

# AMESim: A Brief Technical Overview

Technical Bulletin n° 100



**IMAGINE**

Copyright © IMAGINE S.A. 1995-2000  
5 rue Brison, 42300 Roanne, France

Phone : 04 77 23 60 30 (national)  
Fax : 04 77 23 60 31  
Email : [imagine@amesim.com](mailto:imagine@amesim.com)  
Web : [www.amesim.com](http://www.amesim.com)

Phone: +33.4.77.23.60.37 (international)  
Fax: +33.4.77.23.60.31

**AMESim**

## How to contact IMAGINE:

Phone 04 77 23 60 30 (national)  
(33) 4 77 23 60 37 (international)

Fax 04 77 23 60 31 (national)  
(33) 4 77 23 60 31 ((international)

Mail IMAGINE  
5, rue Brison  
F-42300 ROANNE

Web <http://www.amesim.com>

Email [imagine@amesim.com](mailto:imagine@amesim.com) general information  
[hotline@amesim.com](mailto:hotline@amesim.com) Technical support

*Copyright © IMAGINE S.A. 1995-2000*

*AMESim® and AMESet® are the registered trademark of IMAGINE SA.  
All other product names are trademarks or registered trademarks of their respective companies*

---

**IMAGINE**  
5 rue Brison, 42300 Roanne, France

Phone : 04 77 23 60 30 (national)  
Fax : 04 77 23 60 31  
Email : [imagine@amesim.com](mailto:imagine@amesim.com)  
Web : [www.amesim.com](http://www.amesim.com)

Phone: +33.4.77.23.60.37 (international)  
Fax: +33.4.77.23.60.31



# AMESim :

## A Brief Technical Overview

The **AMESim** package encapsulates more than 150 man years of experience in the simulation of electrohydraulic systems. This experience has been accumulated through more than 300 consultancy projects between the **IMAGINE** company and industries working in the fields of automobile, aerospace, robotics, offshore and general hydraulics engineering.

Many different software packages were used to perform these studies but none offered the full range of capabilities needed. There were deficiency in the numerical capabilities, in the graphical interface and in the general modeling concept. The **AMESim** package was developed to overcome these limitations.

What then are the key features that makes **AMESim** suitable for hydraulic and general simulation? This document gives a description of the technical features which were central objectives in the design of the software.

### 1. The Multiport Approach

In the signal port approach, a single value or an array of values are transferred from one component block to another in a single direction. This is fine when the physical engineering system behaves in the same way such as with a control system. However, problems arise when power is transmitted. This is because modeling of components that transmit power leads to a requirement to exchange information between components in both directions. In order to use a signal port

approach in this situation, two connections must be made between the components where physically there is only one. This leads to a great complexity of connections and means that even very simple models involving power transmission appear complex and unnatural.

In contrast to the signal port approach, with the multiport approach, a connection between two components allows information to flow in both directions. This makes the system diagram much closer to the physical system. Normally there are two values involved and the theory of bond graphs provides a good theoretical background into the relationship between these values and the power transmitted. However, there is no limitation in the number of quantities involved. There may be one quantity or three or more quantities. When there is only one quantity, the situation is just like with signal ports. Thus signal ports can be regarded as a special case of multiports.

