

Development of a Dual-Extraction Industrial Turbine Simulator Using General Purpose Simulation Tools

Philip S. Bartells
Director, Application Engineering
AEGIS Technologies Group, Inc.
631 Discovery Drive
Huntsville, AL 35806

Christine K Kovach
Sr. Engineer, Application Engineering
AEGIS Technologies Group, Inc.
631 Discovery Drive
Huntsville, AL 35806

KEYWORDS

Turbine-Generator Modeling, Simulator, Simulation, Software, Open Systems, Application Programming Interface, API, Graphical User Interface, GUI, SAMA control diagrams.

ABSTRACT

This paper describes the methodology with which a detailed simulation of an industrial turbine was developed for use in training and analysis. The turbine-generator system that was simulated is a vintage dual-extraction condensing steam turbine. It is controlled by a Programmable Logic Controller (PLC), which is interfaced to a Digital Control System (DCS) that provides supervisory control and operator screens.

Many of the personnel who have operated this turbine-generator for the last 30 to 40 years are nearing retirement age, and the customer felt it was important to have a tool which could be used to familiarize new personnel with turbine-generator operations, and to provide training for current operations personnel on how to respond to system upsets. In addition, it could be used to evaluate system design changes prior to them being implemented on the real system.

The customer desired a simulation platform that was "open", that is, not dependent on a control vendor or special hardware. It needed to run in real-time to be useful for training, and it was also required that it be on a personal computer platform utilizing the Windows operating system. The simulator was constructed using general-purpose simulation software and graphic user interface (GUI) tools. The simulator system was delivered on a networked three-PC platform, with touch-screens and a special keyboard for emulating the actual DCS keyboard. The resulting simulation includes very detailed mathematical models of the turbine and generator and auxiliary systems.

INTRODUCTION

This paper describes the methodology with which a detailed simulation of an industrial turbine was developed for use in training and analysis. The customer and powerhouse are typical of what is found in the Pulp & Paper industry. The approach taken could be applied to similar situations. This paper focuses on the technical aspects of the development process rather than the use of the completed product. At the time of performance of this work, Mr. Bartells was an independent simulation consultant. He assumed his current position in January 2002.

The turbine-generator system that was simulated is a vintage dual-extraction condensing steam turbine. It is controlled by a Programmable Logic Controller (PLC), which is interfaced to a Digital Control System (DCS), which provides supervisory control and operator screens.

Many of the personnel who have operated this turbine-generator for the last 30 to 40 years are nearing retirement age, and the customer felt it was important to have a tool which could be used to familiarize new personnel with TG operations, and to provide training for current operations personnel on how to respond to system upsets. In addition, it could be used to evaluate system design changes prior to them being implemented on the real system.

The customer desired a simulation platform that was “open,” that is, not dependent on a control vendor or special hardware. It needed to run in real-time to be useful for training, and it was also required that it be on a personal computer platform utilizing the Windows operating system. This was required so they could maintain it themselves.

A long-term goal was to develop a simulator capability that included all of the major equipment that the operators had to deal with. This project followed the development of a Combination Boiler simulator (ISA 2001 technical paper 1063).

SELECTION OF SOFTWARE TOOLS

Use of a commercial simulator vendor system was ruled out early due to the cost of a software license, and the requirement to have the simulator vendor perform a significant part of the work. The budget did not allow for this approach and a lower cost alternative was needed.

Review of available simulation platforms was done, and the decision to use the Advanced Continuous Simulation Language (ACSL) Suite of Programs was made, for reasons that are discussed below.

The authors had many years of experience with ACSL, and felt confident that it would be able to handle the detailed model that would be required to realistically simulate the turbine-generator and related systems, especially in real time. In addition, an associate had developed a library of component level models that could be used to simulate the turbine-generator and associated systems, and these were compatible. As it turned out, it is extremely doubtful that the alternative general-purpose simulation platforms could have handled this very complex model and run in real time and faster.

Visual Basic was used to emulate the Operator Screens that are on the DCS. They were interfaced to the simulation using an Application Programming Interface (API) that was based on Object Linking and Embedding (OLE2) as the data exchange method. This API is a set of routines that allows programs written in Visual Basic, C or C++ to control an ACSL-based simulation, and to send data to it, and retrieve data from it. Figure 1 shows the software layout.

