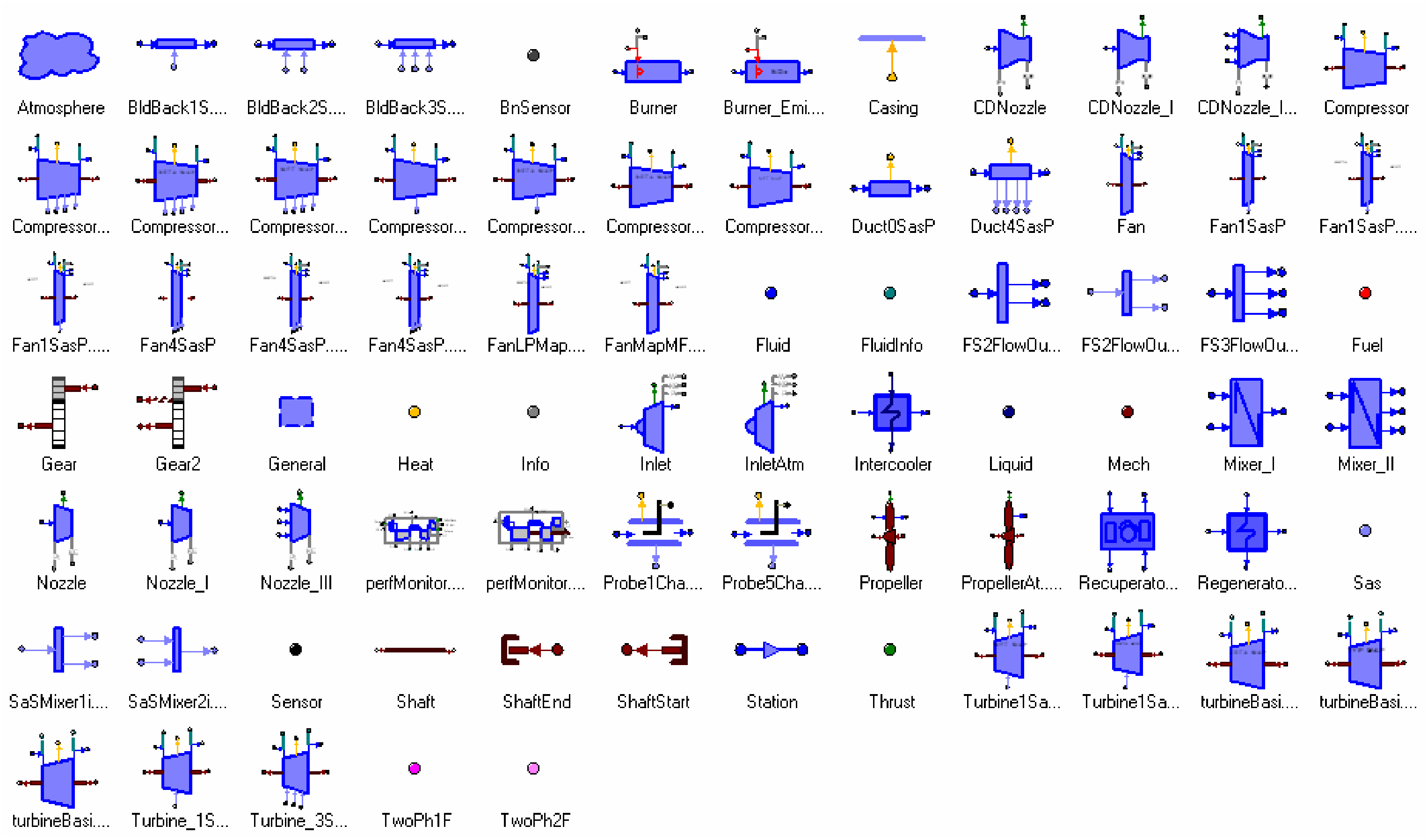
KopooS Consulting Ind.

EA International



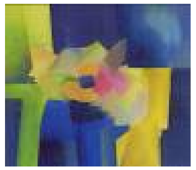
EcosimPro Overview - Libraries

– Gas Turbines



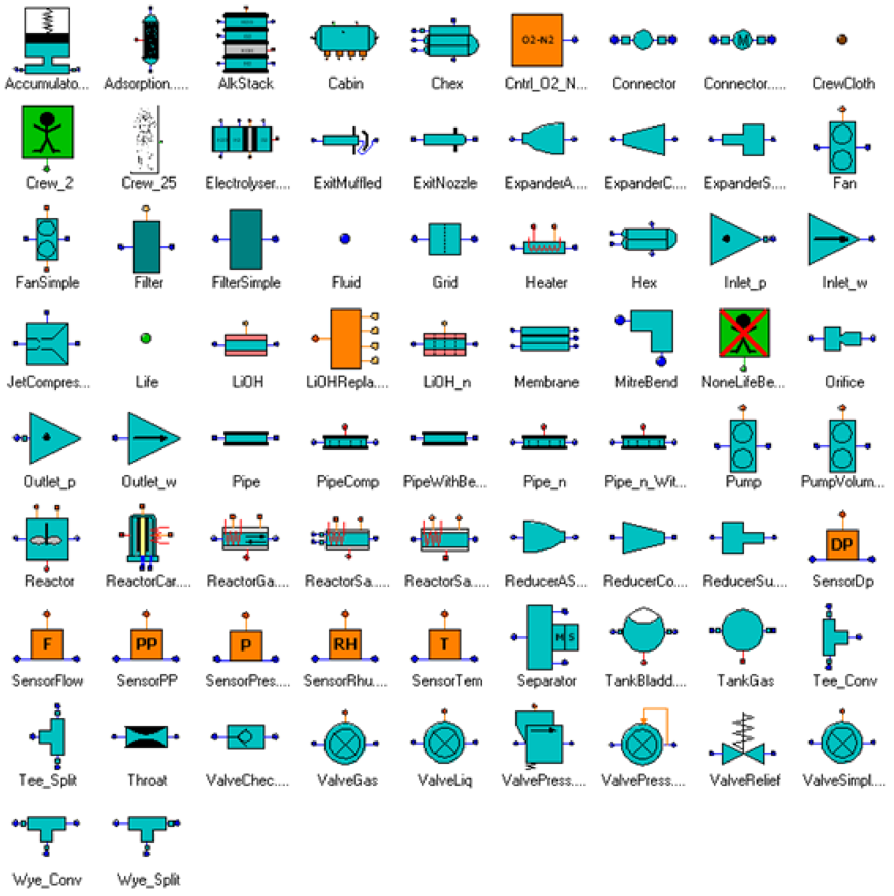
KopooS Consulting Ind.

EA International



EcosimPro Overview - Libraries

– Environment Control & Life Support Systems



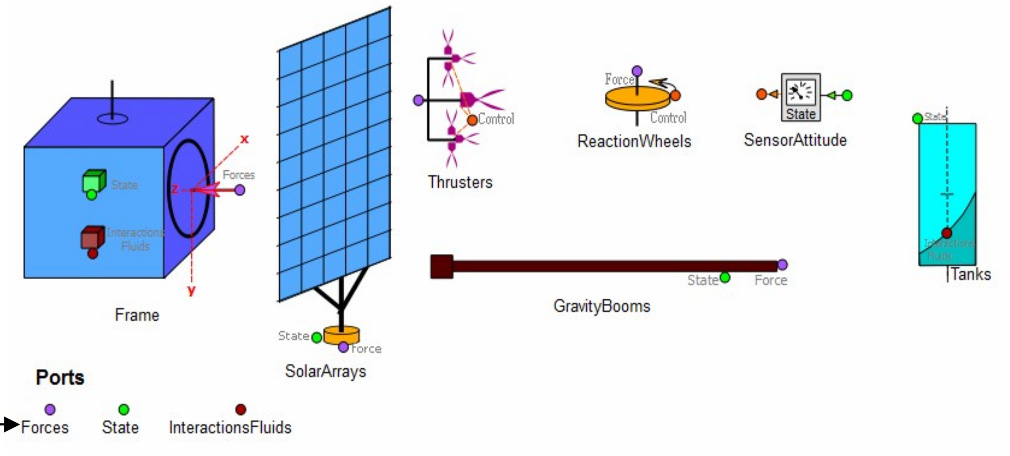
KopooS
Consulting Ind.

EA International

ESPSS* - SATELLITE library

*available only under ESA agreement

- Palette

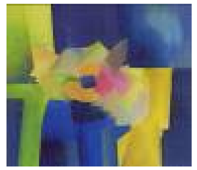


- 3 Ports
  - Port forces : multifunction port for inputs from a set of thrusters, RWs, SAs and gravity booms
    - port directions IN for the satellite frame, OUT for all other components.
    - type SUM in order to automatically account for all mass flow rates, forces, moment, angular momentum, power coming from all connected components.
  - Port State: multipurpose port for the attitude and orbit control and for 3D visualization, as well as the needed inputs for the solar arrays, gravity booms, Tanks
    - port directions OUT for the satellite frame, IN for all other components
  - Port "InteractionsFluids": to transfer the location of the free surface of the liquid, the Archimedes pressure function, the inertia matrix and the location of the fluids Centre Of Mass (COM) of from the Tanks component to the Frame

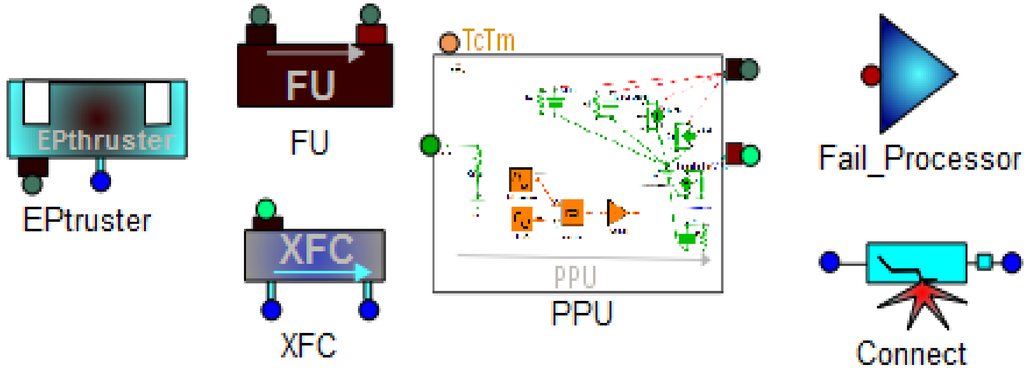
EA International

ESPSS* - EP library

*available only under ESA agreement

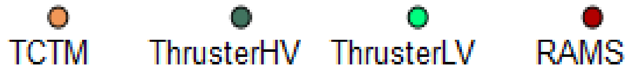


– **Palette**
  - Thruster
  - Filter unit
  - XFC
  - PPU
  - And for operation training
    - Fail processor
    - Leaking tube



Ports

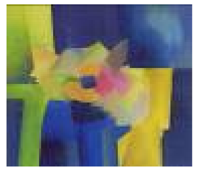
– **4 Ports** ⟶



– **Ports are the key of the** Object oriented system simulation**: like nodes for exchanging multiple variables between components: those are the variables needed as in the real system**

– **For EP library: 4 new ports have been designed**
  - TCTM for discrete and digital telecommands and telemetries
  - Thruster HV for the high voltage inputs/outputs
  - Thruster LV for the low voltage inputs/outputs
  - RAMS (for operations with a fail processor)

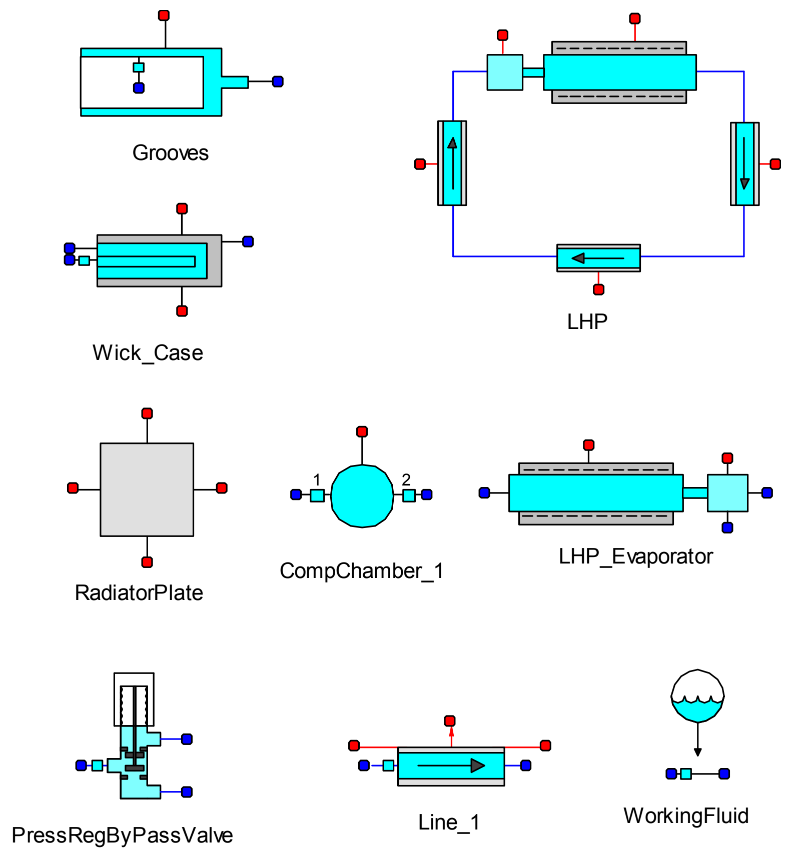
KopooS Consulting Ind.

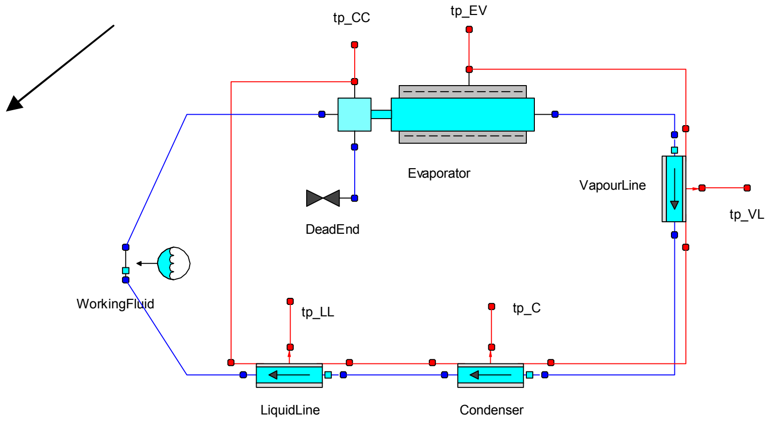


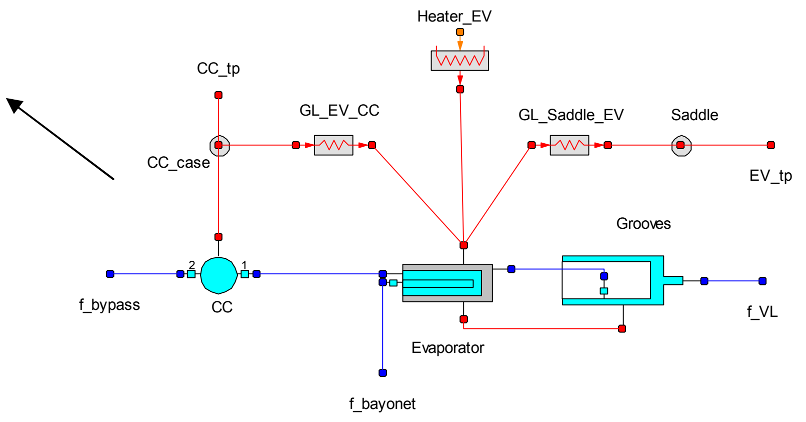
EcosimPro Overview - Libraries

– Loop heat Pipes (LHP) Devices

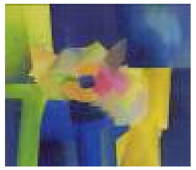
These are a topological components build as indicated here below:







EA International

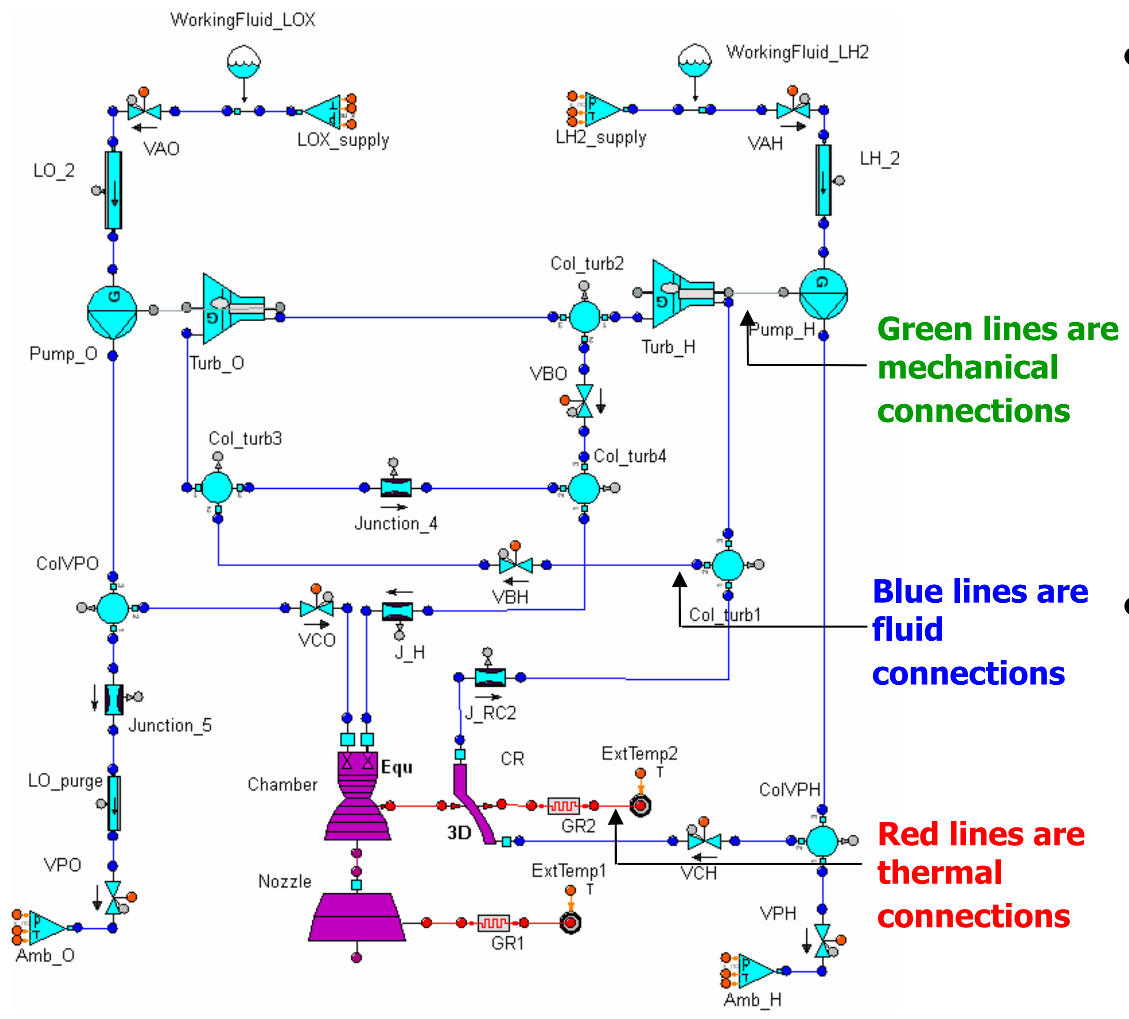


EcosimPro Overview - Components

- The user can define libraries of components using the EL language:
  - A component is the basic simulation unit representing some physical/logical behaviour not necessarily closed
  - Ecosimpro uses these components as symbols that can be **dragged & pasted graphically** to **build more complex models** by linking components' ports
- Components are reusable:
  - New components can be created by **inheritance and aggregation**
  - These **super-components** are automatically updated if the parents' formulation is changed
  - The component formulation --a set of non-necessarily ordered equations and events-- do not force a causality of the simulated problem

EA International

EcosimPro Overview - Model Builder



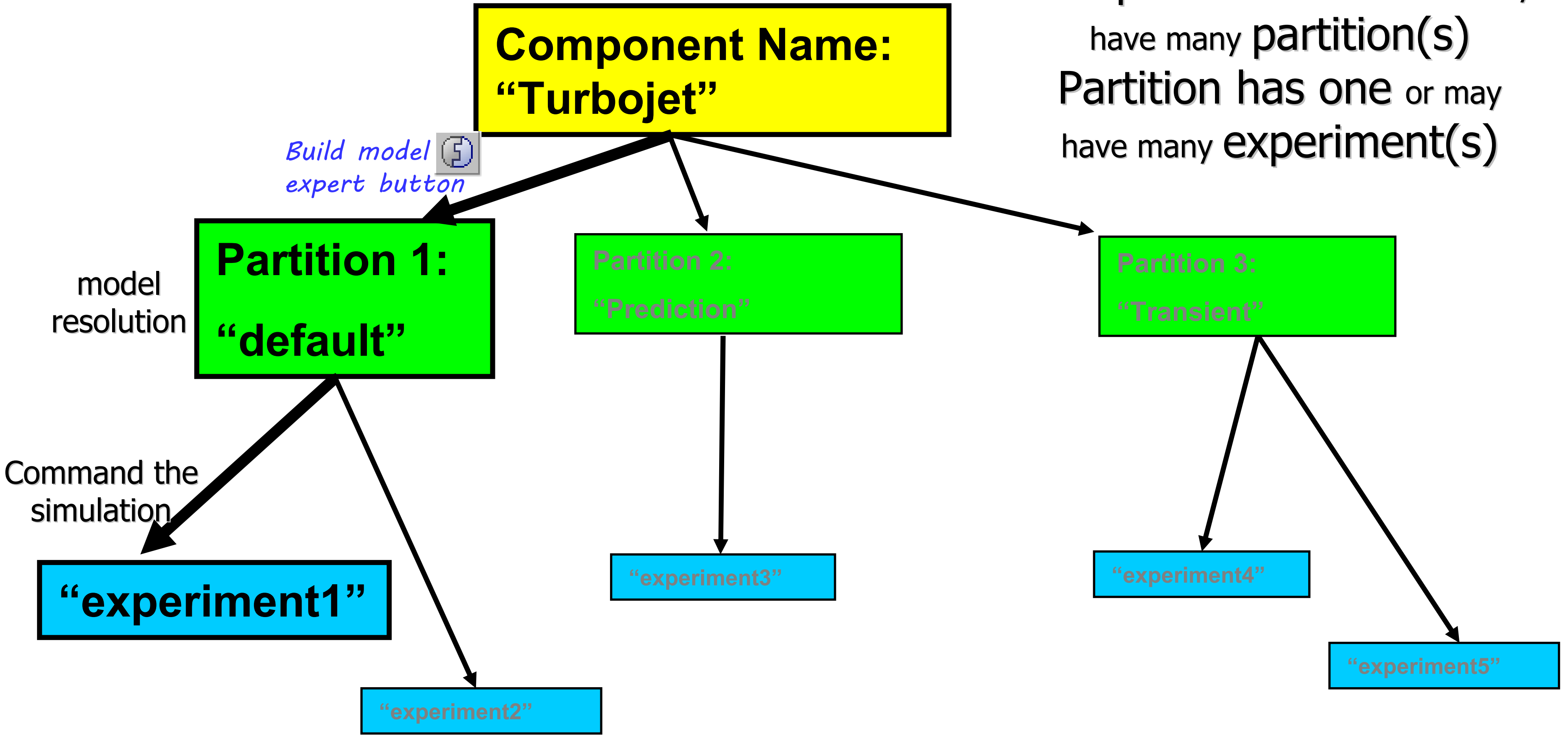
- The user creates models by picking components from the libraries and connecting them as they are connected in the actual physical system
- **Models take advantage to be multi-disciplinary. They can combine components of different libraries**

KopooS Consulting Ind.

