UDC621.783-523:681.518

Application of Object-Oriented Design to Box Annealing Furnace (BAF) Atmosphere Control System

Osamu Sekiguchi*1
Yuji Nakashima*3
Osamu Dairiki*2
Shin,ichiro Tahara*3

Abstract:

In recent years design technology based on object-oriented languages in computer software has found increasing usage in the construction of business systems and production planning systems, among other systems. It is far superior in maintainability and reusability to that based on conventional procedural languages. To put these advantages of object-oriented design to effective use in plant control systems, the control system building tool "GOOD (Global Object-Oriented Designer)" was developed, based on the object-oriented design technology, and applied to a box annealing furnace (BAF) atmosphere control system at the Yawata Works of Nippon Steel. About one year after its hot run, the BAF atmosphere control system is working smoothly.

1. Introduction

The direct digital control (DDC) of instrumentation appeared about 20 years ago. DDC is a control system of extremely high functional freedom that replaces conventional hardware circuit-based instrumentation control functions by computer-based software processing.

The functional design of conventional instrumentation control by the combination of controllers and other hardware was separated into "data" and "processing" under DDC. Despite the excellent functional freedom of DDC, this separation required the construction of instrumentation control systems with great care devoted to tasks not essential for conventional instrumentation design, such as data address and memory allocation. This tendency intensified further in systems where continuous control and sequential control coexisted. It also called for the concept of design in instrument engineering to be drastically changed.

Subsequently, instrumentation control systems advanced from minicomputers to microcomputers and from DDC to distributed control systems (DCSs), and software design tools were augmented. As long as "data" and "processing" were separated, however, these measures were not essential solutions. We thought the most of this fact when we designed next-generation DCSs. That is, we aimed at the realization of a DCS software building environment independent of computer data processing patterns and conforming to essential instrumentation control system design. To this end objected-oriented design was indispensable.

*1 Yawata Works
*2 Electronics & Information Systems Division
*3 Nippon Steel Engineering Yawata Co., Ltd.

2. Idea of Object-Oriented Design Application

The control of a process is considered in two parts: continuous control concerning the dynamic control of the process and sequential control achieving the operation of the process.

In the design of continuous control conventional instrumentation flow sheet images are important, so that objects at this unit level are taken as basic functional elements. Conventional hardware component images (like controllers) are converted to software components as objects, and the objects are linked by input and output data messages to construct a continuous control loop. Instructions for changing the control mode and for automatically changing the control start and end timing and the setpoints are issued as change request messages to the objects. The objects run the instructions to ensure the independence of the components and to facilitate the functional changes.

"Inheritance" and "aggregation" were utilized as characteristic functions of object-oriented design. The inheritance of control functions on a loop basis and the aggregation of control functions on a functional basis enabled the reutilization of large objects, such as combustion control and furnace pressure control objects, and helped to achieve a sizable improvement in the productivity of system building (**Fig. 1**).

Next for sequential control, it is necessary to eliminate all sequential control image programs that describe conventional processing procedures. When the sequential function is described by a program composed of logic portions only, the operating procedure is separated from the function and data of associated continuous control, making it difficult to grasp the entire control function. We thought that a sequence should be essentially designed to achieve an operating plan and that the sequence must be faithfully implemented or that the operations comprising the sequence must be definable by start conditions (events) and operating conditions such as interlocks. The system under consideration adopted the message link that is a large function of object-oriented design and the condition table that is used to run messages on the message link. This function made it possible to describe the start timing generation points and conditions, and various operating conditions. At the same time these items were implemented on the flow sheet of continuous control system design, making it possible to directly act on the control objects to be started and to build all control functions, including those of continuous control and sequential control, by the instrumentation flow sheet alone (**Fig. 2**).

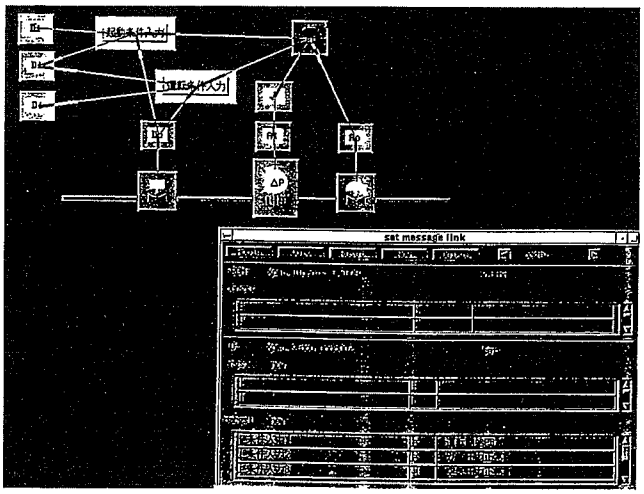

Fig. 2. Example of setup of starting and operating conditions by object-oriented design