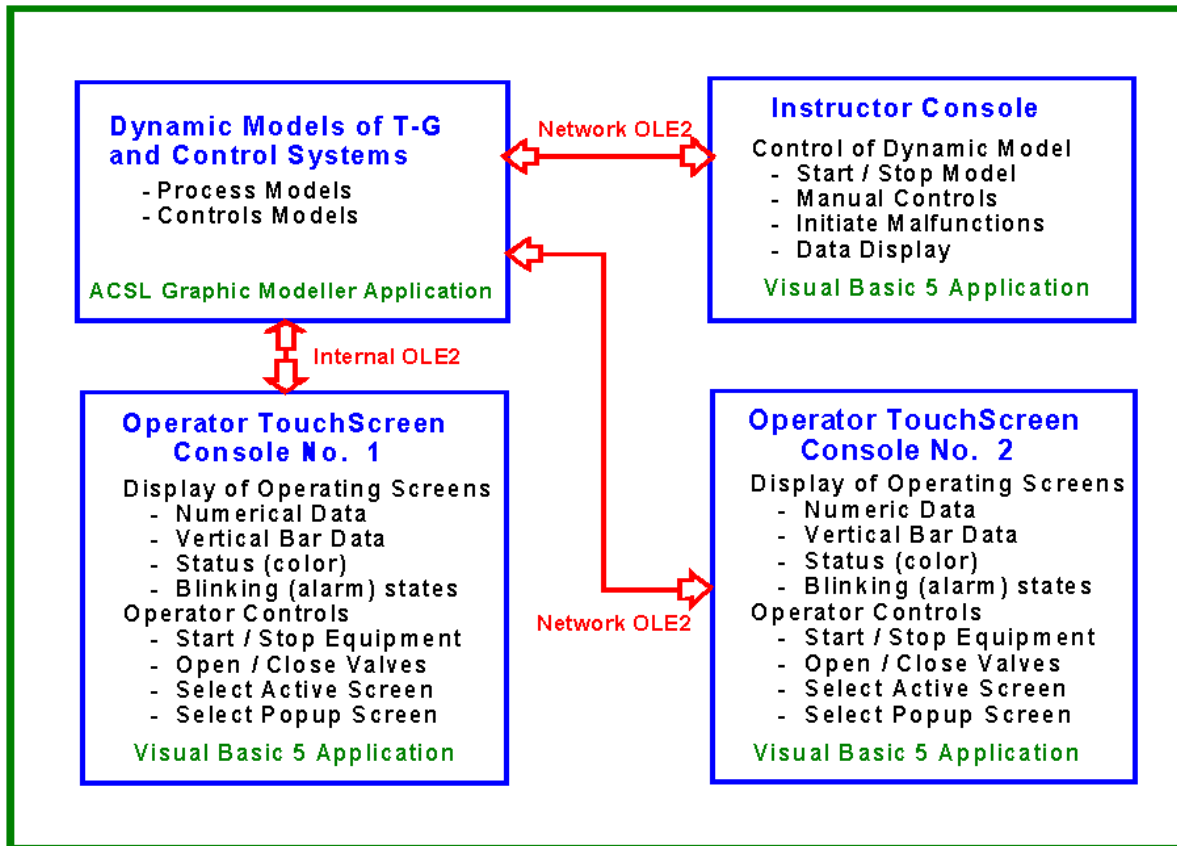


FIGURE 1 –TURBINE-GENERATOR SIMULATOR SOFTWARE LAYOUT

HARDWARE PLATFORM

The initial project started with the purchase of two Personal Computers. For the Operator Station a 20-inch touch-screen monitor was purchased. This system was delivered in mid-1997 and installed at the mill. As subsequent projects were undertaken, the system grew to include two Operator Stations and one Instructor Station.

Each Operator Station is comprised of a PC, a 20-inch touch-screen monitor, a mouse, and a special keyboard that is used to emulate the DCS keyboard. This special keyboard is an array of miniature

pushbuttons, and when a key is pressed, a signal is sent to the PC via the serial port. Each key has its own identifying number, so the software reads the number and performs the operation assigned to that number. The Instructor Station is comprised of a single PC with a 17-inch monitor and mouse. From this station the Instructor can start and stop the simulation, change to fast time or real time, initiate malfunctions, etc.

DEVELOPMENT

To have the simulator result in an effective training tool, it was necessary to perform the following tasks:

- X Model the turbine-generator in great detail;
- X Model the auxiliary systems;
- X Duplicate the control system logic;
- X Duplicate the Operator screens;
- X Interface all the pieces and get them working together;
- X Test the result against actual data and/or operating experience.

