



NLR-TP-2002-613

Enhancements to EuroSim

R.H. de Vries, J. Keijzer, F. van Lieshout, A.A. ten Dam
and J.M. Moelands





NLR-TP-2002-613

Enhancements to EuroSim

R.H. de Vries(1), J. Keijzer⁽²⁾, F. Van Lieshout(2), A.A. ten Dam
and J.M. Moelands

- (1) Dutch Space B.V., P.O. Box 32070, 2303 DB Leiden, The Netherlands,
Email: R.de.Vries@dutchspace.nl
- (2) Atos Origin Nederland B.V. Business Unit Technical Automation / Advanced
Technology Solutions Papendorpseweg 93, 3528 BJ Utrecht, The Netherlands
Email: {Jan.Keijzer,Fred.vanLieshout}@atosorigin.com
- (3) National Aerospace Laboratory NLR, Information and Communication Technology
Division P.O. Box 90502, 1006 BM Amsterdam, The Netherlands Email:
{tendam,moelands}@nlr.nl

This report is based on a presentation held at SESP 2002: 7th International Workshop
on Simulation for European Space Programmes, at ESTEC, Noordwijk,
The Netherlands on 12-14 November 2002.

EuroSim developments are supported by the Netherlands Agency for Aerospace
Programmes (NIVR).

The contents of this report may be cited on condition that full credit is given to NLR
and the authors.

Customer: National Aerospace Laboratory NLR
Working Plan number: 1.1.A.1
Owner: Dutch Space, Atos Origin and NLR
Division: Information and Communication Technology
Distribution: Unlimited
Classification title: Unclassified
November 2002



Summary

EuroSim[®] (European Real-Time Operations Simulator) is a simulation tool for use by multi-disciplinary teams. EuroSim[®] is designed to meet the needs of hard-real-time simulations but can also perform non-real-time and faster-than-real-time simulations. EuroSim[®] provides the means to create and manage knowledge for complex systems and enables users to transfer this knowledge throughout the full project life-cycle. EuroSim[®] is applicable for multi-user simulation solutions for engineering, training and operational purposes.

In this report the enhancements to EuroSim are presented.



Contents

1	Introduction	4
2	Multi-platform availability increases application in the full project lifecycle	4
3	GUI enhancements	5
4	Real-time enhancements	9
5	Miscellaneous enhancements	10
6	Linux RT version	10
7	EuroSim's Common Object Model (COM) interface	11
8	EuroSim Simulator Development Environment	12
9	References	14
10	Acronyms	15

(15 pages in total)



1 Introduction

Originally developed as a simulator framework on IRIX for Real-Time-purposes, Mk3 version [1] of the Eurosim tool has now been released by the EuroSim consortium - Atos Origin, Dutch Space and NLR. This new release has a significantly improved Graphical User Interface, based on an extensive Human Computer Interface study. During this upgrade, also a Windows-based version was developed, completing the earlier released IRIX & Linux versions.

EuroSim is a versatile simulation tool which can be used in all phases of the development of spacecraft. It can be used from the feasibility stage up to operations. It is especially useful during testing as it has extensive support for hardware-in-the-loop real-time simulations. It supports models written in C, C++, Fortran 77 and Ada, hardware interfaces such as MIL1553, IRIG-B, GPIB, Reflective Memory, etc., configuration control systems like CVS, and much more.

Most of our customers start modelling in MATLAB/Simulink, commonly used in engineering companies and universities. Since 1999 it is possible to transfer MATLAB/Simulink models to EuroSim with MOSAIC [2]. MOSAIC automatically converts model source code that has been generated with the Real Time Workshop (RTW) of MATLAB into model source that can be incorporated in EuroSim, where additional EuroSim specific files are also created automatically. While further detailing and expanding models, long execution times and other barriers are encountered. At that moment, EuroSim helps in a growing path to further enhancements [3].

