Security Patterns for Physical Access Control Systems

Eduardo B. Fernandez, Jose Ballesteros, Ana C. Desouza-Doucet,
and Maria M. Larrondo-Petrie

Department of Computer Science and Engineering
Florida Atlantic University
Boca Raton, Florida 33431, USA
ed@cse.fau.edu, jballes2@fau.edu, adoucet@bluefrogsolutions.com,
maria@cse.fau.edu

Abstract. Physical security has received increased attention after 9/11. However, access control to physical units has not been explored much. On the other hand, there is a rich literature on access control to information. These two areas appear converging as seen by recent products and studies. However, use of different notations and implementation details make this convergence harder. We need to try to take this convergence at a more abstract level first. Although introduced about 10 years ago, security patterns have recently become accepted by industry and two books on this topic have appeared recently. Security patterns for information access control have appeared but now we extend this concept to access for physical units. The unification of information and physical access control is just beginning but the strong requirements of infrastructure protection will make this convergence to happen rapidly. Examining existing systems, industry standards and government regulations, we describe, in the form of patterns, the core set of features a physical access control system should have. The paper illustrates the structure and use of these patterns.

Keywords: access control, intelligent buildings, physical access control, security, software patterns.

1 Introduction

Homeland security has brought an added interest in control of access to buildings and other physical structures. The need to protect assets in buildings and to control access to restricted areas such as airports, naval ports, government agencies, and nuclear plants to name a few, created a great business opportunity for the physical access control industry and a good amount of interest in the research community. One of the results of this interest was the recognition that access control to information and access control to physical locations have many common aspects. The most basic model of access control uses a tuple (s,o,t), subject, object, access type. If we interpret s as a person (instead of an acting executing

S. Barker, G.-J. Ahn (Eds.): Data and Applications Security 2007, LNCS 4602, pp. 259–274, 2007.

entity), o as a physical structure (instead of a computational resource), and t as a physical access type (instead of resource access), we can make an analogy where we can apply known results or approaches from information access control. The unification of information and physical access control is just beginning but the strong requirements for infrastructure protection will make this convergence to happen rapidly. Another issue is the fact that there are standard network protocols for building automation, e.g. BACnet [1], which are totally different of the protocols used for manufacturing automation, e.g. DNP3 [2]. Both types of protocols define security standards, which means that a building intended for manufacturing would have two sets of incompatible security standards. We need some way to abstract the security requirements of the complete system without regard to specific standard details.

One way to achieve this unification is using a conceptual abstraction for the definition of security requirements; we consider here the use of analysis and security patterns for this purpose. A pattern is an encapsulated solution to a recurrent problem in a given context [3][4]. In particular, a security pattern defines a solution to a security problem [5]. In general, the use of patterns has been increasing in industry because of their potential to improve software quality. Although introduced about 10 years ago, security patterns have only recently become accepted by industry and Microsoft, IBM, and Sun have web pages on this topic. Also, two books have appeared recently [5][6]. We have presented several security patterns for access control to information, e.g. [7][8], and now we extend this concept to the access of physical units. Standards and products that deal with physical units use a set of common concepts that may appear different due to a different notation; patterns make this commonality apparent. Examining existing systems, industry standards, and government regulations, we describe, in the form of patterns, the relationship and definition of a core set of features a physical access control system should have. From these patterns, it is possible to define more specific patterns that can be used to build systems in a given protocol or to define new protocols. These patterns can also be combined to make up complex systems.

We present here patterns for access control in physical units. While we cannot be complete, we show three of them to illustrate their structure and possibilities:

- Alarm Monitoring. Defines a way to raise events in the system that might require special attention, like the tampering of a door.
- Relays. Defines the interactions with electronically controlled switches.
- Access Control to Physical Structures. Applies authentication and authorization (RBAC) to the control of access to physical units including alarm monitoring, relays, and time schedules that can control when things will happen.

The pattern diagram of Figure 1 shows how these patterns relate to each other and to related existing patterns. The patterns not explicitly described here include:

