ADVANCED TRAINING SIMULATOR DEVELOPMENT TOOLS FOR WINDOWS™

C. A. Jones, N. S. Yee, and G. F. Malan
Framatome Technologies
Lynchburg, Virginia 24506-0935

Key Words

Real-time simulation, Energy, Trainers

Abstract

Compact simulators provide a cost-effective platform for training operators in power plant operation. Traditionally, the development and maintenance of a compact simulator system involves many manhour intensive tasks. Framatome Technologies has produced a suite of high-level tools, designed to run under Microsoft Windows™, that automate many of these tasks. Each member of the suite of tools is discussed along with how it fits into the typical development cycle of a compact simulator.

Introduction

The development of a compact simulator system requires a large amount of plant data being available to the developer. This data includes the plant's physical configuration, component information, and control-system configuration. Bringing this data together into a compact simulator involves the creation of two major systems: the process model and the man-machine interface. The process model provides a dynamic response of the system for the simulator. Process models for Framatome Technologies' compact simulators incorporate the control logic and all physical modeling (e.g., pipes, pumps, turbines) based on first principles into the process model. The man machine interface (MMI) allows for the operation of the simulated plant model. This may involve one or more distributed control systems (DCS) emulations and/or "hard panel" emulation.

Creation of a physical-process model from plant drawings and component data is a complex task. Although it is possible to write a complete custom program to simulate a plant, the use of a component-oriented modular modeling library can reduce this effort. For plants that are in operation, data collected at the plant during transients and steady-state conditions can serve to verify the model.

Many stations now have DCS that provide an MMI to the operators of the plant. This MMI information is specific to each plant, and in most cases, is quite extensive. In simulators where the actual MMI system is emulated, conversion of the information to a format usable by the emulated system can require significant effort.

Once the constituent pieces are combined, the complete simulation is tested. Communication between the process model and the MMI is provided by a pair of companion programs designed to work with all simulations developed with this set of tools. An instructor station is another basic functionality provided with training simulators.

Development Tools

Tools to be discussed in this section include:

- Component Modeling Library
- Graphical Model Builder
- Control Logic Translators
- DCS MMI Translators.

Process model creation uses the first three tools in the above list. Translation of the DCS information into the corresponding MMI programs makes use of the DCS MMI Translators. Steps necessary to build hard-panel emulations are discussed with the DCS MMI translators.

Process Model Creation

The power plant component-oriented modeling library is commercially available through Framatome Technologies as the Modular Modeling System (MMS). Engineering analysis continues as the primary application of the MMS. To meet the real-time constraints of compact simulation, a new set of modules, based on the MMS Release 4.0, known as the MMS Real-Time Capable

(MMS-RTC) library is used to create the process models. MMS makes use of the Advanced Continuous Simulation Language (ACSL) from Mitchell and Gauthier Associates (MGA, 1991). The simulation source code is an ACSL source code with calls to pre-defined ACSL macros from the MMS-RTC library.

Process model source code generation is greatly aided by the use of a graphical program known as the MMS Model Builder (Jones , 1992). Release 2.0, shown in Figure 1 is due for release in the first half of 1995. Models are built by placing graphical icons on the worksheet representing a module from the MMS-RTC library. Once the components are placed, connections are made between the ports on the modules to define the flow of fluids, control signals, heat transfer, or shaft power. Because the source code is generated automatically, source code errors are greatly reduced. Another feature of the model builder that provides a great reduction in effort is a feature known as "Auto-parameterization." Each module has an interactive input form that allows the entry of component data (i.e., pipe length, pipe ID). Auto-parameterization calculates values like flow conductance, heat-transfer parameters, and other parameters needed for simulation to achieve a steady-state condition. Many of these parameters require the engineer to look up values in water tables or other references. Auto-parameterization routines have all of the necessary routines built-in to allow a calculation that would manually take minutes to hours to be done in a few seconds.

