ABSTRACT

Title of Dissertation / Thesis: **DESIGN OF AN INTELLIGENT SENSOR NETWORK FOR BUILDING SECURITY**

Rajeshree Varangaonkar MSSE 2004

Dissertation / Thesis Directed By: **Prof. Dr. John. S. Baras**

The thesis deals with the design of an intelligent sensor network for protecting premises from chemical, biological and intruder attacks. This thesis gives a logical level design along with the architectures at various levels of hierarchy. The use of object technology is proliferating in the development of software, and in order to build robust and maintainable complex systems, mastering object-oriented (O-O) analysis and design is essential. The main goal of this thesis is to report on the experience of applying object-oriented modeling, analysis and design methodology to a real-world complex system represented by an intelligent sensor network for a building. UML has been used to model the software and automation infrastructure, which handles the interactions among processing elements in a modern building.
A set of system design requirements are developed that cover the hardware design of the nodes, the design of the sensor network, and the capabilities for remote data access and management. A formal model is proposed for the architecture, and the behavior diagrams explain the dynamic nature of the system. The static and dynamic diagrams together validate and verify the system. Agent UML is discussed to model evacuation of a room. This thesis discusses some extensions to UML for agent-based

modeling where the agents follow a purely reactive and proactive approach. In this work, agent-based architectures and behavior diagrams are proposed as a method to envision security in buildings. Extensions are provided to support a multi room scenario. Sensor fusion is used to provide a robust functionality and reducing the events of false alarms occurring in the system. Linear programming techniques are used to solve for the minimal point in the cost vs. performance trade off curve for the sensor network as well as for the access system proposed. The tradeoff explores the relation of variables and suggests an operating point satisfying all constraints and without violating any requirement. Solver, an Excel add-in has been used to run the linear optimization.

DESIGN OF AN INTELLIGENT SENSOR NETWORK FOR BUILDING SECURITY

By

RAJESHREE VARANGAONKAR

Thesis or Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park, in partial fulfillment
of the requirements for the degree of
MASTER OF SCIENCE
SYSTEMS ENGINEERING
2004

Advisory Committee:
Professor Dr. John Baras, Chair
Dr. Mark Austin
Dr. Mike Cukier

Acknowledgements

Firstly, I would like to express my most sincere gratitude and appreciation to my supervisor Dr. John Baras for his guidance and encouragement. His experience and insight have been invaluable for my work on this thesis. He has repeatedly given me invaluable insights and pointers on this project. It was a great experience working with him.

Special thanks to Dr. Mark Austin who has been giving me constant advice on this project. I sincerely would like to thanks him for his supportive and encouraging nature.

I also want to thank Dr. Anandalingam for giving me an insight in the world of decision theory. He is a great teacher and I really enjoyed learning Linear Optimization from him.
Thanks to Dr Cukier for being so accommodative. I must also thank Althia Kirlew for her patience and efficiency.

I am grateful to the friends and colleagues at University of Maryland who have made working on this thesis enjoyable.

I must also thank people at Battelle who supported my thesis venture.

Table of Contents

List of Figures

List of Tables

Chapter 1: Problem Statement and an Introduction to UML

1.1 Modeling an Intelligent Sensor Network for a building

Intelligent sensor networks for buildings protect the building from intrusion and terror attacks. The intelligent sensor network will monitor the premises and protect it against any untoward incident. The solution proposed is modular with key areas identified as the premise itself, the central unit overseeing the system, the sensors and the network. A heterogeneous sensor network is envisioned that would sense and monitor the environment. Keeping a low fault rate this network would use wireless communications to detect any untoward incident. A time efficient access system is required to screen entry into the premise and work in conjunction with a hierarchical security system preventing ingress at any point without validation of credentials. A tiered approach to the problem with in-built redundancy is desired. Also contingency plans in the event of an attack should be provided. The design should be scalable and take into account budget constraints.

1.2 Scope and Objectives

The solution is formulated at a level of abstraction and proposes architectures that can meet the requirements. This project was done for Battelle; who would be using their proprietary protocols and pre-decided sensors to constitute their systems. This thesis lays out a framework and an architecture design that can be used to develop the system.

The thesis drafts a design for an intelligent sensor network that will serve to protect the premises from a chemical, biological or intruder attack. UML is used to draft out the design envisioned for this system. Starting with gathering user requirements using Use Cases (Chapter 2), the system goes on to establish requirements for the system (Chapter 3). These requirements have corresponding validation scenarios that provide a check-list to ensure all the requirements are met. Tests, simulations etc are also documented as a means of testing if the requirements are truly met. The class diagrams (Chapter 4) proposes various architectures for the components of the system. With the Interaction and Behavior diagrams the dynamic behavior of the system is put in place. Sensor fusion is explained in the next section as a collaborative means of consolidating information regarding a particular measurement. Agent UML is discussed to model behavior that can not be captured using UML, this primarily comprises of objects rejecting messages received and acting on their own will. This is used to model behavior of people when they are evacuating. The last chapter deals with two tradeoff studies. The first study provides a deployment strategy for sensors keeping in mind the cost, detection probability and fault rates. The other tradeoff study uses queuing models to minimize time taken to clear employees while considering cost and reliability.

The thesis aims at laying down guidelines and recommendation for system architecture and behavior. The task is multi-disciplinary and involves liaison between different departments at a company to achieve a robust, effective solution to the problem of unauthorized access and threat from chemical and biological agents.

UML diagrams have been used in the proposed solution and the thesis starts with an introduction to UML.

1. 3 UML

A Model is an abstract representation of a specification, a design or a system from a particular point of view. It is often represented visually by one or more diagrams. It aims at expressing the abstracts without going into the details. Modeling is an essential part of large software projects, and helpful to medium and even small projects as well. A model plays the analogous role in software development that blueprints and other plans (site maps, elevations, physical models) play in the building of a skyscraper. Using a model, those responsible for a software development project's success can assure themselves that business functionality is complete and correct, end-user needs are met, and program design supports requirements for scalability, robustness, security, extendibility, and other characteristics, before implementation in code renders changes difficult and expensive to make.

A modeling language is a way of expressing the various models produced during the development process. A modeling language defines a collection of model elements. A modeling language is usually diagrammatic but could be text based. It has:

Syntax- in diagram based modeling language, the rules that determine which diagrams are legal

Semantics- the rules that determine what a legal diagram means

The UML is "a language for specifying, visualizing and constructing the artifacts of software systems . . ." [Booch97]. UML is not a methodology. Any method can be used to gather software artifacts that are represented in UML as long as the meaning of those artifacts comply with the definition in the used notation.

UML is a graphical language for {Visualizing, specifying, constructing, documenting} the artifacts for a software intensive system. It has become the de-facto standard for object oriented software modeling. UML allows the modeling of different aspects of systems at different levels of abstraction.

The OMG's Unified Modeling Language™ (UML™) helps you specify, visualize, and document models of software systems, including their structure and design, in a way that meets all of these requirements. [1]

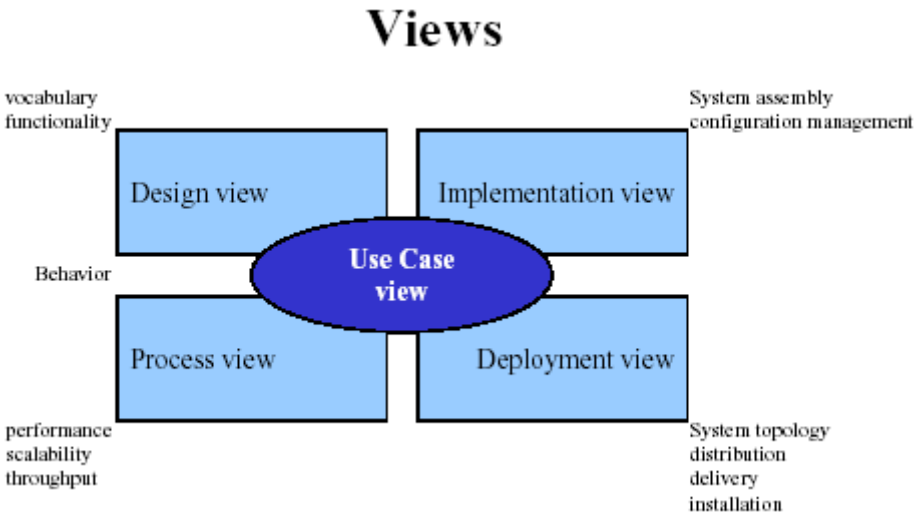


Figure 1 Views

Any development process aims to produce an implemented system. This is a program or collection of programs which work in an appropriate environment to fulfill user needs. The design and the architecture embody the important decision about how the system is built, abstracting away from many details. There are different views to a system such as

- logical view: Modeled to check if functional requirements are met
- process view: to insure that non functional requirements such as performance and availability are met.
- development view: issues such as team assignments and reuse are dealt with for better management of the project
- physical view: running and execution

The following figure further pictorially expresses the essence of the different views adapted. [2]

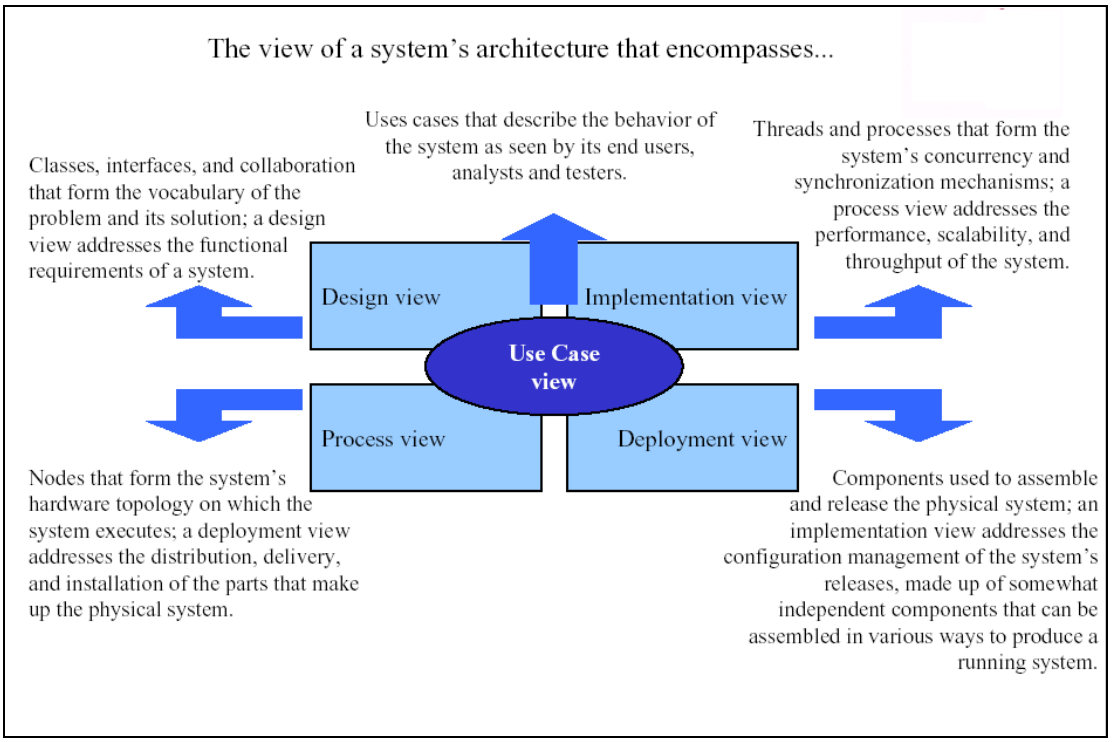


Figure 2 Views Explained

UML's twelve standard diagram types.

UML defines twelve types of diagrams, divided into three categories:

Four diagram types represent static application structure; five represent different aspects of dynamic behavior;and three represent ways you can organize and manage your application modules.

Structural Diagrams include the Class Diagram, Object Diagram, Component Diagram, and Deployment Diagram. These, static models describe the elements of the system and their relationship.

Behavior Diagrams include the Use Case Diagram (used by some methodologies during requirements gathering); Sequence Diagram, Activity Diagram, Collaboration

Diagram, and Statechart Diagram. Used to represent the dynamic model that describes the behavior of the system over time.

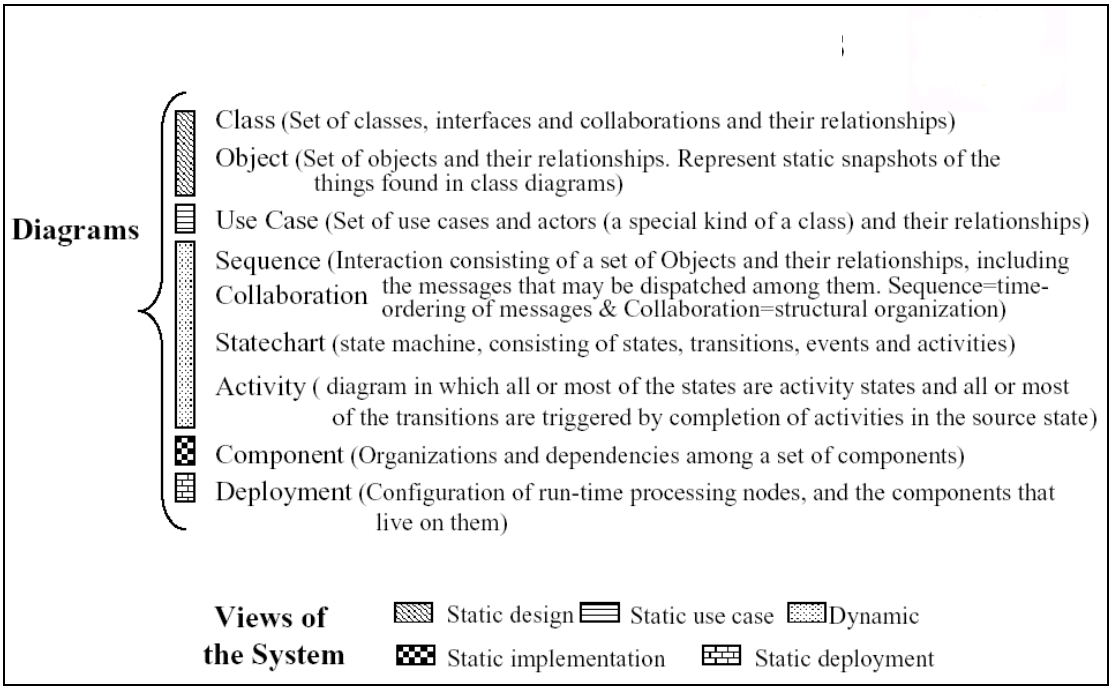


Figure 3 UML Diagrams

UML unites and formalizes the methods of many approaches to the object oriented software lifecycle, including Booch, Rumbagh, Jacobson and Odell

Chapter 2: Capturing System Functionality

2. 1 Introduction to Use Cases

Use Case analysis is one of the first and primary means of gathering requirements in the behavioral methodology. Use Cases are a critical technique in developing an application. Use cases are a standard technique for gathering requirements in many modern software development methodologies. Within the UML Use Cases are used primarily to capture the high level user-functional requirements of a system. Neither can Use Cases be effectively used to capture non-functional requirements nor can they be used to capture "internal" functional requirements. Primarily because Use Cases are an informal and imprecise modeling technique. Secondly because the other use that is made of Use Cases is to define the fundamental structure of the application.

2.1.1 Defining Use Cases

The very first question to be answered then is why do we develop the Use Case model - what Use Cases are and also - very importantly - what they are not.
The Use Case model is about describing “what” our system will do at a high-level and with a user focus for the purpose of scoping the project and giving the application some structure. The Use Cases are the unit of estimation and also the smallest unit of delivery. Each increment that is planned and delivered is described in terms of the Use Cases that will be delivered in that increment.

Use Cases are not a functional decomposition model. Use Cases are not intended to capture all of the system requirements. Use Cases do not capture”how” the system will do anything - nor do they capture anything the actor does that does not involve the system. All of these things are better modeled using other modeling techniques that were developed for those purposes.

- The Object Model to capture the static structure of the system and the composition of the classes.
- Object Sequence Diagrams and State Transition Diagrams to capture the detailed dynamic behavior of the system - the how.

Use Cases are not an inherently object-oriented modeling technique.
Use cases in UML are defined in various but similar ways within the literature. [3]

"A use case is a narrative document that describes the sequence of events of an **actor** (an external agent) using a system to complete a process."[Jacobson92]

"They are stories or cases of using a system. Use cases are not exactly requirements or functional specifications, but they illustrate and imply requirements in the stories they tell."[Larman98]

"...domain processes can be expressed in use cases – narrative descriptions of domain processes in a structure prose format."[Larman98]

"A description of set of sequences of actions, including variants, that a system performs that yield an observable result of value to an actor."[Booch99]

"You apply use cases to capture the intended behavior of the system you are developing, without having to specify how that behavior is implemented. Use cases provide a way for your developers to come to a common understanding with your system's end users and domain experts. In addition, use cases serve to help validate your architecture and to verify your system as it evolves during development."[Booch99]

2.1.2 Types of Use Cases

There are two types of use cases.

- Essential Use Case [Constantine97] :and the other type,
- Real Use Case [Larman98].

These use case types are defined below:

"Essential Use Cases . . . are expressed in an ideal form that remains relatively free of technology and implementation detail; design decisions are deferred and abstracted, especially those related to the user interface."[Larman98]

“Real Use Case concretely describes the process in terms of its real current design, committed to specific input and output technologies, and so on. When a user interface is involved, they often show screen shots and discuss interaction with the widgets."[Larman98]

Essential use cases are of primary importance early in a project’s analysis. Their purpose is to document the business process that the system must support without bias to technology. Later, during project design, real use cases become important

since they document how a specific set of user interfaces will support the business process documented in the essential use case.

The benefits of this style of use for use cases are twofold: the business processes are well documented, and the system requirements are described in terms of the processes they support. This makes for a close mapping between business process and requirements. [7]

2.1.3 What is the objective of use case analysis?

In general terms, the purpose of use case analysis is to document the business process that is to be supported by the system under development. However, to effectively develop a component-based application for that process, use case analysis must have a much more specific purpose. Use cases must document the business process to be supported in such a way as to facilitate the identification of *operations* that support the business process. Use cases must achieve the following goals in order to be effective for this stated purpose. [4] [8] [9]

1. Use cases must be an **Effective Communication Tool**
2. Use cases must be scoped to a **Specific Business Goal**, which means they must identify **Business Decisions and Actions.**
3. Use case steps must be identified as **Automated or Manual**

Effective Communication Tool

Use cases are a tool for customers to communicate the business requirements to software developers. For this to be an *effective* tool, the software analyst must be able to coax the customers to give them the right information. It is possible to perform

business analysis by way of use cases and still not get the information necessary to build a good software solution; however, if the information does not communicate the requirements, or if the software developers find them impossible to use, then the use cases have been done improperly.

Use cases are a tool for software developers to communicate how the system meets the customer's business requirements. The use case documents the business process that the software solution is designed to support. Software developers must be able to map the specific features and functionality of the system to the use case. This mapping allows developers to relate requirements to system functions to prove that the system meets the requirements of the system. If the software solution cannot be effectively mapped to the use case, then the software solution does not meet the business requirements.

Use cases prove their worth when several things happen upon completion of the software solution. First, the customer, upon seeing the use cases again, agrees that the use cases properly describe the business process to be supported. Second, the software developers can show exactly how the system explicitly supports that business process. Third, the customer agrees that the system supports the business process as expected. Last of all, when a *really* good job was performed on the use cases, the customer will state, "I would like to take these use cases and use them as our procedure manual. We never have had this process documented so well and it would really help to train our staff."

Business Goals, Decisions, and Actions

The use case definition provided above mentions that a use case must accomplish a business goal. This concept is very important to use case development and is illustrated in the following quotes.

"An important issue I've come across with use cases is the difference between what I call user goals and system interactions."[Fowler97]

"Both styles of use cases have their applications. System interaction use cases are better for planning purposes; thinking about user goals is important so that you can consider alternative ways to satisfy the goals. If you rush too quickly toward system interaction, you will miss out on creative ways to satisfy user goals more effectively than you might by using the obvious first choice. In each use case it is a good idea to ask yourself, "why did we do that?" That question usually leads to a better understanding of the user goal."[Fowler97]

The identification of the business goal provides the analyst with the invaluable insight of knowing why each of the steps is to be performed. This leads to a system that better supports the business because the analyst can offer alternative solutions and, as a result, creates a system that adds more business value.

Once the goal of the use case has been defined, each of the steps, manual and automated, necessary to achieve that goal are documented. While a use case is supposed to describe the interactions between an actor (user or other system) and the

system, it is too early in the process to distinguish between manual and automated steps. In addition, the documentation of manual steps forms a complete picture by which a user can understand exactly where the system supports the business process and where manual work is required. This documentation of manual steps makes the use case a more effective communication tool.

The best way to perform use case development for a business application is to focus on identifying business decisions and business actions in the use case steps. All steps are documented, but it is important to understand how each one supports a business decision or business action that in turn helps accomplish the use case goal. This approach will help to weed out unnecessary steps and it will cause each of the steps to have a clear purpose.

Identifying business goals, decisions, and actions is the second objective of use case development (the first is being an effective communication tool). Focusing on these items during use case development will greatly enhance the business value of the delivered system. Unfortunately, properly identifying the goals, decisions, and actions can present a challenge to analysts.

Automated vs. Manual Steps

Each use case step must be identified as automated or manual. The focus of each step is to make a business decision or execute a business action. Assigning responsibility for each business decision and business action to either the **system** (automated) or the

actor (manual) directly impacts the system delivered to support the business process because the automated steps will result in system operations to make these decisions or execute these actions.

The system operations will be named according to the decisions or actions for which they are responsible. Naming operations this way will aid the ability to trace requirements to the delivered system because the operation name will reflect its business purpose, which should map to a business requirement.

Use cases are a very useful analysis tool. They can be used to define the business processes and the system requirements that are necessary to support that process, which leads to a natural mapping between the business processes and the requirements.

There are different types of use cases that can be used in different situations. Certain use cases describe business processes and the system response at a high level (essential use cases). These use cases can be refined to describe the interaction that takes place in a particular implementation of a system (real use cases).

The primary goal of use case analysis is to be an effective communication tool that describes the business processes and assigns responsibility to the steps of the process to either the system (an automated step) or to an actor that is outside of the system (a manual step).

Use cases document the behavior of the system from a User's point of view. Anything external to the system and that interacts with it qualifies as a user in this context. Use case modeling helps with the following three important aspects:

- Capturing requirements
- Planning Iterations of development
- Validating systems

2.2 Hierarchical Use Cases

Since the system analyzed is large scale a hierarchical approach to designing use cases has been adopted. Three primary areas have been identified for the system:

- Access consisting of
    - intruder attack,
    - authorized access by employee,
    - visitor access
    - enrollment
    - and tailgating
- Fault consisting of
    - Partial Failure,
    - Power failure,
- Sensor consisting of
    - Sensing and monitoring
    - Add sensor,
    - Delete sensor

The figure below shows the system level Use case model. The use case depicts 3 higher level use cases viz Access, Sensor and Fault. The chief actors are the threats in terms of intruders and threats which interact with the system, the staff including the employees, maintenance and operating staff and the security.

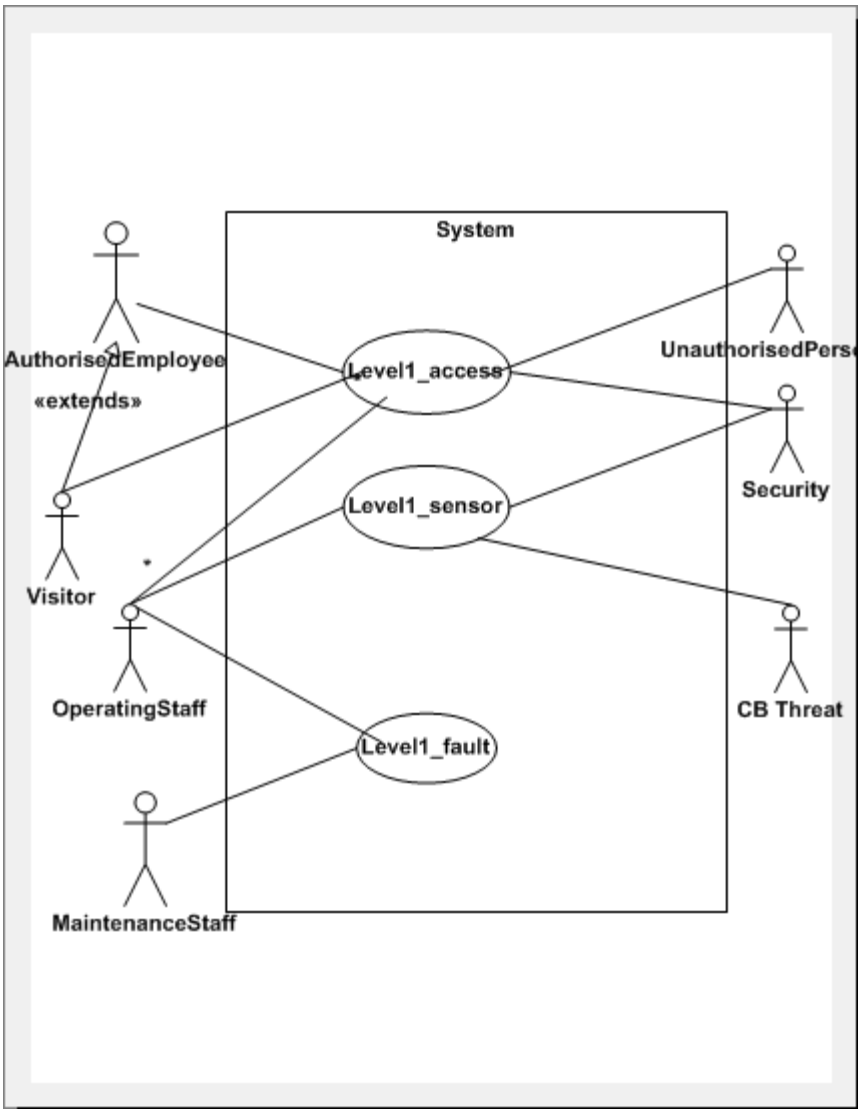


Figure 4 System Level Use Case

2.2.1 Decomposition of higher level Use Cases

Each Use Case in Figure 5 merits a Use Case model as it is a condensation of an aspect of the system under consideration. The system in itself comprises of the protected premise which has its own functions and components such as people, electrical system, plumbing system details of which will be dwelt on in Chapter 4. Anything external to the system which would include people, environment, would instigate a response on interaction with the systems of the protected premises. These reactions could be classified as

1. Normal; comprising of routine activities such as- employee access arising from the fact that employees will enter and leave the building every day in what is classified as an authorized access; a temporary authorized access by a legitimate visitor, checking identification, sensing and monitoring by the sensors, diagnostics checks among others.
2. Abnormal; these in a good system would arise relatively infrequently and comprise of alarms, threat detection, faults etc.

Each of the higher level use cases comprise of sub use cases that are decomposed in the following sections.

2.3 Use Case Access: Decomposed

The next figure gives an insight into the level 1 Use case: Access. This use case captures all conceivable interactions between the access system of the building and an attempt to gain entry. It includes the interaction of intruders; an attempted authorized entry etc. with the system.

In either case whether the access is legitimate or illegitimate the use case VerifyID gets executed for every access attempted. The objective of the use case VerifyID is to allow authorized personnel to enter the protected premises while keeping the intruders out. The execution of this use case is an integral part of any attempt to gain entry and hence has the include relationship with the other use cases.

The tailgating use case deals with the case when an authorized access is followed by an unauthorized entry before the door closes. Such a scenario can be prevented by a motion sensor which will detect two entries instead of the authorized one and signal security. The idea here is one entry for each attempted access. This use case can also be extended to install a check into the information system to see if a person logging into a computer in the company has passed the proper levels of authorization. This could include a check to see if the person has gained legitimate entry; if not then he/she would not be allowed to gain access to the system.

The enrollment process consists of a record that has to be generated between the ID card number of the employee and the employee name and department. Some employees are authorized to gain access into the enrollment system. Once this has been done, the employee swipes his card. The number is captured and the name and department of the employee is thrown up, when the employee verifies his information a record is established in the user account and a random code is generated. The employee now has to enter this code whenever he attempts access.

2.3.1 The Access Use Case:

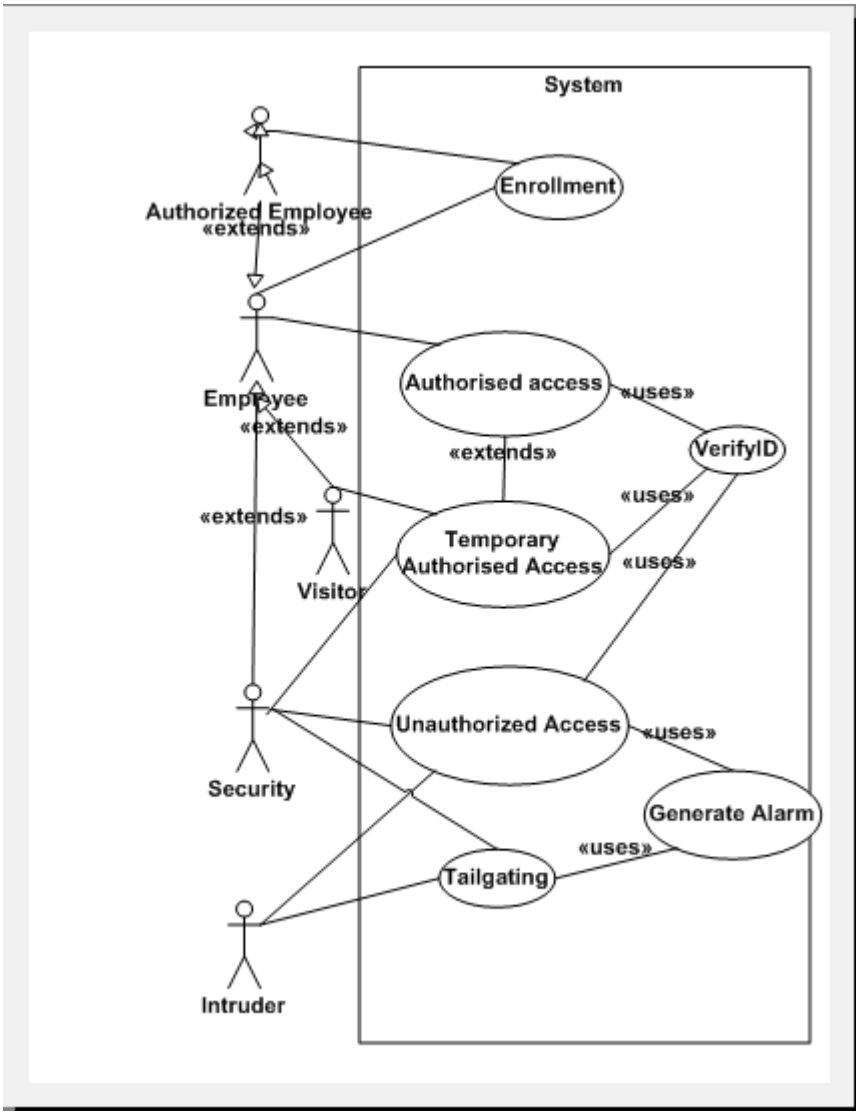


Figure 5 Access Use Case

Figure 6 is a pictorial representation of the Access Use Case. Below is a documentation of the Use case. A standard template [13] has been used throughout to document use cases.

Use Case Name Iteration	Level1_access- Authorized Access
Summary:	All employees who need access to the system need to enter an identification number and swipe a card. If the combination matches with the one stored access is permitted.
Basic Course of Events	1. Authorized personnel enters building 2. Authorized personnel supplies information for identification procedure. Use VerifyID to return result. 3. Authorized person is verified 4. Authorized person is allowed access
Alternative paths **Exception path**	N/A 1. If in step 2 Employee enters invalid ID code 2. Display error message, request id again 3. Return to step 2 4. Allow procedure to repeat 3 times, then confiscate card and lock exits, alert security by alarm generation
Extension Points	At point 1 determine if employee is a visitor switch to use case Temporary authorized access
Trigger	Employee requires access
Assumptions	All sensors are operating and there is no fault in the system
Preconditions	The system is operational
Post conditions	Validated user is allowed access
Related Business Rule	N/A
Author	Rajeshree Varangaonkar
Date	Sep-03

Table 1 Authorized Access

Use Case Name	Level1_access- Temporary authorized Access
Iteration	
Summary:	All visitors who need access to the system need to confirm their identity with the security who verifies it with the employee being visited. Once the visitor has been verified he/she is assigned temporary identification.
Basic Course of Events	1. Visitor requests access at gate
	2. Credentials identified with authorized employee seeing the visitor.
	3. Assigned temporary information to pass the identification process
	4. Post condition
	5. Visitor admitted
	6. On leaving visitor surrenders temporary information which is destroyed
Alternative paths	N/A
Exception path	1. If in step 2 visitor enters invalid ID code
	2. Display error message, request id again
	3. Return to step 2
	4. Allow procedure to repeat 3 times, then confiscate card and lock exits, alert security by alarm generation
Extension Points	N/A
Trigger	Visitor requires access
Assumptions	All sensors are operating and there is no fault in the system
Preconditions	The system is operational
Post conditions	Validated user is allowed access
Related Business Rule	N/A
Author	Rajeshree Varangaonkar
Date	Sep-03

Table 2 Temporary Access

Use Case Name Iteration	Level1_access- Unauthorized Access
Summary:	An unauthorized person tries to gain access and must be apprehended
Basic Course of Events	1. Unauthorized person enters premises 2. Unauthorized person tries to gain access by supplying information. Check by using VerifyID 3. Information fails verification. 4. Lock exits 5. Alarm alert, use Alarm Generation to generate alarm 6. Security is alerted
Alternative paths	1. Detector has failed and no alarm sounds alerting of UAP 2. When the UAP tries to enter any door inside the building the access system can be configured to detect if person has gained access through proper channel. 3. If the person hasn't door will be locked and security alerted.
Exception path	N/A
Extension Points	N/A
Trigger	Unauthorized person tries to gain access
Assumptions	All sensors are operating and there is no fault in the system
Preconditions	The system is operational
Post conditions	Unauthorized person apprehended.
Related Business Rule	N/A
Author	Rajeshree Varangaonkar
Date	Sep-03

Table 3 Unauthorized Access

Use Case Name Iteration	Level1_access- Tailgating
Summary:	An unauthorized person tries to gain access by trying to get in through the door when an authorized employee enters. When the authorized employee has been verified and door has opened there will be a small lag before the door closes again during which the unauthorized person will try to slip in.
Basic Course of Events	1. Authorized employee enters premises 2. Authorized employee supplies information, is verified using VerifyID and enters premises 3. When door opens Unauthorized employee tries to get in through the door too. 4. Detector on door detects two entries of people for one card swipe 5. Exits are locked and Alarm is sounded, using Alarm Generate 6. Security apprehends UAP
Alternative paths	N/A
Exception path	N/A
Extension Points	N/A
Trigger	Intruder tries to tailgate
Assumptions	All sensors are operating and there is no fault in the system
Preconditions	The system is operational
Post conditions	Unauthorized person apprehended.
Related Business Rule	N/A
Author	Rajeshree Varangaonkar
Date	Sep-03

Table 4 Tailgating

Use Case Name Iteration	Level1_Access Enrollment
Summary:	Employees are enrolled into the database for verification of ID upon activation
Basic Course of Events	1. Authorized personnel swipes card and selects enrollment from the menu. 2. Employee to be enrolled then inserts his card in the slot of the card reader and the number on card is captured. 3. Employee's name and department recovered from existing database and displayed 4. Employee verifies information and presses "Enter" 5. System generated a key in ID code to be used by employee when attempting access 6. Authorized personnel then swipes his own card again and exits the enrollment mode.
Alternative paths	1. Employees record not found on card being swiped 2. Go back to step 1 and restart process 3. If error recurs notify administrator
Exception path	N/A
Extension Points	N/A
Trigger	Authorized personnel swipes card and selects enrollment from menu
Assumptions	System operating and there is no fault in the system
Preconditions	The system is operational, Employees that can carry enrollment have been authorized to do so
Postconditions	Employee is enrolled
Related Business Rule	N/A
Author	Rajeshree Varangaonkar
Date	Sep-03

Table 5 Enrollment

Use Case Name	Level1_access- Alarm Generation

Iteration	
Summary:	Alarm condition here would mean violation of limits or setting of a value. Usually the value at which an alarm is generated is set at + XX% of normal value for sensor measurement at the higher side and -XX% of normal value for sensor measurement at the lower side
Basic Course of Events	1. If (sensor measurement IS EQUAL TO alarm condition) then Sound alarm, end. 2. Security alerted 3. Attend reason 4. Reset alarm
Alternative paths	Alarm fails, a backup circuit will kick in setting off the alarm
Exception path	N/A
Extension Points	N/A
Trigger	Sensor measurement reaches alarm condition
Assumptions	All sensors are operating and there is no fault in the system
Preconditions	The system is operational
Post conditions	Alarm condition rectified
Related Business Rule	N/A
Author	Rajeshree Varangaonkar
Date	Sep-03

Table 6 Alarm

Use Case Name Iteration	Level1_access- VerifyID
Summary:	Anybody who tries to gain access has to enter an identification code and swipe his card, if a match is found the person has been authenticated.
Basic Course of Events	1. Read identification code 2. Read Barcode on card 3. Compare combination with stored records 4. If match found return result TRUE 5. If match not found return result FALSE
Alternative paths	N/A
Exception path	If any of the inputs not received in correct format/not received at all request information again
Extension Points	N/A
Trigger	Inputs received as identification code and bar code
Assumptions	System operating and there is no fault in the system
Preconditions	The system is operational
Post conditions	Inputs compared result ready
Related Business Rule	N/A
Author	Rajeshree Varangaonkar
Date	Sep-03

Table 7 Verify ID

2.3.2 Fault Use Case

Figure 7 is a pictorial representation of the fault use case. The fault use case includes all diagnostics aspects of the system model. The system has a normal mode of operation until it encounters a fault in its system. The sensor would be equipped with in-built diagnostics that would trace the fault. The partially faulty mode is best

described as a drift in the measurement that would be detected on comparison with the reading of redundant sensor for e.g. a 3 way voting scheme can be used to detect an errant reading. This would basically include 3 redundant sensors whose measurements are compared among themselves as well as with a floating average and hence any errant measurement would be voted out. This is a good recourse as the errant reading is within error limits and an out voting eliminates the removal of a sensor and reducing the strength of the logic. It also allows the system availability to be unperturbed. In the faulty mode however logic determines a sensor malfunction by comparing the reading with a redundant sensor. Also here the fault threshold is the error limit along with a tolerance limit to account for the manufacturing difference between any two sensors. An alarm again will be generated and hence is included in the system. In the case a sensor malfunction is detected it is transmitted over the network allowing other sensors in the group to recognize that the sensor is faulty and hence is a compulsory part of the faulty mode use case.