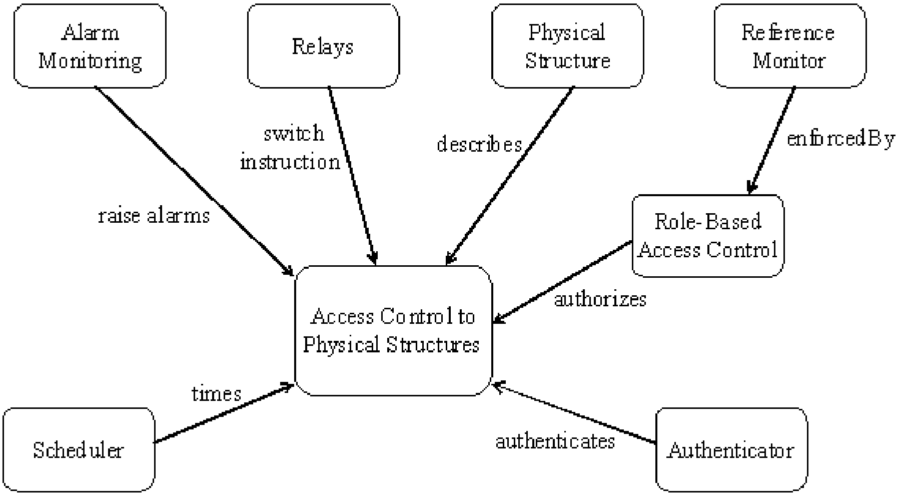

Fig. 1. Pattern diagram of physical access control patterns

- Physical Structure [9]. Defines the structure and use of physical sites such as buildings, parking lots, and similar, as well as their divisions and compartments.
- Scheduler. Provides timing information to control access.
- Role-Based Access Control [7]. Describes the standard RBAC model used here to describe authorization to access a physical unit.
- Reference Monitor [5]. Enforces authorizations when a process requests access to an object.
- Authenticator [10]. Verifies that a subject is who it says it is.

Alarm Monitoring, Relays, Scheduler, and Physical Structure are composed to form the Access Control to Physical Structures pattern. RBAC defines the type of authorization rules used in the system, while the reference monitor indicates an abstract pattern representing enforcement mechanisms. Authenticator is an abstract pattern defining the requirements for authentication. We present these patterns following the POSA template [3]. This template intends to provide enough detail to be a guideline for a designer building a system that requires this pattern. The set of patterns can also be used to define precise requirements and to evaluate existing systems. The solutions are described by UML diagrams, which although not strictly formal, are precise and unambiguous (UML has a well-defined syntactic metamodel). Because each pattern must be reusable on its own, there is some amount of redundancy between patterns. Each pattern is relatively simple but it can be combined with others to build complex systems.

Section 2 describes some background. Section 3 presents first two patterns that complement physical access (Alarm Monitoring and Relays). These patterns are then combined with other patterns to define a composite pattern that describes the necessary elements of a physical access control system. Section 4 includes a short discussion and comparison to other approaches. We end with some conclusions.

2 Background

Physical access control systems are widely used today and they can be implemented with a wide range of technologies. The basic idea is that something controls access to someplace; it could as simple as a key lock and as complex as a face recognition device. Moreover, there are ways to detect and inform when someone violates the access rules or tries to force the system. This simple definition leaves room for a great number of features and combinations in software and hardware that vary from product to product. To understand access control, we must understand the language of the industry. Terms like the ones discussed in the patterns presented here are commonly used when discussing access control systems. Other terms include:

- Access control panels. Serves as an interface to the readers and door locks. Wiring in a network interconnects most of these panels.
- Electric door locks. Keep the doors locked and secure, and release the door when a valid credential is used.
- Shunting of alarm devices. Means to bypass or ignore an alarm for a specified period of time.
- Anti-pass back. Used to prevent tailgating (when one user enters with a valid credential, and several people enter without using theirs).

As indicated earlier, a pattern describes a solution to a specific problem in a given context. Patterns are abstractions of real systems, emphasizing best practices and fundamental features, we found these patterns by analyzing real systems or standards. A security pattern describes an abstraction of a security mechanism able to avoid or mitigate some threats [5][6]. In addition to the two mentioned books about security patterns, a variety of security patterns have appeared in the literature [11]. It is common practice to describe the solution encapsulated in a pattern using UML (Unified Modeling Language) diagrams. UML is a standard for software development and its models are graphic and intuitive and can be conveniently converted into code. In addition, as indicated earlier, UML is a semi-formal language that provides a good amount of precision. There have been attempts to further formalize patterns but their importance comes not from an ability to prove security properties of the system but because of the fact that they are known to be good practices, based on experience. In addition, since each pattern is rather simple, they can be used by practitioners; many software developers know UML but are not able to use formal methods.