EA International

EcosimPro Overview - Model Generation

Component has one or may have many partition(s)
Partition has one or may have many experiment(s)



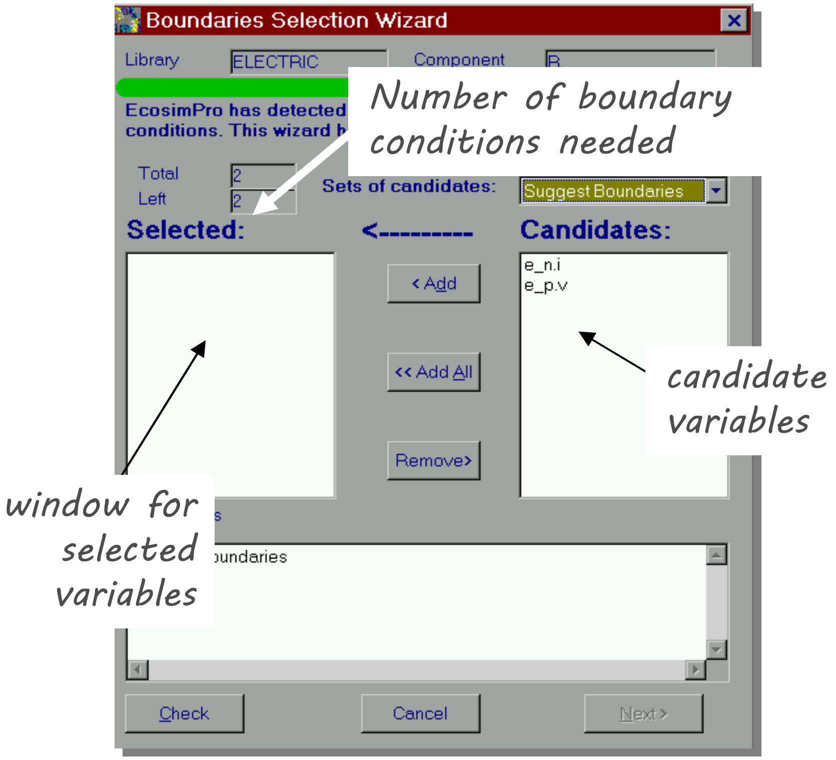
EA International

EcosimPro Overview - Model Generation for advanced users

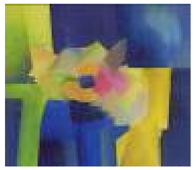
- The model generation can be controlled by an advanced and EXPERT user :
  - It's at the model generation phase (partitions) when EcosimPro will **re-order** all the components' equations.
  - EcosimPro GUI helps the advanced expert user with wizards selecting boundaries, tearing possible algebraic loops and solving high index problems

Given a system of differential algebraic equations, *f(x,x′, t)* the index of the system is defined as the number of times that it is necessary to derive all or part of *f* until it is converted to a system of differential equations.

Assistant for Boundary Conditions



KopooS Consulting Ind.



EcosimPro Overview – Kernel (I)

- EcosimPro is designed to solve **Differential and Algebraic <u>systems</u>** of Equations (DAEs). Like:

$$\boxed{\vec{f}\left(\vec{x},\ \frac{d\,\vec{x}}{dt},\ t\right) = \vec{0}}$$

$$m\,c\,\frac{dT}{dt} = h\,S\,(T_f - T) + \dot{W}_u$$

$$\frac{d^2 r}{dt^2} - r\left(\frac{d\theta}{dt}\right)^2 = \gamma$$

Even time and space derivative with 1-D models

$$\frac{\partial u}{\partial t} = D\frac{\partial^2 u}{\partial x^2} + Q(x,t)$$

- Ordinary Differential Equations and algebraic equations are special cases of DAEs:

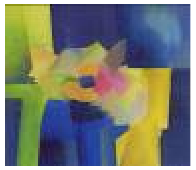
ODE

$$\frac{d\,\vec{x}}{dt} = \vec{f}\,(\vec{x}\,,\,t) \longrightarrow \vec{f}\,(\vec{x}\,,\,t) - \frac{d\,\vec{x}}{dt} = \vec{0}$$

Algebraic Equations

$$\vec{f}\,(\vec{x}\,,\,t) = \vec{0} \longrightarrow \vec{f}\,(\vec{x}\,,\vec{0},\ t) = \vec{0}$$

EA International



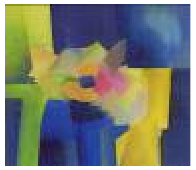
EcosimPro Overview - Kernel (II)

EcosimPro solvers:

- DASSL* is based on the Gear method for stiff problems (slow and fast dynamic together). Its main features are:
  - Automatic selection of the integration step and order
  - Integration order changes between 1 and 5
  - Use of Newton-Raphson to solve the implicit system of the equations
- EcosimPro provides as well a sparse version of DASSL to deal with big models (Jacobian size is very big)
- A classic fourth order Runge-Kutta is provided for solving ODE systems.

*DASSL: Differential Algebraic System Solver (Livermore Lab; L. Petzold)

EA International

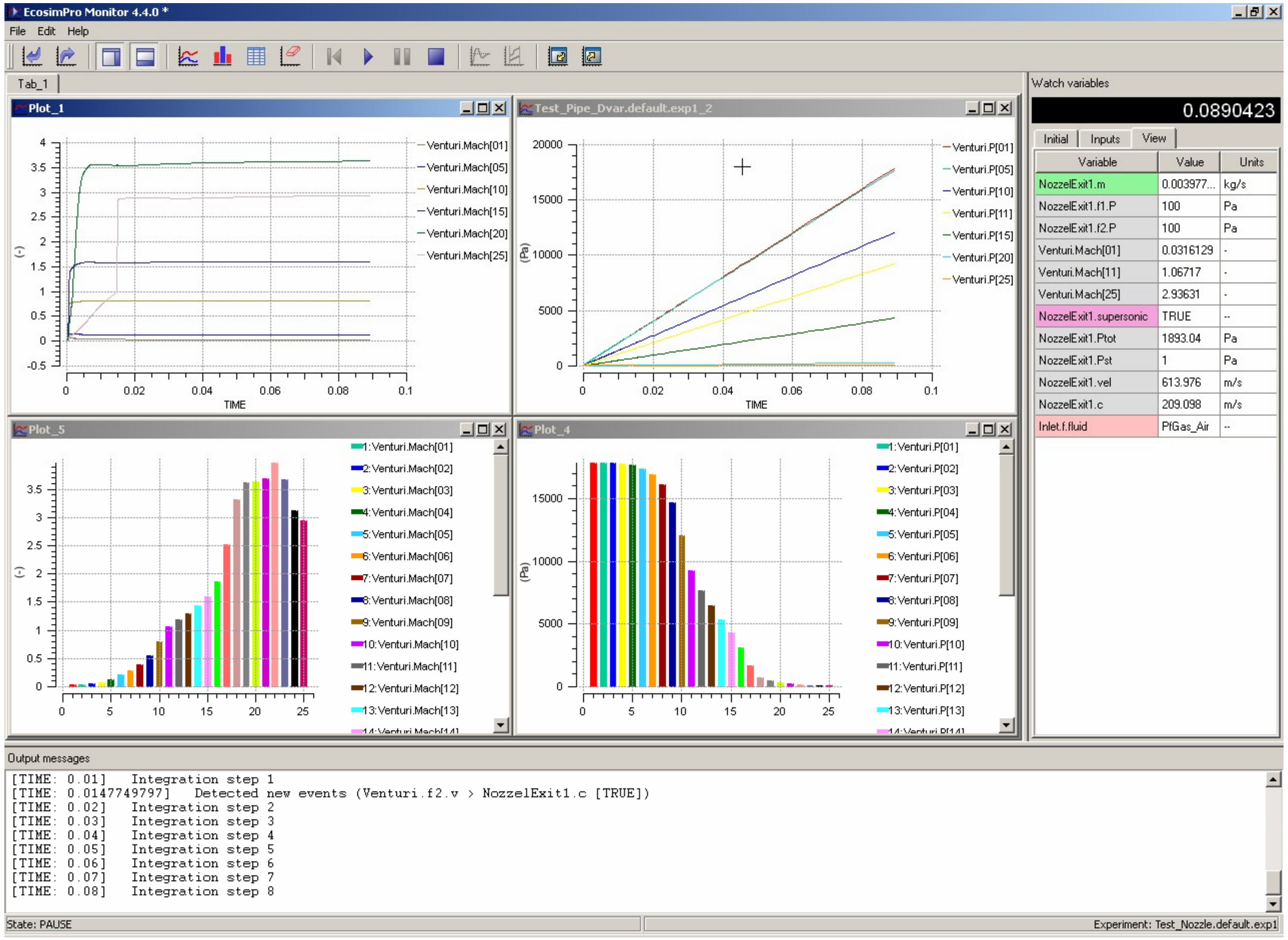


EcosimPro Overview - Monitoring

- There’s a experiment file where commands, data, boundaries and integration options can be set.
- The tool for interactive display of the experiments (simulation cases) is called Monitor. Variables can be monitored by various means including
  - Sliders & Thermometers
  - Gauges
  - Plots
- Simulation results can be exported to Microsoft Office applications for further analysis

KopooS Consulting Ind.

EcosimPro Overview - Monitor tool



Watch variables

0.0890423

Variable	Value	Units
NozzelExit1.m	0.003977...	kg/s
NozzelExit1.f1.P	100	Pa
NozzelExit1.f2.P	100	Pa
Venturi.Mach[01]	0.0316129	-
Venturi.Mach[11]	1.06717	-
Venturi.Mach[25]	2.93631	-
NozzelExit1.supersonic	TRUE	--
NozzelExit1.Ptot	1893.04	Pa
NozzelExit1.Pst	1	Pa
NozzelExit1.vel	613.976	m/s
NozzelExit1.c	209.098	m/s
Inlet.f.fluid	PfGas_Air	--

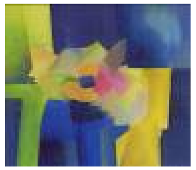
Output messages

```
[TIME: 0.01]    Integration step 1
[TIME: 0.0147749797]    Detected new events (Venturi.f2.v > NozzelExit1.c [TRUE])
[TIME: 0.02]    Integration step 2
[TIME: 0.03]    Integration step 3
[TIME: 0.04]    Integration step 4
[TIME: 0.05]    Integration step 5
[TIME: 0.06]    Integration step 6
[TIME: 0.07]    Integration step 7
[TIME: 0.08]    Integration step 8
```

State: PAUSE

Experiment: Test_Nozzle.default.exp1

EA International

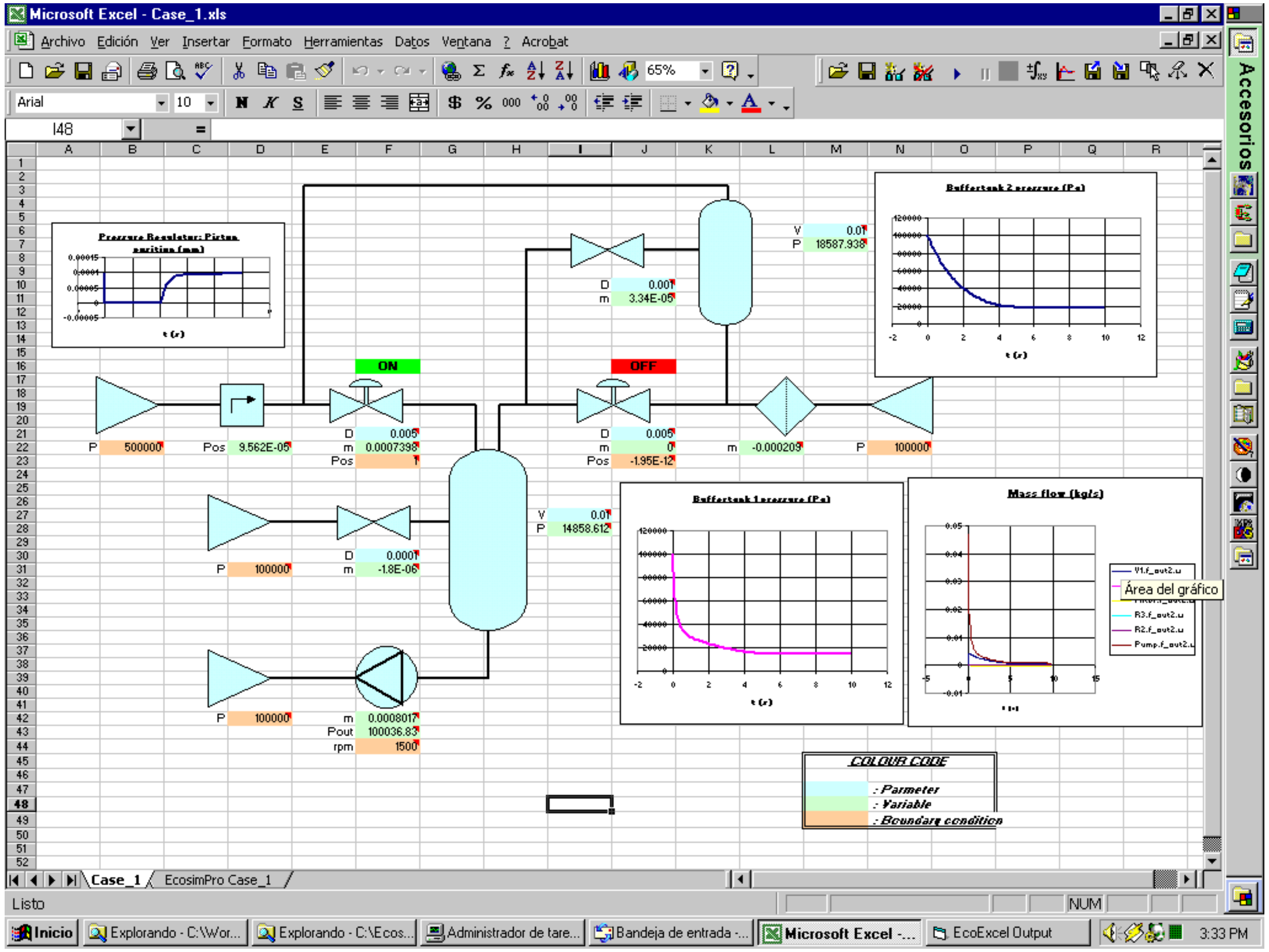


Connection to the External World

- EcosimPro generates C++ class with the final simulation code. This class can be used to include the model in other programs
- Executables of EcosimPro Models are also Active X servers. This enables to build experiments directly in Visual Basic
- An Excel Add-in is provided for easy connection to EcosimPro models from Excel
- It is possible to call Fortran, C and C++ functions from the equations (For example: property routines for H2O real fluid, etc.)

EA International

Connection to Excel


Microsoft Excel - Case_1.xls

Archivo Edición Ver Insertar Formato Herramientas Datos Ventana ? Acrobat

I48 =

ON

OFF

V 0.01
P 18587.938

D 0.001
m 3.34E-05

P 500000 Pos 9.562E-05 D 0.005 m 0.0007398 Pos 1

D 0.005 m 0 Pos -1.95E-12 m -0.000209 P 100000

V 0.01
P 14858.612

P 100000 D 0.0001 m -1.8E-06

P 100000 m 0.0008017 Pout 100036.83 rpm 1500

COLOUR CODE
: Parmeter
: Variable
: Boundary condition

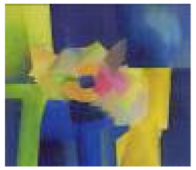
Case_1 / EcosimPro Case_1 /

Listo NUM

Inicio | Explorando - C:\Wor... | Explorando - C:\Ecos... | Administrador de tare... | Bandeja de entrada -... | Microsoft Excel -... | EcoExcel Output | 3:33 PM

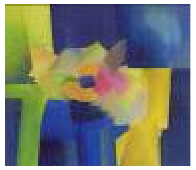
KopooS Consulting Ind.

EA International



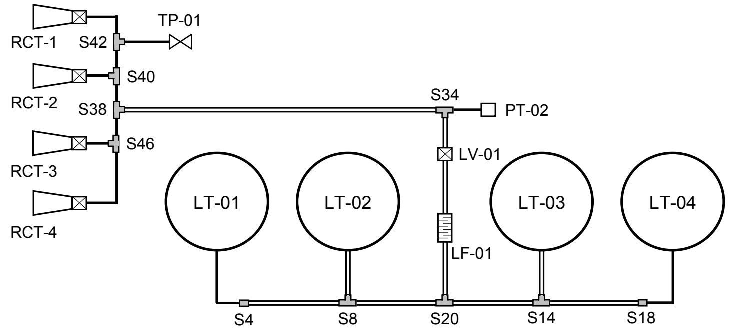
First Examples

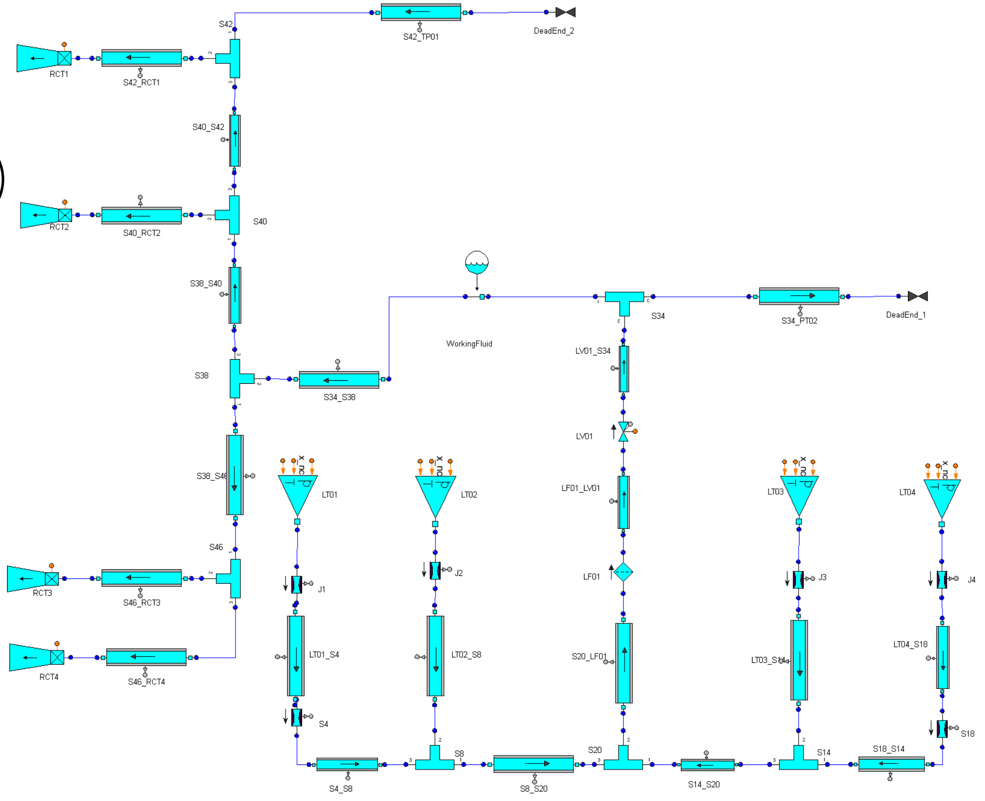
KopooS
Consulting Ind.



RCS* Case: Thrusters feeding system

This model is representative of a satellite propulsion system with hydrazine:

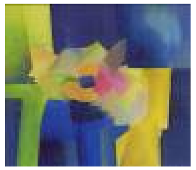




The EcosimPro schematic of the model appears in figure below, where the main feeding lines are included:

(To be explained interactively under EcosimPro environment)

*RCS: Reaction control system for satellites



A First Component in EL (I)

- Create a simple component to solve the next differential equation:

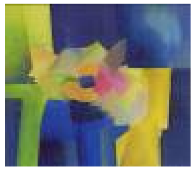
$$\frac{dy}{dt} = \frac{(x - y)}{tau}$$

- tau is a data (tau = 0.6 seconds)

- The variable x will be a boundary condition and it will be variable with time:

  x = sin(2 * TIME)

EA International

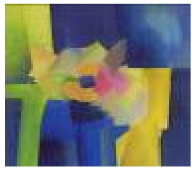


A First Component in EL (II)

```
-----------------------------------------------------------
--A first component in EcosimPro Language (EL)-------------
-----------------------------------------------------------
-- Component to solve a differential equation which is ----
-- used to introduce a delay to variable "x" --------------
-----------------------------------------------------------

COMPONENT equation

   DATA
        REAL tau = 0.6        "delay time (s)"

   DECLS
        REAL x, y

   CONTINUOUS
   -- Differential equation to introduce a delay to variable "x"
        y' = (x - y) / tau       --  y’ is the derivative dy/dt

END COMPONENT
```

KopooS Consulting Ind.

EA International



A First Component in EL (III)

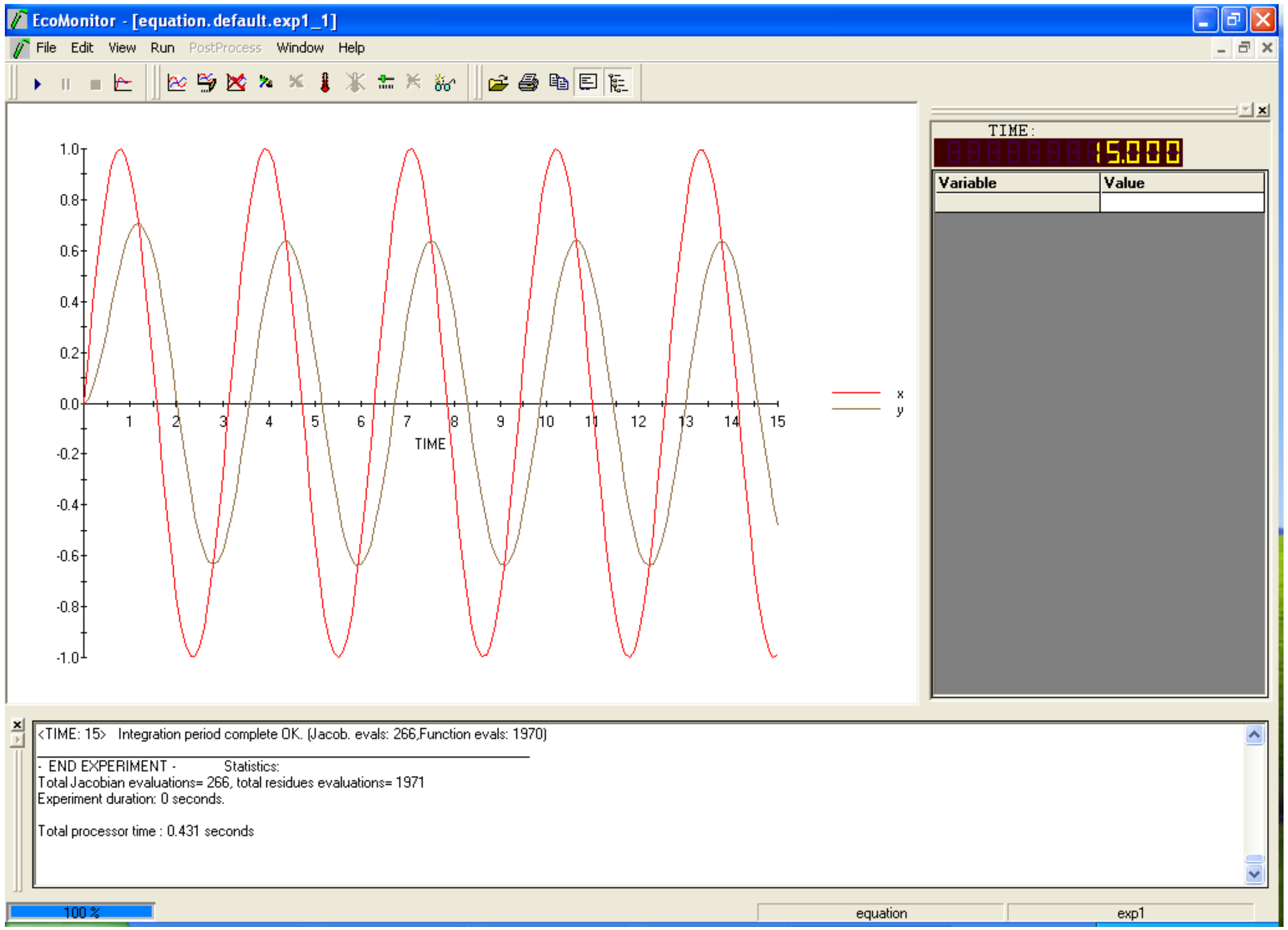
```
EXPERIMENT exp1 ON equation.default

DECLS

INIT      -- set initial values for variables
     -- Dynamic variables
   y = 0.

BOUNDS     -- set expressions for boundary variables: v = f(t,...)
   x = sin(2. * TIME)

BODY
   REPORT_TABLE("reportAll", " * ")

   TIME = 0.

   TSTOP = 15.
   CINT = 0.1
   INTEG()

END EXPERIMENT
```

Note: the boundary conditions can be static or dynamic (here function of TIME)

KopooS Consulting Ind.