2 Multi-platform availability increases application in the full project lifecycle

With the port of EuroSim to the Windows and Linux (Real-Time) operating systems a large leap towards application of EuroSim in the full project lifecycle has been made. The port of EuroSim to the various operating systems has been done in such a way that for a EuroSim simulation developer the environment looks the same and has basically the same functionality. This specifically holds for the way the various software simulation tasks are treated during scheduling. This is important, since it causes the simulation engineer early in the project to already adhere to the basic rules that real-time simulation poses in later phases of the project. So by using EuroSim already in the earliest phases of a project, e.g. feasibility studies, later on a smooth transition can be made towards system engineering and operations simulations, in which real-time constraints appear. Since the EuroSim simulation code base can be reused between the various project phases, the total cost of ownership decreases, project-planning risks decrease and shorter schedules are possible.



With EuroSim being available on various platforms, several growth paths appear. One can, for example, start developing EuroSim simulations on the Windows desktops and later on transit smoothly to the PC-based Linux-version, when multi-processing, Real-Time-performance and common hardware I/F's are required. From there the next step can be to switch to dedicated hardware using the IRIX-version. The latter is especially useful where real-time operation (frequencies up to 1000 Hz) and highly demanding visualizations come into play (e.g. training-simulators).

So with the availability of EuroSim on diverse platforms the at least following benefits for the customer are created:

- During all phases, the same models can be employed, greatly reducing the effort of reprogramming earlier (auto-generated) code for Real-Time purposes and Hardware-In-The-Loop testing.
- Since EuroSim fully (and actively) supports ESA's SMP-standard, the continued use of C++ models generated from UML by e.g. Rational Rose or Software through Pictures is ensured.
- All supported platforms share the same GUI design and all files can be re-used, converted or re-built when moving to another platform.
- Procurement of (costly) dedicated hardware can be postponed till all requirements are known and frozen.

3 GUI enhancements

The EuroSim Mk2 GUI was written using the Tcl/Tk GUI toolkit. This toolkit is not very rich in functionality and does not offer many standard widgets. The look of the Tk toolkit is not very modern and is based on the Motif look and feel. Furthermore the existing implementation was not portable to the Windows platform and several deficiencies in the usability were known.

Therefore it was decided to rewrite the entire EuroSim GUI in a new toolkit. This toolkit had to be portable, fast and programmer friendly. The obvious choice is the Qt toolkit from Trolltech. This toolkit has proven itself as the basis of the highly successful KDE desktop environment under Linux. The toolkit has also support for various GUI styles, so that the application looks exactly like the native applications, but can also have the same look as another platform. If the user wants to have a Windows look&feel under Linux, that is no problem.

Comments from existing users, a GUI expert and results from usability tests have been taken into account to produce an optimized GUI. This has resulted in a GUI which is much more compact and requires much fewer mouse clicks to operate than the previous version.

The Project Manager tool originally offered only easy access to the default model. In the current version all files are readily accessible and can be opened directly from the project file overview.