Patterns are described using some type of template, consisting of a specific set of sections with predefined meanings. We use here the so-called POSA template [Bus96] which includes the following sections:

- A *Name*, that should be unique and precise.
- A thumbnail *summary* of the pattern (what problem does it solve?)
- An *example* of a situation where a solution is needed.
- The *context* where the pattern is valid or applicable.
- The general *problem* solved by the pattern.
- A *solution* section, describing the idea of the solution. This includes two subsections, a *Structure* section describing a class diagram of the solution, and a *Dynamics* section describing some typical use case sequences.
- The *Implementation* section provides hints in using the pattern. The example resolved shows how the pattern can solve the problem of the example presented earlier.
- The *Known Uses* section indicates some real uses of the pattern.
- The *consequences* indicate the advantages and disadvantages of using this solution.
- The section on *related patterns* enumerate patterns which solve similar problems or are complementary to this pattern.

3 Patterns

3.1 Alarm Monitoring

Defines a way to raise events in the system that might require special attention, such as someone tampering with a door.

Example. Building management wants to raise alarm events when someone opens a door without using the right credentials or when someone tries to use a card that was reported as lost.

Context. Physical environment with an access control system where we want to be able to raise filtered alarm events and we want to differentiate between alarms that are generated because of a physical violation of the system and alarms generated because of a system rule violation.

Problem. In an Access control system, there are two types of alarms, physical and logical. Many times we might want inform more than one subsystem that a change of state happened in order to generate different types of configurable actions that can vary. Physical alarm inputs are used to monitor various devices. These alarms can be shunted. Logical alarms are used to monitor various system rules. For example, a system may generate an alarm after three invalid tries to get access with the same credentials.

The following forces will affect any solution:

- There are two types of alarms, physical and logical.
- Alarms can be ignored.
- Logical alarms have two possible states, reset and alarm.

- Physical alarms have four possible states: reset, alarm, cut or short. The last two states are known as trouble states, caused by faulty wiring or tampering.
- We may need to know what alarms have been set and when they were reset.
- We need a way to inform interested parties about a change of a state of an alarm.

Solution. Have an alarm entity that describes the general concepts of an alarm and use generalization to separate logical alarms from physical alarms and their particular characteristics. Add information about the time the alarm occurred. Use the Observer Pattern [4] to advise any interested party of a change of state in an Alarm.

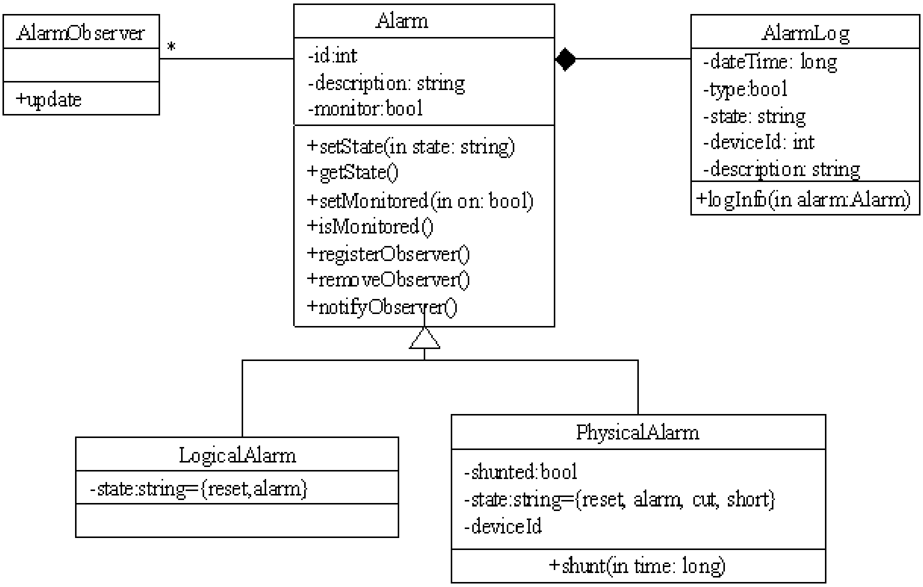

Fig. 2. Class Diagram for Alarm Monitoring

Structure Figure 2 shows a UML class diagram for the Alarm Monitoring pattern. Abstract class Alarm indicates a general class for any type of alarm. The PhysicalAlarm class and the LogicalAlarm class inherit the Alarm class. These classes allow for alarms to be controlled and monitored. By implementing the AlarmObserver interface any class (not shown here) interested on this alarm can be added to the list of observers for the alarm and will be advised when a change of state happens. Moreover, we log every alarm activity with a time stamp.