EA International

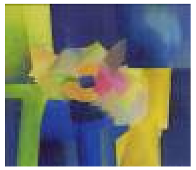
A First Component in EL (IV)



```
<TIME: 15>   Integration period complete OK. (Jacob. evals: 266,Function evals: 1970)

- END EXPERIMENT -          Statistics:
Total Jacobian evaluations= 266, total residues evaluations= 1971
Experiment duration: 0 seconds.

Total processor time : 0.431 seconds
```

KopooS Consulting Ind.

EA International



A Component using Schematic(I)

- Open the EcosimPro under schematic view
- Press button New schematic
- Choose a name under the COURSE_EXAMPLE Library
- Select the COURSE_THERMAL library

- Build by click and drag components the model shown in next figure. Following tips are useful:

KopooS Consulting Ind.

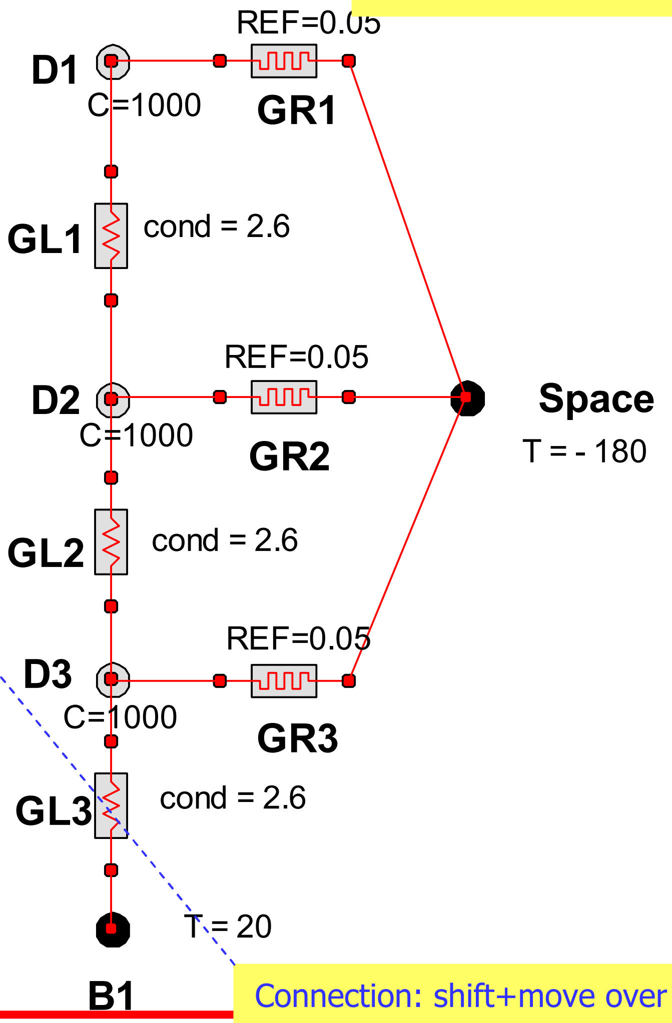
EA International

A Component using Schematic (II)

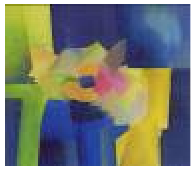
Size: right, Shape, dragg

Label: shift+move over it

- Arranging components:
  - To change the size of a component, select the component, right button, select "component shape option" and change the size by dragging the symbol's corners
  - To change the position of the component's name, press the SHIFT key at the same time as you move the mouse pointer over the labels, and drag it
- Draw connectors between the components like in the figure above. A tooltip will appear whenever the mouse runs over a port, displaying the information of that port:
  - Select the connection button on the right-hand toolbar or press the SHIFT key at the same time as you move the mouse pointer over a port
  - Left-click on the port to be connected. (Click the various points of the schematic drawing where the connector is required to run, if any). Left-click the target port, which must be of the same type as the origin port



Connection: shift+move over port



A Component using Schematic (III)

Once we have built the schematics of our thermal network, let’s do the following steps:

- Generate build model by pressing the “F” button This action implies:
  - generation of the EL source code from the graphic information
  - Compilation of the EL file
  - Generation of the default partition (**ordering** and **resolution** of all the components’ **equations**) in a C++ class
- Under Simulation view, generate a new experiment and simulate it with the monitor tool

EA International

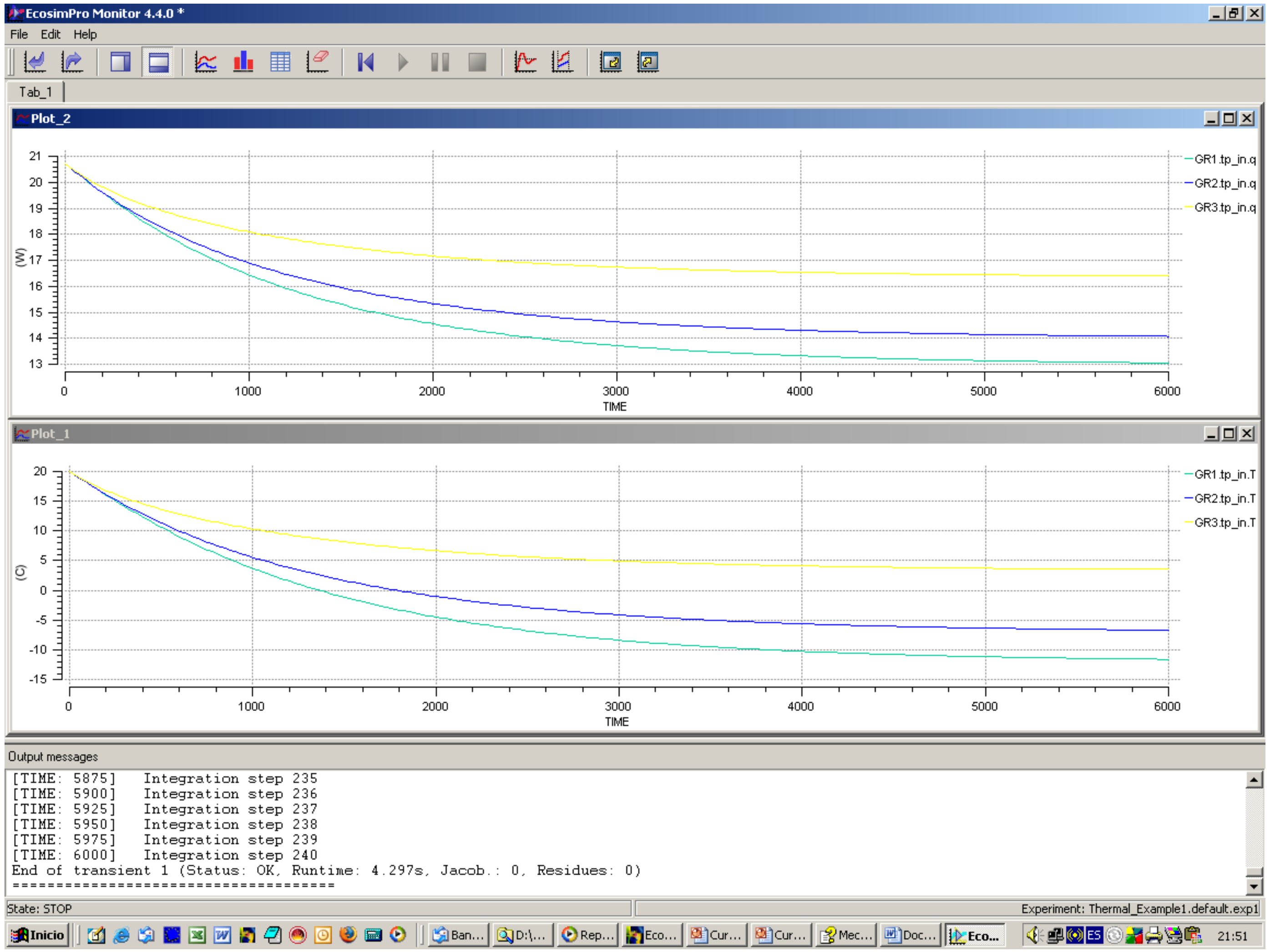
A Component using Schematic (IV)

```
EXPERIMENT exp1 ON ThermalNetwork1.default
DECLS
INIT     -- set initial values for variables
      -- Dynamic variables
   D1.T = 20
   D2.T = 20
   D3.T = 20

BOUNDS    -- set expressions for boundary variables: v = f(t,...)

BODY
   REPORT_TABLE("reportAll", " * ")
   TIME = 0
   TSTOP = 6000
   CINT = 25
   INTEG()
END EXPERIMENT
```

KopooS Consulting Ind.

EA International

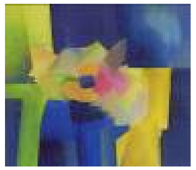
A Component using Schematic (V)



```
[TIME: 5875]   Integration step 235
[TIME: 5900]   Integration step 236
[TIME: 5925]   Integration step 237
[TIME: 5950]   Integration step 238
[TIME: 5975]   Integration step 239
[TIME: 6000]   Integration step 240
End of transient 1 (Status: OK, Runtime: 4.297s, Jacob.: 0, Residues: 0)
=====================================
```

KopooS Consulting Ind.

EA International

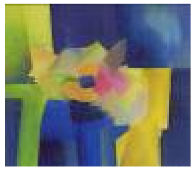


Summary

- EcosimPro is a user-friendly SYSTEM simulation tool for modelling simple and complex systems with **physical processes** expressing the behaviours in terms of :
  - equations,
  - differential equations
  - algebraic equation (DAEs)
  - discrete events
- It provides a powerful Graphical User Interface (GUI) to build systems models based on reusing existing libraries
- We can distinguish several types of users of EcosimPro:
  1. Component-Library Developer
  2. System Model Builder from pre-existing libraries
  3. System Users that exploit pre-existing models

...

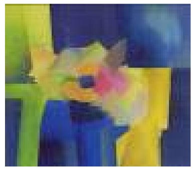
KopooS Consulting Ind.



To go further

...

EA International



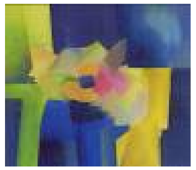
To go further

To go further

- ECOSIMPRO LANGUAGE (EL) INTRODUCTION
  - Statements, Ports, Components, Equations, Library, Experiments
- BUILDING A THERMAL NETWORK LIBRARY
- DISCONTINUITIES
  - Statements: When, After (delay), Assert, Zone
- HANDLING ARRAYS into COMPONENTS
- Expand (equations), Sum
  - ARRAYS OF COMPONENTS AND ARRAYS OF CONNECTIONS
  - VECTORIZED PORTS FOR HEAT TRANSFER
- MATHEMATICAL PROCESS
- USING ECOSIM MODELS FROM OTHER TOOLS
- FUNCTIONS

KopooS Consulting Ind.

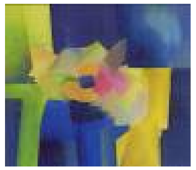
EA International



ECOSIMPRO LANGUAGE (EL) INTRODUCTION

KopooS Consulting Ind.

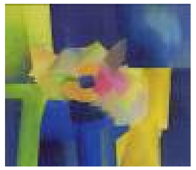
EA International



INDEX 1/3

KopooS Consulting Ind.

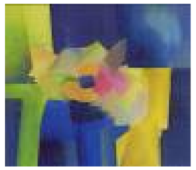
EA International



INDEX 2/3

KopooS Consulting Ind.

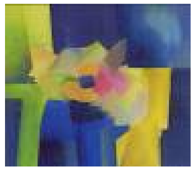
EA International



INDEX 3/3

KopooS Consulting Ind.

EA International



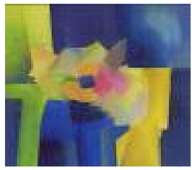
EL Introduction: Statements

EcosimPro provides 4 kinds of statements:

- **Sequential statements like IF, WHILE, FOR, etc.** The order of the statements is fundamental**. Supported in Fortran, Java, C++**
- **Continuous statements like differential or algebraic equations.** The order is indifferent**. Used to express the dynamic behaviour of the dynamic model.**
- **Discrete statements (or events) like WHEN.** The order is theoretically indifferent**. Used to express the discrete behaviour of the dynamic model.**
- **Directive statements like USE (for using components from other library), EXPL (explicit), IMPL (implicit), BOUND (boundary), CONST (constant), etc**

KopooS Consulting Ind.

EA International



EL Introduction: PORT entity

♦ **EcosimPro has a fundamental** interface **entity for connecting together components: the PORT.**

♦ **A PORT encapsulates the variables that represent the actual physical exchange among components.**

♦**Ports allow intelligent behavior to be defined when connecting to other ports. EcosimPro will automatically introduce additional equations when the mathematical model is generated.**

```
-- Fluid port type
PORT Fluid
     SUM    REAL      m          "mass flow (kg/s)"
     EQUAL  REAL      P          "pressure (Pa)"
            REAL      rho        "density (kg/m3)"

END PORT
```

Note: ports prefixes SUM or EQUAL … are used for describing the behaviour in case of multiple connections

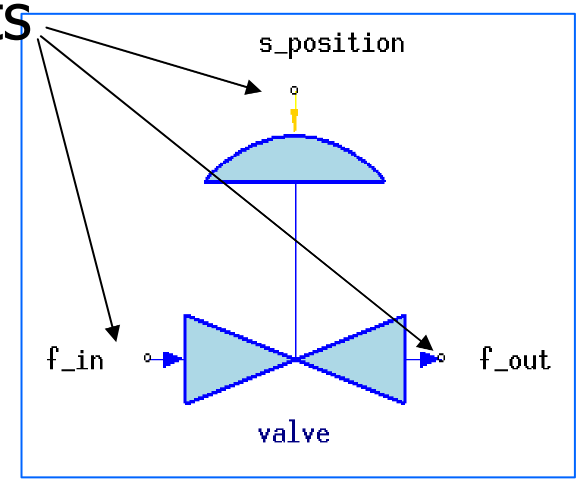
KopooS Consulting Ind.

EL Introduction: COMPONENT entity (I)

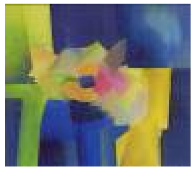
- **Components**
  - They are the elementary blocks to build the models
  - There is a language to define new components

- **Components are defined by:**
  - declaration of the ports
  - declaration of data & variables (valve area, pressure difference,….)
  - equations that represent the behaviour

Ports



$$\boxed{q_m = C_d A \sqrt{2 \rho \Delta P}}$$

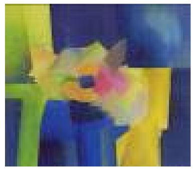


EL Introduction: COMPONENT entity (II)

```
COMPONENT Valve
   PORTS
        IN    Fluid   f_in          --Fluid inlet
        OUT Fluid   f_out         --Fluid outlet     (Algebraic ports variables)
        IN    Signal position   --Position signal
   DATA
        REAL Cd, A                  --Cd at full open position of the A area
   DECLS
        REAL dP                     --Pressure difference
        REAL m                      --Mass flow
   CONTINUOUS
        dP = f_in.P - f_out.P
        m  = Cd*A * position.signal * sqrt(2* f_in.rho*dP )
        f_in.m = m
        f_in.m = f_out.m
END COMPONENT
```

Note: Here typical equations between ports variables: f_in… f_out…

EA International



EL Introduction: COMPONENT entity (III)

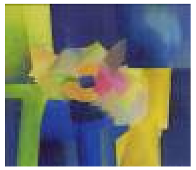
•**Public** parts of the component:

1. Construction **Parameters** (e.g. number of nodes)
2. **PORTS** (interfaces)
3. **DATA** (e.g. user data: diameter, length, etc)

•**Private** parts (not accessible from outside) of the **component**:

1. **DECLS** (e.g. local variables)
2. **TOPOLOGY** (e.g. relations between ports, components and their interconnection)
3. **INIT** (e.g. initial values when needed)
4. **DISCRETE** (e.g. actions in case of events)
5. **CONTINUOUS** (e.g. equations)

KopooS Consulting Ind.



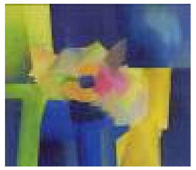
EL Introduction: Equations

- **Symbolic** Manipulation
  - It is important to recognize that the "=" operator does not represent in **EcosimPro** an assignment. For example, lets consider the following set of 2 equations:

    x = TIME
    x = 4 * y
  - In a **Procedural** (sequential) language, the first statement would **<u>assign</u>** a value to x, and the second would overwrite the value of x with a new value. In **EcosimPro**, the '=' represents **<u>equation</u>**, and the previous set will be re-arranged by EcosimPro into the following set of assignments:

    x = TIME
    y = x / 4
- The equations do not assume any causality.
  - It is possible to exchange known and unknown variables to solve design problems

EA International



EL Introduction: Library functions

- **The user can define its own functions in EL and then call them from any component or port**

```
FUNCTION REAL square(REAL x)
BODY
      RETURN x * x
END FUNCTION
```

- **The user can reuse existing FORTRAN, C and C++ functions by pre-declaring them before**

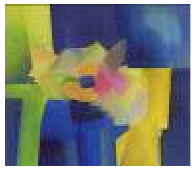
```
"FORTRAN" FUNCTION REAL square(REAL x)
```

- **Then, at component or port level codification we can use the function:** `x = square(y)`
- **At compilation time the user needs to specify the object file to link with**

EL Introduction: Experiments (I)

- **The EL allows any sequential statement in the experiment, for example:**
  - Statements re-defining input data
  - FOR, IF statements programming parametric studies
  - Call to external functions, etc
- **Some of specific experiment functions are:**
  - INTEG(), INTEG_TO, INTEG_CINT, INTEG_STEP for integration of a model between an initial time (TIME) and a final time (TSTOP), with a communication interval (CINT)
  - STEADY() for calculation of steady states
  - REPORT_MODE, REPORT_TABLE, REPORT_LIST, WRITE statements, etc
  - Interpolation functions defining boundaries. See example here after
  - ABS_ERROR, REL_ERROR control the effective integration time step

EA International

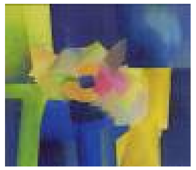


EL Introduction: Experiments (II)

```
EXPERIMENT exp1 ON model.default

DECLS
    TABLE_1D Pup =  {{0, 0.5, 1, 1.5, 2.}, {1, 1000, 1000, 1, 1}}
    REAL Pini = 1e2

INIT    -- set initial values for variables
   -- Dynamic variables

   -- Algebraic variables

BOUNDS    -- set expressions for boundary variables: v = f(t,...)
   Inlet.s_pres.signal[1] = Pini * timeTableInterp(TIME,Pup)

BODY
  REPORT_TABLE("reportAll", " * ")
  TIME = 0.
  TSTOP = 15
  CINT = 0.1

  REPORT_MODE = IS_STEP
  INTEG()

END EXPERIMENT
```

Note: the boundary condition is function of TIME

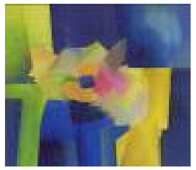
EA International



BUILDING A THERMAL NETWORK LIBRARY

KopooS Consulting Ind.

EA International



Preparation for the Practical Exercises

- Extract all the contents of the file curso.zip to this (or other) directory: C:\EcosimPro\USER_LIBS
- Start EcosimPro
- Create New existing Libraries named:
  - COURSE_EXAMPLES
  - COURSE_THERMAL

KopooS Consulting Ind.