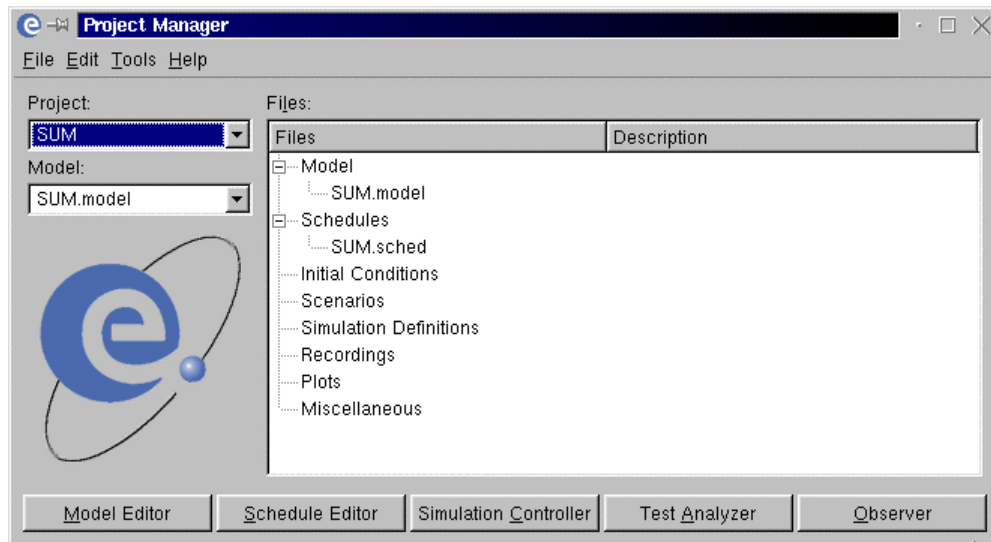


Fig.1. EuroSim Project Manager

The Model Editor originally had a large number of screens, the main model editor window, the API editor window and the build log window a.o. These windows have been merged together in one main window. The API editing function is now integrated into the model tree view. Opening a file node automatically triggers the parsing of the file and loading the API information. Parse errors and compilation errors now go into the log section of the main window.

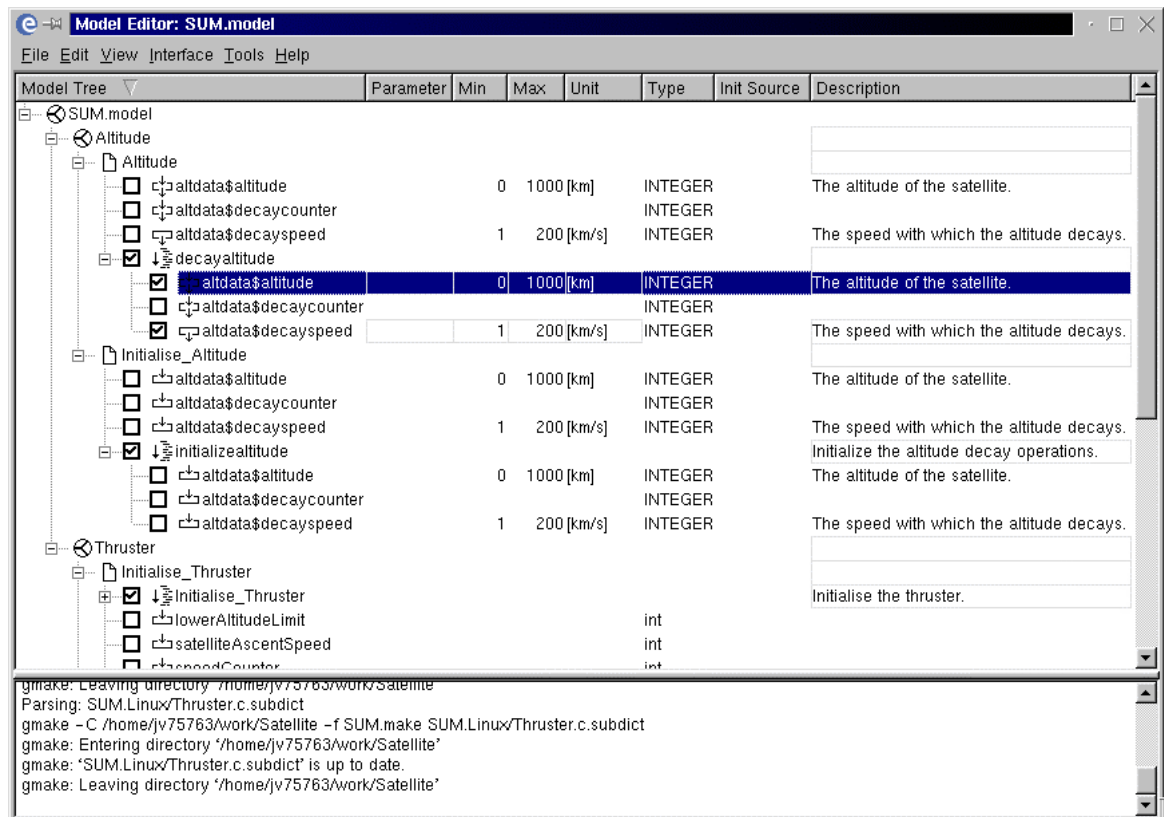


Fig.2. EuroSim Model Editor

The Schedule Editor is now an independent tool. Before the schedule file was part of the model definition. This dependency has now been removed. Users can define multiple schedules for the same simulator. The schedule file does not need to be compiled anymore for the run-time environment.