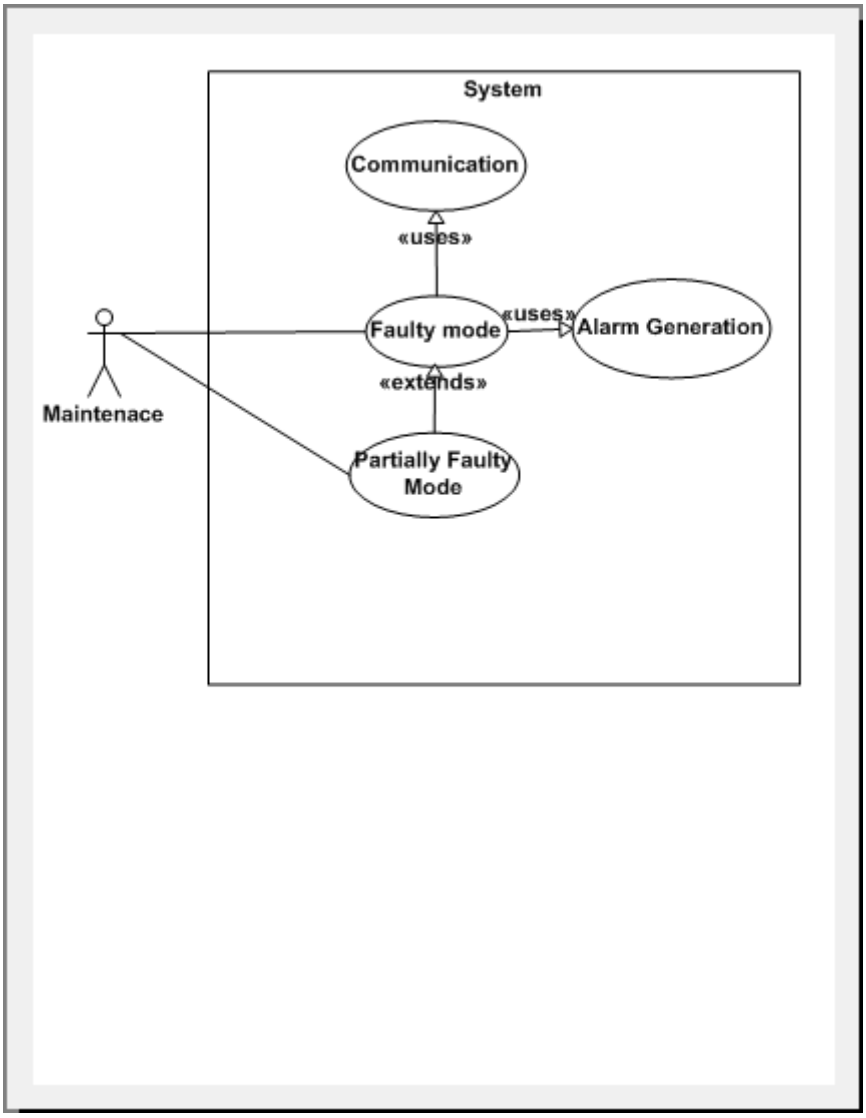


Figure 6 Fault Use Case

Use Case Name Iteration	Level1_fault- Faulty Mode
Summary:	Sensors have their own fault detection circuitry and a fault is discovered and indicated. Every sensor has a redundant circuit and a slave. In case of fault in Master control is switched to redundant circuit/slave
Basic Course of Events	1. Diagnostics detect fault in sensor and transfer control to redundant circuit/slave 2. The faulty mode light comes on. 3. Alarm is generated and maintenance staff alerted. 4. Sensor is attended to and then replaced 5. Sensor is switched back to master mode.

Alternative paths	Alarm fails to generate.
	The measurement form the sensor can be compared with a redundant sensor and also a pre stored average.
	If (reading_redundantsensor IS EQUAL TO (+/- 10%) average)
	reading_redundantsensor=correct
	If (master_sensor IS NOT EQUAL TO (+/-10%) reading_redundantsensor)
	master_sensor has developed an error
	End
	* +/- is set as a tolerance limit to account for manufacturing difference between sensors

Exception path	N/A
Extension Points	N/A
Trigger	Fault signal received
Assumptions	System operating and there is no fault in the system
Preconditions	The system is operational
Post conditions	Fault attended and rectified
Related Business Rule	N/A
Author	Rajeshree Varangaonkar
Date	Sep-03

Table 8 Faulty Mode

Use Case Name Iteration	Level1_fault- Partially Faulty Mode
Summary:	In the partially faulty mode no alarm is generated as the error is usually a drift that does not violate limits. In this case the reading is compared to two other sensors and voting takes place.
Basic Course of Events	Sensor generates errors within limits e.g. a drift or a bias No fault alarm generated as no limits violated A redundancy scheme (for e.g. Triple redundancy scheme) for voting of errant measurement. Sensor measurement tagged as errant by diagnostics, fault should be attended to by maintenance
Alternative paths	N/A
Exception path	N/A
Extension Points	N/A
Trigger	Errant behavior
Assumptions	System operating and there is no fault in the system
Preconditions	The system is operational
Post conditions	Fault attended and rectified
Related Business Rule	N/A
Author	Rajeshree Varangaonkar
Date	Sep-03

Table 9 Partially Faulty Mode

2.3.3 Sensor Use Case

The Use cases highlight the interaction between the external actors and the system components. In the case of the sensor use case, the actors become the external

environment parameters. These parameters such as temperature, pressure, humidity etc interact with the sensing mechanism and produce a change in the sensor output.

The Use Case model includes some standard normal functions like addition of sensors, deletion of sensor, sensing & monitoring and again communication. These methods deal with the inherent functioning of a sensor group and maintain protocol and group dynamics. Use case Threat detected is an abnormal event and enumerates the steps that need to be followed in the event of an untoward incident. Chapter 4 Sensor deals more in detail with various threats and system architectures to counter these threats.

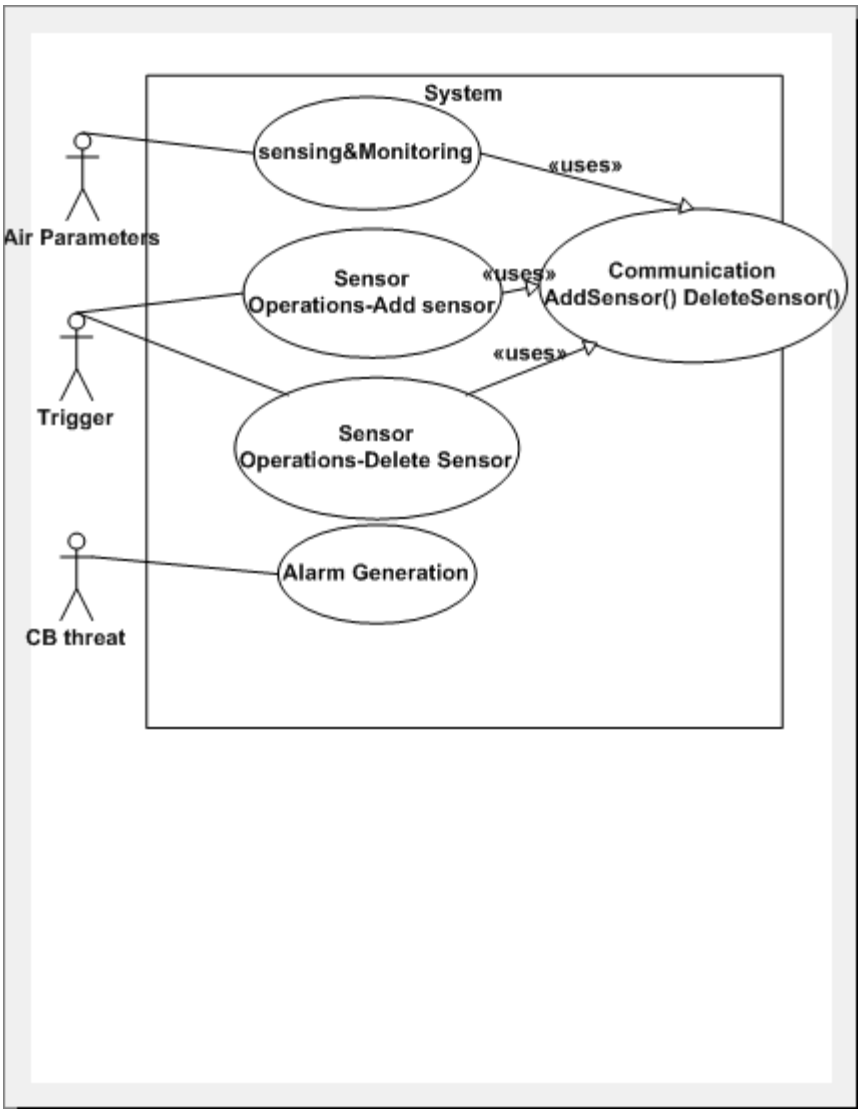


Figure 7 Use Case Sensor

Use Case Name Iteration	Level1_sensor- Threat detected
Summary:	A change in the air quality and measurements indicating the presence of a biological/chemical agent will indicate a threat to the system and evacuation, emergency measures are applied.
Basic Course of Events	1. Sensors detect threat in environment (change in air quality) 2. Alarm is generated 3. suspected Ducts are closed; dampers and fans closed 4. Switch to alternate ducts on the other side of building 5. Isolate area 6. Pressurize remaining areas 7. Apply emergency measures for people 8. Compare outer contamination levels with inner 9. If lower, evacuate; else let people stay in, continue monitoring end 10. Track movement of gas in area check spreading
Alternative paths	1. Alternate ducts also set of alarms 2. Close ducts 3. Isolate suspected area 4. Pressurize uncontaminated area 5. Use emergency measures for people 6. Lead people to safer area 7. Compare outer contamination levels with inner 8. If lower, evacuate else let people stay in, continue monitoring end
Exception path	N/A
Extension Points	N/A
Trigger	change in measurement
Assumptions	System operating and there is no fault in the system
Preconditions	The system is operational
Post conditions	System is protected from attack
Related Business Rule	N/A
Author	Rajeshree Varangaonkar
Date	Sep-03

Table 10 Threat Detected

Use Case Name	Level1_sensor- Sensing&Monitoring
Iteration	
Summary:	All parameters are measured.
Basic Course of Events	1. Sensors are operating and sensing
	2. All measurements are communicated to other sensors
	3. All measurements are within normal limits
Alternative paths	N/A
Exception path	N/A
Extension Points	N/A
Trigger	Change in measurement
Assumptions	System operating and there is no fault in the system
Preconditions	The system is operational
Post conditions	System is continuously monitored
Related Business Rule	N/A
Author	Rajeshree Varangaonkar
Date	Sep-03

Table 11 Sense and Monitor

Use Case Name	Level1_sensor- Add Sensor
Iteration Summary:	Sensors are added dynamically to the group
Basic Course of Events	1. New sensor gets added dynamically 2. Verification for sensor access 3. All sensors update information stored for peers using AddSensor(), communicate
Alternative paths	N/A
Exception path	N/A
Extension Points	N/A
Trigger	Sensor requests to be added
Assumptions	System operating and there is no fault in the system
Preconditions	The system is operational, Number of sensors is known
Post conditions	Group size changed
Related Business Rule	N/A
Author	Rajeshree Varangaonkar
Date	Sep-03

Table 12 Add sensor

Use Case Name Iteration	Level1_sensor- Delete Sensor
Summary:	Sensors are deleted dynamically from the group
Basic Course of Events	1. Sensor drops out dynamically 2. All sensors update information stored for peers using DeleteSensor(), communicate
Alternative paths	N/A
Exception path	N/A
Extension Points	N/A
Trigger	Sensor deleted
Assumptions	System operating and there is no fault in the system
Preconditions	The system is operational, Number of sensors is known
Post conditions	Group size changed
Related Business Rule	N/A
Author	Rajeshree Varangaonkar
Date	Sep-03

Table 13 Delete Sensor

Chapter 3: Requirements

The requirements here are divided into two areas one is the sensor network and the other is the access system. The sensor network as mentioned earlier will have predefined sensors and protocols so the requirements developed are very general and independent of sensor chosen. The sensor requirements are just a framework that can be used while documenting requirements for the real system.

The approach in defining the sensor requirements is slightly unorthodox in that it suggests class diagrams, and some algorithms such as “the lost station algorithm” to illustrate some of the requirements. There is no intention to convert these requirements to specifications. However a map between generic requirements for the system and their validation scenarios is given. So that each requirement is associated to a validation scenario allowing for easy cross-checking.

The access system however is a smaller system and has been dealt with by defining more detailed requirements which are converted to specifications. These specifications have been drawn out by collecting market data from two biometric card companies. [14] [15]

The requirements – validation scenarios follow the orthodox way in which higher level requirements are broken down into lower level requirements. A requirements traceability matrix is drawn. In addition verification scenarios have been included.

3.1 Higher Level Requirements for the sensor network:

Higher level requirements:

- Speed of response – Expected speed of response is targeted around X ms.
- Fault tolerance – The number of fault alarms should be low. An acceptable level is 0.1%. The alarms generated must correspond to untoward events.
- Availability – System downtime must be kept to a minimum. An availability of 95% is desired.
- Tamper proof- system should be tamper proof and give an indication of tampering.
- Cost – low cost. A budget has to be maintained in this implementation. The cost of the system should be lesser than 10000$.
- Hierarchical order A hierarchical order should be ensured. This may be used as a conjunction between two systems to ensure a two tiered approach to security.
- IS system must be protected – the Information system should be protected against unauthorized access.
- Sensors must be equipped with diagnostics. All sensors must have inbuilt diagnostics that would indicate drift, error and faulty modes.
- Remote control: Apart from local computing, remote control must also be enabled.
- Monitoring software is that it should be highly scalable. It should be able to work for a small retail store, an office building, a warehouse or a multi-building complex.

- Sensor network longevity. Power consumption – Low power consumption and an estimated life of XX hours is desired.
- From both a systems and end-user perspective, it is critical that sensor networks exhibit stable, predictable, and repeatable behavior whenever possible. An unpredictable system is difficult to debug and maintain.
- Sensors and sampling: For our particular applications, the ability to sense light, temperature, infrared, relative humidity, and barometric pressure provide an essential set of useful measurements. The ability to sense additional phenomena, such as acceleration/vibration, weight, chemical vapors, gas concentrations, pH, and noise levels would augment them.
- Data archiving: Archiving sensor readings for offline data mining and analysis is essential. The reliable offloading of sensor logs to databases in the wired, powered infrastructure is an essential capability. The desire to interactively "drill-down" and explore individual sensors, or a subset of sensors, in near real-time complement log-based studies.

3.2 Breaking down requirements for sensors

Breaking these down we concentrate on some for the sensors:

- No source of interference to systems being monitored and/or surrounding systems.
- Totally portable and self-sustained (power, communication, intelligence).
- Capable to survive harsh environments (heat, humidity, corrosion, rocket exhaust, etc).

- Minimize operating and maintenance costs.
- Be a highly reliable system, assure data integrity and availability.
- Capable of embedded complex data processing.
- Capable to self-diagnose the communication links and automatically reconfigure upon failure detection

Performance

- Low RF output power (≤10 mW) to minimize any interferences to surrounding systems.
- Battery-operated system with smart embedded power management algorithms to maximize battery life.
- Self-contained system with signal conditioning, data acquisition, data manipulation and data transmission capability.
- Modular flexible architecture reconfigurable to accommodate most sensing technologies**.**

Hardware

- Sensor Network composed of one or more Central Stations and a number of remote stations.
- Central Station:
  - Performs sensor network data collection, storage and distribution to users.
  - Contains data polling schedules and ID tables for the remote stations in sensor network.

- Contains remote stations calibrations curves and engineering unit conversions
- Perform data trending and historical archiving of sensor data and remote station health status information for each of remote stations.
- Contains software algorithms to support a) automated data polling operation, b) customer requested data polling and c) troubleshooting capabilities.
- Allows offline data mining.

The class diagram in chapter 4 captures these requirements:

**** These characteristics can be embedded at the remote station level if desired.***

Software

- Each module has its unique embedded software algorithms to control their assigned operation.
- The Power Management module contains software algorithms to monitor battery health status and to maximize battery life. The Power Management software controls the "ON/OFF" power cycles of the rest of the modules.
- The Analog Interface module and Embedded knowledge module contain software algorithms specific to the application or sensor technology being monitored.
- The RF Core module contains software algorithms to assure data communication integrity and availability.

The algorithm below and the activity diagram together provide an example of one of the algorithms that could be implemented in the system to meet the requirements of the system [17]

Lost station

- The Network starts in a nominal master/remote (point-to-point) protocol.
- A communication failure is detected between the Central Station and a Remote Station.
- The Central Station commands the remote stations to locate and communicate with the lost station
- The assigned Remote Station establishes communication with the “Lost Station” on a secondary established frequency.
- Information is relay back and forth between the Central Station and the “Lost Station” through the assigned Remote Station.
- The Sensor Network automatically reconfigures from a traditional configuration (point-to-point protocol)

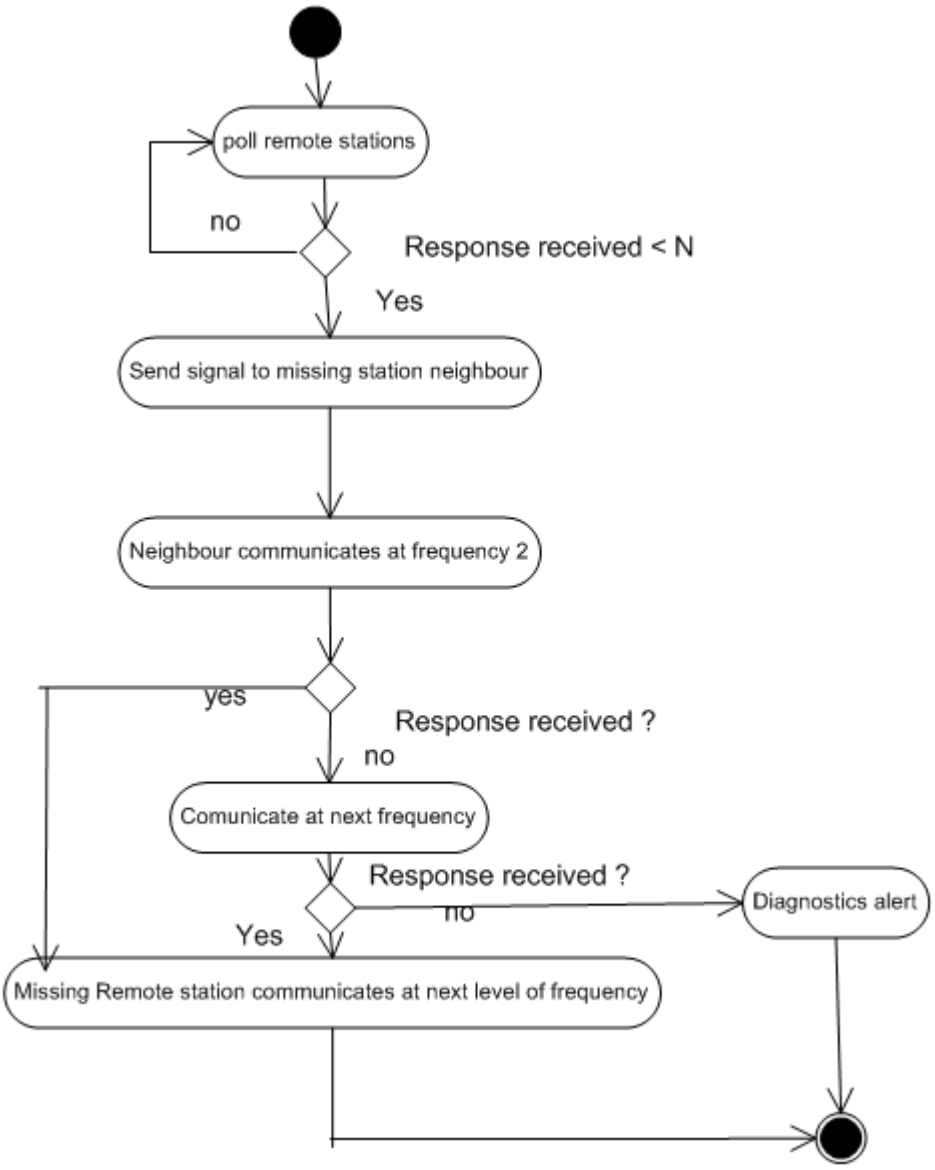


Figure 8 Activity: Lost station protocol

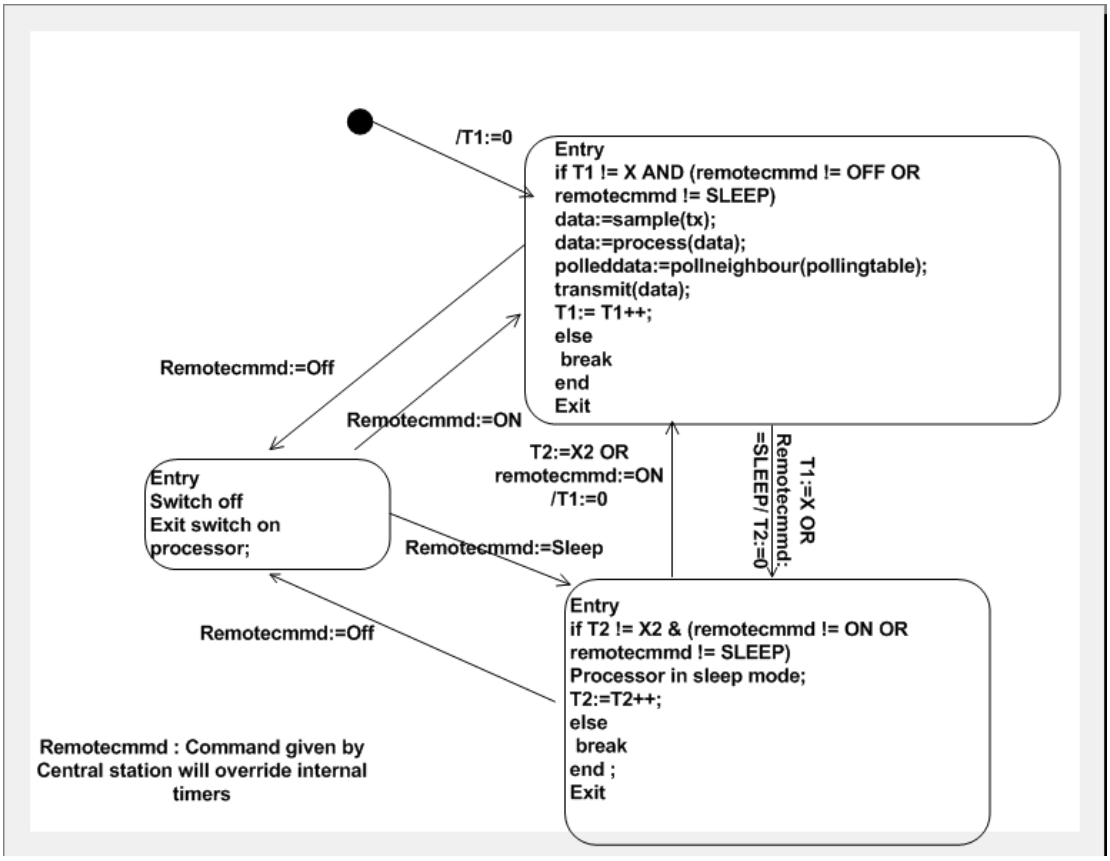


Figure 9 Power Management

3.2.2 Establishing validation scenarios

To see if these requirements are met we design validation scenarios to confirm that the design meets the client needs. The design of these scenarios would be a top down approach and as the requirements are decomposed so would the validation scenarios. Intuitively we can conclude that these components of validation when taken bottom up should combine to form the top level scenario. Such a model would ensure that breaking down of the higher layers into the lower ones would ensure conformity to the architecture when combined the reverse way.

Our premise lends to the formation of a trace between each requirement its corresponding validation scenario which may or may not be decomposed to several

lower level scenarios mapping to lateral layers in lower levels of the requirements structure. Such a tiered structure helps to establish a one to one mapping between the requirements and the validation scenarios.

Requirement # 1: Speed of response – Expected speed of response is targeted to be the lowest

Validation Scenario # 1: This can be done using manufacturer's data as well as a simulation can be carried out to attest if the requirement is indeed met.

Simulation Simulation tests can be carried out to time the response of the system.

Set up a sensor measuring phenomenon(e.g. temperature, pressure etc)

Induce change in phenomenon value (change in temperature) to pre-decided value

Check time taken for sensor to register change in measurement.

Record time

Check time against requirement.

Criteria for passing: Time <= requirement

Criteria for failing: Time exceeds requirement

Requirement # 2: Fault tolerance – The number of fault alarms should be low. An acceptable level is 0.1%. The alarms generated must correspond to untoward events.

Validation Scenario # 2: Chapter 8 of this thesis describes a trade off study that minimizes the false rate subject to constraints and arrives at a value of 0.0615% for the fault rate.

Analysis: The result of the tradeoff study validates this result.

Requirement # 3: Availability – System downtime must be kept to a minimum. An availability of 95% is desired.

Validation # 3: Most manufacturers carry records of equipment, downtime, scheduled maintenance, availability based on historical data. While selecting components of the system; manufacturer data can be consulted to see if the requirement is met.

Documentation: Manufacturer's documents can be used to cross check the requirement

Requirement # 4: Tamper proof- system should be tamper proof and an indication of evident tampering.

Validation Scenario # 4: Manufacturer's documents usually indicate stress tests carried out for tamper proof testing.

Documentation: Manufacturer's documents can be used to cross check the requirement

Requirement # 5: Cost – low cost

Validation Scenario # 5: Costs of individual components can be added together to verify that the overall cost is met. A sample cost analysis test for the card reader system is covered in Section 3.3.3.

Accountability test needs to be done. Figure # 10 shows an example of such a worksheet.

Requirement # 6 Hierarchical order

Validation Scenario # 6: While designing the system a hierarchical order has to be included in the architecture. Such a hierarchy is introduced also by including 2-tiered security measures such as checking in the IS system if user has cleared proper security channels.

Design: Design of the system must include hierarchy, for e.g. Figure 24 IS security

Requirement # 7 IS system must be protected – the Information system should be protected against unauthorized access.

Validation Scenario # 7 Sequence diagram 28 checks for this

Simulation: A simulation can be carried out to attest if this requirement is met.

Deactivate the entry door access system

Employee enters user name and password

Message displayed "Access Denied"

Alarm generated.

Criteria for passing: Message "Access Denied" displayed

Criteria for failing: System allows user to log into the system

Requirement # 8 Sensors must be equipped with diagnostics

Validation Scenario # 8: Manufacturer's documentation carries information regarding in-built diagnostics and their capabilities. Check these to see if they meet the requirement

Documentation: Inspection of the manufacturer's documentation to verify presence of diagnostics.

Requirement # 9 Remote control

Validation Scenario: The presence of a central system that communicates records activities validates this requirement. The system is shown in figure 8.

This system is also found in class diagrams for the system overview.

Design: Design must include the architecture of the central control system with remote abilities.

Requirement # 10 Monitoring software should be highly scalable. It should be able to work for a small retail store, an office building, a warehouse or a multi-building complex. That goal has implications for the way that one designs the monitoring software.

Requirement # 11 Sensor network longevity. Power consumption – Low power consumption and an estimated life of XX hours is desired. Sensor networks that run for 9 months from non-rechargeable power sources would have significant audiences today.

Validation Scenario: Manufacturer's data provides information about the power consumption, batteries, battery life. Also the power management software designed for the sensor will control the modes of the power consumption.

Simulation: To check the power management software a simulation should be carried out to test if the modes are operational and execute appropriately.

Documentation: The documentation provided by the manufacturer should be checked.

Requirement # 12 From both a systems and end-user perspective, it is critical that sensor networks exhibit stable, predictable, and repeatable behavior whenever possible. An unpredictable system is difficult to debug and maintain.

Validation Scenario #12 A check of the fault rate is required which can be checked through the trade off studies. The characteristics of the sensor can be obtained from the manufacturer.

Documentation: A check of the manufacturer documents should be done for this requirement.

Requirement # 13: Sensors and sampling: For our particular applications, the ability to sense light, temperature, infrared, relative humidity, and barometric pressure provide an essential set of useful measurements. The ability to sense additional phenomena, such as acceleration/vibration, weight, chemical vapors, gas concentrations, pH, and noise levels would augment them.

Validation Scenario # 13: Sensors picked for the application should be crosschecked with the list.

Documentation: See all necessary sensors are included

Requirement # 14: Data archiving: Archiving sensor readings for offline data mining and analysis is essential. The reliable offloading of sensor logs to databases in the wired, powered infrastructure is an essential capability. The desire to interactively "drill-down" and explore individual sensors, or a subset of sensors, in near real-time

complement log-based studies. In this mode of operation, the timely delivery of fresh sensor data is key.

Validation Scenario# 14: A Data log system should be included in the system architecture.

Inspection: Presence of an offline system will indicate completion of the requirement.

3.2.3Validation Scenarios for lower level requirements:

- No source of interference to systems being monitored and/or surrounding systems.
- Be a highly reliable system, assure data integrity and availability
- Low RF output power (≤10 mW) to minimize any interferences to surrounding systems.

These three requirements are explained by the following scenario:

Validation Scenario: This is a field test, but at the same time the range of transmission can be verified and appropriate measures taken.

For e.g. a sensor reports if it is in the range of a phenomenon. Assume there are N sensors out of which M are in the interference range with each other(the transmission range is greater than equal to the sensing range). Of the M sensors each S_i will transmit data with bit rate $b(S_i)$. The total data in transit from time T to T+$\partial$where ∂ is the average latency can be expressed as

$$Data=\sum_{i=1}^{M} b(S_i)$$

If this value reaches a certain fraction of the channel capacity congestion will occur. If C_{total} is the channel capacity

$$\text{Data}=\sum_{i=1}^{M} b(S_i) \leq \alpha\, C_{total}$$

where α is the fraction of the capacity dictated by the self interference.

Thus the upper bound on the reporting rate is dictated by channel capacity. On the other hand application specific criteria such as the required accuracy places a lower bound on the reporting rate. The reporting rate should be high enough to satisfy accuracy. At any point in time the number of active sensors should be such that the application specified requirements are met. If in order to meet accuracy requirement $C_{application}$ is the required channel capacity then

$$C_{application} \leq \sum_{i=1}^{M} b(Si) \leq \alpha\, C_{total}$$

i.e. $C_{application} \leq \alpha\, C_{total}$ to support application requirements.

Not all sensors have the same accuracy. Accuracy is a function of both location and quality of information.

Using this analysis and the range of the sensor given by the manufacturer a deployment strategy can be proposed.

Chapter 8 describes a trade off study in which the deployment strategy is varied to get maximum detection while maintaining inter sensor distances and the fault rate minimum.

- Totally portable and self-sustained (power, communication, intelligence).
- Capable of embedded complex data processing.

Validation: Manufacturer's documentation will give specification regarding the size, portability and microprocessor used.

Documentation: Inspection of documentation to see if criteria are met.

- Capable to survive harsh environments (heat, humidity, corrosion, etc).

Validation: Manufacturer's documentation will give specification regarding tests conducted to check the strength of the system. Specifications regarding temperature range, humidity sustainable and corrosion proof etc will be supplied. While selecting sensors compare specifications with desired requirement.

Documentation: Inspection of documentation to see if criteria is met.

- Capable to self-diagnose and automatically reconfigure upon failure detection

Validation: Manufacturer's documentation will give specification. Also a simulation can be carried out to check for requirement conformance.

- Battery-operated system with smart embedded power management algorithms to maximize battery life.

Validation: Manufacturer's specifications are a good starting point. Stress test to see if power is maintained at desired level. Assuming the manufacturers allow one sensor as a sample test piece the following test can e conducted.

Tests:

1. Initially keep sensor "ON" to check for drainage, power fluctuations and performance. Record observations and time to battery drainage.
2. Leave Sensor on "Auto" mode where the internal power management will take over and manage the modes. Observe performance in each mode and record battery drainage.
3. Repeat step 1 for all modes.

4. Check to see if power management software is configurable
5. Repeat step 1 with your own algorithm
6. Repeat steps from 1 through 5 for sensors from multiple manufacturers, compare outputs
7. Select best sensor conforming to power requirements

- Self-contained system with signal conditioning, data acquisition, data manipulation and data transmission capability.

Validation: Manufacturer's specifications are a good starting point. Check to see if specifications conform to desired requirements.

- Modular flexible architecture reconfigurable to accommodate to most sensing technologies**.**

Validation: This refers to the design of the system which is also covered in chapter 4

- Sensor Network composed of one or more Central Stations and a number of remote stations.

Validation: This refers to the design of the system which is also covered in chapter 4

- Central Station:
  - Performs sensor network data collection, storage and distribution to users.

- Contains data polling schedules and ID tables for the remote stations in sensor network.
- Contains remote stations calibrations curves and engineering unit conversions
- Perform data trending and historical archiving of sensor data and remote station health status information for each of remote stations.
- Contains software algorithms to support a) automated data polling operation, b) customer requested data polling and c) troubleshooting capabilities. Allows offline data mining.

Validation: Most manufacturers supply central systems with these capabilities, however also supplied is a configurable unit which can be programmed by the user to include custom functions. Check for the presence of such a unit. This also refers to the design of the system which is covered in chapter 4

3.3 Higher level Requirements for the system of access control:

1. System should prevent unauthorized access into the premises of the building
2. System should allow visitors inside. Should maintain a record of visitors for future use
3. System should use a biometric to confirm identity of person along with access code and identity cards.
4. System should be tamper proof and should give a visual indication along with an alarm to indicate tampering

5. System should be environment resistant

6. System should be easy to maintain and install

7. System should be able to communicate with the Network Management and control system

8. System should have enough capacity to store biometrics locally without increasing storage burden on the central computer.

