

Door Objects for Access Control

[History](#)

[Economic Perspective](#)

[Flexibility of Door Objects](#)

[Door Objects as the Solution to Ease of Programming](#)

[Implementation of Door Objects in an Access Control System](#)

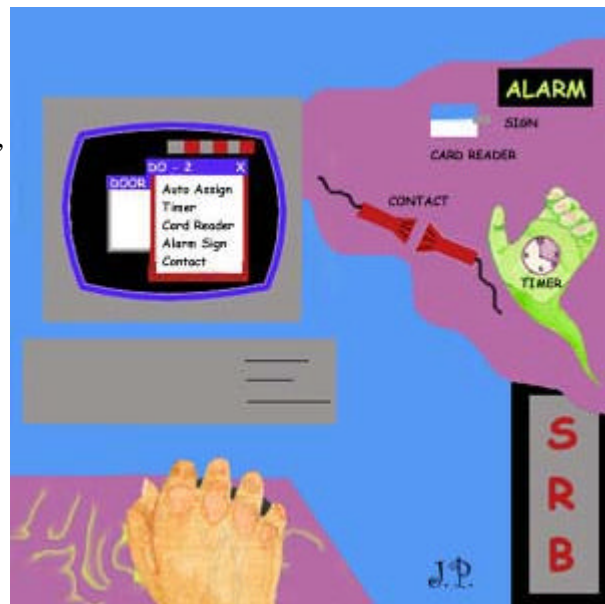
[Benefits of Door Objects](#)

[Chart for Door Objects in Access Control Systems](#)

It happened again yesterday. We were talking about a new project. The manufacturer that we wanted to use did not have, as part of their standard offering, a feature of door control that we would need. The site has several existing motorized revolving doors that are access controlled. The clean antipassback operation of these doors is critical to the success of the access control design. How should we proceed?

A concept has been forming for several years now that will allow greater flexibility for systems, simplicity of the setup programming, and a more realistic approach to the financial aspects of modifying the microcode that makes an access control door function. One size does **not** fit all in terms of the doors that are programmed into an access control system. The manufacturers continue to present reader or door screens with lots of options, switches, and relationships, and still we often need more to satisfy a clean installation. Because there were many different styles of airport jetway doors implemented during the big FAR107.14 days, many of the manufacturers created one or more separate subprograms called jetway doors. That is an example of a door object. Why not design the systems from the ground up with door objects?

The access control industry is not properly positioned to meet the customer needs for door operations. Even with some of the largest manufacturers, we have waited years for them to include a function that will allow us to lower the false alarm count from specific door applications. This is not because these manufacturers don't see the use or the need for the requests. It is because every piece of software that is written into the systems is a business decision that must be weighed against other priorities. And, it is not easy to change the "one size fits all" code for a single case without affecting all of the other applications that are already installed.



This article presents a future concept, not a fait accompli. Some manufacturers are close to having door objects. Let's get the access control industry out of the microcode business, at least in terms of how a door operates. It is clear that the industry does not make a profit on that portion of their enterprise, a fact that is probably the driving factor in the intransigence of door operation changes. Most of the industry will be designing with some form of door objects soon, simply because the economic case is so strong. Implementation of this concept will allow for almost infinite flexibility while not being constrained by legacy code and the "one size fits all" constraints. To put all of this in perspective, let's look at a short history of the industry.

HISTORY

The first versions of access control used early CPUs with several readers connected to multiplex communication boxes. As systems grew, the communications strategies became more complex, unwieldy, and slow. The industry moved to distributed processors that could make the access decisions promptly and subsequently transmit the history of the transactions back to the central systems. These distributed processors were set up to store card numbers, access criteria for the doors that are served by them, and some number of historical transactions. The history was added for the case where the communication line is lost to the central CPU (host computer) or if the system is using dial-up communication. Since all of the manufacturers call their distributed processors by different names, I will use the generic term Smart Remote Box (SRB) for these distributed processors.

Along with the complexity of the SRBs came the full interaction of access control systems with alarms for the access controlled doors. An alarm contact on a card-in/free-exit door must be shunted for valid transactions. When entering, a valid card read will shunt the alarm. When exiting, various devices that will automatically detect a person that was exiting are used. These devices are typically called a REX, which stands for Request to Exit. The process of accurately determining when a door is secure, in transition, held open, or in a forced open state continues to be a challenge today.

Attempts at false alarm reduction continue to be made by the users of large systems. Many large systems today have more false door forced alarms and too many door held alarms to allow the central monitoring area to function appropriately. There are only so many calls to which a given set of rovers can respond. And it is useless to spend the dollars for a rover to respond to a false signal from the system. There is a great economic need to clean up this part of the access control industry. There are enough differences in the various doors and hardware that are served by any manufacturer's systems that all of the possible scenarios of operation have not been fully considered in the design. Thus, there is a need for operational change. Getting the manufacturers to make these changes is difficult because it is very expensive to do safely and correctly. The ramifications are great.