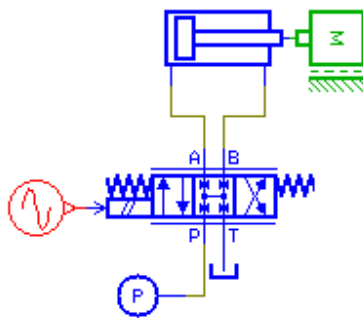


Figure 1: the multiport approach

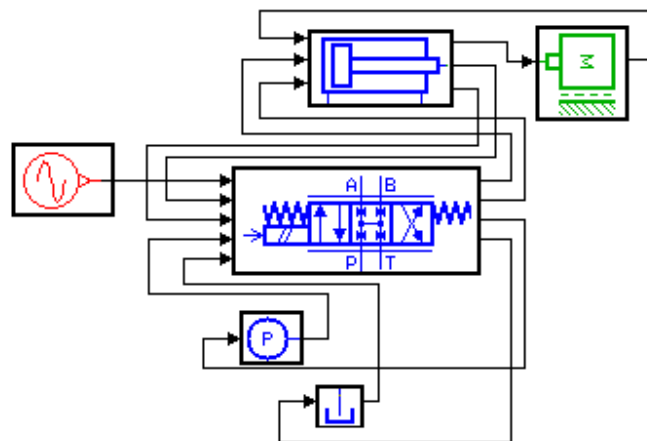


Figure 2: the signal port approach

**AMESim** has always used the multiport approach and Figure 1 shows part of a simple electrohydraulic system using multiport block diagrams. Figure 2 shows the same system with signal ports. The control for the valve is identical in both case since for this port, the multiport reduces to a signal port. However, for the hydraulic and mechanical ports, the extra connections needed for the signal port approach are apparent.

## 2. Strong Numerical Capability

The analysis of the steady state and dynamic behavior of an engineering system leads to a mathematical model of the system. This is in the form of algebraic, ordinary differential and partial differential equations. More recently differential-algebraic equations are also used to model the system. The role of simulation software is provide an environment in which this model can be solved efficiently.

For models with large numbers of partial differential equations there are specialist packages such as those for computation fluid dynamics. Such software is used for detailed analysis of individual components of a system. However, it is often necessary to simulate a whole engineering system or a subsystem of it. The concept of the virtual prototype, in which physical prototypes are replaced by mathematical computer models, makes simulation of this type vital. In this case it is normal to reduce any partial differential equations to ordinary differential equations. This leads to models with either ordinary differential equations (odes) or differential algebraic equations (daes). Many general and specialized simulation software packages are available for solving such systems of equations.

Models arising from engineering systems vary greatly in their character. Thus the equations of the model can be:

- ↗ linear,
- ↗ non-linear,
- ↗ numerically stiff i.e. with very small time constants compared with the overall simulation period,
- ↗ oscillatory,
- ↗ continuous,
- ↗ discontinuous.

A large variety of numerical integration methods can be employed to solve such problems. Traditionally the user of simulation software is presented with a menu of typically seven methods from which a choice must be made. Selection of the wrong method may lead to failure or unnecessarily long run times. Even specialist numerical mathematicians find such a choice very difficult. The situation is made worse because the characteristics of the equations may change during the simulation. Thus initially they may be numerically stiff but become very highly oscillatory when a valve opens during the course of the simulation. It is

unreasonable to expect the user to stop the simulation at the point where the characteristics change and then restart with a different integrator.

**AMESim** attempts to automate the process of switching between methods. First the presence of daes is detected and an appropriate is employed. This is a much modified version of the famous **DASSL** algorithm [1], [2]. If the equations are purely odes, a modified version of the **LSODA** integration algorithm is employed [3]. This algorithm uses a total of 17 different methods and monitors the characteristics of the equations detecting stiffness and the absence of stiffness. Switching is organized between two distinct solvers. An **ADAMS** code with 12 different methods is used when the equations are non-stiff. A **GEAR** code with 5 different methods is used when the equations are stiff.

Both **DASSL** and **LSODA** use linear multistep methods. These methods are noted for intolerance of discontinuities. For this reason special discontinuity handling procedures are incorporated in **AMESim**.

When we use a calculator to compute the sine of an angle, we happily accept the result displayed. A few decades ago the process was more difficult and there were a number of methods available for performing the calculations. We would have had to select a method from those available. Perhaps we would have tried several methods to confirm the result was accurate.

This same process of automatic selection of method and of increasing reliability is already beginning to happen with integration algorithms for odes. Much progress has been made. Perhaps in 10 years we will be as confident in ode integrators as we are now with our calculators.

Integrators for daes are less highly developed but are improving rapidly.

The policy with **AMESim** will be to regularly update the numerical solvers to take advantage of new developments.

### 3. Full Graphical User Interface

Many older simulation packages were developed before modern graphical user interfaces were available. The only graphical facilities provided were for producing simple plots of results. The suppliers of these packages have had to

introduce new graphical preprocessing facilities to build the system. More modern software has been designed from the start with a full graphical user interface.