TURBINE-GENERATOR MODELING

It was necessary to model the turbine-generator in great detail if it could be expected to react to upsets or operator inputs in a realistic manner. A library of FORTRAN-based subroutines was utilized to develop the model of the turbine-generator and related systems. This library is proprietary and was the only piece of non-commercial software that was utilized.

System descriptions and operating procedures were gathered and, together with test data, were used to build the model and test the resulting simulation. This involved development of the model, written in Continuous Simulation Language (CSL) code. This model code contains calls to the appropriate subroutines and includes the data necessary to make the subroutine into a mathematically equivalent representation of the real equipment. The resulting code was documented with a block diagram

Figure 2 shows some of the steam circuits and is just one of several diagrams. Flow stream variables are defined on the block diagram and it is used extensively at run time for debugging purposes.

Also included are routines for calculating steam and water properties. The model inter-connection network is then solved using a pressure-flow matrix solution technique, which utilizes the mathematical system of equations and partial derivatives. It is beyond the scope of this paper to describe this solution technique in detail, but it is very similar to that used by the commercial simulator vendors.

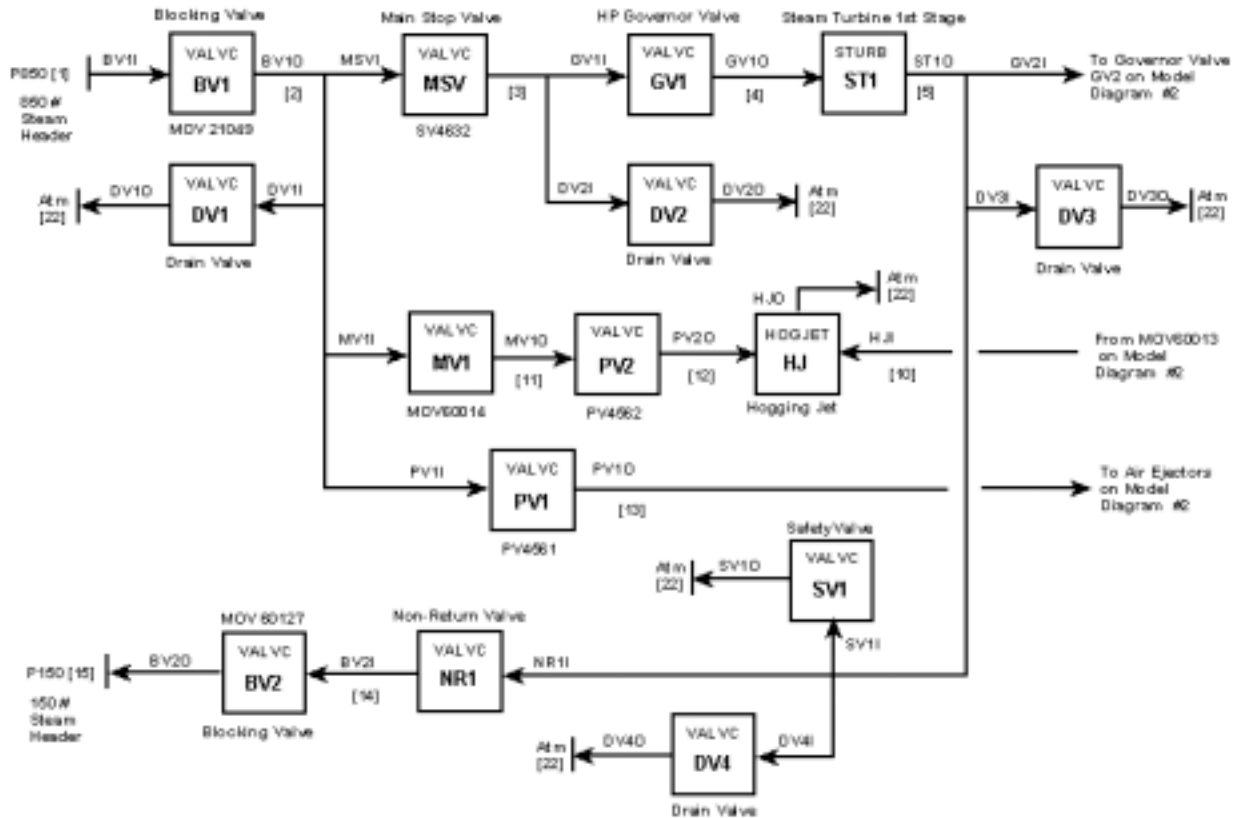