*Dynamics***.** Figure 3 shows the main success scenario for the use case "activate an alarm". The request to set the alarm active is sent by an external actor that detected the need to raise the alarm. The change of state is logged and only if the

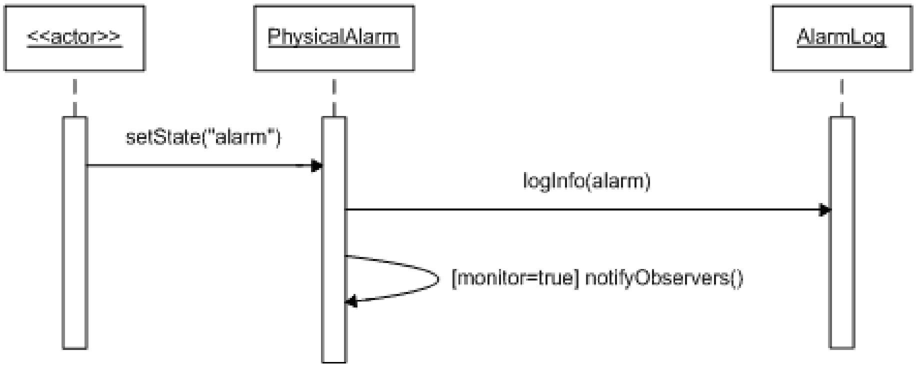

Fig. 3. Sequence Diagram for Activating an Alarm

alarm is being monitored interested parties are advised of the change, otherwise the alarm is ignored.

Implementation. There are many ways to create alarms in a physical access control system. For example, when a card reader is installed on a door, an alarm contact is usually installed as well. The alarm point is used to monitor whether the door was forced or held for too long after a valid access was granted. Logical Alarms can be generated for maximum tries with invalid credential, invalid credential and communication errors. The call to change the state of an alarm could in turn generate other actions when observers are notified, like displaying a message, activating a siren, etc.

Example Resolved. Every time someone opens a door without proper permission an alarm can be created and if a person uses a lost badge the system can generate a logical alarm.

Known Uses. Many commercial Access Control systems have the concept of logical and physical alarms that generate some actions when the states are changed.

Consequences. Advantages include: We can only pay attention to the alarms we are interested in.

– Every alarm change of state is logged, and interested parties are advised.
– This model provides the basic structure for supporting alarms in an access control system.
– We make a distinction between logical and physical alarms, which supports the creation of any alarm independently of the existing hardware.

A disadvantage is: The pattern may create overhead for systems that only care about logging the alarm.

Related Patterns. This pattern is based on the Observer Pattern [4]. The Access Control for Physical Structures [9] complements this pattern by adding the concept of Zones that control Alarms.

3.2 Relays

This pattern defines the interactions with electronically controlled switches.

Example. Building management wants to be able to open the main gate for visitors that do not have credentials. Moreover, doors should be opened when someone presents valid credentials.

Context. Physical environment with an access control system where we want to be able turn on/off devices; and lock/unlock doors.

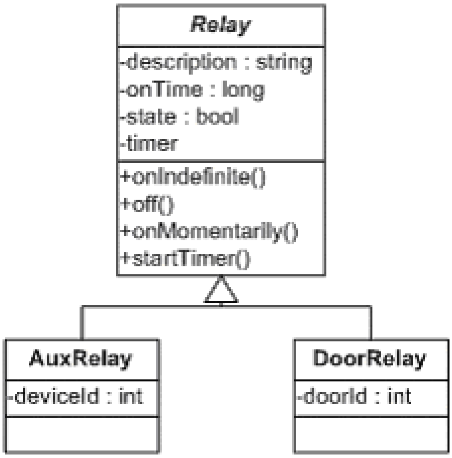

Fig. 4. Class Diagram for Relays

Problem. A relay is an electronically controlled switch. Similar to a light switch on the wall being used to turn on or off a light, a relay can be used to turn on or off other devices. Relays are used to activate the electric door lock, or to activate a variety of other items such as: bells, sirens, turn lights on and off, trip a digital dialer and many other uses. Typically, one relay is used to control the electric strike of doors. The others, usually called auxiliary relays, can be used as needed. When a relay is activated or deactivated, the device wired to it is turned on or off.

The following forces will affect the solution:

- We need to distinguish between door relays and auxiliary relays.
- Relays can be activated and deactivated.

- Relays can be activated indefinitely or for a defined period of time.
- Relays have two possible states: on and off.

Solution. Have a relay entity that describes the general concepts of a relay and use generalization to separate door relays from auxiliary relays and their particular characteristics.

*Structure***.** Figure 4 shows a UML class diagram for the Relays pattern. Abstract class Relay indicates a general class for any type of relay. The DoorRelay class and the AuxRelay class inherit the Relay class. These classes allow for relays to be controlled.

*Dynamics***.** The sequence diagram is trivial. The request to set the relay active is sent by an external actor. The relay is activated for the previously defined "on time." If the relay was already active the timer is restarted. Once the timer expires the relay is turned off.

