

Description

Automation device

5 The invention relates to an automation device as claimed in the precharacterizing clause of claim 1. The invention furthermore relates to a universal, distributed and unified automation and management engineering and information system.

10 An automation device having the features of the precharacterizing clause of claim 1 is known from Siemens Catalog ST 70, 1995 Issue. A user uses a programming device to create a program for controlling a technical process, which comprises software function
15 modules, for example in the form of organization modules, program modules and entity data modules. The automation device is connected to the programming device via a bus system, via which the programming device transmits the control program to the automation
20 device. Furthermore, a control and monitoring device can be connected to the bus system, which is intended for process management and which allows the display of process images which comprise a plurality of image objects.

25 Frequently, it is necessary for software function modules of a control program to be transmitted by an automation device at one production location or from a software pool at this production location to an automation device at another production location.
30 Particularly if the production locations are a very long distance away from one another, for example owing to globalization of production

activities, these software function modules are transmitted via the "INTERNET" global network. Servers having suitable communications interfaces are required for this purpose, which on the one hand support the
5 INTERNET communications protocol, and on the other hand support the communications protocol of the automation devices. Owing to these different protocols and the architecture of the automation devices, it is impossible to include the software function modules for
10 the running time of the control program particularly if it is necessary to supply these software function modules to automation devices from different manufacturers.

The present invention is based on the object of
15 providing an automation device of the type mentioned initially, which is suitable for use in a globally distributed automation installation.

Furthermore, it is intended to specify a universal management engineering system in the form of a
20 programming device as well as a control and monitoring device for a globally distributed automation installation and, moreover, a management engineering and information system in the form of workstations and databank servers.

25 With respect to the automation device, this object is achieved by an automation device of the type mentioned initially and having the measures specified in the characterizing part of claim 1, and with respect to the universal management engineering system, this
30 object is achieved by the measures specified in the features of claim 6 and of claim 10.

Advantageous refinements of the invention result from the dependent claims.

The invention, its refinements and its advantages are explained in more detail in the following text with reference to the drawing, in which
5 an exemplary embodiment of the invention is illustrated and in which

Figure 1 shows a schematic illustration of a universal, distributed automation and management engineering and information
10 system,

Figure 2 shows a program translation structure diagram, and

Figures 3 and 4 show software function module execution systems (PLC object engine systems).
15

Two production locations 1 and 2 of a globally distributed automation installation are connected to one another via an "INTERNET" global network 3 which is known per se, suitable facilities 4, 5 being provided
20 which make it impossible to transmit unauthorized data to the data-processing components at the production locations 1, 2. The production locations 1, 2 comprise a plurality of devices (which are each provided with an INTERNET communications interface) in the form of
25 automation devices 6, program devices 7, control and monitoring devices 8 and workstations 9. These INTERNET communications interfaces allow the devices to communicate with one another using TCP/IP protocol. It is an essential requirement for an automation device
30 which, during control operation, processes a control program

(which is formed from a plurality of software function modules) cyclically and/or by interrupt control that these software function modules are designed to be loadable and such that they can be included in the control program for its running time. In order that this requirement is satisfied and the software function modules can be loaded directly into an automation device via the INTERNET and the INTERNET communications interface and can be included in the control program for its running time, the software function modules are designed on an object-oriented basis. The software function modules can be loaded and upgraded dynamically via the INTERNET and the automation device is provided with a software function module execution controller (PLC object engine system) which includes these software function modules in the control program and processes them during control operation.

A programming language which produces an object-oriented code from a source language and is intended for use in the INTERNET has been described in the book "Java!", Tim Ritchey, published 1995 by New Riders Publishing. A source language "JAVA C" is described there, from which an object-oriented Java-byte code can be produced. Further advantageous characteristics of this language means are, in particular, the portability of the code, as well as the mechanisms for error handling. The portability of the code ensures that an automation device having an execution controller in the form of a Java-byte interpreter can process the Java function modules supplied to the automation device via the INTERNET, irrespective of the processor hardware architecture of the automation device (manufacturer-independently). For performance reasons,

however, it is advantageous to provide the automation device with a Java processor 12, which processes the Java code directly.

5 The portability of an object-oriented code is indicated in Figure 2, which illustrates a program translation structure diagram.

