CONCEPT REPRESENTATION OF THE SOFTWARE TOOL PIDMASTER FOR SYSTEMS MODELING AND CONTROLLERS TUNING

M. Ostroveršnik†, Z. Šehić‡, B. Zupančič‡, M. Šega*

†"Jozef Stefan"Institute,
Jamova 39, 61 000 Ljubljana, Slovenia.
‡Faculty of Electrical and Computer Engineering,
61 000 Ljubljana, Tržaška 25, Slovenia.
*Metronik d.o.o.,
Stegne 21, 61 000 Ljubljana, Slovenia.

Keywords: OOP, Software Engineering, Control System Design, PID Control.

Edited by: Miroslav Kubat
Received: October 22, 1993 **Revised:** January 11, 1994 **Accepted:** February 15, 1994

The work deals with the representation of the concepts of program package PIDMaster for on-site signal analysis, system modeling and controller's synthesis. A user-friendly interface is introduced which enables comfortable work also for less experienced users. The required user-friendliness has been achieved by the use of window interface completed with some elements of the object oriented user interface management system. Central points of the new interface are the so-called movable graphical objects. Their usage in the phase of system parameter estimation is explained thoroughly.

1 Introduction

Despite tremendous development of modern control theory, PID controllers are still widely used in process control applications. Moreover, there are no indications that the situation will change in the near future. The reason why PID is so popular in industrial environment is its ability to master the process dominant dynamics with (in principle) only three parameters, which are relatively easy to understand by process operators [1].

Roughly speaking, there are three categories of PID controllers available on the market [1]:

- PID controllers realized in classical analog technique,
- digital PID controllers,
- knowledge-based PID controllers with the abilities for autotuning and adaptation [1].

Most of the control equipment in industrial plants belongs to the first two categories. Tuning of such a controller is entirely in the domain of process operators and maintainers who approach tuning pragmatically using heuristics and intuition. Unfortunately, from the general point of view, PID controller tuning has remained an art rather than exact science [2]. Therefore, people in industry involved in maintenance need a tool that will help them tune the controller without requiring too detailed knowledge about the control theory. Such a tool is also desirable by control people involved in industrial applications and projects. Therefore, we decided to design and realize a tool that will provide the following:

- "manual" PID tuning on the process model,
- optimization based PID tuning for difficult and nonstandard problems,
- PID tuning by means of tuning rules,
- graphical modeling,
- intuitive and user friendly interface with extensive on-line context-sensitive and hypertext based help,
- ability for easy upgrading without reconstruction of the whole program,

- possibility to work in industrial area (portable, robust, relatively inexpensive).

Two years ago we started the project called PIDMaster. The final result of this project will be a tool which will fulfil previously mentioned goals. Such a tool should provide tuned controller parameters, process models and various progress and status reports, all on the basis of input data files obtained directly by data acquisition equipment, i.e. by measurements of the real process. Input to such a tool will be various data files obtained by data acquisition equipment through direct sensoring of vital variables of a target process. The output will be tuned controller parameters, models of process and various progress and status reports made by user of PIDMaster. In summary, the PIDMaster project has been started with the idea of supporting the above mentioned tasks, preventing huge human errors, offering the possibility of prototyping and thus implicitly learning by example.

2 Project Background

We decided to develop our tool on a PC platform because PC compatible hardware is really widespread and relatively inexpensive. Besides we already had experience with similar projects. Let us mention only two of them, the so-called ANA [3] and SIMCOS [4]. ANA has been made for system analysis and design of controller synthesis [3] whilst SIMCOS [4] has served for simulation of dynamic systems. Both tools were developed for experienced users and are fairly sophisticated. Major drawbacks of these two powerful tools, especially ANA, are relatively poor user interface and lack of project documentation, which caused difficulties with maintenance and upgrading. With the new project we wanted to dodge these problems.

The PIDMaster project differs from our previous projects in the following:

- we established a mixed development team,
- the project lifecycle was strictly divided into phases, and
- IBM's Common User Access (CUA, [5]) standard was used for design and implementation of the user interface.