All of the module and connection names can be set from the Model Builder. This allows the designer to match the internal names of the simulation with the names used by the customer in their documentation as closely as possible. Other features such as pan and zoom; split views; cut, copy and paste; print preview and printing; and documentation through a text output/input feature, provide an increase in modeling efficiency. A major new feature of Release 2.0 of the Model Builder is Hierarchical Blocks. Hierarchical Blocks allows a group of modules to be reduced to a single icon. When a user clicks on the reduced icon a new window opens and allows the user to edit the modules inside the block. This makes representation and duplication of complex systems much easier.

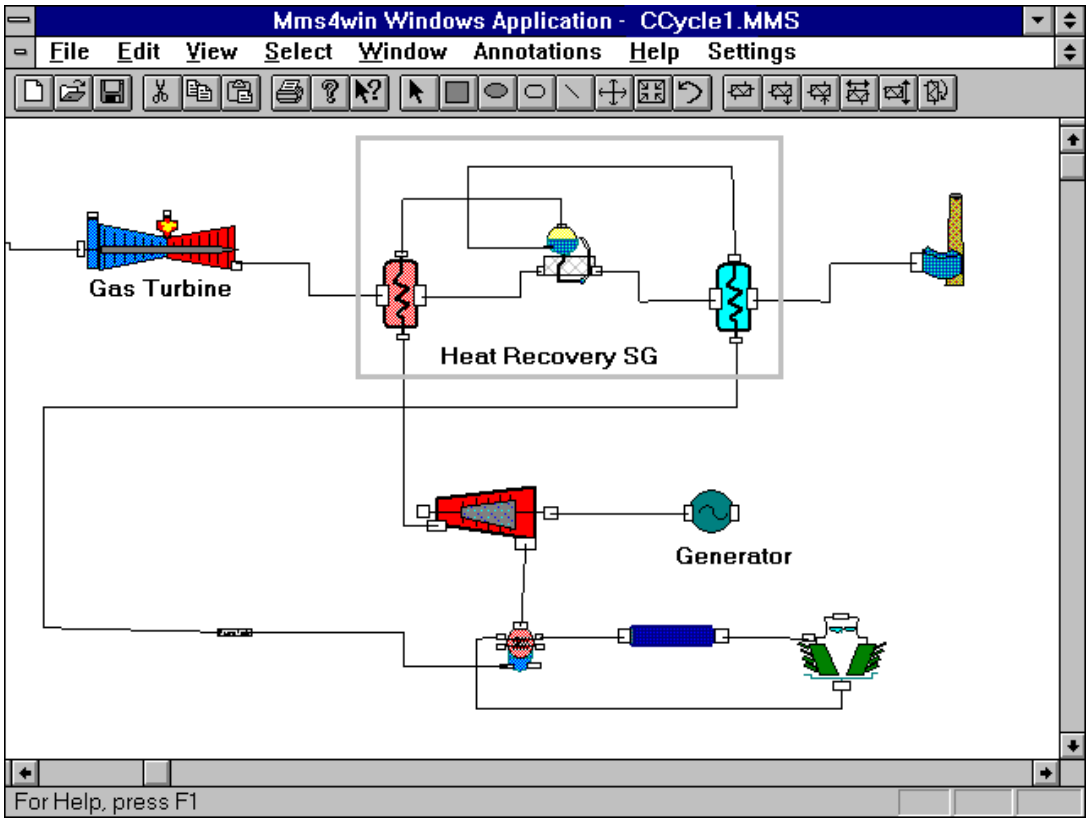

Figure 1. MMS Model Builder

In plants with a DCS, a Control Logic Translator converts the control system configuration data directly into source code that is merged with the source for the physical-component process model. The same Windows™ interface program known as the MMS DCS Translator is provided for all translators. After the MMS DCS Translator is started, a translator "personality" is loaded to translate the particular type of input desired. In this way the user has a common interface from which to work. Input to the translator is an ACSII file containing the DCS configuration of interest. Currently the translation of control logic for the Honeywell TDC-3000 system is supported.