The Mk2 Mission Tool and Test Controller have been merged into one new Simulation Controller tool. This tool has been enhanced with new Man-Machine Interface (MMI) screens. These screens replace the alphanumeric and graphical monitors in Mk2. One MMI screen can hold multiple monitors. The layout of each screen can be customized by the user to suit his needs. As a simulation configuration can hold many MMI screens it is very easy to create one screen for each purpose. Each screen is placed under a separate tab to allow easy switching between views. It is not needed to create a new monitor if you just want to have a quick look at one variable. A new API tab has been created for that purpose. This tab page shows the entire data dictionary tree of the simulator with the values of each variable. The data is updated at the usual frequency of 2 Hz. It is also possible to change values there.

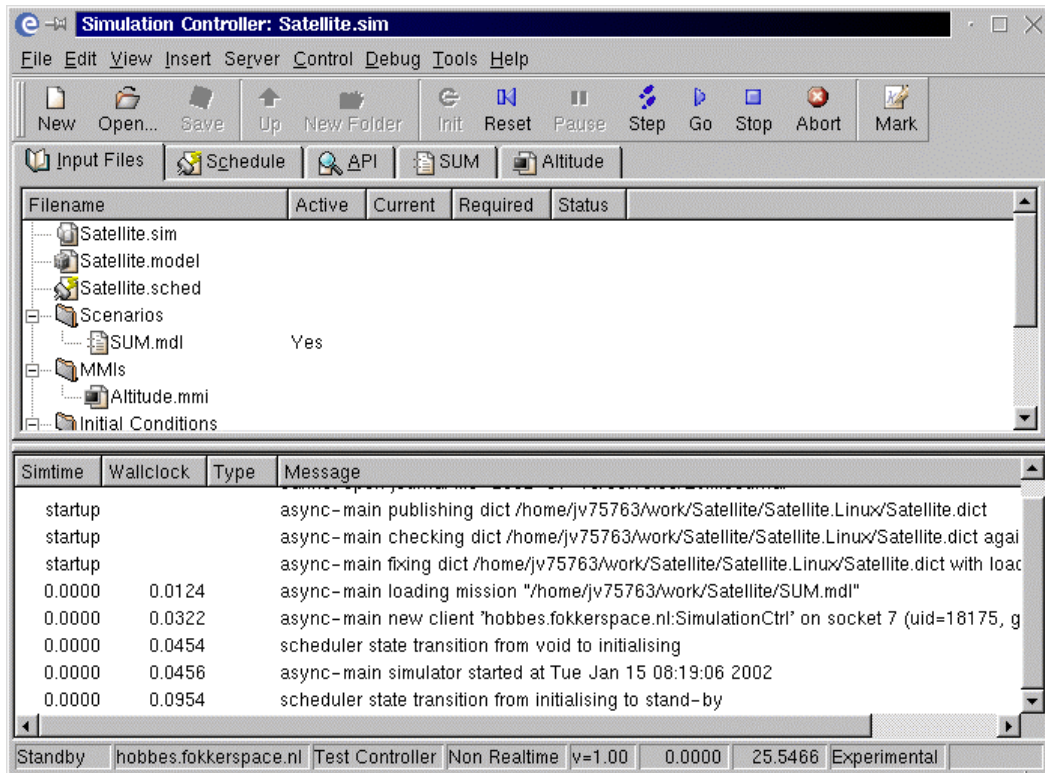


Fig.3. The EuroSim Simulation Controller

The Test Analyzer has been enhanced with a plugin interface. This interface allows the user to choose between two plot engines: one for PV-WAVE and one for gnuplot. EuroSim Mk2 supported only PV-WAVE. PV-WAVE is a commercial package and gnuplot is free. It is now also possible to integrate with other plot engines. A wizard has been added to guide the user through the process of producing a plot.