The microcode structure that defines the SRB actions at a particular door have historically been stored in Programmable Read Only Memory (PROM). To change the

code in a PROM, the chip must be replaced, thus requiring physically visiting each SRB. In the last few years, FLASH memory has become available which allows the microcode program in the SRB to be downloaded from the host without the need to physically visit each SRB. With flash memory, the process of changing the operation of an SRB is much more efficient. Flash memory has made it possible to accomplish a door object based microcode download from the host computer.

ECONOMIC PERSPECTIVE

Security is big business today. However, the industry has some inherent economic challenges. Any medium sized commercial software company will sell thousands of one program, and a large software company will sell millions of copies. Yet, even the largest access control companies sell only hundreds, or fewer, of any one large system software product. Large systems can be considered to be those with greater than 256 readers. These programs are complex and require many lines of code, perhaps as many as 500,000 lines.

Programmers can only produce so many lines of code per year, yet each line of code costs the same to both the large software company and access control company. Microsoft has said that NT is over 20 million lines of code. Therefore, the return on investment per line of code is hundreds to thousands of times greater for the commercial software company than for the access control company. The difference is made up by all of the hardware sales in the security world. The fact remains that **changing the software to fix a slight anomaly in a timer for a door cannot easily be justified on a "return on investment" basis.** Money is lost on every change, and changes are made only because of the long term goal of having a good system and good customer relations.

The biggest single cost when changing SRB microcode is the engineering and testing to make sure that the change will not adversely affect the installed base of doors. I call this the legacy code cost. The door objects concept eliminates this cost because a newly created door object would only affect the doors to which it is applied.

FLEXIBILITY OF DOOR OBJECTS

One solution to the high cost of microcode changes is to utilize an "engine" that has field adaptable code. The process control and Supervisory Control and Data Acquisition (SCADA) industries have developed around this format, and the building management industry is generally heading in this direction. A relatively generic product "engine" would be provided to the installing contractor. Microsoft Operating Systems:

Libraries of functionality would also be provided to the contractor to facilitate the implementation work. There would be no need to try to make one solution fit all cases if the generic engine allows the installing contractor to modify the microcode to fit the customer's physical and operational needs. This is the basis for a "door object" in the access control industry.

Systems are typically installed with sets of similar doors. If, on a site, there are several buildings, typically all of the lobbies will be configured in the same way. The same is true of all of the access controlled perimeter non-lobby doors and other groups of portal types for the buildings. Included in the list are alarmed doors that may have local sounders or other attributes. The timers will be different for each type of door, but the initial default time may logically be the same. Even with all the similarity, there is still a need to be able to uniquely adjust virtually every timer in the system. One example is the case where one door that is functionally similar to another may architecturally need to have the reader placed farther from the door. In this case, the unlocking should be set to a longer time. Each set of similar doors will be a door object. The timers within the object will have a default setting, and each individual door will be able to either use the default or be adjusted to suit the unique conditions.

Without making unrecognizable changes to the access control products that exist today, it would be possible to format the systems with door objects. Initial libraries can be shipped with the systems. These could include:

- a simple card-in/free-exit door with a door contact and a REX
- a card-in/free-exit door as above but with the addition of a "please close the door" pre-held alarm
- a card-in/free-exit door as above with an "DOOR ALARMED" light and reader lockout function for night operation
- a card-in/card-out door with a door contact

The list will be quite long. The ability to create new objects to add to the initial set will be central to the success of this process. Possibly, some of the library of objects could be sold at additional cost. A jetway door and a mantrap are two specialized objects that come to mind.

DOOR OBJECTS AS THE SOLUTION TO EASE OF PROGRAMMING

One of the industry's difficulties today is the complexity of the user programming of the doors. The application screen for a reader or door has become very complex to accommodate all of the variations that have been programmed into the systems. In contrast, the screen for a door object can be as simple as the door that it represents. To add a door, the operator simply will pick the object that is appropriate to the physical and operational need of the door. This object will then be presented on the screen with a choice to self configure or to allow the operator to assign the actual point addresses for each reader, input, and output. The object's default times will be used for each of the timer values, while also allowing each of these to be user changeable. This process will greatly simplify the user programming of the system.

When a condition is found where there is no door object that meets the needs of the operator, a new object will be able to be created. Depending on the implementation, this

may be a relatively simple process or one that is complex requiring a special training class. The installing contractor or the very sophisticated end user would be typical of a person authorized to create a new object. There may also be some hardware and/or software costs. If all of the door object creation code can be integrated into the access control application, then the capability could be locked out of the system without the use of an appropriate password. The name of the person creating a new door object could be attached to the object for control purposes. Scratch creation of a new door object is needed, as well as the ability to take an existing object and do a "save as" to a new name and modify the former object with new functionality. It would be wise to lock out the ability to change the initial library, other than for the timer default settings.

The logic modules for door objects in an SRB will probably need to be compiled. This will be true of any design that uses compiled door objects instead of a somewhat less flexible table driven approach. To handle the compilation, one approach is for the manufacturer to provide a WEB site, with appropriate controls that would allow the same set of sophisticated installing contractors and end users to log into the site. These authorized persons would be the ones to create the object at the manufacturer, get the manufacturer's compiler to compile the object, and then file transfer the finished code back to them. This approach would save potentially ten to fifteen thousand dollars per group that would be able to create objects.