Implementation Relays could be activated or deactivated by a variety of events. An alarm input, a valid credential, an egress button being pushed, or a time event, can all activate a relay.

Example Resolved. Doors that have relays defined can be opened when a valid credential is presented. Also the main gate can be assigned an auxiliary relay that can be activated at will.

Known Uses. Many commercial Access Control systems have the concept of relay management and distinction between door and aux relays.

Consequences. Advantages include:

- We can distinguish between auxiliary and door relays.
- We can activate relays for a predefined amount of time.
- This model provides the basic structure for supporting relays in an access control system.

A disadvantage is:

- We might want the same kind of functionality for devices that are not exactly door or aux relays.

Related Patterns. The Access Control for Physical Structures [9] complements this pattern by adding the concept of Zones that control Relays.

3.3 Access Control to Physical Structures

Applies authentication and authorization to the control of access to physical units including alarm monitoring, relays, and time schedules that can control when things will happen.

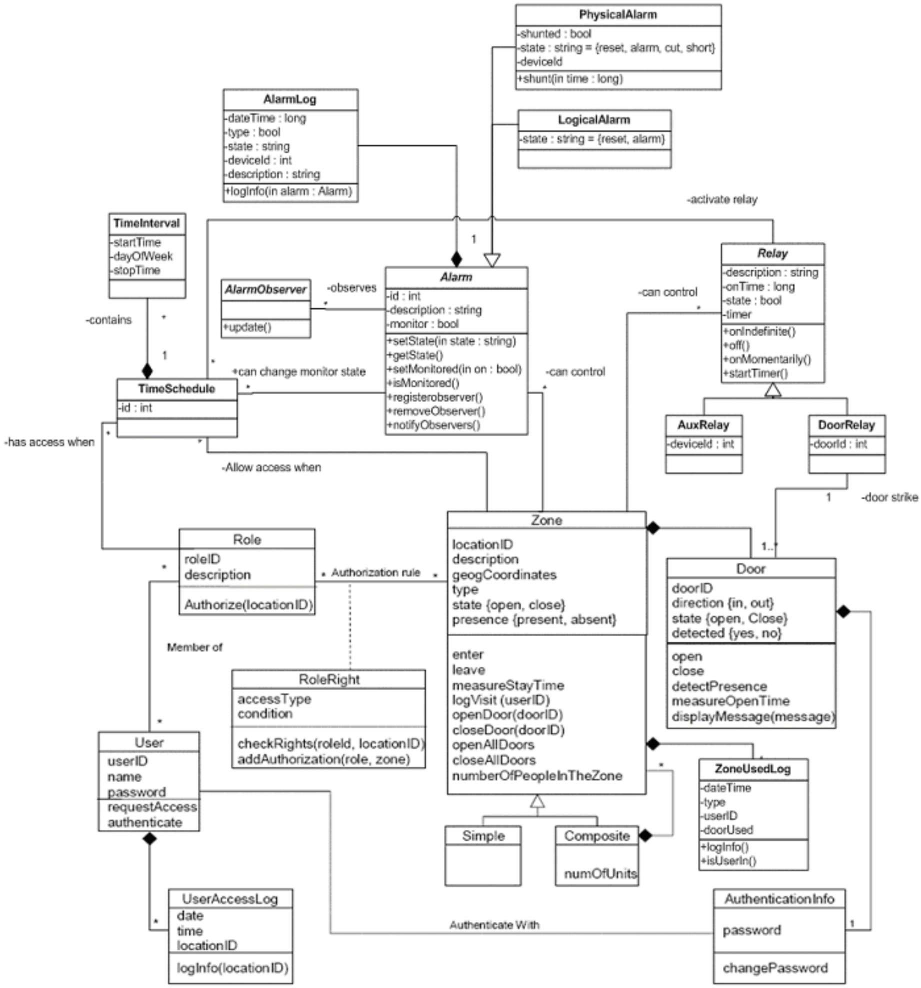

Fig. 5. Class Diagram for Access Control to Physical Structures

Example. Building management wants to put in place an access control system to control access to certain zones and to control who can access the zones. They need to deny all access to certain zones after 5pm. They want to generate alarms when someone tries to access a zone for which they do not have permission and start monitoring alarms for all the exterior doors at 8pm. Moreover, they want to turn on the main door light at 7pm.

Context. Physical environment with access control system where we need to control access and turn on/off devices based on time constrains.