Thermal Library (I)

- A Library for thermal network models is similar to an Electrical library
  - Heat flux plays a similar role to Current and
  - Temperature plays a similar role to Voltage
  - Diffusive nodes are similar to capacitors
  - Thermal conductors are very similar to resistors
- Open the file **Thermal.el** in Library COURSE_THERMAL

Thermal Library (II)

```
CONST REAL STEFAN = 5.6696e-8 "Stefan constant (W/m^2/K^4)"
CONST REAL TZERO  = 273.15    "Shift of Centigrade zero"

PORT Thermal
    SUM   REAL q    "Heat Flux (W)"       --additive heats
    EQUAL REAL T    "Temperature (C) "    --Temperatures are equal
END PORT

ABSTRACT COMPONENT Node
    PORTS
        IN Thermal tp_in
    DATA
        REAL qi = 0.  "impressed heat flux (W)"
    DECLS
        REAL q_total
CONTINUOUS
        q_total = tp_in.q + qi
END COMPONENT
```

IN & OUT prefixes in the ports provide the sign of the port variables of type prefixed SUM

Thermal Library (III)

```
COMPONENT DNode IS_A Node
    DATA
        REAL C = 1  "Thermal capacity (J / kg K)"
    DECLS
        REAL T  = 20
    CONTINUOUS
        C * T'  = q_total  -- differential equation
        tp_in.T = T
END COMPONENT

COMPONENT BNode IS_A Node  --Boundary Node
    DATA
        REAL T
    CONTINUOUS
        tp_in.T = T
END COMPONENT
```

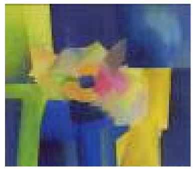
EA International

Thermal Library (IV)

```
ABSTRACT COMPONENT Conductor
     PORTS
        IN Thermal tp_in
        OUT Thermal tp_out
CONTINUOUS
        tp_in.q = tp_out.q
END COMPONENT

COMPONENT GL IS_A Conductor  --Linear conductor
    DATA
        REAL cond "Thermal conductance (W/C)"
    CONTINUOUS
        tp_in.q = cond * (tp_in.T - tp_out.T)
END COMPONENT
```

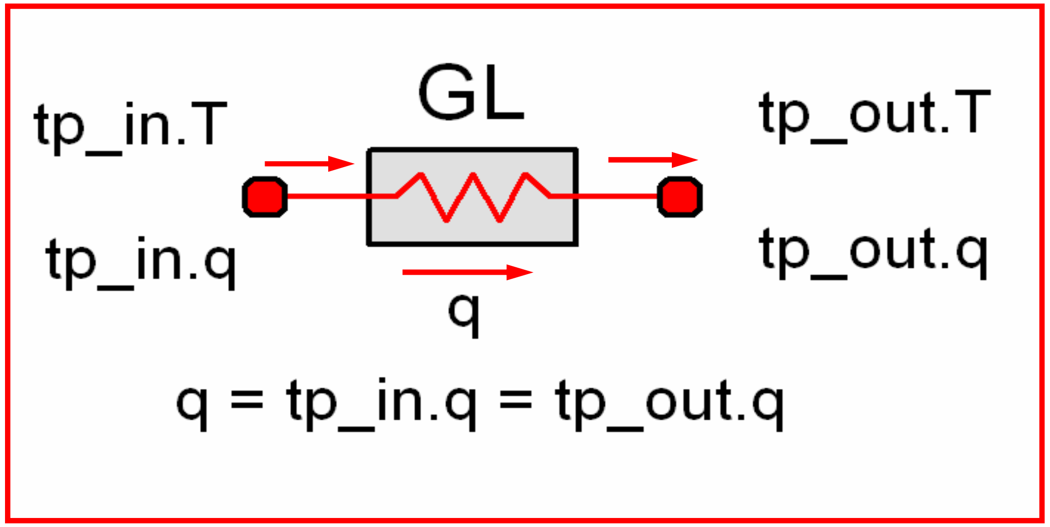
Thermal Library (V)

```
COMPONENT GR IS_A Conductor
    DATA
        REAL REF = 0. “Radiative Exchange Factor (m^2)”
    CONTINUOUS
        tp_in.q = STEFAN * REF * ((tp_in.T + TZERO)**4 - \
                                  (tp_out.T + TZERO)**4)
END COMPONENT
```

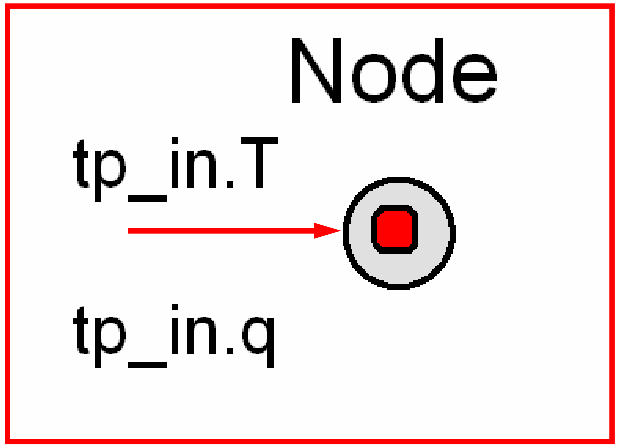


EA International

Sign Criteria in Thermal Library

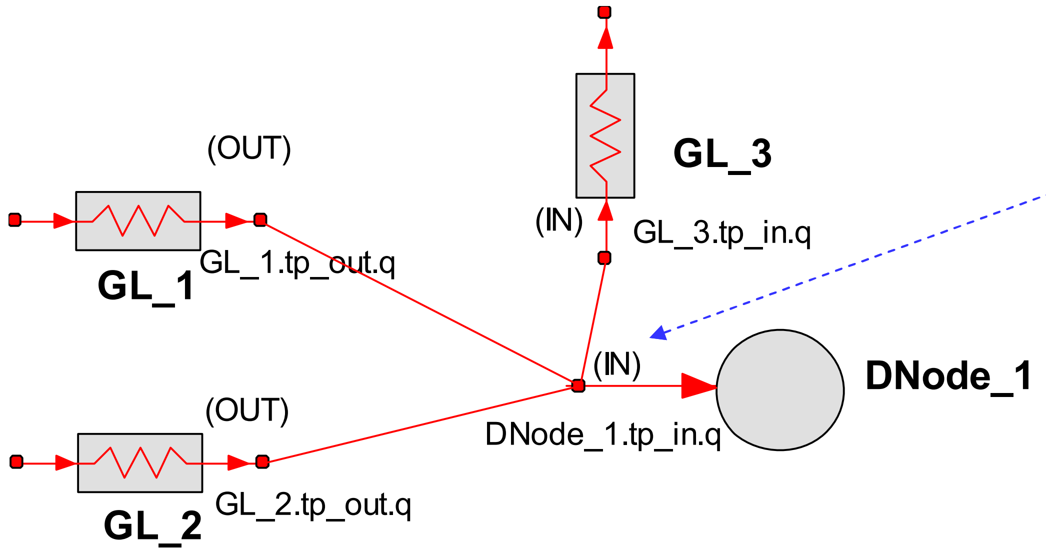


$$q = tp\_in.q = tp\_out.q$$



EA International

PORT Declarations - SUM & EQUAL



IN & OUT prefixes in the ports provide the sign of the port variables of type prefixe SUM

Rule for the port variable prefixed SUM with multiple connections:

`+GL_1.tp_out.q +GL_2.tp_out.q -GL_3.tp_in.q -Dnode_1.tp_in.q = 0`

Rule for the EQUAL prefix:

`GL_1.tp_out.T  = GL_2.tp_out.T = GL_3.tp_in.T  = Dnode_1.tp_in.T`

Exercise 1 - Creation of a New Component Diffusive Node with T dependant capacity

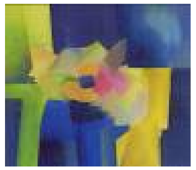
- **Create a new file and define a new component, named DnodeT, representing a diffusive thermal node with temperature dependant capacity, based on the abstract component Node implementing the following equations (where a, b, c are data):**

$$\boxed{C = a + b \cdot T + c \cdot T^2}$$

Tip: You can copy the DNode and modify it. Now C is not a datum, it is a variable

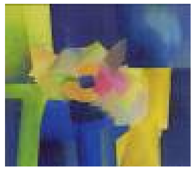
EA International

Exercise 1 - Creation of a New Component Diffusive Node with T dependant capacity



- **Save the file in Library** COURSE_THERMAL **and compile it, then select the Library folder in the Library Window, and look at the tree.**
- **Recompile the file** Thermal.el **and see what happens with the component** DnodeT **in the Library Window**
- **Update the diffusive thermal node** DnodeT **component by hand**

KopooS Consulting Ind.



WaveForms or **Forcing** Functions

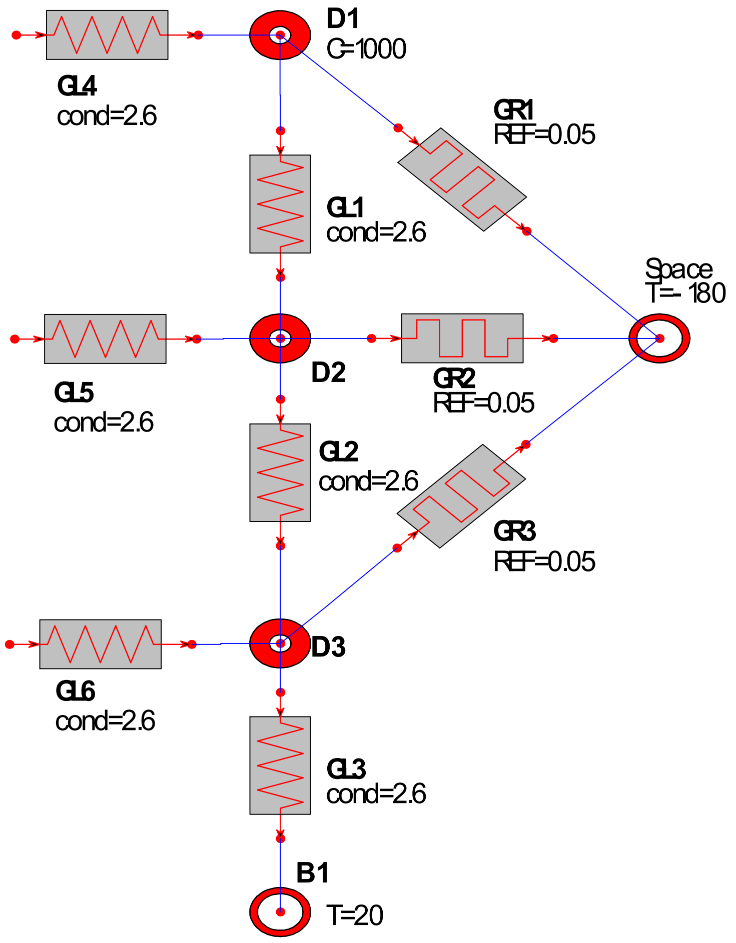
- Purpose: Define typical **boundary conditions** and waveforms
- Specific Features:
  - Automatic generation of discontinuities

```
square(TIME, REAL period)
step(TIME, REAL Tstart)
pulse(TIME, REAL period, REAL width)
ramp(TIME, REAL period)
timeTableInterp(TIME, TABLE_1D table)
timeTableStep(TIME, TABLE_1D table)
periodTimeTableInterp(TIME, TABLE_1D table)
periodTimeTableStep(TIME, TABLE_1D table, REAL period)

Also accepted e.g.:
timeTableInterp(TIME-100, TABLE_1D table)
```

EA International

Exercise 2 - Boundary Conditions (I)

- **Open the file Thermal_Network2.eds in Library COURSE_EXAMPLES, that represents the following model**
- **Generate model (default partition)**
- **Edit the partition to see the boundaries, mathematical order, etc. Modify it to get the following boundary conditions:**
  - **GL4.tp_in.T**
  - **GL5.tp_in.T**
  - **GL6.tp_in.q**
- **Generate an experiment**



EA International

Exercise 2- Boundary Conditions (II)

- **Define the following boundary conditions in the experiment:**

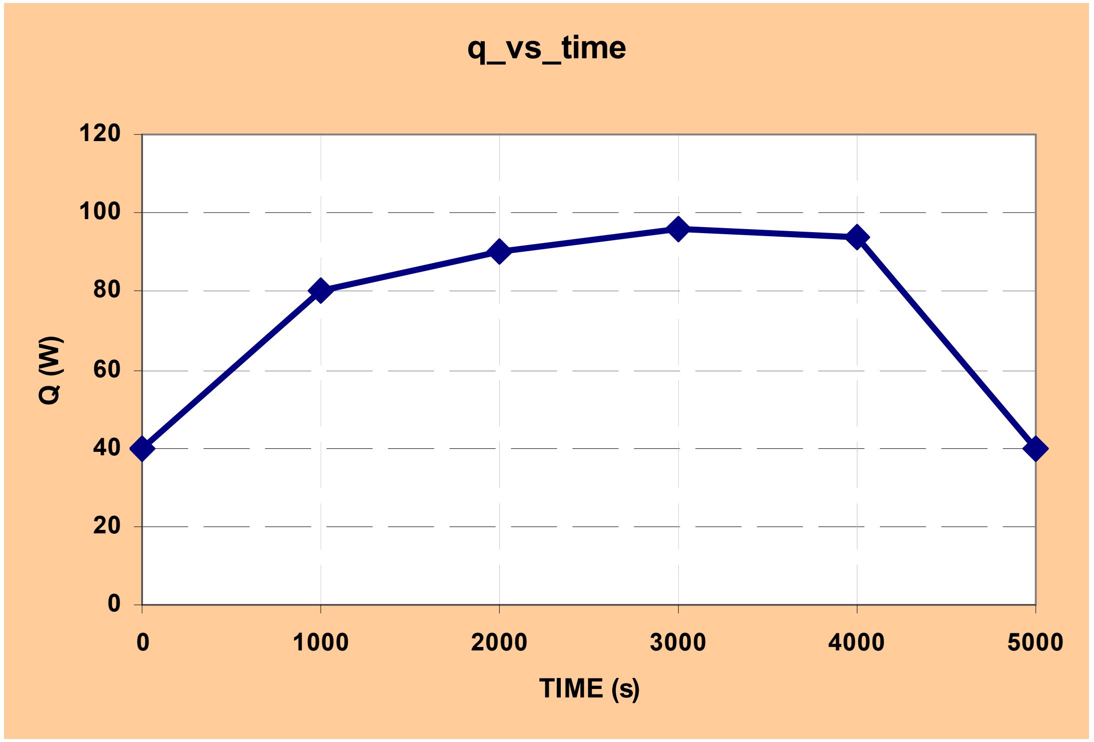
```
DECLS
     TABLE_1D q_vs_time  = \
             {{ 0., 1000.,  2000., 3000., 4000., 5000.},
              {40.,   80.,    90.,   96.,   94.,   60.}}
INIT       -- set initial values for variables
           -- Dynamic variables
    ...
BOUNDS     -- set expressions for boundary : v = f(t,...)
    GL4.tp_in.T = 20 + 30 * step(TIME, 100)
    GL5.tp_in.T = 20 + 10 * pulse(TIME, 1000, 250)
    GL6.tp_in.q = timeTableInterp(TIME, q_vs_time)
```

- **Execute the experiment from 0 to 10000 s, and plot the temperatures in the nodes and the boundary conditions**

EA International

Exercise 2 - Table 1D Detail

```
TABLE_1D q_vs_time  = \
        {{ 0., 1000.,  2000., 3000., 4000., 5000.},
         {40.,   80.,    90.,   96.,   94.,   60.}}
```

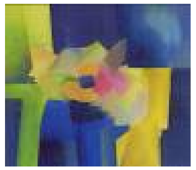


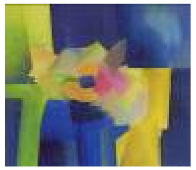
KopooS Consulting Ind.

Exercise 2 (III)

- **Modify the previous experiment to test the following waveforms, and setting TSTOP** = 20000

```
square(TIME, 500)
ramp(TIME-1, 3)
timeTableStep(TIME, q_vs_time)
periodTimeTableInterp(TIME, q_vs_time, 5000.)
periodTimeTableStep(TIME, q_vs_time, 5000.)
```





Design Partitions

For Advanced users:

- A data is a value that it is known and it remains constant during the integration. The design partition enables to convert a data into an unknown.
- Reasons for converting data into unknowns

1) Design Problems

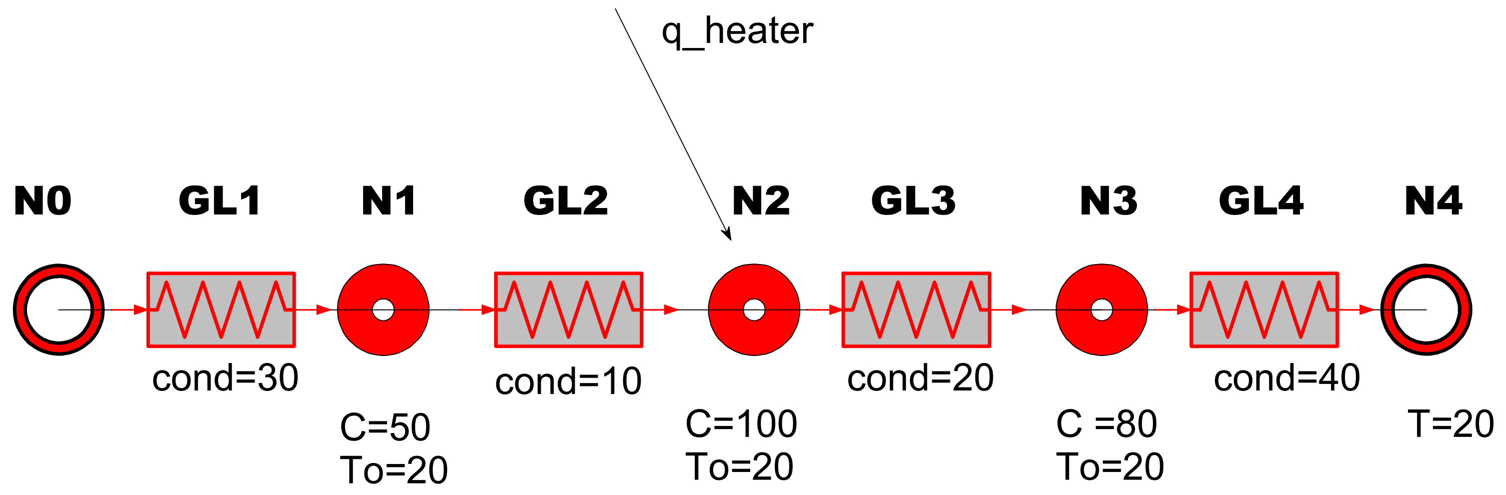
– Sometimes, there are problems where the unknown is the value of a datum in order to obtain a given performance.

2) Data is not constant

– Sometimes, the user wants to change the value of a datum as a function of time. For example, the diameter of a pipe is a value that it is typically constant, but the user might want to simulate the diameter decrease that occurs as a consequence of fouling or depositions inside.

EA International

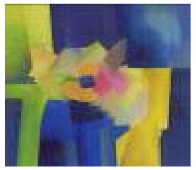
Exercise 3 (I)



q_heater a pulse at time 2 seconds with total energy 50000 W and lasting 1e-6 seconds

$$\Delta T = \frac{50000 \text{ W}}{100 \text{ W / K}} = 500 \text{ K}$$

EA International



Exercise 3 (II)

- **Open and compile the model `ThermalNetwork3.eds` that represents the previous Thermal Network**
- **Go under Code view. Define a design partition and enable `N2.qi` to be a variable, and then select it as boundary condition**
- **Generate a new Experiment and define the following value for the Boundary Condition**
- **Note that executing an experiment can be done in several views**

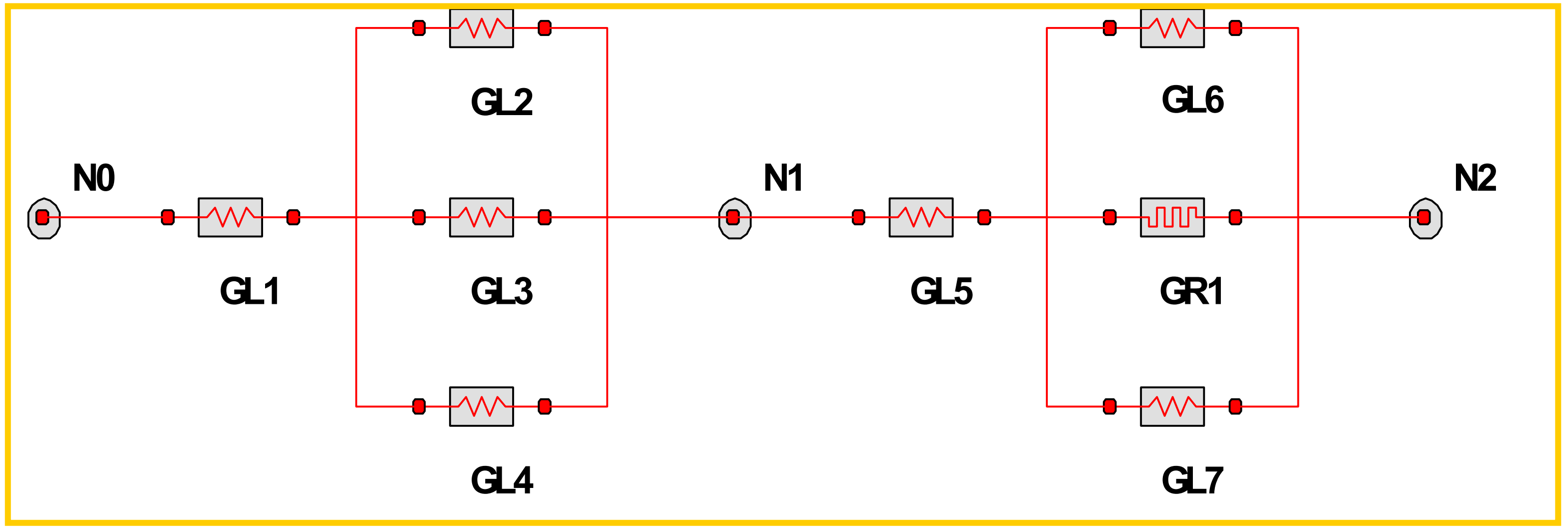
```
N2.qi = 5e4 * pulse(TIME-2, 1.e20, 1e-6)  --very long period! 1µs pulse
```

- Execute the experiment and plot `N0.T, N1.T, N2.T, N3.T, N4.T & N2.qi`

KopooS Consulting Ind.

EA International

Exercise 4 (I)

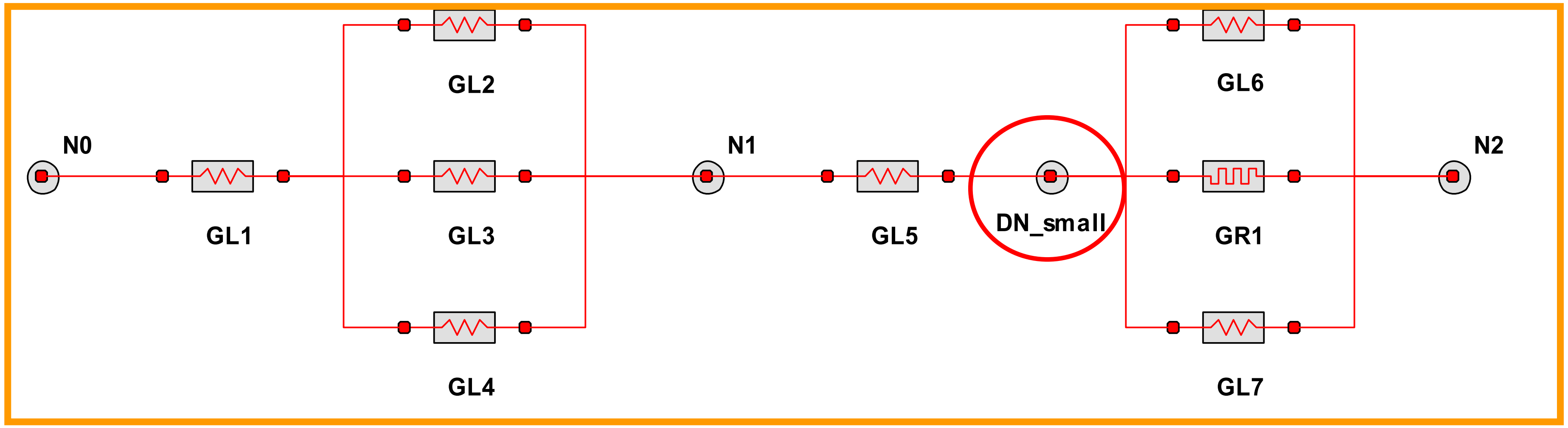


- Open file ThermalNetwork4.eds in COURSE_EXAMPLES
- Make a model with the default partition
- Simulate it with TSTOP=1000 & CINT=10
- Look at the Math Model (Partition > Mathematical View)

KopooS Consulting Ind.

EA International

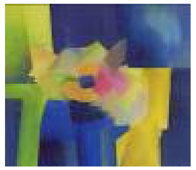
Exercise 4 (II)



- Modify ThermalNetwork4 model to avoid the algebraic loop adding a diffusive node with small capacitance (**C = 1e-6 W/K**)
- Rename the model, e.g. ThermalNetwork4_NAL
- Make a partition and Look at the Mathematical View
- Simulate it with TSTOP=1000 & CINT=10
- Compare the results with those of ThermalNetwork4

KopooS Consulting Ind.

EA International



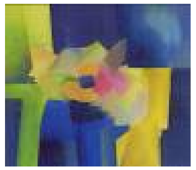
DISCONTINUITIES

KopooS
Consulting Ind.

