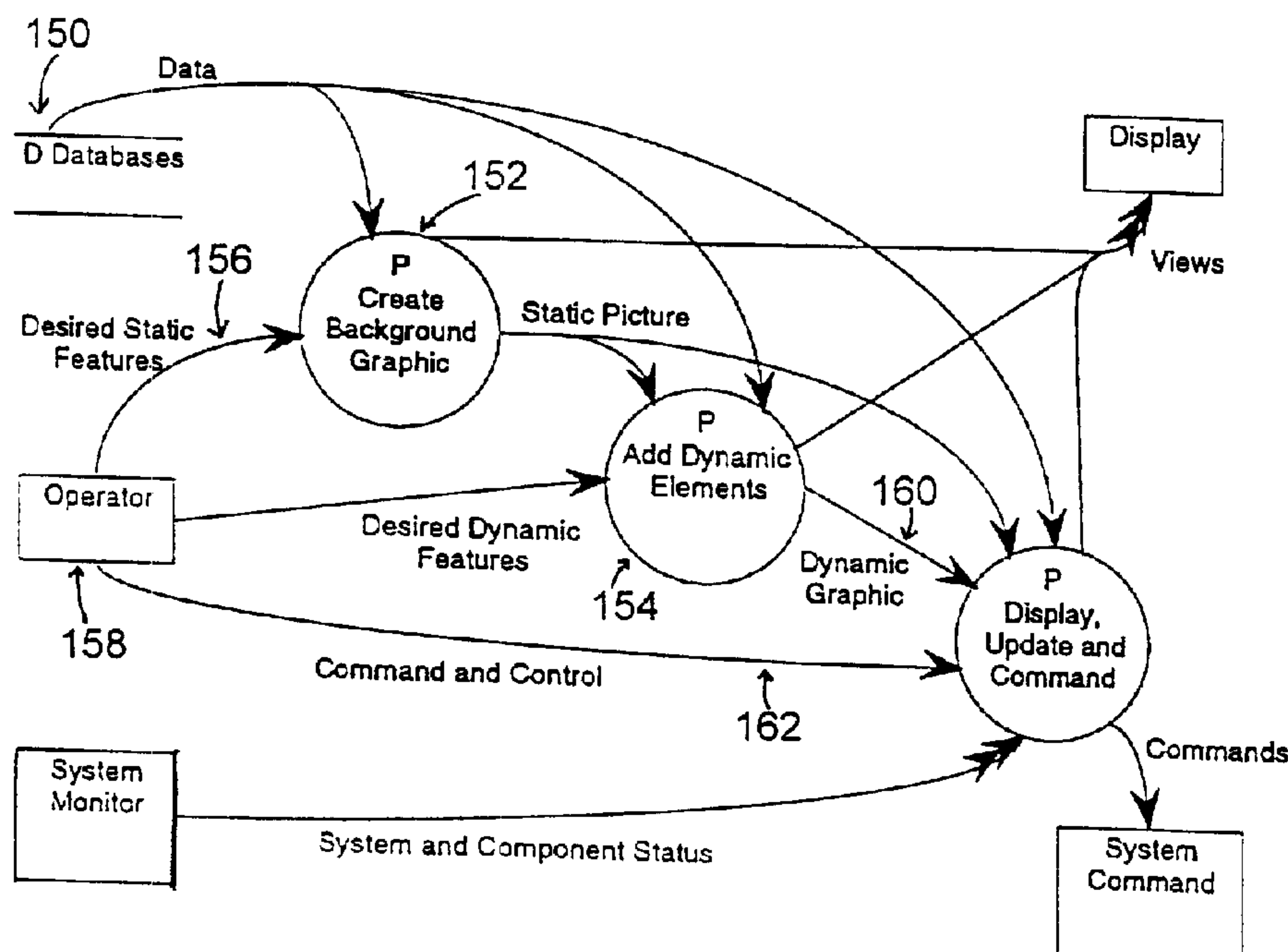




(22) Date de dépôt/Filing Date: 1998/04/27  
(41) Mise à la disp. pub./Open to Public Insp.: 1999/01/17  
(45) Date de délivrance/Issue Date: 2001/10/09  
(30) Priorité/Priority: 1997/07/17 (08/895,801) US

(51) Cl.Int.<sup>6</sup>/Int.Cl.<sup>6</sup> G05B 19/4063, G05B 19/414,  
G05B 19/4068  
(72) Inventeurs/Inventors:  
RAMIREZ, Pete, US;  
HAN, James K., US  
(73) Propriétaire/Owner:  
SIEMENS BUILDING TECHNOLOGIES, INC., US  
(74) Agent: OYEN WIGGS GREEN & MUTALA

(54) Titre : METHODE ET DISPOSITIF DE CONTROLE ET DE COMMANDE D'INFORMATION EN TEMPS REEL DANS  
UN SYSTEME DE CONTROLE AUTOMATIQUE DE BATIMENT  
(54) Title: METHOD AND APPARATUS FOR MONITORING AND CONTROLLING REAL-TIME INFORMATION IN A  
BUILDING AUTOMATION SYSTEM



(57) **Abrégé/Abstract:**

The building automation control system of the present invention includes a supervisory control station for monitoring and controlling process information (control points) from at least one environmental control device. The supervisory control station is provided with a memory for storing data, including a dynamic link library (DLL) containing: first data for displaying a plurality of independently selectable graphic objects, and link data specifying an association between a selected graphic object and the control point. The supervisory control station dynamically displays the received real-time process state information for the associated control point using stored state table data, link data, and run time values from the environmental control device.

1 METHOD AND APPARATUS FOR MONITORING AND CONTROLLING  
2 REAL-TIME INFORMATION IN A BUILDING AUTOMATION SYSTEM

3 ABSTRACT OF THE DISCLOSURE

4 The building automation control system of the present invention includes a  
5 supervisory control station for monitoring and controlling process information (control  
6 points) from at least one environmental control device. The supervisory control station is  
7 provided with a memory for storing data, including a dynamic link library (DLL) containing:  
8 first data for displaying a plurality of independently selectable graphic objects, and link data  
9 specifying an association between a selected graphic object and the control point. The  
10 supervisory control station dynamically displays the received real-time process state  
11 information for the associated control point using stored state table data, link data, and run  
12 time values from the environmental control device.

1 METHOD AND APPARATUS FOR MONITORING AND CONTROLLING  
2 REAL-TIME INFORMATION IN A BUILDING AUTOMATION SYSTEM

3 BACKGROUND OF THE INVENTION

4 The present invention relates to a method and apparatus for monitoring and  
5 controlling real-time information in a building automation system. More specifically, the  
6 present invention discloses a flexible, modular apparatus for dynamically displaying real-  
7 time process information. The present invention also discloses a uniform approach by which  
8 one section of code can access the services of another section of code, thereby promoting  
9 reuse of code and ease of maintenance.

10 Conventional systems for monitoring and controlling graphical displays are  
11 inflexible and utilize graphical controls or images which are rigidly linked or associated with  
12 specific control points during the coding of the user interface. In this context, a control point  
13 is a physical point connected to a sensor or apparatus such as an environmental control  
14 device which can either be monitored or commanded. Because the linkage or association is  
15 coded into the interface, these prior art systems are inflexible, and the subsequent addition  
16 of further control points requires significant program modifications. This inflexibility is a  
17 serious drawback which adds to the cost of upgrading a building automation system.

1           An additional drawback associated with prior art graphical display systems is  
2 that the various attributes associated with these graphical controls or images are fixed. Thus,  
3 changes to the size of a control, font, scale or the like could only be achieved by modifying  
4 and recompiling the underlying program. From a user perspective, this inflexibility is highly  
5 undesirable.

6           An additional drawback associated with prior art graphical display systems  
7 relates to the manner in which these systems access the software services provided by  
8 another piece of software, application, or library. A conventional graphical monitoring  
9 application would link to a library and then access the library's services by calling the  
10 functions in the library. Alternatively, the graphical display application would use the  
11 services provided by another application, which runs in an entirely separate process. In this  
12 scenario, the two local processes would communicate using an inter-process communication  
13 mechanism, which requires a protocol between the two applications (a set of messages  
14 allowing one application to specify its requests and the other to respond appropriately). In  
15 yet another scenario, the graphical display application uses the services provided by the  
16 operating system, i.e., the application makes a series of system calls, each of which is  
17 handled by the operating system.

18           Due to the lack of any standard approach for sharing services, a conventional  
19 graphical monitoring application may include any number of these different approaches  
20 to access the services provided by another. In turn, the use of these disparate approaches  
21 inhibits the reuse of code sections and makes it difficult to maintain the software. Thus, a  
22 single, uniform approach was needed for providing one code section with access to the  
23 services of another.



## SUMMARY OF THE INVENTION

Accordingly, in response to the problems discussed above, one object of the present invention is to provide an improved building automation control system that has a set of flexible, modular controls that enables a user to add, change or delete an association or link between a graphic object and a control point without requiring modification and/or recompiling of the underlying programming.

Another object of the present invention is to provide such a system wherein a user can modify attributes associated with a graphic object without requiring modification and/or recompiling of the underlying programming.

Another object of the present invention is to provide such a system that includes a graphical monitoring application including a uniform approach by which one section of software supplies its services to another section of software, thus promoting reuse of code sections, ease of development and maintenance of the software.

These and other objects of the present invention will be apparent from the following detailed description of the invention, while referring to the attached drawings.

## DESCRIPTIONS OF THE DRAWINGS

FIGURE 1 illustrates a graphical application control object according to the present invention;

FIG. 2 illustrates a client with an interface to a needle gauge control object;

FIG. 3 shows a standard model used to access services provided by various kinds of software according to the present invention;

FIG. 4 shows an example of reusing an object through containment according to the present invention;

FIG. 5 shows a user interface for modifying properties of a control;

FIG. 6 illustrates a flow diagram for modifying the properties of a control;

1 FIG. 7 illustrates an Associated Point Control;

2 FIG. 8 illustrates a flow diagram of a method of associating a graphical object  
3 with a control point according to a first embodiment of the present invention;

4 FIG. 9 illustrates an operator interface for entering a point name to be  
5 associated with a graphic object;

6 FIG. 10 illustrates an operator interface for entering a state information for a  
7 control point name to be associated with a graphic object;

8 FIG. 11 illustrates a diagram showing data and command flows for creating a  
9 dynamic control overlaid on a static background picture;

10 FIG. 12 illustrates a Needle Gauge Control;

11 FIG. 13 illustrates an Analog Bar Control;

12 FIG. 14 illustrates a Point Information Block;

13 FIG. 15 illustrates a flow diagram for creating a dynamic control according to  
14 the second embodiment of the present invention;

15 FIG. 16 illustrates an operator interface for specifying style information for an  
16 analog bar control; and

17 FIG. 17 illustrates an operator interface for specifying a style information for  
18 an Information Block Control.

## 19 DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

20 Broadly stated, the above objects are met or exceeded by the present modular  
21 graphical display method for monitoring real-time information in a building automation  
22 system. According to the method of the present invention the user selects a graphic object  
23 to be displayed from among a plurality of independent graphic objects, and links the selected  
24 object with a control point. Next, the user defines a normal state and at least one alarm state  
25 for the associated control point, and assigns state characteristics corresponding to each of the

1 defined states. Finally, the state of the control point is dynamically displayed on the  
2 supervisory control station using the alarm characteristics.