9. System should prompt user to provide information and display error messages to user when wrong data is entered.

10. System should be highly accurate and reliable.

11. System should track an employee inside the building

These requirements are ambiguous and are not properly quantified. This is typically the case at the beginning of design and these requirements are typically expressed with words like should, may etc. So the requirements are basically expressed in English without any proper quantification. We break down these requirements to arrive at requirements mapping to the system, subcomponent and component levels. This is a top down approach. These lower level requirements should trace up to one or more higher level requirements. If it doesn't then it probably is not required. If a higher level requirement doesn't break down to a lower level requirement the requirement is not being satisfied by the system design being considered. Similarly every requirement should trace to a component in the system.

3.3.1 Requirement Synthesis:

1	The building should be protected from unauthorized access

2	Every Employee will be equipped with an identification card to gain access in the building
3	The card size shall be 2 1/8" by 3 3/8" (standard credit card size).
4	The identification card will be equipped with an identification number that will indicate that the card belongs to the company.
5	It shall be impossible to change or erase the information contained in the card by exposing the card to an electro-magnetic field of any kind, or physically alter the code without destroying the card.
6	The employee will start the access process by swiping the card in the reader
7	The reader will be wall mounted and of dimensions 7 X 4 X 3 and weigh < 5 pounds
8	It should be possible to program the reader. Programming shall be accomplished by means of an integrated 12 key keypad and 16-character LCD display. Employee will key in the access code with the help of the keyboard on the reader.
9	The Card Reader/Memory unit shall be immune to weather, moisture and any environmental hazards. It should typically withstand extreme temperature conditions and moisture levels.
10	Process of Enrollment (Defined as capturing biometric and information of a new user) should take < 40 seconds
11	It shall be housed in a structure of high impact material for complete protection against weather or tampering.
12	It shall be possible to place the associated electronics in a protected location preventing exposure of sensitive components to the elements and preventing tampering or vandalism.
13	During a power failure, the memory unit shall maintain its memory content for a minimum of 72 hours. Restart after power restoration shall be automatic. Reliability should be > 0.95
14	Card will be equipped with a biometric template to indicate person belongs to company.
15	The card will have to be put in a card reader. The card reader will identify the card identification number by comparing with an inbuilt database.
16	The reader database should have a capacity to store at least 800 biometric templates and should have an expandable memory
17	System verification should be completed in < 25 seconds

18	The reader will be equipped with the Door Controls, Tamper Switch and should be able to Lock exits.
19	System will prevent tailgating or piggybacking.
20	The card reader with an integrated scanner will capture a biometric from the person and compare it with the existing template on the card and in the database.
21	An alarm will be generated alerting security and the exit near the access area is blocked if unauthorized the person fails the identification.
22	The system must be cost effective. Budget for employee identification is restricted to 5000 $
23	When inside the building a person will be tracked with the aid of his identification tag when he gains access at any door.

Table 14 Requirement synthesis for Access System

3.3.2 System Test, Verification and Validation

The importance of manufacturing and issuing reliable card products is vital to maintaining a good client relationship. In days past, card quality was overlooked as a minor issue; after all, it was only a low value plastic token that was easily replaced. Today, with increasing card usage and reliance, and the added expense of smart card (chip card) productions, there is no room for complacency. The image and performance of your card is a direct representation of your organization. The system developed should be tested before handing it over to the customer. Essentially it would require that all the requirements are adhered to.

A system test can have a bottom up approach; starting at the unit level (a unit is the smallest whole of a system which cannot be divided further), going up to the module (a module is an aggregation of units), integration testing- the process of bringing together all of the modules that a program comprises for testing purposes) and finally

the system test that comprises of a top down approach to integrating all system components.

System requirements are used to test the system. By developing a test scenario for every requirement, a confirmation can be made if the system adheres to the user requirements.

Requirement #	Test	Accountability	Simulation	Examination
1		X		
2				X
3				X
4	X			
5				X
6				X
7	X			
8			X	X
9				
10				X
11	X			
12	X			
13			X	X
14	X			
15			X	X
16				
17			X	X
18			X	
19			X	
20	X			
21		X		

Table 15 Traceability Matrix

TEST TEMPLATE

REQUIREMENT
TEST TYPE
CRITERIA TO PASS
FAILURE
TOOLS
DOCUMENTS NEEDED
DOCUMENTS AT THE END OF TEST

Table 16 Test Template

3.3.3 Validation Scenarios

1. Every employee will be equipped with an identification card to gain access to the building.

Accountability 1.1: Verify the number of employees with cards available

Check employee name with name on card to verify allocation

Criteria to pass: every employee must have and only one card allocated to him/her.

Failure: Any discrepancy in allocation will result in failure.

Tools: List of names of employees

Personnel required to do the task

Documents at end of test: A validated list of all card numbers and employee names associated with the card

2. The card size shall be 2 1/8" by 3 3/8" (standard credit card size).

Examination 2.1: Match specification provided by physically measuring the size of the card

Criteria to passing: Requirement +/-0.001% tolerance

Failure: Any discrepancy will result in failure.

Documents required: Specification provided by manufacturer

Tools: A scale to measure length, breadth of card.

Software required: Database containing employee and card information.

Personnel required to conduct test

3. It shall be impossible to change or erase the information contained in the card by exposing the card to an electro-magnetic field of any kind, or physically alter the code without destroying the card.

Examination 3.1 Check for compliance with standards

Failure: Non-Conformance

4. The employee will start the access process by swiping the card in the reader

4.1 *Test Steps:*

Setup card reader:

Use power supply as indicated in specification

Power up card reader. Record time to start up (normal startup)

Go through instruction manual and follow setup procedure

On completing setup swipe card

Reader displays message: "Enter identification Number"

Criteria for passing: Reader displays correct message at Step 5

Failure points:

Thickness of card doesn't match thickness of reader slot. Check specifications.

Document thickness of card and report to manufacturer.

Card not recognized by reader

Reader does not display any information.

Setup doesn't complete properly.

Tools: Card, Reader

Document: Incident Report, Setup Manual, contact manufacturer.

5 The reader will be wall mounted and of dimensions 7 X 4 X 3 and weigh < 5 pounds

Examination 5.1:

Weigh the reader.

Criteria for passing: Weight < 5 pounds

Failure: Weight does not match requirement. Document error and report to manufacturer

Documents needed: Specifications

Tools: Weighing scale

Examination 5.2:

Measure dimensions

Criteria for passing: Dimensions=7x4x3(+/-10% tolerance)

Failure: Dimensions do not match requirement. Document error and report to manufacturer

6. It should be possible to program the reader. Programming shall be accomplished by means of an integrated 12 key keypad and 16-character LCD display.

Examination 6.1The reader should be ISO compliant.

Employee will key in the access code with the help of the keyboard on the reader.

7 The identification card will be equipped with an identification number that will indicate that the card belongs to the company.

Test7.1:

Check if card identification follows ANSI/ISO/IEC 7812 standard. [19] Also see Appendix.

Test 7.2

Key in the number in the database

Criteria for passing: A match with the name of the person on the card and employee name in database.

Test Failure: Print out the employee names and corresponding identification number

Case1: Employee name without number: Failure of Test 1

Enter card number against employee number

Store entry

Run search for number again

Result should indicate match between employee name and identification number

Case 2: All Employee names have numbers

Case a: Redundant entry for an employee: Failure of Test 1, Failure of test 2

Check code for "Employee" Table in Database. If error-Redefine constraints.

Document changes.

Case b: No Redundant entry found

Check number of cards against number of employees

Document discrepancy.

Tools needed: Reader, card

Document: Standards, Incident report, Setup Manual

8. The Card Reader/Memory unit shall be immune to weather, moisture and any environmental hazards. It should typically withstand extreme temperature conditions and moisture levels.

Examination 8.1

Check standards for card testing

e.g. According to the DIN ISO 7810 "Identification cards" standard, a minimum bond strength of 6N/cm is required during the T-peel test. The test data of cards conforming to this standard would be made available by the manufacturer.

Failure to conform will lead to the failure of this test.

Document needed: Manufacturer's specification, Standards.

9. Process of Enrollment (Defined as capturing biometric and information of a new user) should take < 40 seconds

Simulation 9.1:

Installation personnel swipes card through the slot in the reader.

Wait for prompt “Enter Code"

Enter code using key board.

Reader should display "Select Program"

Program selected" Enrollment"

Swipe employee card through reader

Simultaneously start a stopwatch

Reader should display "Hold finger close to scanner"

The employee should hold his/her finger near the scanner

Reader should display message "Scanning complete- Press any key to exit"

Stop the stop watch

Record time

Criteria for passing: time recorded< 40 seconds

Failure

If time > 40 seconds error generated inform manufacturer.

Reader does not display correct message at Step 2, 4, 5 or 8

Result	Step 2	step 4	Step 5	Step 8
Fail	0	0	0	1
Fail	0	0	1	0
Fail	0	0	1	1
Fail	0	1	0	0
Fail	0	1	0	1

Fail	0	1	1	0
Fail	0	1	1	1
Fail	1	0	0	0
Fail	1	0	0	1
Fail	1	0	1	0
Fail	1	0	1	1
Fail	1	1	0	0
Fail	1	1	0	1
Fail	1	1	1	1

Table 17 Result Table

Tools: Card, Reader

Documents: Incident Report, Setup Manual

10 It shall be housed in a structure of high impact material for complete protection against weather or tampering. & It shall be possible to place the associated electronics in a protected location preventing exposure of sensitive components to the elements and preventing tampering or vandalism.

10.1 Examination: Check assembly specifications

Refer Standards

10.2 Test: Test assembly with procedure described in Standards.

Failure: Failure to conform to standards

Document: Incident Report, Specifications, Standards

11 During a power failure, the memory unit shall maintain its memory content for a minimum of 72 hours. Restart after power restoration shall be automatic.

Test 11.1: Step1

After reader has been set up, extract a set of ID codes from the list. Use these to test if system retains accurate information regarding owner-number details.

Remove power supply, start watch simultaneously

monitor unit for 72 hours.

At an interval of every hour check memory contents by entering set of numbers for identification.

Also start an access process and an enrollment process

Record output for all commands entered.

Note any discrepancy.

Check for display properties

Criteria to pass test: All the outputs are same.

Failure to deliver same output in all runs indicates a fault. Unclear numbers and an inability to read display messages. Error messages generated by the system. System crashes mid way through cycle test time. Wrong display messages. Check for stored data after an enrollment process by initiating an access process.

Test 11.2:

Repeat above procedure

In the last one hour restore power supply

Record any discrepancies

Test the system again for enrollment and access procedures, key in the identification numbers to check output

Record any discrepancies

Remove power supply again

Midway through the second run i.e. in the 36th hour restore supply.

Repeat steps 3 and 4 and 5.

After another hour of operation remove supply again

Wait for an hour and repeat steps 3 through 5.

Criteria to pass: step 1 successful and All the outputs are same.

Failure to deliver same output in all runs indicates a fault. Unclear numbers and an inability to read display messages. Error messages generated by the system. System crashes mid way through cycle test time. Wrong display messages. Check for stored data after an enrollment process by initiating an access process.

12 Reliability should be > 0.95

Test 12.1: Check for Reliability data from manufacturer

Check for MTBF, MTTR

Failure: Reliability< 0.95

Documents: Specification

13 Card will be equipped with a biometric template to indicate person belongs to company.

Examination13.1:

Biometric strip is located on the card.

Refer to standards by International Biometric group to ascertain the Dimensions of template

Simulation 13.2

Capture Biometric of employee on template through Enrollment process

Swipe card

Reader displays message: “Enter code"

Enter access code using keyboard

Reader displays message: "Hold Finger near scanner"

Hold your finger in front of the scanner

Reader displays message: “Enter"

Criteria for passing: Examination 13.1 & Simulation 13.2 both are passed.

Failure:

Reader doesn't recognize template

Examination test fails.

14 The card reader will identify the card identification number by comparing with an inbuilt database.

Test 14.1:

*Swipe car*d A through the reader

Swipe another card (any card that has not been configured)

Criteria for passing:

The reader should display message "Enter code" in case of card A

The reader should display "Invalid Card" for card B

Failure:

Result	Card A	Card b
Fail	0	1
Fail	1	0
Fail	1	1

Table 18 Result Table

15 The reader database should have a capacity to store at least 800 biometric templates and should have an expandable memory

Examination 15.1:

Check specifications given by manufacturer

Check for memory configuration

Criteria for passing: Memory capacity=> 800 and should be expandable

Failure: Capacity<800 and memory is not expandable

Documents: Incident Report, Specifications

16 System verification should be completed in < 25 seconds

Simulation 16.1:

Swipe card

Start Stopwatch

Reader displays message: “enter code"

Enter access code using keyboard

Reader displays message: "Hold Finger near scanner"

Hold your finger in front of the scanner

Reader displays message: “Enter"

Stop stopwatch

Record time

Criteria of passing: Time recorded < 25 secs

Failure:

Time recorded> 25 seconds

Reader does not display “enter code" message at Step 3

Reader does not display "Hold Finger near scanner" message at Step 5

Reader does not display "Enter" message at Step 7

Key 0 indicates wrong message; 1 indicates right message

Result	Step 3	step 5	Step 7
Fail	0	0	1
Fail	0	1	0
Fail	0	1	1
Fail	1	0	0
Fail	1	1	0
Fail	1	0	1
Fail	1	1	1

Table 19 Matrix

17 The reader will be equipped with the Door Controls, Tamper Switch and should be able to Lock exits.

Examination17.1

Specification check

Criteria for passing: Conformance to requirement

Failure: Fail to conform to requirement

18 System will prevent tailgating or piggybacking.

Simulation18.1:

Swipe card

Reader displays message: “enter code"

Enter access code using keyboard

Reader displays message: "Hold Finger near scanner"

Hold your finger in front of the scanner

Reader displays message: “Enter"

Door lock opens, enter through door

Second person follows immediately before door closes

Alarm sounds; all exits closed

Criteria for passing: Alarm sounds

Failure: Test fails if alarm doesn't sound indicating that the detector doesn't record second movement. Document error report to manufacturer

19 The card reader with an integrated scanner will capture a biometric from the person and compare it with the existing template on the card and in the database.

Simulation 19.1: Step 1

Swipe card

Reader displays message: “enter code"

Enter access code using keyboard

Reader displays message: "Hold Finger near scanner"

Hold your finger in front of the scanner

Reader displays message: “Enter"

Door lock opens, enter through door.

Criteria for passing: Reader successfully displays message 6 for the right biometric

Simulation 19.2:

Swipe any card different from the person bearing it.

Reader displays message: “enter code"

Enter access code for the card using keyboard

Reader displays message: "Hold Finger near scanner"

Hold your finger in front of the scanner

Reader displays message: “Type Mismatch"

Door locks do not open.

Criteria for passing: Simulation 1 successful and Simulation 2: reader displays “Type Mismatch" in step 6

Result	Card A	Card b
Fail	displays "Enter"	does not display "Type Mismatch"/random error
Fail	displays "Type Mismatch"	Displays "Type Mismatch"
Fail	displays " Type Mismatch"	Displays "Enter"
Fail	displays " Enter"	Displays "Enter"

Table 20 Result

20 An alarm will be generated alerting security and the exit near the access area is blocked if unauthorized the person fails the identification.

Simulation 20.1:

Swipe card

Reader displays message: “enter code"

Enter incorrect access code using keyboard. Set Count=1

Reader displays message: "Invalid entry try again"

Enter incorrect access code using keyboard. Set Count=2

Reader displays message: "Invalid entry try again"

Enter incorrect access code using keyboard. Set Count=3

Reader displays message: "Authorization failure"

Alarm sounds

All exits should be locked

Criteria for passing: Reader displays correct message at step 3, 5 and 8

Failure:

Test fails if

Result	Step 4	step 6	Step 8
Fail	0	0	1
Fail	0	1	0
Fail	0	1	1
Fail	1	0	0
Fail	1	1	0
Fail	1	0	1
Fail	1	1	1

Table 21 test

Key: 0 indicates wrong message i.e.

step 4 Reader displays message “Enter"

step 6 Reader displays message “Enter"

step 8 Reader displays message “Enter"; 1 indicates right message as indicated in steps

2. Alarm fails to sound

3. Any or all of the exit fails to lock

21 The system must be cost effective. Budget for employee identification is restricted to $500,000.

Accountability 21.1:The cost of the system is not the capital expenditure you put upfront but the cost that will be incurred over the entire lifetime of the system and these include licensing costs, maintenance etc. So to meet the cost constraint all these costs should be met.

Fixed Cost	*Alt A*	
Cost of Reader + card	$	3,150.00
Power Supply	$	1,600.00
Communication System	$	600.00
Central system		
Hardware	$	15,000.00
Software	$	20,000.00
Personnel	$	12,328.77
Recurring Cost	*$/year*	
Maintenance	$	12,000.00
Upgrade	$	1,000.00
Staff Cost	$	375,000.00

	Year				
Security System	*0*	*1*	*2*	*3*	*4*
Benefits	$400,000.00	$400,000.00	$400,000.00	$400,000.00	$400,000.00
PV of benefits	$50,000.00	$363,636.36	$330,578.51	$300,525.92	$273,205.38
NPV of all Benefits	$50,000.00	$413,636.36	$744,214.88	$1,044,740.80	$1,317,946.18
Costs					
Fixed	$52,678.77	$0.00	$0.00	$0.00	$0.00
Recurring	$388,000.00	$388,000.00	$35,000.00	$35,000.00	$35,000.00
Total	$440,678.77	$388,000.00	$35,000.00	$35,000.00	$35,000.00
PV of Costs	$440,678.77	$352,727.27	$28,925.62	$26,296.02	$23,905.47
NPV of all Costs	$440,678.77	$793,406.04	$822,331.66	$848,627.68	$872,533.15
Yearly Cash Flow	($390,678.77)	$10,909.09	$301,652.89	$274,229.90	$249,299.91
Overall NPV	($390,678.77)	($379,769.68)	($78,116.78)	$196,113.12	$445,413.03
Overall ROI	0.510482645				

Figure 10 Worksheet

Chapter 4: System Architecture and Access Graphs

4.1 Introduction

Class diagrams represent the static structure of the classes and their relationships (e.g., inheritance, aggregation) in a system. It does not indicate how they interact to achieve particular behavior. Every piece of behavior which is required of the system must be provided by objects of the classes. A good class model consists of classes which represent enduring classes of domain objects which don't depend on a particular functionality required today.

4.2 System architecture

The Figure below is an abstraction of the system structure and gives the various components comprising it. Each class/component is dealt with individually in the following sections.

We now describe the system architecture using Figure 10, functionality of individual components and how they operate together. Included in the Figure is the HVAC class which models the air supply of the building. This is important as we need to prevent the building from a chemical or biological attack and the HVAC system is the first weak link in the architecture against any such attack. Any agents introduced through this system will be circulated throughout the buildings.

The IS system is the information system of the building, connecting the computers, laptops, network and the access system. In line with the requirements the IS system

has a built-in protection of security levels. An unauthorized access in the system will be detected either at the card reader level or it will be detected at the IS system.

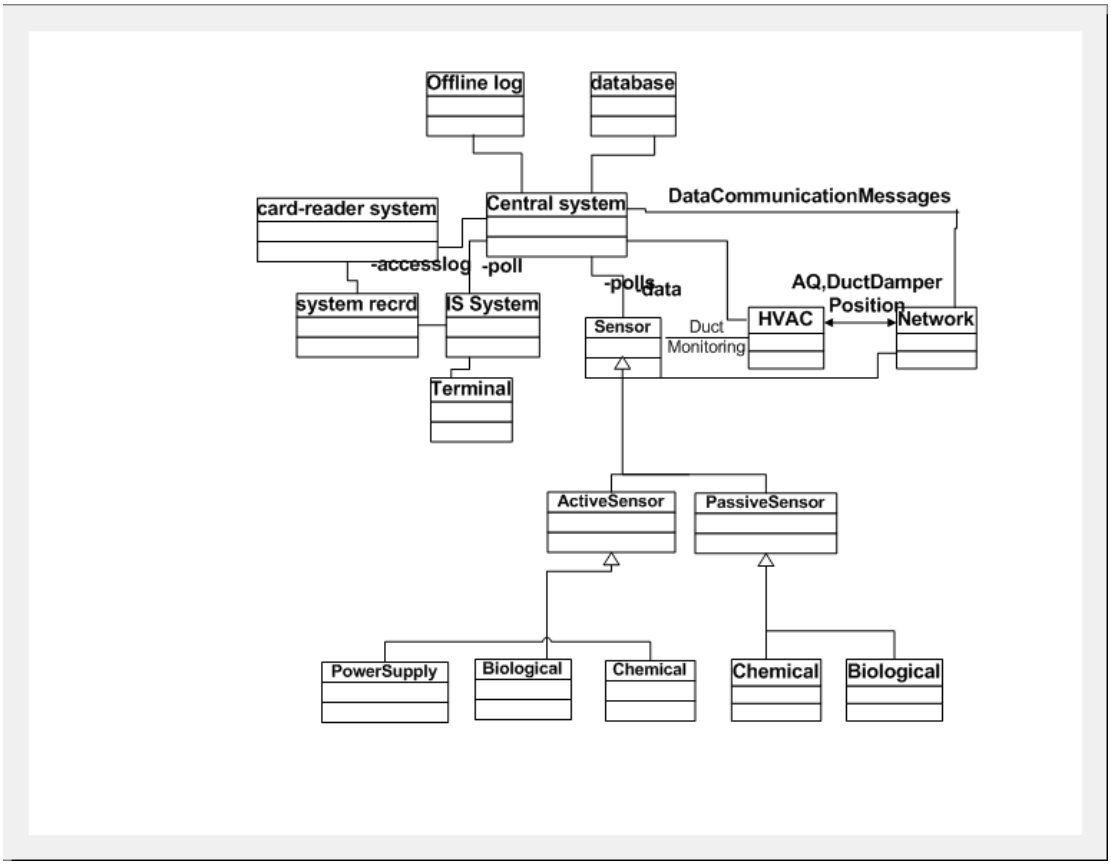


Figure 11 system structure

In the following sections we explain how the architecture addresses the requirements set forth in Chapter 3. We developed a tiered architecture. The lowest level consists of the sensor nodes that perform general purpose computing and networking in addition to application-specific sensing. The sensor nodes may be deployed in dense patches that are widely separated. The central station connects to a database as well as an offline logging system. At the central station, the data is displayed to employees through a user interface. The full architecture is depicted in Figure 11.

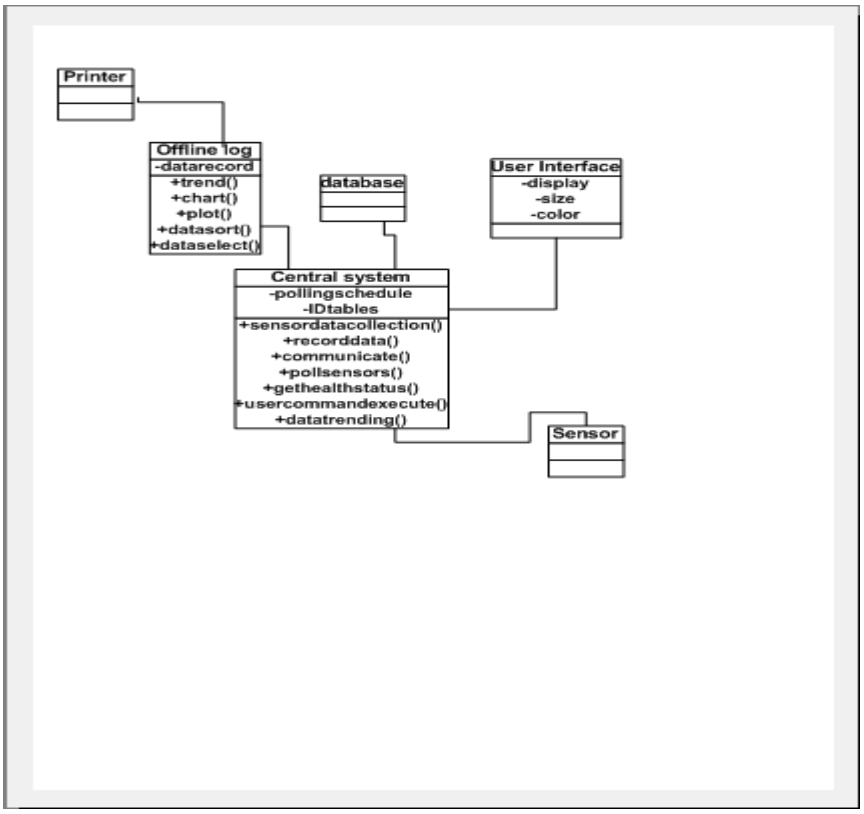


Figure 12 Central system

The lowest level of the sensing application is provided by autonomous sensor nodes. These small, battery-powered devices are placed in areas of interest. Each sensor node collects environmental data primarily about its immediate surroundings. Because it is placed close to the phenomenon of interest, the sensors can often be built using small and inexpensive individual sensors. High spatial resolution can be achieved through dense deployment of sensor nodes. Compared with traditional approaches, which use a few high quality sensors with sophisticated signal processing, this architecture providing collaboration with other sensors measuring different phenomena; provide higher robustness against occlusions and component failures.

A computational module in the sensor is a programmable unit (Refer section 4.2) that provides computation, storage, and bidirectional communication with other nodes in the system. The computational module performs basic signal processing (e.g., simple translations based on calibration data or threshold filters), and dispatches the data according to the application's needs. Compared with traditional data logging systems, networked sensors offer two major advantages: they can be retasked in the field and they can easily communicate with the rest of the system.

Ultimately, data from each sensor needs to be propagated to the central station. The propagated data may be raw, filtered, or processed data. Bringing direct wide area connectivity to each sensor path is not feasible – the equipment is too costly, it requires too much power and the installation of all required equipment is quite intrusive to the environment. The base station may communicate with the sensor patch using a wireless local area network.

The components must be reliable, enclosed in environmentally protected housing, and provided with adequate power, as per Requirement # 4.

The architecture needs to address the possibility of disconnection at every level (also refer "lost station: activity diagram). Each layer (sensor nodes, clusters, central stations) has some persistent storage which protects against data loss in case of power outage. Each layer also provides data management services. At the sensor level, these will be quite primitive, taking the form of A/D conversions, amplification, signal conditioning etc. (refer section 4.5)

Remote control of the network as per Requirement # 2 is also provided through the central system. Typically useful when it is required for the autonomous sensors to be controlled through a single point. More of sensors are dealt with in the section Sensor class.

The next section examines the remote station (sensor node) architecture.

4.2 Remote Architecture explored

4.2.1 Remote station architecture:

Based on the requirements listed in chapter 3 and section 4.1 we propose an architecture for the remote station. In this section we consider the requirements for the remote stations [17]

- Design should be modular in nature.
- Number and types of modules will depend on specific application.
- Basic configuration starts with a RF Core Module:
  - RF Core Module contains a RF Transceiver and a dedicated micro-controller.
  - Connector interfaces are established for all modules for compatibility and data connectivity.
- A second module (analog module) normally provides sensor interface capability:

  - Sensor excitation, signal conditioning, signal amplification and signal (A/D) conversion.
  - Tailored for the specific application or sensor to be instrumented.
- A third module normally provides power management functions to the Remote Station:
  - Battery monitoring and health status.
  - Specific algorithms to control “power on/off” cycles for all other modules.
  - Controlled by a very low power dedicated micro-controller.
  - This is a generic module in each Remote Station.
- A fourth module provides embedded knowledge capability to the Remote Station:
  - A DSP contained in this module performs higher mathematical functions, statistical analysis more complex reasoning.
  - This module is also an application specific module.

The Figure below illustrates the modules needed to meet these requirements:
As can be seen all the requirements are met in this design and hence it forms a framework for any further development. To cross check if the system indeed satisfies the requirements, all designs that contain these modules can be referenced.

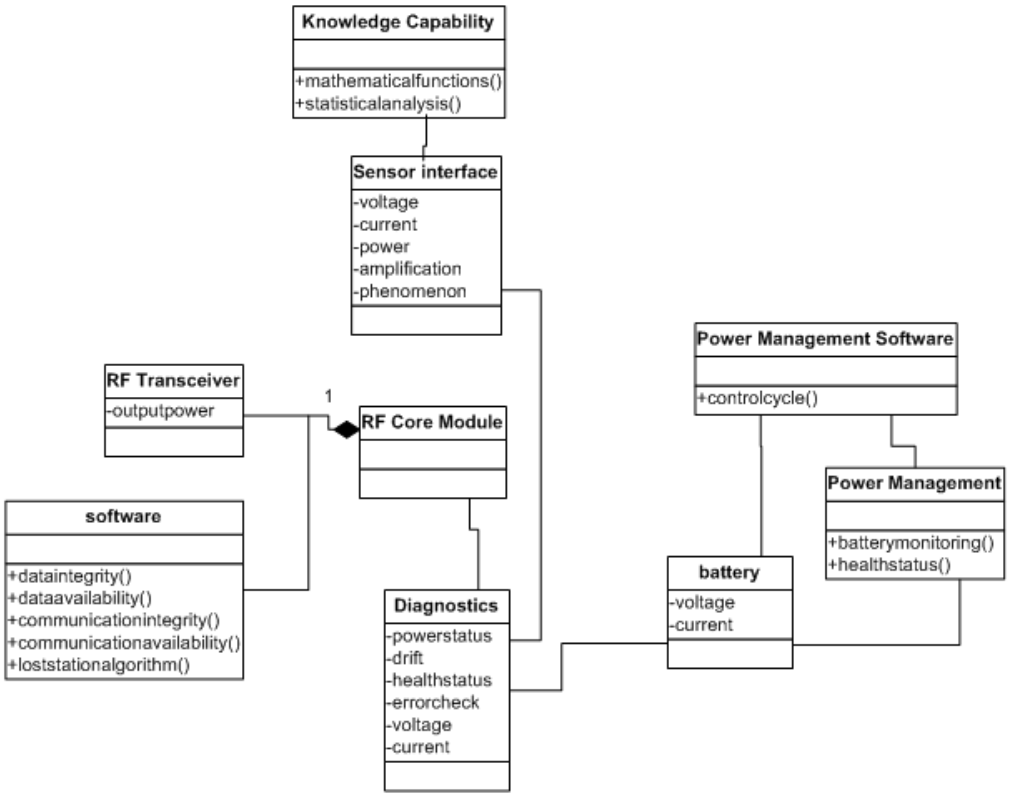


Figure 13 Remote Station architecture

All of the components in the system must operate in accordance with the system's power budget. In a running system, the energy budget must be divided amongst several system services:

- sensor sampling,
- data collection,
- routing and communication,
- health monitoring and
- network retasking.

Environment monitoring applications may need other important services in addition to those mentioned in this section. These services include localization, time synchronization, and self configuration. [20]

Data sampling and collection

In environment monitoring the ultimate goal is data collection; sampling rates and precision of measurements are often dictated by external specifications. For every sensor we can bind the cost of taking a single sample. By analyzing the requirements we can place a bound on the energy spent on data acquisition. We trade the cost of data processing and compression against the cost of data transmission. We can estimate the energy required by data collection by analyzing data collected from indoor monitoring networks.

Communications

Power efficient communication paradigms for environment monitoring must include a set of routing algorithms, media access algorithms, and managed hardware access. The routing algorithms must be tailored for efficient network communication while maintaining connectivity when required to source or relay packets.

Network Retasking

As the researchers refine the experiment, it may be necessary to adjust the functionality of individual nodes. This refinement can take several different forms. Scalar parameters, like duty cycle or sampling rates, may be adjusted through the application manager. Most of the time such updates can be encapsulated in network maintenance packets

Health and Status Monitoring

A major component of use to the application is one that monitors the sensor's health and the health of neighboring sensors. Health and monitoring is essential for a variety of purposes; the most obvious is retasking. Although the health messages are not critical for correct application execution, their use can be seen as preventive maintenance. For this reason, we advocate a health and monitoring component that transmits status messages with lower latency in exchange for strict reliability. Health messages may be sent rather infrequently (about once per hour or less dependent on the duty cycle) with no guarantee on their delivery.

The Figure below shows the software class that will address the requirements stated in this section.

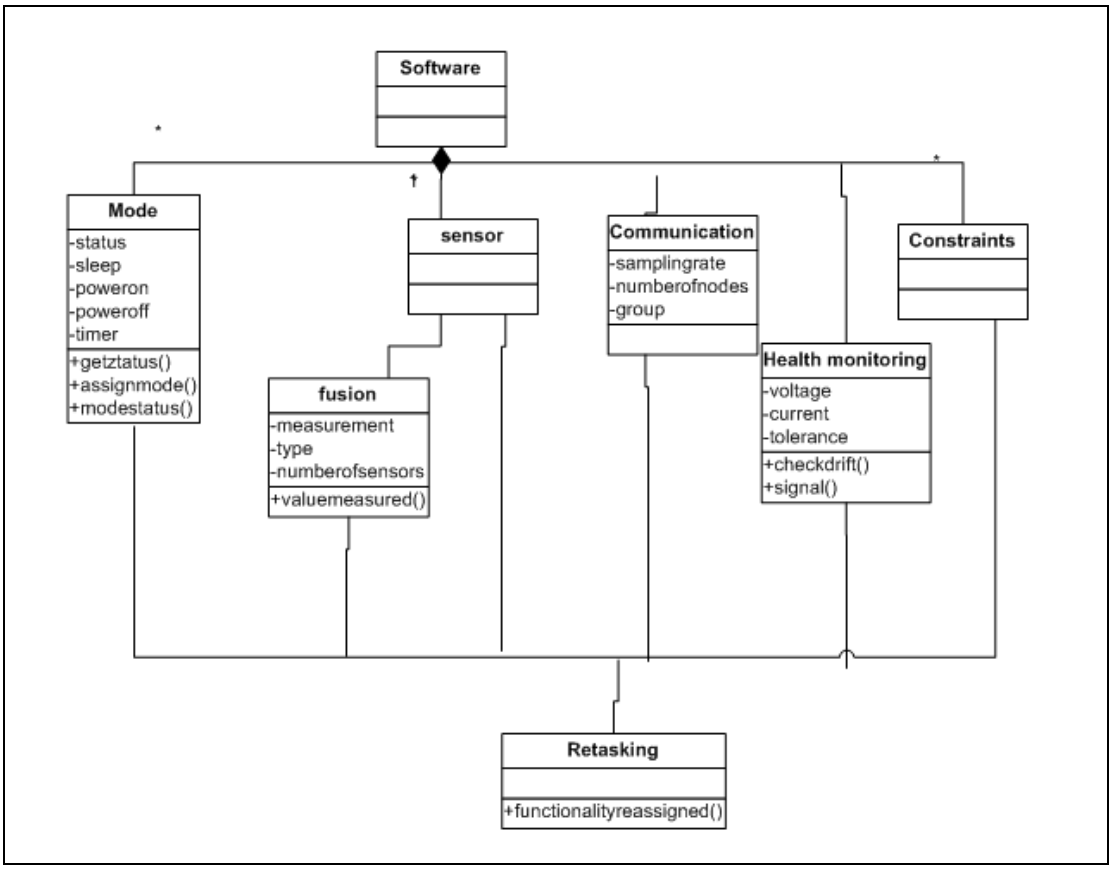


Figure 14 Software class

A sensor when mounted has some constraints viz operational e.g. whether the range is adequate to allow uninterrupted monitoring, distance from adjacent sensor; for e.g. if an optical sensor is mounted such that it is obstructed by an opaque surface the sensor should be able to diagnose this error. The other constraint being physical wherein the location may be a problem and the sensor might not fit into the area drawn out for it. What this class illustrates is that upon installation the sensor will execute logic to see if all its constraints are executed before it is ready for operation.