A user uses a programming device to create (on the basis of a control task to be solved) a control program in the form of a contact plan KOP, a function plan FuP, and instruction list AWL, or in some other suitable form described, for example, in Standard IEC 1131. At the user level, the programming device translates the control program into a source language Qu, for example into the source language "JAVA C", or directly
15 (illustrated by interrupted lines in the figure) into an object-oriented machine language Ms, for example into the Java-byte code, which is loaded into automation devices AG1, AG2, AG3, AG4 of different architecture. At the user level, only one compiler is
20 required to create the machine codes for all automation devices. It is assumed that the automation device AG4 has a code generator G for processing the machine language Ms, by which means this automation device AG4 can process the code directly. It is furthermore
25 assumed that the automation devices AG1, AG2, AG3 are not provided with such a code generator, but comprise different processes PR1, PR2, PR3. In order that the automation devices AG1, AG2, AG3 can process the code MS, these automation devices are each provided with a
30 code interpreter IP1, IP2, IP3. These interpreters IP1, IP2, IP3 each produce

a code which can be interpreted by the processors PR1, PR2, PR3 during the time in which the control program is running.

The object-oriented software function modules are programmed by the respective programming devices 7 (Figure 1) at the production locations 1, 2 or by a programming device 14, which is likewise connected to the INTERNET. In addition to the control and monitoring devices 8 and the workstations 9, these programming devices 8, 14 are components of the management engineering system. The program devices supply these software modules to the appropriate automation devices via the respective INTERNET communications interface and the INTERNET. In the situation where, for example, modules have to be changed, the automation device 6 or a server 13 first of all transmits the appropriate software function module via the INTERNET to one of the programming devices 7. Finally, the programming device 7 completes or modifies this module and can transmit it once again to one of the automation devices. The programming device is furthermore provided with a software function module execution system (PLC object engine system; Bos, ExE, Wd, IO) which is intended for simulation of the control program.

The progress control and progress management of the processes to be controlled at the production locations 1, 2 are carried out by means of control and monitoring devices 8, which can likewise be connected to the INTERNET and can be operated on the INTERNET. One control and monitoring device 8, for example the control and monitoring device 8 at the production location 1, produces a control and monitoring program, comprising control and monitoring software modules,

for creating and displaying a process image which comprises a plurality of image objects, the image objects being related to software function modules of the control program (interactively). The control and
5 monitoring software modules are designed on an object-oriented basis and can be transmitted directly via the INTERNET. It is, of course, possible to create the process image on the programming device 7 and to supply it to the control and monitoring device 8, via the
10 INTERNET, for process management.

Local subsystems, for example in the form of intelligent in-field devices, are used in order to reduce the number of input and output components installed in an automation device in widely distributed
15 automation systems. The distributed automation and management engineering system has an intelligent in-field device, which is not illustrated here, to which at least one software function module of a control program can be supplied, which the in-field
20 device processes cyclically and/or by interrupt control during control operation, the software function module being designed to be loadable and such that it can be included in the control program for its running time. The software function modules are designed to be
25 object-oriented and can be loaded into the in-field device via the INTERNET and an INTERNET communications interface of the in-field device, the in-field device having a software function module execution system (PLC object engine system; Bos, ExE, Wd, IO) for inclusion
30 of the software function modules SFO1, ..., SFO4 and for processing of the control program.

The following text refers to Figures 3 and 4, which illustrate a software function module execution system (PLC object engine system) of an automation device and/or of an intelligent in-field device and/or of a programming device (for simulation of a control program). First of all, it is assumed that a control program has to be processed cyclically (Figure 3), which means that irrespective of the signal states of process inputs and outputs of a technical process to be controlled, for example, the CPU of an automation device cyclically

- a. interrogates the signal states of the process inputs and stores them in a process map of the inputs,
- 15 b. processes the control program step-by-step in accordance with the specifications in the control program to be processed, and
- c. stores the calculated signal states in a process map of the outputs, these signal states passing
20 from there to the process outputs.