Discontinuities

- There are four ways of defining discontinuities in the behaviour of a model.
  - **Discrete** part: WHEN Statement, AFTER Suffix
    - To define an action that it is executed when an event occurs or at a given time (e.g. replacement Valve = ON)
    - Time delayed Event or Assignation after the current local time e.g. x=5 AFTER 0.3 --seconds
  - **Continuous** part: ZONE Statement
    - It defines alternative equations
    - This can increase the complexity of the model: sometime it is better to manage those conditions into specific functions…
  - Waveform Functions

EA International

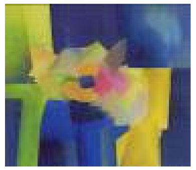


WHEN Statement

```
COMPONENT WhenExample
     DATA
         REAL Tmin = 20
         REAL Tmax = 50.
     DECLS
         REAL HeaterPower
         REAL T = 10. -- T Temperature
     DISCRETE
         WHEN (T < Tmin) THEN
             HeaterPower = 50.
         END WHEN
         WHEN (T > Tmax) THEN
             HeaterPower = 0.
         END WHEN
      CONTINUOUS
          T' = 0.1 * (HeaterPower - 10) -- Here a differential equation
                                         -- T' is used for dT/dTIME
END COMPONENT
```

KopooS Consulting Ind.

EA International



Delayed Assignation

```
COMPONENT WhenExample
     DATA
         REAL Tmin = 20
         REAL Tmax = 50.
     DECLS
         DISCR REAL HeaterPower
         REAL T = 10. -- T Temperature
     DISCRETE
         WHEN (T < Tmin) THEN
             HeaterPower = 50. AFTER 5
         END WHEN
         WHEN (T > Tmax) THEN
             HeaterPower = 0.  AFTER 2
         END WHEN
      CONTINUOUS
          T' = 0.1 * (HeaterPower - 10) -- T' = dT/dTIME

END COMPONENT
```

KopooS Consulting Ind.

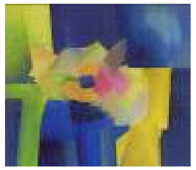
EA International

Bouncing Ball Problem

```
COMPONENT BouncingBall
    DATA
        REAL g = 9.806
        REAL k = 0.8 --two dashes begin a comment
    DECLS
        REAL y
    DISCRETE
        WHEN (y < 0) THEN
            y' = - k * y'
        END WHEN
    CONTINUOUS
        y'' = -g        -- Here a second order differential equation
END COMPONENT
```

KopooS Consulting Ind.

EA International



ASSERT Statement

The purpose of the ASSERT statement is to check the validity of the math model

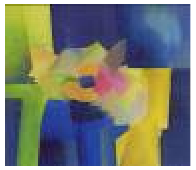
```
COMPONENT BouncingBall
    DATA
        REAL g = 9.806
        REAL k = 0.8
    DECLS
        REAL y
    DISCRETE
        WHEN (y < 0) THEN
            y' = - k * y'
        END WHEN
        ASSERT (y > -1e-3)
            FATAL "Ball going below floor"
    CONTINUOUS
        y'' = -g    -- y'' is used for d²y/dTIME²

END COMPONENT
```

KopooS Consulting Ind.

ZONE Statement

```
--Limitation of  a variable
COMPONENT Limits_0
    DECLS
        REAL x
        REAL xmax
        REAL xmin
        REAL y

    CONTINUOUS
        xmax = 0.5 + 0.2 * sin(TIME)
        xmin = -0.5 - 0.2 * sin(2 * TIME)
        x = sin(3*TIME)
        y  = ZONE (x > xmax ) xmax
             ZONE (x < xmin ) xmin
             OTHERS   x
END COMPONENT
```

EA International

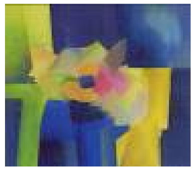


1-D or Multi-Dimensions discretisation

HANDLING ARRAYS into COMPONENTS

KopooS Consulting Ind.

EA International



Handling Arrays - EXPAND & SUM

- **EXPAND**

It generates multiple equations with an array index changing in a given range. For example,

```
EXPAND (j IN 2,4) T[j] = alpha[j-1] **2
```

generates the three following equations

```
T[2] = alpha[1]**2
T[3] = alpha[2]**2
T[4] = alpha[3]**2
```

- **SUM**

It generates a summation of elements in a given range. For example

```
v = SUM (k iN 2,4; x[k] * alpha[2*k])
```

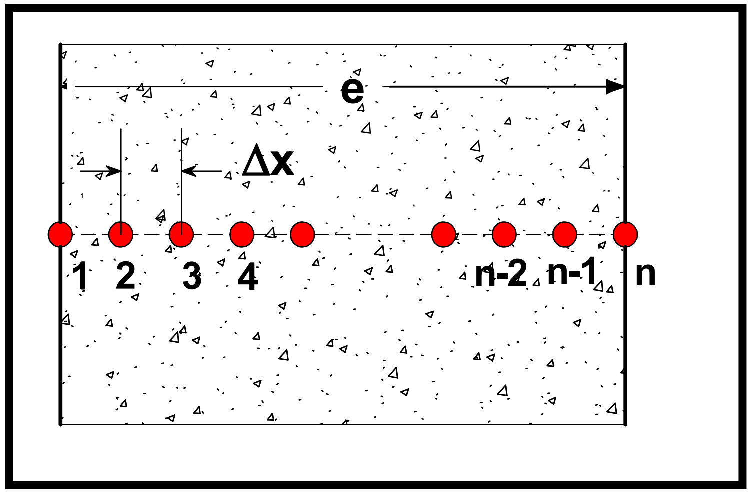
generates the following equation:

```
v = x[2]*alpha[4] + x[3]*alpha[6] + x[4]*alpha[8]
```

*Note: **SUM** can be also used in Experiments.*

EA International

Handling ARRAYS - Example: Thermal Wall (I)



$$\frac{\partial T}{\partial t} = \alpha \frac{\partial^2 T}{\partial x^2}$$

$$q(x=0) = -k \left. \frac{\partial T}{\partial x} \right|_{x=0}$$

$$q(x=e) = -k \left. \frac{\partial T}{\partial x} \right|_{x=e}$$

Method of Lines

- To discretise the space domain
- To substitute the partial space derivatives by finite differences
- To substitute the partial derivatives with respect time by total derivatives

$$\frac{dT_i}{dt} = \alpha \frac{\partial^2 T}{\partial x^2} = \alpha \left(\frac{T_{i-1} - 2T_i + T_{i+1}}{\Delta x^2} \right) + O(\Delta x^2)$$

$$q(x=0) = -\frac{k(T_2 - T_1)}{\Delta x} + O(\Delta x)$$

$$q(x=e) = -\frac{k(T_n - T_{n-1})}{\Delta x} + O(\Delta x)$$

KopooS Consulting Ind.

Handling ARRAYS - Example: Thermal Wall (II)

Here example of a construction **parameter**: nodes

```
COMPONENT Wall (INTEGER nodes=10) --nodes shall be >=2
    PORTS
        IN  Thermal tp_in
        OUT Thermal tp_out
    DATA
        REAL A    = 1.      "Area (m^2)"
        REAL e    = 0.1     "Thickness (m)"
        REAL cond = 10.     "Thermal conductivity
        REAL rho  = 1000.   "Density (kg/m^3)"
        REAL cp   = 2000.   "Specific Heat (J/kg C)"
    DECLS
        REAL alpha
        REAL dx                --Node thickness
        REAL T[nodes]          --Temperatures
    TOPOLOGY
        PATH tp_in TO tp_out
        --…/…
```

EA International

Handling ARRAYS - Example: Thermal Wall (III)

```
        CONTINUOUS
            dx = e / (nodes - 1) --nodes shall be >=2 else error here!
            alpha = cond / (rho * cp)

            EXPAND (i IN 2, nodes-1)
               T[i]' = alpha * (T[i-1] - 2*T[i] + T[i+1]) / dx**2

            tp_in.q = A * cond * (T[1] - T[2]) /  dx

            tp_out.q = A * cond * (T[nodes-1] - T[nodes]) / dx

            tp_in.T = T[1]
            tp_out.T = T[nodes]

    END COMPONENT
```

KopooS Consulting Ind.

EA International

Handling ARRAYS - Wall Simulation (I)

- Open file Wall.el and in Library COURSE_THERMAL
- Compile file Wall.el
- Simulate the thermal wall component with (choose the appropriate partition for it):
  - Boundary conditions

    ```
    tp_in.T = 100
    tp_out.q = 0
    ```

  - Initial Conditions (all nodes at 20 C)

    ```
    T[1] = 20
    T[2] = 20
    T[3] = 20
    …
    ```

- Find a suitable value for TSTOP (just enough to reach steady state) and a suitable value for the communication interval

Handling ARRAYS - Wall Simulation (II)

- Introduce the following modifications in the wall component to facilitate the initialisation of temperatures

```
DATA
    ...
    REAL To    = 20.      --Initial Temperature
DECLS
    REAL alpha
    REAL dx               --Node thickness
    REAL T[nodes]          --Temperatures

TOPOLOGY
    PATH tp_in TO tp_out
INIT
    FOR (i IN 2, nodes-1)
        T[i] = To
    END FOR
CONTINUOUS
```

Note: The initialisation made in the INIT block of the component overrides the initialisation made in the INIT block of the experiment

EA International

Model of a Wall with 3 layers (I)

Based on the wall model, a model of wall with three different layers can be built (file **Wall_3.el** in **COURSE_EXAMPLES**)

```
USE COURSE_THERMAL
COMPONENT Wall_3
     TOPOLOGY
         Wall (nodes=3) layer1 (cond = 0.1, e=0.005)
         Wall (nodes=5) layer2 (cond = 5, e= 0.098)
         Wall (nodes=3) layer3 (cond = 0.1, e= 0.005)
         CONNECT layer1 TO layer2 TO layer3
END COMPONENT
```

Notes:

Wall (nodes=3) layer3 (cond = 0.1, e= 0.005)

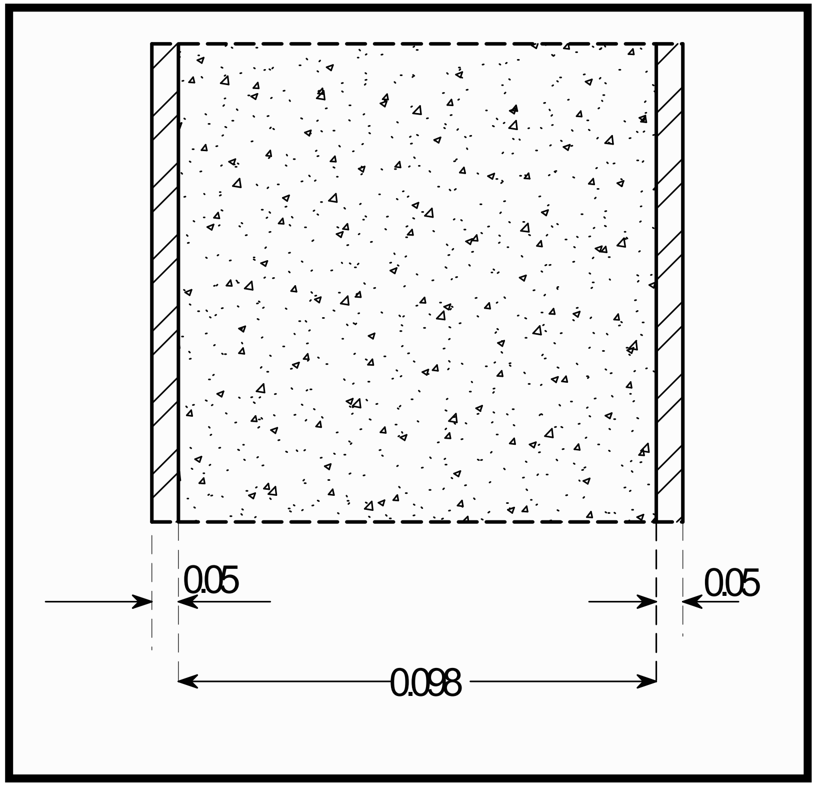
Construction **Parameters** like the number of nodes are assigned values just after the name of the component **type** (here **Wall**)

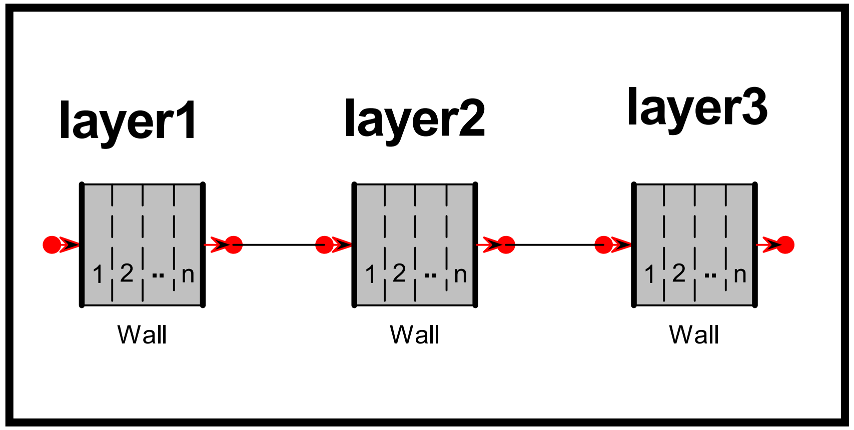
While the **Data** are assigned after name of the component **name** (here **layer3**)

KopooS Consulting Ind.

EA International

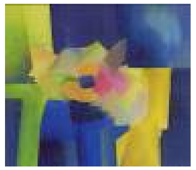
Model of a Wall with 3 layers (II)





KopooS Consulting Ind.

EA International



HANDLING ARRAYS into COMPONENTS

Option 1-ARRAYS OF COMPONENTS AND ARRAYS OF CONNECTIONS

KopooS Consulting Ind.

EA International

Arrays of components

- The EcosimPro language enables to define arrays of components having equal data

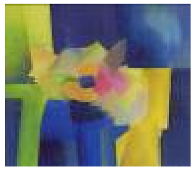
```
COURSE_THERMAL.DNode D[nodes] (C = 100)
```

- The language also enables to expand connect statements between individual components of an array

```
 EXPAND_BLOCK (i IN 1, nodes-1)
     CONNECT D[i].tp_in TO GL[i].tp_in
     CONNECT  GL[i].tp_out TO D[i+1].tp_in
END EXPAND_BLOCK
```

KopooS Consulting Ind.

EA International



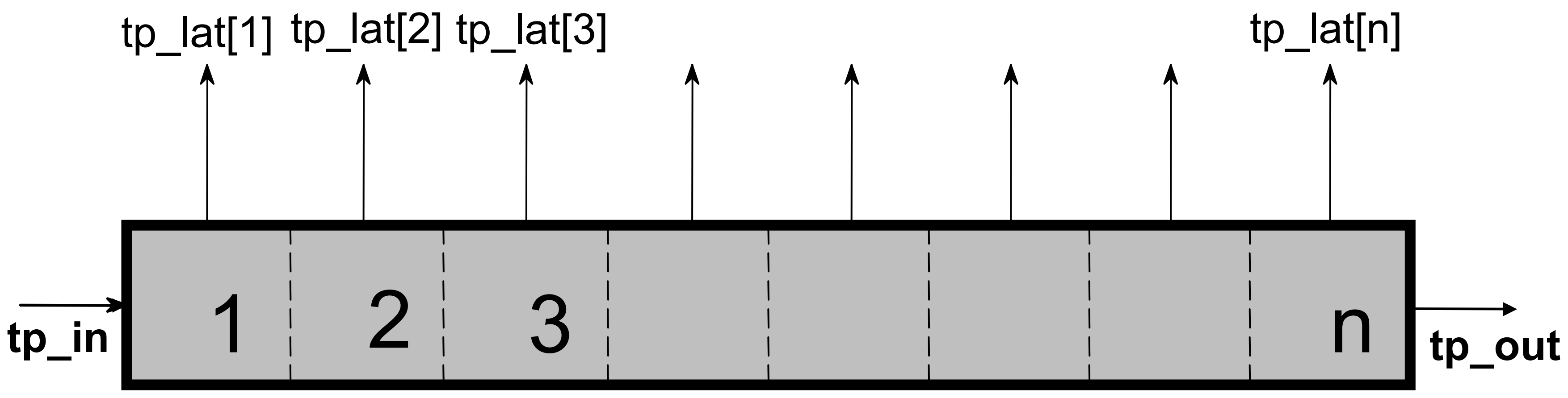
Alternative coding of Wall Example (I)

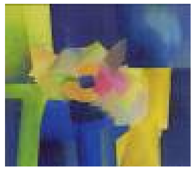
```
COMPONENT Wall_MC (INTEGER nodes=8)
    PORTS
        IN Thermal tp_in
        OUT Thermal tp_out
    DATA
        REAL A    = 1.       "Area (m^2)"
        REAL e    = 0.1      "Thickness (m)"
        REAL cond = 10.      "Thermal conductivity (W/m °C)"
        REAL rho  = 1000.    "Density (kg/m^3)"
        REAL cp   = 2000.    "Specific Heat (J/kg °C)"

    TOPOLOGY
        COURSE_THERMAL.DNode D[nodes] (C = rho*cp*A*e / nodes)
        COURSE_THERMAL.GL GL[nodes-1] (cond=cond*A/ (e/nodes))
        COURSE_THERMAL.GL GL_in(cond = 2*cond*A / (e/nodes))
        COURSE_THERMAL.GL GL_out(cond = 2*cond*A / (e/nodes))
        EXPAND_BLOCK (i IN 1, nodes-1)
           CONNECT D[i].tp_in TO GL[i].tp_in
           CONNECT  GL[i].tp_out TO D[i+1].tp_in
        END EXPAND_BLOCK
        CONNECT GL_in.tp_out TO D[1].tp_in
        CONNECT D[nodes].tp_in TO GL_out.tp_in
        --External connections
        CONNECT tp_in TO GL_in.tp_in
        CONNECT GL_out.tp_out TO tp_out
END COMPONENT
```

KopooS Consulting Ind.

Port Arrays

- Definition of a component to represent a fin based on the wall component





Option 2-HANDLING ARRAYS into COMPONENTS

VECTORIZED PORTS FOR HEAT TRANSFER

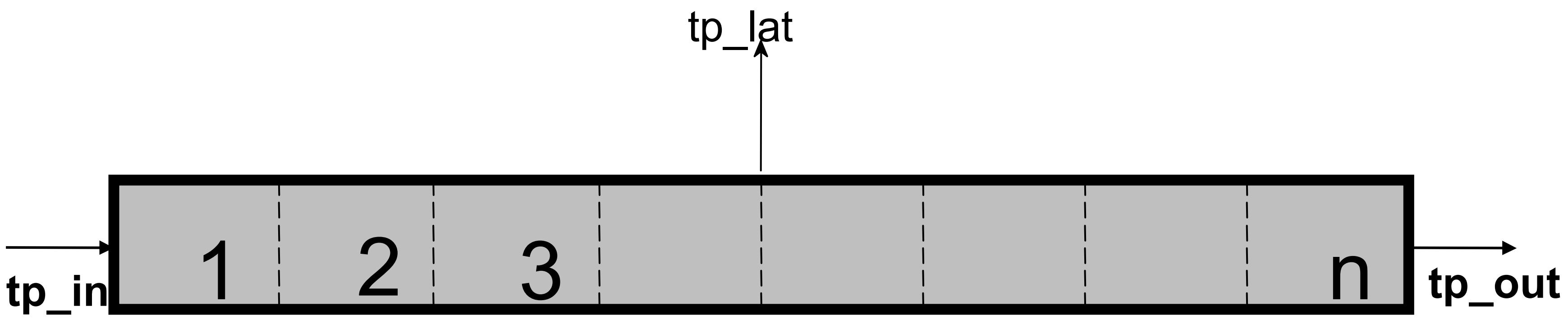
➔ solution for 1-D or Multi-Dimensions discretisation

EA International

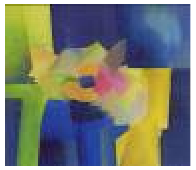
Array Type Variables in Ports

- Definition of a component to represent a fin based on the wall component using a **port with array type variables**

```
PORT Thermal (INTEGER nodes = 1 "Number of nodes in array (-)")
     SUM   REAL q[nodes]                        "Heat Flux (W)"
     EQUAL REAL T[nodes]                        "Temperature (C)"
END PORT
```



EA International



Thermal Library 2 (I)

- The differences between the two versions of the thermal network libraries (COURSE_THERMAL and COURSE_THERMAL2) are the following:
  - The port of the library COURSE_THERMAL2 is an **array of temperatures and heat fluxes**
  - The library COURSE_THERMAL2 is supported by the EcosimPro graphical tool
  - The GLs and GRs are arrays of conductors of the same characteristics

KopooS Consulting Ind.

Thermal Library 2 (II)

```
CONST REAL STEFAN = 5.6696e-8 "Stefan constant    (W/m^2 K^4)"
CONST REAL TZERO  = 273.15    "Shift of Centigrade zero (°C)"

PORT Thermal (INTEGER nodes = 1 "Number of nodes in array (-)" )
    SUM   REAL q[nodes]          "Heat Flux (W)"
    EQUAL REAL T[nodes]          "Temperature (°C)"
END PORT
ABSTRACT COMPONENT Node
   (
   INTEGER N = 1    "Dimension of thermal port (-)"
   )
    PORTS
        IN Thermal (nodes = N) tp_in
    DATA
        REAL qi = 0.  "impressed heat (W)"
    DECLS
        REAL q_total
    CONTINUOUS
        q_total = qi + SUM(i IN 1,N; tp_in.q[i])
END COMPONENT
COMPONENT DNode IS_A Node
    DATA
        REAL C = 1   "Thermal capacity (J / kg °C)"
    DECLS
        REAL T  = 20 "Node temperature (°C)"
    CONTINUOUS
        C * T'  = q_total
        EXPAND (i IN 1,N)
            tp_in.T[i] = T
END COMPONENT
```

Thermal Library 2 (III)

```
COMPONENT BNode IS_A Node  --
    Boundary Node
     DATA
          REAL T
     CONTINUOUS
          EXPAND (i IN 1,N)
           tp_in.T[i] = T
END COMPONENT

ABSTRACT COMPONENT Conductor
    (
    INTEGER N = 1            "Dimension
    of thermal port (-)"
    )
      PORTS
       IN Thermal (nodes = N) tp_in
       OUT Thermal (nodes = N) tp_out
      CONTINUOUS
          tp_in.q = tp_out.q
END COMPONENT
```

```
COMPONENT GL IS_A Conductor  --Linear
    conductor
  DATA
        REAL cond      "Conductance (W/°C)"
  CONTINUOUS
    EXPAND(i IN 1, N)
    tp_in.q[i] = cond / N * (tp_in.T -
    tp_out.T)
END COMPONENT

COMPONENT GR IS_A Conductor
  DATA
       REAL REF = 0.  "REF (m^2)"
  CONTINUOUS
    EXPAND (i IN 1,N)
    tp_in.q[i] = STEFAN *  REF / N *  \
    ((tp_in.T[i] + TZERO)**4 - (tp_out.T[i] +
    TZERO)**4)
END COMPONENT
```

Handling ARRAYS - Example: Fin Wall (I) Variable Arrays in Ports

```
COMPONENT Wall_fin (INTEGER n=10 "Internal number of nodes")
    PORTS
        IN  Thermal tp_in
        OUT Thermal tp_out
        OUT Thermal (nodes = n) tp_lat -- Array in the port

    DATA
        REAL A    = 0.01       "Area (m^2)"
        REAL e    = 0.1        "Thickness (m)"
        REAL cond = 10.        "Thermal conductivity (W/°C)"
        REAL rho  = 1000.      "Density (kg/m^3)"
        REAL cp   = 2000.      "Specific Heat (J/kg °C)"
    DECLS
        REAL dx           "Distance between nodes (m)"
        REAL T[n]         "Internal temperatures (°C)"
        REAL q[n+1]      "Heat flows between nodes (W)"
        -- …/…
```

EA International

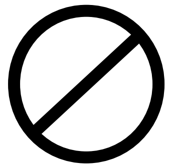
Handling ARRAYS - Example: Fin Wall (II) Variable Arrays in Ports

```
CONTINUOUS
   dx = e / n

   EXPAND (i IN 1, n)
     A * dx * rho * cp  * T[i]' = q[i] - q[i+1] - tp_lat.q[i]

   EXPAND (i IN 2, n)
     q[i] = A * cond / dx * (T[i-1] - T[i])

   q[1]       = A * cond / (dx/2.) * (tp_in.T[1] - T[1])
   q[n+1] = A * cond / (dx/2.) * (T[n] - tp_out.T[1])

   tp_in.q[1] = q[1]
   tp_out.q[1] = q[n+1]

   EXPAND(i IN 1, n)
     tp_lat.T[i] = T[i]

END COMPONENT
```

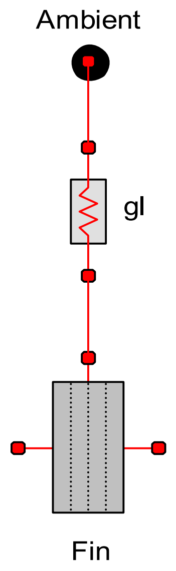
EA International

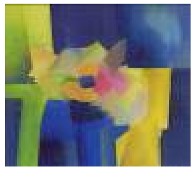
Handling ARRAYS
Array of ports – Arrays' Variables in Ports

- The **arrays of ports** has the drawback of not being supported by the graphical tool Schematic. *The number of ports **being not defined** when the user tries to create the symbol.*

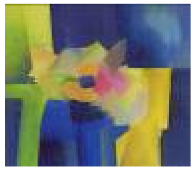


- Using **arrays of variables** in the **vectorized** ports avoids this problem and the user can create the symbol.