3 The above objects are also met or exceeded by the apparatus of the present  
4 invention for monitoring and controlling real-time information in a building automation  
5 system. Generally, in the preferred embodiment, the present invention is a building  
6 automation control system that includes a supervisory control station for monitoring and  
7 controlling process information (control points) received from at least one environmental  
8 control device. The supervisory control station preferably includes a CPU, communications  
9 apparatus for sending and receiving data to and from the environmental control device, a  
10 display for displaying graphical information, and an input unit. The supervisory control  
11 station receives real-time process state information for a selected control point from the  
12 environmental control device via the communications apparatus. Moreover, the real-time  
13 process state information is dynamically displayed using a control object. The supervisory  
14 control station is provided with a memory for storing data, including a dynamic link library  
15 (DLL) for each control object. The DLL for each control object contains first data for  
16 defining each of plural services as one or more object oriented linking and embedding (OLE)  
17 objects, each OLE object supporting one or more interfaces. In turn, each interface includes  
18 a number of methods, which are callable only by invoking the OLE object's interface. The  
19 DLL further contains second data for displaying at least one independently selectable  
20 graphical object, and link data specifying an association between a selected graphic object  
21 and the control point. The supervisory control station dynamically displays real-time process  
22 state information for the associated control point using stored state table data, link data, and  
23 run time values from the environmental control device.

24 According to the graphical application design architecture of the present  
25 invention, each section of software implements its services as one or more OLE objects,  
26 where OLE refers to the Microsoft Corp.'s object oriented linking and embedding language

1 (OLE) and is a registered trademark of the Microsoft Corp. Each OLE object supports one  
2 or more interfaces, each of which in turn includes a number of methods. A method is a  
3 function or a procedure that performs a specific action and is callable by other sections of



1 software using the object (the client of that object). The methods that make up each interface  
2 are related to one another in some predetermined manner. Clients can access the services  
3 provided by an OLE object only by invoking the methods in the object's interface, clients  
4 cannot directly access any of the object's data.

5 By manner of illustration, a needle gauge control according to the present  
6 invention is implemented as an OLE control object. This OLE control object supports an  
7 interface which includes methods such as SetMaxTickAngle, and GetMaxTickAngle, where  
8 SetMaxTickAngle is a method for defining a maximum tick angle, i.e., the maximum  
9 displacement of the needle. In turn, GetMaxTickAngle is a method which returns the  
10 maximum tick angle as a parameter. If at a later date the object's developer desired to add  
11 support for zooming to this same object, the object would need to support another interface  
12 (perhaps with a single method such as ZoomIn) with an argument specifying the  
13 magnification factor. The methods in each interface collectively provide related services,  
14 such as angle displaying or access to zooming.

15 A graphical representation of an application control object is illustrated in FIG.  
16 1. The object 30 is always implemented inside a server 32, shown as the rectangle around  
17 the object. This server can either be a dynamic-link library (DLL), which is loaded as needed  
18 when an application is running, or a separate process of its own. The object 30 is provided  
19 with several interfaces 34 which are used to invoke the methods in a graphical application  
20 control object's interface. An object typically provides its services through several other  
21 interfaces, and the client must have a separate pointer to each interface whose methods it  
22 plans to invoke. For example, a client of a needle gauge control object would need one  
23 interface pointer to invoke the methods in the object's tick angle interface and another  
24 pointer to invoke the method in the object's zooming interface. FIG. 2 shows a client 40  
25 having two methods (SetMaxTickAngle, GetMaxTickAngle) and a needle gauge control  
26 object 42 with two interfaces 34. One of the interfaces 34 has a pointer 44 to the client 40,

1 thereby enabling the needle gauge control object 42 to invoke the client's methods  
2 (SetMaxTickAngle, GetMaxTickAngle).

3 Every graphical application control object is an instance of a specific class. For  
4 example, one class may contain objects that provide monitoring functionality in a form of  
5 a needle gauge while another class may contain objects representing information in the form  
6 of an analog bar. Each class has a separate DLL containing all of the various data definitions  
7 used by the objects of that class.

8 A graphical representation of the present model for providing shared access to  
9 various services is depicted in FIG. 3. Once a client has a pointer to the desired interface on  
10 a running object, it can start using the object's services simply by invoking the methods in  
11 the interface. From the programmer's perspective, invoking a method looks like invoking  
12 a local procedure or function whereas in fact, the actual code that is executed might be  
13 running in a library, a separate process, as part of the operating system, or even on an entirely  
14 different system. Thus, for example in FIG. 3, Applications 50 and 52 are both running on  
15 operating system 54. Application 50 has two pointers 56, 58 which point to methods 60, 62  
16 respectively. Moreover, Application 50 has a pointer 64 to a method 66 in Application 68  
17 which is running on operating system 70. Using the present graphical application design  
18 architecture, clients need not be aware of these distinctions because everything is accessed  
19 in the same manner.

20 One of the primary goals of the present invention is to provide a building  
21 automation control system utilizing an object oriented approach which incorporates an  
22 effective reuse of existing code. By allowing the creation of reusable components (graphical  
23 monitoring OLE controls) with well defined interfaces and methods, the new design  
24 architecture provides an infrastructure that makes this possible.

25 Conventional object oriented technologies rely on implementing inheritance  
26 (in which a new object inherits the actual implementation of methods in an existing object)

1 as their fundamental mechanism for reusing existing code. This kind of inheritance is  
2 impractical for an object oriented system in a very heterogeneous programming environment  
3 because changes in base objects could have unexpected effects on objects that inherit  
4 implementation from them. For example, a situation in which a base object is either updated  
5 or released from memory independent of the objects which inherit from the base object, may  
6 cause unexpected results. In contrast, the present graphical display system does not rely on  
7 such inheritance, and instead provides for reuse through a concept termed containment.

8           Containment in the present invention provides for reuse by relying on the  
9 relationship between objects. An outer object is one that reuses the services of an inner  
10 object. Thus, the the outer object simply acts as a client of the inner object. As shown in  
11 FIG. 4, the outer object 72 invokes the methods of an inner object 74 in order to carry out its  
12 own functions; however, the outer object 72 does not make those methods visible to its client  
13 76. Instead, when the client 76 invokes a method in one of the outer object's interfaces 34,  
14 the execution of that method includes a call to a method in an interface 34 of the inner object  
15 74. In other words, the outer object's interface contains methods that call the inner object's  
16 methods.

17           The use of control object is transparent to the end user, who only sees a typical  
18 graphical user interface displaying various buttons to click on, sliders to drag, text boxes to  
19 fill in, and so on. Most operating system such as Windows NT allow applications to present  
20 this kind of interface to the user. In the prior art or design architecture, the code necessary  
21 to display and work with these user interface features was an integrated part of the system  
22 or of a specific application. In contrast, according to the graphical application design  
23 architecture of the present invention, the user interface features are not an integrated part of  
24 the system. Therefore, the user interface may be designed separately or in parallel with the  
25 development of the control system. However, from the user perspective, the interface and  
26 the control system constitute a seamlessly integrated whole. Specifically, the user sees a



control container that includes a number of OLE controls (such as the Analog Bar, Point Information Block, Needle Gauge, Hyperlink, and Associated Point control each of which will be described in detail below). A control container is analogous to an OLE compound documents container, but it supports a few extra interfaces for working with OLE controls. Each control is plugged into the container, and each commonly presents its own user interface as an embedded object supporting in-place activation. A slider on the screen, for example, can be used to command a specified output point to a new value simply by moving it along its axis. By moving the slider, the operator interacts with the code that actually triggers an OLE control object. The integrated user interface which the end user sees as a single application, is in fact a control container populated with various discrete monitoring OLE controls, each providing part of the complete solution.

The container provides a direct, intuitive way for a user to examine and modify a control properties. By manner of illustration, FIG. 5 depicts a user interface 82 termed a Properties window which enables a user to directly view and modify a control's properties. However, not all control containers provide this kind of access. Moreover, rather than letting each control display provide its own idiosyncratic kind of user interface, the notion of property pages is used. With property pages, the properties of any control can be examined and modified in a standard way using a standard user interface. The Properties window interface 82 is intuitively easy to grasp, although fairly complex to design. Each page in the tabbed dialog box is provided by its own property page object. A styles property page is shown in FIG. 5.

A property page object is an OLE object, complete with its own class ID (CLSID) that supports the IPropertyPage interface. The CLSID is a 128-bit unique number which identifies a particular server implementation to OLE, and the IPropertyPage interface is a standard OLE interface which must be implemented in order for the object to be an OCX control.



1           Interaction between a container and a control object will now be described with  
2 reference to FIG. 6. As shown, a control object 88 may implement an ISpecifyPropertyPage  
3 interface (shown as dashed arrow 90) to allow its container 92 to learn about the property  
4 page objects 94, 96 it supports. When a user asks to see a control's properties, the control  
5 container 92 calls a method which in turn gets a list of CLSID's, one for each property page  
6 object the control supports.

7           Once the container 92 knows which property pages 94, 96 a control supports,  
8 the container 92 creates a property frame 98, which in turn activates an instance of each  
9 property page object using the CoCreateInstance method. For each property page object, the  
10 property frame provides a page site object 100, 102, each of which supports the  
11 IPropertyPage Site interface. Using this interface, a property page object can learn about the  
12 property frame that created it. Each property page object presents its page to the property  
13 frame, which assembles them into a tabbed dialog box like the one shown in FIG. 5.

14           Using this structure of objects, a user can directly examine or modify the  
15 control's properties. Any changes are communicated from the property frame to the property  
16 page objects via IPropertyPage and then by the property page objects directly back to the  
17 control itself through its IDispatch interface. Moreover, a graphics control container may be  
18 equipped with properties to facilitate the integration of a control into the overall display  
19 environment, i.e., when a new control is inserted into the graphics container. For example,  
20 an inserted control may adopt the current background color, cause its own text to appear in  
21 the container's default font, or decide whether or not it should be in a dynamic (runtime) or  
22 edit mode.

23           To allow a control to learn about the environment in which it finds itself, the  
24 graphics control containers supports ambient properties, which include a default background  
25 color, default font, mode, and more. A control can learn about these properties via an  
26 IDispatch interface. Once it has obtained the values of its container's ambient properties, a

1 control can modify its own properties to these values, allowing it to visually integrate with  
2 the other controls in the container.