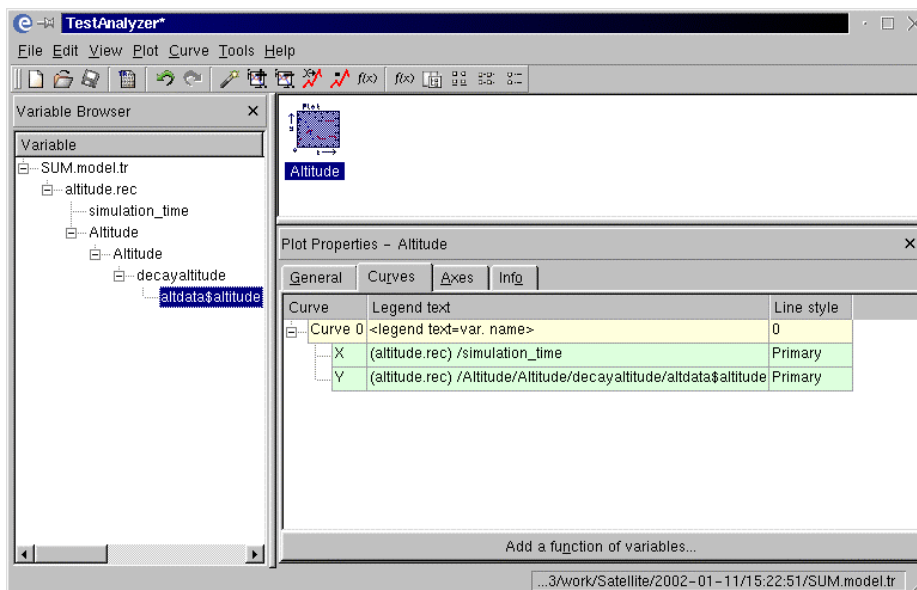


Fig.4. The EuroSim Test Analyzer

4 Real-time enhancements

EuroSim has added support for several hardware interfaces. The IRIG-B interface is used to synchronize the simulator in a large test facility. The scheduler is driven directly by the IRIG-B interface card. This eliminates any drift and assures synchronization over long periods of time (a week). The PCI-VME interface allows integration with VME based systems such as front-ends. The interface allows memory mapping (direct access) to VME memory and interrupts. The Reflective Memory interface card is also used to integrate with front-ends. Reflective memory is used to exchange data with very low latency, which is especially useful in real-time test facilities with a large number of systems connecting to the simulator.

The scheduler has been enhanced to make it easier to start tasks on hardware interrupts. This makes it possible to let the simulation models respond to hardware interrupts within 500 μ s.

For integration of the simulator in a large test facility a timed state transition facility has been added which allows you to start the simulation synchronously with other systems in the facility. The user requests to start the simulation on a certain time in the future (or to stop) and the scheduler will perform the action on exactly that time.

The scheduler can be set to run as fast as possible. This will greatly increase the speed of your simulation if you are only interested in the result of the simulation and not in the precise timing



as would be the case in the earlier stages of a project where the focus lies on engineering. Later in the projects the models can be reused in a simulator used in the test facility.

5 Miscellaneous enhancements

A powerful batch utility has been added to allow users to run simulations in a fully automated manner. The utility is implemented as a perl module. The perl language is a very versatile scripting language which is commonly used to perform text processing, web site scripting and various other tasks. The utility can be used in conjunction with other perl modules to create large applications automating complete test procedures. Automated tests can be used for regression testing or for running tests repeatedly with varying start conditions for sensitivity analysis.

The Simulation Model Portability (SMP) standard is developed under the responsibility of ESA. It aims at standardizing the interface between simulation models and the simulation infrastructure (EuroSim). Support for this model interface allows users to develop models which can run on other platforms or to reuse models developed for other SMP compliant simulation environments.