FIGURE 2 – STEAM CIRCUIT (PARTIAL) FLOW DIAGRAM

CONTROL SYSTEM MODELING

The turbine-generator (T-G) control system is implemented on a Programmable Logic Controller (PLC) and the source code is in a proprietary format. It is constructed with Ladder Logic and Blocks that perform specific functions. To provide a realistic representation of the control system, it is necessary to replicate each of these Blocks as accurately as possible. It was possible to print the diagram and use the printed copy to create an equivalent model.

Development of this model involved analyzing the control functionality and developing code blocks that represent the desired operation. Since this is a small turbine, the approach decided on was to develop subroutines for each of the required Blocks. The blocks were then interconnected and documented as shown in Figure 3.

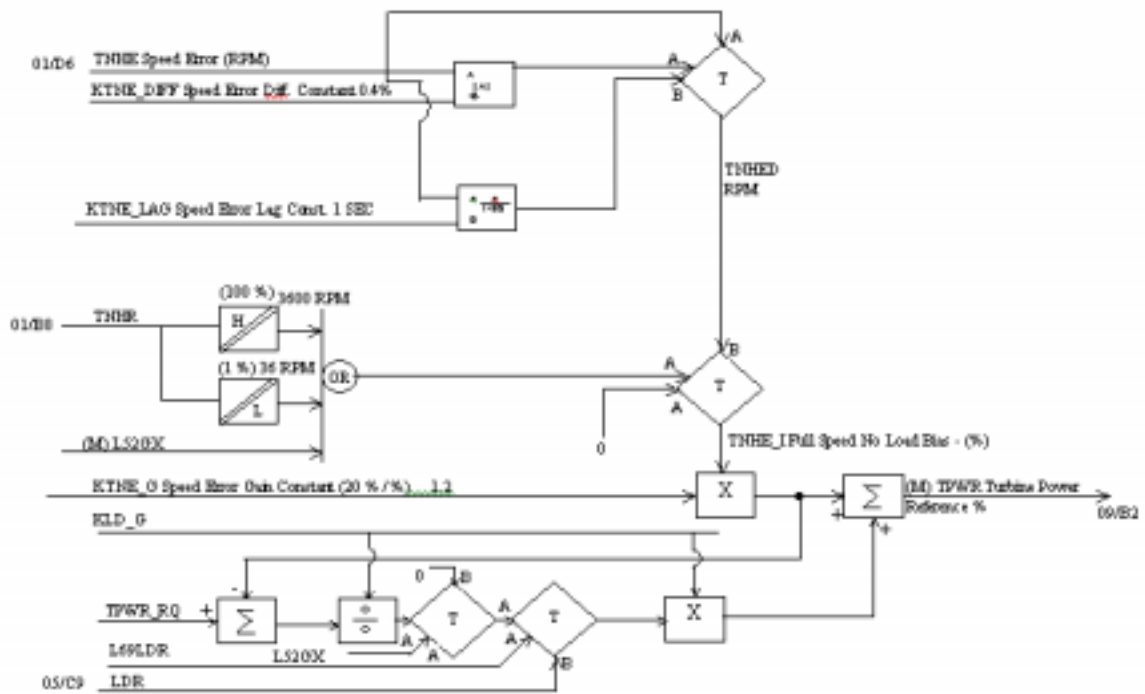


FIGURE 3 – EXAMPLE OF T-G CONTROL DIAGRAM

For the controls that are outside the scope of the T-G PLC, SAMA-based control diagrams existed; it was decided to take a graphic modeling approach to this part of the development task. The ACSL Graphic Modeller module software tool that was used allowed the use of pre-defined graphic icons, which can be dragged and dropped and then inter-connected to construct a model. To utilize this tool, an equivalent graphic module for each control block had been developed for the previous project. A typical example of part of a graphic model screen is shown in Figure 4.