3 A method of monitoring a control point using an associated object control is  
4 explained with reference to FIGS. 7 - 9. In particular, FIG. 7 is a representative illustration  
5 of an associated object control, and FIG. 8 is a flow diagram for creating an associated object  
6 control according to the present invention.

7 An associated object control according to the present invention is a composite  
8 graphic including several independently selectable objects. The composite graphic depicts  
9 a system or a portion of a system in pictorial form, and dynamically displays state  
10 information in an intuitive manner. By manner of illustration, FIG. 7 shows a portion of an  
11 air handling system 110 in a building automation system, including several independently  
12 selectable objects, 112, 114, 116.

13 Using the method of the present invention, a user associates a selected object  
14 with a desired control point, and selects various style characteristics used to visually display  
15 the present state of the control point. This method is now explained with reference to FIG.  
16 8.

17 First, a user selects a composite graphic from a library of stored composite  
18 graphics (block 122). Next, a user selects one of the independently selectable objects using,  
19 for example a keyboard, mouse, touch sensitive screen or other data entry method (block  
20 124).

21 Then a user associates the selected object by entering in a control point or  
22 selecting a control point from a list of defined control points (block 126). An operator  
23 interface for entering or selecting a control point is shown, for example, in FIG 9. A point  
24 name according to the present invention is a string of alpha-numeric characters separated by  
25 one or more delimiters which hierarchically describe a unique point. Using the interface of

FIG. 9, the user can elect to have the entire point name displayed. Alternatively, the user can specify a delimiter and elect to have one of the first and last fields displayed.

Referring again to FIG. 8, the user designates the state(s) for the associated control point by specifying a set point and one or more offsets (block 128). An operator interface for entering this information is shown, for example, in FIG 10. The set point designates the center point of the desired or normal state for the control point. A first alarm state is specified by entering an offset. For example, if for a given control point is 50 and a first offset is 10, then the normal state would encompass values ranging from 40 to 60. Any value falling outside of this range would trigger an alarm state. Moreover, the user elect to specify additional alarm states by entering additional offsets.

Referring again to FIG. 8, the user next selects the state colors used for displaying each of the states which the control point can assume (block 130). Thus, a normal state could, for example, be designated by the color green, and a first alarm state by the color orange.

Moreover, the user can elect to have the graphic object and/or the point name blink when the control point enters an alarm state. Finally, the user alerts the system that the state of the associated object should be dynamically updated by refreshing the displayed object in accordance with the specified state characteristics (block 132).

An important characteristic of the present invention is that the user can add, change or delete the link between a graphic object and a control point. This can be accomplished using the same procedure described for linking an object with a control point. Thus, the addition new control points to be monitored can be accomplished without requiring reprogramming of the supervisory control system.

Another important characteristic is that the display characteristics associated with each state can easily be customized to suit the individual preferences the user using the described operator interfaces.



1           According to another aspect of the present invention, the composite graphic can  
2 be overlaid over a static background picture. For example, a graphic representing an  
3 environmental control device can be overlaid over a static layout of the building. This  
4 feature facilitates the use of existing bit mapped graphics to provide the user with a graphic  
5 representation of the location of a control within a larger system, or provide an aesthetic  
6 background.

7           The data and command flows for creating a dynamic control including a static  
8 background picture are now explained with reference to FIG. 11. In particular, a database  
9 150 contains data for creating one or more static background pictures 152 and one or more  
10 composite graphics 154 which include one or more independently selectable graphic control  
11 objects. In process step 156 (shown by an arrow), an operator 158 elects to create a  
12 background graphic by selecting a static picture from among the pictures stored in the  
13 database 150. In the next process step (block 160), the operator selects a desired composite  
14 graphic to be overlaid on the background pictures, and creates a dynamic graphic by  
15 associating selected ones of the independently selectable objects with desired control points.  
16 Moreover, the operator selects desired dynamic features including alarm and display  
17 characteristics for each associated object. In the final process step (block 162), the operator  
18 158 commands and controls selected ones of the associated control points using the dynamic  
19 graphic.

20           According to yet another aspect of the present invention, each independently  
21 selectable graphic object is a separate OCX control composed of one or more dynamic sub-  
22 objects. The implementation of each graphic object as a separate OCX control facilitates the  
23 development of modular controls. Additionally, this aspect of the present invention enables  
24 a faster refresh response, since an individual graphic object may be refreshed, independent  
25 of the background and the other graphic objects.



1 A second embodiment of the graphical display method of the present invention  
2 is now described with reference to FIGS. 12 - 15. Briefly, there are three types of controls  
3 which can be utilized to dynamically display run time values and state information according  
4 to the second embodiment of the present information. Namely, a needle gauge control,  
5 analog bar control, and point information block. A needle gauge control according to the  
6 present invention is shown in FIG. 12, an analog bar control is shown in FIG. 13, and a point  
7 information block control is shown in FIG. 14.

8 These controls are distinguishable from prior art building automation controls  
9 in several aspects. One of these aspects pertains to the modularity of the controls.  
10 Specifically, each of the controls of the present invention is a separate OCX stored in an  
11 individual data file. A user can elect which control points are monitored, and can specify the  
12 type of control used to monitor the control point. Further, each instance of a control is  
13 independent of other controls, with each control being refreshed, as needed, during its own  
14 processing cycle. In other words, the OCX controls of the present invention are  
15 independently and dynamically updated in a multi-threading environment.

16 More particularly, each control is an object composed of static and dynamic  
17 sub-objects. Static sub-object are those parts of the object which are not refreshed or  
18 repainted during a refresh cycle. By manner of illustration, in the needle gauge control 168  
19 of FIG. 13, the scale 170, and control point name 172 are static sub-objects. In contrast, the  
20 arrow segment 174 and arrow head 176 are dynamic sub-objects because they are  
21 dynamically refreshed to reflect the current run time values of the control point. The  
22 independence of the static sub-objects from the dynamic sub-objects according to the present  
23 invention facilitates rapid multi-threading operations as processing resources during each  
24 processing cycle need only be utilized to refresh the dynamic sub-objects.

25 A method of creating a dynamic control will now be explained with reference  
26 to FIG. 15. This method closely parallels the method of creating an associated object

1 control, the main difference being the customizable options associated with each individual  
2 control.

3 The user selects a needle gauge, for example, from the group including needle  
4 gauge, analog bar, and point information block (block 182). Next the user selects a point to  
5 be linked or associated with the control (block 184).

6 Subsequently, the style characteristics for the selected control are specified  
7 (block 186). The ability of the user to modify the style and display options of each type of  
8 control is another important feature of the present invention. In contrast, prior art graphical  
9 controls in building automation systems were inflexible because the various style and display  
10 characteristics were coded into the graphical display program. Thus, modifying the size of  
11 a prior art control would require programming changes. In contrast, style and display  
12 characteristics of a control according to the present invention can be modified by a user  
13 without requiring programming changes.

14 For example, FIG. 5 shows an operator interface 82 for setting style  
15 characteristics for a needle gauge. Using the operator interface 82, the user elects whether  
16 to show an arrow head and/or the pivot point of the needle. Moreover, the user specifies the  
17 direction of rotation, pivot position, as well as the maximum and minimum needle deflection  
18 (in degrees).

19 In contrast, if the selected control was an analog bar, then the user would be  
20 able to select the orientation of the bar (block 186). For example, FIG. 16 shows an operator  
21 interface 200 wherein the user can choose a horizontally oriented analog bar or a vertically  
22 oriented bar. Further, the user can elect the direction in which bar will deflect. Thus, for a  
23 horizontally oriented bar, the user chooses whether the bar will increase to the left or to the  
24 right, and for a vertically oriented bar, the user chooses whether the bar will increase in an  
25 upward or downward direction.

1           Still further, if the type of control was a point information block, then in step  
2 186, the user can select various style characteristics using an operator interface 210 (FIG.  
3 17). Among other things, the user can elect whether to display the name of the associated  
4 control point, a descriptor, its status, and its alarm priority.

5           Then, in step 188, the user defines the state(s) for the control point, and  
6 specifies a unique color corresponding to each state. Thus, in addition to dynamically  
7 displaying run time values for a control point, state information is displayed using the colors  
8 associated with each state.

9           The user then selects display options for the selected control; such as a title to  
10 be displayed, whether to display the associated point name and alarm marks, as well as  
11 specifying the axis labels and associated fonts (block 190). Next, the user selects the state  
12 colors used for displaying each of the states which the control point can assume (block 192).

13           Finally, the state of the linked point is dynamically displayed by refreshing the  
14 dynamic-sub-objects in accordance with the specified alarm characteristics and run time  
15 values (block 194).

16           As discussed above, a user has the ability to specify the alarm state  
17 characteristics for each type of control. Thus, a user can specify that the color of the control  
18 change color to reflect the alarm state of the control point in addition to showing the actual  
19 run time value via needle deflection or displacement of the analog bar.

20           From the foregoing, it should be appreciated that a modular graphical display  
21 method for monitoring and controlling dynamic process information in a building automation  
22 system, which utilizes reusable control objects for graphically displaying real-time process  
23 state information has been shown and described. It should be further be appreciated that a  
24 building automation control system utilizing this modular graphical display method has been  
25 shown and described.

1           While various embodiments of the present invention have been shown and  
2 described, it should be understood that other modifications, substitutions and alternatives  
3 may be apparent to one of ordinary skill in the art. Such modifications, substitutions and  
4 alternatives can be made without departing from the spirit and scope of the invention, which  
5 should be determined from the appended claims.

6           Various features of the invention are set forth in the appended claims.



## WHAT IS CLAIMED IS:

1. A modular graphical display method for monitoring and controlling dynamic process information in a building automation system, comprising:
  - 5 selecting a graphic object to be displayed from among a plurality of independent graphic objects;
  - linking said selected object with a control point;
  - defining a normal state and at least one alarm state for said control point;
  - 10 assigning state characteristics corresponding to each of said defined states; and
  - dynamically displaying the state of said control point using said state characteristics.
2. A modular graphical display method according to claim 15 1 wherein said state characteristics are selected using a state colour table.
3. A modular graphical display method according to claim 2 wherein said alarm characteristics include displayed text and blinking said graphic object.
4. A modular graphical display method according to claim 20 1 wherein said graphic object is overlaid over a static background picture.
5. A modular graphical display method according to claim 1 wherein each said independent graphic object is an OCX control.
6. A modular graphical display method for monitoring and controlling dynamic process information in a building automation
  - 25 system, comprising:
    - selecting a graphic object from a group of graphic objects comprising an information block control, an analog bar control, and a

gauge needle control, said selected graphic object comprising at least one static sub-object and at least one dynamic sub-object;

5 specifying style characteristics for said selected graphic object from a predetermined range of style characteristics;

linking said selected object with a control point;

defining a normal state and at least one alarm state for said control point;

10 assigning state characteristics to each said defined state; and dynamically displaying run time values and state information of said control point by refreshing said dynamic-sub-objects in accordance with said state characteristics.