The major components of the software function module execution system are units (which are programmed on an object-oriented basis) in the form of a bootstrap Bos, an input/output module IO, an Exe-engine object
25 ExE and a watchdog Wd. The watchdog Wd does not, of course, need to be designed as a software module, but can be implemented in hardware. In a practical exemplary embodiment of the invention, the units Exe-engine object ExE and the watchdog Wd are so-called
30 "threads". The function and method of operation of a "thread" are known from the document "Supporting Microsoft Windows 95, Student Workbook", 07/95 from the Microsoft Co. and thus do not need

to be explained in any more detail. A class of software function modules and a class of input/output modules are stored in the bootstrap unit Bos. These classes are created, for example, by a user in accordance with the specifications of a control task to be solved on a programming device and are transmitted, for example, to an automation device or to an in-field device. Before the start of the control operation, the bootstrap unit Bos uses the class of software function modules to produce software function module objects, and uses the class of input/output modules to produce input/output module objects. In the present example, only four software function module objects SFO1...SFO4 and one input/output module object IO are illustrated, in which a process map of inputs and outputs is stored and to which signal states of process inputs of the technical process can be supplied, and by means of which signal states can be supplied to process outputs of this technical process. Furthermore, at the start of control operation, the bootstrap unit Bos supplies the Exe-engine object ExE with a list of the software function module objects SFO1...SFO4 to be processed. At the start of control operation, the bootstrap unit Bos transmits a message Nas (method call) as a result of which the Exe-engine object ExE is started. In a first processing step, the Exe-engine object ExE supplies the watchdog Wd with a message Naw, which causes the watchdog Wd to monitor the cycle time of the Exe-engine object ExE. If the Exe-engine object ExE exceeds the intended cycle time, the watchdog Wd resets the Exe-engine object ExE, by the watchdog Wd transmitting a message Nar to the Exe-engine object ExE. Furthermore, if the cycle time is exceeded, the watchdog Wd

resets the outputs of the process map and the process outputs, in which case the watchdog Wd supplies a message Nia for this purpose to the input/output module object IO. Once the Exe-engine object ExE has started
5 the watchdog Wd, the Exe-engine object ExE starts control operation and first of all carries out a processing cycle which comprises the following steps:

- A) updating of the inputs of the process map, by the Exe-engine object ExE reading the input signal
10 states of the process map from the input/output module object IO, the signal states being interchanged between the objects by means of a method call Nae,
- B) processing in each case one processing step of the
15 software function module objects SFO1 ... SFO4, the Exe-engine object ExE supplying appropriate method calls NSF1 ... NSF4 to the software function module objects SFO1 ... SFO4,
20 accessing the process map of the input/output module object IO via method calls Nso, and, finally,
- C) updating of the outputs of the process map, by the Exe-engine object ExE writing the outputs of the
25 process map to the input/output module object IO, the signal states once again being interchanged between the objects by a method call Naa.

The input/output module object IO supplies, via
30 suitable interfaces, the process outputs of the technical process to be controlled and - as described - the software function module objects SFO1 ... SFO4 with the input and output signal states of the process map.

The situation is now assumed in which a control program is to be processed by interrupt control (Figure 4) which means that, when signal state changes occur at one of the process inputs of a technical process to be controlled, suitable control measures must be taken without delay. The same parts in Figures 3 and 4 have the same reference symbols. Only the differences relating to the cyclic processing of the control program will be described in the following text (Figure 3). In the case of interrupt-control processing of the control program, at the start of the control operation, instead of transmitting to the Exe-engine object ExE the list of software function module objects to be processed, as in the case of cyclic processing, the bootstrap unit Bos transmits a list of software function module objects "to be called" per process input. This means that a software function module object is allocated to each process input and, whenever a signal state change takes place at one process input, the software function module object allocated to this input must be started. In contrast to cyclic processing with respect to processing step B) the Exe-engine object ExE carries out a step D) during the processing cycle, which step D) comprises:

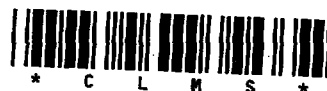
D) detecting signal state changes at the inputs of the process map and processing software function module objects allocated to these inputs.

The other tasks that occur in a company, such as stock keeping, production planning, use of labor resources etc., are combined under the generic term of management and information systems and are processed by

workstations or servers which are likewise connected to the INTERNET. These operate large databanks which maintain persistent object-oriented software function modules as representatives of process elements, and
5 cause them to be executed as required.

The software function modules of the automation device as well as those of the intelligent in-field device are compatible with those in the management engineering system and with those in the management
10 engineering and information system with respect to design, engineering, programming of automation tasks and communication between the components of the automation installation. Task extensions and task movements to other locations can thus be accomplished
15 clearly and more easily than in the past.