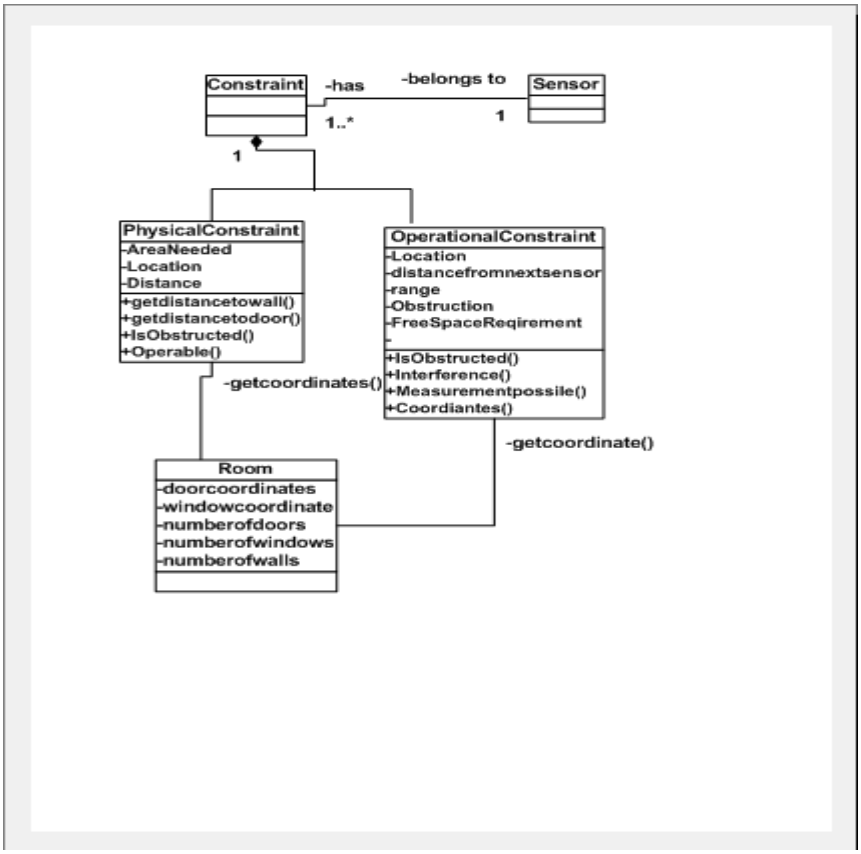


Figure 15 Constraints class

This section covers fully all the requirements stated in chapter 3 for sensors.

4.3 Building

An *intelligent building* can be viewed as a system consisting of many individual entities, either “passive” such as doors, windows, furniture, or “intelligent”, i.e. capable of computing, holding and communicating their state, such as computer terminals, mobile communicators, sensors, actuators etc. These objects are fixed or mobile. However there is a line dividing the structure of the building and the intelligence as it sits on this structure. In the buildings class we concentrate merely on

the structural part of the building. The first two sections comprising the central and the remote architecture and the sections to follow cover the “intelligent” entities.

The Interior of the building can be thought of as having a Structure, Functionality, and Other Systems. While the exterior consists of Surrounding, Environment

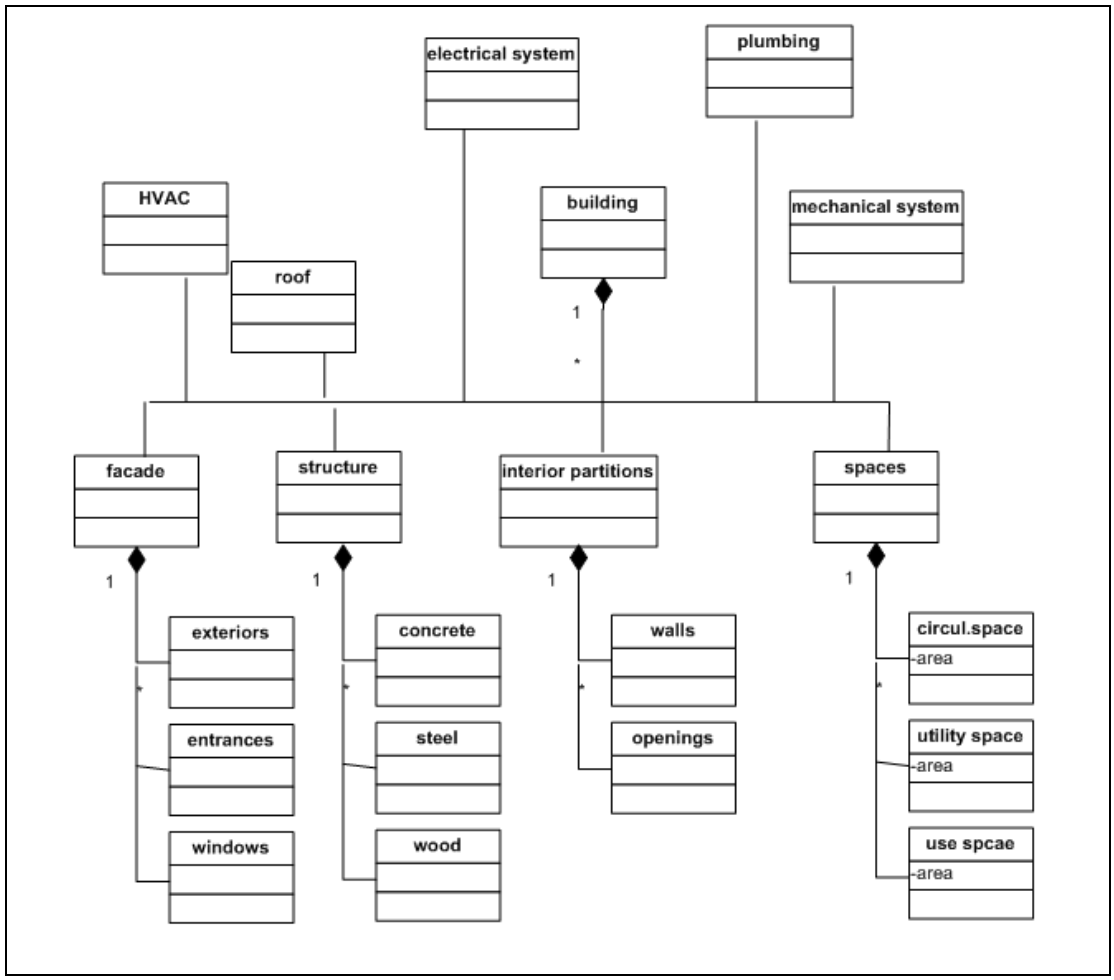


Figure 16 Building explored

Figure 16 above roughly decomposes a building class. Another attempt to get a macro view of the interior of the building containing aspects of interest and dedicated solely to the building is illustrated in the Figure 17 below

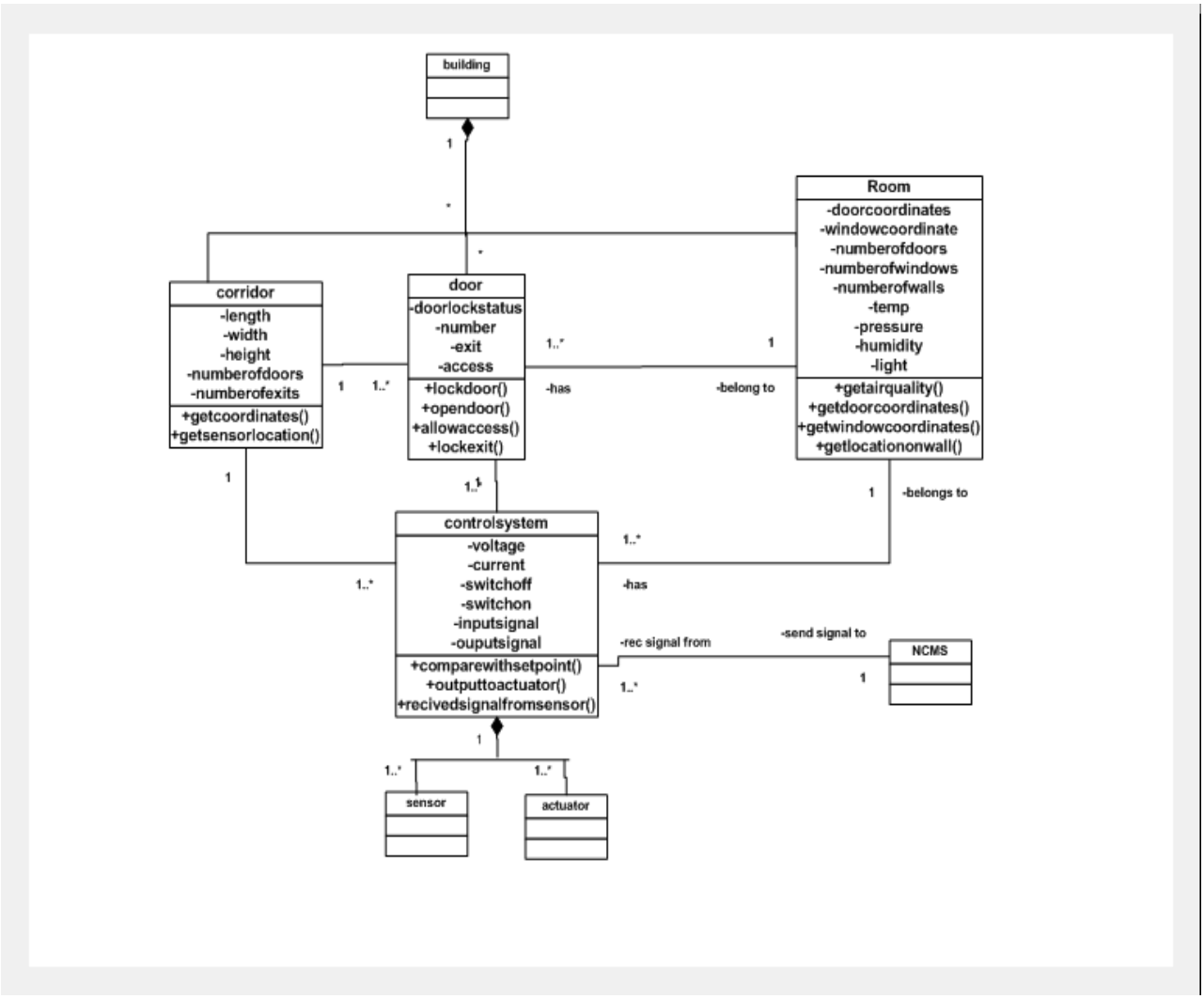


Figure 17 Building Class

The class- Building is a key domain abstraction; the domain being the building that needs to be protected. It includes the area that needs to be protected along with the rooms and corridors. A control system is included to monitor the room continuously.

4.3.1 Access Graphs

Consider the floor plan shown in figure 18 below

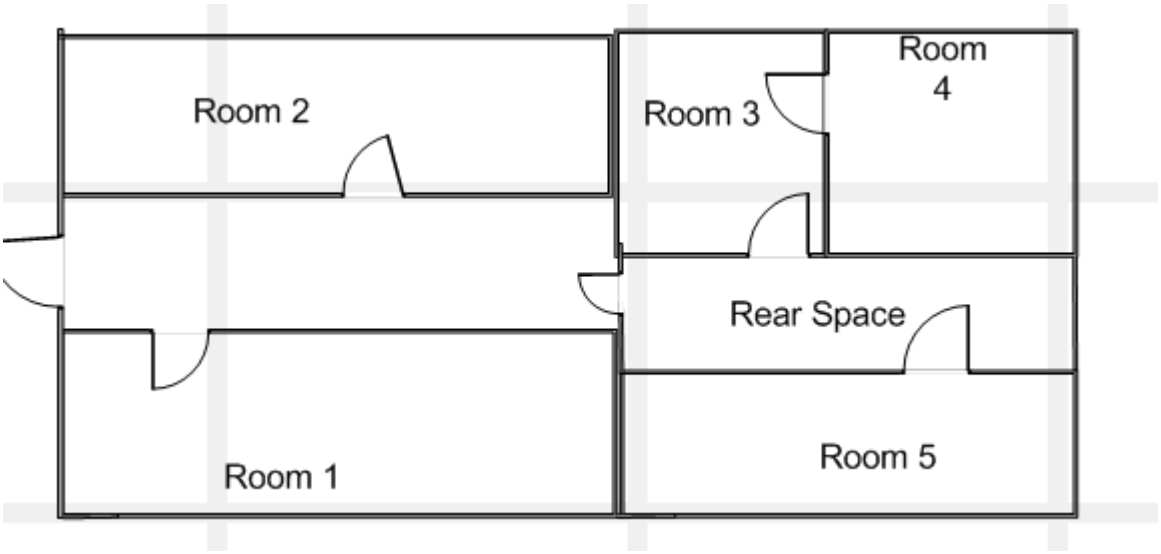


Figure 18 Floor plan

Note: The lines indicated in the graph are not to scale and are non-directional. They are merely links between spaces.

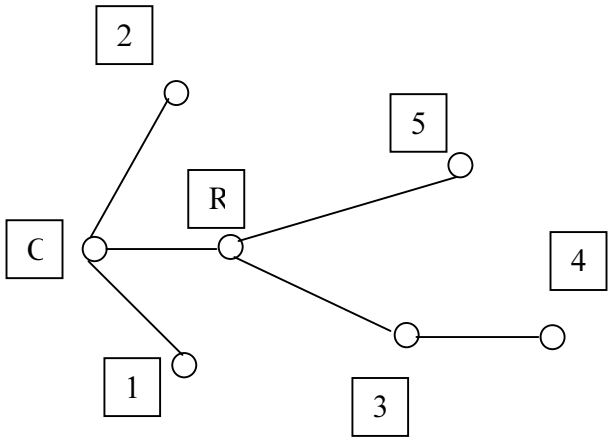


Figure 19 Access Graph

The floor plan consists of a layout comprising of rooms and doors. Rooms in the layout can be thought of as nodes while doors between the rooms are lines connecting the nodes. According to graph theory, rooms are vertices and doors are edges. The graph of a structure's plan can be represented in another form – as an adjacency

matrix **A** with entries *a*ij. The rows and columns of the matrix represent vertices, while the entries in the body of the matrix represent whether or not the vertices have edges between them. The matrix is square, with *n* rows and columns, where *n* is the number of vertices. The entries in the matrix are of course symmetric about the diagonal – if room 1 is connected to room 2, room 2 is connected to room 1. Rooms – vertices – are not connected to themselves, so the diagonal entries are 0.

The formal representation provided by the graph in Figure 19 makes it easier to define different aspects of the pattern of connection among rooms. The portrayal of the connections on the graph in the adjacency matrix facilitates computing quantitative measures of the pattern displayed by the graph. Aspects of the pattern for particular rooms that are important include depth, connectivity, control value, and integration. These measures can be used to describe relationships between rooms as they are envisioned on a floor plan.

The graph in Figure 19 above gives the connectivity with rooms as nodes/vertices and doors as edges justified so that the carrier is at the entrance. A justified access graph requires choosing a door from which the interior in initially entered. The door becomes an edge, connecting a hypothesized exterior vertex to the vertex representing first room entered. The exterior vertex is called “the carrier” in space-syntax; in this case marked as C in the Figure.

Figure 19 above indicates:

Once an entrance is made from the exterior (C), one can enter Room 1 and Room 2 directly; one has access to Rear passage (1) directly. One has to follow the path Rear-Room 5/Room3 to get to either rooms and finally the path Rear-Room3-Room4 to get to Room 4. Access to Room 4 is completely controlled by Room 3.

The justified access graph makes it clear that some rooms are more accessible than others with rooms of equal depth being equally accessible. For example the Room 2 (depth=1) is more accessible than Room 4 (depth=3).

	Room						
	R	1	2	3	4	5	C
R	0	0	0	1	0	1	1
1	0	0	0	0	0	0	1
2	0	0	0	0	0	0	1
3	1	0	0	0	1	0	0
4	0	0	0	1	0	0	0
5	1	0	0	0	0	0	0
C	1	1	1	0	0	0	0

Table 22 Adjacency Matrix

4.3.2 Characteristics of the graph:

Connectivity

Connectivity of a single room is simply the number of doors into it. Summing the rows or columns of the adjacency matrix yields connectivity values for the rooms:

$$c_{ij} = \sum_{j=1}^{n} a_{ij}$$

Connectivity of a room can also be thought of at the number of routes into or out of that room from or to an adjacent room.

Control

Control is a measure of the extent to which a given room controls access to the rooms that are adjacent (immediately connected by a door) to it. Consider, as an example, two rooms, *A* and *B*, connected by a door. We are interested in the control value for *A*. If the only entry into *B* is the connection to *A*, then *A* controls access to *B* entirely. On the other hand, if *B* has connections to other rooms, in addition to *A*, then *A* has less control over access to *B*. In general, control for a room in inversely proportional to the connectivity of the adjacent rooms. The formula is:

$$ctrl_i = \sum_{j=1}^{n} a_{ij} * (1/c_{ij})$$

In other words, control for the i'th room can be computed by multiplying its adjacency vector – the row of 0's and '1's in the adjacency matrix – by the reciprocal of the connectivity values for all the rooms and summing the products. The products for rooms that are directly connected will equal the connectivity reciprocals, while they will equal 0 for those rooms that are not connected. The sum of products is therefore the sum of the connectivity reciprocals for the connected rooms.

Room	*Depth*	*Mean Depth*	*Connectivity*	*1/c*	*ctr*
R	1	1.5	3	0.333333	1.833333
1	1	2.5	1	1	0.333333
2	1	2.5	1	1	0.333333
3	2	2	2	0.5	1.333333
4	3	2.833333333	1	1	0.5
5	2	2.333333333	1	1	0.333333
C	1	1.666666667	3	0.333333	2.333333

Table 23 characteristics of an access graph

Depth: The accessibility of rooms within a structure to individuals entering from outside is determined by drawing a justified access graph. Rooms are assigned depths relative to the carrier, which are the number of edges (doors) crossed as one travels along the shortest path from the carrier to the room.

Of course many buildings have multiple exterior entries. Here there are at least two analytical options. In the first, we connect multiple exterior doors to a single exterior carrier. In the second, we draw different justified graphs, one for each entry.

	Room						
	R	1	2	3	4	5	C
R	0	2	2	1	2	1	1
1	2	0	2	3	4	3	1
2	2	2	0	3	4	3	1
3	1	3	3	0	1	2	2
4	2	4	4	1	0	3	3
5	1	3	3	2	3	0	2
C	1	1	1	2	3	2	0

Table 24 Depth Matrix

To extend this simple floor plan to account for an entire building consisting of floors, elevators, rooms etc; the following diagram representation is used as a starting point. In the access graph we primarily define rooms (vertices) and doors (edges). The relationship between the rooms is quantified by:

1. the depth; number of doors one would pass to reach a room
2. control; whether one room controls access to another
3. connectivity; how well is a room connected

we now extend this to a more generalized representation.

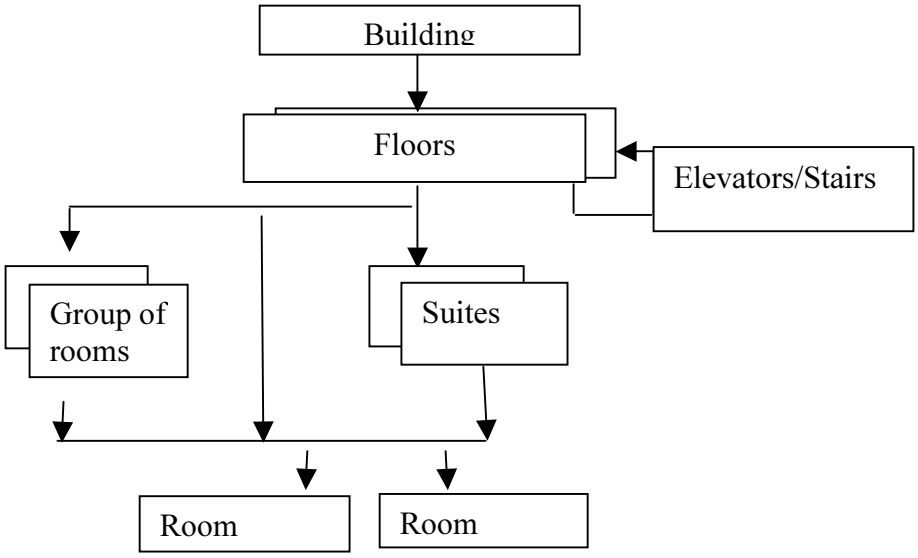


Figure 20 Space Hierarchy

At the root of the space hierarchy is the building, which can be thought of as consisting of several floors. The floors which form the next level of the hierarchy are interconnected by elevators or staircases. Below the floors are the groupings of the rooms and finally at the bottom are the individual rooms.

A building consists of several rooms arranged relative to each other. The chief components of such a layout would be the Class Room describing the types of rooms, the number of any individual room type, etc. The exact relations between the rooms will be characterized by a relationship class Relation. Broadly the attributes of the relationship class can be classified as:

1. Distance: specifies the distance between the two rooms. Depth, mean depth (depth of a room/ (total number of rooms-1)), travel time can be used to characterize this relationship.
2. Control: Access to one room may be completely controlled by another, e.g. Room 4 in Figure 1 is completely controlled by Room 3.
3. Connectivity: Quantifies if the room is well connected or not. In the example above the rear area has very high connectivity indicating that it has to be monitored more than say Room 4.
4. is visible: a Boolean variable that will indicate the visibility relation between two rooms; For e.g. a value of 1 could indicate that an office is visible to a storage room
5. Inside: states whether a room is inside another; for e.g. a value of one indicates that a storage room is inside an office.
6. Connects to: =N; states that one room connects to a maximum of N of type 2 rooms. E.g. a conference room connects to 1 office.
7. Separated: would indicate that a certain class of rooms would be always separated from another class.
8. Connectionthroughdoor: Specifies if the rooms connect through a door. Could be a Boolean variable that can take a value of 1 or 0.
9. mintraveltime: could specify the minimum time taken to reach a room an inviolable constraint
10. maxtraveltime: could specify the minimum time taken to reach a room an inviolable constraint

11. Connectedthroughducts: specifies if one room connects to another through a common duct. This is important in the case of a chemical or biological attack as it will help to trace the path of the agent.

A floor can be considered as a collection of the rooms while maintaining the relationships between the rooms and without violating attributes like area, maximum number.

We start with the first classification that of the class room which is a basic description of what rooms should have and the basic type of rooms contained in the building. The parent class room has attributes like length, width, height, number occupancy etc. These are inherited by the subclasses and each has its own function as a distinguishing factor.

Certain attributes are used to characterize features of the room such as the maximum number of a particular type of room that can be there in a building (maxnumber) for e.g. the class office has the maxnumber attribute=4 implying only 4 offices can exist.

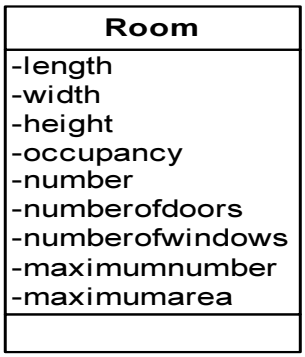


Figure 21 Room class

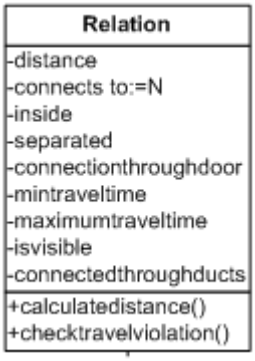


Figure 22 Relationship Class

A room is defined as a super class consisting of various subtypes based on functionality such as office, storage etc.

The relation between two rooms is specified through the relationship class. This class captures the above mentioned features that characterize the relationship between two rooms; whether they are adjacent, separated, connected via ducts etc. In doing this the quantitative measures described in section 1 can be used to explain relationships between different rooms.

When a graph of all the rooms is created weighting each edge with the travel time between the rooms-each relationship can be checked. All rooms of a particular class must satisfy a relationship with another class. E.g. Certain checks include if two rooms with a given depth are placed closer to each other, or the area occupied condition is violated. While placing any room simple guidelines can be deciphered it's proximity to another room or an exit. These cardinalities can be used to classify the strength of the relationships between the rooms

Rooms which are the lowest level of the space hierarchy can be grouped together to form suites, floors etc. Groups of rooms can be clubbed together and this can help to partition major areas like floors of the building. A group in the diagram will represent more than one room that should be located close together in a building. Figure 12 describes one such scenario where it might be required to place 2 offices always connected to a cafeteria and a conference room -forming a floor cluster.

Floor Cluster:

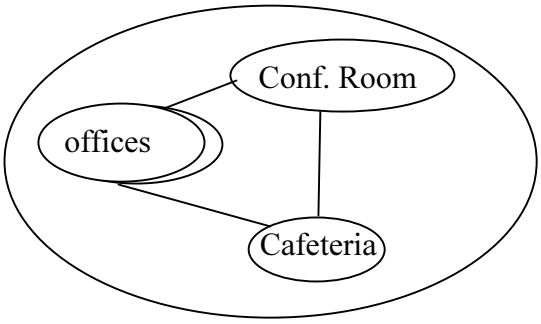


Figure 23 Floor Cluster

The building class consists of several floors. The number of floors can be set by defining an attribute constraint in the class floor. The floors are connected to each other using an elevator or a staircase. Each floor has many rooms as well as groups of rooms.

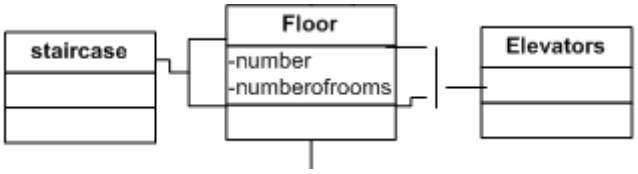


Figure 24 Floor Class

Expanding further to include systems such as HVAC systems that provide ventilation through ducts which become important in the wake of a chemical/biological attack.

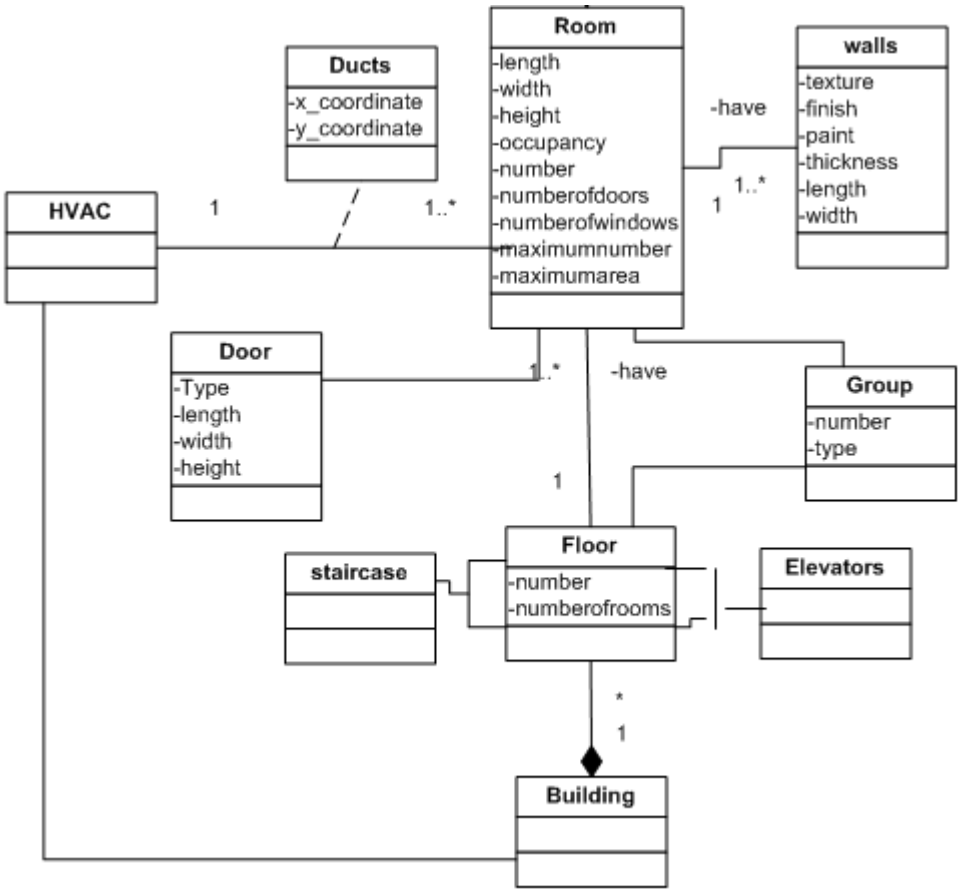


Figure 25 Hierarchy

The figure below is the entire class diagram obtained from the discussion

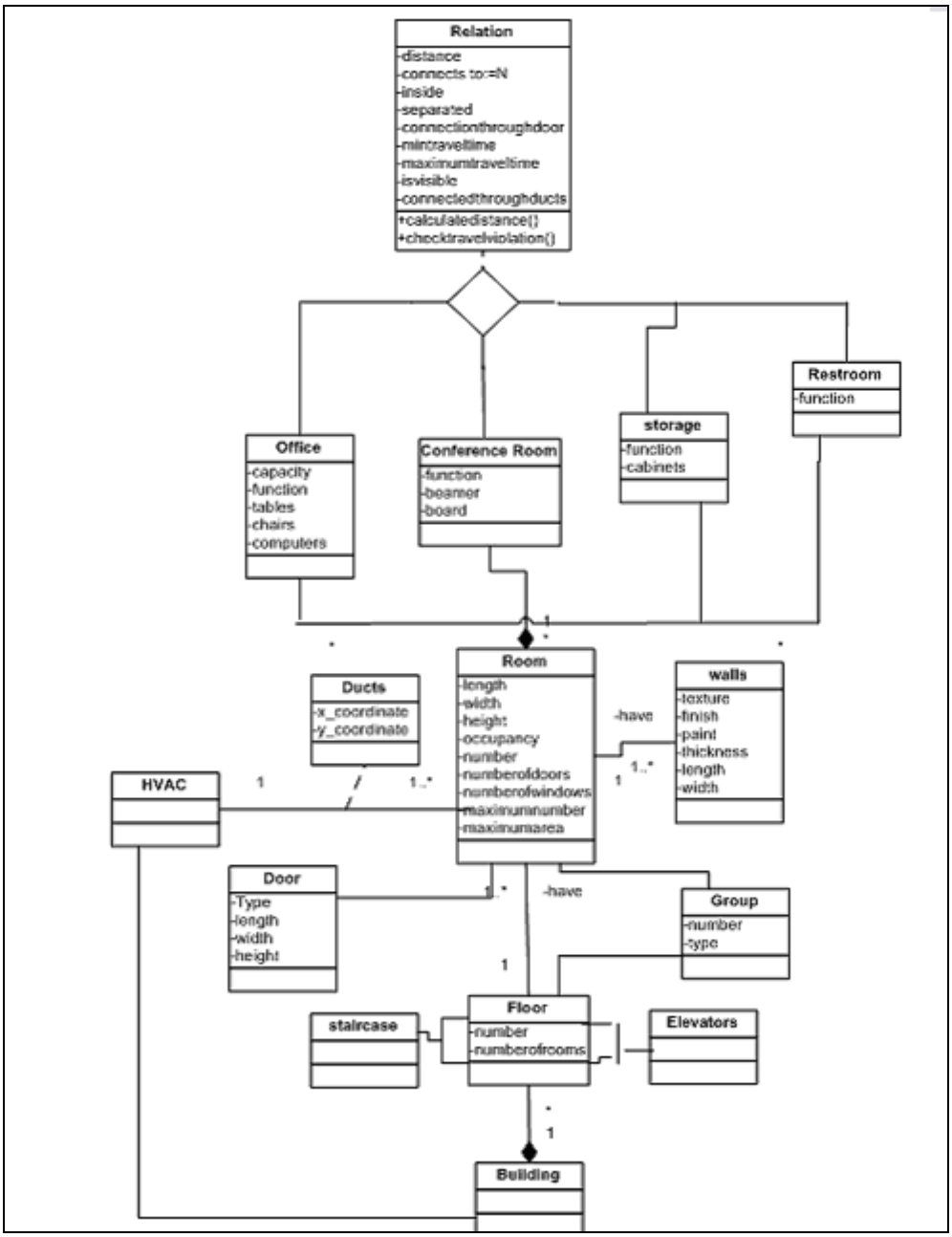


Figure 26 Class

Floor plans with loops (Doors depiction to indicate presence of doors in the layout, does not indicate status of the door):

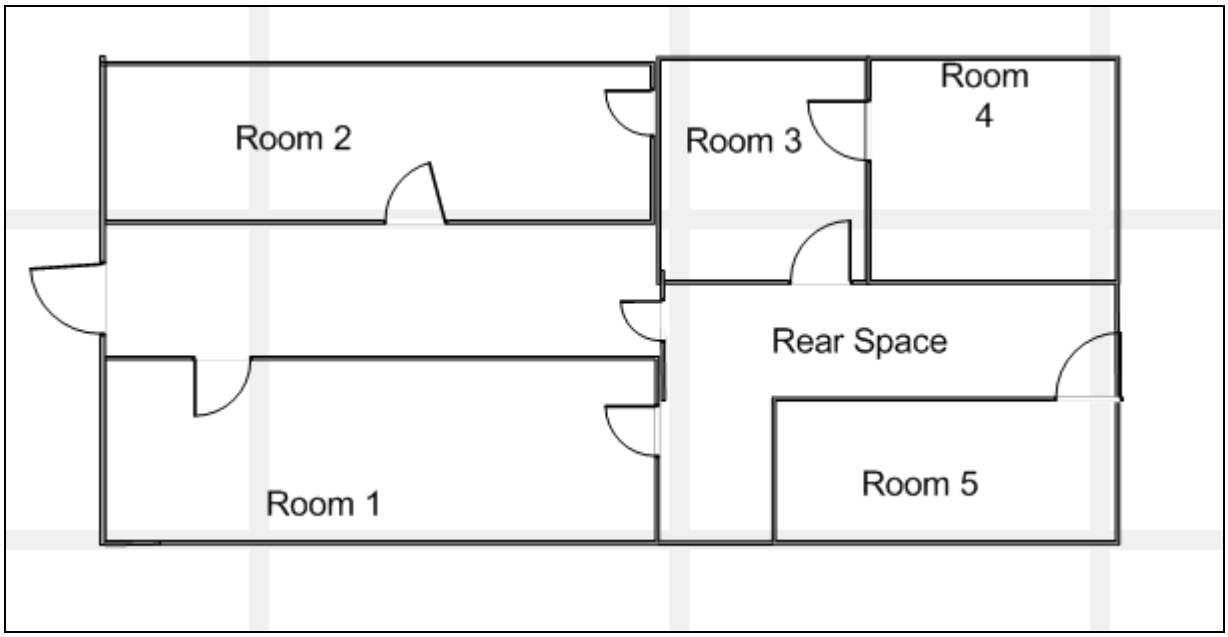


Figure 27 Floor plan with loops

Note: links between nodes are not to scale and are not directional.

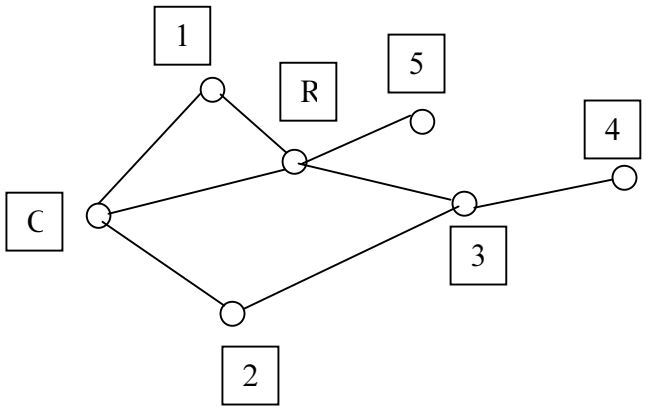


Figure 28 Access Graph

Room							
	R	1	2	3	4	5	C
R	0	1	0	1	0	1	1
1	1	0	0	0	0	0	1
2	0	0	0	1	0	0	1
3	1	0	1	0	1	0	0
4	0	0	0	1	0	0	0
5	1	0	0	0	0	0	0
C	1	1	1	0	0	0	0

Table 25 Adjacency matrix

Room	Depth	Mean Dep	Connectivity	1/c	ctr
R	1	1.333333	4	0.25	2.166667
1	1	1.833333	2	0.5	0.583333
2	1	1.833333	2	0.5	0.666667
3	2	1.5	3	0.333333	1.75
4	3	2.333333	1	1	0.333333
5	2	2.166667	1	1	0.25
C	1	1.666667	3	0.333333	1.25

Table 26 Quantitative measures

Room							
	R	1	2	3	4	5	C
R	0	1	2	1	2	1	1
1	1	0	2	2	3	2	1
2	2	2	0	1	2	3	1
3	1	2	1	0	1	2	2
4	2	3	2	1	0	3	3
5	1	2	3	2	3	0	2
C	1	1	1	2	3	2	0

Table 27 Depth matrix

The changed values in depth are shown in the Table 27. Table 26 gives changes in the other measures. In line with our definitions of connectivity the value for the rooms 1 and 2 have changed from their previous values to account for the increase in connectivity due to addition of doors. Rear Space now has the maximum connectivity and should be the area that needs to be monitored the most. Similarly, mean depth and control also change to correspond to the increased connectivity. The depth matrix

shows the increased routes in and out of rooms and increased connectivity between rooms. The exercise was a simple extension to the earlier example as doors introduced into the floor plans merely represent links between rooms which weren't present earlier on.