We established a mixed development team from target domain and computer science experts which can cover both areas without segregation of one area. Actually, in the earlier project phases, target domain experts played dominant role and now, when we are in the implementation phase of the project, computer science component is more stressed. As mentioned above, to overcome the problem with a user interface we decided to use IBM's Common User Access standard, which is roughly supported with Microsoft windowing system Windows 3.x. Since the project is in the implementation phase we haven't had maintenance problems, but we hope that we can surmount such difficulties with thorough project documentation. The documentation includes several documents:

- system requirements document,
- system analysis document,
- design of user and tool interface document,
- design of important objects document, and
- database structure document.

3 PIDMaster Overview

The PIDMaster communicates with environment in three ways (fig. 1, i.e. Signals, Reports, User). Fundamental data inputs are process signals collected with other tools (i.e. Labtech, G-2). The main outputs are various reports that include figures of signals, descriptions of models and controller setups. We cannot eliminate user cooperation, so that all interactions are supervised by user. PIDMaster can import and export signals in various file formats like Matlab binary file format, Labtech ASCII format, the flat ASCII format etc. Reports can be written on any Win 3.x supported printer.

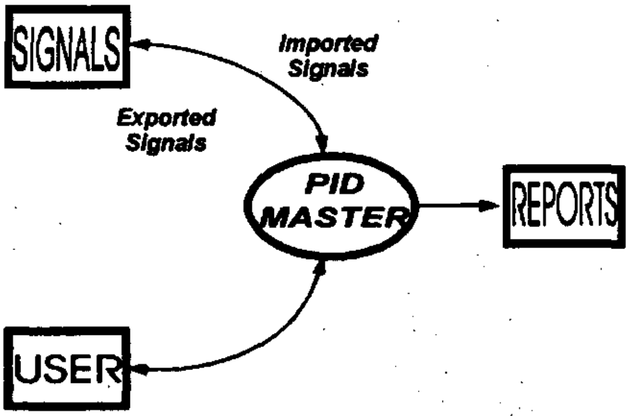

Figure 1: PIDMaster environment

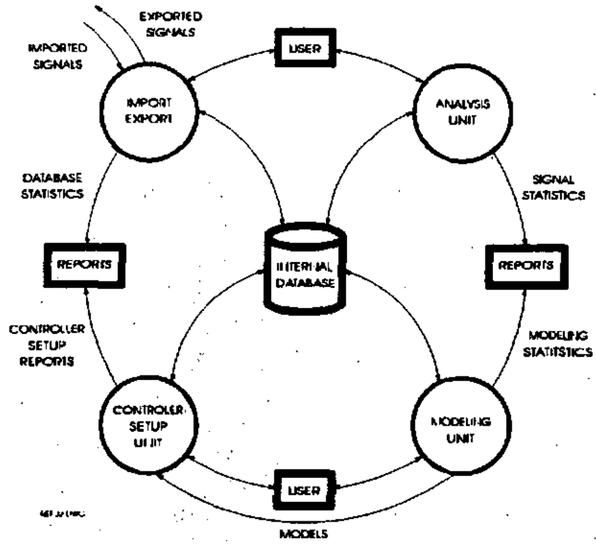

Figure 2: PIDMaster decomposition

From the user point of view PIDMaster incorporates four major units (fig. 2):

- analysis,
- modeling,
- controller tuning, and
- data import / export unit.

All units are linked via an internal database that includes all imported and/or generated signals, system models and controller setup parameters. All unit input and output formats are compatible, so that results obtained in a previously used unit can be used in the next unit. The most natural way of work is to start with the import of signals via analysis and modeling unit and then to proceed to the controller setup unit. Of course, the user can enter the process of controller tuning at any mentioned point (see fig. 2).

The PIDMaster has certain characteristics that are common for tools of its sort and we will mention them only briefly. The main point of distinction regards system modeling from measured data. Since it is almost a way of communication with the user we shall explain it later in user interface section (see section 4).