7. A modular graphical display method according to claim 15 6 wherein said graphic object is an analog bar and includes a caret for commanding said linked point, said caret being a dynamic sub-object.

8. A modular graphical display method according to claim 7 wherein said caret is moveable only if a user profile indicates that a user may command said linked point.

20 9. A modular graphical display method according to claim 7 wherein said analog bar has user adjustable attributes including bar orientation, size, scale, and alarm limit.

10. A modular graphical display method according to claim 6 wherein said graphic object is capable of being navigated to a point 25 commander for commanding the state of said point.

11. A building automation control system, comprising:

at least one environmental control device;

a supervisory control station including a CPU, memory means for storing data, communications means for sending and receiving data to and from said environmental control device, display means for displaying graphical information, and an input unit, said supervisory control station receiving real-time process state information for at least one control point from said environmental control device;

at least one control object for graphically displaying said real-time process state information for a selected said one control point;

said memory means storing a dynamic link library (DLL) for each said at least one control object, said DLL containing:

first data for defining each of plural services as one or more OLE objects, each said OLE object supporting one or more interfaces, each said interface including a number of methods, said methods being callable only by invoking said OLE object's interface;

second data for displaying at least one independently selectable graphical object;

link data specifying an association between said graphic object and said control point;

a state table containing user selectable change of state information for graphically displaying a change of state for said control point; and

style data specifying optional, user selected preferences including at least one of displayed fields, position, and size;

wherein said control station dynamically displays said received real-time process state information for said control point using said state table data, said link data and said first data.



1           12.    A building automation control system according to claim 11 wherein  
2 each said graphic object is comprised of graphic sub-objects selected from the group of static  
3 sub-objects and dynamic sub-objects, said dynamic sub-objects being capable of being  
4 refreshed independent of said static sub-objects;

5           said state table data including data for each state which said dynamic sub-  
6 objects can assume.

1           13.    A building automation control system according to claim 11 further  
2 comprising:

3           data for displaying a static background graphic stored in said memory;

4           wherein said display means displays at least one graphic object overlaid on said  
5 static background graphic, said graphic object being addressed and refreshed independent of  
6 said static background graphic.

1           14.    A building automation control system according to claim 11 further  
2 comprising:

3           hyper-link data specifying an associated graphic object;

4           wherein said display means displays at least one graphic object overlaid on said  
5 static background graphic, said graphic object being addressed and selected independent of  
6 said static background graphic.

1           15.    A building automation control system according to claim 11 wherein  
2 said control object is an analog control, and said DLL further comprises:

3           analog bar style data specifying the orientation of said analog control, alarm  
4 marks, axis labels, point name, and high and low tick numbers specifying the displayed range  
5 of values for said control point.



1                   16.    A building automation control system according to claim 11 wherein  
2   said control object is a needle gauge control, and said DLL further comprises:  
3                   analog bar style data specifying the orientation, size and position of said needle  
4   gauge control.

1                   17.    A building automation control system according to claim 16 wherein  
2   said DLL further comprises data specifying a selected needle style from among a plurality  
3   of needles styles.

