Human Machine Interface (HMI) in Simulation of Nonlinear Dynamic Systems Using Matlab-Simulink and InTouch Interface

ALAM JAN, Javed1 & SULC, Bohumil2
1 Ing., PhD Student, Department of Instrumentation and Control Engineering, Faculty of Mechanical Engineering, Czech Technical University in Prague, Technická 4, 166 07 Prague 6, The Czech Republic. Phone: +420 2 2435 2531, Fax: +420 2 311 6414
jlaved@student.fsid.cvut.cz
2 Doc., Department of Instrumentation and Control Engineering, Faculty of Mechanical Engineering, Czech Technical University in Prague, Technická 4, 166 07 Prague 6, The Czech Republic. Phone: +420 2 2435 2531, Fax: +420 2 311 6414
jsulc@fsid.cvut.cz

Abstract: It is very difficult to train an operator and perform experiments on an actual plant. There is a need for a simulator equipped with an interface that corresponds to the real conditions in the operation room. A computer-based plant model for such a simulator achieves a reduction of development time, costs and risks in comparison with manufacturing and testing numerous hardware prototypes, which are not feasible in a laboratory environment or a research centers.

This paper presents a Human Machine Interface (HMI) in a PC based simulation of a nonlinear model of a coal-fired steam boiler. The HMI is an important instrument in creating training simulators. It has been developed for data exchange between a Matlab-Simulink model of a steam boiler and an InTouch visualization program. InTouch is a commonly used program in professional power plant control rooms. It is a software package used to create a PC based man-machine interface. The interface between Matlab and InTouch is based on the Dynamic Data Exchange (DDE) protocol. During a DDE conversation, the client and server applications exchange data. Besides this solution, which is suitable for industry, similar visualization has also been prepared by means of Graphical User Interface (GUI) tools available in the MATLAB package. The advantage of this solution is that only a Matlab package is used, and some special simulation functions for operator training, such as backtracking, changing (fast/slow) simulation speed, freeze, definition of simulation conditions (e.g., size and time of disturbance changes, possibility of continuous controller parameter change, etc.) can easily be included.

Besides this main information the introduction to this paper also mentions some other tasks that are being solved by Mr. Jan in his PhD studies.

Keywords: visualization, simulation, thermal power plant, HMI, nonlinear systems

1 INTRODUCTION

A traditional role of simulation consists in providing the most precise replies to questions about how a device (usually very expensive or/dangerous) will behave under various (usually exceptional) operation conditions, and how to ensure its safe operation as far as possible. Simulation of a physical plant/process on a PC requires a mathematical model of the relevant processes and subsequent programming in a language that can be implemented into a PC or in some of simulation packages like Matlab, Mathematica, Simnon etc. In a simulator, the most important consideration is not the highest level of numerical exactness, but the highest likeness, especially in trends. For this reason the physical plant/process is
represented by simplified models using, e.g., continuous time differential equations, empirically obtained constants, estimated dynamic blocks etc. Such models are usually referred to as engineering models.

Engineering modeling of thermal power plants suitable for tuning control loops was already presented [1-9] before the project a part of a PhD study program went on to become. Some of the models are reported in this paper. The first model developed were visualization tools:

- in Matlab so that this can perform some special simulation functions for operator’s training, such as backtracking, changing (fast/slow) simulation speed, freeze, definition of simulation conditions (e.g., size and time of disturbance changes, possibility of continuous change of controller parameters change, etc.) - a detailed description is given in Section 4
- in InTouch (industrial visualization program) so that they can exchange data between the model in Matlab/Simulink and the visualization program in InTouch through a standard Dynamic Data Exchange (DDE) protocol developed by Microsoft for Windows operating system based applications- a detailed description is given in Section 3.

The next task is to suggest new approaches for engineering modeling based both on deterministic nonlinear modeling of thermal plants and neuro-fuzzy concepts. Some basic mathematical formulation of nonlinear models is shown in Section 2.

2 Modeling of the system

Modeling and control of large-scale systems, which has received great attention in recent years, is mainly based on ordinary differential equations. In modeling a power plant simulator, it is also reasonable to use nonlinear ordinary differential equations instead of the previously used concept of the modified Hammerstein model. As shown in Figure 1, in this concept the linear dynamic part of the model represented by transfer functions is connected with the nonlinear steady state model. A completely nonlinear solution makes it easily possible to simulate situations that need to be trained such as manual changes in opening water supplying valves, and operator reactions to falls in temperature and pressure.