3.1 Analysis unit

The analysis unit enables the user to display and edit signals on a screen in various ways (y=f(t), or z=f(x(t),y(t))). The tool can calculate standard integral criteria (e.g., Sx, IAE, ISE, ITAE, ITSE), compare two signals and calculate standard deviation, correlation, etc.

3.2 Modeling unit

In the modeling unit eight different mathematical models can be used. The common working pattern consists of selecting the model and entering the values of the parameters. These values can be later improved with a computer supported optimization or computer supported verification. The following models can be used in the process of modeling :

$$M1 : Y(s) = \frac{Ke^{-sT_d}}{1+sT_1}U(s)$$

$$M2 : Y(s) = \frac{Ke^{-sT_d}}{(1+sT_1)^n}U(s)$$

$$
\begin{aligned}
M3 : Y(s) &= \frac{e^{-sT_d}}{sT_1(1+sT_1)^n}U(s) \\
M4 : Y(s) &= \frac{Ke^{-sT_d}}{(1+sT_1)(1+sT_2)}U(s) \quad (1) \\
M5 : Y(s) &= \frac{Ke^2e^{-sT_d}}{s^2+2\xi\omega s+\omega^2}U(s) \\
M6 : Y(s) &= \frac{K(1+sT_0)e^{-sT_d}}{(1+sT_1)(1+sT_2)}U(s) \\
M7 : Y(s) &= \frac{K\omega^2(1+sT_0)e^{-sT_d}}{s^2+2\xi\omega s+\omega^2}U(s) \\
M8 : Y(s) &= \frac{KP_2(s)e^{-sT_d}}{Q_3(s)}U(s)
\end{aligned}
$$

where s, $Y(s)$, $U(s)$, K, T_d, T_i, ξ, ω, $P_2(s)/Q_3$ represent complex variable, Laplace transform of system output, Laplace transform of system input, process gain, pure time delay, time constant, damping factor, natural frequency and general transfer function, respectively parameters for equations (1) can be entered manually or via the so-called engineering method, which encompasses graphical and numerical estimation of parameter values. We believe that the support of the so-called engineering method for the parameter estimation is an important novelty and the main point of distinction between our tool and other available tools of this type. This procedure will hence be briefly described in the next section.

3.3 Controller tuning unit

The user can enter a controller tuning unit directly or via the modeling unit. Here he can select the appropriate controller and tune it's parameters. User can setup parameters directly or with the help of tuning rules (e.g., Ziegler-Nichols, Oppelt, Chien-Crones-Reswick, Rutherford-Aikman, Kessler [6, 7, 8, 9, 10]). It is possible to select both continuous and discrete controllers, different structures (e.g. P, PI, PD, PID) including modified structures (e.g. classical, noninteracting, industrial [10]). Selected parameter values can be optimized and/or verified with computer supported optimization and/or verification. The criterion function can be one of the output error based functions.

4 Engineering Method for System Parameter Estimation (EMSysPE)

Target user group of the PIDMaster are callow computer and control engineers so that we had to pay more attention to design of a friendly and especially forgiving user interface, which should attract new users. We decided to use Microsoft's windowing system and to follow the CUA standard if possible. The PIDMaster interface incorporates two distinct approaches, i.e.:

- diagrams (windows), and
- dialogues.

We use graphical windows for displaying values of signals in the form of diagrams and dialogues for numerical input and output. This is traditional approach alo used in other tools of this type. The main difference of our tool are the so-called movable graphical objects which combine both approaches in a new and more effective way. Movable graphical objects are part of module for estimation of system parameter values.

If we want to model an unknown system in the traditional way, we have to excite it with a step signal on input and observe its response on output. A typical response of a low-pass system is a sort of sloped step. The shape of the sloped step determines the type of the model. Both objects figure out model constants (delay, gain, time constant). That is almost a general algorithm for selecting the type of model and finding its constants. That sounds simple, but there are two problems. First, the algorithm is too general for computer implementation and can not be easily automated. Second, it includes relatively complex calculations that have to be executed several times. We decided to give the user additional support in areas where computers are generally more efficient and reliable (i.e., numerical calculations, memory capacity and reliability, ability of errorless repetition), and we should leave all freedom where man's intuition is essential.