Problem. We need a way to enforce business rules in an access control system that take effect at a given time and day of the week. For example, a user may have different access needs on different days, as in the following example: 08:30 - 17:00 Mon Wed Fri Standard Hours 08:30 - 19:00 Tue Thur Stays Late 2 Days a Week 08:00 - 12:00 half a day on Saturday

The following forces affect the solution:

– We need a way to automatically cancel access for everyone to some areas of a building at given time and day of the week. Some users might have access to some areas only during a time range of the day.
– We need to provide a way to automatically activate devices based on the time and day of the week.
– We need a way to represent a day of the week and time.
– This pattern expands the Access Control to Physical Structures Pattern [9]. Therefore all the forces presented in that pattern are present in this pattern as well.
– We need to restrict access to some users depending on the identity or other characteristics of the visitor.
– The boundaries of the unit of access must be clearly defined.
– There is a variety of users such as employees, contractors, repairpersons, etc., that require access to different parts of a building.
– Since some units will be normally closed, it is necessary to create policies and plans in case of an emergency, such as earthquake or fire.
– We may need to keep track of who entered each unit and when.

Solution. Define the structure of an access control system using an RBAC pattern [7]. Integrate the Alarms Monitoring and Relays patterns and introduce the concept of a time schedule to control when things can/must happen. Time Schedules have two uses; one is to control access times and the other is to configure automatic actions.

Structure. Figure 5 shows a UML class diagram for the Access Control to Physical Structures pattern. We can see how the Alarms Monitoring pattern can be integrated so that **Zones** can control **Alarms**. We also use the Relays pattern so that each door has its own **Door Relay** and the **Aux Relays** can be used to turn on/off devices. Zones can control Relays.

We introduce a **TimeSchedule** class that consists of a few time intervals. A **Time Interval** consists of a start time, a stop time, and selected day of the week. When the system clock activates a Time Schedule, it can automatically perform some actions. Relays can be turned on and off, alarm zones can be activated or deactivated, and doors can be unlocked. To control access times, roles are combined with Time Schedule to determine both where and when a user can gain access to zones and we also need to be able to assign a Time Schedule for the entire zone that applies to all users.

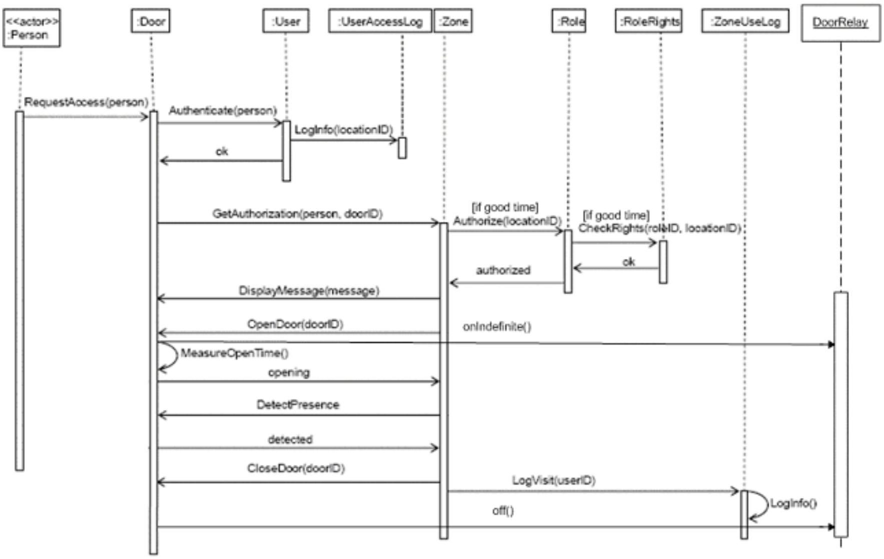

Fig. 6. Sequence Diagram for Entering a Zone

*Dynamics***.** Figure 6 shows a main success scenario for the use case "enter a zone". When the control is passed to the zone, the Zone and the Role will consider their Time Schedules before authorizing a person. Also when the Zone opens or closes a door, the Door calls its Relay to perform the action. Other use cases include: "zone access denied by zone time schedule", "role access denied by time schedule", and "activate alarm by role access denied"; they are not shown for conciseness.

Implementation. Access control systems use centralized processing, distributed processing, or hybrid arrangement. The system architecture should be taken into consideration when designing an access control system, since it can have a significant effect upon operation during a catastrophic system failure.

Centralized Processing. In computer dependent processing systems, all events are gathered by the field panels, and are then sent to the computer for processing. For example if a credential is presented at a door, the door sends the credentials to the central computer or processor. The computer checks the credential against its programming and determines if it is allowed through that door at that time. If valid the computer sends a command back to the panel to release the door. In these systems if the computer goes down, or if communications between the panel and computer is lost, the system can no longer function, to verify proper access, and to process alarms.