MATHEMATICAL PROCESS

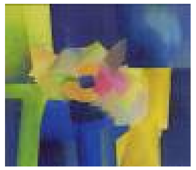


Mathematical Processing (I)

- Steps:
  1. Check for High Index Problems (number of dynamic variables greater than degrees of freedom)
  2. Check for Boundary Conditions
  3. Tear Algebraic Loops
  4. Generate DAE function in a C++ program
  5. Solve DAEs using DASSL or DASSL sparse

EA International

Mathematical Processing (II)



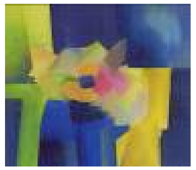
- High Index Problems
  - It occurs when the math equations have links between the state variables (their derivatives appear in the formulation)
  - The most efficient solution of a high index problem is obtained by symbolic differentiation of the equations
  - Pantelides algorithm is applied to reduce the high index of the problem

KopooS Consulting Ind.

Mathematical Processing (III)

- Algebraic Loops
  - There are two cases:
    - Case 1
      - All the equations of the block are linear with respect to the block variables, then a solver for linear equations will be used, and problem will be hidden to the user
    - Case 2
      - Some of the equations are not linear, then the algebraic loops have to be torn off, and a non-linear equation linear solver will be used (Equation Tearing)
      - There are no algorithms for finding an optimum tearing because the Tearing algorithms is based on heuristic rules
      - The knowledge of the model topology and the knowledge of the structure equation helps to understand how to break the loops

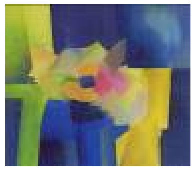
EA International



USING ECOSIM MODELS FROM OTHER TOOLS

KopooS
Consulting Ind.

EA International



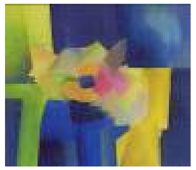
Connection to the External World

- EcosimPro generates C++ class with the final simulation code. This class can be used to include the model in other programs
- Executables of EcosimPro Models are also Active X servers. This enables to build experiments directly in Visual Basic
- An Excel Add-in is provided for easy connection to EcosimPro models from Excel
- It is possible to call Fortran, C and C++ functions from the equations (For example: property routines for H2)

KopooS Consulting Ind.

EA International

Example of calling a Ecosim model from C++

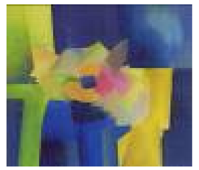


- EcosimPro automatically generates two C++ classes:
  - A class that embraces the partition
    - Header file: COMPONENT.PARTITION.h
    - Implementation file: COMPONENT.PARTITION.cpp
  - A class that embraces the experiment
    - Header file: COMPONENT.PARTITION.EXPERIMENT.h
    - Implementation file: COMPONENT.PARTITION.EXPERIMENT.cpp

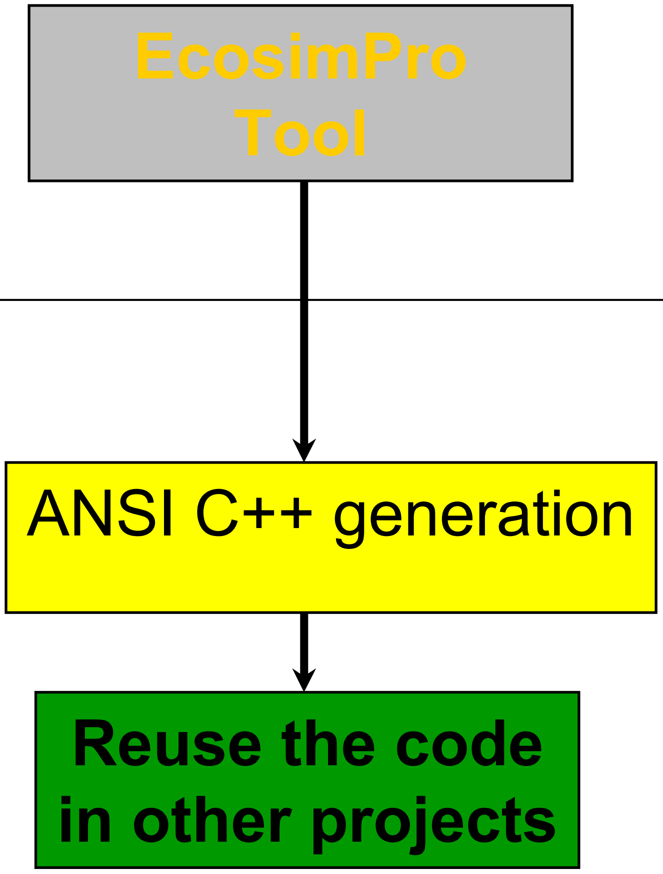
KopooS Consulting Ind.

EA International

Example of calling a Ecosim model from C++



Any model is converted to a C++ class automatically

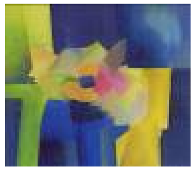


Users of EcosimPro

System programmers

No run-time license is needed

Example of calling a Ecosim model from C++



- **Steps to be followed:**
  - Create an empty Win32 Console Application project
  - Add the following libraries to the project:
    - INTEG_DEBUG_MT.LIB (DEBUG mode) or DEBUG_MT.LIB (RELEASE mode)
    - DFORMD.LIB
    - DFCONSOL.LIB
  - Copy and Paste the header and the implementation files of the partition and the experiments in the project folder
  - Add the external libraries to the project
  - To put visible two files generated by EcosimPro apart from the header and implementation files:
    - A file with symbols table. It is labelled as .txt

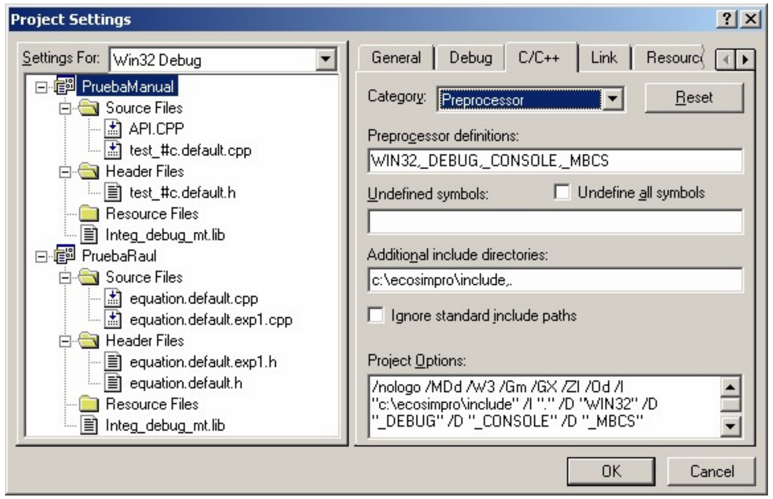
      COMPONENT.PARTITION.txt
    - A file with the Jacobian sparsity. It is labelled as .spa

      COMPONENT.PARTITION.spa

```
m_sparseFileName = "USER_LIBS\\DEFAULT_LIB\\EXPERIMENTS\\equation.default\\equation.default.spa";
```

Example of calling a Ecosim model from C++

- Change the following project settings in C/C++ tab
  - Precompiled Headers -> Not using Precompiled headers
  - Code Generation -> Use run-time library -> Debug Multithreaded DLL
  - Pre-processor -> Additional include directories -> C:\EcosimPro\include, .



Settings For: Win32 Debug

```
PruebaManual
  Source Files
    API.CPP
    test_#c.default.cpp
  Header Files
    test_#c.default.h
  Resource Files
  Integ_debug_mt.lib
PruebaRaul
  Source Files
    equation.default.cpp
    equation.default.exp1.cpp
  Header Files
    equation.default.exp1.h
    equation.default.h
  Resource Files
  Integ_debug_mt.lib
```

Category: Preprocessor

Preprocessor definitions: WIN32,_DEBUG,_CONSOLE,_MBCS

Undefined symbols: ☐ Undefine all symbols

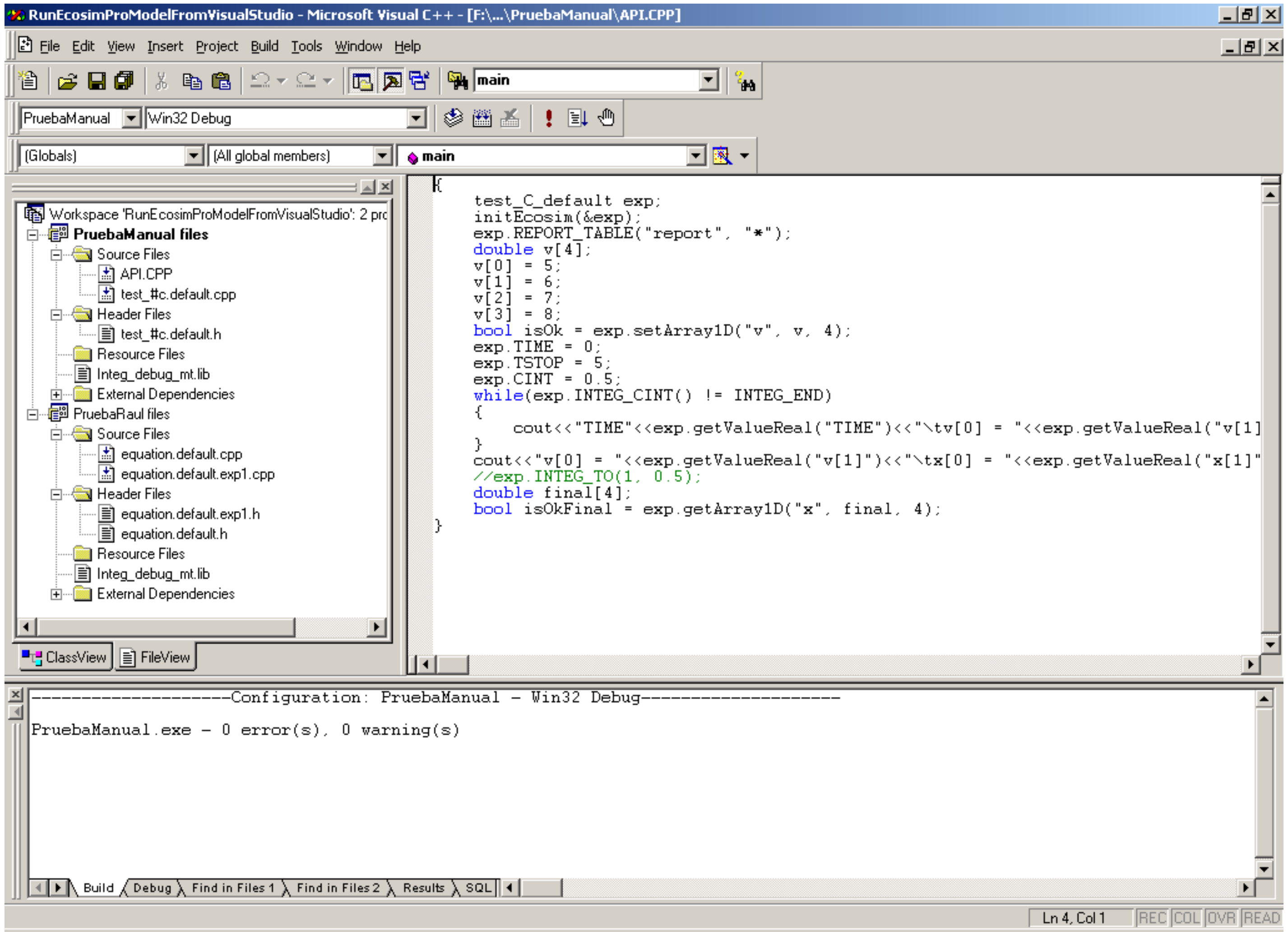
Additional include directories: c:\ecosimpro\include,.

☐ Ignore standard include paths

Project Options:

```
/nologo /MDd /W3 /Gm /GX /ZI /Od /I
"c:\ecosimpro\include" /I "." /D "WIN32" /D
"_DEBUG" /D "_CONSOLE" /D "_MBCS"
```

Example of calling a Ecosim model from C++



```
{
    test_C_default exp;
    initEcosim(&exp);
    exp.REPORT_TABLE("report", "*");
    double v[4];
    v[0] = 5;
    v[1] = 6;
    v[2] = 7;
    v[3] = 8;
    bool isOk = exp.setArray1D("v", v, 4);
    exp.TIME = 0;
    exp.TSTOP = 5;
    exp.CINT = 0.5;
    while(exp.INTEG_CINT() != INTEG_END)
    {
        cout<<"TIME"<<exp.getValueReal("TIME")<<"\tv[0] = "<<exp.getValueReal("v[1]
    }
    cout<<"v[0] = "<<exp.getValueReal("v[1]")<<"\tx[0] = "<<exp.getValueReal("x[1]"
    //exp.INTEG_TO(1, 0.5);
    double final[4];
    bool isOkFinal = exp.getArray1D("x", final, 4);
}
```

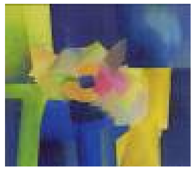
```
--------------------Configuration: PruebaManual - Win32 Debug--------------------

PruebaManual.exe - 0 error(s), 0 warning(s)
```

Excel Interface

- There are two options to run an EcosimPro model from Excel:

  - Interface with Microsoft Excel using VB functions
  - Interface with Microsoft Excel using EcosimPro Toolbar

EA International

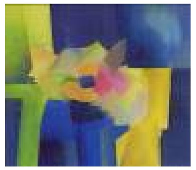


Excel Interface using VB functions

- EcosimPro has an interface based on small number of VB functions
- There is an Active X that provides the programmer with a series of object and their associated methods.
- Because it is an Active X, a user interface can be created from any program or language to access ActiveX objects such as VisualBasic, VisualC++, Delphi or Microsoft Office (Excel, Access).

KopooS Consulting Ind.

EA International



Excel Interface using VB functions

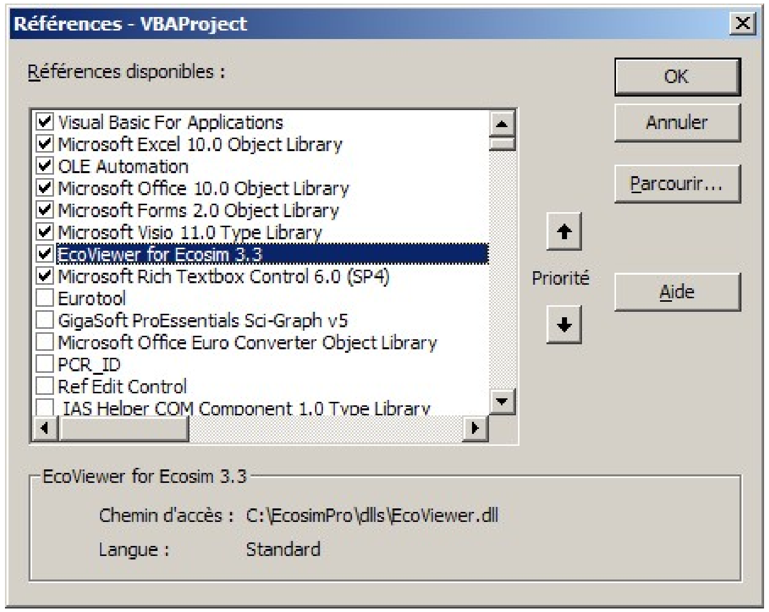
- Experiments generated by EcosimPro can be used easily in Excel or Visual Basic. For this purpose there is an interface based on a small number of VB functions.
- The Excel spreadsheet can be independent of EcosimPro, i.e. **it is not necessary to install EcosimPro to run the experiments, although in this case a minimum installation called EcoViewer would be required**.
- EcoViewer.dll is a DLL ActiveX that provides the programmer with a series of objects together with their associated methods.
- As it is a DLL Active X, a user interface can be created from any program or language to access ActiveX objects such as VisualBasic, VisulaC++ and Delphi, as well as Microsoft Office (Excel, Access, etc).

KopooS Consulting Ind.

Excel Interface using VB functions Installation

- The Active X "EcoViewer for Ecosim 3.3" will be registered, which must be included as reference in each project:
- Open the Visual Basic Editor of Excel (Tools->Macro->Visual Basic Editor) Then open the References window (Tools->References) and activate "EcoViewer for Ecosim 3.3".
- If the Active X does not appear in the list, then you will have to register it manually. To do this, open a DOS box, go to the folder where EcosimPro is installed and go to the folder containing the dlls. Use the regsvr32.exe command to register the dll.

  regsvr32 ecoviewer.dll
- Or else you can look for it directly using the Examine button in the previous window.

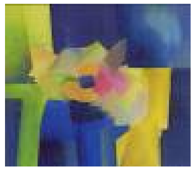


- [✓] Visual Basic For Applications
- [✓] Microsoft Excel 10.0 Object Library
- [✓] OLE Automation
- [✓] Microsoft Office 10.0 Object Library
- [✓] Microsoft Forms 2.0 Object Library
- [✓] Microsoft Visio 11.0 Type Library
- [✓] EcoViewer for Ecosim 3.3
- [✓] Microsoft Rich Textbox Control 6.0 (SP4)
- [ ] Eurotool
- [ ] GigaSoft ProEssentials Sci-Graph v5
- [ ] Microsoft Office Euro Converter Object Library
- [ ] PCR_ID
- [ ] Ref Edit Control
- [ ] IAS Helper COM Component 1.0 Type Library

EcoViewer for Ecosim 3.3

Chemin d'accès : C:\EcosimPro\dlls\EcoViewer.dll

Langue : Standard



Excel Interface using VB functions Example

- The first thing we have to do is to create an object of class Ecoview. If necessary, we can also create objects for EcoGenStatus (practically indispensible), EcoIntStatus, EcoType and EcoCategory.

```
Public WithEvents ECO As EcoView
Public ECO_GS As EcoGenStatus
Public ECO_IS As EcoIntStatus
Public ECO_Type As EcoType
Public ECO_Categ As EcoCategory
```

- The only one which has events is EcoView, which is why WithEvents is used.
- Next, we have to provide these objects with information. If we want, we can do this when we open the Excel project (also possible in Visual Basic).

```
Private Sub Workbook_Open()
    Set ECO = New EcoView
    Set ECO_GS = New EcoGenStatus
    Set ECO_IS = New EcoIntStatus
    Set ECO_Type = New EcoType
    Set ECO_Categ = New EcoCategory
    ECO.Environment
End Sub
```

Excel Interface using VB functions Example

- Before we can use the ActiveX methods we have to add the dll search path, which done simply by using the following command:

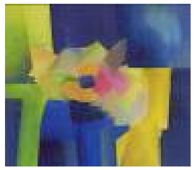
```
ECO.Environment
```

- Lastly, we load the experiment.

```
'Path and file name of the experiment
msExpFile = "aircraft#gear.default.exp1.dll"
sPath = ECO.GetInstallPath + _
    "\Interface Monitor\Excel\aircraftGear\"
'Argument 0 is reserved to "DllVB.dll" (automatic)
Dim sArguments(1 To 4) As String
sArguments(1) = "/OUTDIR"
sArguments(2) = sPath
sArguments(3) = "/INDIR"
sArguments(4) = sPath

Hoja1.MessagesBox.Text = ""
Hoja1.MessagesBox.Height = 130
Hoja1.MessagesBox.Width = 600
```

Excel Interface using VB functions Example

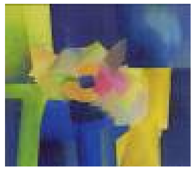
```
'Load the experiment

Dim hExperiment As Long
hExperiment = ECO.LoadDll(msExpFile, sPath, sArguments, _
    bPpr, Hoja1.MessagesBox.hWnd)
If (hExperiment = 0) Then
    MsgBox "Error opening file: " + sPath + msExpFile
    bLoaded = False
Else
    bLoaded = True
End If
```



Excel Interface using VB functions Example - Run

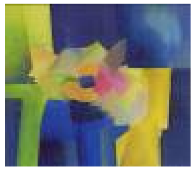
- Running the experiment

```
Public Sub Run_experiment()
    On Error GoTo Myerror
    'Check if experiment was loaded
    If (Not bLoaded) Then
        MsgBox "The experiment is not loaded"
        Exit Sub
    End If

    If ECO.GetGeneralStatus <> ECO_GS.PAUSE_ Then
        Reset_experiment
    End If

    ECO.SetGeneralStatus ECO_GS.RUN_

    Exit Sub
Myerror:
    MsgBox "Error running experiment!"
End Sub
```