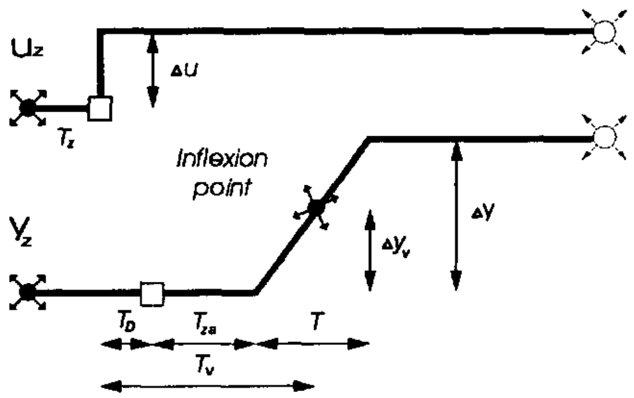

Figure 3: MGO's for model 1

Our concept is based on the so-called moveable graphical objects (MGO). MGO is a graphical object (see fig. 3) composed of markers and lines connecting them. The user can change dimensions or position of MGO simply by manipulating MGOs' markers. The visual appearance of a marker rules its function. The user can handle markers and thus MGO primarily with mouse click and drag. As a backup option we incorporate digital entry too. The system carries out two major MGO groups: simple and complex. The group of simple MGOs incorporates cursor, vertical and horizontal line, slope and parabola. The group of complex MGOs includes four different steps (step (see fig. 3, top), slopped step(see fig. 3, bottom), endless slopped step, slopped step with overshoot). The set of simple MGOs is primarily used in the phase of signal analysis. The set of complex MGOs is used in the system parameter estimation unit. The power of exploiting complex MGOs lies in the so-called engineering method for estimation of system parameters as can be shown in the following example.

Let's try to estimate system parameters with EMSysPE. The model is defined with equation M1 (see (eq. 1)) and we have to find gain, time delay and time constant (i.e., K, T_d and T_1). After selecting the proper model the program enables two MGOs. The first one is always a step and the second is one of the sloped steps. The type of the sloped step depends on the selection of the model. In our example the PIDMaster would give us the picture in fig. 3. As we can see, we have two MGOs, i.e. step and slopped step. The user's task is to properly relate the step on input and the ramp on output measured signal. Hence the user is responsible for correct positioning and dimensions of available objects. The position and dimensions of MGO can be set with the mouse. When the user is sure that both objects are positioned and scaled correctly he can start the process of calculation of model constants. It is the computer that is responsible for that. In our example the computer calculates the model constants from marker positions (e.g., $K = \Delta y/\Delta u, T_d = T_D + T_{za}, T_1 = T$). As can be seen, constants can be calculated directly from positions of markers. Estimated parameters can be saved in a database for later use in this or controller setup unit.

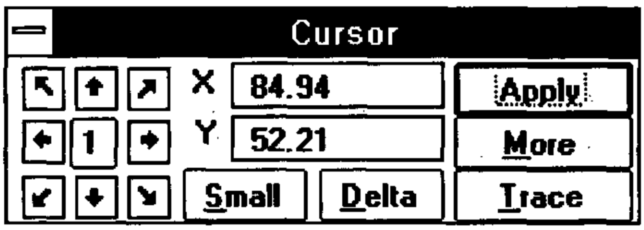

Figure 4: MGO's control window

Parameters can be verified and/or optimized. All intermediate steps can be saved in a log file. A snapshot of actual modeling with the PIDMaster is depicted in fig. 6, where the most important windows and controls can be seen.