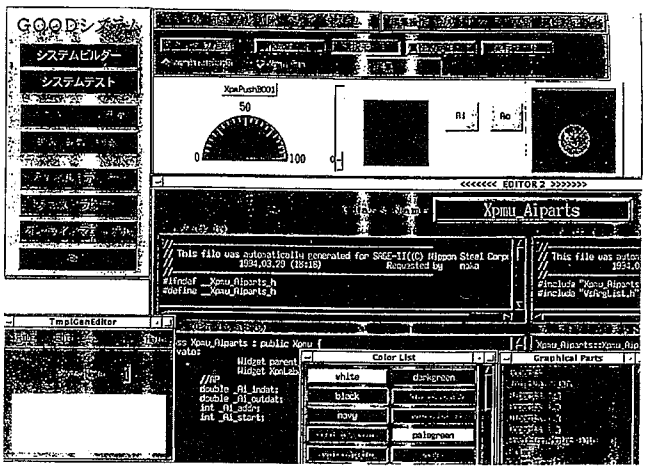

Fig. 3. GOOD (Global Object-Oriented Designer) system

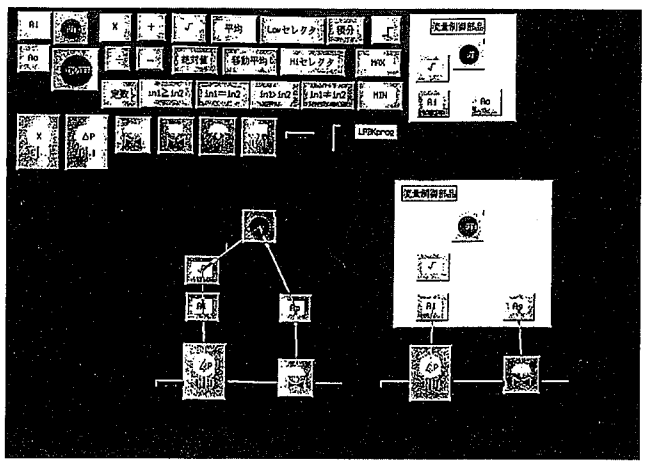

Fig. 1. Example of continuous control system design by object-oriented design

The accomplishment of visual programming by the graphical user interface (GUI) was indispensable in enabling conventional instrument engineers to perform object-oriented design and software production. The present GOOD (Global Object-Oriented Designer) system was completed by utilizing the GUI functions under development for use in the object-oriented design and production environment at the then Electronics Research Laboratories (present Systems Research & Development Center, Electronics & Information Systems Division) and by strengthening the GUI functions for process control (**Fig. 3**).

3. GOOD System

The mechanism and functions of the GOOD system are described in this chapter.

3.1 Operating environment

A low-cost UNIX*1 open system was adopted as a concrete operating environment for the purpose of eliminating hardware dependency as much as possible.

- Hardware

CPU section: Workstation (WS) or personal computer (PC)
PIO section: General-purpose sequencer

- OS

Solaris 2.x (WS)
SunOS 4.1.x (WS)
SVR4 (WS)
Linux (PC; since December 1996)
PANIX*2 (PC; since December 1996)

- GUI

Motif*3 (WS and PC)

- Compiler

C_{++} ver 3.1 or up

3.2 Mechanism of control system creation

System creation is entirely performed on the window. When the system builder is selected from the menu, tools containing various instrumentation control objects as described above are started. A blank window is opened to create the control system in question.

Necessary objects are arranged in the same way as an instrumentation flow sheet is created. As the flow sheet that is the most essential element of instrumentation functional design is generated, the objects are automatically arranged on the program (**Fig. 4**).

In this condition, however, the objects simply exist and do not operate. Using the message link mechanism, the link between the objects (program start information definition) is given by a message, and each message is started accordingly. With conventional feedback control, the output of each component is sequentially link connected from the input side toward the output side, and the result of each processing is outputted to the next object. The control start message is issued to each object by a timer at given intervals, and continuous control is performed at constant intervals (**Fig. 5**).