EA International

Excel Interface using VB functions Refresh

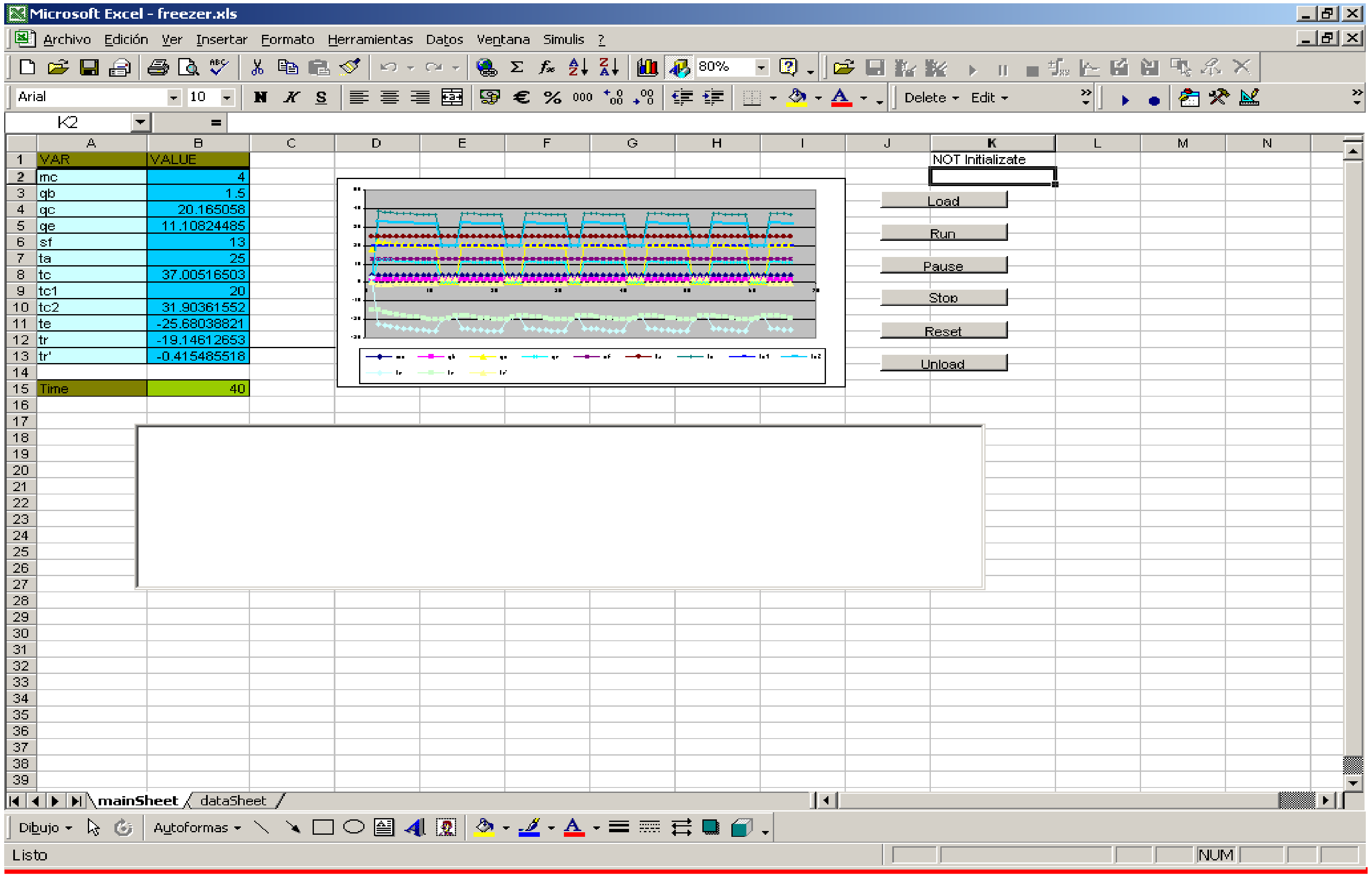


- Once the experiment is running, we can access the variable values implementing the RefreshView event of the EcoView object.

```
Private Sub ECO_RefreshEvent()
   If (Not bLoaded) Then
        Exit Sub
   End If
   Set gPlotData = Worksheets("dataSheet")
   With gPlotData
         .Range("A" & rowpos).Value = ECO.GetTime()
         Dim i As Integer
         .Range("B" & rowpos).Value = _
             ECO.GetVarValue(Variables(1))
       rowpos = rowpos + 1
         Dim rowIni As Integer
         rowIni = 2
         gPlot.SetSourceData Source:=.Range("A1:D1," & _
             "A" & rowIni & ":D" & rowpos), PlotBy _
             :=xlColumns
         gplotsheet.Cells(11, 11) = _
             ECO.GetVarValue(Variables(1))
   End With
   DoEvents
```

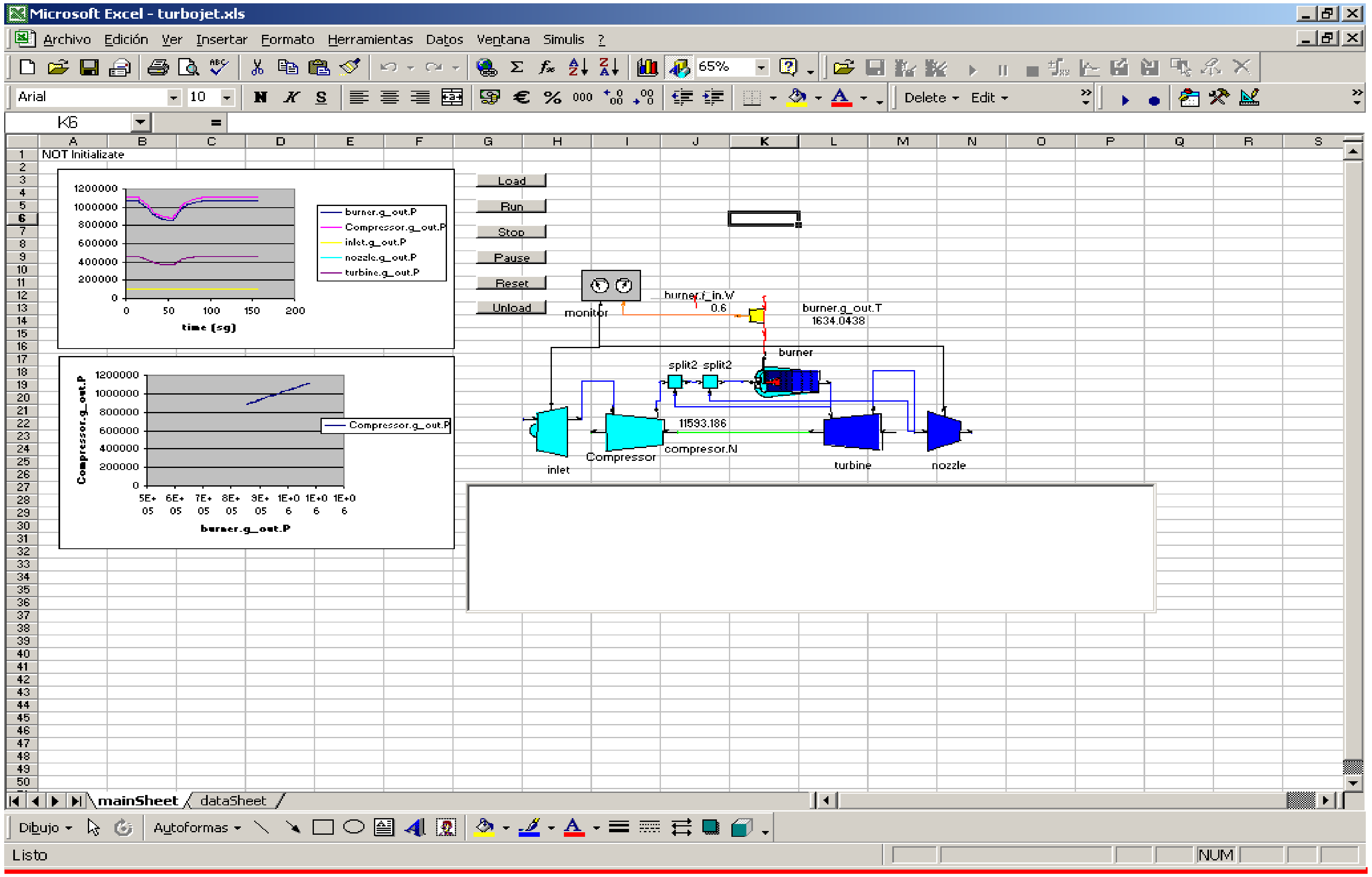
KopooS Consulting Ind.

Excel Interface using VB functions Example I: freezer



VAR	VALUE
mc	4
qb	1.5
qc	20.165058
qe	11.10824485
sf	13
ta	25
tc	37.00516503
tc1	20
tc2	31.90361552
te	-25.68038821
tr	-19.14612653
tr'	-0.415485518
Time	40

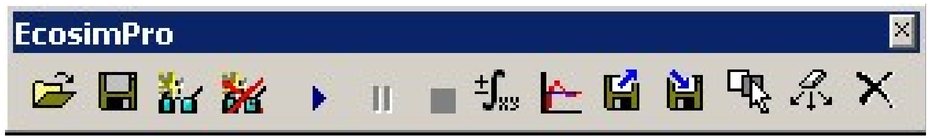
Excel Interface using VB functions Example II: turbojet



EA International

Excel Interface using Ecosim Toolbar

- In order to make the user easier to run EcosimPro models from Excel, EcosimPro has an Interface with Microsoft Excel using a toolbar

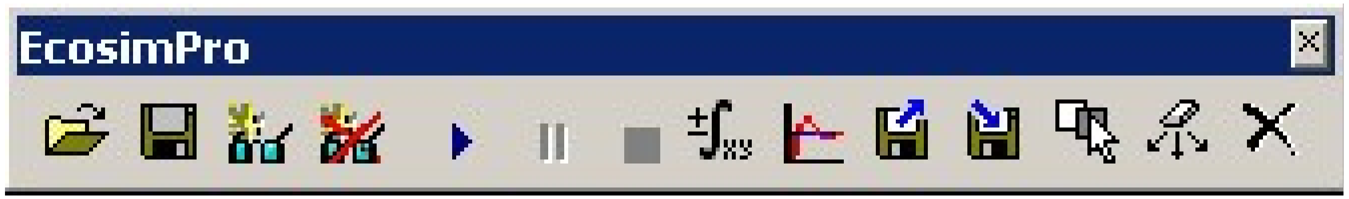


- To use this interface, it is only necessary to run a small program named Register.exe in the EcosimPro installation folder. This allows to record the EcosimPro toolbar for Excel. The installation is only valid for Microsoft Excel 2000 or greater
- If the user wants to eliminate the EcosimPro toolbar in Excel, the user must also run this program and mark the "Unregister add-in for Excel" code.
- Finally the user must load the EcosimPro toolbar

EA International

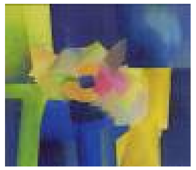
Excel Interface using Ecosim Toolbar

- The options available in the EcosimPro toolbar are the following:
  - Open an experiment
  - Save an experiment configuration
  - Assign a variable to a selected cell
  - Delete a variable
  - Run, pause or stop an experiment
  - Perform a new integration
  - Calculate a steady state
  - Save or restore a snapshot of the simulation
  - Reset an experiment to the initial values
  - Clean the EcosimPro result Sheet



KopooS Consulting Ind.

Excel Interface using Ecosim Toolbar Methodology

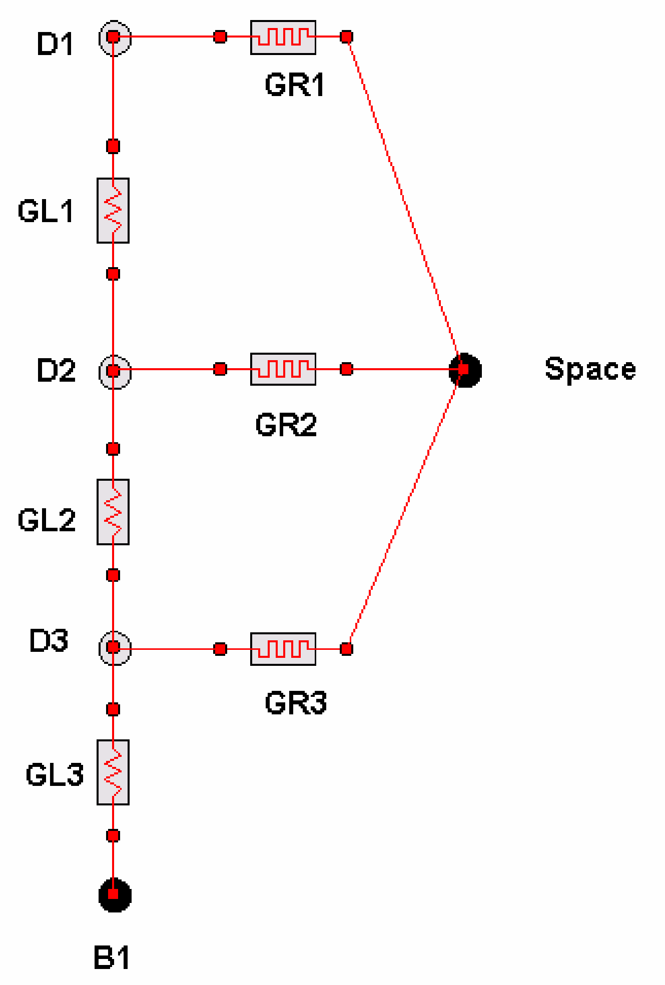


1. Simulate in EcosimPro the experiment the user wants to run from Excel (in order to generate the DLL file)
2. Create an Excel template. Design the Worksheet format and the schematic diagram in Excel.
3. Create the EcoExcel configuration file
4. Simulate the experiments in Excel
5. Create the plots of the most interesting variables in the Excel template
6. Report the results

EA International

Excel Interface using Ecosim Toolbar Example (I)

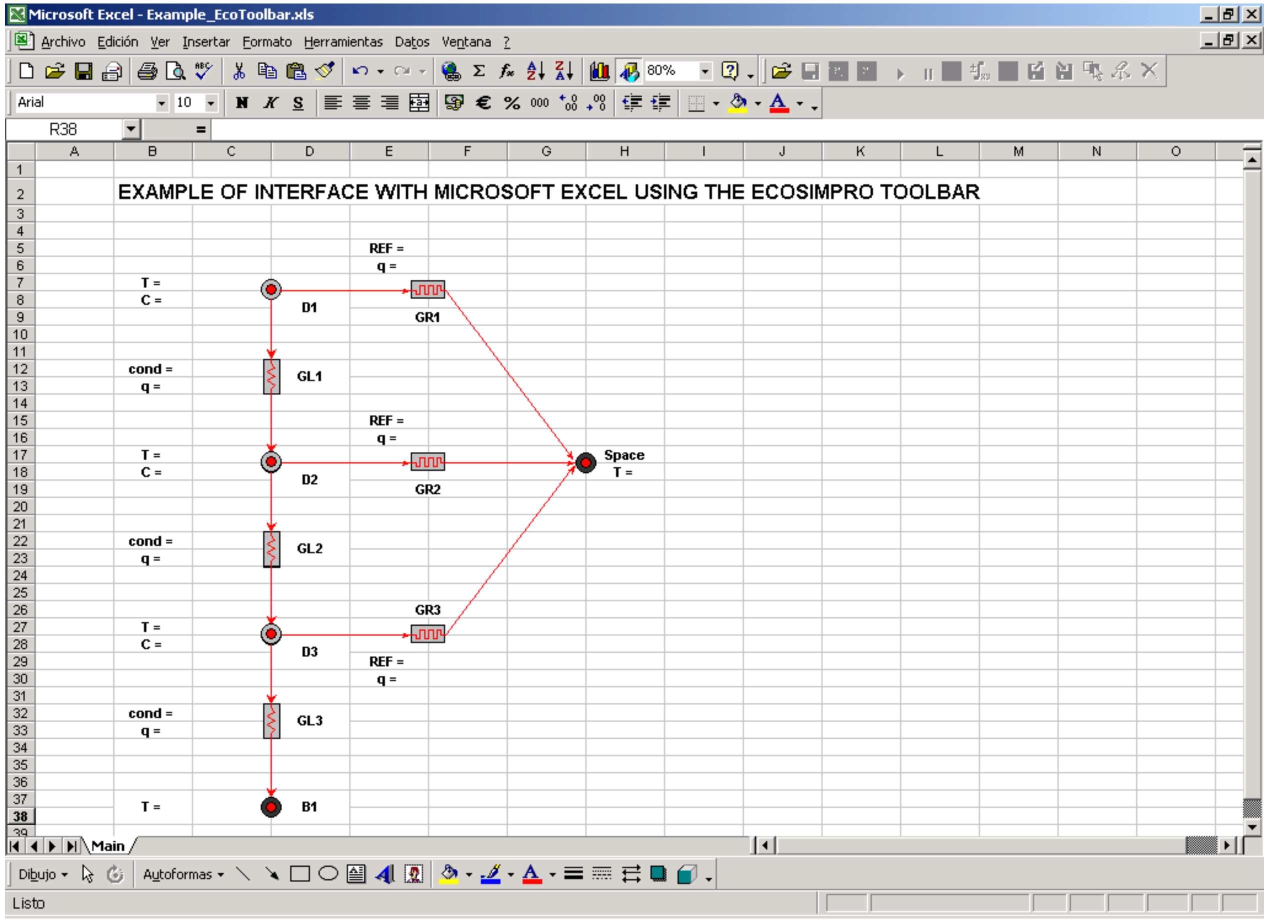
- Build an Excel Interface for the thermal network model developed during the first day of the course
- Start from an Excel Template of the system already created
- Create the EcoExcel Configuration file
- Simulate the model from Excel
- Generate two plots in the Excel Template:
  - Temperatures of the nodes
  - Heat flows across the nodes



EA International

Excel Interface using Ecosim Toolbar Example (II)

- Excel Template of the thermal network model: **Example_EcoToolbar.xls**



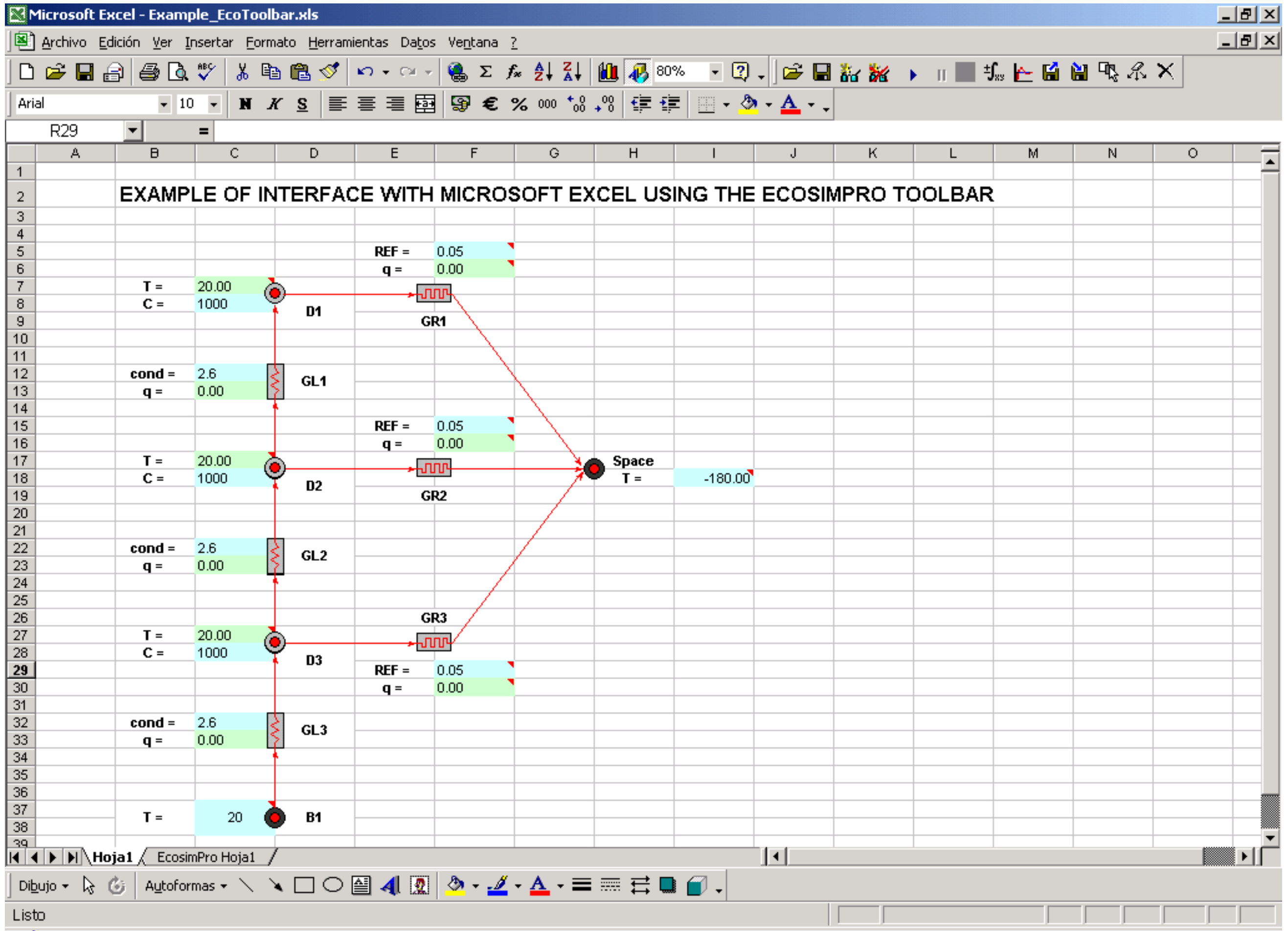
Microsoft Excel - Example_EcoToolbar.xls

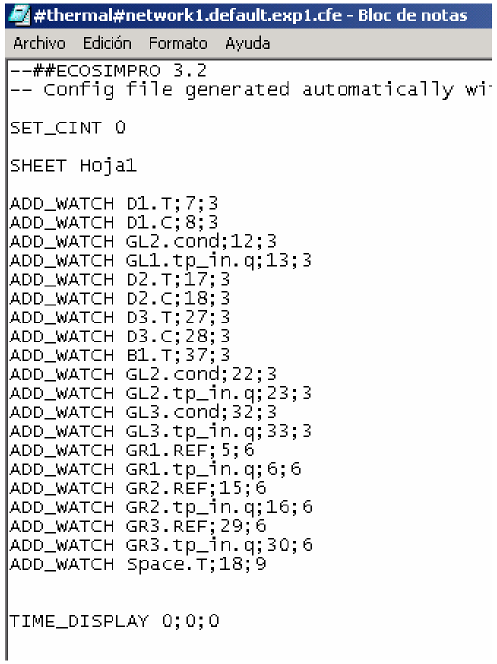
EXAMPLE OF INTERFACE WITH MICROSOFT EXCEL USING THE ECOSIMPRO TOOLBAR

KopooS Consulting Ind.