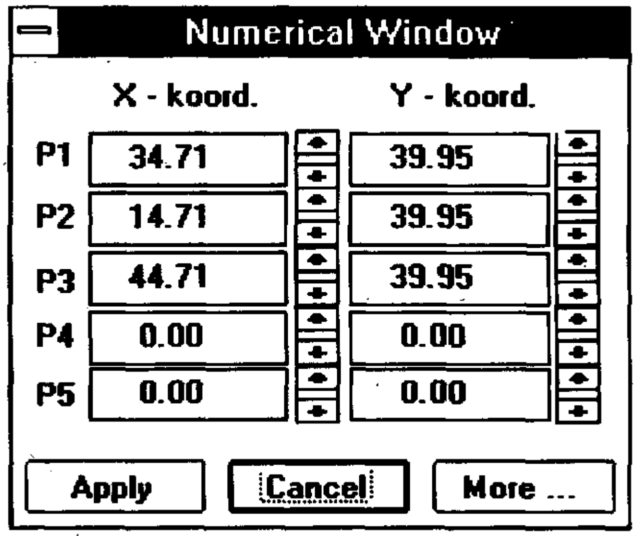

Figure 5: MGO's numerical window

The example shown is quite simple because there is no transitional step between graphical

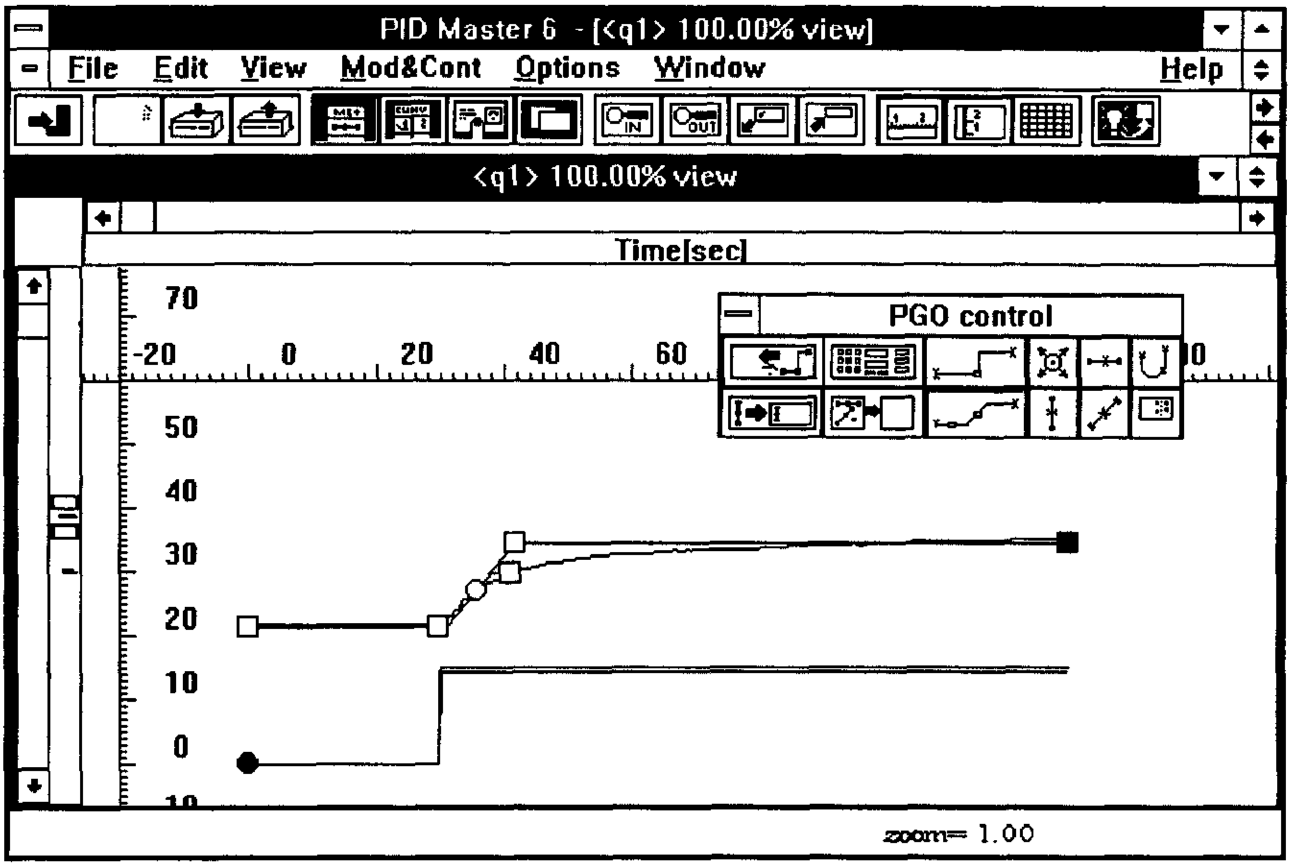

Figure 6: Graphical modeling of model 1

phase and calculation. In case of other models there is always an intermediate step in which one has to decide which possibility to choose. Anyhow the procedure is not so demanding since the user can select a certain value, check the effect and, if not satisfied, select another one, etc.

We believe that graphical estimation of parameters is sufficient for an average user but, nevertheless, we added also an option for numerical positioning of markers. The numerical input method is useful in case of low resolution graphic card or mouse, or in case of repetitive operations. The numerical input is accomplished via two additional modeless dialogues (see fig. 4 and fig. 5). The dialogue in fig. 4 (MGO Control) starts when the user selects an MGO. Values X and Y represent absolute coordinates of the active marker on MGO. New values can be entered directly in edit fields X and Y or via eight buttons. A click on button increases/decreases the selected value in a specified direction. For example, button ($\Rightarrow$) causes the value of field X to increase for delta, the opposite button ($\Leftarrow$) causes the value of field X to decrease for the same delta. The delta value is a user-defined value, with ability to be changed during program execution. Dialogue in fig. 4 enables user to read or change position of the active marker. On the other hand, dialogue in fig. 5 (the numerical window) gives the opportunity to read or change location of all markers in the same time. This dialogue can be activated with a click on button (More) on MGO's control window (fig. 5).

5 Status of Project and Further Improvements