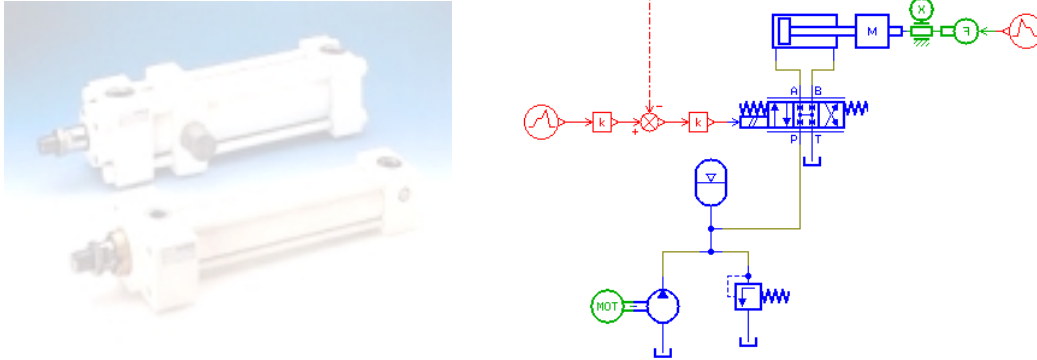


Figure 3: system using mostly standard symbols

Two important objectives in the design of the graphical user interface for AMESim were:

- ↳ Whenever possible, icons for components were based on internationally recognized standard symbols. Thus for hydraulic systems icons are based on CETOPS symbols. Figure 3 shows a typical example. Where there are no such standardized symbols, icons are constructed which can be instantly recognized by engineers working in the field. Figure 4 shows a display of a fuel injection system employing icons of this type.

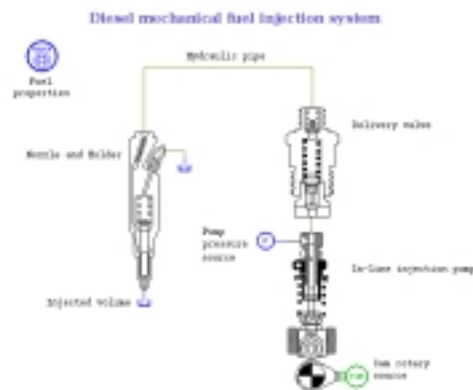


Figure 4: system using mostly 'natural' symbols

- Throughout the simulation process the system diagram is displayed. Thus for example when parameters are changed for a particular component, the user points at the icon in question and clicks the mouse button. This produces a menu of items that may be changed, as in Figure 5. Similarly to plot graphs of results, the user points at the component and clicks the mouse button to produce a menu of items associated with the component that may be plotted.

Normally at some stage in the simulation process a demonstration must be organized. If a good system diagram is displayed and parameters can be changed and results plotted rapidly, a good impression is created. If 'what would happen if?' questions can be answered quickly, the demonstration is very successful. If the system diagram looks unnatural or if it is removed from the process of changing parameters and plotting graphs, a demonstration is much less successful.