4.4 HVAC system class

An HVAC system has to be especially guarded and monitored as it is a potential target for biological/chemical attack since it circulates air supply of the building.

HVAC system and access graphs

The first step for effective detection is isolating the source. Abnormal sensor readings, changes in room air quality as compared to other rooms etc are some of the measures that can be adopted to indicate the "source" of the agent.

Once the potential source has been identified in the building; the focus would be on containing and isolating the threat.

There are two ways a chemical/biological agent could spread in a building.

- Doors: the connectivity of a room to other rooms can be used to identify which rooms are under immediate threat owing to the presence of the agent.
- HVAC: the agent can spread through the duct and into other rooms connected via the duct network.

These two components together constitute the directional graph and can be used to contain damage when a threat is detected. Consider the Figure below:

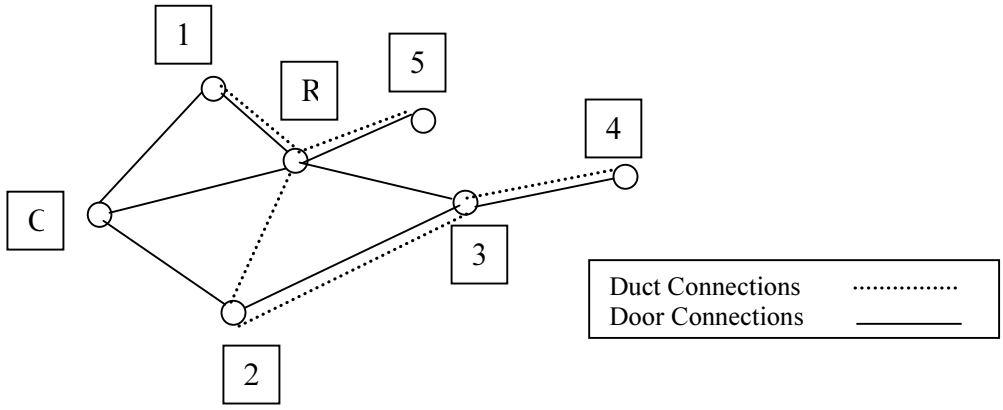


Figure 29 HVAC and Access Graph

If a chemical/biological agent is present in Room 3 from Figure 29 and a corresponding connectivity matrix it is observed that the rear space(R), Room 2 and Room 4 are under immediate threat. Once this has been established an effective control can be designed to protect these rooms. The duct and door connectivity together provide an effective framework for the development of a detection/classification algorithm.

Sensing Bio/Chemical substances

In the "normal" operation, the air in a building will be continuously filtered in a passive mode so any chemical or biological agent is captured as soon as it arrives at the filters. Continuous filtration has the additional benefit of providing a clean background for sensors. The fastest sensors are those that simply detect the presence of biomass without being able to distinguish whether that biomass is a bio warfare agent or a naturally occurring substance such as skin cells, pollen, or mold.

Since the filters normally maintain a low background level of biomass in the air, any sudden rise in internal concentration, such as might accompany a biological attack, is suspicious and sufficient to switch the building into a "precautionary" mode. In this

mode, techniques that are not appropriate for full-time, continuous operation might be used. For instance, high-power ultraviolet lamps may be turned on to kill any bioagent in the return air ducts even though these would not be used continuously because of concerns about operational cost. Another example would be to monitor air in the alternate ducts and switch supply to those ducts if contamination is observed in the operational ducts.

If the building is not under attack, it returns to normal mode without the building occupants ever having been disturbed, an important consideration given the performance of today's sensors. These modes of operations are illustrated by the sequence diagrams and activity diagrams. [21]

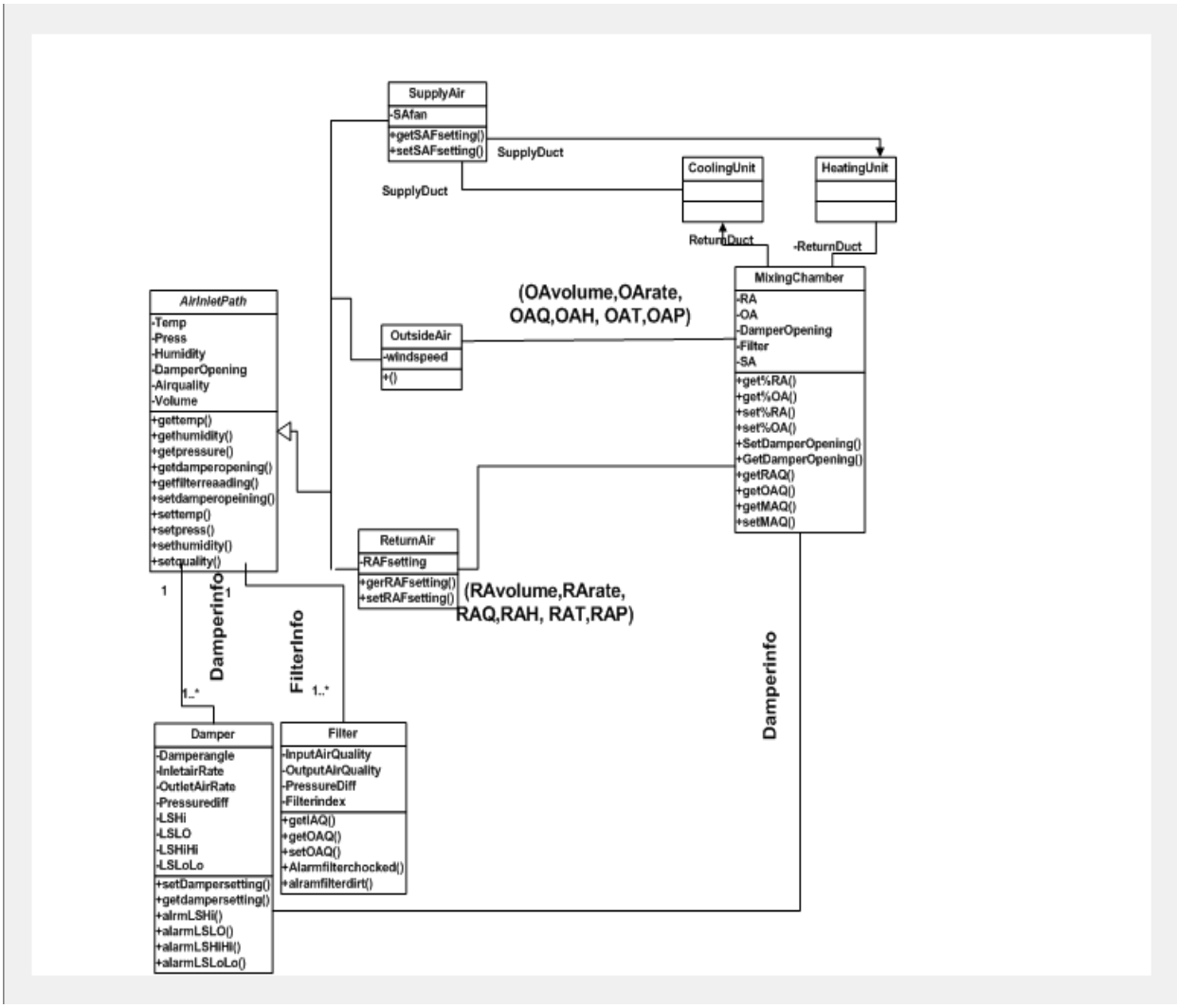


Figure 30 HVAC

4.3.1 Actuator Class

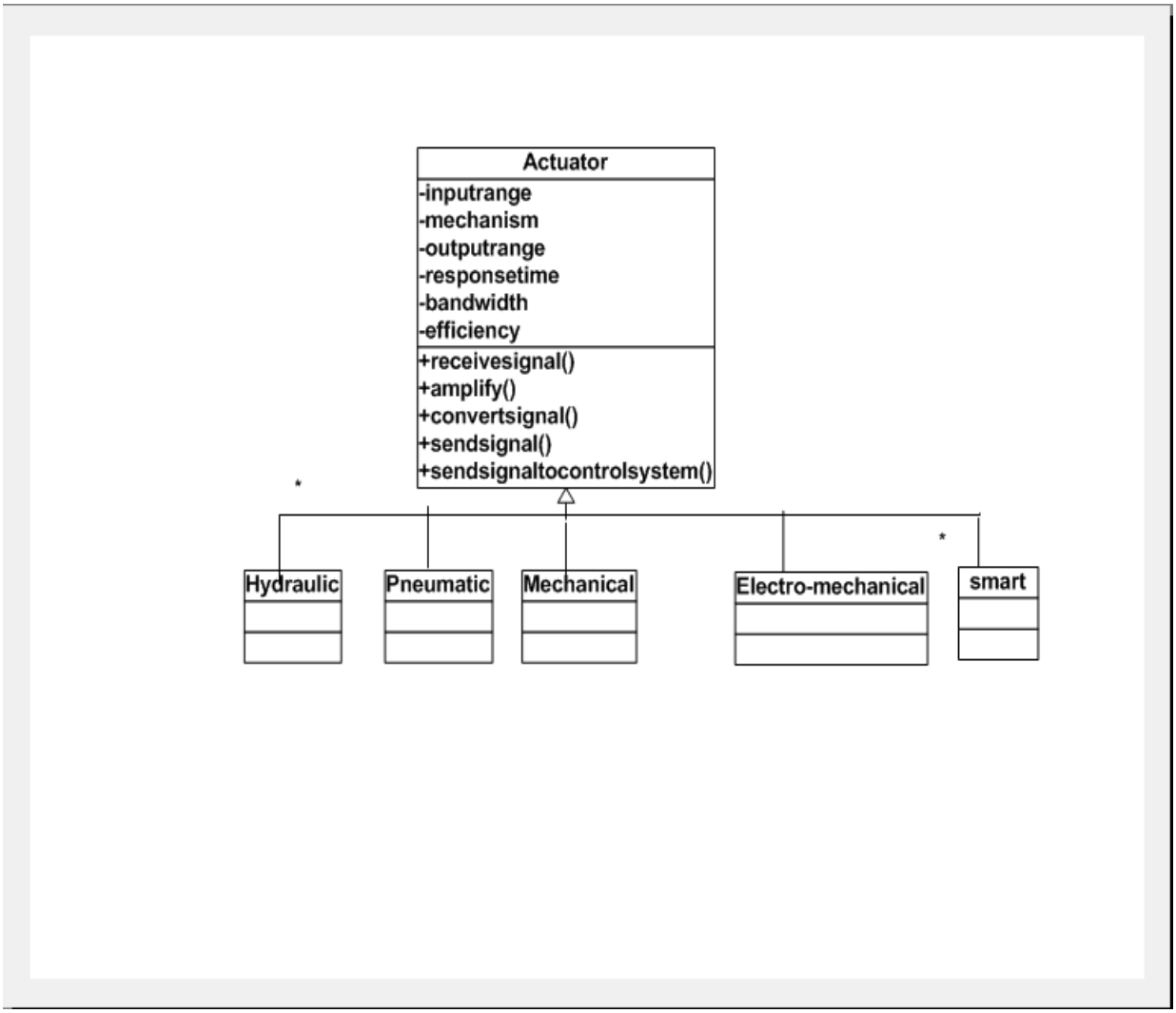


Figure 31 Actuator Class

An actuator is an end device that translates the control logic of the sensor and central control unit together. They are of various types as indicated in the Figure above. These actuators form an integral part of any control system logic implemented.

4.5 Sensor Class

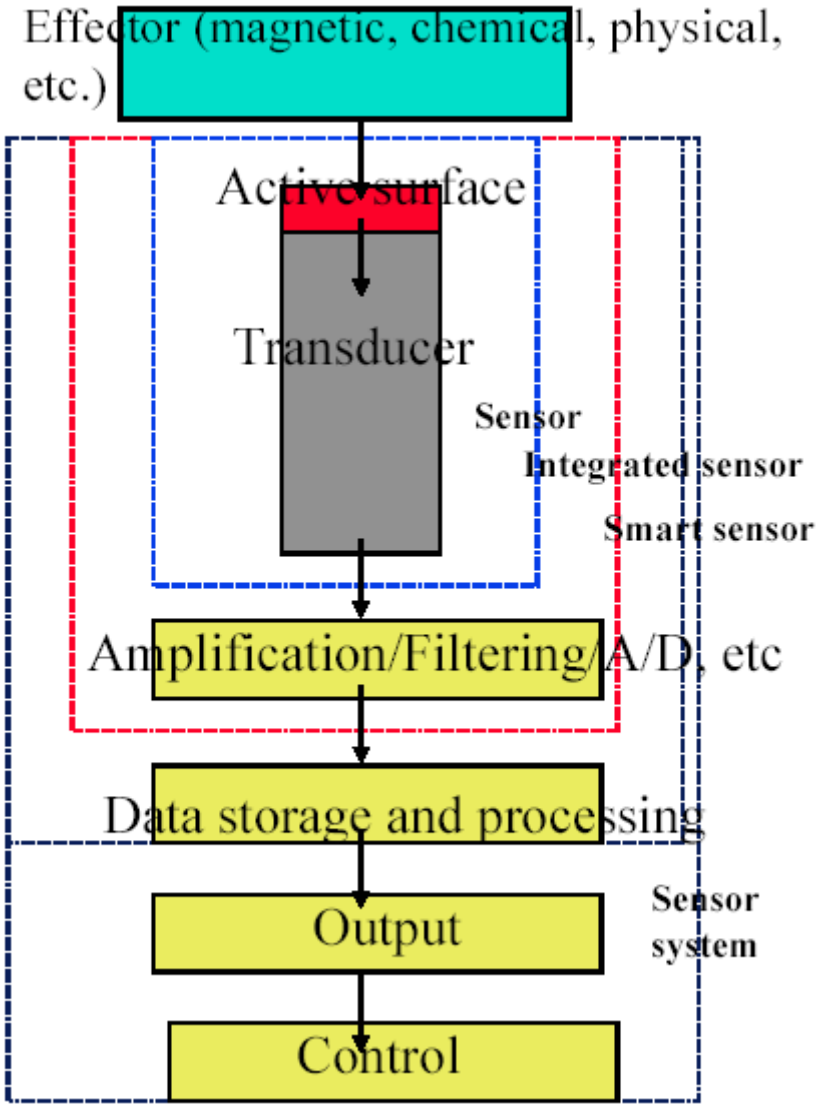


Figure 32 Components of a sensor

Senor as a system can be classified into the following broad categories:

1. Sensing Mechanism
    i. Range
    ii. Sensitivity
    iii. Accuracy
    iv. Resolution

v. Mechanism(Electrical, Mechanical, etc)

vi. Linearity

vii. Hysterisis

viii. Backlash

ix. Scaling

x. Input Value

xi. Output Value

xii. Precision

2. Amplification/Filtering, A/D conversion etc

i. Input value

ii. Output value

iii. Frequency

iv. Amplitude

v. Analog

vi. Digital

vii. Gain

viii. A/D

3. Data Storage and Processing

i. Last measurement

ii. Sampling frequency

iii. Moving average

iv. Set Point

v. Diagnostic(Set Point comparison)

vi. Signal output

4. Output(communication)
    i. Signal
    ii. Signal_to_Alarm
5. External interfaces
    i. Power supply
6. Physical
    i. Assembly
    ii. Location

The Figure below explains the relation between the phenomena measured and the output of the sensor. [22]

	Output (Secondary Signal)					
Input (Primary Signal)	**Mechanical (Mechano-)**	**Thermal (Thermo-)**	**Electrical (Electro-)**	**Magnetic (Magneto-)**	**Radiant (Photo- or Radio-)**	**Chemic (Chemo**
Mechanical	Acoustics Fluidics	Friction calorimeter Cooling effects	Piezo electricity Piezo resistivity	Piezo magnetic effect	Photo-elasticity	
Thermal	Thermal expansion Bimetallic strip		Pyroelectricity Seebeck effect		Radiant emission	Reaction
Electrical	Piezoelectricity Electrometer	Joule heating, Peltier effect	Langmuir probe		Electro-luminescence	Electroly
Magnetic	Magneto-striction Magnetometer	Thermo-magnetic	Magneto resistance			
Radiant	Radiation pressure	Thermopile Bolometer	Photo-electric Dember			Photo-reactions
Chemical	Hygrometer	Calorimeter Thermal conductivity	Amperometry Flame ionization Volta effect	Nuclear magnetic resonance	Chemi-luminescence	

Figure 33 Physical/Chemical Phenomena employed in Sensor

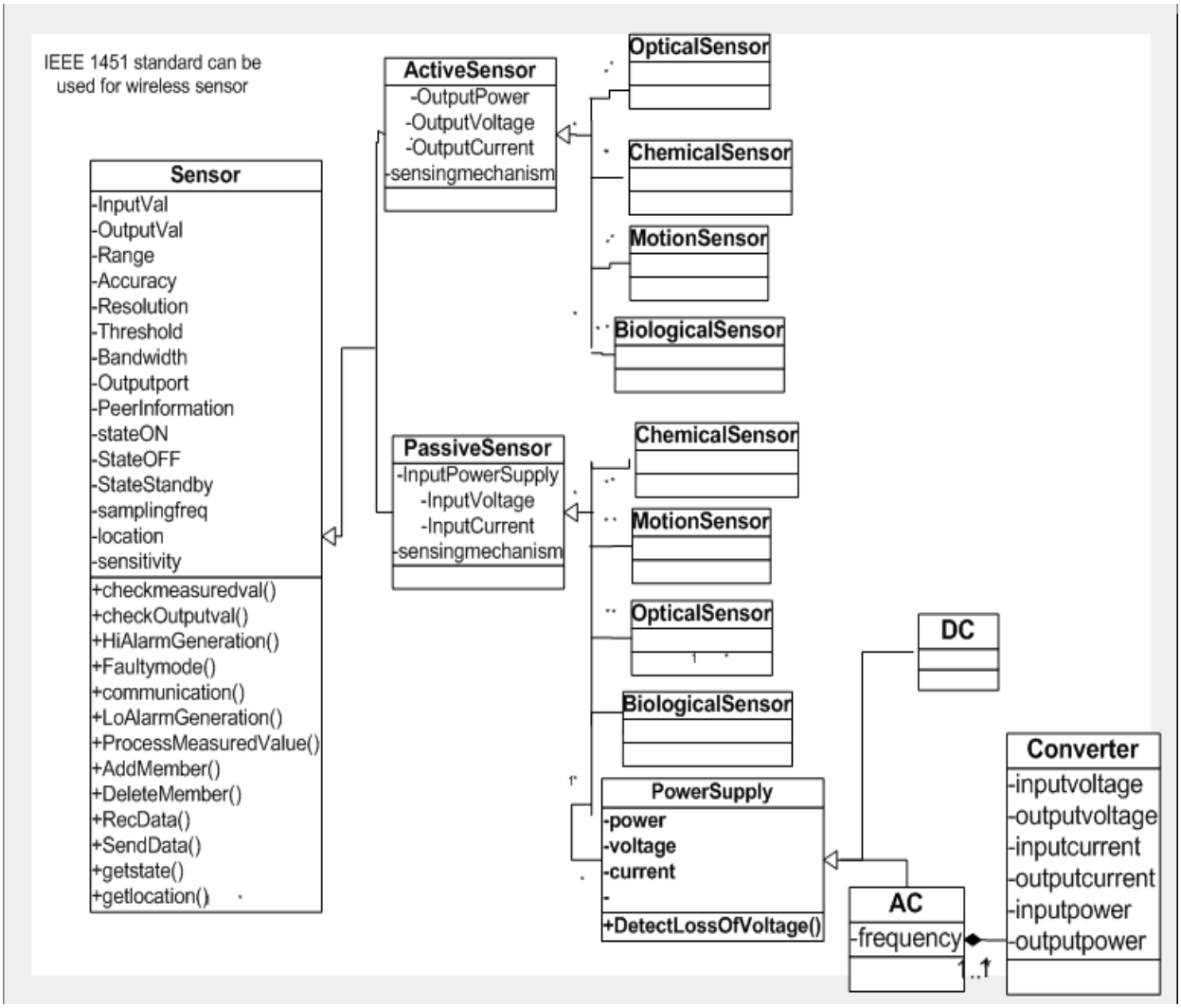


Figure 34 Abstract level

This is an abstract representation of a sensor. All sensors will have some basic abilities and they are classified here based on phenomena. Basically apart from the sensing mechanism used all sensor should be capable of the activities listed in the sensor class. Sensors can be defined as Active or Passive depending on their power supply requirements. Active sensors generate power and do not require an external source while passive do. The Figure below takes an orthogonal view into what lies at the component level inside a sensor. Every sensor has to be equipped with these

functionalities irrespective of its sensing mechanism. All the sensors have a physical cover, a communication port etc.

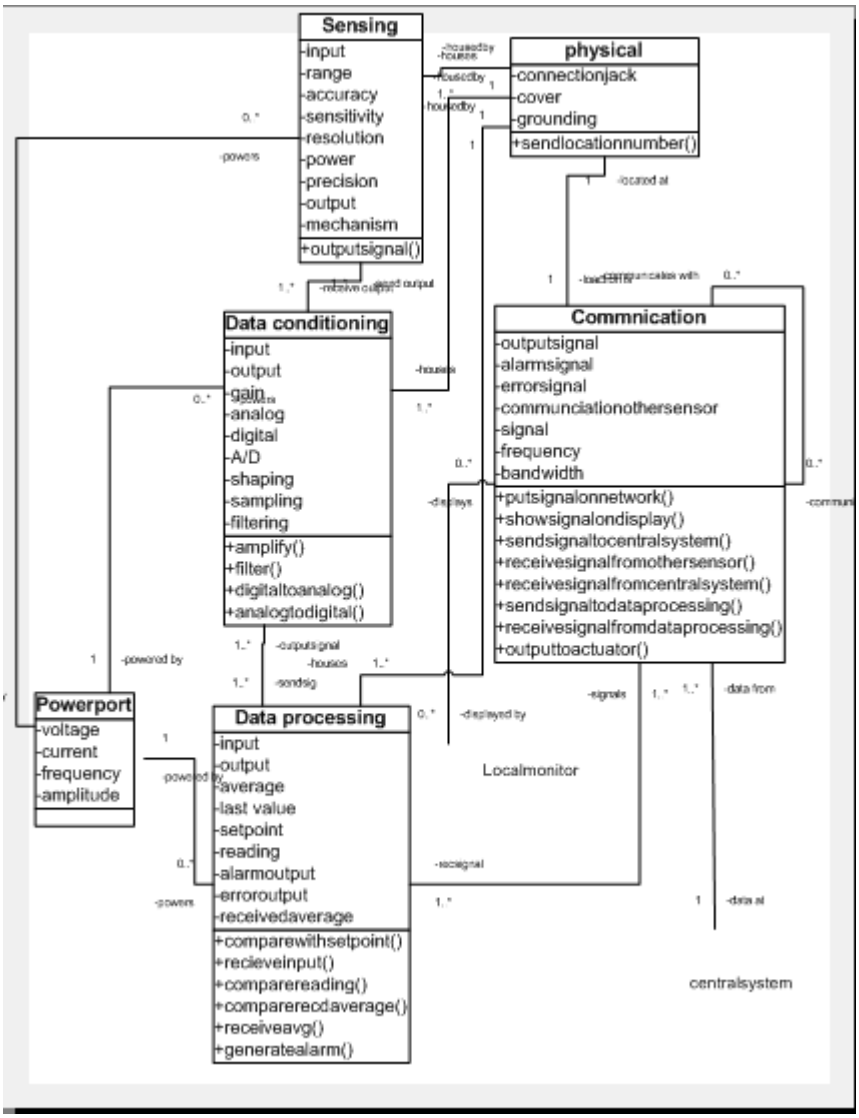


Figure 35 Sensor object

Each sensor internally will have an architecture as shown in Figure 10. The only place they would differ in is in the mechanism used for sensing.

4.5.1 Example: Biological sensor

The effective detection of biological agents in the environment requires a multicomponent analysis system because of the complexity of the environment. Other variables contributing to the effectiveness of detection of biological agents are the detection process itself and the efficient use of consumables in the field. Biological agent detection systems generally consist of four components: the trigger/cue, the collector, the detector, and the identifier. Figure 41 shows a flow diagram for a typical point detection automated architecture system. The function of these components is described in the remainder of this section, while section 5 will provide representative examples of each component. [23]

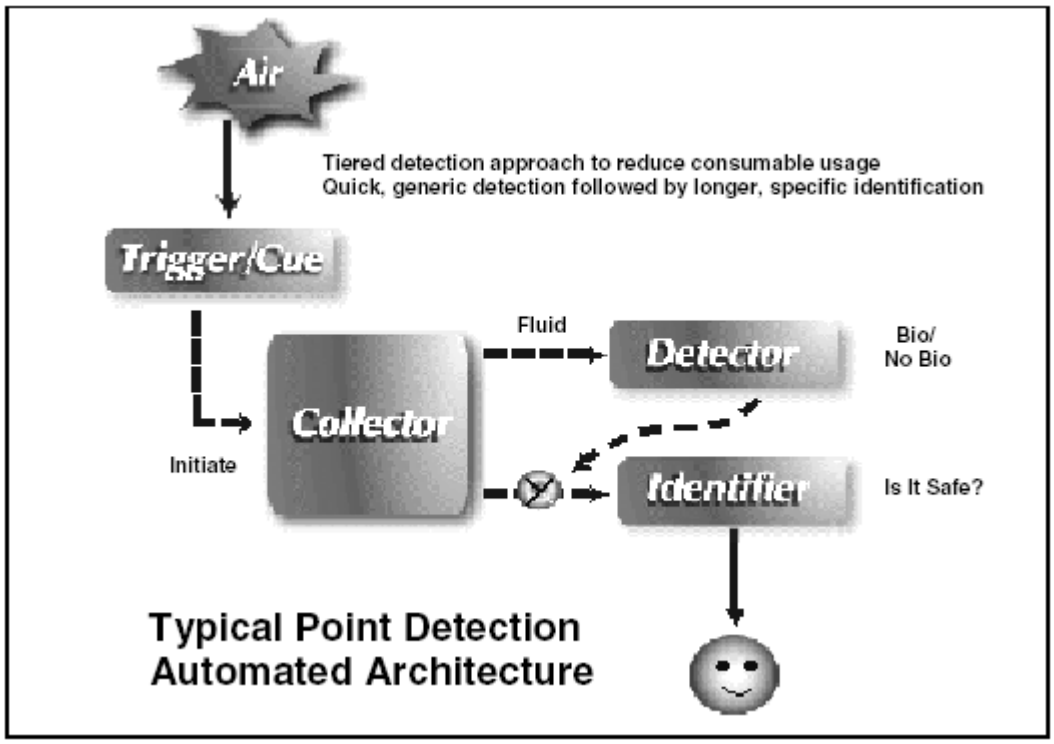


Figure 36 Biological sensor

Trigger/Cue

Trigger technology is the first level of detection that determines any change in the particulate background at the sensor, indicating a possible introduction of biological agents. Detection of an increase in the particulate concentration by the trigger causes the remaining components of the detection system to begin operation. The trigger function typically provides a means of continuously monitoring the air without unnecessary use of consumables, thus keeping the logistical burden of biological agent detection low.

To reduce false positives (alarm with no biological agent) and false negatives (no alarm with agent), many detection systems combine trigger technology with a second detector technology (such as fluorescence that provides more selectivity) into a single technology known as cueing.

Most effective cueing technologies can detect airborne particulates in near real time and can discriminate between biological agent aerosol particles and other particles in air, avoiding unnecessary system activation. For example, a cueing device monitors the air for particulates as does any other trigger device. When the particulate concentration increases, the cue determines if the particulates are biological in nature. The cue device generally uses a fluorescence detector to make this determination. If the particulates are found to be biological, the cue device activates the collector for sample collection.

Collector

Sampling of the biological agent is a crucial part of the identification system. The effective dose for some agents is extremely small; therefore, highly efficient

collection devices must be employed. One type of collector pumps large volumes of air through a chamber where the air mixes with water. The water scrubs all the particulates from the air, resulting in a sample containing particulates suspended in water. Once collected in the water, the sample is further concentrated by evaporation of a portion of the water. After concentration, the sample moves into the analytical section of the biological agent detection system.

Detector

Once a sample has been collected/concentrated, it must be determined if the particulates are biological or inorganic in origin. To accomplish this, the sample is passed to a generic detection component that analyzes the aerosol particles to determine if they are biological in origin. This component may also classify the suspect aerosol by broad category (e.g., spore, bacterium, toxin/macromolecule, or virus). In its simplest form, the detector acts as a "gateway" for further analysis. If the sample exhibits characteristics of biological particles, it is passed through to the next level of analysis. If the sample does not exhibit such characteristics, it is not passed to the next level of analysis, thereby conserving analytical consumables. It is important to note that detection has traditionally taken place after the trigger function. For example, an aerosol particle sizer (APS) triggers, then a detector (e.g., flow cytometer) examines the aerosol for biological content. Many of the newer detection technologies combine the trigger and detection functionalities into a single instrument, creating a cueing instrument. As described in section 4.1, the cue first

detects a rise in particulates then determines if the particulates are of biological origin. If the sample is biological, the collector gathers a sample and passes it directly to the identifier.

Identifier

An identifier is a device that specifically identifies the type of biological agent collected by the system. Identifiers are generally limited to a preselected set of agents and cannot identify agents outside of this set without the addition of new identifier chemistry/equipment or preprogramming. Because the identifier performs the final and highest level of agent detection, it is the most critical component of the detection architecture and has the widest variety of technologies and equipment available. The information obtained from the identifier is then used to determine protection requirements and treatment of exposed personnel.

4.6 System access

The system access prevents unauthorized access and protects the premises from an intruder attack. It comprises of the card reader and the ID card-purpose being every employee is equipped with an ID card that they swipe in the card reader to gain access into the protected premise. It uses biometrics to strengthen the identification process as the ID card tells that the card belongs to the company while the biometric tells that you belong to the company. The following class diagram explains this architecture.

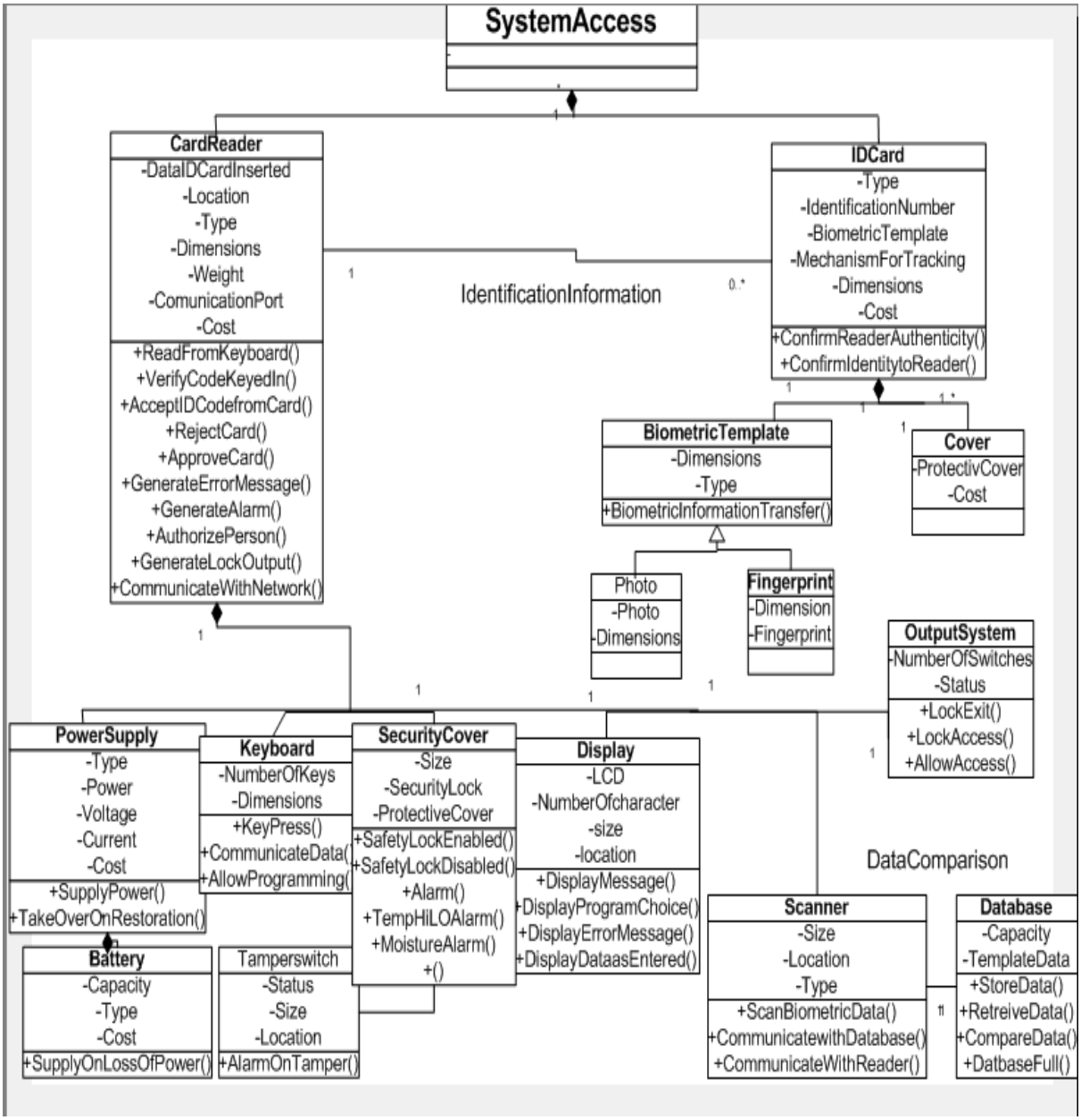


Figure 37 Access architecture

Chapter 5: System Behavior

Once the static structure is in place identifying all the objects of the system, the system behavior diagrams show the interactions between these objects as they achieve the various tasks chalked out for the system.

5.1 Interaction Diagrams:

Interaction diagrams model the behavior of use cases by describing the way groups of objects interact to complete the task. The two kinds of interaction diagrams are **sequence** and **collaboration** diagrams.

Interaction diagrams are used when it is required to model the behavior of several objects in a use case. They demonstrate how the objects collaborate for the behavior. Interaction diagrams do not give an in depth representation of the behavior Sequence diagrams, collaboration diagrams, or both diagrams can be used to demonstrate the interaction of objects in a use case. Sequence diagrams generally show the sequence of events that occur. Collaboration diagrams demonstrate how objects are statically connected. Both diagrams are relatively simple to draw and contain similar elements. [13]

5.1.1 Sequence diagrams

Sequence diagrams demonstrate the behavior of objects in a use case by describing the objects and the messages they pass. Consider for example the Use case for Access control; in this the user submits a card which has an access code associated with it. The user is prompted to enter the code upon swiping the card. If the code is wrong a

prompt is issued for reentry of the code. The user is allowed to enter the code thrice before the card is confiscated and security alerted.