The main concept of the door object is that the manufacturer would be creating a standard product that does not require modification by the manufacturer to meet all of the physical and operational conditions of the site. Such needs are constantly changing.

The sphere of influence of an SRB should be the same as that of a door object. This seems to be the logical boundary for an object based on factors of complexity and cost. Inputs and outputs could be grouped to create a mantrap or turnstile controller, or a complex set of alarm and output functions. These functions are basically that of a programmable logic controller or PLC.

IMPLEMENTATION OF DOOR OBJECTS IN AN ACCESS CONTROL SYSTEM

- Create and supply an initial set of door objects.
- Each object should include the readers, inputs, and outputs that need to be assigned.
- Each timer within the object should be set to an adjustable default, with each timer adjustable on a per door basis.
- Each object should include all of the linking logic to tie the parts together.
- An operator screen must be available within the main ACS application for each object. The generation of this screen is necessary when new objects are created.

Components that can be part of the object are:

1. Readers

1. Inputs
2. Outputs
3. Keypads
4. Timers
5. Default times
6. Logic design
7. Simple English operational description
8. Name (renamable)

The components would include the logic microcode, the design presented in editable form at the head end of the system, and an implementation screen for the end user.

To add a door, the operator would pick an object from a list. The object could be presented in graphic as well as text form. Once a choice is made, the implementation screen would pop up on the monitor showing exclusively those items that apply to this type of object. All variables would show the default, with the ability to modify any of the variables. The operator could chose "auto-address" or assign the points. If there are not sufficient resources in the SRB that is chosen for the object, the system would pop up an edit that would prompt the operator to either use a different SRB with sufficient resources or add specific resources to the SRB. The operator should then be able to add the resources in software and continue on with the process. This would allow the system to be configured without any of the field hardware installed. Once configured, the hardware list could be printed out for physical installation. When the appropriate microcode information has been entered and accepted by the system, the microcode would be sent to the appropriate SRBs.

The system should be able to create several types of printouts that reflect the information that is in the system database relative to door objects. One of these would be a printout that lists all door objects with the associated points and timers for each. Obviously, other reports, such as by SRB and board within the SRB also would be critical to using and maintaining the system.

BENEFITS OF DOOR OBJECTS

There will be cost savings to the ACS manufacturer, the installing contractor, and the end user. This is a no-lose situation. The volume of software sold by large access control manufacturers is small compared to large software companies. Obviously, it is the unique functionality of Access Control Systems that allows the prices to be what they are. The ultimate cost to the manufacturer for a more flexible system would be less, based on the lack of needing to respond to the many requests for microcode fixes. The entire chain of manufacturer, installer, and user will benefit.

The time required to solve a problem will be significantly reduced. From several large manufacturers, the delay for changes in microcode has been, in some cases, greater than a year. Temporary work-arounds have been used. But, this type of delay is not comfortable for any of the parties. In other cases, the delays have been only hours or days, depending

on the urgency and the profile of the job. Unfortunately, these are awkward criteria for the smaller end user who just wants the system to operate as "advertised." Since each door object is an entity, the legacy of one type of door has no relationship to a new object. The programmer of a new door object can look forward without the responsibility and cost to look back.

Long term flexibility will be greatly enhanced because each SRB will be capable of accomplishing the types of functions that are common in PLCs. It has always been awkward and costly to use PLCs to run revolving doors, mantraps, optical turnstiles, sally ports, various styles of elevator control, and the other special security portals that the industry can dream up. By allowing enough flexibility in the object design limits, each of these could be achieved within the structure of one SRB.

DOOR OBJECTS IN ACCESS CONTROL SYSTEMS

FUNCTION	DO THIS	HOW IT WORKS
<ul style="list-style-type: none"> • To program a new door. 	<ul style="list-style-type: none"> • Pick an appropriate object from the library. • The object will hold all of the logic relationships and the default timers for this type of door. 	<ul style="list-style-type: none"> • The setup screen reflects all addresses for the door, card reader, door contact, REX, lock, etc. and only those addresses. • The user will have the choice to auto-assign or to manually assign addresses. • The user can modify all timers.
<ul style="list-style-type: none"> • To change how a door object works or to create a new object. 	<ul style="list-style-type: none"> • The manufacturer, or a sophisticated dealer or end user can take an object that is close and "save-as" to a new name. • Then modify this new file to add or delete the attributes that are required. 	<ul style="list-style-type: none"> • The new door object stands on its own. • It does not need to be legacy tested against all prior conditions, since it is designed for a new condition. • It has no relationship to any other doors that are already installed.
<ul style="list-style-type: none"> • Special objects can be built. 	<ul style="list-style-type: none"> • Jetway Doors • Revolving Doors • Alarm to Output Relationships. • Virtually all points in a system can be 	<ul style="list-style-type: none"> • Special objects can reflect what they are. • They do not need to fit any standard access control design. • The sphere of influence of a Smart Remote Box is the

	programmed in this way.	logical scope of any one object.
--	-------------------------	----------------------------------

[BACK TO TOP](#)