This newly developed nonlinear model includes all traditional exploitations of use simulation, e.g., for

- reduction of development time, costs and risks in comparison with manufacturing and testing numerous hardware prototypes
- automation and acceleration of repetitive design and analysis tasks as well as trade-off studies and evaluation of alternative designs, components or parameters
- prediction of product behavior, function and acceptance tests, especially in critical situations, avoiding costly and/or dangerous experiments.
- supplementing experiments on prototypes for final verification of the specifications as well as validation of model assumptions.

One of the main keys for generating efficient simulation tools is to formulate the mathematical model in a computer-oriented way, i.e., in a framework that can easily be transferred into efficient computer codes.

Mathematical models may assume many different forms. Depending on the particular system and the particular circumstances, one mathematical model may be better suited than other
models. Once a mathematical model of a system is obtained, various analytical and computer tools can be used for analysis and synthesis purposes.

3 Suggested Approach to a Nonlinear Model of a Thermal Power Plant

This new kind of power plant model is now being developed. As shown in Figure 2, condensed water is pumped from the condenser to the deaerator via a preheater. The deaerator is used to increase the boiling point of the water. The feed water is then pumped from the deaerator to the drum via an economizer, which heats up the water to boiling temperature. The water in the drum is circulated through the evaporator, and the two phases-water and steam then separate into the drum. Saturated steam is extracted from the drum and passes through two super-heaters to produce super-heated steam before it enters the steam turbine. To control the temperature of the steam a spray cooler is placed between the super heaters. In the turbine the steam flow expands due to low pressure, and it is then condensed to water again in the condenser and can be re-circulated to the deaerator.

The model is based on dividing the plant into different control volumes and defining the mass, energy and momentum balances for these volumes. Here are some examples of the notions and mathematical formulation that are used.

Figure 2 Functional scheme of the Power Plant
Control volume
A fixed space, which flow passes through is called the control volume, and its surfaces are called control surfaces.

Law of Conservation of Mass for a Control Volume
Mass can neither be created nor destroyed, i.e., rate of mass flow ($q_{in}$) into the control volume equals rate of mass flow out ($q_{out}$) plus rate at which mass ($m$) accumulates inside.

$$\frac{dm}{dt} = q_{in} - q_{out} \tag{1}$$

Law of Conservation of Energy
The first law of thermodynamics states that the increase in internal energy ($E$) of a system of fixed identity is equal to the work done ($W$) on the system plus the heat ($\Delta Q$) added to the system.

$$\frac{dE}{dt} = \Delta Q + (q_{in} h_{in} - q_{out} h_{out}) + (m_{in} \frac{v_{in}^2}{2} - m_{out} \frac{v_{out}^2}{2}) - \frac{dW}{dt} \tag{2}$$

where $q$ is mass flow rate, $h$ - enthalpy, $m$ - mass, $v$ – velocity and indexes $in$ and $out$ are used for inflow and outflow.

Law of Conservation of Momentum
Rate of change of momentum is equal to the net force on the system.

$$\frac{d(mv)}{dt} = q_{in} v_{in} - q_{out} v_{out} + \sum F$$

where $F$ is force.

The whole system is divided into the control volumes as shown in Fig. 2.

Boiler
The dynamics of the boiler has been expressed by three state variables: pressure ($p$), volume of water ($V_w$) and steam quality ($x_m$) and are explained as under.

The boiler consists of two control volumes:
- drum boiler (including evaporator)
- evaporator

Applying equations (1) and (2) to the boiler two equations are obtained

$$\frac{dm_b}{dt} = q_{fo} - q_s, \quad \frac{dE_b}{dt} = \Delta Q_b + q_{fo} h_{fo} - q_s h_s$$

where total mass and energy of steam and water in the boiler are defined by equations

$$m_b = \rho_s V_{sb} + \rho_w V_{wb}, \quad E_b = \rho_s h_{sb} V_{sb} + \rho_w h_{wb} V_{wb} + m_{wb} C_m T_s$$

The dynamics of the evaporator can be expressed by the equation

$$\frac{dm_{ev}}{dt} = q_{fe} - q_r, \quad \frac{dE_{ev}}{dt} = \Delta Q_b + q_{fe} h_{fe} - q_r [(1 - x_m) + x_m h_s]$$

where again masses and energy in the evaporator are defined by equations

$$m_{ev} = \rho_s V_{sv} + \rho_w V_{sw}, \quad E_{ev} = \rho_s h_s V_{sv} + \rho_w h_w V_{sw} + m_{sw} C_m T_s$$

Solving the above equations, the following form of equations can be derived:

$$\frac{dp}{dt} = f(\Phi) \quad \frac{dV_w}{dt} = f(\Phi) \quad \frac{dx_m}{dt} = f(\Phi)$$

where $\Phi$ is a function of $m$, $\rho(p)$, $h(p)$, $T(p)$, $x_v(p, x_m)$ etc and $m$ stands for mass, $\rho$ for density, $V$ for volume, $C$ for specific heat, $h$ for enthalpy, $E$ for internal energy, $T$ for temperature, $Q$ for heat, $x_v$ for quality of steam (volume fraction), $x_m$ for quality of steam (mass fraction)
4 Visualization program in InTouch and its interface with Matlab:

We can logically break the work into three parts:
- Simulation of the plant / process using Matlab/Simulink.
- Design of the control panel as it works in real plants.
- Real time link between simulated plant and control panel.