This message link can describe the conditions for the transmission of messages and change the contents or conditions of the messages. The functions associated with operating methods, such as batch control and automatic change of processing modes and parameters, can all be accomplished by the message link mechanism.

The descriptions of the GOOD system directly point to such functions as start conditions and operating conditions, unlike conventional programs. These functions can be directly described on the instrumentation flow sheet, so that there is no need to search for data addresses and program instructions.

Fig. 6 illustrates an example of combustion control system in which the functions of ignition, heatup and extinguishment are built by the control objects and message link mechanism. When the upper left heating start button is pressed, for example, the PID controller, shut-off valve, and combustion blower connected by the link lines start operating as a result. The upper right table describes the conditions to start the combustion blower according to the gas pressure. In this way all functions are represented on the flow sheet, and the event and condition parts of the operating plan are implemented in easy-to-understand form.

3.3 Test of system

The system is usually tested for program operation and overall operation, including both software and hardware.

In the program operation test, the objects (such as first-order lag and dead time) to simulate the controlled process are arranged

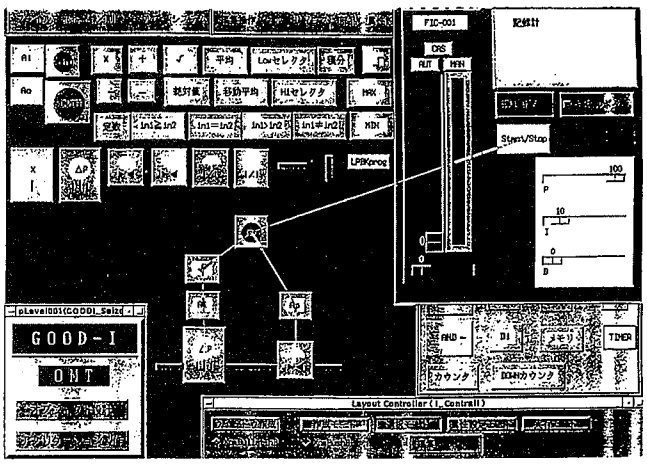

Fig. 5. Connection between objects by message link

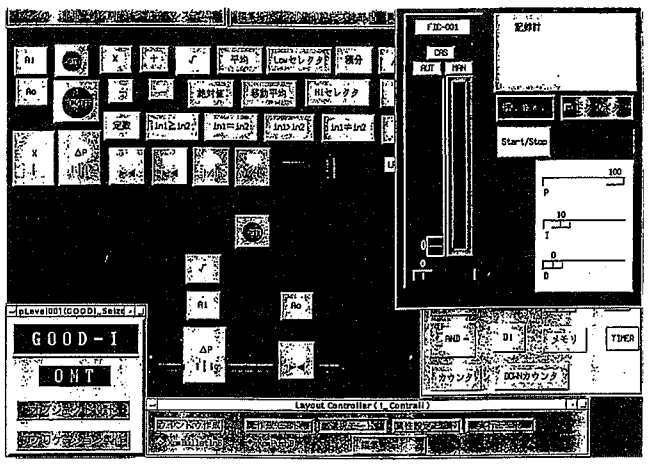

Fig. 4. Implementation of instrumentation control objects

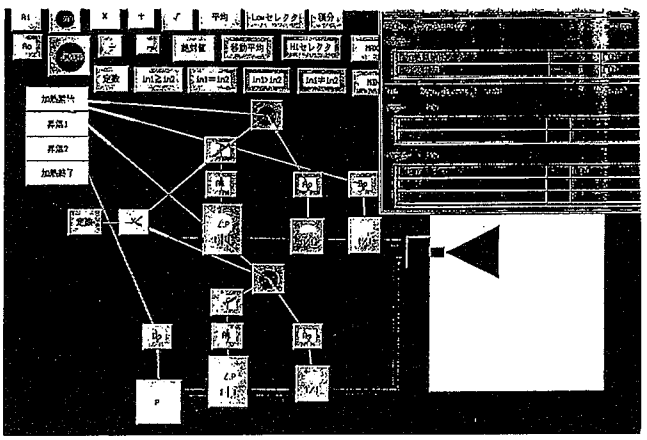

Fig. 6. Example of combustion control system design and implementation

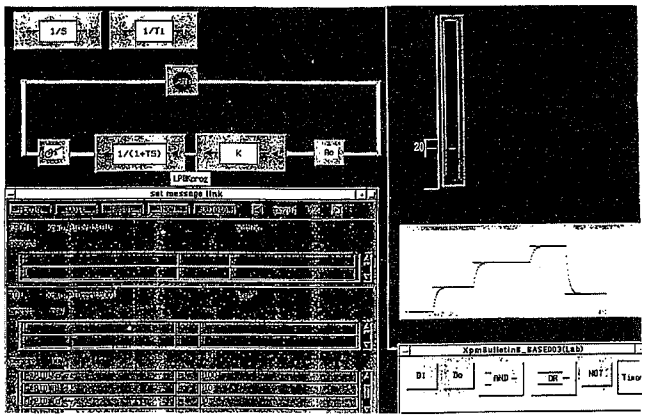

Fig. 7. Simulation of process control

on the instrumentation flow sheet and connected with the controller objects by message link. The test is then continuously performed under close to actual operating conditions. The debugging of the control functions is practically completed in this stage and the overall test run can be smoothly carried out (**Fig. 7**).

This simulation function proves powerful when the switch from the old system to the new system as a result of replacement, for example, must be made in a short period of time.

3.4 Change and addition of functions