Source code for the physical process model and the control logic are merged into one source code. This code is then used by the ACSL translator to produce a FORTRAN file. The FORTRAN source code is then linked with a set of MMS-RTC runtime routines to produce the process model executable. ACSL provides a runtime executive with a command-line interface for control and testing of the model. Control system and other analyses are part of the ACSL feature set which expands the ability of the developer to test and tune the model for maximum performance and stability.

MMI Emulation

Personalities for DCS MMI translations are also available for the MMS DCS Translator. Westinghouse WDPF and Honeywell TDC-3000 systems are currently supported. Files from the plant's DCS system are transferred to the PC and the translator personality for the specific system creates source code for the MMI screens. The current version of the translator creates source code for Microsoft's Visual Basic (Microsoft, 1993) which has the added value of providing an open system development environment and has the availability of complementary third party libraries and supplemented software. The use of Visual Basic provides a number of benefits for the developer and the user. The Visual Basic programming interface is completely graphical with very high-level graphical and source code editing features. Visual Basic also has a complete interactive debugger, project control, and the ability to create compiled executables. The combination of these features provides a great deal of development
support as well as a high-level interface to the user who needs to modify any of the MMI screens.

When the DCS MMI translators are used with the related control logic translator, all of the variable names in the source code are based directly on the names used by the DCS. This feature is a great benefit in debugging of the process model. Because the names produced by the control and MMI translators are the same, no name-matching or translation is needed for connection from the MMI to the process model.

The coverage, that is the amount of information recovered from the DCS source files, is very complete for the MMI translators. Hot-spot information, visual feedback (e.g., color change, blinking, sound), and any other logic in the MMI files are automatically picked up and converted to Visual Basic source code.

Simulators that require hard panel emulations benefit from the fact that the same high-level interface is available to create custom screens. Users can create new screens from scratch or modify the source code output from the DCS MMI translator. Visual Basic supports the loading of scanned images and creation of "hot-spots" in a user friendly manner.

Figure 2 is the typical compact simulator configuration. The network can be easily configured to support additional features or special requirements.

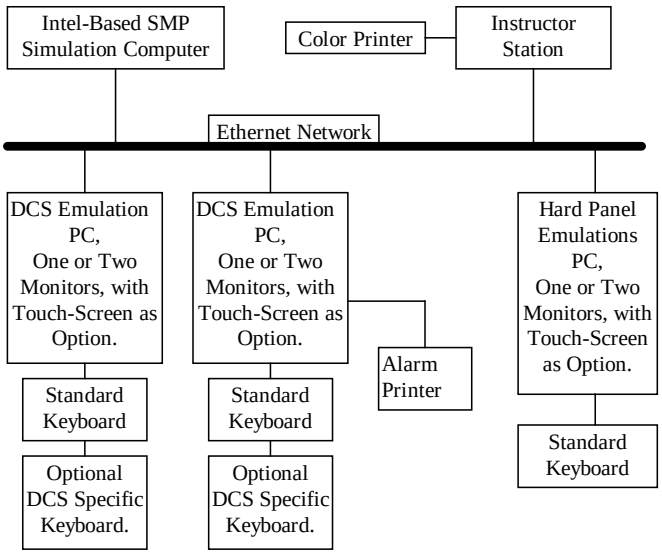

Figure 2. Typical Configuration

Runtime Configuration and Tools

Runtime tools provide the necessary communication and control support for the simulator. These tools provide the means of connecting the process model and MMI emulation. An Instructor Station allows the person conducting the training to control the simulation and to

build and introduce system malfunctions. Process model execution takes place on an Intel-based Symmetric Multi-Processor (SMP) machine running Windows NT™. Machines of this type are currently available with one to four processors. All other machines are single-processor, Intel-based, machines running Windows™ or Window NT™.