As discussed previously, simulation of the power plant was performed in Simulink, visualization was designed in InTouch, as shown in Figure 5 and discussed in section 3.4.

Communication for data exchange between the model and visualization program was performed based on the DDE protocol.

3.1 Dynamic Data Exchange (DDE)

DDE is one of several mechanisms of inter process communication supported by Windows. It is an extension of the messaging scheme around which Windows is designed. Of the two applications involved in the data exchange, one is known as the server and the other as the client. The client application is the consumer of the data and the server is the provider.

DDE Concepts and Terminology

Applications communicate with each other by establishing a DDE conversation. The application that initiates the conversation is called the client. The application that responds to the client application is called the server.

When a client application initiates a DDE conversation, it must identify two DDE parameters that are defined by the server:
- The name of the application that it intends to have the conversation with, called the service name.
- The subject of the conversation, called the topic name.

When a server application receives a request for a conversation involving a supported topic, it acknowledges the request, establishing a DDE conversation. The combination of a service and a topic identifies a conversation uniquely. The service or topic cannot be changed for the duration of the conversation, although the service can maintain more than one conversation. During a DDE conversation, the client and server applications exchange data between concerning items. An item is a reference to data that is meaningful to both applications in conversation. Either application can change the item during a conversation.

Service Name
Every application that can be a DDE server has a unique service name. The service name is usually the applications executable file name without any extension. Here are some commonly used service names:
- The service name for Matlab is Matlab,
- Microsoft excel is excel,
- Wonder ware's InTouch is view.

Topic
The topic defines the subject of a DDE conversation and is usually meaningful to both the client and server applications.
- Matlab's topic is system.
- Wonder ware's InTouch topic name is tagname.
- Excel topic name is ‘file name’ opened.
Item
An item identifies the data being passed during a DDE conversation. Matlab items are the
names of the variables. InTouch items are the actual tag names. In Excel the row and column
name is e.g., ‘r2c5’. For Mininet (PLC controller) the items are the addresses of the registers
with their appropriate data types.
To obtain data from another application, the client program opens a channel to the server
application by specifying these three items.

3.2 Matlab
Matlab provides functions that enable Matlab to access other Windows applications and for
other Windows applications to access Matlab in a wide range of contexts. These functions use
dynamic data exchange (DDE)/ software, which allows Microsoft Windows applications to
communicate with each other by exchanging data.
Matlab supports dynamic data exchange. A few functions are as follows

DDEINIT
This function is used to establish the channel between client and server applications. The
syntax is:
channel = ddeinit('application name', 'topic name')
If the value of the channel returned is non-zero, this means that the channel is successfully
initiated, while zero means a failure, for example to initiate a channel with Microsoft Excel:
channel = ddeinit('excel', 'send.xls')
where
channel: is the name of the link established between Matlab and excel.
send.xls: is the name of spread sheet already opened in excel.

DDEREQ
If Matlab requires data from an application (through a DDE channel established before), this
ddereq (DDE request) function is used:
x  = ddereq ( channel , ' item name' )
The value requested will be stored in ‘x ’ or any other variable in Matlab. For example to
request a value from Excel:
x  =  ddereq ( channel , ' r1c1 ' )
where x is a name of the variable in which the value is to be stored and r1c1 is the cell in
Excel from which the value is requested.

DDEPOKE
Data is sent from Matlab to the other application through the ddepoke function. The syntax is
ddepoke ( channel , ' item name ', variable name )
Data in the variable name is transferred to the item name of the other application. For example
sending a value to Excel from Matlab is as follows

Figure 3 Dynamic Data Exchange (DDE) Conversation
ddepoke ( channel , 'r1c1' , x )

where x is the variable whose value is to be poked and r1c1 is the variable in which the value is to be stored.

**DDETERM**

The DDE channel initiated earlier is terminated by the ddeterm function. The syntax is

ddeterm ( channel )

where

channel : is the link established earlier between the two application.