The following sequence diagram illustrates the class objects and interaction between them that is needed to accomplish this entire card verification process:

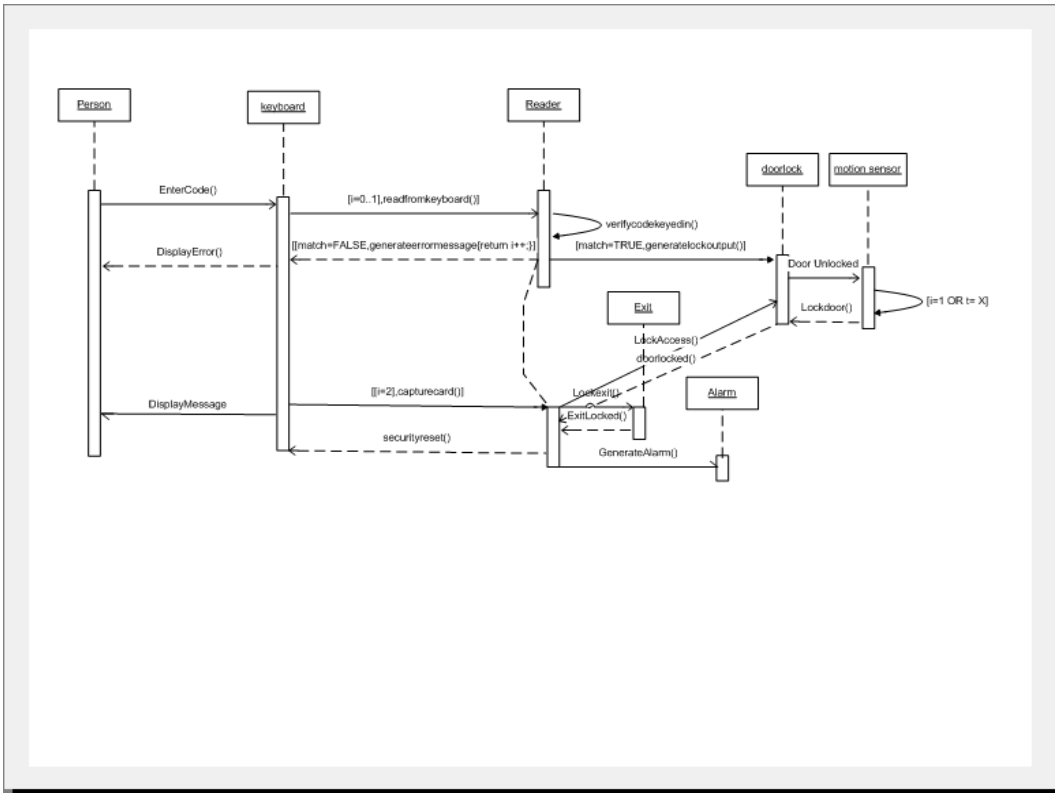


Figure 38 Card Verification

A person swipes his ID card and enters code. The keyboard object communicates with the reader which has an inbuilt processor that checks the code, if there is an error or a match is not found the error message is displayed. At the same time the function will keep track of the number of times the code has been entered by incrementing variable “i”. If of course the code keyed in was right the door lock is opened and the motion sensor will register how many people have tried to enter. In case the number

of persons entering is more than one an alarm is sounded or if the number of times the code entered is >3 the card is captured, door is locked and security alerted.

The following sequence diagram explains the messages exchanged when a threat is detected in the environment. The sensor that detects abnormality in the environment sets off an alarm as well as puts the signal on the network. If the threat is determined to be correct (See Sensor Fusion Chapter 6) the dampers are actuated to seal off the ducts of the HVAC system, the UV lights are switched on and the ducts are pressurized.

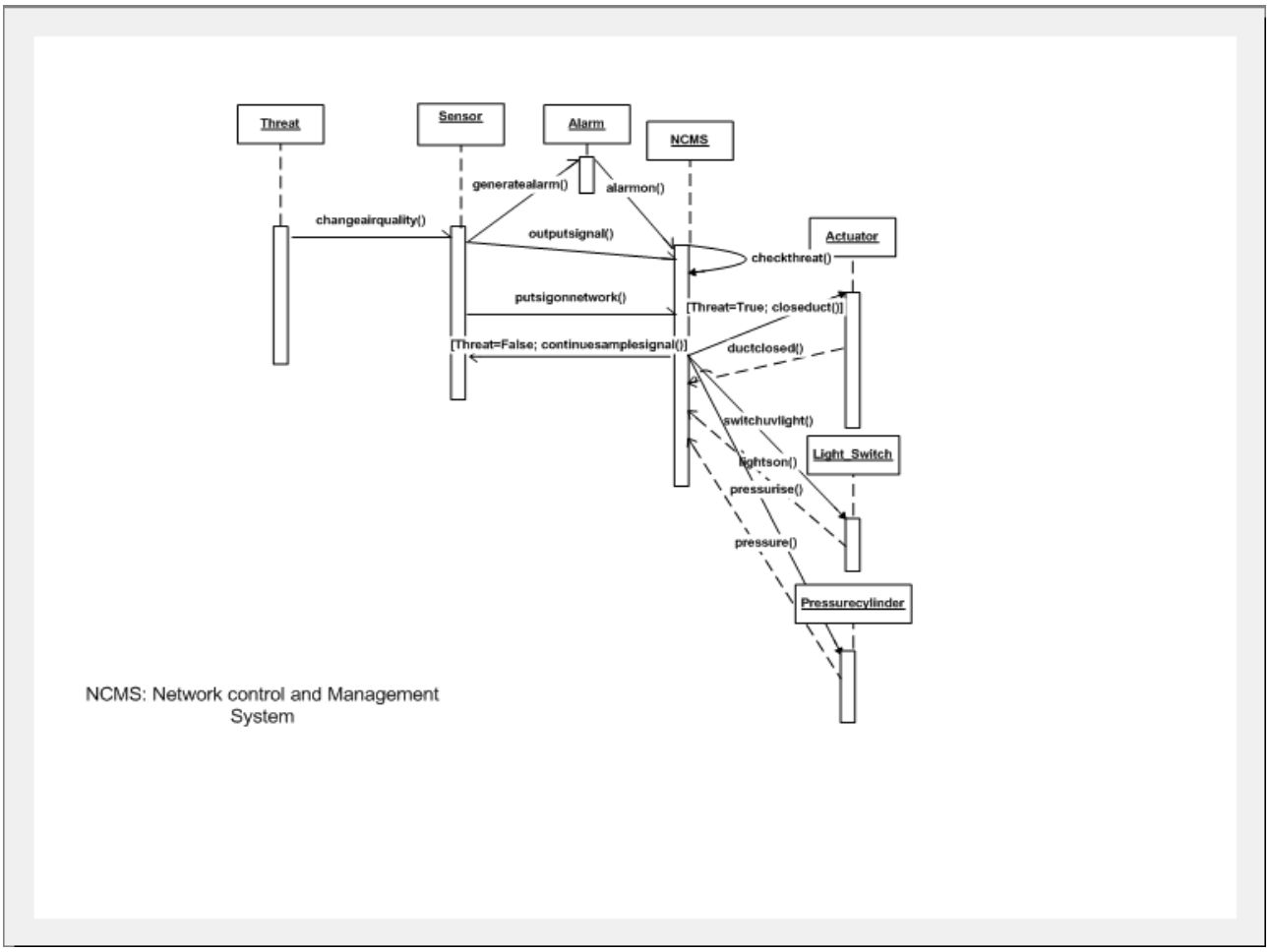


Figure 39 Sensor Alarm

The next sequence diagram explains the alarm generation process. It illustrated the response of the sensor on detecting an abnormality. An alarm is sounded and the message is put on the network.

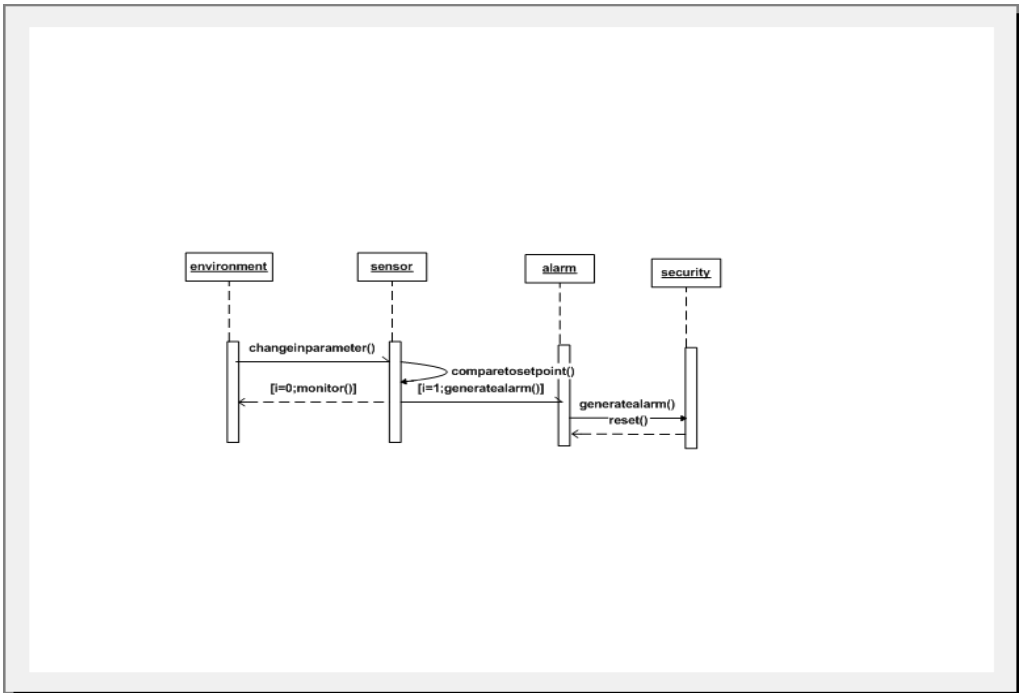


Figure 40 Alarm

The alarm system is set off after receiving messages from any sensor. However this diagram is an abstraction of the redundant logic used by sensors for determining if a measurement is indeed violating its limits. The internal logic of the sensors is hidden and is shown in the sequence diagram covered in the Sensor Fusion chapter # 6.

The next diagram explains the messaging that has to take place in order to protect the IS system form unauthorized access. This works in conjunction with the reader system, giving a two layered security system where the first prevents unauthorized access while the other makes sure that the person has passed all the appropriate channels before logging onto the IS system. Here the IS system checks for the user account after receiving the correct username and password. Once the account is verified, the IS system then checks to see if this was an authorized entry into the

premises. Only then does it let the person access the system. If the person has not passed through the appropriate channels the system alerts security.

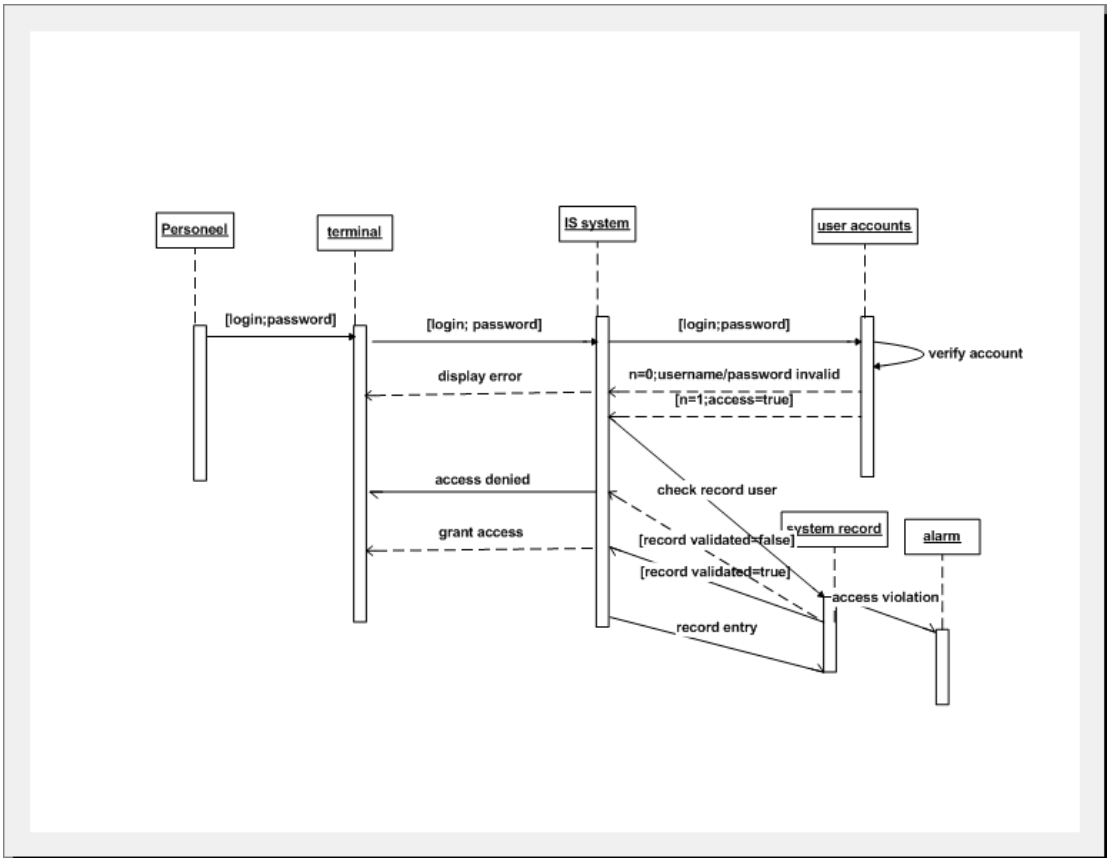


Figure 41 IS Security

5.2 State and Activity Diagrams

5.2.1 State Charts

State diagrams are used to describe the behavior of a system. State diagrams describe all of the possible states of an object as events occur. Each diagram usually represents objects of a single class and tracks the different states of its objects through the system.

State diagrams demonstrate the behavior of an object through many use cases of the system. Not all classes will require a state diagram and state diagrams are not useful for describing the collaboration of all objects in a use case. State diagrams are other combined with other diagrams such as interaction diagrams and activity diagrams. The following state diagram divides the system into 4 major states:

1. Normal state: Indicating the system is fully functional, with all alarms reset and faults cleared. The transition out of this state is when an alarm occurs. The system alert state is a super state comprising of two smaller states
2. Check for False alarm: Here based on information received there is a transition to the maintenance state(this occurs if a false alarm is received) or a transition to the two sub states
   - Intruder threat: The threat is determined to be that of an intruder and appropriate actions are taken. Once the situation is restored back to normal(indicated by the setting of the Situation Normal flag to 1) a transition is made to the system normal state
   - Bio Threat: The threat is determined to be that of a chemical/biological agent and appropriate actions are taken. Once the situation is restored back to normal(indicated by the setting of the Situation Normal flag to 1) a transition is made to the system normal state
3. System Wait: The system is normal but monitoring still continues, any further faults are logged and transitions are made to maintenance. Once timer expires the SysNormal flag is set to 1 and system goes back to normal state

4. Maintenance: All faults are attended and transition is to the normal system state when flag Fault Cleared is set to 1

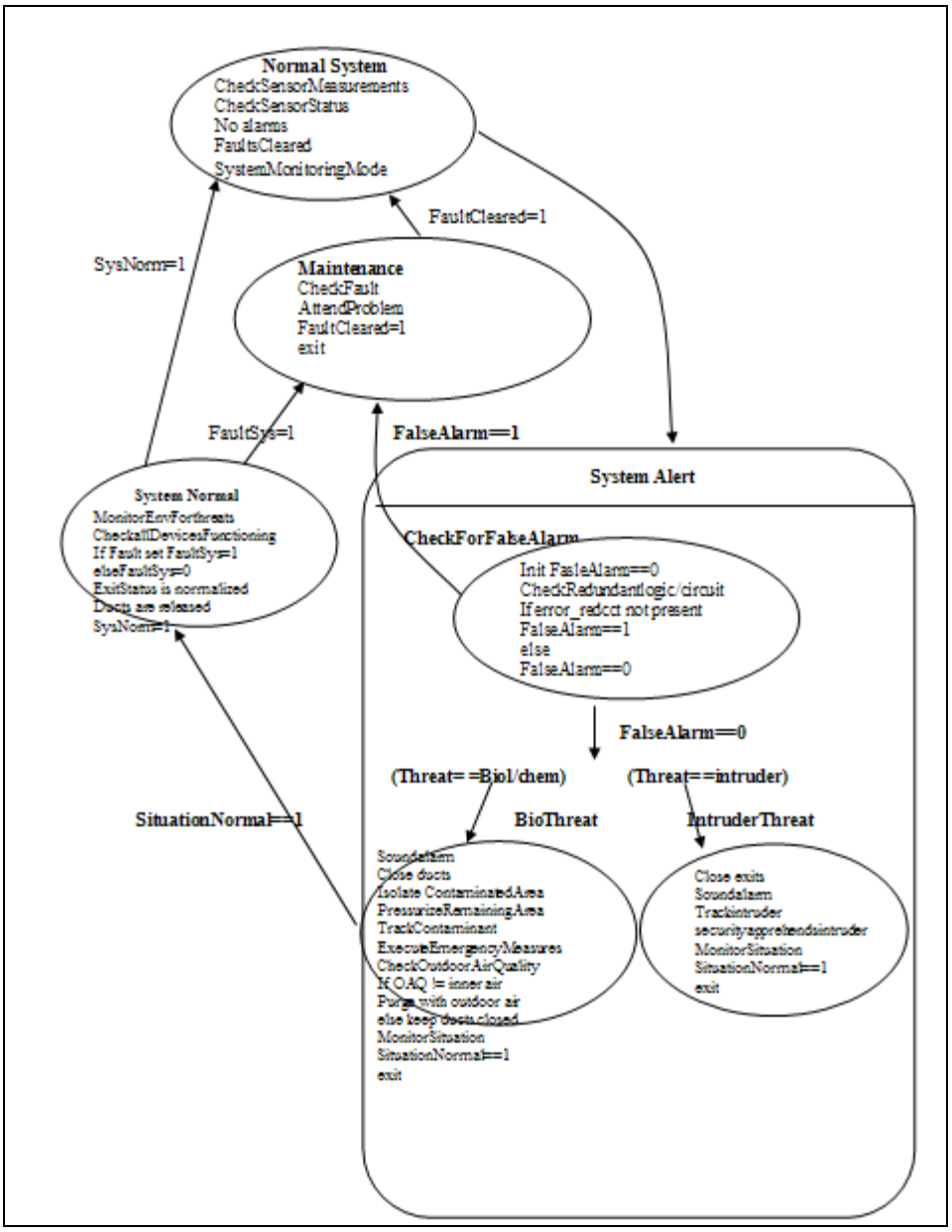


Figure 42 System Statechart

The statechart in Figure 42 shows the various states the system is in when an employee attempts access to the system. The states explain the entire approach of validating an employee while allowing him access to the premises.

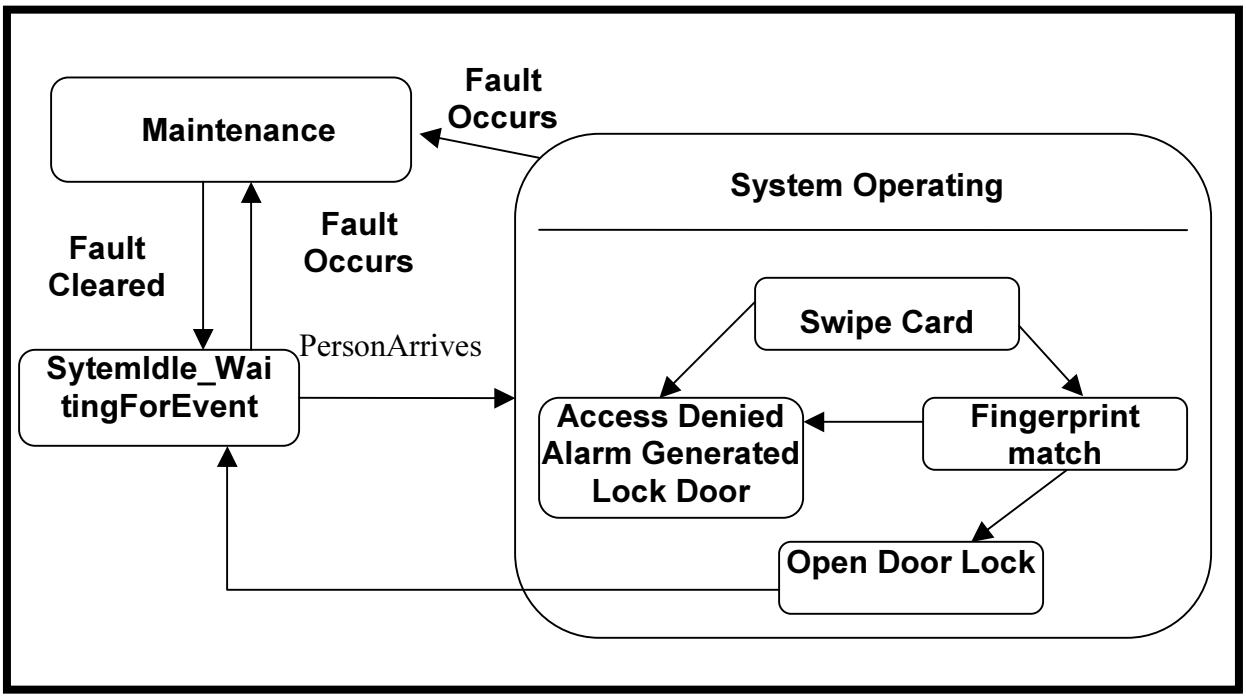


Figure 43 Access System Statechart

5.2.2 Activity Diagrams

Activity diagrams describe the workflow behavior of a system. Activity diagrams are similar to state diagrams because an activity is the state of doing something. The diagrams describe the state of activities by showing the sequence of activities performed. Activity diagrams can show activities that are conditional or parallel.

Activity diagrams are used in conjunction with other modeling techniques such as interaction diagrams and state diagrams. The main reason to use activity diagrams is to model the workflow behind the system being designed. Activity Diagrams are also useful for: analyzing a use case by describing what actions needs to take place and when they should occur; describing a complicated sequential algorithm; and modeling applications with parallel processes.

However, activity diagrams should not take the place of interaction diagrams and state diagrams. Activity diagrams do not give detail about how objects behave or how objects collaborate.

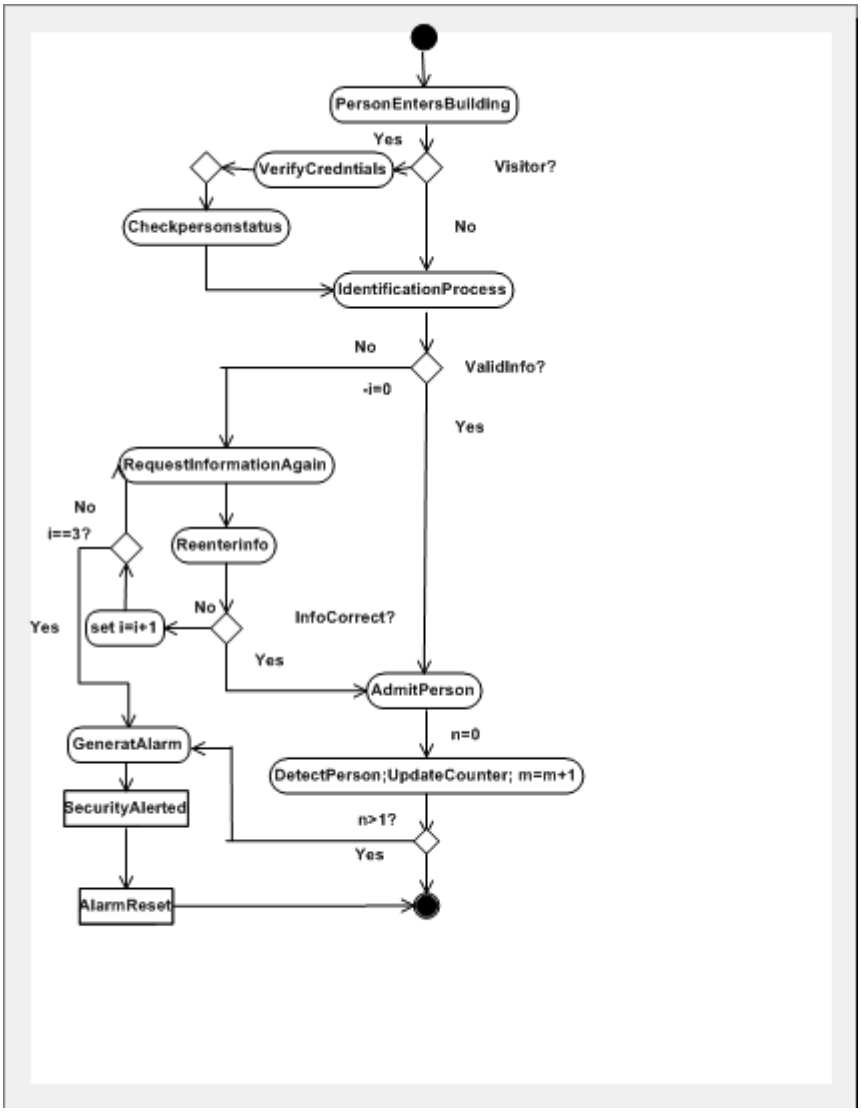


Figure 44 Authorized entry

The activity diagrams show the sequence of events that have to be followed to correctly realize authorized entry. It starts with deciding if the access is temporary or

regular authorized access. The system then continues with the verification process and enlists the steps to be taken to prevent unauthorized access and tailgating. It summarizes the sequential flow of events for authorized and unauthorized access into the building. The diagram shows the card being captured after three attempts at the access code.

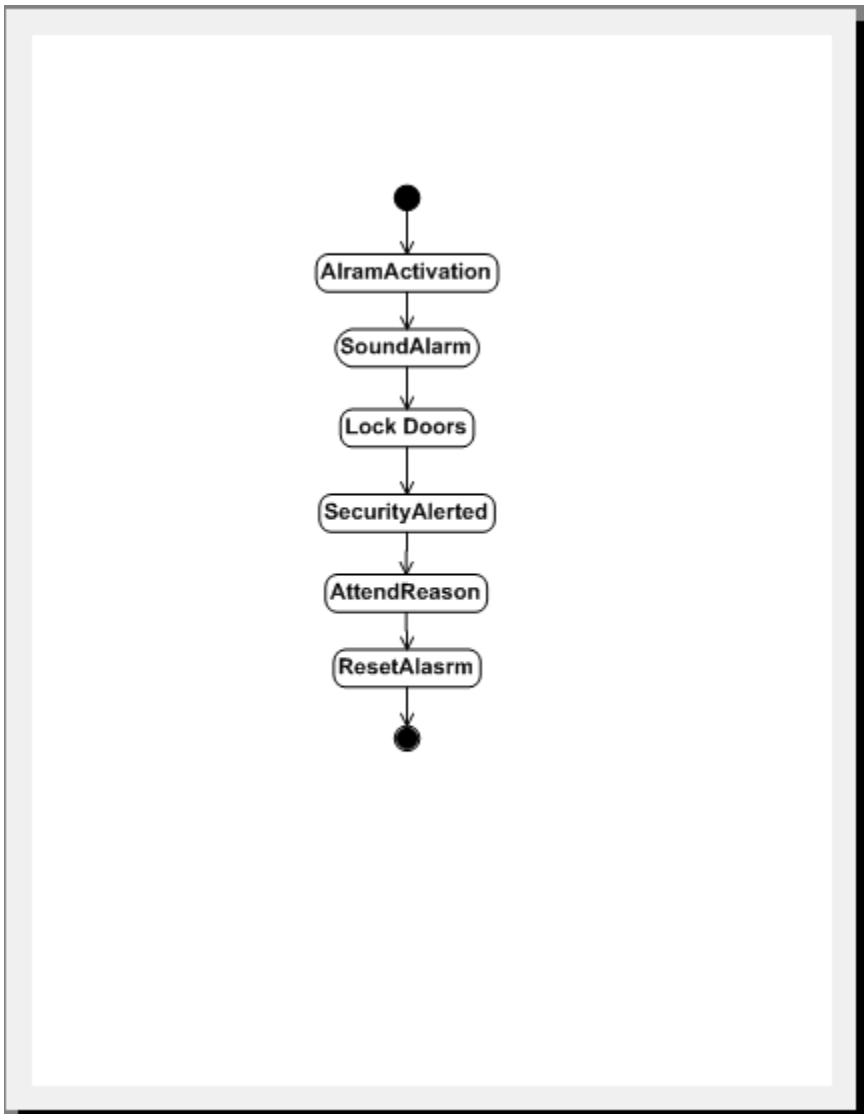


Figure 45 Alarm

Alarm activation incase of intruder entry. The alarm action is accompanied by locking the doors to cut out any attempts at retreat.

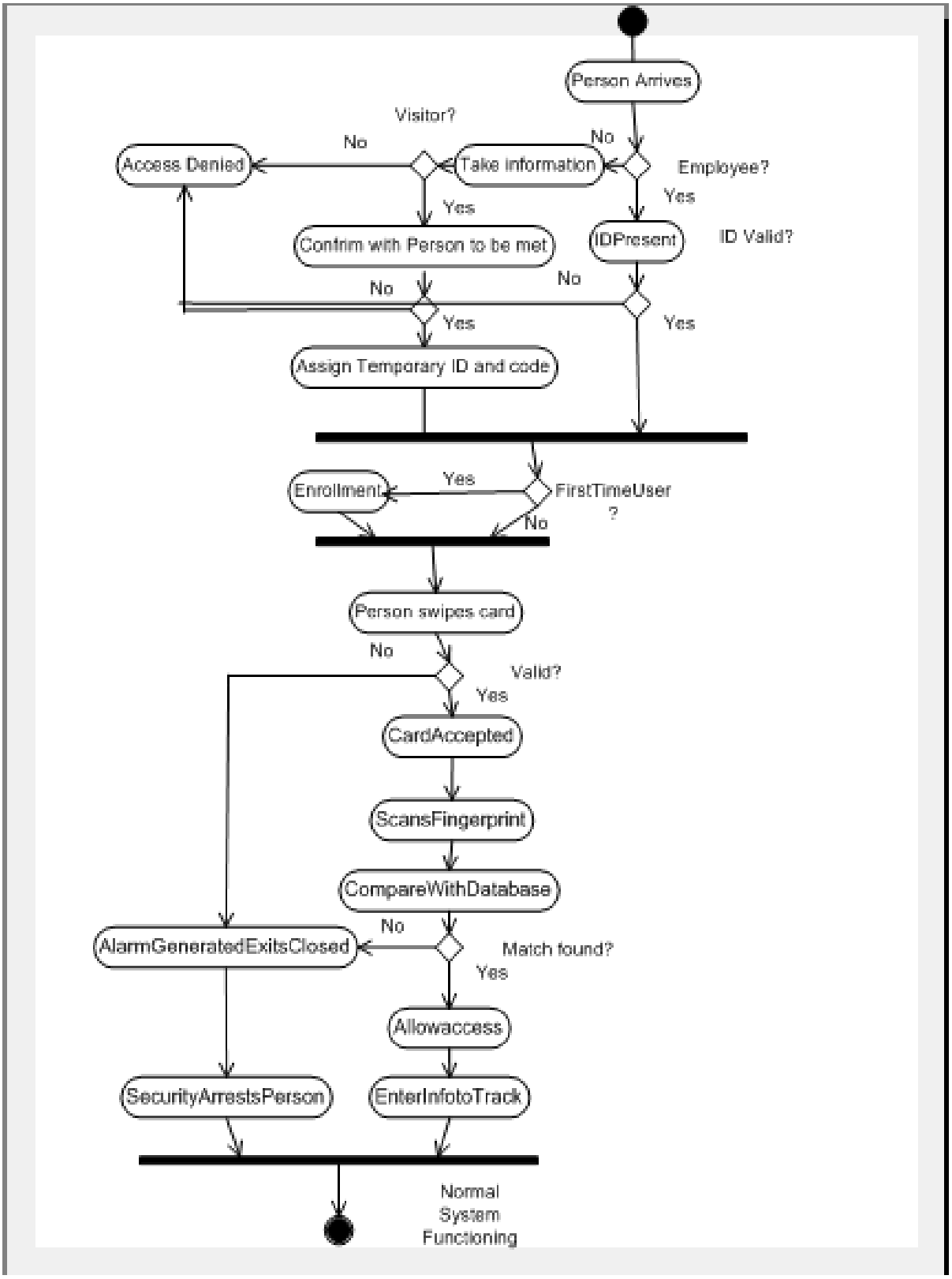


Figure 46 Access

This activity diagram condenses all the use cases for the access into one single diagram. It covers entry, check, authorized/temporary/unauthorized access and alarm generation.

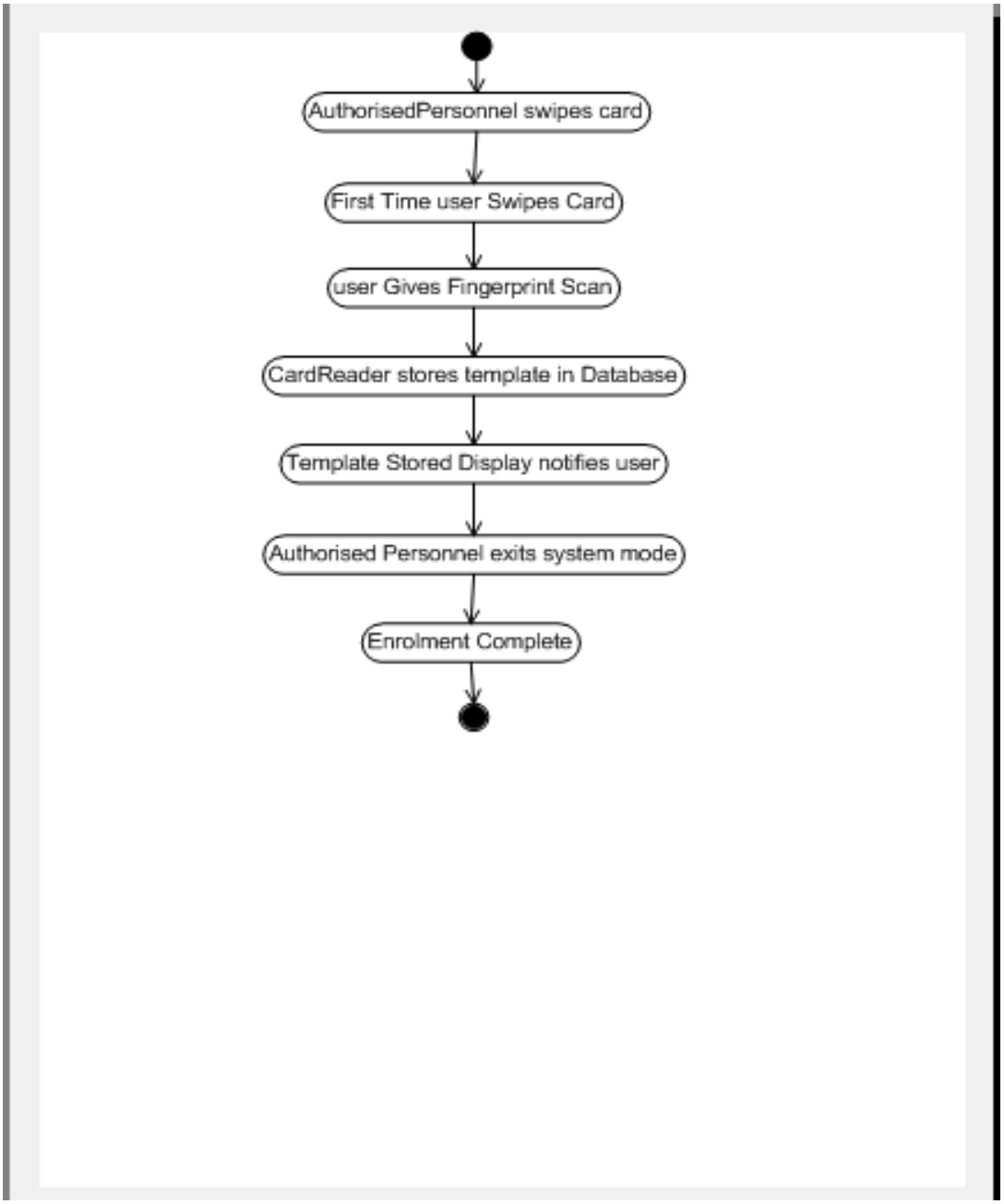


Figure 47 Enrollment

This covers the enrollment process. A standard procedure required for registering the user and building up the database. The whole process requires an authorized personal to take the steps of enlisting the employee into the system.