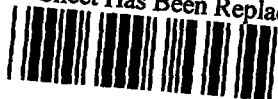
The invention provides a unified automation system which allows worldwide operation. This avoids the large-scale decoupling of automation devices and standard computers known from the prior art. The
20 capability is provided of modeling all the company elements such as processes, resources and organizations on an object-oriented basis. The implementation of software from these object models is carried out via a standard tool chain as part of a unified architecture.
25 In the process optimization and modification phase, simple software adaptations are possible by moving the locations of objects. Furthermore, this allows worldwide, flexible and distributed production, with central planning, simulation and optimization.



Patent claims

1. An automation device to which software function modules of a control program can be supplied, which control program is processed by the automation device cyclically and/or by interrupt control during control operation, the software function modules being designed to be loadable and such that they can be included in the control program for its running time, characterized
- in that the software function modules (SF01, ..., SF04) are designed on an object-oriented basis and can be loaded into the automation device via the INTERNET for an INTERNET communications interface of said automation device, and
 - in that the automation device has a software function module execution system (PLC object engine system; Bos, ExE, Wd, IO) for inclusion of the software function module objects (SF01, ..., SF04) and for processing the control program.
2. The automation device as claimed in claim 1, characterized
- in that the software function module execution system comprises an Exe-engine object (ExE), a watchdog (Wd), a bootstrap (Bos) and an input/output module object (IO) in which a process map of inputs and outputs is stored and to which signal states of process inputs can be supplied and by means of which signal states can be supplied to process outputs.
 - in that, before the start of control operation, the bootstrap (Bos) produces the software function module objects (SF01, ..., SF04) and

This Sheet Has Been Replaced



REPLACED SHEETS SG

- ☐ Description : paged to paged
- ☒ Claim(s) : paged ¹³ to paged ³⁰
- ☐ Drawing(s) : paged to paged
- ☐ Abstract : paged to paged



the input/output module object (IO) and supplies to the Exe-engine object (ExE):

- 5 - if the control program is processed cyclically, a list of the software function module objects (SFO1, ..., SFO4) to be processed,
 - 10 - if the control program is processed by interrupt control, a list of the software function module objects (SFO1, ..., SFO4) to be processed for each process input,
 - 15 - in that, at the start of control operation, the bootstrap (Bos) starts the Exe-engine object (ExE) which first of all starts the watchdog (Wd), which resets the Exe-engine object (ExE) when the cycle time is exceeded and, subsequently and cyclically
 - updates the inputs of the process map,
 - 20 - if the control program is processed cyclically, in each case processes one processing step of the software function module object (SFO1, ..., SFO4),
 - 25 - if the control program is processed by interrupt control, detects changes of signal states at the inputs and processes the software function module objects (SFO1, ..., SFO4) allocated to these inputs,
 - updates the outputs of the process map.
3. The automation device as claimed in claim 2, characterized in that the Exe-engine object (ExE) and the watchdog (Wd) are designed as "threads".

This Sheet Has Been Replaced





4. The automation device as claimed in one of claims 1 to 3, characterized in that the communications interface allows TCP/IP protocol communication.

5. The automation device as claimed in one of
5 claims 1 to 4, characterized in that the software function modules (SFO1, ..., SFO4) are Java-byte coded and can be created in the programming language "JAVA C" or in a programming language complying with Standard IEC 1131.

10 6. A programming device for creating software function modules of a control program which control program can be supplied to an automation device which, during control operation, processes the control program cyclically and/or by interrupt control, the software
15 function modules being designed to be loadable and being designed such that they can be included in the control program for its running time characterized

- in that the programming device creates the
20 software function modules (SFO1, ..., SFO4) designed on an object-oriented basis,

- in that the programming device supplies the software function modules (SFO1, ..., SFO4) to the automation device via the INTERNET and an INTERNET
25 communications interface, and/or

- in that the software function modules (SFO1, ..., SFO4) can be supplied to the programming device via the INTERNET and the INTERNET communications interface.

30 7. The programming device as claimed in claim 6, characterized in that, for simulation of the control program, the programming device





has a software function module execution system (PLC object engine systems; Bos, ExE, Wd, IO).

8. The programming device as claimed in claim 6 or 7, characterized in that the communications interface
5 allows TCP/IP protocol communication.

9. The programming device as claimed in one of claims 6 to 8, characterized in that the software function modules (SFO1, ..., SFO4) can be created in the programming language "JAVA C", which can run on the
10 programming device, or in a programming language conforming with Standard IEC 1131, and can be translated by the programming device to be Java-byte coded.