3.3 INTOUCH

InTouch is a software package used to create a PC based man-machine interface. The package consists of two major elements: *Window Maker* and *Window Viewer*. Window Maker is the development environment, where object-oriented graphics are used to create animated touch-sensitive display windows. These display windows can be connected to industrial I/O systems (e.g., PLC) and other Microsoft Windows applications (e.g., Matlab, Excel). Window Viewer is the run time environment used to display the graphic windows created in Window Maker.

InTouch can acquire a data value from another application. For this purpose it must know the name of the application providing the data value, the name of the topic within the application that contains the data value, and the name of the specific item within the topic. In addition, InTouch needs to know the data type (e.g., discrete, integer, real, message). This information determines the DDE type for the tagname when it is defined in the In Touch database. When Window Viewer is running, it will automatically perform all of the actions required to acquire and maintain the value of this item.
As in Matlab, variables are used to store data. InTouch stores data in different types of tags, e.g., discrete, integer, real, message. The tags can be internal as memory and external as I/O. InTouch can receive data from other Windows applications by creating DDE type tag names in its tagname data dictionary. The steps involved in defining a DDE type tagname are as follows: (see Figure 4)
1. To define a new tagname, invoke the Special / tagname dictionary command.
2. Click on the tagname dictionary (command).
3. The dictionary tagname definition dialogue box will appear. Click on the New button.
4. The tagname to be defined for the DDE item in the database is now entered in the tagname field. Click on the Type button to select the type for the tagname.
5. The tag types dialogue box will appear. Select the appropriate DDE type for the tagname and press OK.
6. The tagname dictionary definition dialogue box will reappear along with the 'details' dialogue box for the DDE type selected. Once all details have been entered, click on the DDE Access Name (button).
7. Each DDE type tagname must be associated with a DDE access name that defines the application name and topic name to which the DDE tagname will read / write its value. Any access name can be given; it provides the user the information to which the application the tag is connected. For example, with the Matlab interface the access name Matlab can be used, similarly with the PLC controller PLC can be given. To define a new DDE access name, click on the Add (button).
8. The Add DDE Access Name dialogue box will appear. This dialogue box is used to enter the application name and topic name to be associated with the specified DDE access name. Once all entries have been made, click OK to accept the new DDE access name and close the dialogue box.
9. The DDE access name definition dialogue box will reappear with the new DDE Access name pre selected in this list. Click close to close the dialogue and return to the Dictionary Tagname Definitions and 'Details' dialogue boxes.
10. The dictionary - tagname definitions and "Details" dialogue boxes will appear. The DDE access name will be displayed adjacent to the DDE Access name (button).
11. The final step in defining the DDE tagname is entering the item name for the data point in the remote application database in the Item: field.

Figure 5 InTouch Control Panel of the Plant
This completes the steps required to define a DDE tagname in InTouch. When another Windows application wants to request a data value from InTouch, it also needs to know the three DDE addresses for InTouch:

1. View (application name) refers to the run time of the InTouch program that contains the data elements.
2. Tagname (topic name) is the word always used when reading / writing to a tagname in the InTouch database.
3. Actual tagname (item name) is the actual tagname defined for the item in the InTouch Tagname Data Dictionary.

For example to access a data value in InTouch from Excel, a DDE remote reference formula would be entered in the cell into which the data value is to be written:

=View / Tagname | Actual Tagname

In the window shown in Figure 4 SteamPressure is the tagname for steam pressure. Some results of simulation can be seen in Figure 6 and Figure 7.

![Figure 6 InTouch Control Panel of the Plant showing plots](image)

3.4 Graphical User Interface (GUI) in Matlab:

This panel was designed using the MATLAB-based Graphical User Interface (GUI) tools. It falls naturally into two parts:

- GUI design, or how to make something that is useful
- GUI implementation, or how to make something that works

The principles of good GUI design are, for the most part, timeless and universal. They apply in MATLAB as much as they apply anywhere else.

The model is implemented in Simulink and interfaced with this panel, as shown in Fig.8. Different values, e.g., simulation time, backtracking time, disturbances and set points, etc., can be changed from the panel. Controller parameters can also be changed by clicking Controller1 and Controller2 buttons. Similarly values of simulation speed and nominal loads can be varied. By pressing the start button simulation is started, by clicking the pause and stop buttons it can be held and stopped. By pressing the B.T. button backtracking can be initiated. Options are also provided to display different plots separately and in combination.
5 CONCLUSIONS

Simulation using Human Machine Interface (HMI) provides better visualization and understanding of a process. Using DDE, interface data generated by various applications can be shared. Also a practical controller, e.g., PLC can be tested on a model using a DDE interface. This means that before connecting a controller to a process its performance can be checked by a model using this technique.

6 References