The GOOD system can operate a control system on the spot as soon as it is created, so that the control system can be designed while actually checking the control functions as they are created. Any functional changes and additions are accommodated in this phase as much as possible. Actual plant instrumentation systems cause various changes and additions in the field adjustment stage, however. In this case it is desirable that these changes and additions should not adversely affect the other parts of the plant instrumentation system. The adverse effects of such changes and additions were unavoidable at the program level for conventional systems due to their structure.

The GOOD system, thanks to its object-oriented design feature, can change the functions by adding objects and changing message links, so that it can reduce the adverse effects of the changes on other objects.

4. Application of GOOD System to Box Annealing Furnace (BAF) Atmosphere Control System and Its Evaluation

The GOOD system was applied for the first time to the replacement of a box annealing furnace (BAF) atmosphere control system at the Yawata Works of Nippon Steel. Taking this project as an example, the technical points and evaluation of the GOOD system as applied to an actual control system and the future development of the GOOD system are described below.

4.1 System configuration

The main functions required of the BAF atmosphere control system are:

- Gas flow control
- Automatic gas changeover
- Pressure and temperature monitoring
- Fully automatic operation by reception of setpoints from host computer (process computer)
- Transmission of actual values to host computer (process computer)

Gas flow control, automatic gas changeover, and pressure and temperature monitoring are performed on each stand. The corresponding hardware configuration is as follows:

- Instrumentation control workstation: SPARC 75 MHz, MEM 64 MB
- R-PIO: General-purpose sequencer
- Instrumentation CRT: i486DX2 DOS/V, MEM 32 MB

The hardware configuration and the operator panel screen display are shown in **Figs. 8** and **9**, respectively.

4.2 Technical points of application

The control loops amount to a great number and are each equipped with flow control and gas changeover functions. The construction of these control loops by combining basic parts constitutes a considerable fabrication load and makes maintenance difficult. New objects were made by integrating basic parts as described below (**Fig. 10**).

Basic parts → Aggregation → Flow control parts → Addition of gas changeover function after inheritance → New flow control parts