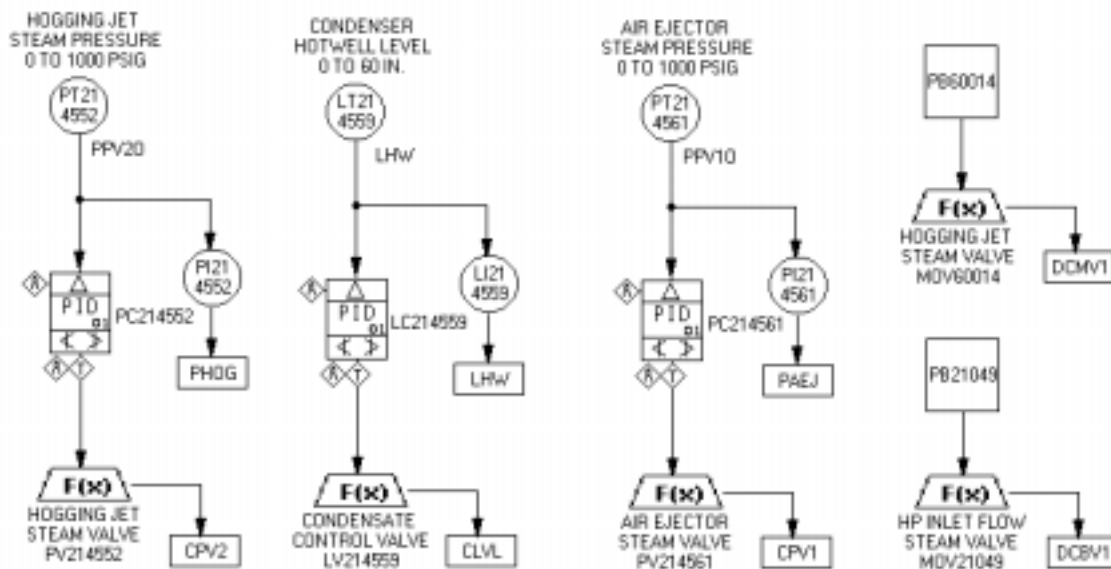


FIGURE 4 – EXAMPLE OF DCS DIAGRAM

Another advantage to this graphic model approach is that the control blocks can be labeled with the same nomenclature as on the original SAMA diagram. The graphic model also allows the viewing of the outputs of each control block while the simulation is running. This is a feature that provides significant user feedback when tuning and debugging.

OPERATOR SCREENS

Probably the single most important feature for a training simulator is to maintain the look and feel of the actual operating environment. Several graphics packages were considered, but since the package needed to be programmable as well as provide graphics capability, and Visual Basic (VB) had been used on another simulator project successfully, it was decided to utilize it. It should be noted that C or C++ could have been utilized as well, but the ability to make changes quickly and see immediate results with VB was the deciding factor.

In order to develop the screen images, photographs of the actual screens were taken, then cropped and enlarged, scanned, and edited. All static parts of the screens became a bitmapped image that was placed on the VB form in the background. All dynamic parts of the screen were added in the foreground, and consisted of bitmapped images or label or text boxes for the most part. This was a fairly labor intensive part of the development, but the resulting screens are virtually indistinguishable from the real ones. An example is shown in Figure 5.