Excel Interface using Ecosim Toolbar Example (III)

- Create the EcoExcel configuration file

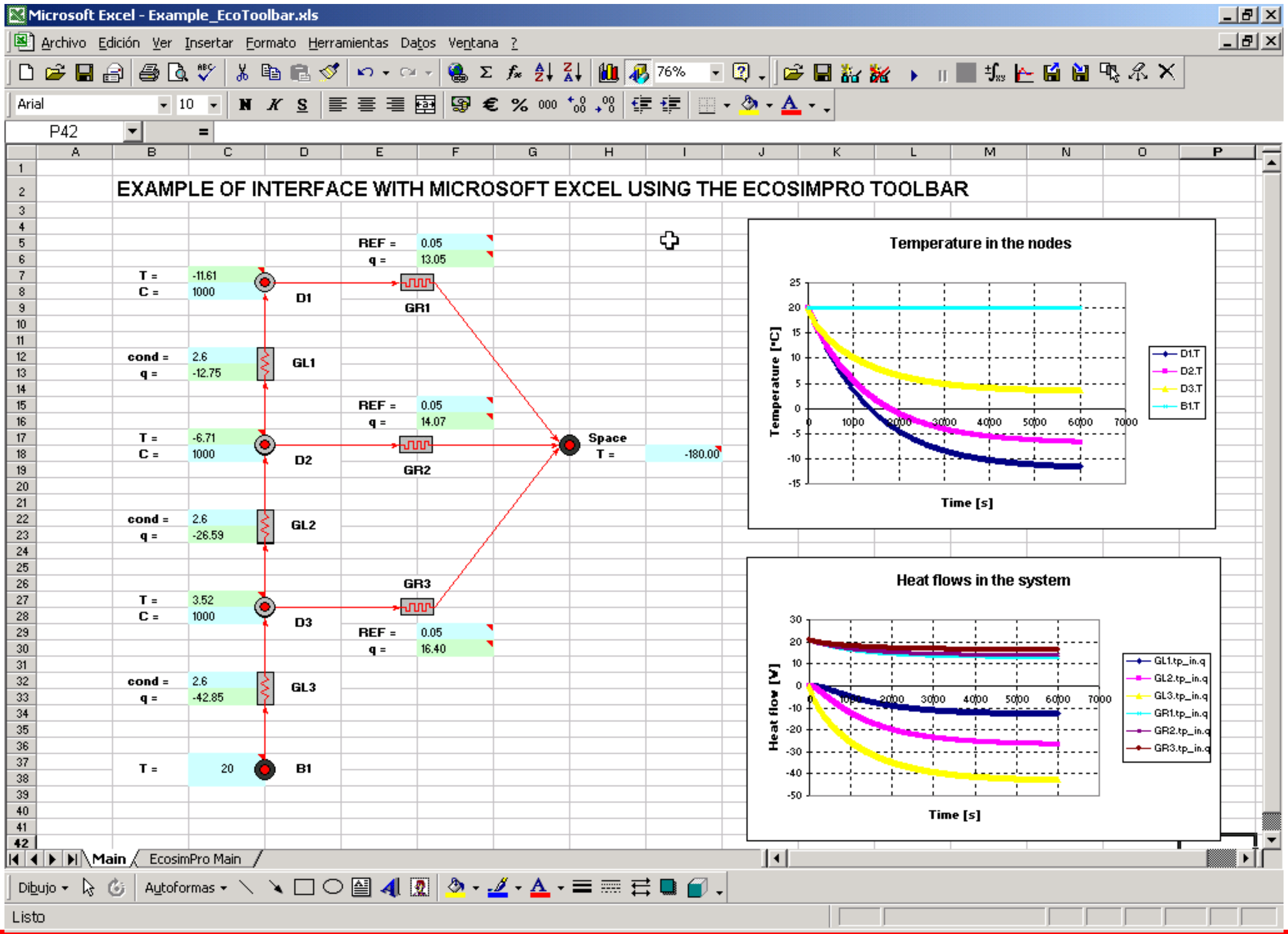



#thermal#network1.default.exp1.cfe - Bloc de notas
Archivo Edición Formato Ayuda

```
--##ECOSIMPRO 3.2
-- Config file generated automatically wi

SET_CINT 0

SHEET Hoja1

ADD_WATCH D1.T;7;3
ADD_WATCH D1.C;8;3
ADD_WATCH GL2.cond;12;3
ADD_WATCH GL1.tp_in.q;13;3
ADD_WATCH D2.T;17;3
ADD_WATCH D2.C;18;3
ADD_WATCH D3.T;27;3
ADD_WATCH D3.C;28;3
ADD_WATCH B1.T;37;3
ADD_WATCH GL2.cond;22;3
ADD_WATCH GL2.tp_in.q;23;3
ADD_WATCH GL3.cond;32;3
ADD_WATCH GL3.tp_in.q;33;3
ADD_WATCH GR1.REF;5;6
ADD_WATCH GR1.tp_in.q;6;6
ADD_WATCH GR2.REF;15;6
ADD_WATCH GR2.tp_in.q;16;6
ADD_WATCH GR3.REF;29;6
ADD_WATCH GR3.tp_in.q;30;6
ADD_WATCH Space.T;18;9


TIME_DISPLAY 0;0;0
```

Excel Interface using Ecosim Toolbar Example (IV)

- Simulate the model and create the plots in Main Sheet

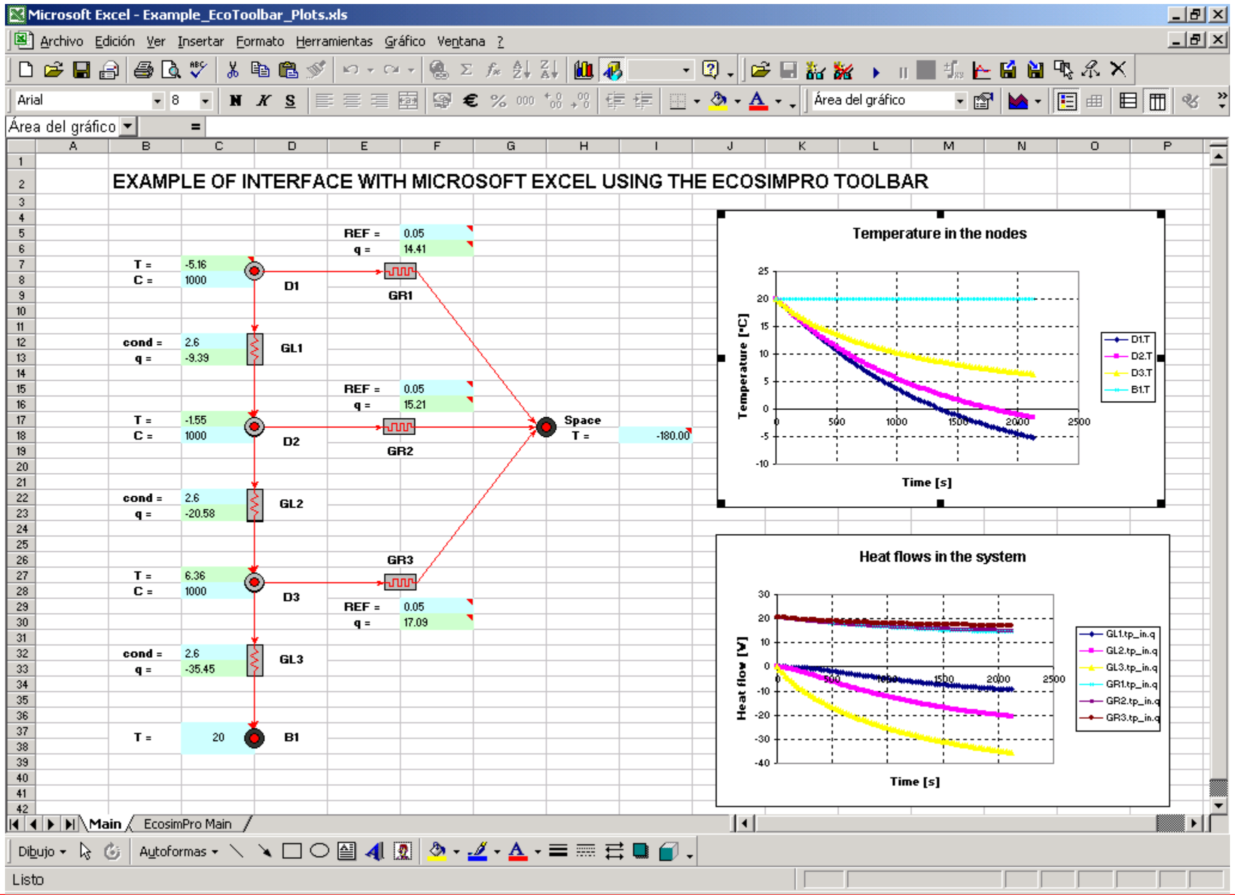

Microsoft Excel - Example_EcoToolbar.xls

EXAMPLE OF INTERFACE WITH MICROSOFT EXCEL USING THE ECOSIMPRO TOOLBAR

REF =	0.05	
q =	13.05	
T =	-11.61	D1
C =	1000	
cond =	2.6	GL1
q =	-12.75	
REF =	0.05	
q =	14.07	
T =	-6.71	D2
C =	1000	
cond =	2.6	GL2
q =	-26.59	
T =	3.52	D3
C =	1000	
REF =	0.05	
q =	16.40	
cond =	2.6	GL3
q =	-42.85	
T =	20	B1
Space T =	-180.00	

Excel Interface using Ecosim Toolbar Example (V)

- Clean EcosimPro Main Sheet, Reset to initial values and Simulate the model



Microsoft Excel - Example_EcoToolbar_Plots.xls

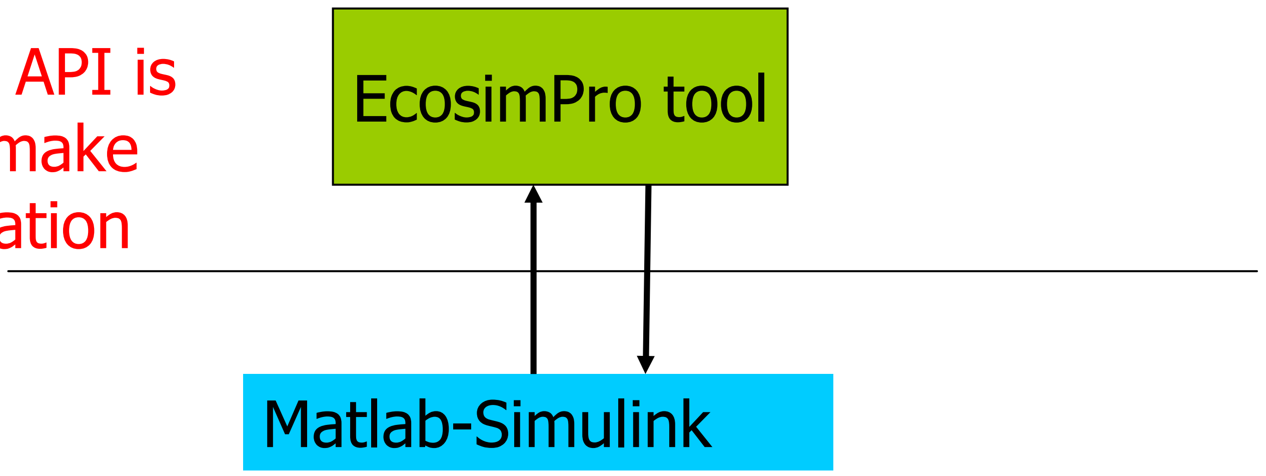
EXAMPLE OF INTERFACE WITH MICROSOFT EXCEL USING THE ECOSIMPRO TOOLBAR

D1: T = -5.16, C = 1000
GR1: REF = 0.05, q = 14.41
GL1: cond = 2.6, q = -9.39
D2: T = -1.55, C = 1000
GR2: REF = 0.05, q = 15.21
Space: T = -180.00
GL2: cond = 2.6, q = -20.58
D3: T = 6.36, C = 1000
GR3: REF = 0.05, q = 17.09
GL3: cond = 2.6, q = -35.45
B1: T = 20

Matlab Interface

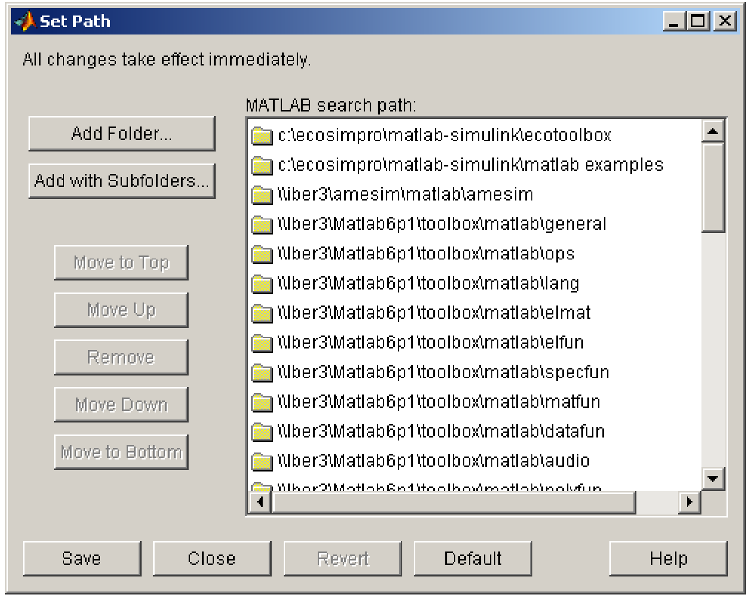
- Based on Matlab version R13 (which requires Service Pack 1 – Generic dll_1p1.exe), an interface has been established to directly call from this program, C functions exported in a dynamic link library (DLL).
- Taking advantage of this interface, a series of functions have been created to comprise a toolbox, with which an EcosimPro model can be easily and quickly used.

A simple API is used to make co-simulation



Matlab Interface (Installation)

- The "ecoToolbox" is located in the following directory:
- INSTALL_DIR\ Interface Monitor\ ecoToolbox
- and EcosimPro is installed in directly INSTALL_DIR.
- To be able to use the toolbar, the path prior to the MATLAB environment has to be added via the menu option:
- File Set Path...
- The following dialogue will appear:



All changes take effect immediately.

MATLAB search path:

```
c:\ecosimpro\matlab-simulink\ecotoolbox
c:\ecosimpro\matlab-simulink\matlab examples
\\iber3\amesim\matlab\amesim
\\Iber3\Matlab6p1\toolbox\matlab\general
\\Iber3\Matlab6p1\toolbox\matlab\ops
\\Iber3\Matlab6p1\toolbox\matlab\lang
\\Iber3\Matlab6p1\toolbox\matlab\elmat
\\Iber3\Matlab6p1\toolbox\matlab\elfun
\\Iber3\Matlab6p1\toolbox\matlab\specfun
\\Iber3\Matlab6p1\toolbox\matlab\matfun
\\Iber3\Matlab6p1\toolbox\matlab\datafun
\\Iber3\Matlab6p1\toolbox\matlab\audio
```

EA International

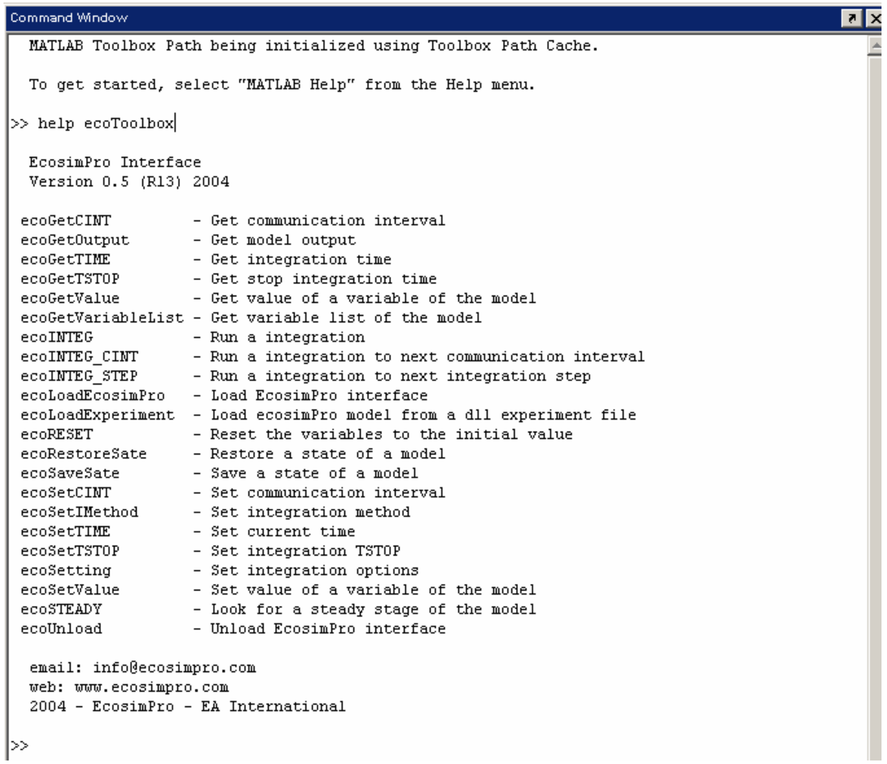
Matlab Interface (Installation)

- The toolbox functions have to make calls to different libraries located in:
- INSTALL_DIR\dlls
- and so this directory must be added to the operating system environment variable PATH as a system variable and not as a user variable.
- When we reach this point the toolbox is installed and can be used both by the MATLAB environment and from Simulink.
- To call DLL functions from MATLAB 6.5, you need to download and install the DLL Interface Library. The library is available from our Web site: http://www.mathworks.com/support/solutions/files/s33513/GenericDll_1p1.exe

EA International

Matlab Interface (Functions)

- The following is a list of the toolbox functions. This list can be obtained directly in MATLAB with the sentence:
  help ecoToolbox



```
  MATLAB Toolbox Path being initialized using Toolbox Path Cache.

  To get started, select "MATLAB Help" from the Help menu.

>> help ecoToolbox

  EcosimPro Interface
  Version 0.5 (R13) 2004

 ecoGetCINT         - Get communication interval
 ecoGetOutput       - Get model output
 ecoGetTIME         - Get integration time
 ecoGetTSTOP        - Get stop integration time
 ecoGetValue        - Get value of a variable of the model
 ecoGetVariableList - Get variable list of the model
 ecoINTEG           - Run a integration
 ecoINTEG_CINT      - Run a integration to next communication interval
 ecoINTEG_STEP      - Run a integration to next integration step
 ecoLoadEcosimPro   - Load EcosimPro interface
 ecoLoadExperiment  - Load ecosimPro model from a dll experiment file
 ecoRESET           - Reset the variables to the initial value
 ecoRestoreSate     - Restore a state of a model
 ecoSaveSate        - Save a state of a model
 ecoSetCINT         - Set communication interval
 ecoSetIMethod      - Set integration method
 ecoSetTIME         - Set current time
 ecoSetTSTOP        - Set integration TSTOP
 ecoSetting         - Set integration options
 ecoSetValue        - Set value of a variable of the model
 ecoSTEADY          - Look for a steady stage of the model
 ecoUnload          - Unload EcosimPro interface

  email: info@ecosimpro.com
  web: www.ecosimpro.com
  2004 - EcosimPro - EA International

>>
```

Matlab Interface – Examples

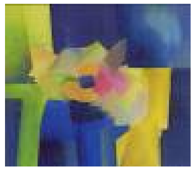
- To illustrate the use of the toolbox, we are going to describe two small examples included in the directory:

  INSTALL_DIR\\Matlab-Simulink\Matlab examples

- In the script ecoExample1 we will see how to run a model in a simple fashion by first initialising a variable

- In the script ecoExample2 we will run the same model that in ecoExample1 but we will show the results in a plot

EA International



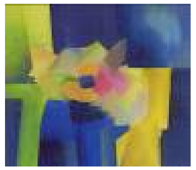
Matlab Interface – ecoExample1

- First, with the function **ecoLoadEcosimPro** we load EcosimPro's generic DLL with which MATLAB will communicate with our model. In this case it is INSTALL_DIR, and therefore the directory in which EcosimPro was installed, is: "C:\EcosimPro"
- We then load the model (**ecoLoadExperiment**) which we will find in the examples library DEFAULT_LIB called aircraftGear displaying the model variables
- Using the sentence **ecoSetting** we can tell EcosimPro several things. In this case, we say we do not want it to return the integration messages, which will increase the process speed; in addition, with **ecoSetIMethod** we change the integration algorithm to a fourth order Runge-Kutta
- With **ecoSetTSTOP** we establish the end of simulation in 15 seconds. As nothing has been indicated, it is understood that simulation will start in 0 seconds. With **ecoSetValue** we also initialise the variable 'x' to a value of 200 metres
- The function **ecoINTEG** will run an integration with the parameters defined previously
- To end the process we obtain the value of 'x' and of its first derivative with **ecoGetValue** since they are real values
- When the complete process is finished, we unload the model with **ecoUnload**

EA International

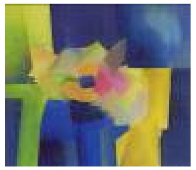
Matlab Interface – ecoExample2

- The first steps are similar to those in the previous example, using the same model aircraftGear (functions **ecoLoadEcosimPro** and **ecoLoadExperiment**)
- In this case we cannot use the same mechanism as we did previously because we need all the intermediate values of the simulation to be able to create the graphic. So instead of telling the program to integrate from an initial time to a final time, we will define the interval in which we want the data to be delivered. We define this with the function **ecoSetCINT** and we use **ecoINTEG_CINT** to give the command to calculate up to the next communication level (or event)
- The integrator will end on TSTOP and this time will be used in comparison with the actual integration time (**ecoGetTIME**) to end the data retrieval loop, this being 'x' and its first and second derivatives
- With the integration finished, we will plot the data obtained
- When the complete process is finished, we unload the model with **ecoUnload**



FUNCTIONS

EA International



EL FUNCTIONS

- The syntax is as follows:

```
FUNCTION data_type IDENTIFIER( argument-list ) STRING_VALUE?  EOL*
[DECLS
Local-variables]
BODY
   Sequential-stms
END FUNCTION
```

- The argument list contains all the function's arguments, separated by commas. All declarations have the following syntax:

```
[IN | OUT] data-type IDENTIFIER
```

- If the argument has the IN prefix, this indicates that it is passed by value. OUT indicates that it is passed by reference and the value of the call variable will be updated.

KopooS Consulting Ind.

EA International

EL FUNCTIONS

```
FUNCTION REAL friction
    (
         IN REAL  eps ,          --relative rugosity
         IN REAL  Re             --reynolds number
     )
     DECLS
         REAL fric
         REAL a
         REAL b
     BODY
         Re = abs(Re)
         IF (Re < 1.e-30) THEN
                Re = 1.e-30
         END IF
         a = (2.457*log(1./((7./Re)**.9+.27*eps)))**16.
         b = (37530./Re)**16.
         fric = ((8./Re)**12.+1./((a+b)**1.5))**.0833333333
         RETURN 8*fric
END FUNCTION
```

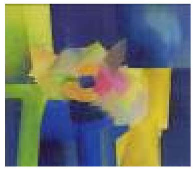
KopooS Consulting Ind.

EA International

Use of EL function

```
CONST REAL G    = 9.806          "Gravity acceleration (m/s^2)"
CONST REAL VISC = 0.001          "Viscosity (Kg/m s)"

COMPONENT Pipe IS_A Channel
    DATA
        REAL l = 1                   "Pipe length (m)"
        REAL d = 0.1                "Pipe diameter (m)"
        REAL rug = 1.5e-6          "Pipe rugosity (m)"
    DECLS
        REAL A
        EXPL REAL f                  "Friction factor (-)"
        EXPL REAL Re                 "Reynolds number (-)"
    CONTINUOUS
        A = 0.25 * PI * d**2
        f_out.m = f_in.m
        Re = (f_in.m * d )  / (A * VISC)
        f = friction(rug/d , Re)
        f_in.P - f_out.P + RHO * G * (z_in - z_out) \
               - 0.5 * f * (l/d) * spow2 (f_in.m) / RHO /A**2 = 0
END COMPONENT
```

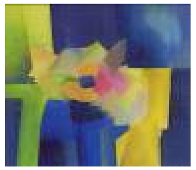
EA International



EXTERNAL FUNCTIONS

- EL allows the user to reuse existing functions written in C, C++ or Fortran.
- There are very powerful libraries available commercially that supply a wide range of mathematical functions (optimisation, properties calculation, etc)
- The users can re-use their own C, C++ or FORTRAN functions easily

EA International



EXTERNAL FUNCTIONS

```
---------------------------------------------------------------------
--Declaration of the FORTRAN Function to calculate the property array
--vs two thermodynamic variables of a real gas
---------------------------------------------------------------------
"FORTRAN" FUNCTION NO_TYPE thermo_prop
--**
--** Purpose: FORTRAN subroutine to calculate thermodinamic and physical
--**                  properties of the following chemical components in liquid
--**                  and vapor phase:
--**                  H2O_nist, H2_nist, N2_nist & O2_nist
--**
--** Note:
--** Arguments:
(
        IN      INTEGER     IG,                         --Fluid identification  number
        IN      INTEGER     J1,                         --First independent variable
        IN      REAL                    V1,             --First independent variable value
        IN      INTEGER     J2,                         --Second independent variable
        IN      REAL                    V2,             --Second independent variable value
        IN      INTEGER     IC,                         --Outlet variables key
        OUT     REAL                    VAR[3, 20],     --Outlet variables values
        OUT     REAL                    Q,              --Fluid quality
        OUT     INTEGER     IER,                        --Error index
        IN      INTEGER     N_OR,                       --Interpolation grade
        OUT     INTEGER     HIS_T,                      --Interpolation memory
        OUT     INTEGER     HIS_P                       --Interpolation memory
) IN "FLUID_TABLES\thermo_table_interp.lib"
```