Distributed Processing. In distributed processing systems the database is loaded to the field panel. All decisions are made at the field panel and are passed to the computer or logging printer for storage. In these systems if communications is lost, access control continues uninhibited. Furthermore, the events can sometimes be stored in the panels, and can be sent up to the computer once communications is restored. Due to their architecture, systems which employ distributed processing generally offer better reliability, and faster response than systems that rely on central computers for all decision making.

Example Resolved. Building management can configure time schedules and assign them to Zones and Roles, that a way a Zone could have a time schedule from 8-5pm. Moreover, time schedules can be assigned to Relays and Alarms so that they can be activated and monitored respectively when the system clock activates these time schedules.

Known Uses. Many commercial or institutional buildings control access to restricted units using the concepts presented here. This model corresponds to an architecture that is seen in commercial products, such as: Secure Perfect, Picture Perfect and Diamond II. BACnet is a standard that includes access control to buildings and incorporates RBAC, zones, and schedules [12].

Consequences. They are a combination of the previous patterns.

Related Patterns. This pattern is a combination of the RBAC (or another suitable authorization model [7]), the Physical Structures pattern, the Access Control for Physical Structures pattern, the Alarm Monitoring pattern, and the Relays pattern.

4 Related Work and Discussion

There is a considerable amount of work on the security of SCADA (Supervisory Control and Data Acquisition) networks, e.g. [13][2]. However, that type of security is not applicable to physical protection, SCADA security applies only to the information functions of these networks. Building security appears as a neglected area, the only analysis of this problem come from industry white papers, e.g. [14], which emphasizes the need for the convergence of physical and logical security. Some documents about BACnet discuss its rationale, which has some close aspects to our work, they use RBAC and try to derive conceptual models of their protocols but they are tied to some implementation details. R.Martin's book [15] has a detailed building security system used as an example of applying object-oriented methods, but he does not use patterns.

Some work attempts to find ways to control location information of people, e.g. [16]. A model of that type can be made more precise by specifying more closely the location information, as we can do with our patterns. [17] proposes a formal model for the release of information about user location in a smart building while respecting privacy constraints. Our approach can be used to define location of sensors and for interpreting information about user location. By defining more closely the location of a user, her privacy can be protected more precisely. Our finer location definition comes from the fact that we use the Composite pattern [4], a recursive structure that can have any levels of containment. This approach can also be combined with approaches that use geometric models of location [18] by providing an anchor or context for the location information.

There is also much work on context-dependent security, e.g. [19][20], where access to services or resources depends on the location of the user. It is clear that our model can make contexts more specific by defining them in relation to zones or access points (gates). Context models are usually applied in wireless environments, a user could prove wirelessly that he is in front of a gate and that gate could be opened remotely. In the literature, the rights considered are about information, e.g. to a list of nearby restaurants, our approach can unify both types of accesses.

Emergency situations are very important for physical access, in the case of fire all doors should be opened, in the case of an attack as the recent one at Virginia Tech, all doors could be closed. Our model handles these cases very well, all the doors are just instances of a class Door and we can have operations such as 'open all doors' or 'close all doors', that apply to all the objects of that class.

As indicated earlier, the presented model is semi-formal. It can be made more formal by adding Object Constraint Language (OCL) constraints [21]. OCL can be used to add formal annotations to a UML model. Another possibility is the use of the template notation of [22]. In that way, we can at least define more precise requirements and maybe prove properties of some sections of the system.

5 Conclusions

We have described using patterns the basic features and concepts that any modern Physical Access Control system must have. As indicated, these patterns can guide the design of physical access control systems or they can be used to evaluate current products of this type. Other possibilities include the dynamic restriction of the locations where a suspicious user could go or reconfiguration of exits in case of emergencies. They can also be used in conjunction with privacy-oriented models. Another next step is a pattern that could be more event-driven in a way that any subsystem that generates events could be hooked dynamically. Our Access Control system could be one of the subsystems as well as a video system, a metal detector, a drug detector, and so forth.

The contributions of this paper include a constructive semi-formal model of access control to physical structures and a set of patterns, which can be used on their own to build other models. This model also illustrates composition of features by composing patterns. Our future work will include combining this model with identity management patterns [23], with context-based models, and with traditional RBAC models (users could have their access to information and to physical locations defined by the same model). Further formalization is another aspect we are considering. Finally, physical accesses may involve not just entering a zone but may include physical removal of specific objects identified by RFID devices.