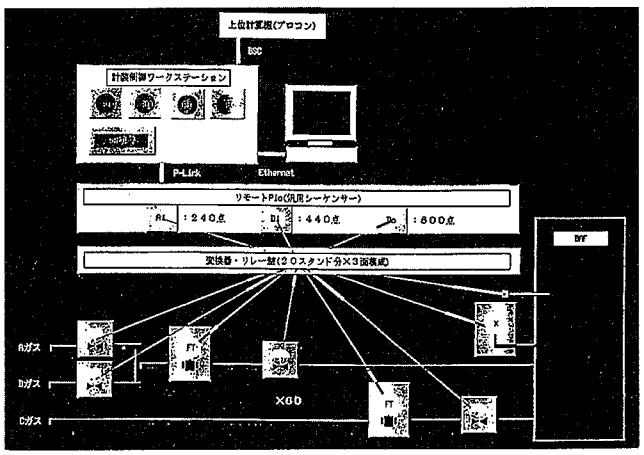

Fig. 8. Hardware configuration of BAF atmosphere control system

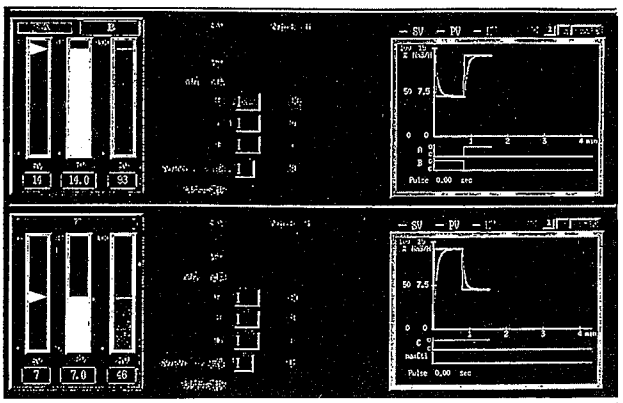

Fig. 9. Operator panel screen display of BAF atmosphere control system

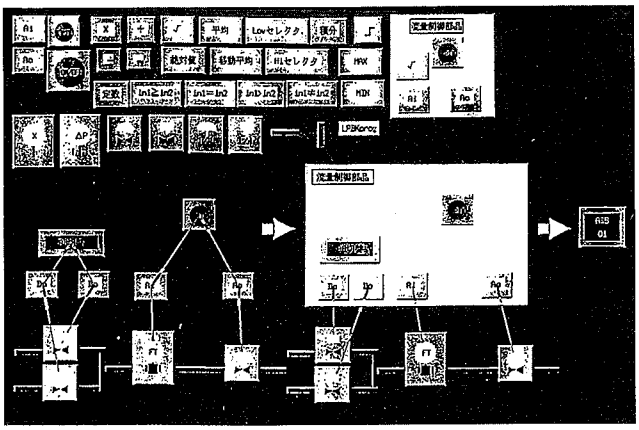

Fig. 10. Inheritance and aggregation of control objects in BAF atmosphere control system

New flow control parts → Addition of attribute to each control loop after inheritance and creation of instance for each stand

Another point is effective utilization of memory. Control functions created by using the GUI are excellent in maintainability, but always accompany graphics and consume a very large amount of memory. The function whereby the GUI not required for execution of control can be deleted during compilation for execution module creation was added to the automatic generation function, thereby enabling compilation without the GUI. This procedure reduced the memory consumption to one-fifth of the previous level.

4.3 Evaluation

The BAF atmosphere control system developed by using the GOOD system succeeded in achieving satisfactory response and control accuracy as an instrumentation control system for the BAF process and in accomplishing high software productivity as follows:

- Control period: Control of about 120 loops at 1.5-s intervals
- Control accuracy: Within 1% (with respect to setpoint after settling)
- Software productivity: 1.3 times higher than that of conventional systems

The GOOD system is compared with a conventional distributed control system (DCS) in detail in **Table 1**.

Table 1 Comparison of GOOD system with conventional DCS

Evaluation item		Conventional DCS	GOOD system
Software	Productivity	1.0 Internal data are all controlled by TAG numbers and they must be completely determined before programming, resulting in high load. Basic processing is packaged.	1.3 Linkage of internal data is achieved by lines between graphical symbols on screen, so that functions can be added and changed with low load and few mistakes. Basic processing is performed in parts, making it easy to expand parts in future.
	Reusability	It is difficult to reuse software between different manufacturers.	Software can be reused between different manufacturers.
Document reusability		Same as above	Same as above
Education period		About 1 month	About 1 to 2 weeks

5. Future Development

The GOOD system is now under rapid development on a personal computer to promote the downsizing of hardware. A DOS/V model for factory automation was selected as the personal computer by considering its environmental resistance. As of February 1997 the porting of the GOOD system proper to the personal computer is completed, and various control objects are being ported. The GOOD system will enter a full-fledged DCS area with the development of a distributed control system on a eight-loop basis, for example.

Both personal computers and workstations have phenomenally advanced in CPU performance in recent years. The results of simulation in January 1996 confirmed the ability of the GOOD system to perform six-zone table speed setup control and leveler roll position setup control at intervals of about 50 ms. The GOOD system is now applicable to simple electrical control systems.

Objects for such purposes as general-purpose database connection and material flow control are under development. The GOOD system will be developed as a system integration platform.

*1 UNIX is the registered trademark licensed by X/Open Company Limited in the United States and other countries.
*2 PANIX is the tradename of A. I. SOFT. INC.
*3 Motif is the registered trademark of the Open Software Foundation.

The other company names and product names are the trademarks or registered trademarks of the respective manufacturing or marketing companies.