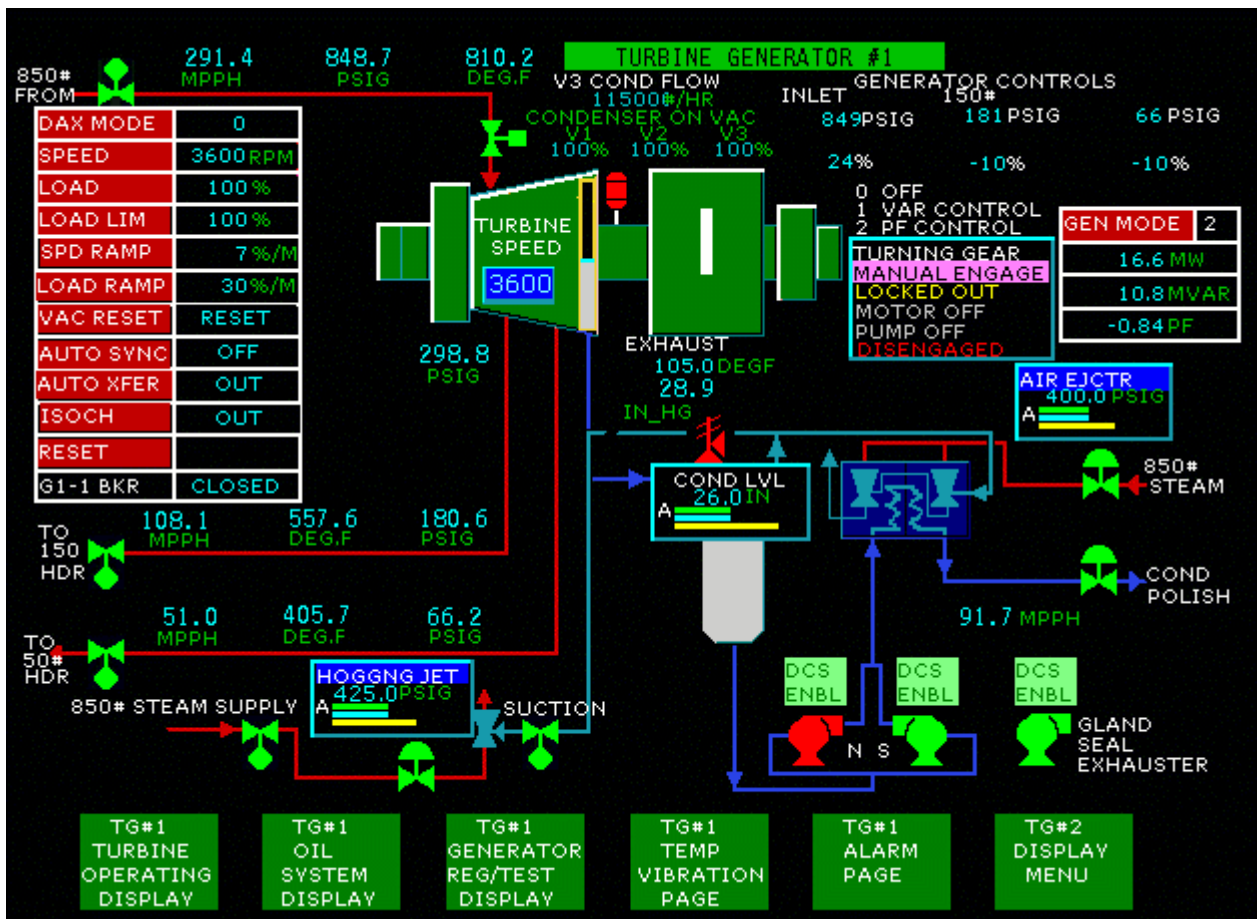


FIGURE 5 - EXAMPLE OF OPERATOR SCREEN

HARD PANEL EMULATIONS

There are some controls (that are part of the T-G process) that are not implemented within the DCS. The most notable of these is the synchronization controls. This requires the operators to go out to the turbine deck to synchronize the Generator to the mill grid. Therefore, readily available graphics software was used to emulate the hard panel systems. An example screen is shown in Figure 6.