References

1. SSPC135/LSS-WG: Physical access control with BACnet (October 2006), http://www.bacnet.org/bibliography/bac-10-06.pdf
2. Majdalawieh, M., Parisi-Presicce, F., Wijesekera, D.: Dnpsec: A security framework for dnp3 in SCADA systems. In: Internat. Joint Conf. on Computer Information and Systems Sciences and Engineering, Bridgeport, CT (December 10-20, 2005)
3. Buschmann, F., Meunier, R., Rohnert, H., Sommerlad, P., Stal, M.: Pattern-Oriented Software Architecture: A System of Patterns, vol. 1. Wiley, Chichester (1996)
4. Gamma, E., Helm, R., Johnson, R., Vlissides, J.: Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley, Boston, Mass (1994)
5. Schumacher, M., Fernandez, E.B., Hybertson, D., Buschmann, F., Sommerlad, P.: Security Patterns: Integrating security and systems engineering. J. Wiley & Sons, Chichester (2006)
6. Steel, C., Nagappan, R., Lai, R.: Core Security Patterns: Best Strategies for J2EE, Web Services, and Identity Management. Prentice Hall, Upper Saddle River, New Jersey (2005)
7. Fernandez, E.B., Pan, R.: A pattern language for security models. In: Procs, of PLoP (2001), http://hillside.net/plop/plop2001/accepted_submissions/accepted-papers.html
8. Priebe, T., Fernandez, E.B., Mehlau, J.I., Pernul, G.: A pattern system for access control. In: Procs. of the 18th Annual IFIP WG 11.3 Working Conference on Data and Applications Security, Sitges, Spain, pp. 235–249 (July 2004)
9. Desouza-Doucet, A.: Controlling access to physical locations, M.S. Thesis, dept. of computer science and eng., Florida Atlantic University (April 2006)
10. Fernandez, E.B., Sinibaldi, J.C.: More patterns for operating system access control. In: Proc. of the 8th European conference on Pattern Languages of Programs, pp. 381–398 (2003), http://hillside.net/europlop
11. Fernandez, E.B.: Security patterns (keynote talk and paper). In: Procs. of the Eigth International Symposium on System and Information Security - SSI2006, Sao Jose dos Campos, Brazil (November 08-10, 2006)
12. Ritter, D., Isler, B., Mundt, H., Treado, S.: Access control in bacnet. BACnet today (supplement to ASHRAE Journal) B26–B32 (November 2006)
13. Chandia, R., Gonzalez, J., Kilpatrick, T., Papa, M., Shenoi, S.: Security strategies for SCADA networks. In: Procs. First Annual IFIP WG 11.10 International Conference on Critical Infrastructure Protection,
14. Bridging the great divide: The convergence of physical and logical security (August 2006), Imprivata(http://www.imprivata.com)
15. Martin, R.: In Designing Object-Oriented C++ Applications Using the Booch Method (Chapter 6). Prentice-Hall, Englewood Cliffs (1995)
16. Hengartner, U., Steenkiste, P.: Implementing access control to people location information. In: Procs. of the ACM Symposium on Access Control Models and Technologies (SACMAT'04), ACM Press, New York (2004)

17. Boyer, J., Tan, K., Gunter, C.: Privacy-sensitive location information systems in smart buildings. In: Procs. of the 3rd Int. Conf. on Security for Pervasive Computing, York, England (April 2006)
18. Atluri, V., Shin, H.: Efficient enforcement of security policies based on tracking of mobile users. In: 20th Annual IFIP WG 11.3 Working Conference on Data and Applications Security (July 2006)
19. Corradi, A., Montanari, R., Tibaldi, D.: Context-based access control management in ubiquitous environments. In: Proceedings of the Third IEEE International Symposium on Network Computing and Applications (NCA'04), IEEE Computer Society Press, Boston, MA (August 30 - September 01, 2004)
20. Huldenbosch, R., Salden, A., Bargh, M.S., Ebben, P.W.G., Reitsma, J.: Context-sensitive access control. In: Procs. of SACMAT (2005) 111–119 (2005)
21. Warmer, J., Kleppe, A.: The Object Constraint Language, 2nd edn. Addison-Wesley, Reading (2003)
22. Ray, I., Li, N., Kim, D., France, R.: Using parameterized UML to specify and compose access control models. In: Proceedings of the Sixth IFIP WG 11.5 Conference on Integrity and Control in Information Systems. Lausanne, Switzerland (November 2003)
23. Delessy, N., Fernandez, E.B., Larrondo-Petrie, M.: A pattern language for identity management. In: Delessy, N. (ed.) Accepted for the 2nd IARIA Int. Multiconference on Computing in the Global Information Technology (ICCGI 2007), Guadeloupe, French Caribbean (March 4-9, 2007)