FIG. 1

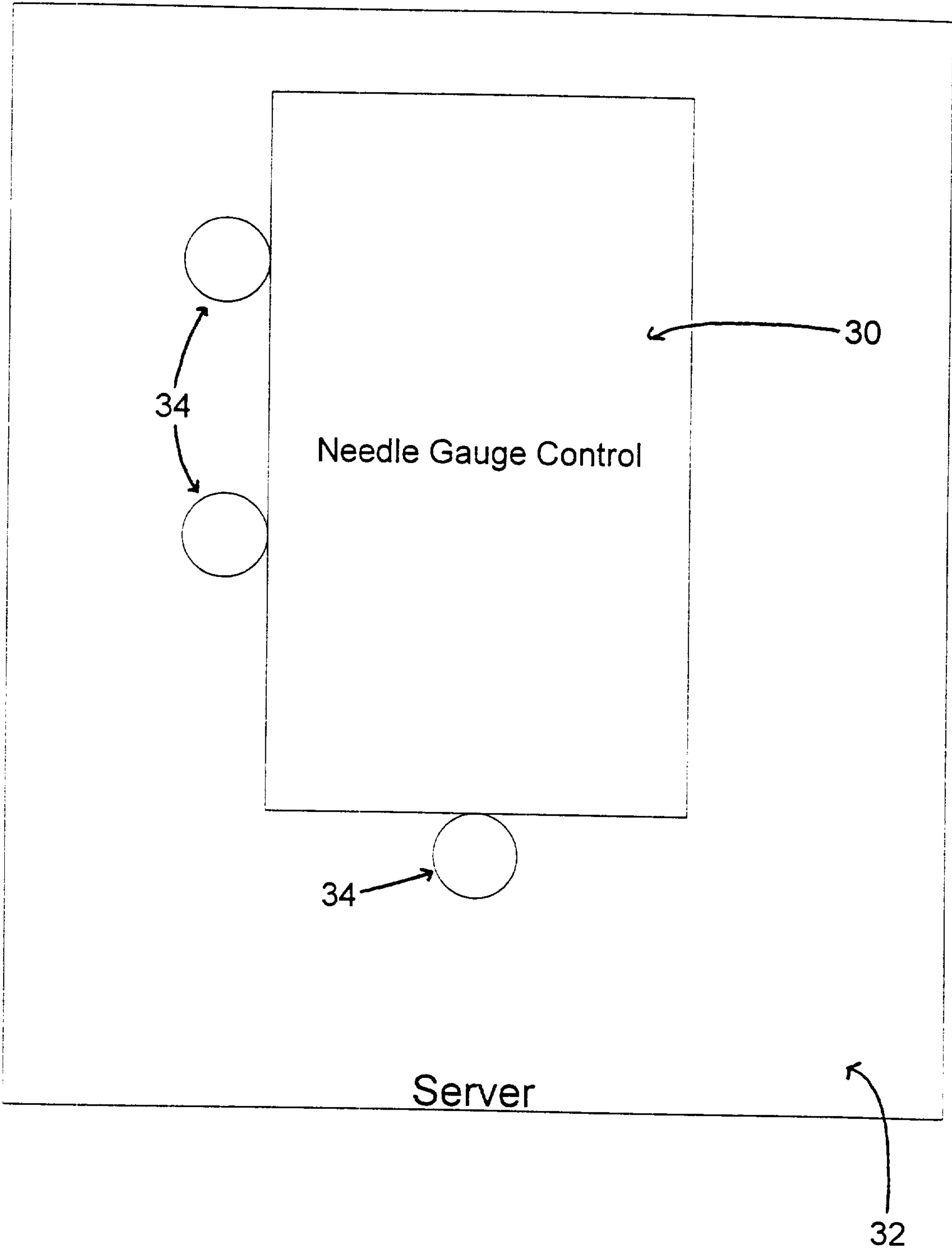


FIG. 2

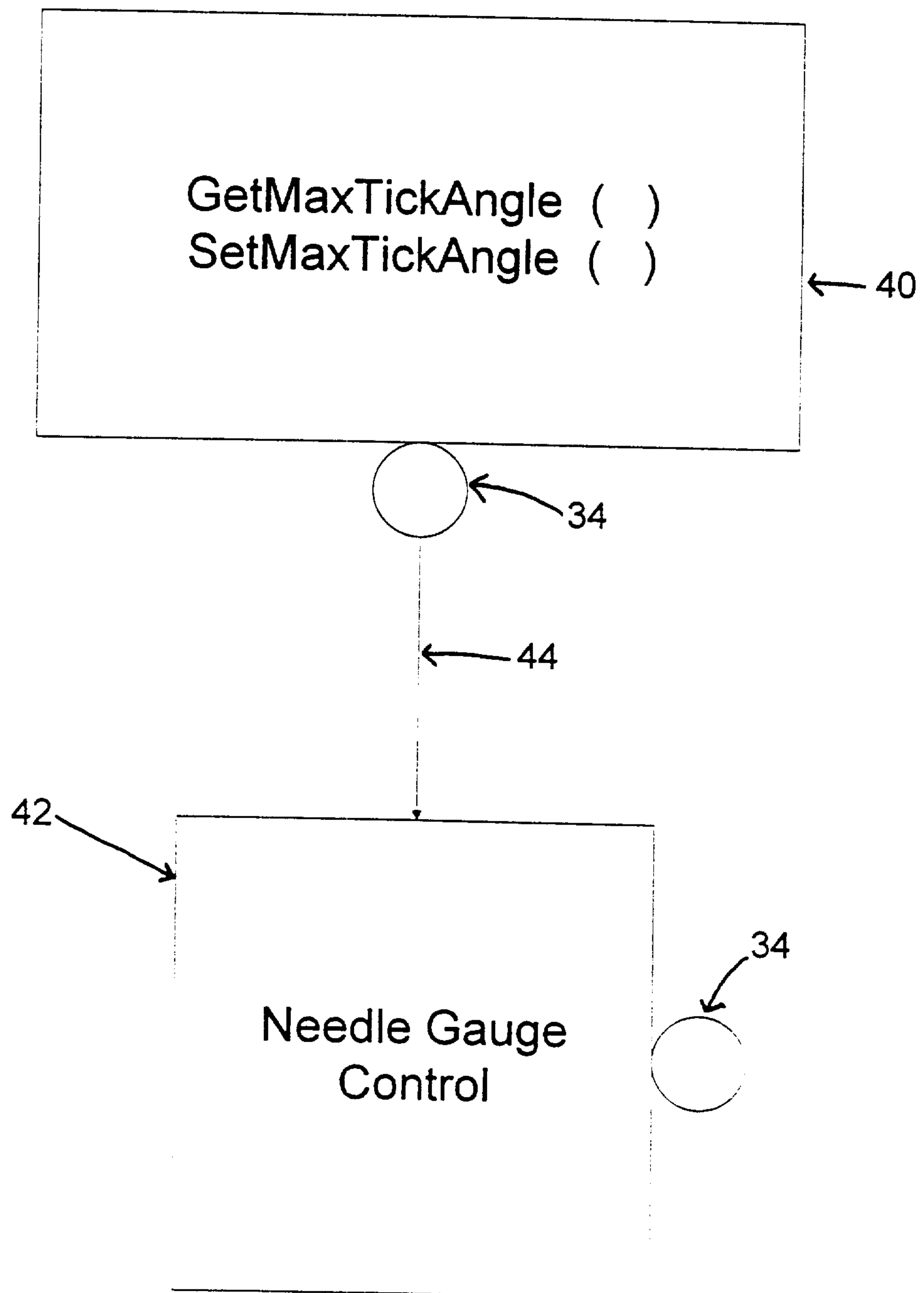


FIG. 3

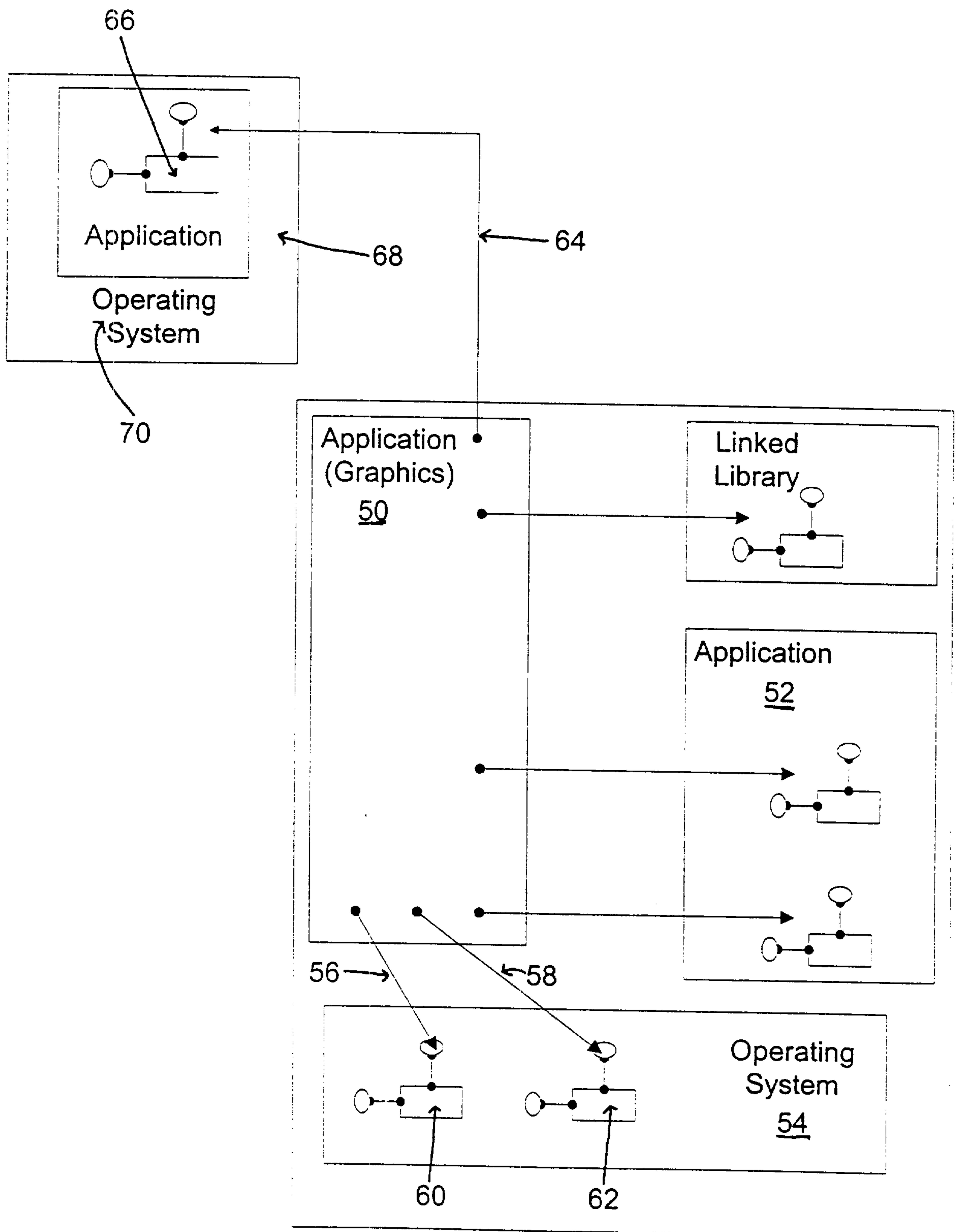
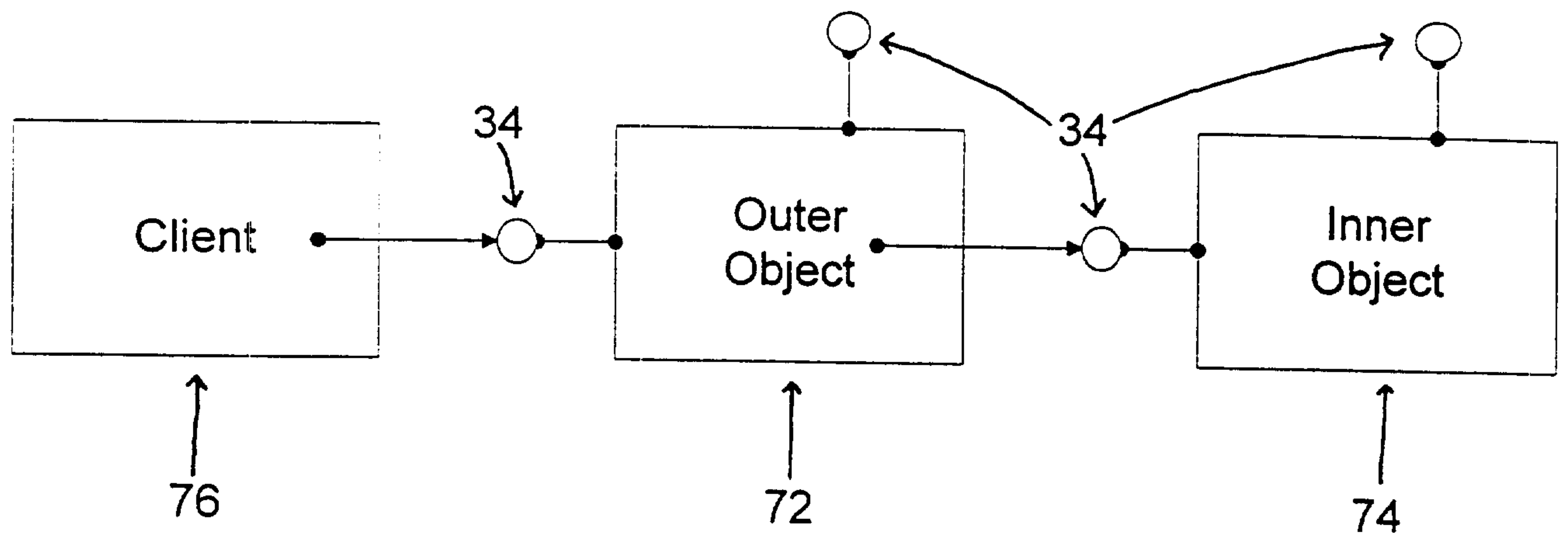




FIG. 4



Revising an object through containment.