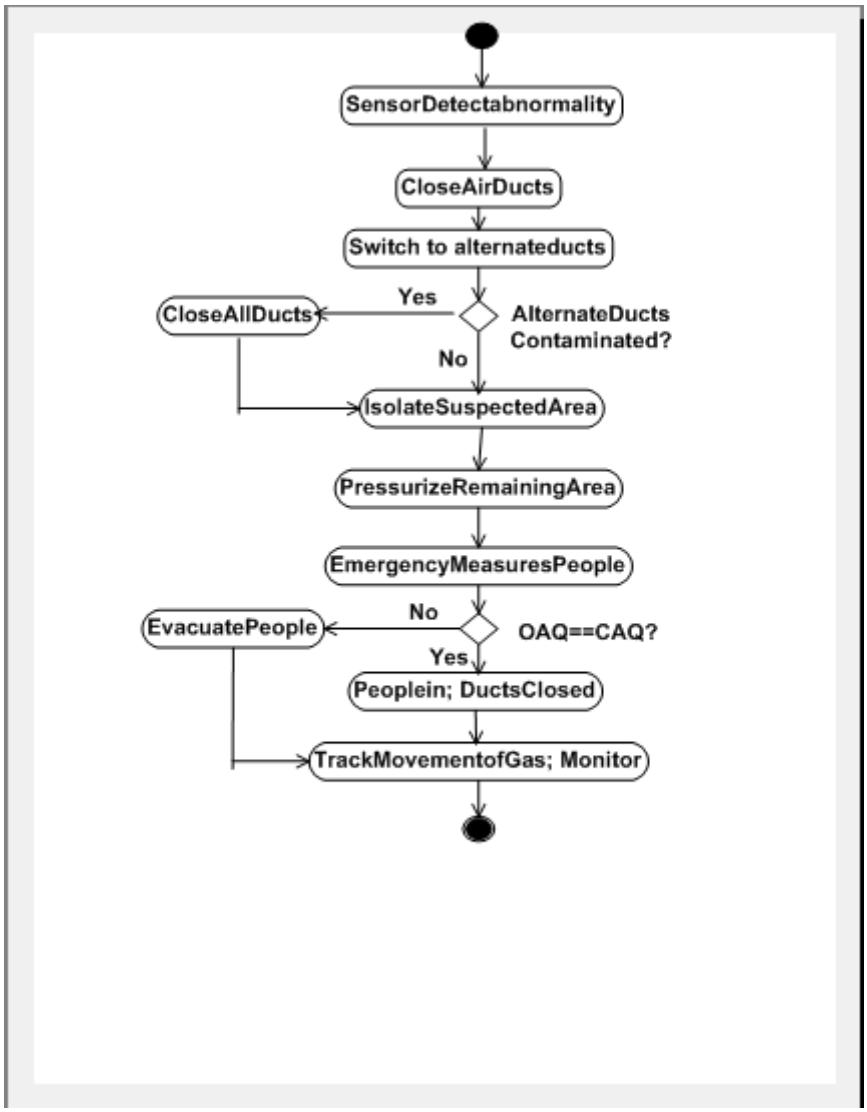


Figure 48 Monitoring CB threat

An activity diagram that explains the steps that need to be adapted in the wake of a chemical/biological attack. Here the focus is to switch to alternate ducts provide they are not contaminated. Air quality checks are made to determine either case. Isolation

of the suspected area is carried out and followed y a pressurization to prevent leaks into the environment. Thus containing the attack and not allowing it to spread.

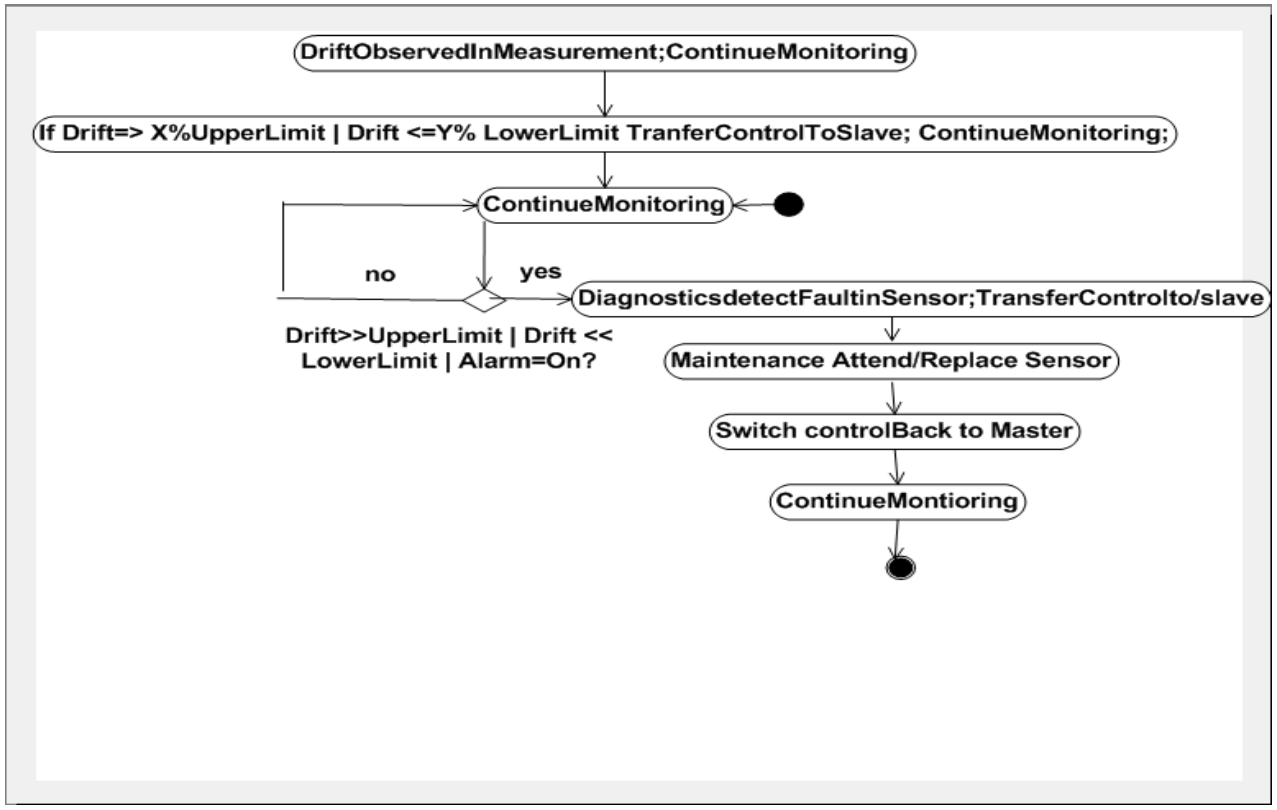


Figure 49 Error Checks

Here a master/slave configuration is used. If a drift is observed in the measurements logic is used to determine whether the drift is +/- the tolerance levels. If it is control is switched over to the slave and diagnostics signal a fault in the master system.

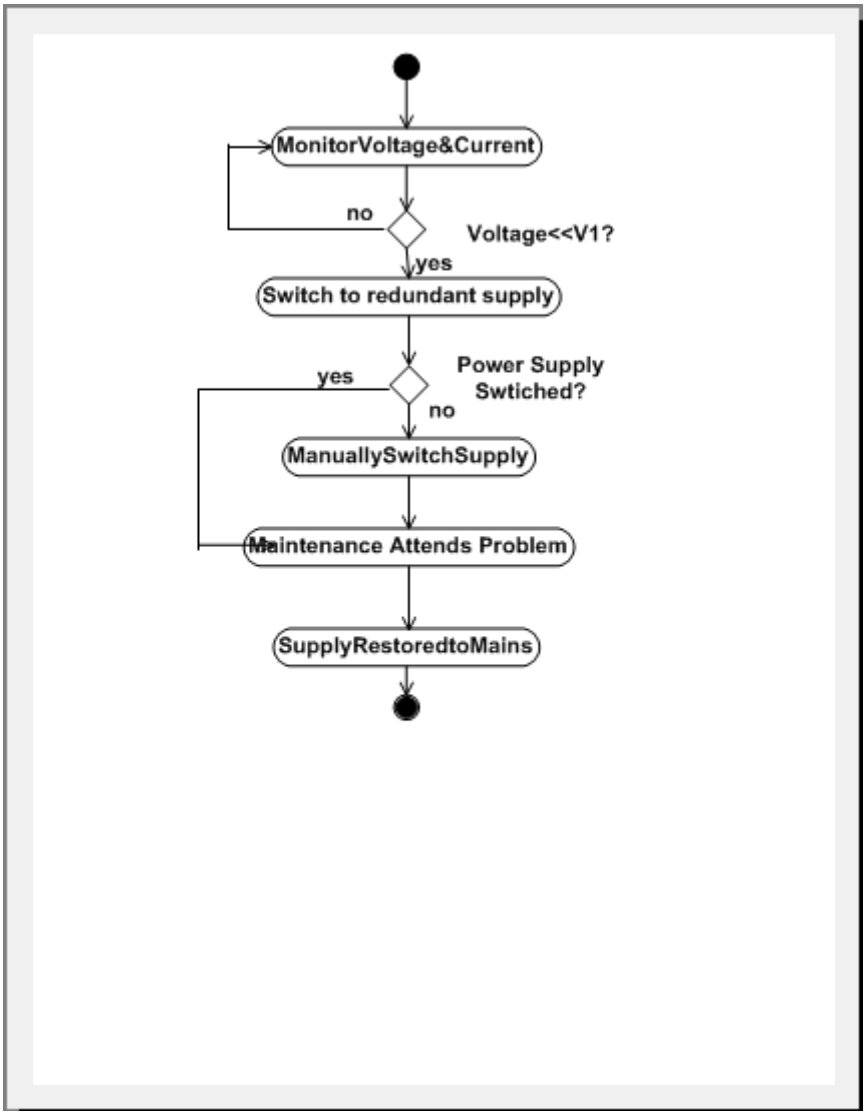


Figure 50 Power Failure

If the power supply to the system fails there has to be back up supply to provide uninterrupted service. The activity diagram illustrates how an uninterrupted power supply can be ensured to the system.

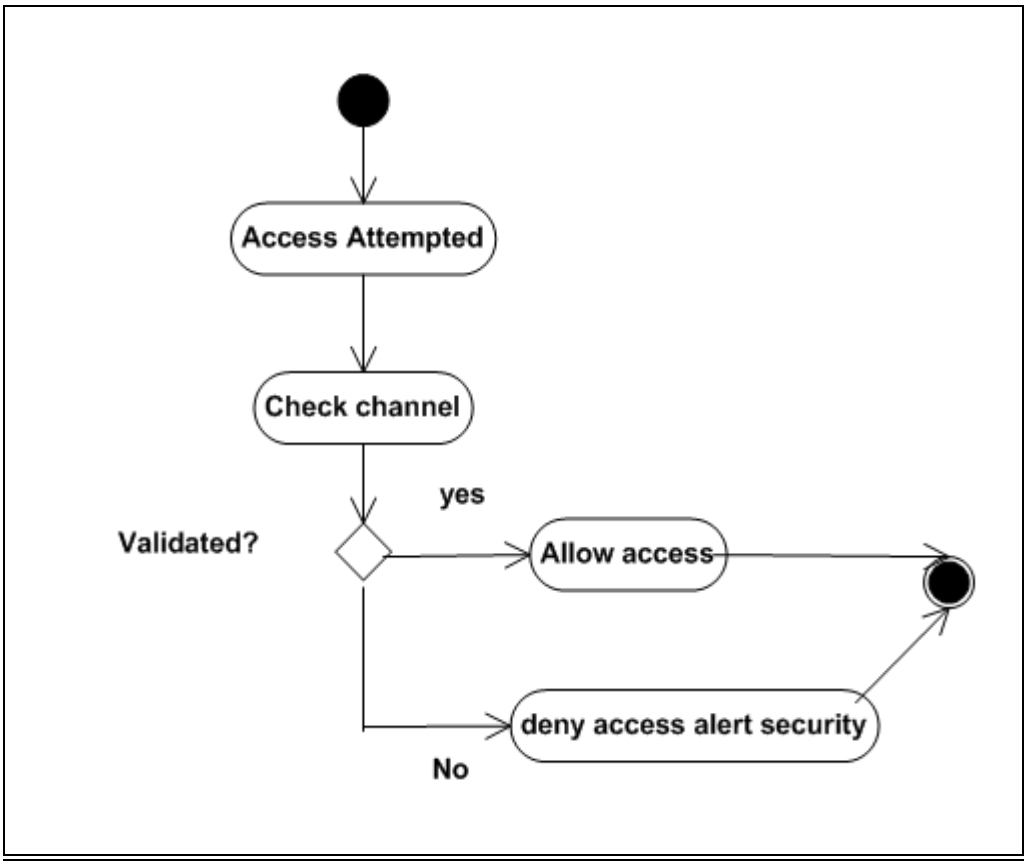


Figure 51 IS system

The steps to be taken to protect the IS system from any breach in security. If access has not been attempted after clearing appropriate channels access is denied alerting security.

Chapter 6: Sensor Fusion

As per Requirement # 2 , the fault alarm must be kept low. This can be done if sensors collaborate and share their measurements to arrive at a consensus. This approach is called sensor fusion, not only does it increase the reliabilty of the measurement it also decreases the contingency of a false alarm.[25]

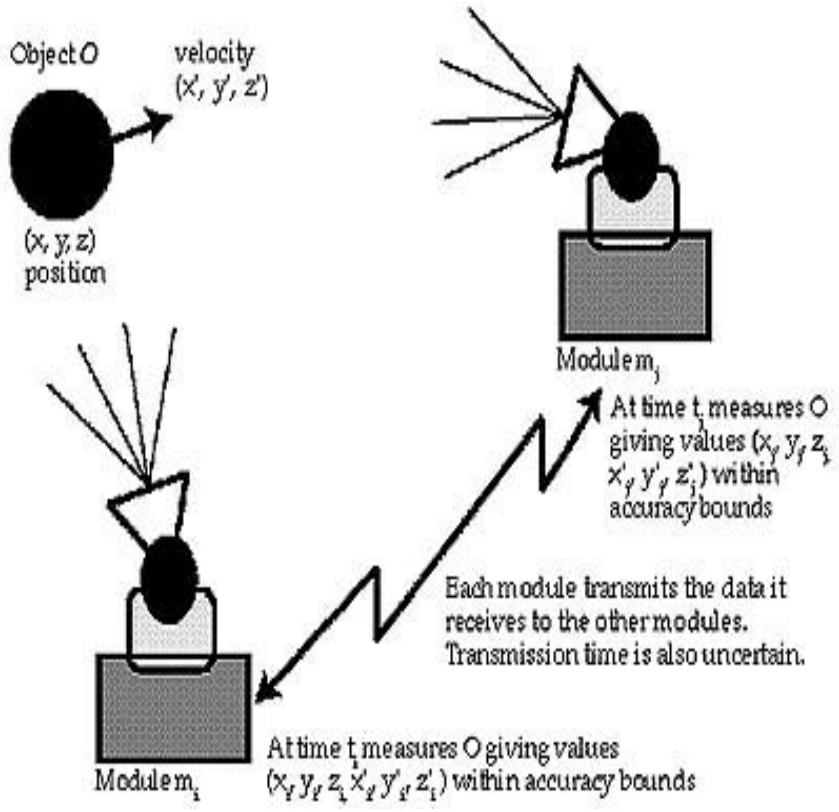
6.1 Introduction

A human being recognizes external environment by using many kinds of sensory information. By integrating these information and making up lack of information for each other, a more reliable and multilateral recognition can be achieved. *Sensor Fusion* realizes new sensing architecture by integrating multi-sensor information so that reliable and multilateral information can be extracted, which can realize high level recognition mechanism. ***The fusion method must be designed carefully, because an inappropriate fuser can render the system worse than the worst individual sensor.***

Sensor fusion can be divided into three classes: complimentary sensors, competitive sensors, and cooperative sensors.

- Complimentary sensors do not depend on each other directly but can be merged to form a more complete picture of the environment, for example, a set of radar stations covering non-overlapping geographic regions. Complementary fusion is easily implemented since no conflicting information is present.

- Competitive sensors each provide equivalent information about the environment. A typical competitive sensing configuration is a form of N-modular redundancy. For example, a configuration with three identical radar units can tolerate the failure of one unit. This is a general problem that is challenging, since it involves interpreting conflicting readings.
- Cooperative sensors work together to drive information that neither sensor alone could provide. An example of cooperative sensing would be using two video cameras in stereo for 3D vision. This type of fusion is dependent on details of the physical devices involved and cannot be approached as a general problem.There are two types of major algorithm areas Value Fusion and Detection Fusion; these topics are covered in detail in Chapter 8.Consider the example where two sensors are tracking object motion



Here the object has certain attributes: The position information determines where objects are, whereas the identity information determines what they are.

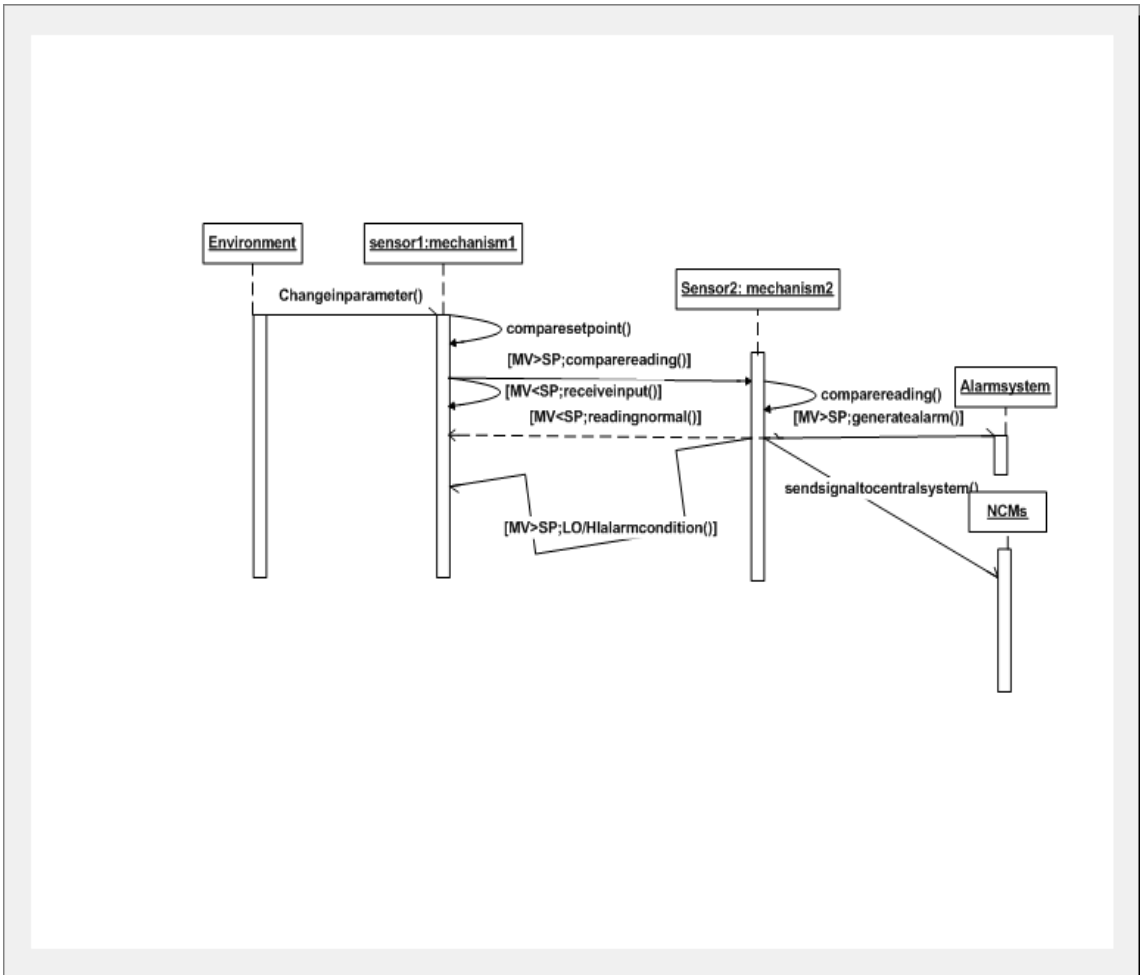


Figure 52 sensor fusion

The sequence diagram illustrates how sensors group and communicate with each other to confirm information regarding measurements. When one sensor detects a threat, it passes the information to the next configured sensor that may/may not confirm the threat. If with the combined information from both the sensors the threat is confirmed then the alarms are set off and the central unit is notified.

Sensor fusion not only makes the system more robust it also minimizes the likelihood of having non diagonal elements dominating the confusion matrix. In view of requirement# 2 a confusion matrix with diagonal elements>0 and non diagonal elements =0 is desired.

A confusion matrix (Kohavi and Provost, 1998) contains information about actual and predicted classifications done by a classification system. Performance of such systems is commonly evaluated using the data in the matrix. The following table shows the confusion matrix for a two class classifier.[26]

The entries in the confusion matrix have the following meaning in the context of our study:

- *a* is the number of **correct** predictions that an instance is **negative**,
- *b* is the number of **incorrect** predictions that an instance is **positive**,
- *c* is the number of **incorrect** of predictions that an instance **negative**, and
- *d* is the number of **correct** predictions that an instance is **positive**.

		Predicted	
		Negative	Positive
Actual	Negative	**A**	**b**
	Positive	**C**	**d**

Table 28 Confusion Matrix

Several standard terms have been defined for the 2 class matrix:

- The *accuracy* (*AC*) is the proportion of the total number of predictions that were correct. It is determined using the equation:

$$AC = \frac{a+d}{a+b+c+d} \quad [1]$$

- The *recall* or *true positive rat*e (*TP*) is the proportion of positive cases that were correctly identified, as calculated using the equation:

$$TP = \frac{d}{c+d} \quad [2]$$

- The *false positive rate* (*FP*) is the proportion of negatives cases that were incorrectly classified as positive, as calculated using the equation:

$$FP = \frac{b}{a+b} \quad [3]$$

- The *true negative rate* (*TN*) is defined as the proportion of negatives cases that were classified correctly, as calculated using the equation:

$$TN = \frac{a}{a+b} \quad [4]$$

- The *false negative rate* (*FN*) is the proportion of positives cases that were incorrectly classified as negative, as calculated using the equation:

$$FN = \frac{c}{c+d} \quad [5]$$

- Finally, precision (P) is the proportion of the predicted positive cases that were correct, as calculated using the equation:

$$P = \frac{d}{b+d} \qquad [6]$$

We obviously desire that the matrix terms a and d be high while b and c be low. This chapter helps to quantify requirement# 2 that asks for a low fault rate. Chapter 8 involves finding the minimum fault rate for a given deployment strategy that will not violate the cost constraint.

Chapter 7: Characterizing Evacuation behavior with Agent UML

7.1 An Introduction

Agents are an extension of active objects, exhibiting both dynamic autonomy (the ability to initiate action without external invocation) and deterministic autonomy (the ability to refuse or modify an external request). Thus, our basic definition of an agent is "an object that can say 'go' (dynamic autonomy) and 'no'(deterministic autonomy)."

The Unified Modeling Language (UML) is gaining wide acceptance for the representation of engineering artifacts in object-oriented software. The view of agents as the next step beyond objects led to exploration of extensions to UML and idioms within UML to accommodate the distinctive requirements of agents. The result is Agent UML (AUML). Agent UML (AUML) synthesizes a growing concern for agent based software methodologies with the increasing acceptance of UML for object-oriented software development.

7.1.1 UML and AUML

To make sense of and unify various approaches on object oriented analysis and design, an Analysis and Design Task Force was established within the OMG. By November 1997, a de jure standard was adopted by the OMG members called the Unified Modeling Language (UML) formalizes the methods of many approaches to the object-oriented software lifecycle, including Booch, Rumbaugh, Jacobson, and Odell.

Many researchers have argued that UML provides an insufficient basis for modeling agents and agent-based systems. Basically, this is due to two reasons: Firstly, compared to objects, agents are active because they can take the initiative and have control over whether and how they process external requests. Secondly, agents do not only act in isolation but in cooperation or coordination with other agents. Multiagent systems are social communities of interdependent members that act individually. To employ agent-based programming, a specification technique must support the whole software engineering process — from planning, through analysis and design, and finally to system construction, transition, and maintenance.

A proposal for a full life-cycle specification of agent-based system development is beyond the scope for this thesis. Both FIPA and the OMG Agent Platform SIG are exploring and recommending extensions to UML. Moreover it is planned that within the European network of Excellence AgentLink a working group should be established on this topic. In this thesis, we will focus on a new subset of an agent-based UML extension for the specification of the agent internal behavior of an agent and relating it to the external behavior of an agent using and extending UML class diagrams.

7.2 UML Class diagrams –revisited

First of all a closer look at the concepts of object oriented programming languages, namely the notions of object and class and adapt it afterwards to agent based systems.

7.2.1 Basics

In object oriented programming languages an object consists of a set of instance variables, also called attributes or fields, and its methods. Creating an object its object identity is determined. Instance variables are identifiers holding special values, depending on the programming languages these fields can be typed. Methods are operations, functions or procedures, which can act on the instance variables and other objects. The values of the fields can be either pre-defined basic data types or references to other objects.

A class describes a set of concrete objects, namely the instances of this class, with the same structure, i.e. same instance variables, and same behavior, i.e. same methods. There exists a standard method 'new', to create new instances of a class. A class definition consists of the declaration of the fields and the method implementations. It consists of a specification or an interface part as well as of an implementation part. The specification part describes, which methods with which functionality are supported by the class, but not how the operation is realized. The implementation part defines the implementation / realization of the methods and is usually not visible to the user of the method. The access rights define which methods are visible to the user and which one are not. In most programming languages classes define also types, i.e. each class definition defines a type of the same name.

Some programming languages allow in class definitions also the definition of class variables, which are shared by all classes, in contrast to instance variables belonging to a single object. i.e. each instance of a class has its own storage for its instance

variables, in contrast to class variables which share the same storage. Class variables are often used as a substitute for global variables. Beyond class variables, there are often used class methods which can be called independently of a created object and are used as global procedures.

7.2.2 Relating Objects with Agents

As already stated, an agent is more than an object, see figure 52

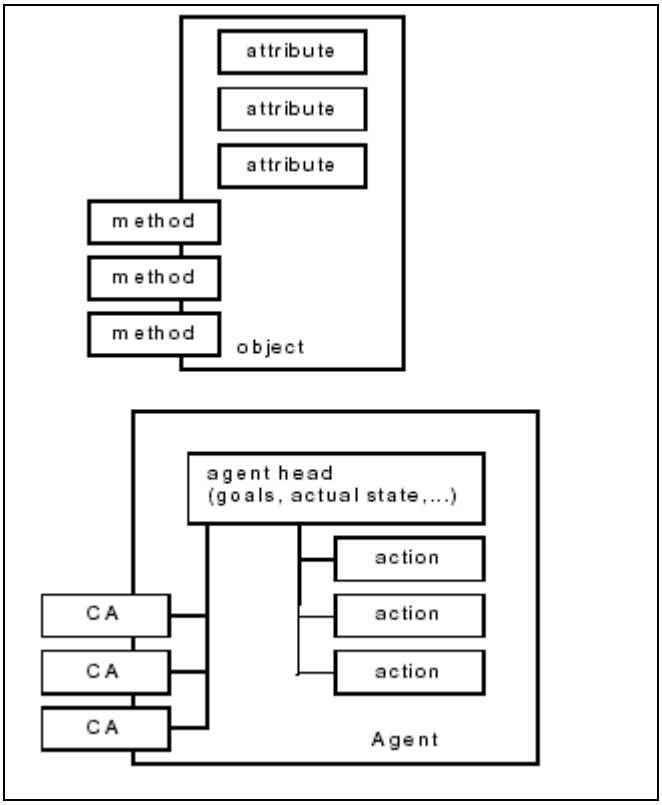


Figure 53 Object vs. Agent

We have autonomy, pro- and re-activity, the communication is based on speech act theory (communicative acts, CA for short), the internal state is more than only fields with imperative data types, and additional features. All these concepts have to be supported by a class diagram for agents. In the agent oriented programming paradigm

we have to distinguish between an agent class defining on the one side the type of an individual agent and being on the other side a blue print for individual agents, i.e. an (individual) agent is an instance of an agent class. Therefore we specify the schema of an agent class which is then used in programs as instantiated agents. An agent can be divided into the communicator - doing the physical communication, head - dealing with goals, states, etc. of an agent - and body - doing the pure actions of an agent. For the internal view of an agent we have to specify the agent's head and body.
The reaction to events and pro-active behavior can be defined either by pro-active actions or agent head automata for pro-active behavior. Not only methods can be defined for an agent which are only visible to the agent itself, but actions which can be accessed by other agents. But in contrast to object orientation the agent decides itself whether some action is performed or not. [27]

.

7.2.3 Agent Class Diagrams

A class diagram is a graphical view of the static structural model. A class in the sense of object oriented programming is a blueprint for objects, in our context an agent class has to be a blueprint for agents. A class describes a set of concrete objects, namely the instances of this class, with the same structure, i.e. same instance variables, and same behavior, i.e. same methods. There exists a standard method 'new', to create new instances of a class. A class definition consists of the declaration of the fields and the method implementations. It consists of a specification or an interface part as well as of an implementation part. The specification part describes which methods with which functionality are supported by the class, but not how the

operation is realized. The implementation part defines the implementation / realization of the methods and is usually not visible to the user of the method This can be either an instance of an agent or a set of agents satisfying some special role or behavior. These describe the kind of agents that exist in the system. In standard UML, there is a notation for active objects (with their own thread of execution).

Agents must have their own thread of execution, but they are not mere objects: they have a knowledge of their environment (by means of sensors) and may act upon it (by means of effectors). They have abilities and can be requested to perform a certain action. This is different from invoking a method on them, as the agent may refuse to perform the action. Figure 42 (Agent class Person) shows the symbol we use for Agent Classes. White dots are used to indicate sensors and effectors. It is possible to connect other (Agent) classes to these dots to mean that the agent can sense or act upon that other class. Most of the times, Agent Classes have a statechart diagram specifying the agent behavior (see Figure 41).A static object diagram is an instance of a class diagram, where objects and their relationships may appear. It shows a snapshot of the state of the system at a point in time. Here we include Agents in this kind of diagrams. These are instances of Agent Classes, and are represented in a similar way (see Figure 41).

7.2.4 Single room model

This section deals with the simpler case, in which we consider evacuations of single rooms. In our model, rooms are discretized and represented as two-dimensional,

rectangular grids. The Figure attempts to explain the behavior of the agent using a statechart. In the initial state the agent enters the “moving randomly” state. Time is discretized, so that movement of agents can be recorded. Agents with the capabilities described in Figure 42 can only see and move; they do not communicate. Once an agent sees a door, its objective is to move towards it.

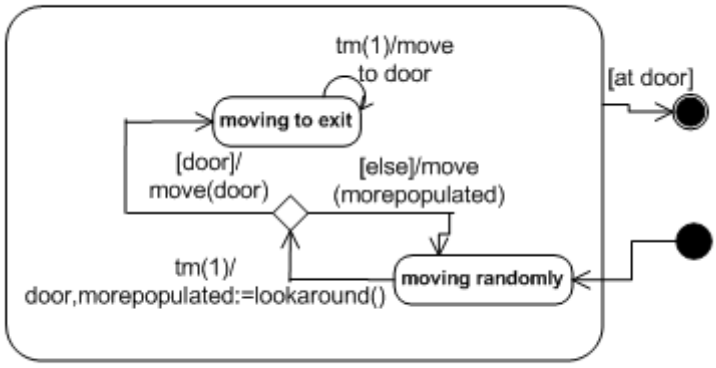


Figure 54 Statechart

Figure 53 is a Statechart representing this behavior. Transitions in the model invoke methods (lookaround() and move()), which should be considered as the agent capabilities. These capabilities make use of the agent’s sensors and effectors which are described in the class diagram. When the agent is in the state

1. Move Randomly. It looks around(lookAround()) and if it finds a door the transition move(door) causes it to go to moving to exit. If it doesn’t find the door it goes to morepopulated area which leads it back to the moving randomly state with other agents.The agent’s structure is shown in Figure 42.
2. In the Moving to exit state the sensor will always move from one door to another until it finally finds the exit which does not connect to any other door.

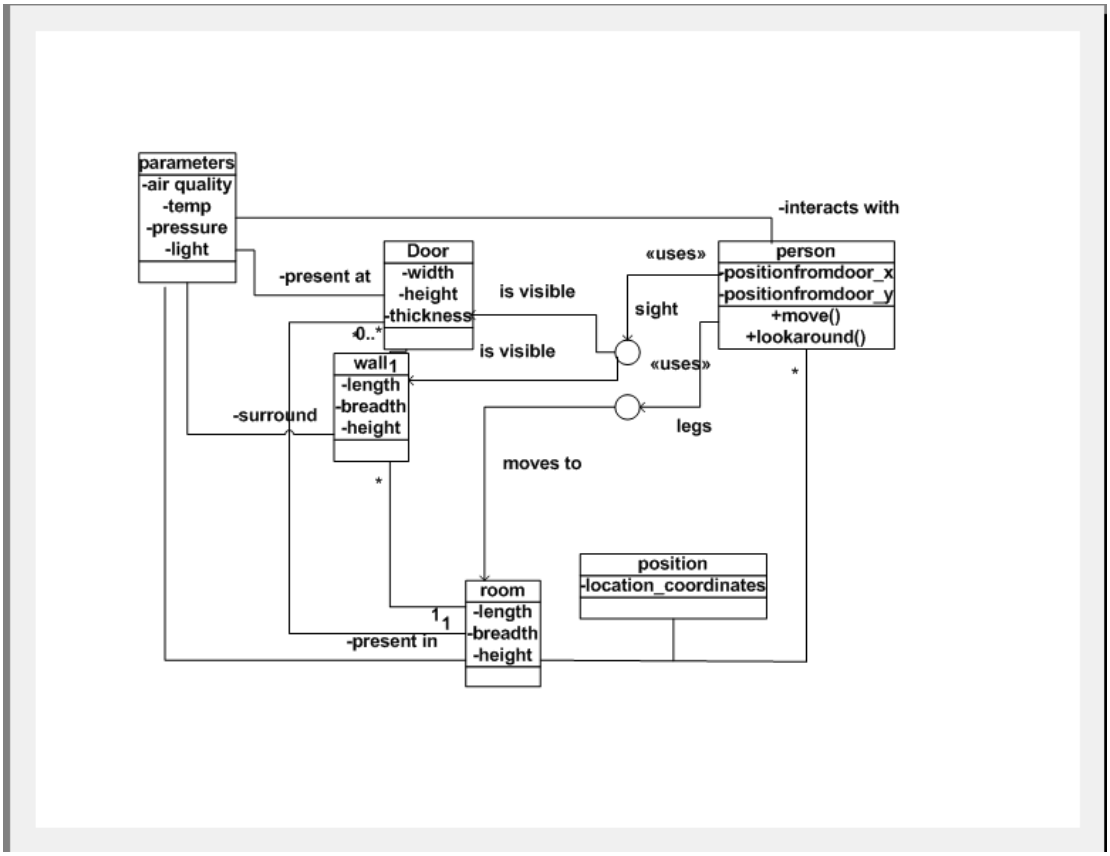


Figure 55 Person class

Figure 54 shows an agent class Person, which has a sensor sight and an effector - legs. Relationship “is visible” and “move” state that the sensors and effectors can act (move) or sense(see). In this case, the sight sensor can sense either Doors, Walls or other person. The legs effector can act on Rooms i.e., agents can walk into/in the room. Agent capabilities move and lookaround are specified in the Agent class, these were used in the statechart of Figure 41. Attributes positionfromdoor_x and positionfromdoor_y are used to store the position of the door the agent is moving towards in the case he has seen a door before. A Person is situated in a room, and this is expressed with the relationship class Position.

The area of building is restricted to consist of a room made of several doors and walls. Doors are placed in walls with the relationship has. The dimensions of the room are stored in attributes breadth, height and length. The door coordinates are stored in its attributes. The interaction between the environment and the agents is expressed by using the sensor/effector notation.

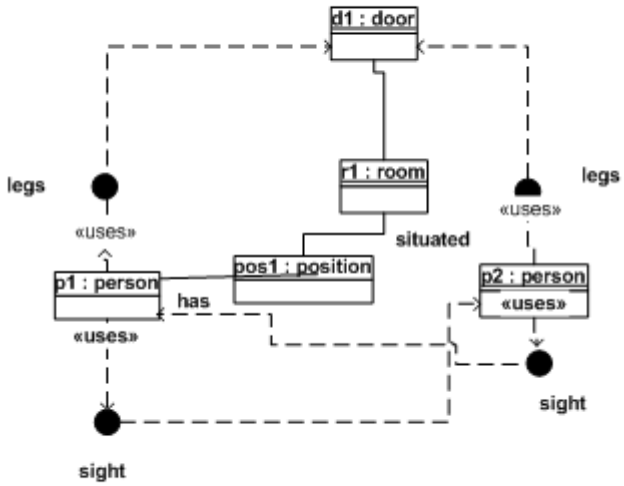


Figure 56 An agent diagram

Figure 55 shows an agent diagram that reflects the way in which an agent can sense the presence of doors or other agents. The Figure shows a situation in which an agent (r1) is able to see another agent (r2) and a door. The condition for this to happen is that no other visible object must be between r1 and r2 or the door.