10. Control and monitoring device having control
15 and monitoring software modules of a control and monitoring program for creating and presenting a process image which comprises a plurality of image objects and is intended for process management, the image objects being related to software function
20 modules of a control program which an automation device processes during control operation, the control and monitoring software modules being designed to be loadable and such that they can be included in the control and monitoring program for its running time,
25 characterized

- in that the control and monitoring device creates the control and monitoring software modules on an object-oriented basis,
- in that the control and monitoring software
30 modules can be transmitted by the control and monitoring device via the INTERNET and





an INTERNET communications interface of the control and monitoring device, and/or

- in that control and monitoring software modules and/or process variables can be supplied to the control and monitoring device via the INTERNET and the INTERNET communications interface, and
- in that the control and monitoring device has a control and monitoring software module execution system (control and monitoring object engine system) for processing the control and monitoring software modules.

11. The control and monitoring device as claimed in claim 10, characterized in that the communications interface allows TCP/IP protocol communication.

12. The control and monitoring device as claimed in claim 10 or 11, characterized in that the control and monitoring software modules can be created in the programming language "JAVA C", which can run on the control and monitoring device, or in a programming language conforming with Standard IEC 1131, and can be translated by the control and monitoring device to be Java-byte coded.

13. Intelligent in-field device, to which at least one software function module of a control program can be supplied, which the in-field device processes cyclically and/or by interrupt control during control operation, the software function module being designed to be loadable and such that it can be included in the control program for its running time, characterized

This Sheet Has Been Replaced





- in that the software function module (SFO1, ..., SFO4) is designed on an object-oriented basis and can be loaded into the in-field device via the INTERNET and an INTERNET communications interface of the in-field device and
 - in that the in-field device has a software function module execution system (PLC object engine system, Bos, ExE, Wd, IO) for inclusion of the software function module (SFO1, ..., SFO4) and for processing the control program.
14. The intelligent in-field device as claimed in claim 13, characterized
- in that the software function module execution system comprises an Exe-engine object (ExE), a watchdog (Wd), a bootstrap (Bos) and an input/output module object (IO) in which a process map of inputs and outputs is stored and to which signal states can be supplied from process inputs and by means of which signal states can be supplied to process outputs.
 - in that, before the start of control operation, the bootstrap (Bos) produces the software function module objects (SFO1, ..., SFO4) and the input/output module object (IO) and supplies to the Exe-engine object (ExE):
 - if the control program is processed cyclically, a list of the software function module objects (SFO1, ..., SFO4) to be processed,
 - if the control program is processed by interrupt control, a list of the software function module objects (SFO1, ..., SFO4) to be processed for each process input,





- in that, at the start of control operation, the bootstrap (Bos) starts the Exe-engine object (ExE) which first of all starts the watchdog (Wd), which resets the Exe-engine object (ExE) when the cycle
- 5 time is exceeded and, subsequently and cyclically
- updates the inputs of the process map,
 - if the control program is processed cyclically, in each case processes one processing step of the software function

10 module objects (SFO1, ..., SFO4),

 - if the control program is processed by interrupt control, detects changes of signal states at the inputs and processes the software function module objects (SFO1, ...,

15 SFO4) allocated to these inputs,

 - updates the outputs of the process map.
15. The intelligent in-field device as claimed in claim 14, characterized in that the Exe-engine object (ExE) and the watchdog (Wd) are designed as "threads".
- 20 16. The intelligent in-field device as claimed in one of claims 13 to 15, characterized in that the communications interface allows TCP/IP protocol communication.
17. The intelligent in-field device as claimed in
- 25 one of claims 13 to 16, characterized in that the software function modules (SFO1, ..., SFO4) are Java-byte coded and can be created in the programming language "JAVA C" or in a programming language complying with Standard IEC 1131.





18. An automation system

- having at least one automation device as claimed in one of claims 1 to 5,
- having at least one programming device as claimed in one of claims 6 to 9, and/or
- having at least one control and monitoring device as claimed in one of claims 10 to 12.

19. The automation system as claimed in claim 18, having at least one intelligent in-field device as claimed in one of claims 13 to 17.

20. An automation installation

- having an automation system as claimed in claims 18 or 19 and
- having at least one workstation and/or one server, which have means for creating and processing object-oriented software function modules (SFO1, ..., SFO4).

This Sheet Has Been Replaced