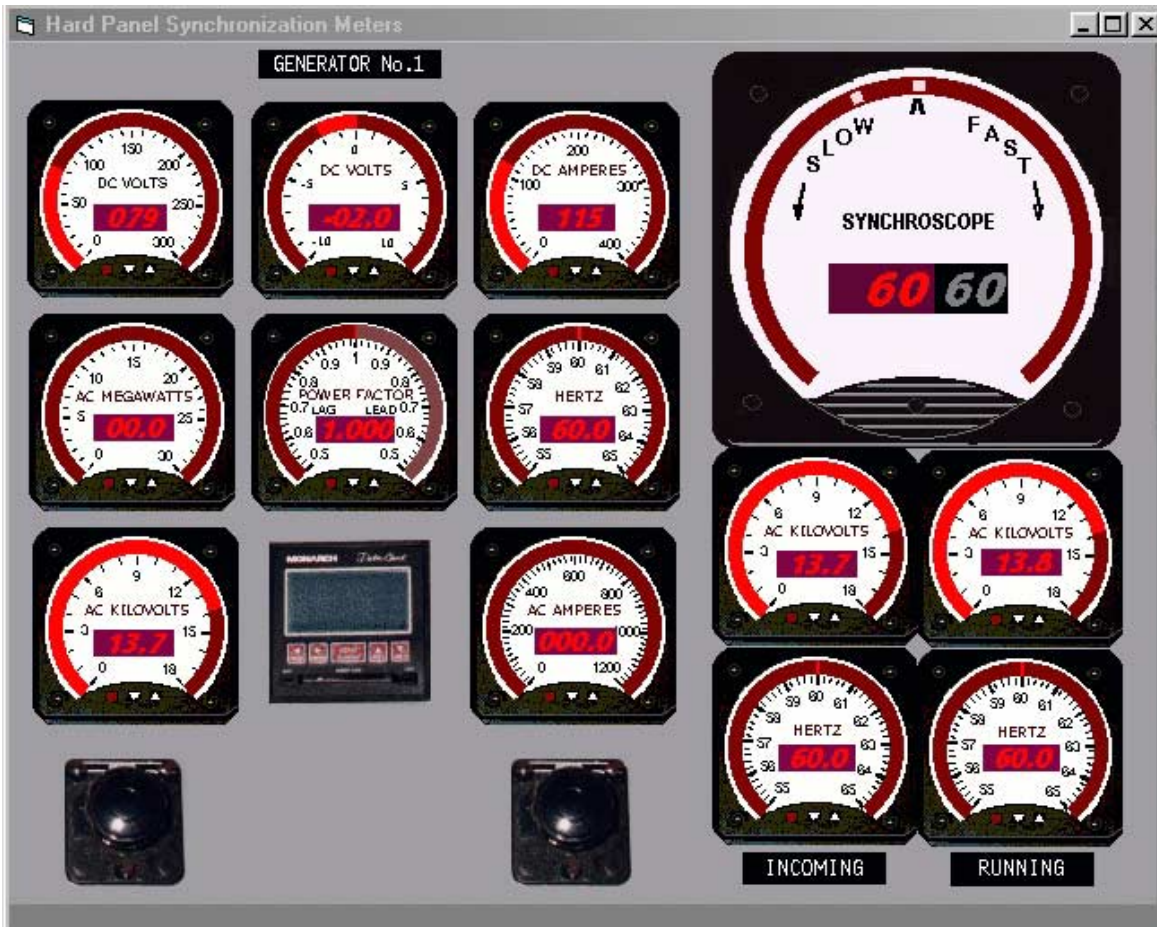


FIGURE 6 - EXAMPLE OF HARD PANEL SCREEN

INTERFACE SOFTWARE

Utilizing the API of the ACSL simulation model was the methodology used for interfacing all the pieces. This software exposes all of the objects in the simulation so values can be read from or written to variable locations in the simulation. It also allows program control for starting and stopping, plotting results, etc. Essentially, it converts a simulation into a server application by using OLE2. The computers are connected to a TCP/IP Local Area Network (LAN), which uses the standard Windows 95 TCP/IP protocol.

Referring back to Figure 1, The Instructor Console is connected to the LAN, and is used to control the simulation, which is running on Operator Console 1. Data transfer to and from the Instructor Console is via TCP/IP. Operator Console 1 also has the Operator Screens. Data transfer to and from the simulation is handled by internal OLE2 in this case. An identical set of screens is loaded on Operator Console 2, without a simulation, and it receives and sends data over the LAN via use of TCP/IP. The beauty of this method of data transfer is that multiple Operator Screens can be connected to the LAN, and operate independently of each other. Speed of response is limited only by the LAN speed and the API interface routines.

The Instructor Console (see Figure 7) was developed to allow the trainer to start the simulation, stop it (Freeze), save conditions for use as initial conditions (ICs) for future runs, save conditions for review and evaluation, restore saved conditions, insert malfunctions to see how the trainee responds, change the speed between real time and fast time for situations where things are moving slowly (such as a cool-down), etc. It also allows for use in debugging by allowing the trainer to interrogate the simulation for values, or to set a value in the simulation.