FIG. 5

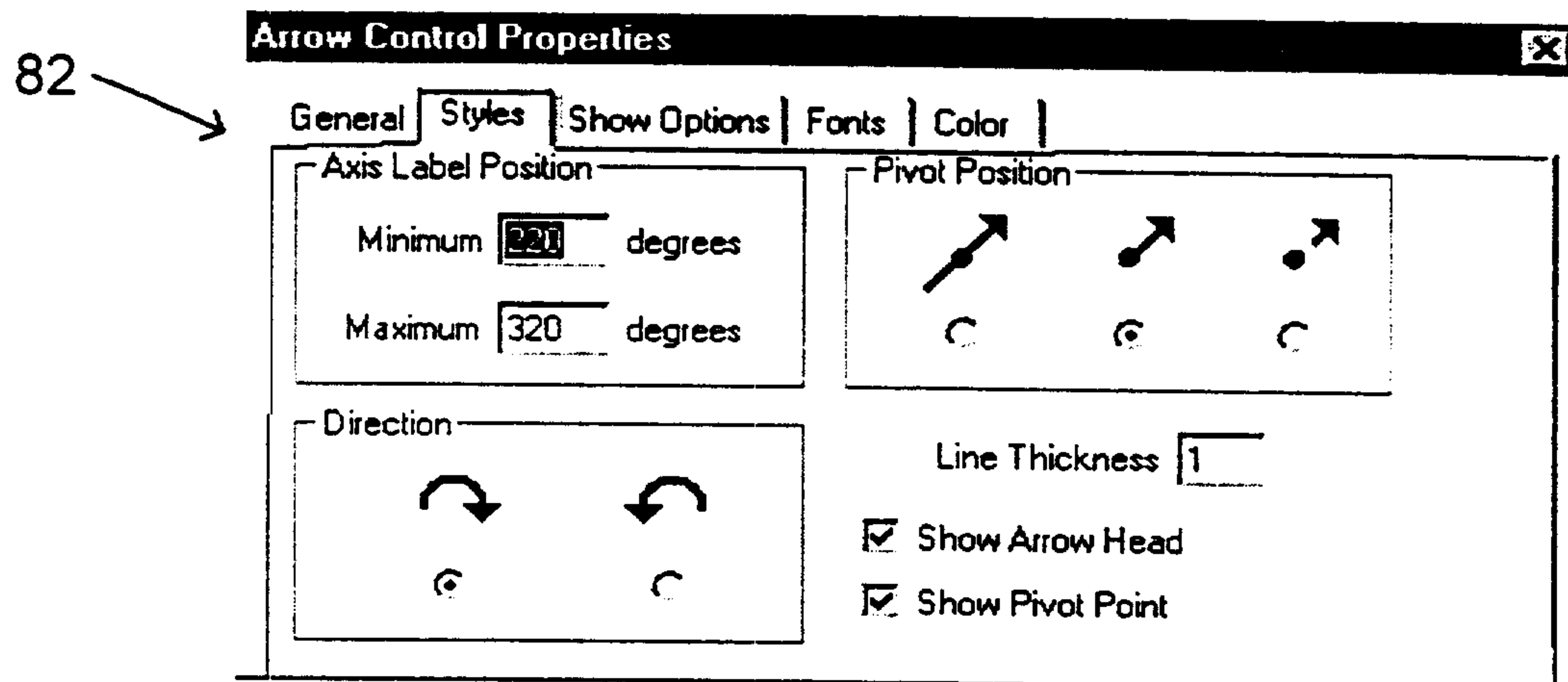


FIG. 6

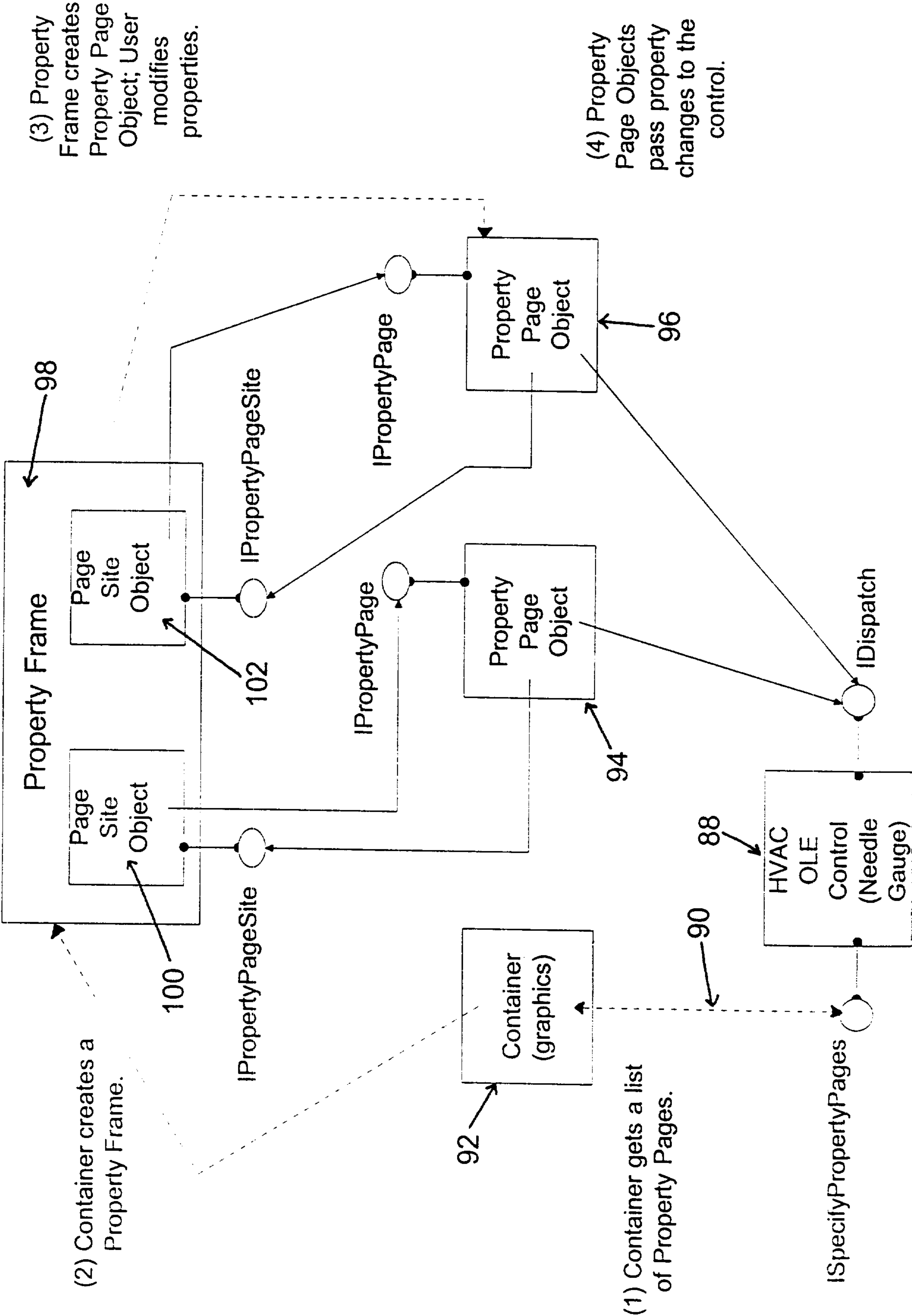




FIG. 7

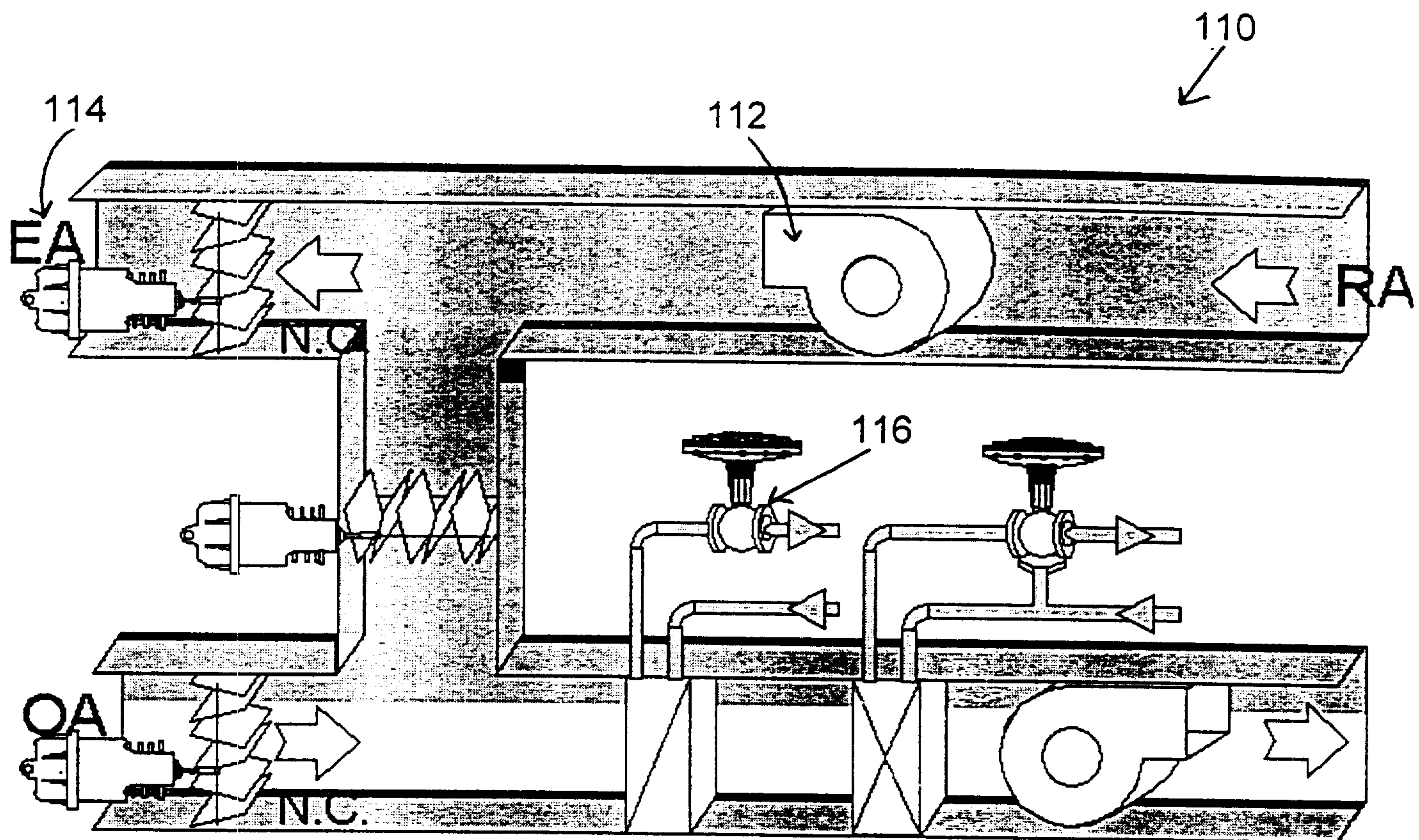


FIG. 8

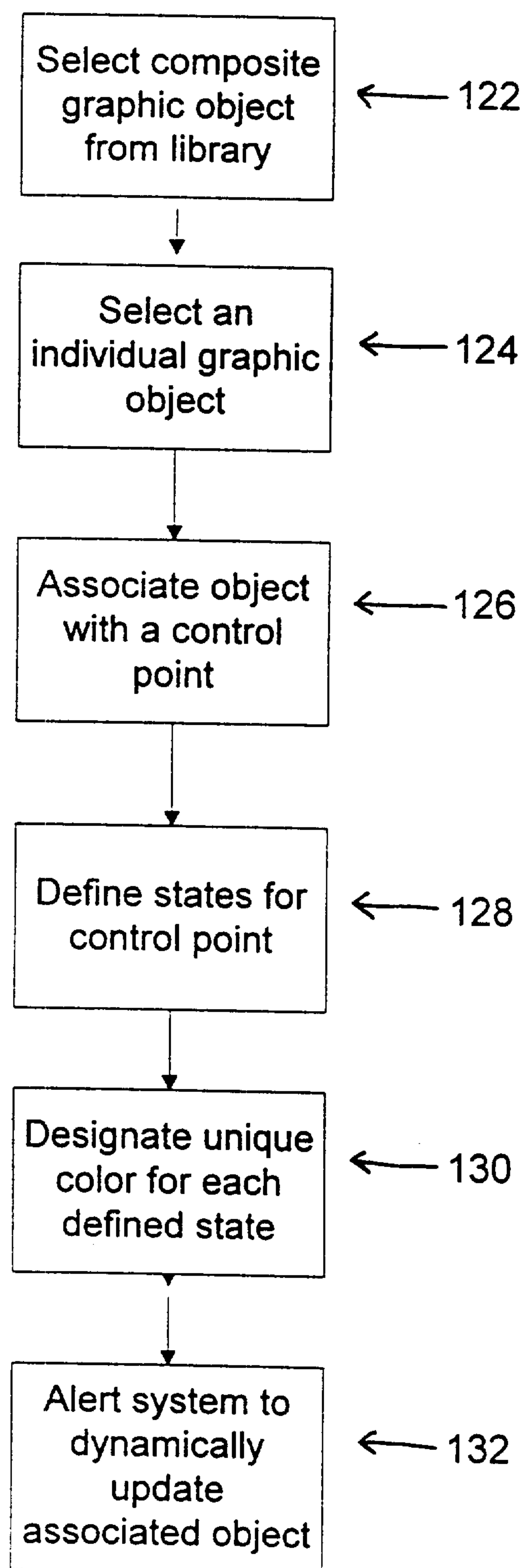


FIG. 9

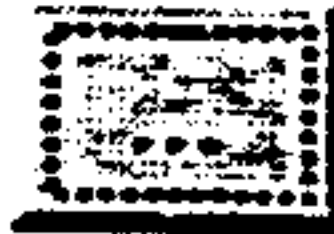

Point Name:	<input type="text"/>	
Configure Name To Display		
Sample:	<input type="text"/>	
<input checked="" type="radio"/> Display Entire Name		
<input type="radio"/> Display First Field of Name		
<input type="radio"/> Display Last Field of Name	Field Delimiter:	<input type="text"/> 



FIG. 10





Enhanced Alarm Setup					
Alarm Destinations	<div>001) Dest1</div> <div>(None)</div> <div>(None)</div> <div>(None)</div>				
Mode Point:	LAI001				
Mode Delay:	0	(min.)			
Level Delay:	0	(sec.)			
Differential:	0				
<input type="checkbox"/> Acknowledge Return to Normal					
Day	(Night)	(Special 2)	(Special 3)	(Special 4)	(Special 5)
<input checked="" type="checkbox"/> Alarm Mode Enabled		Set Point: <div>             Name: <input type="text"/> </div> <div>             Value: <input checked="" type="radio"/> 50           </div>			
Offset	Priority	Extra Destination	Enhanced Alarm Message		
1) 10	PRI1	(None)	(None) 		
2) 20	PRI2	(None)	(None) 		
3) 30	PRI3	(None)	(None) 		
4) 40	PRI4	(None)	(None) 		