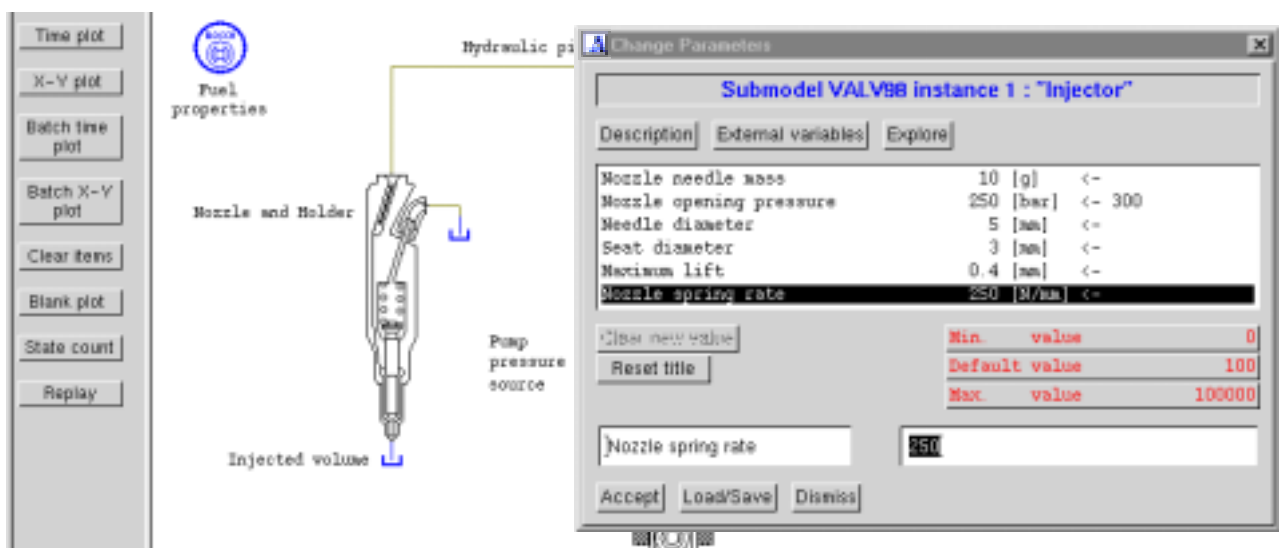


Figure 5: at all times the system sketch is a background for the current operation

## 4. Advanced Modeling Environment

**AMESim** was designed for use in a whole range of industries. Each industry has a variety of systems for which simulation is to be employed. The physical system is made up of components which are assembled to form the complete system. **AMESim** follows precisely the same strategy. Each physical component is represented by an appropriate icon and is associated with one or more submodel. The icons and associated submodels are assembled to form the model of the complete system. In addition, just as it is impossible to connect for instance a mechanical spring to the inlet port of a hydraulic pump, **AMESim** makes checks to prevent such connections.

To make this process work, it is necessary to customize a collection of icons and submodels for a particular industry. This can be done at the **IMAGINE** company but since all the tools are available to perform this task, a competent **AMESim** user can perform this task.

## 5. Expanding AMESim

Associated with **AMESim** is another utility which partially automates the process of building new submodels. This is called **AMESet** which is the Submodel editing tool. It is a utility which allows the specification of a new submodel to be entered and skeleton code to be produced. The submodel writer then takes the submodel code and enters the equations of the submodel.

It would be wonderful if it could be said that the process of writing submodels was very easy. However, in truth writing good submodels does require great care, attention to detail, an understanding of the physics involved and some flair and talent. However, once the submodel is written and tested, it can be reused over and over again in completely different systems.

On many occasions, just two or three new icons and submodels added to the **AMESim** standard library are all that is needed to adapt **AMESim** to your requirements. Alternatively it is possible for a user to open up a completely new area and design a completely new icon and submodel library.

## 6. Interfaces with other Software

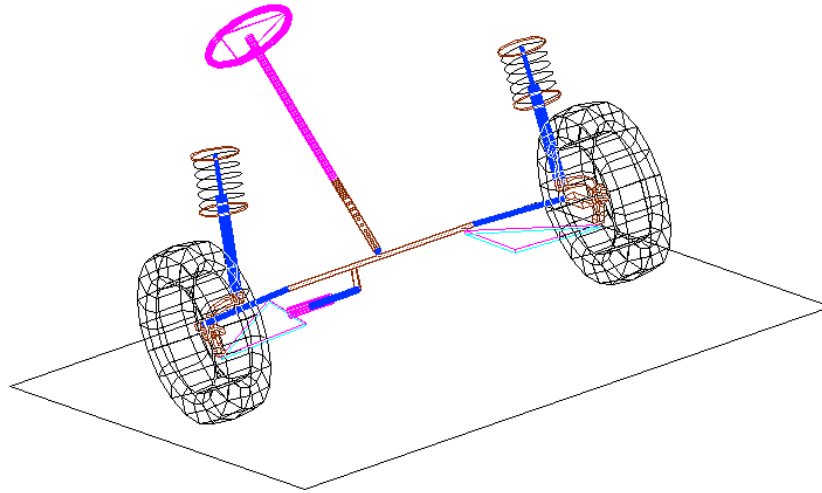
The concept of the virtual prototype has already been mentioned. This concept implies simulation of large systems which will contain subsystems from particular domains. Thus a complete model of a car might contain mechanical (multibody), hydraulic, pneumatic, electrical and thermal subsystems. Traditionally software has been developed for particular domains. Much very valuable simulation can be performed with this domain specific software. However, it is possible that there are complex interactions between subsystems from different domains. These interactions cannot always be ignored. This leads to the concept of multidomain or mechatronics simulation.

How then is this to be achieved? One approach is to take a domain specific software and add a capability in other domains. This is done in many software packages including **AMESim**. The approach can be very successful if the 'foreign' subsystem is reasonably simple.