Configuration control is an important aspect in any software engineering project but especially so for test facilities where it is very important to be able to trace exactly which version of the simulator you have used for a particular run. EuroSim Mk2 only supported the Cadese configuration control system. EuroSim Mk3 has added CVS (Concurrent Versions System) support. CVS allows multiple users to access a source code repository in a controlled manner. It is free and widely used.

6 Linux RT version

EuroSim has been available on the Linux platform since 2 years now. This version was not capable of running hard real-time. This capability was only available on the Silicon Graphics hardware and operating system (IRIX). It requires a multi-processor machine to run hard real-time. The SGI hardware is quite expensive and users were immediately interested in having a real-time version running on the cheaper PC hardware.

Using a multi-processor x386 based computer running Linux, it is also feasible to achieve hard real-time performance with some adaptations to the standard Linux operating system. These



adaptations remove the few remaining sources of non-real-time behavior of the OS. The result is a EuroSim simulator running hard real-time with an even better stability than achievable under IRIX.

Further work has been done to port various drivers needed for hardware in the loop simulations to the Linux platform. Currently, drivers for the following interfaces are available or under development: IRIG-B, PCI-VME bridge, Reflective Memory and MIL1553.

7 EuroSim's Common Object Model (COM) interface

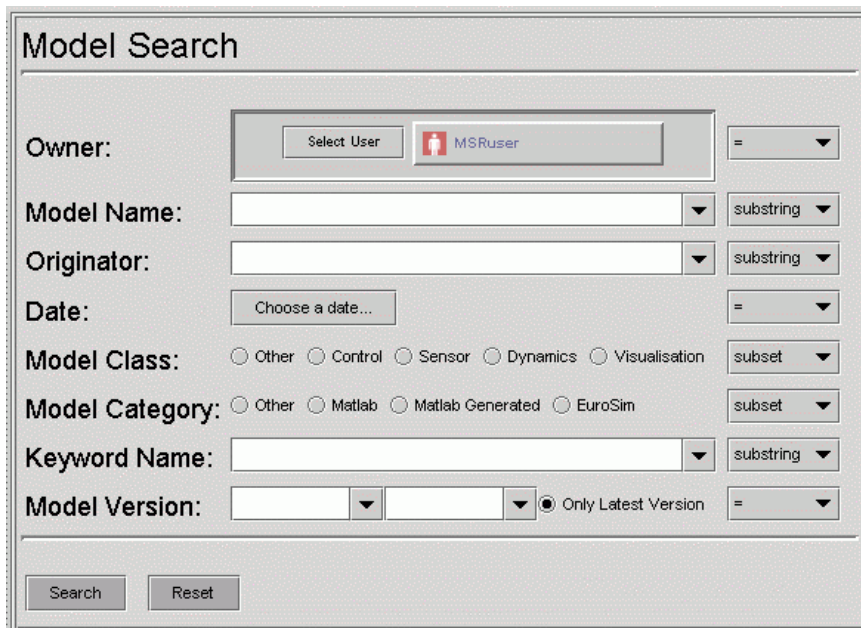
EuroSim also includes a Common Object Model (COM) interface component. The common Object Model is the interface standard of Microsoft, which ensures that applications on at least the Windows platforms can interact and exchange information. With the COM interface for EuroSim we enable the users to control and to monitor simulations from other (client-) applications. These applications are all applications that can communicate through COM. As an example applications written in Visual Basic or applications that use Visual Basic for Applications, such as Microsoft Excel or Microsoft Access, can now control and monitor EuroSim simulations. It is, for example possible to start a EuroSim simulation from an Excel workbook and store the results of that (real-time) simulation in the very same workbook. The simulation itself can run on every platform that EuroSim currently supports, for example, a real-time simulation on an IRIX machine. For the users this means that a simulation infrastructure based on the Windows platform can be extended with real-time simulation capabilities without the need to port the rest of the infrastructure to the real-time platforms. Again this decreases the total cost of ownership for simulation projects and diminishes project-planning risks.