FIG. 11

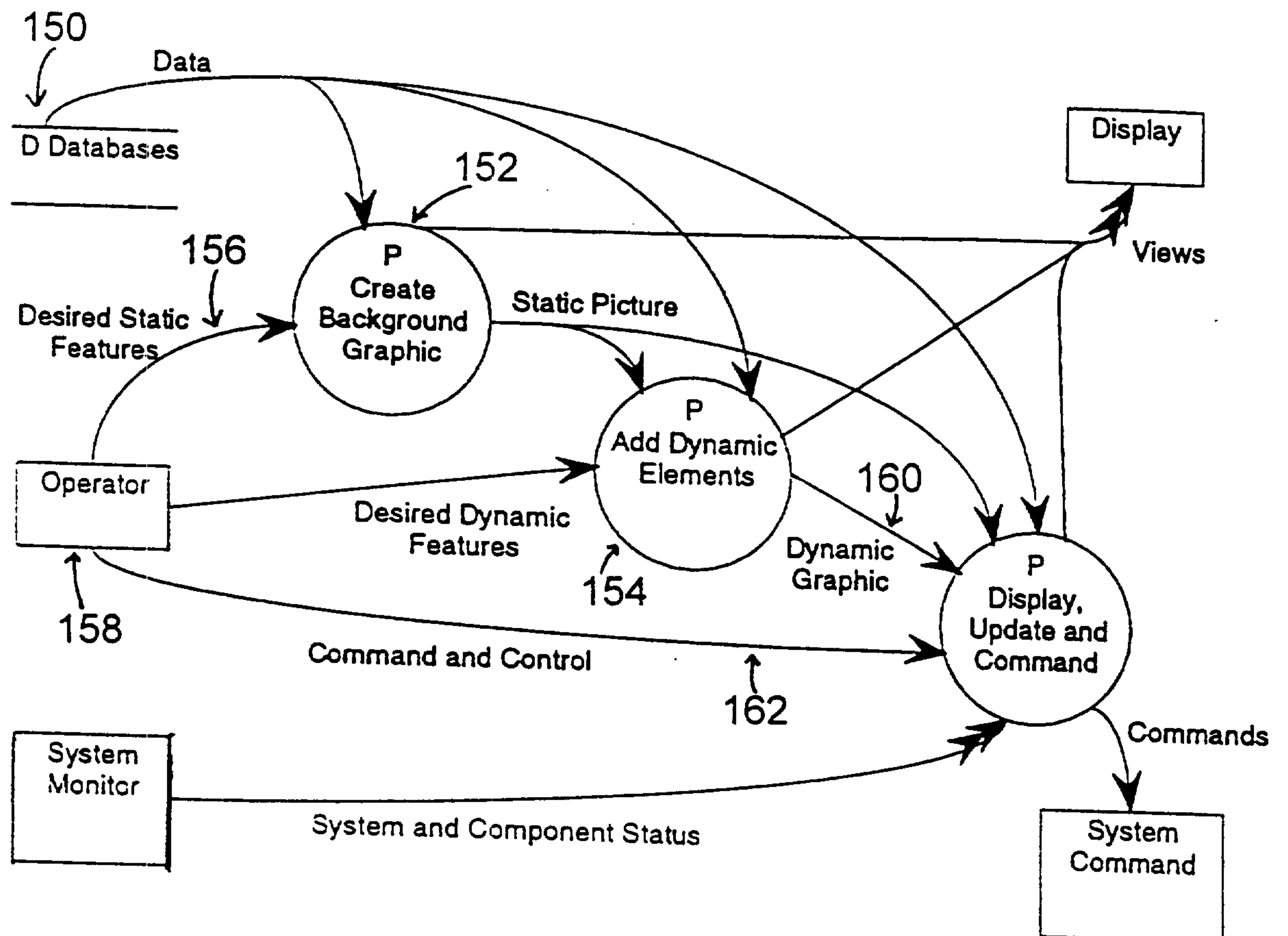


FIG. 12

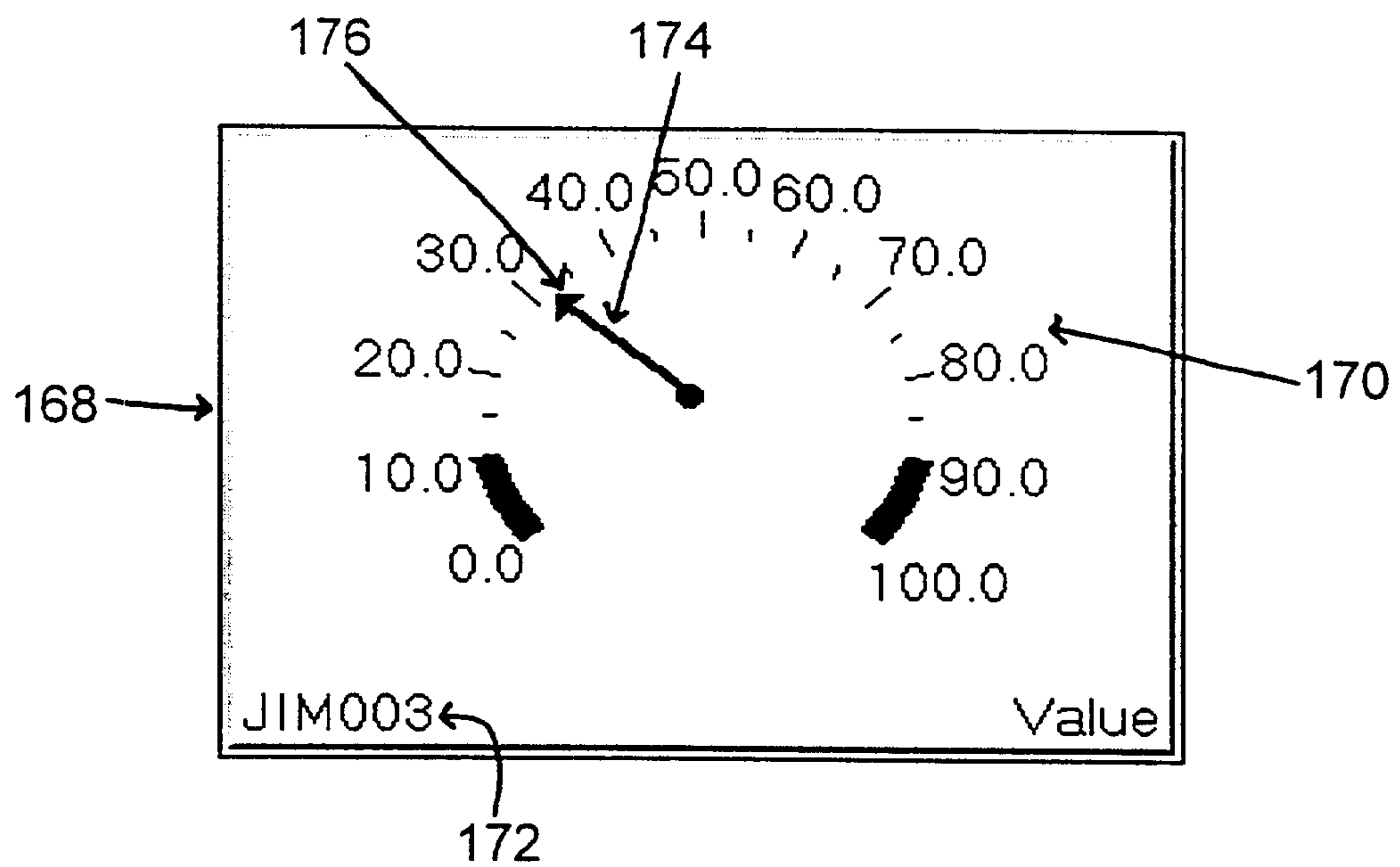


FIG. 13

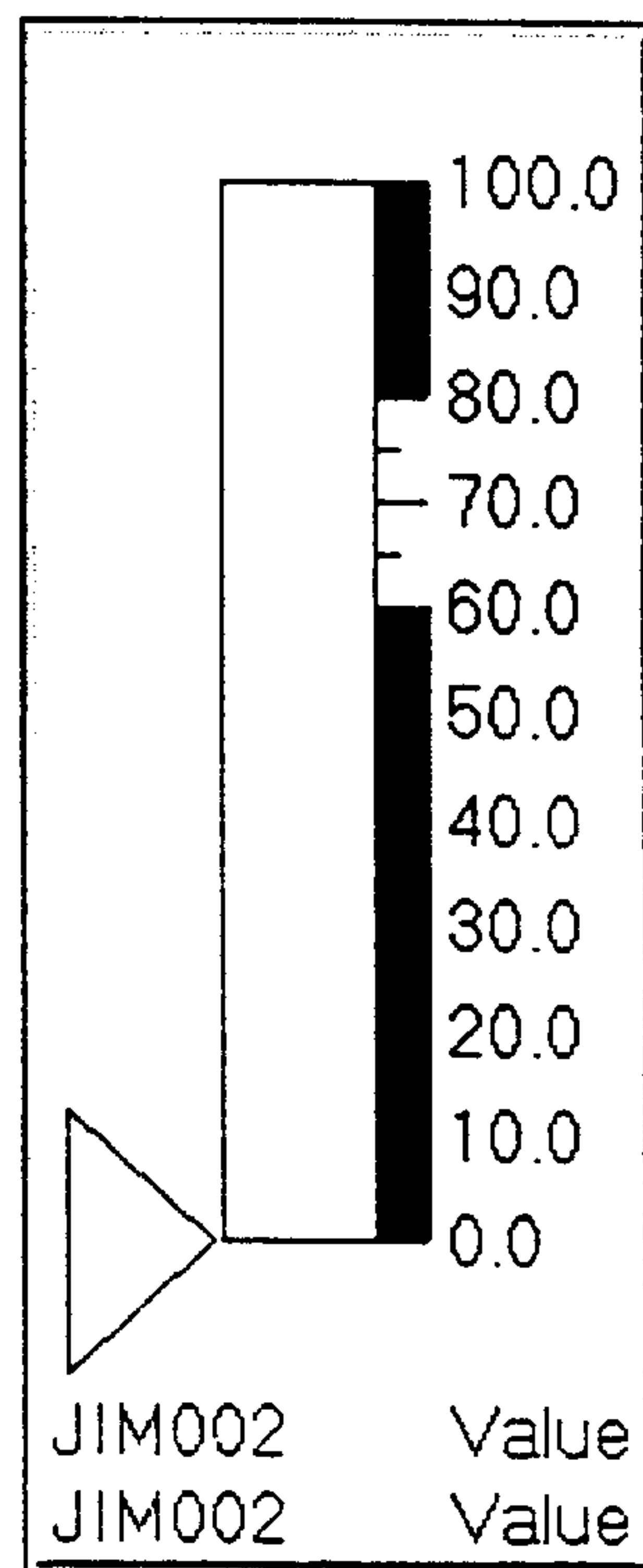




FIG. 14

JIM002U	JIM003
JAMES POINT	TEST POINT
Status	Status
Priority	Priority
Value	Value