The limitations of this strategy are that a software producer will have strong expertise in a particular domain and very limited expertise outside of this area. To produce a good collection of submodels in a new domain requires an enormous investment in time and money. It would also produce great duplication of effort.

There is another more fundamental objection. Software from each domain has a specialized user interface which has developed over the past decade. This user interface is not the result of chance but rather the result of genuine needs. To illustrate this point compare the typical multibody software user interface with the typical hydraulic software interface. The multibody software has the following characteristics:

- ↳ It is very geometrical with lengths and angles represented to scale.
- ↳ It is 3-dimensional and often axes are displayed.
- ↳ It is useful to be able to rotate the axes and display the system from a number of view points.
- ↳ Animation is extremely important and is often the main way of displaying results.



**Figure 6: the multibody part of a power assisted steering system modeled using ADAMS**

In contrast for the hydraulic software interface:

- ↪ The display is a schematic and lengths and angles are not represented to scale.
- ↪ The physical system will be 3-dimensional but the schematic is mapped onto 2-dimensions.
- ↪ It is not meaningful to display axes.
- ↪ It is not useful to rotate the schematic and display from a number of view points.
- ↪ Animation is unimportant and not normally provided.
- ↪ The main way of displaying results is simple graphs, bode plots etc.

In view of these differences **AMESim** is interfaced with other software to allow more complex mixed domain simulation. Each subsystem is built with the appropriate domain specific software. A combined simulation is performed and results from each domain examined using the postprocessing facilities of each software.

From version 1.5 of **AMESim** an interface is available with the multibody software **ADAMS**. (See Figure 6 and Figure 7) This is particularly useful in automobile applications.

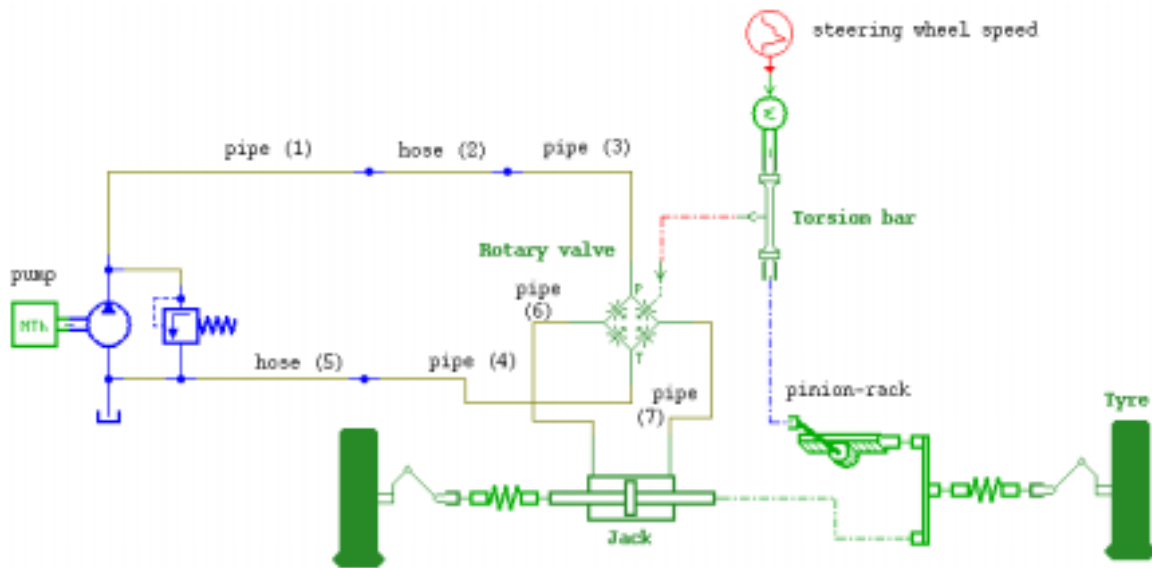


Figure 7: the hydraulic part of a power assisted steering system modeled using AMESim

Also available is an interface between **AMESim** and **MATLAB/Simulink**. This allows the controller design facilities of **MATLAB/Simulink** to be used for a hydraulic system. The non-linear hydraulic system can be linearized about an operating point or the full hydraulic system converted to an S-function for use within **Simulink**. (See Figure 8 and Figure 9)

Interfaces between domain specific software packages is now a reality. Already interest is developing in extending the process to 3 or more packages with **AMESim** providing the hydraulic capability. This is likely to happen in the near future and the virtual prototype will truly have arrived.