Control and communication between the process model and the remote machines is provided by a program known as Simulation Master. Simulation Master runs on the same machine as the process model and supports any network configuration and protocol stack. TCP/IP is the current default protocol and either ethernet or Token-Ring can be supplied as the network.

A companion program that controls the MMI screens on each MMI workstation is known as the MMI Manager. As data is received from the process model, the MMI Manager places the data in memory so that the MMI screens can be updated. Then the MMI Manager sends a message to all currently running MMI screens that an update is to occur.

The MMI Manager also supports all custom keyboard and touch screen support. Framatome Technologies supports an emulated version of the WDPF keyboard and uses Honeywell keyboards directly. Capacitve touch-screen monitors are shipped with current simulators; however, other types can be used.

Figure 3, shows the MMS Instructor Station. The instructor can start, freeze and restart the simulation from the instructor workstation. Snapshots can be taken of the simulation and used to backtrack the simulators states. Malfunctions, trend charting and other predefined actions are available from menus. Any value from the simulator can be viewed and any constant variable value can be changed using the instructor workstation.

Complete Development Cycle

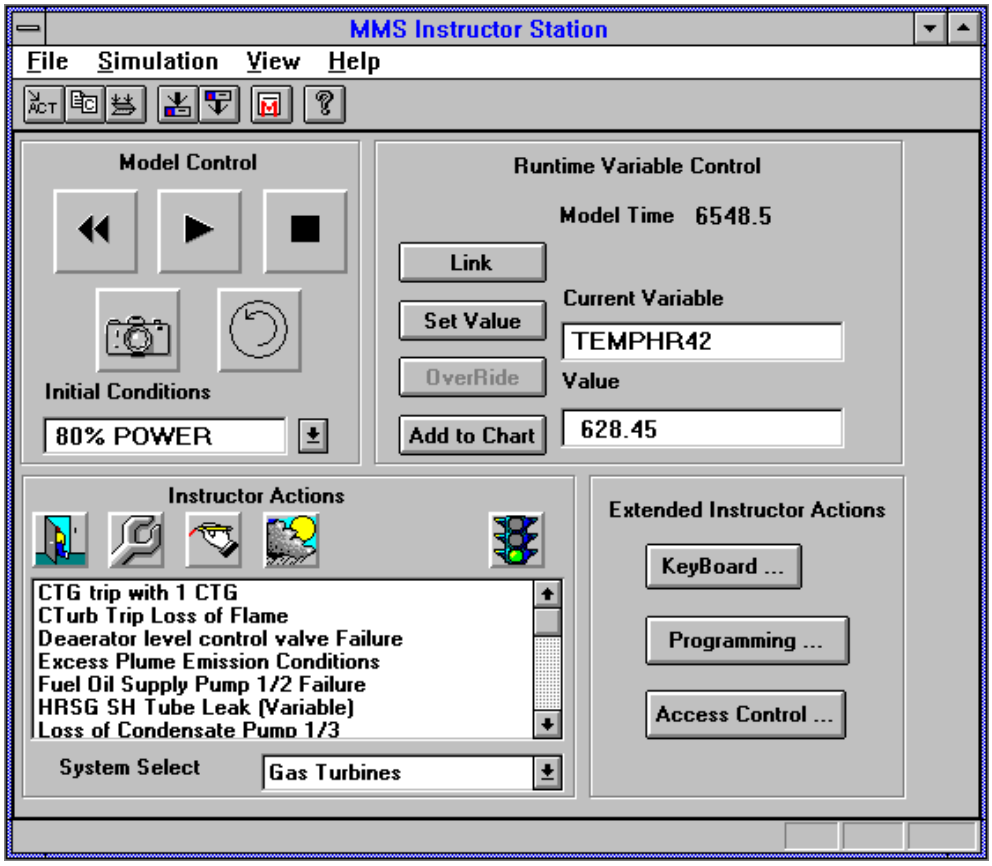

Figure 3. MMS Instructor Station

Figure 4 shows the complete cycle and how the discussed tools are involved. Three development activities can occur in parallel. One teams of engineers collects data and completes the modeling of the physical plant by using the MMS Model Builder program. A second effort, which usually requires a single engineer, is the

translation and verification of the DCS control logic source files into source code. The last major effort is to translate the DCS MMI information into source code for the MMI emulation programs.