The COM interface gives customers the flexibility to write their own dedicated client applications, which give the end-users control and access to only those parts of the simulation that they are allowed to. The COM interface component has support for on-line help in VB/VBA and example applications, allowing rapid client application development.

8 EuroSim Simulator Development Environment

In large (space) programmes the road from feasibility study to operations and/or training may be long and demanding. During a typical programme many application models will be developed, and many simulators will be built to study each aspect of the system under consideration. The EuroSim Simulator Development Environment (SDE), part of EuroSim Mk4, provides a way to make sure that you can trace all models that have been used in the past, and integrate them in an easy way into a EuroSim simulator. The SDE consists of the Model & Simulator Repository (MSR) and Simulator Composition Tool (CT).

The MSR provides the means to store and retrieve in a coherent manner all information that is relevant with respect to models. This information includes for example model code, design documentation, test information, user manuals, and related information. Integrated configuration management enables the storage of multiple versions of a model or simulator. Smart algorithms reduce the disc space used when storing multiple versions of an object. Proprietary rights are protected by means of different levels of visibility of objects stored in the MSR. A dedicated search form is available to retrieve models and accompanying information.



The screenshot shows a web-based search interface titled "Model Search". It contains several search criteria with corresponding input fields and dropdown menus:

- Owner:** A "Select User" button and a dropdown menu showing "MSRuser".
- Model Name:** A text input field and a dropdown menu set to "substring".
- Originator:** A text input field and a dropdown menu set to "substring".
- Date:** A "Choose a date..." button and a dropdown menu set to "=".
- Model Class:** Radio buttons for "Other", "Control", "Sensor", "Dynamics", and "Visualisation", followed by a dropdown menu set to "subset".
- Model Category:** Radio buttons for "Other", "Matlab", "Matlab Generated", and "EuroSim", followed by a dropdown menu set to "subset".
- Keyword Name:** A text input field and a dropdown menu set to "substring".
- Model Version:** Two dropdown menus and a radio button labeled "Only Latest Version" which is selected, followed by a dropdown menu set to "=".

At the bottom of the form are "Search" and "Reset" buttons.

Fig.5: Example of the MSR search (screen capture)

Although the MSR is a distributed system, a user experiences the MSR as a single logical entity. An MSR user accesses the MSR using web-based technology, using the Internet and/or a secure network for connection to a central MSR server. The graphical user interface of the MSR uses Java and can therefore be used on almost any platform. All communication between the MSR

components is based on the World Wide Web Consortium (W3C) defined Extensible Markup Language (XML) standard. All MSR models are stored in a database. The MSR uses the JDBC protocol to access this storage medium. This makes the MSR independent of specific vendors and almost any database, commercial as well as public domain, can be integrated in the MSR.

EuroSim's SDE also includes the Composition Tool, which acts as a model integration platform. The CT has a user-friendly GUI to create simulators from a number of models in an object-oriented fashion. The models the simulation will contain can be selected, e.g. from the MSR and placed on the simulator's canvas. Input and output variables in the model's interfaces can be interconnected through data flows in a graphical way. No EuroSim schedule needs to be created by hand, as the data flows implicitly define the default scheduling order. If any more information is needed for the creation of the schedule (e.g. scheduling frequencies), the CT prompts for this information. Closed loops in the composed simulator are detected automatically and the user is able to select the model that resolves the loop by acting as an initial starting point.