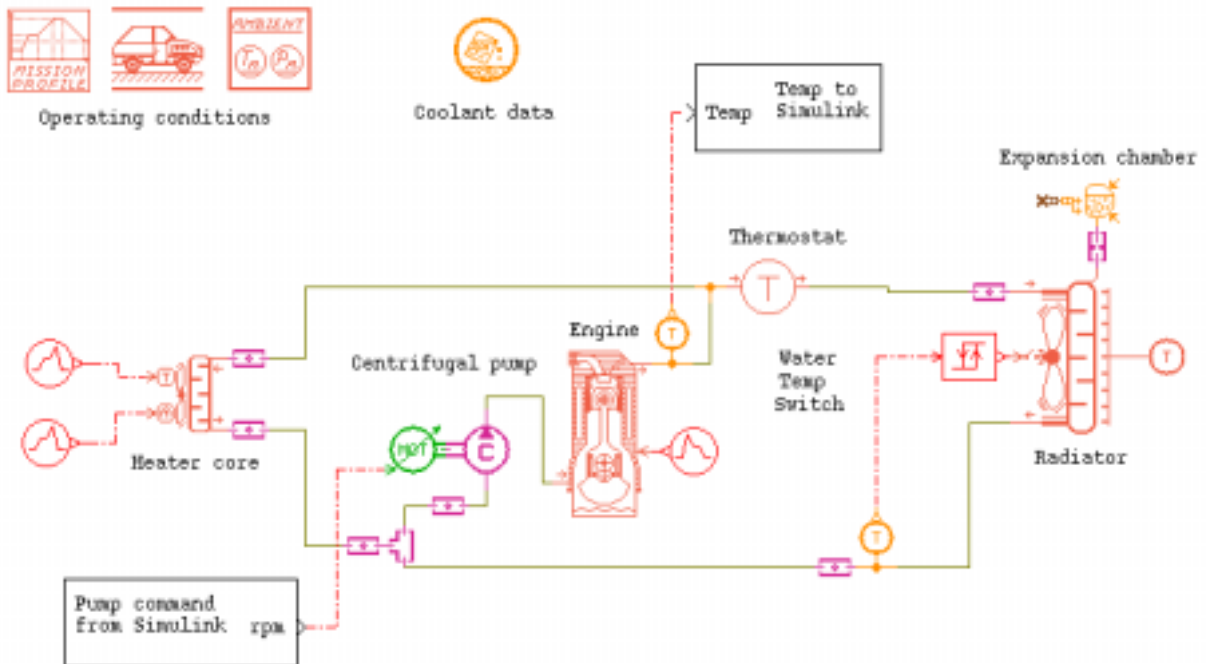


Figure 8: an engine cooling system modeled using AMESim

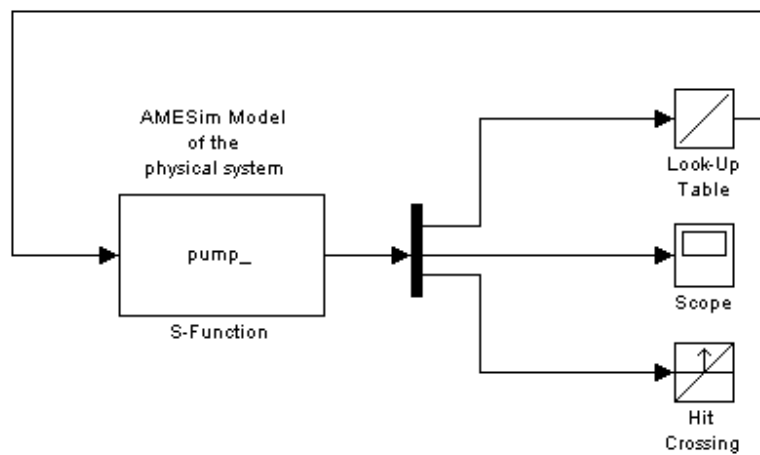


Figure 9: the cooling system imported into Simulink as an S-Function