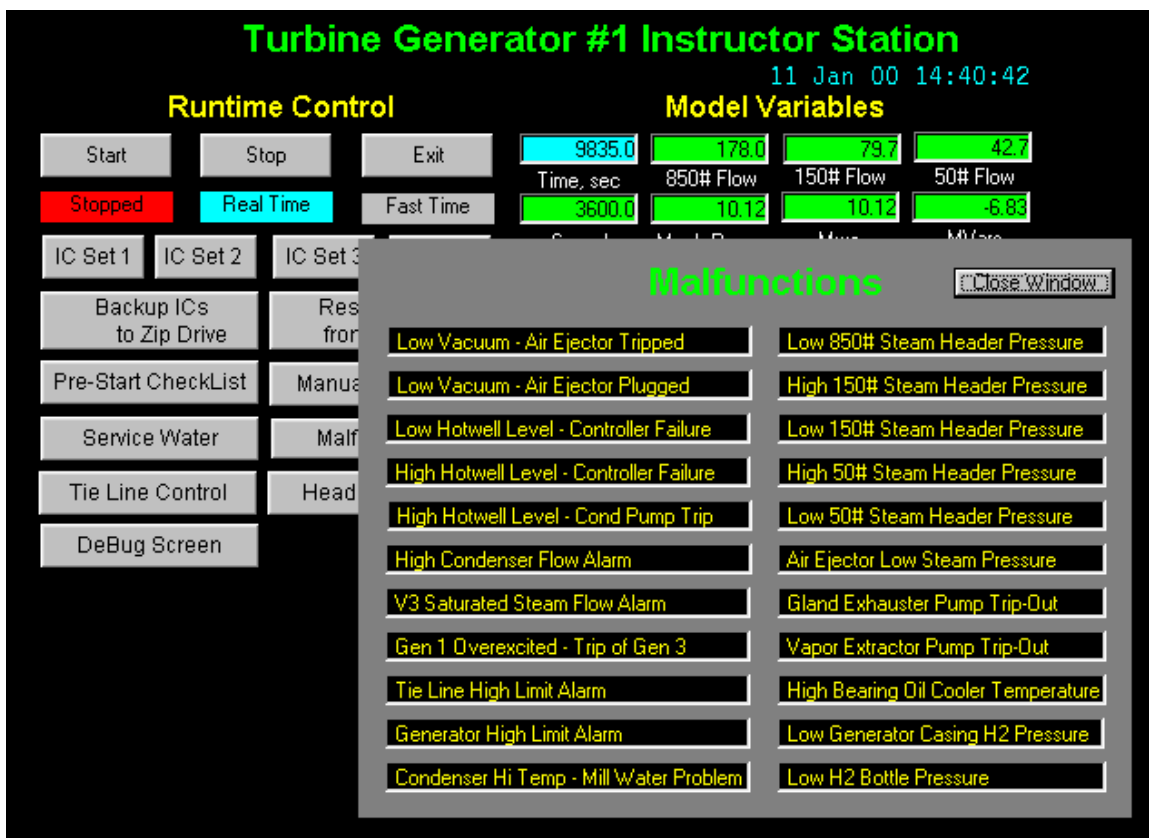


FIGURE 7 – INSTRUCTOR STATION

SYSTEM CHECKOUT AND USE

An extensive checkout of the completed system was made at the mill over a period of several weeks. Changes were made as needed, and the simulator was accepted in the fall of 1998. Elapsed time for the project was approximately one year. The resulting simulation includes very detailed models of the turbine-generator and auxiliary systems. The simulator is used for new employee familiarization, to train on startup, shutdown, maneuvering, and system upsets. It is extremely representative of the “look and feel” of the real control system interface to the real turbine-generator, and includes alarm sounds and other audio cues that are utilized in the real system.

SUMMARY

This paper has described the use of the general-purpose simulation tool ACSL with the Graphic Modeller module, Graphical User Interface Software (Visual Basic), and the Application Programming Interface as the basis for construction of a unit-specific training simulator for an industrial class turbine-generator. Used together with a proprietary set of FORTRAN-based modeling routines, the result was an economical alternative to the use of commercially available simulator software. It satisfied the customer needs, and set the foundation for follow-on work with regard to other mill powerhouse systems.

The technologies employed to construct this simulator can also applied to non-simulator applications, such as are used for dynamic system analysis for many engineering disciplines. Copyright 2002 Instrument Society of America – all rights reserved.

REFERENCES

1. Bartells, Philip, and Gauthier, Joseph, “Development of a Combination Boiler Simulator using General Purpose Simulation Tools”, ISA01-P1063, paper presented at the 2001 ISA Technical Conference in Houston, Texas, on September 11, 2001.
2. Advanced Continuous Simulation Language Reference Manual, The Aegis Technologies Group, Inc., 6703 Odyssey Dr., Suite 200, Huntsville, AL 35806, Published in 1999.
3. Microsoft Visual Basic Version 5.0 Programmers Guide, Microsoft Corporation, One Microsoft Way, Redmond, WA 98052, Copyright 1991-1997.
4. Bartells, Philip S. and Weber, David S., Process Modules Library - Developers Manual, SIMTECH Services, Inc., 1237 Woodbrook Lane, Forest, Virginia 24551, August 1998, unpublished in the general literature.