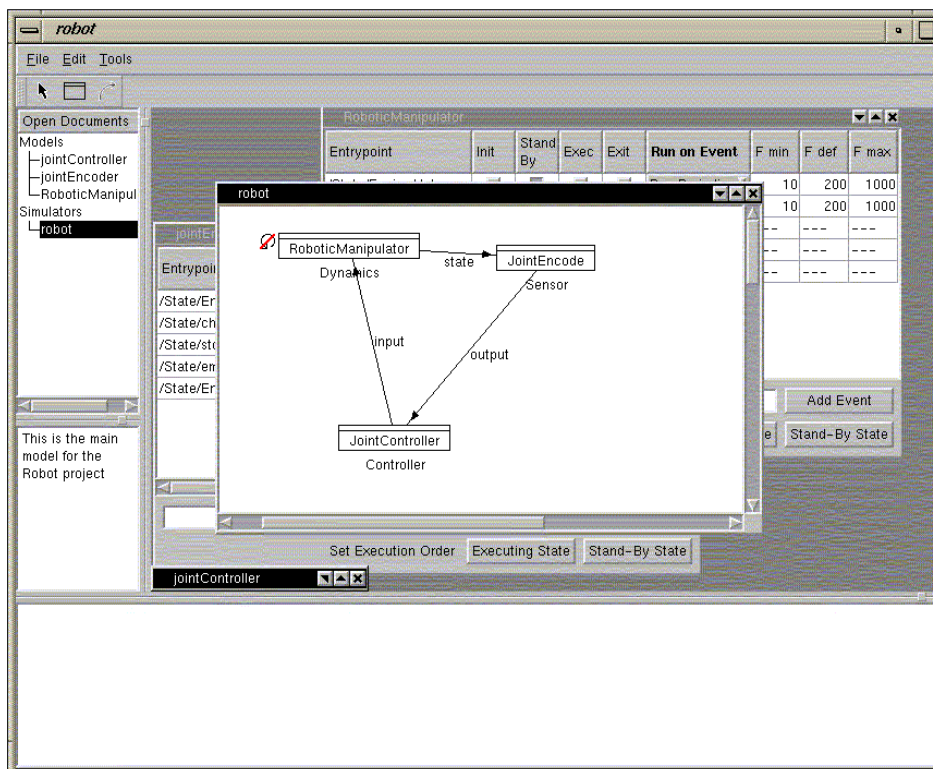


Fig.6. The EuroSim Composition Tool (screen capture)

The Model & Simulator Repository acts as a central storage for your project's or company's models, encouraging model reuse. The architecture of the MSR supports a distributed way of working in recognition that many of today's projects are executed in consortia that are located at



various places in Europe or the world. Integrating models in a simulator will be much easier than before when using the EuroSim Composition Tool. Creating a simulator from existing models can reduce simulator development time and cost considerably.

9 References

- [1] EuroSim product information: URL: www.eurosim.nl.
- [2] W.F. Lammen, A.H.W. Nelisse and A.A. ten Dam, "MOSAIC: Automated Model Transfer in Simulator Development", Proceedings 7th International Workshop on Simulation for European Space Programmes SESP 2002, pages 281-286, 12-14 November 2002, ESTEC, Noordwijk, The Netherlands, also NLR TP-2002-628.
- [3] E. Mooij and Q. Wijnands, "Generic Attitude and Orbit Control Simulator development supporting the AOCS software lifecycle", Proceedings 7th International Workshop on Simulation for European Space Programmes SESP 2002, pages 303-315, 12-14 November 2002, ESTEC, Noordwijk, The Netherlands.



10 Acronyms

API	Application Programming Interface
COM	Common Object Model
CT	Composition Tool
CVS	Concurrent Versions System
ESA	European Space Agency
EuroSim	European Real-Time Operations Simulator
GPIB	General Purpose Interface Bus
GUI	Graphical User Interface
I/F	Interface
IRIG	Inter-Range Instrumentation Group
JBDC	Java Database Connectivity
KDE	K Desktop Environment
NLR	National Aerospace Laboratory NLR
Mk	Mark
MMI	Man Machine Interface
MOSAIC	Model-Oriented Software Automatic Interface Converter
MSR	Model & Simulator Repository
OS	Operating System
PCI	Peripheral Component Interface
RTW	Real-Time Workshop
SDE	Simulator Development Environment
SMP	Simulation Model Portability
UML	Unified Modelling Language
VB	Visual Basic
VBA	Visual Basic for Applications
VME	Versabus Motorola Europe
W3C	World Wide Web Consortium
XML	Extensible Markup Language