The project is in the implementation phase and the beta version is in testing phase. We have succeeded to completely implement EMSysPE method, while controller setup and tuning is still pending. The significant delay was caused by severe problems with compilers and graphical environment. In case of good acceptance of PIDMaster, we will upgrade our tool from the target domain experts side. The support for real-time signal acquisition can be added, as well as more support for various import/export file formats and support for OLE (object linking and embedding). A module for direct loading of parameters in a particular commercially available controller could also be developed.

References

[1] Åström, K.J., Hang C.C., Persson P., Ho, W.K.: Towards Intelligent PID Control, Automatica, vol 28, p.p. 1-9, 1992.

[2] Gilbert Polonyi, M.J.: PID Controller Tuning Using Standard Forma Optimisation, Control Engineering, p.p. 102-106, March 1989.

[3] Šega, M., Strmčnik, S., Karba, R., Matko, D.: Interactive Program Package ANA for System Analysis and Control Design, Preprints, 3^{rd} IFAC/IFTP Int. Symposium CADCE '85, p.p. Copenhagen, August 1985.

[4] Zupančič, B.: SIMCOS - jezik za simulacijo zveznih in diskretnih sistemov, Fakulteta za elektrotehniko in računalništvo, Univerza v Ljubljani, Ljubljana, 1992.

[5] IBM Systems Application Architecture: Common User Access Advanced Interface Design Guide.

[6] Ziegler, J.G., Nichols, N.B.: Optimum Settings for Automatic Controllers, Trans. ASME, vol. 64, p.p. 759-768, 1942.

[7] Opelt, W.: Kleines Handbuch Technicher Regervorgänge, Verlag Chemie, GmbH, Frankfurt am Main, 1964.

[8] Hofmeister, W.: Prozessregler - Auswahlkriterien und Einstellungshilfen, VDI - Verlag, GmbH, Düseldorf, 1975.

[9] Isermann, R.: Digital Control Systems, Springler Verlag, Berlin-Heiderberg-New York, 1981.

[10] Kaya, A.: Tunning of PID Controls of Different Structures, Control Engineering, p.p. 28-31, July 88.