FIG. 15

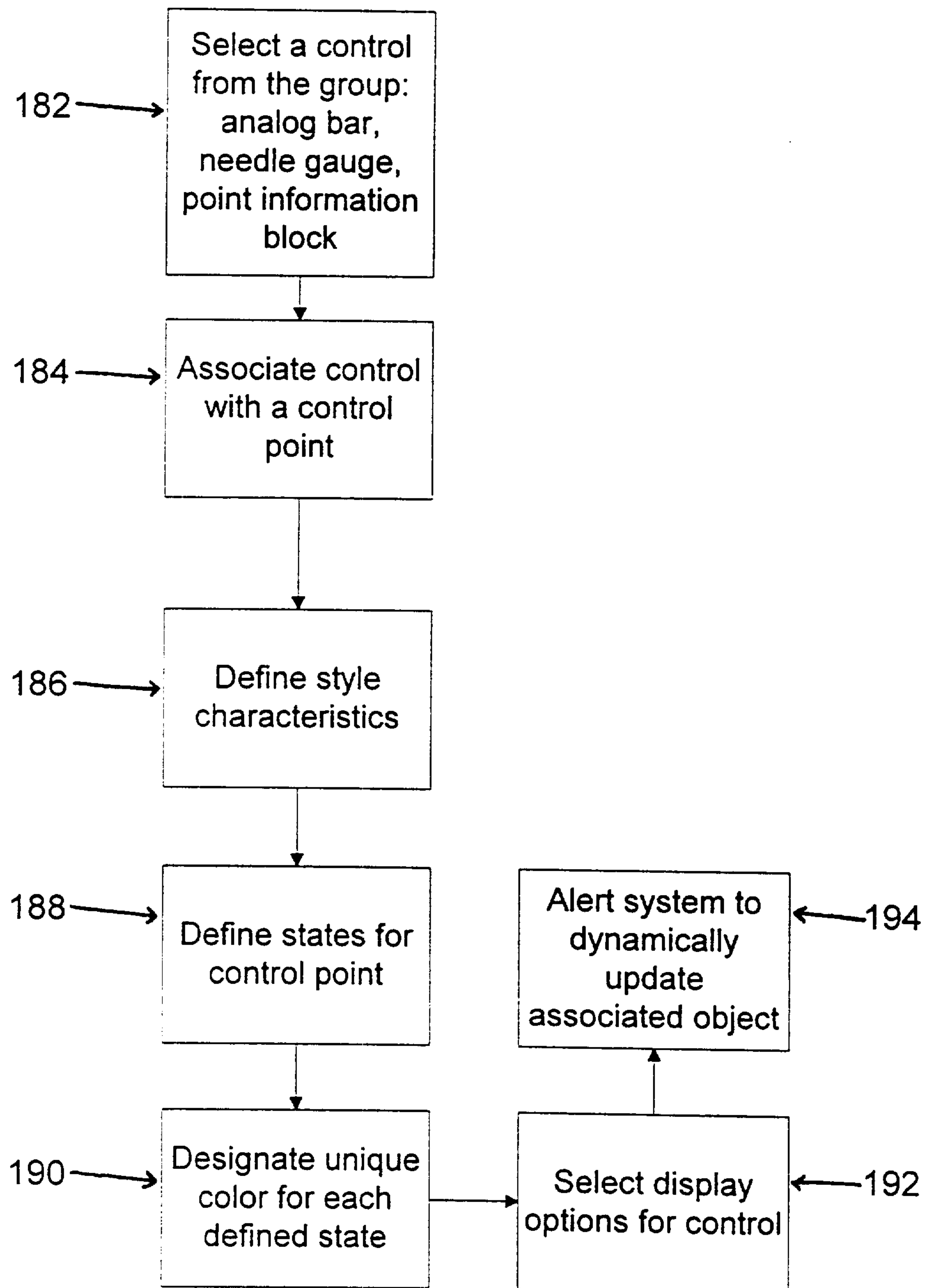


FIG. 16

# Analog Bar Control Properties

200

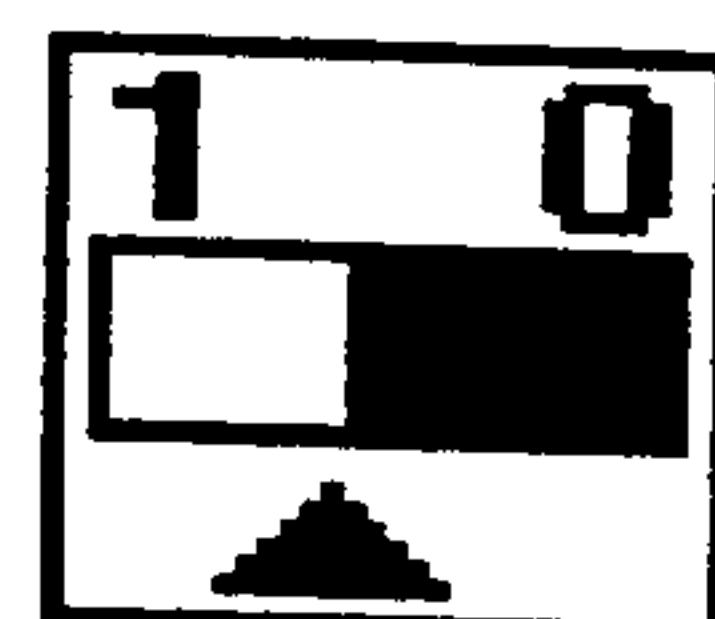
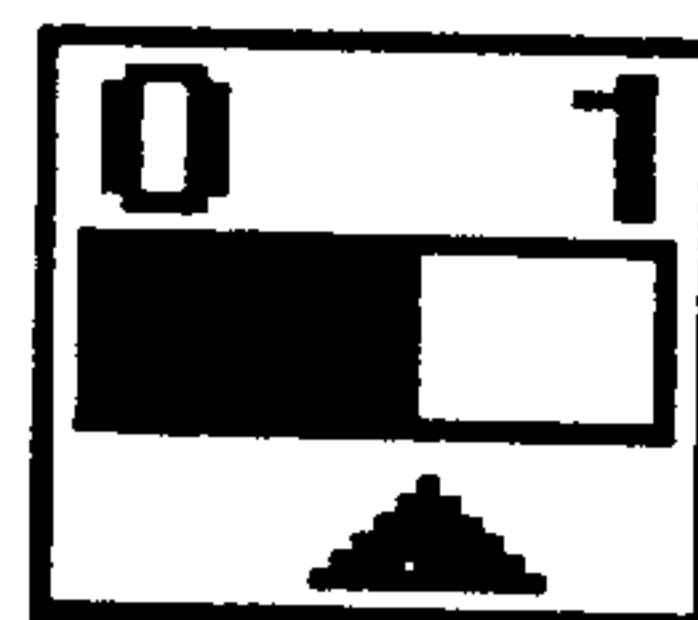
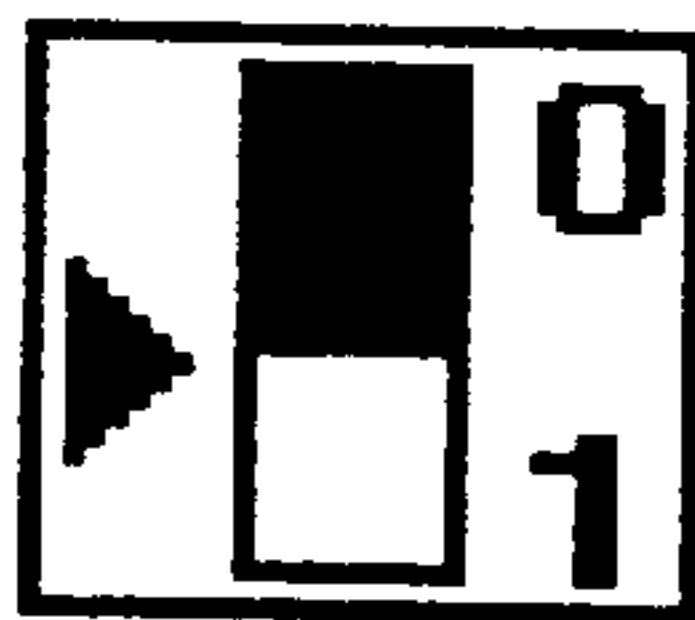
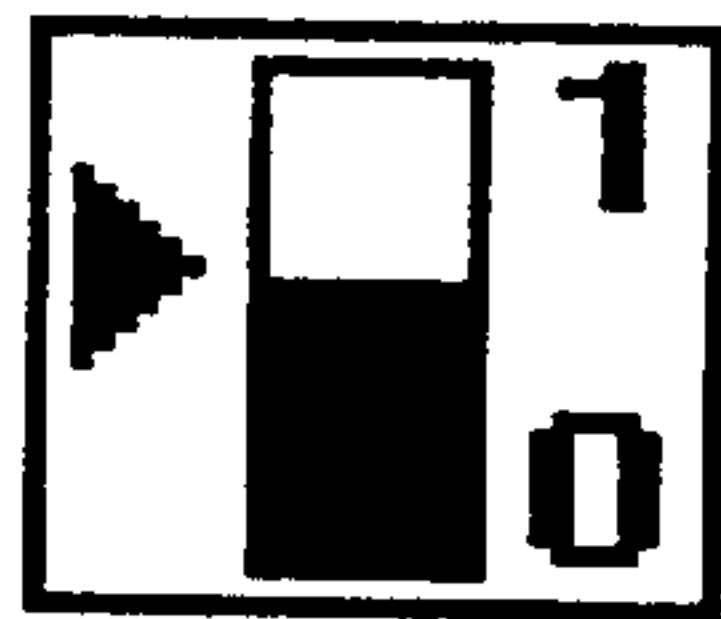


FIG. 17

210

Fields

- ☒ Display Name
- ☒ Descriptor
- ☒ Status
- ☒ Priority
- ☒ Current Value    ☒ Units
- ☐ Totalized Value

Display Style

☒ Multiple Lines    ☐ Single Line

State Color

☒ Use For Background Rectangle

☐ Use For Font Color

☐ Use Reverse Video Instead

☒ Border