Conclusions

Use of the Framatome Technologies suite of Windows™-based compact simulator development tools greatly reduces the effort in creating a compact simulator. Automation of many of the tasks, such as Control Logic and MMI translation, reduces the number of errors introduced in hand coding. Using these tools with the runtime control programs (Simulation Master, MMI Manager, and the Instructor Station) provides a complete simulation environment.

References

Mitchell & Gauthier Associates Inc., 1991. *Advanced Continuous Simulation Language Reference Manual.*

Jones, C. A., 1992. *8th Power Plant Dynamics, Control & Testing Symposium.* University of Tennessee, College of Engineering. Knoxville, Tennessee.

Microsoft Corporation, 1993. *Microsoft Visual Basic Language Reference.*

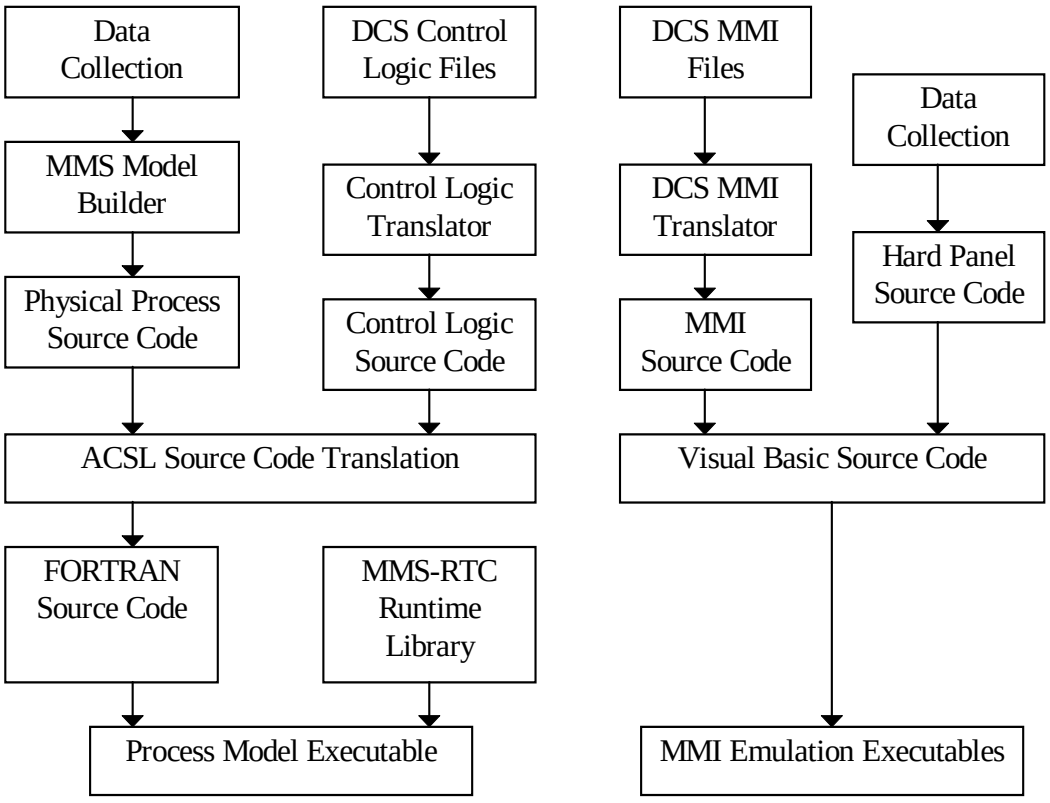

Figure 4. Compact Simulator Development