7.2.5 Extending the model for multiple rooms

In this section, we consider buildings with multiple rooms. The agent structure must be extended with a "mental" representation of the map of the building to guide the agent in his navigation towards the exit.

Considered here are two situations:

1. in the first one the agent does not have any a prior knowledge of the building layout, he builds his mental map while exploring the building looking for the exit.
2. In the second situation, we assume that the agents have partial or total information about the building.

In both cases, the mental map is used by the agent to navigate trough the building.

Class Building has been introduced, composed by a number of rooms. Class Door has been extended with the attribute type indicating if the door is an exit or leads to another room. Inner doors are connected to other inner doors leading to other rooms; exit doors are not connected to other doors, as they lead to the outside.
The mental map of the environment the agent builds and uses for navigation is shown. The agent is able to recognize a room if he has been in the room before. The same happens with doors inside rooms. The agent also remembers if he has explored the door before or not. This is because as per the statechart in Figure 45 the agent memorizes rooms and locations once they have been visited. The agent capabilities

have been extended with the possibility to memorize new rooms or doors as they are discovered. If the agent has a prior knowledge of the building map, then this capability guides the agent through the rooms towards the exit. If the agent does not have a prior knowledge, then his mental map may not be complete, and several situations can arise. In the first case, if he knows an exit door in the current room, this is the door it will take. If an exit door is not present in the current room, then the agent moves out of the current room and looks for it in other rooms.

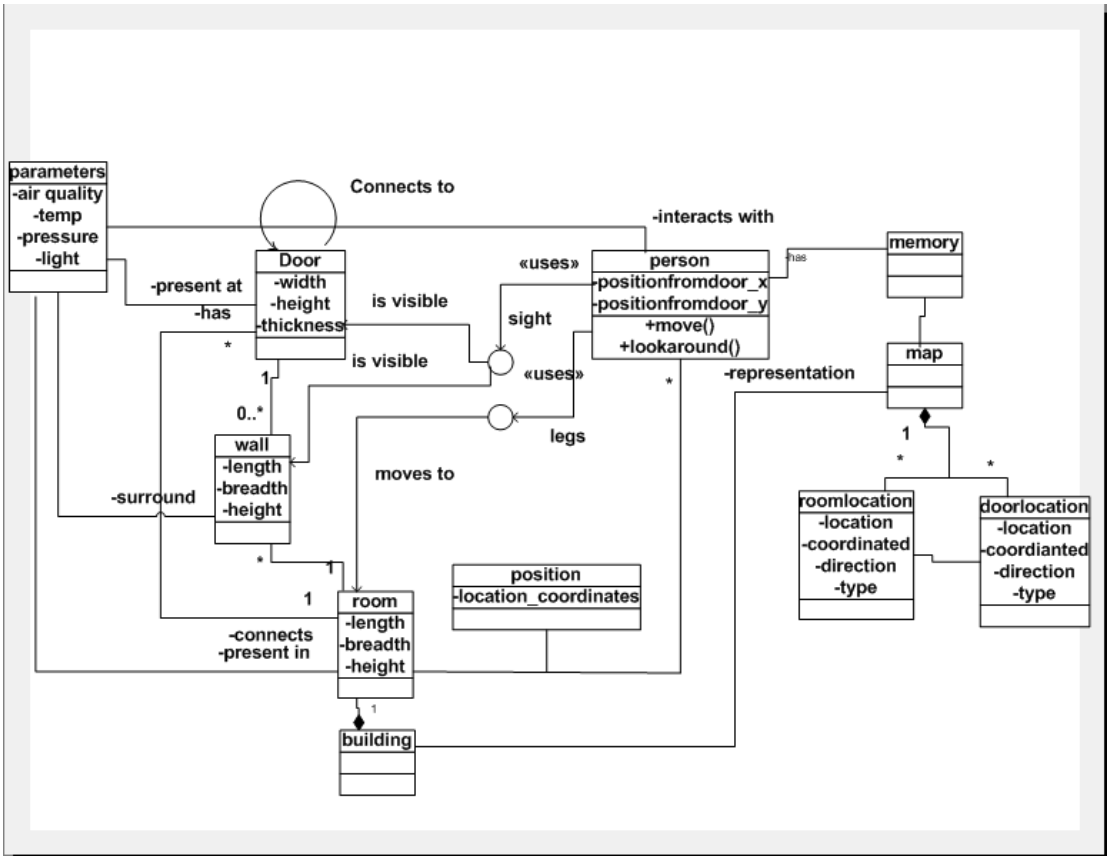


Figure 57 Class Diagram for multiple rooms

In the statechart diagram below the agent is assumed to be in a room. It identifies three possible states in the superstate where the agent is trying to get to the exit. The three possible states arise due to the three decisions to be made viz;

1. does the agent know the exit door- in this case the agent moves directly to the Moving to exit door state. This state evaluates when the exit is in the room the agent currently is in.
2. The second state is when agent does not have knowledge of the exit and enters the moving randomly superstates. This has two possible states one is where the agent moves from one room to another and checks for the exit. So by default it enters the state in new room. The agent checks for the door, if the door is present and is the exit the exit_door causes a transition to the Moving to Exit Door state. If it is not the exit the agent moves to the Move to inner door state
3. In this state the agent comes out of the door and checks his location if the location is familiar and he knows the location of the exit he moves towards the exit if not he moves to the next room being the one it hasn't entered before.

On exiting from both the states in moving randomly the agent memorizes the room as well as the location.

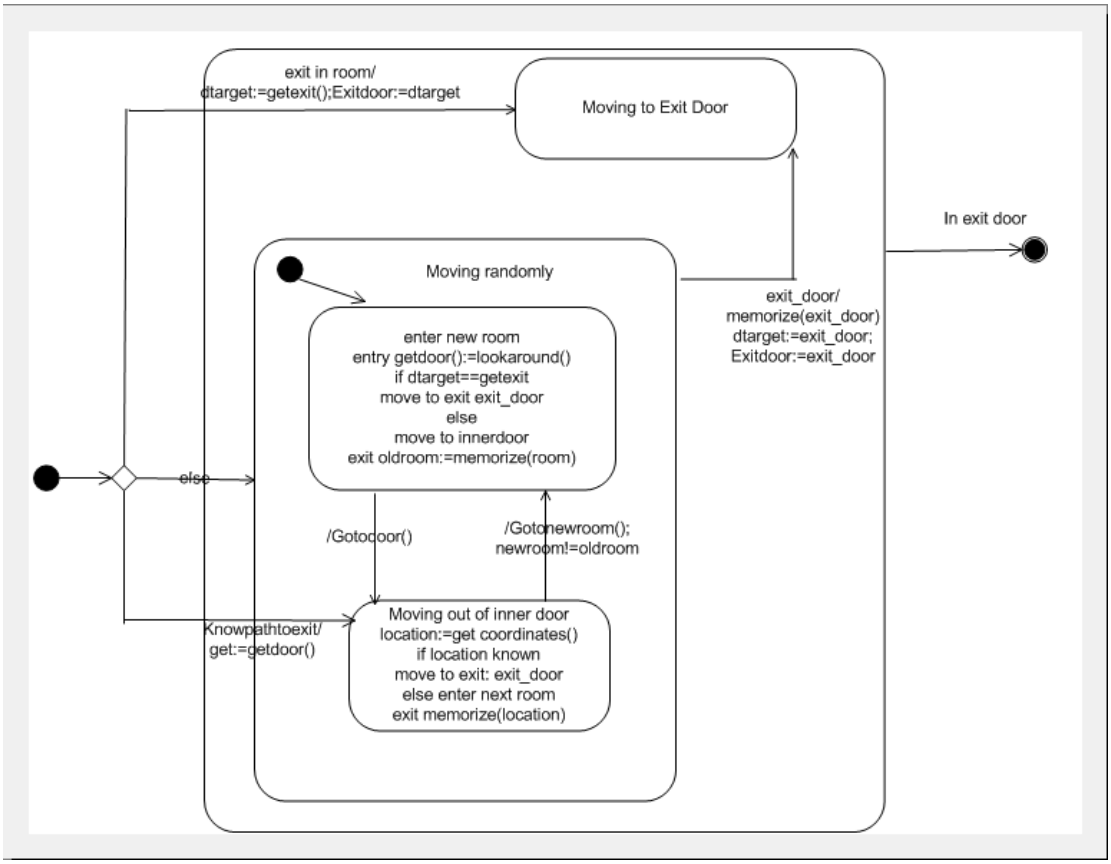


Figure 58 Behavior

Chapter 8: Tradeoff Studies – Cost vs. Performance

Two types of tradeoff studies are considered in the following sections. The first study is that of a card reader system used for access control. It is designed to prevent unauthorized access with the conflicting requirements of minimizing time and maximizing reliability while keeping costs at a minimum. The other study is of positioning sensors in an area so as to ensure redundancy, detect threats with a low fault rate and at a low cost.

8.1 One Dimensional Solution

Using Depth as a main control factor an algorithm was devised to switch the sensors depending on the inputs received. This basically switches on the sensor in a room at a distance of a single door to the room that has been entered. A major premise here is that upon receiving a trigger the sensors of the closest rooms should be alert. The problem was implemented using Visual Basic for Macros in Excel

This is a one dimensional solution using mainly a trigger to operate a group of sensors. For e.g. as per Fig. 58, when an intruder enters a room the nearest sensors will be triggered from their sleep state.

Assume if the object of interest is in R4 and the intruder knows that

Then the potential path of the intruder would be

Entrance-Rear-Room3-Room4

When the entrance button is selected the closest rooms viz; room1, room2 and the rear space are in an alert state. The depth matrix is sampled to get the proximity information. When the rear space door is opened Room 3 Room 5 and the main

entrance are alert. If the door is locked in R4 then there is no reason for the sensor to pick up but as soon as the door in Room 3 is opened the sensor should be switched on as Room 3 controls full access to room 4.

This way the path of an intruder can be tracked by adaptively switching on sensors.

The following illustrations show the status when each room is entered.

Now indicates the sensor is switched on

Wait indicates sensor is in passive mode

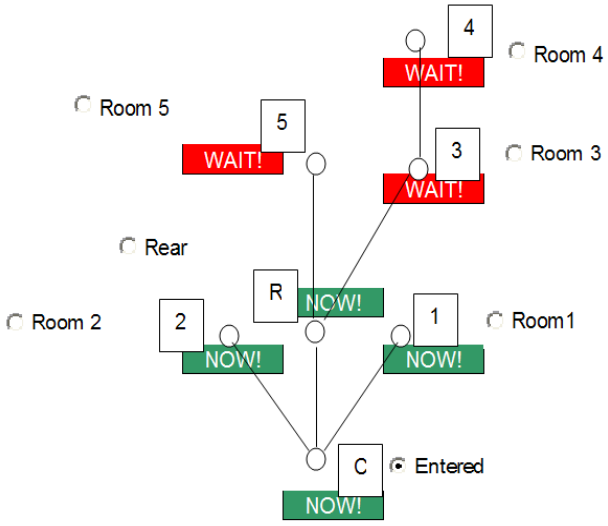


Figure 59 Entrance entered

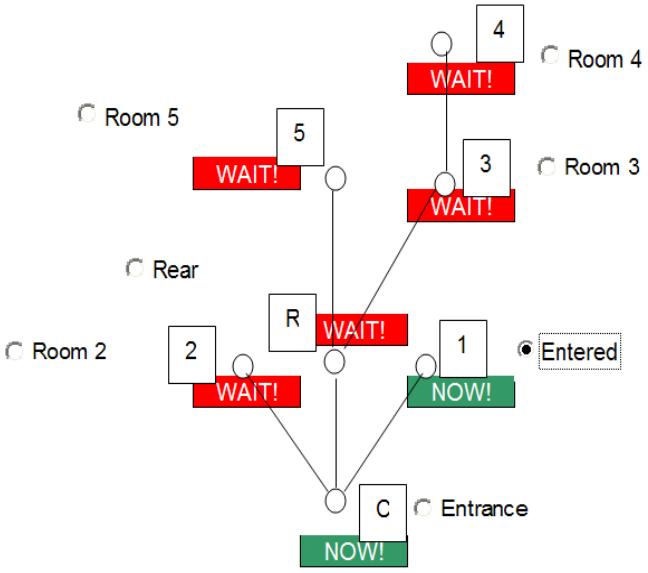


Figure 60 Room1

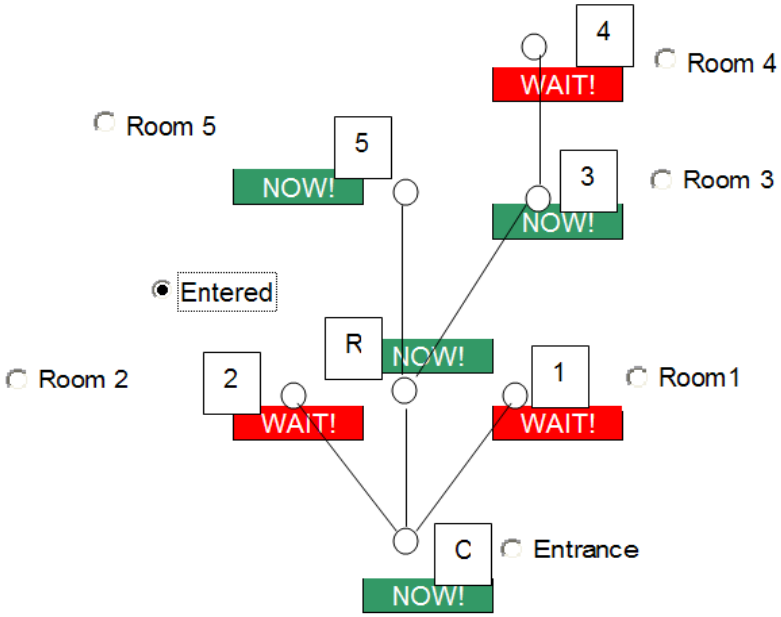


Figure 61 Rear entrance

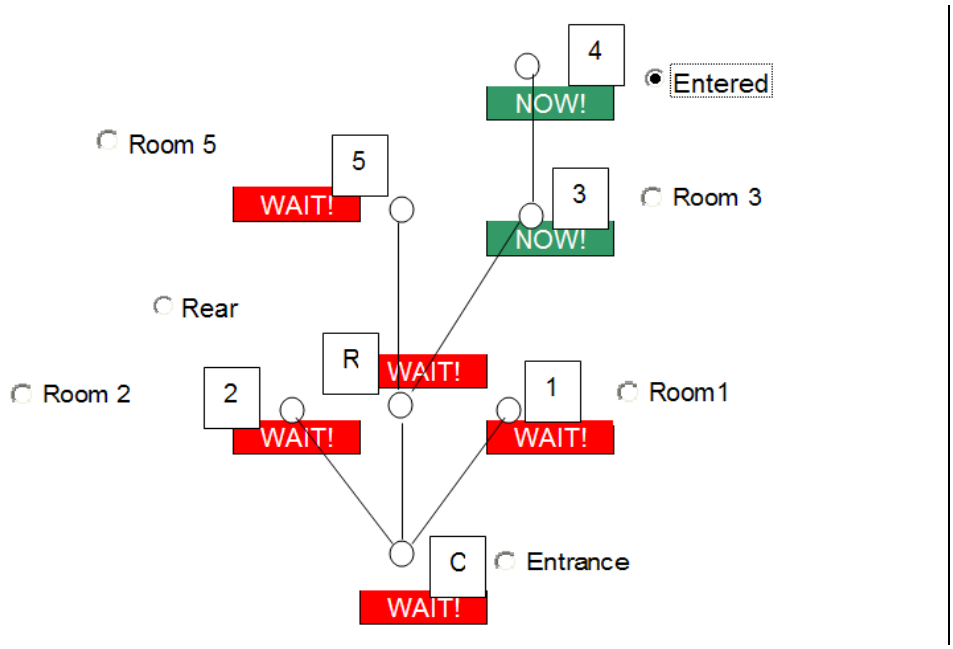
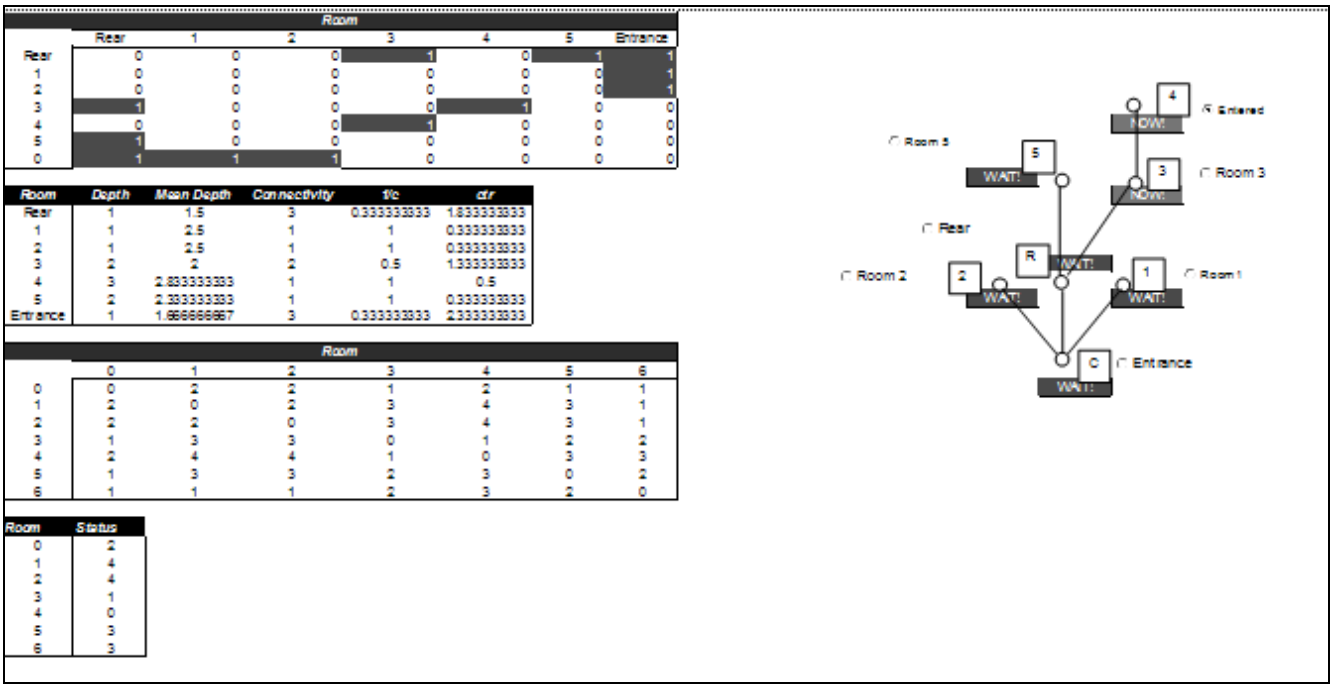


Figure 62 Room 4 entered

The sheet used for calculations and running the macros is shown below



Room	Rear	1	2	3	4	5	Entrance
Rear	0	0	0	1	0	1	1
1	0	0	0	0	0	0	1
2	0	0	0	0	0	0	1
3	1	0	0	0	1	0	0
4	0	0	0	1	0	0	0
5	1	0	0	0	0	0	0
0	1	1	1	0	0	0	0

Room	Depth	Mean Depth	Connectivity	1/c	ctr
Rear	1	1.5	3	0.333333333	1.833333333
1	1	2.5	1	1	0.333333333
2	1	2.5	1	1	0.333333333
3	2	2	2	0.5	1.333333333
4	3	2.833333333	1	1	0.5
5	2	2.333333333	1	1	0.333333333
Entrance	1	1.666666667	3	0.333333333	2.333333333

Room	0	1	2	3	4	5	6
0	0	2	2	1	2	1	1
1	2	0	2	3	4	3	1
2	2	2	0	3	4	3	1
3	1	3	3	0	1	2	2
4	2	4	4	1	0	3	3
5	1	3	3	2	3	0	2
6	1	1	1	2	3	2	0

Room	Status
0	2
1	4
2	4
3	1
4	0
5	3
6	3

Figure 63 Worksheet

Using the logic of closest rooms getting triggered to alert state the results for the floorplan with loops are:

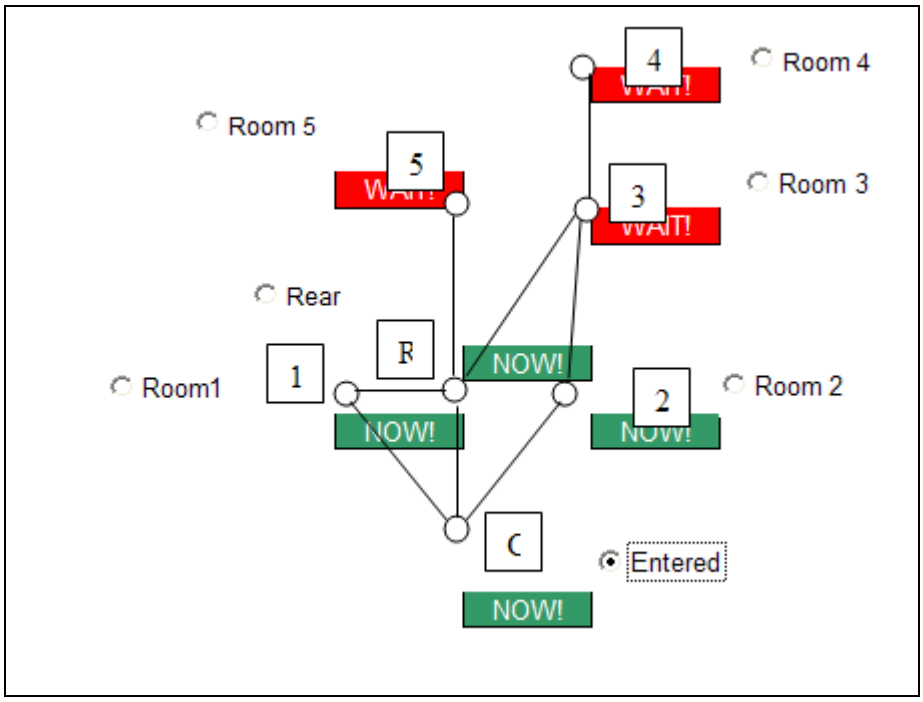


Figure 64 Entrance

When the main door is opened the closest rooms are obtained from the depth matrix and room number 2, 1 and the rear space is triggered to Alert state.

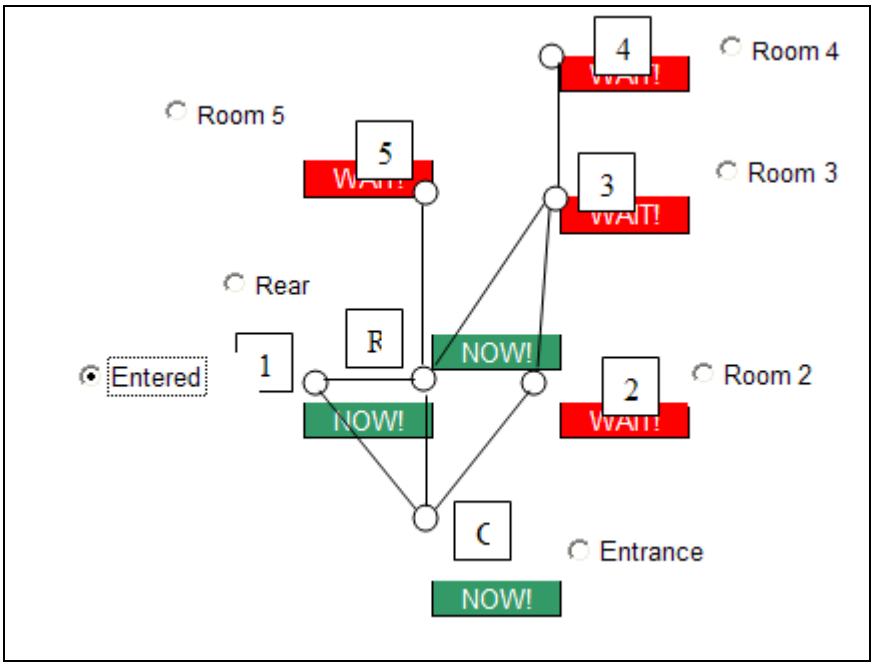


Figure 65 Room 1

When room 1 is entered the closest rooms are obtained from the depth matrix and the area to the main entrance and the rear space is triggered to Alert state.

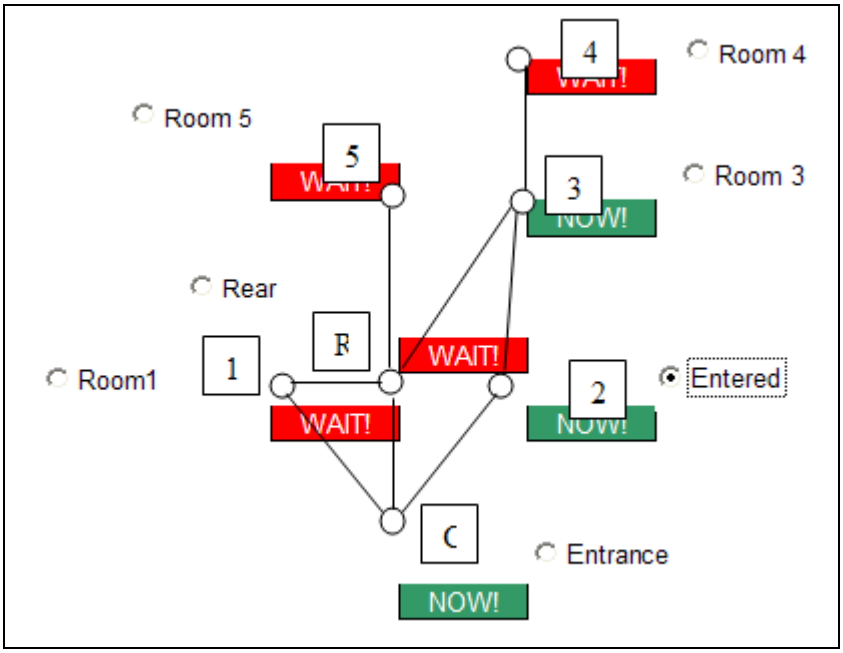


Figure 66 Room 2

When room 2 is opened the closest rooms are obtained from the depth matrix and room number 3 and the main entrance space is triggered to Alert state.

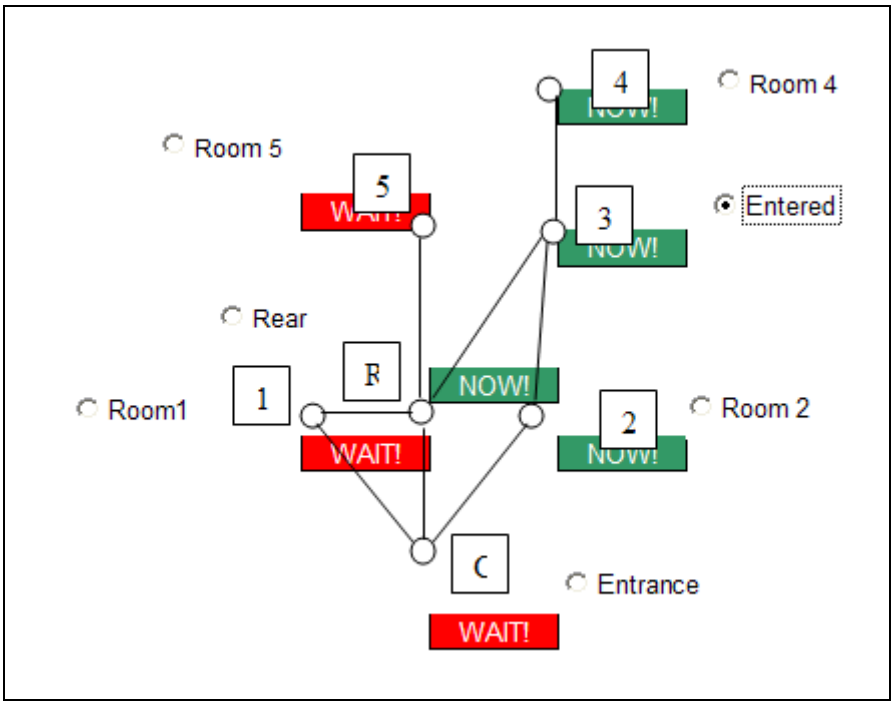


Figure 67 Room 3

When room 3 is entered rear space, room2, room4 are alerted

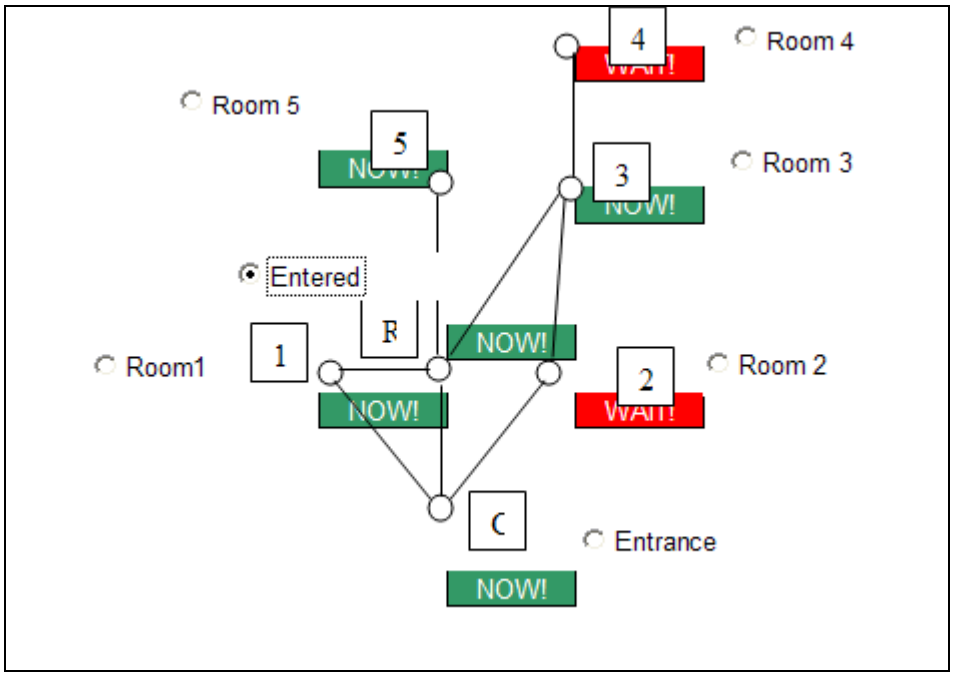


Figure 68 Rear

When the rear space is entered room3, 1, 5, main entrance.

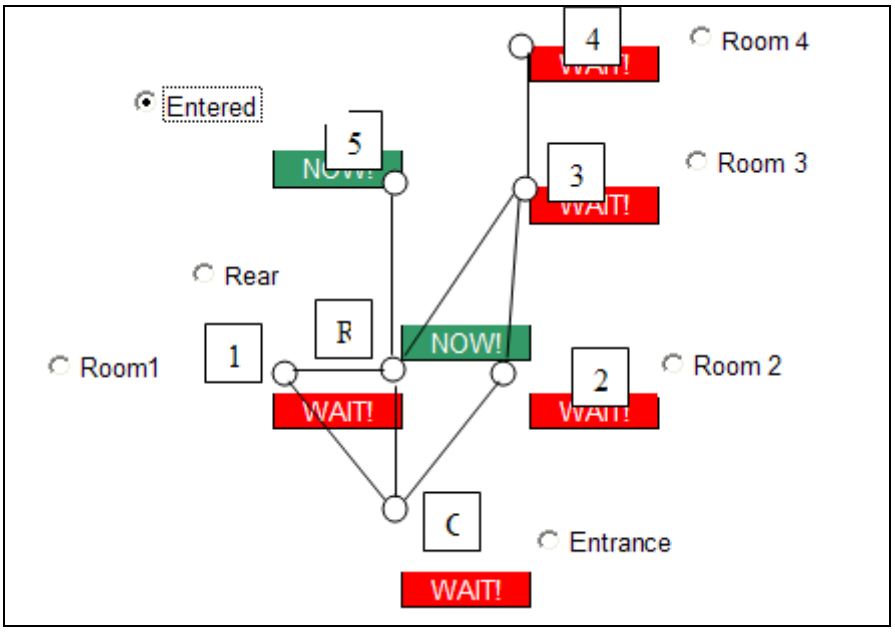


Figure 69 Room 5

When the room 5 is entered rear space is alerted. This is a one dimensional solution to the problem. However there are other operations that can be used to ensure protection against intruders and this can be done using data from multiple sensors.

8.2 Cost-Performance tradeoff sensor network

8.2.1 Introduction

The sensor network considered can be used to monitor the environment, detect, classify and locate specific events, and track targets over a specific region. Examples of such systems are in surveillance, monitoring of pollution, agriculture or civil infrastructures. The deployment of sensor networks varies with the application considered. It can be predetermined when the environment is sufficiently known and under control, in which case the sensors can be strategically hand placed. The deployment can also be undetermined when the environment is unknown or hostile in which case the sensors may be deployed by other means, generally resulting in a random placement.[28]

We consider deployment strategies for sensor networks performing target detection over a region of interest. In order to detect a target moving through the region, sensors have to make local observations of the environment and collaborate to produce a decision that reflects the status of the region covered [29]. This collaboration requires local processing of the observations, communication between different nodes, and information fusion [30]. Since the local observations made by the

sensors depend on their position the performance of the detection algorithm is a function of the deployment.

One possible measure of the goodness of deployment for target detection is called path exposure. It is a measure of the likelihood of detecting a target traversing the region using a given path. The higher the target detection, the better the deployment. The set of paths to be considered may be constrained by the environment.

In this study, the deployment is assumed to be random which corresponds to many practical applications where the region to be monitored is not accessible for precise placement of sensors. The room under consideration is divided into a 10 X 10 grid and random placement of sensors is allowed. The focus of this study is to determine the number of sensors to be deployed to carry out target detection in a region of interest. The tradeoffs lie between the fault rate, the cost of the sensors deployed, and the redundancy used.

8.2.2 Model Used

Consider a 10 X 10 room with n sensors deployed at locations S_i, i = 1…n. A target at location u emits a signal which is measured by the sensors. The signal from the target decays as a polynomial of the distance. If the decay coefficient is k, the signal energy of a target at location u measured by the sensor at s_i is given by

$$Si(u) = K / ||u - s_i||^k \quad (7)$$

where K is the energy emitted by the target and $||u- s_i ||$ is the geometric distance between the target and the sensor. Depending on the environment the value k typically ranges from 2.0 to 5.0 [4]. Energy measurements at a sensor are usually corrupted by noise. If Ni denotes the noise energy at sensor i during a particular measurement, then the total energy measured at sensor i, when the target is at location u, is

$$E_i(u) = S_i(u) + N_i \quad (8)$$

The sensors collaborate to arrive at a consensus decision as to whether a target is present in the region. There are two basic approaches for reaching this consensus: Value fusion and Decision fusion [31].

In value fusion, one of the sensors gathers the energy measurements from the other sensors, totals up the energy and compares the sum to a threshold to decide whether a target is present. If the sum exceeds the threshold, then the consensus decision is that a target is present. In contrast, in decision fusion, each individual sensor compares its energy measurement to a threshold to arrive at a local decision as to whether a target is present. The local decisions (1 for target present and 0 otherwise) from the sensors are totaled at a sensor and the sum is compared to another threshold to arrive at the consensus decision. In some situations, value fusion outperforms decision fusion and vice versa.

8.2.3 Value Fusion.

The probability of consensus target detection when the target is at location u is

$$D(u) = \sum \text{Prob}(E_i(u)) \geq \eta. \quad (9)$$

where η is the fusion threshold. The noise processes at the sensors are assumed to be independent and white Guassian. Due to the presence of noise the sensors may incorrectly decide that the target is present even though there is no target in the field/ The probability of a consensus false target detection is

$$\sum Ni \geq \eta \tag{10}$$

8.2.4 Example

Assumptions:

Assume the area of interest is can be thought of as a 10 X 10 unit^2 grid. The sensor locations will be decided by the permissible range between them which is assumed to be 1 unit. Assume the noise process at each sensor is Gaussian with mean 0 and variance 1. Further assume that the sensors use value fusion to arrive at a consensus decision. Then, from Equation 2, we chose a threshold $\eta = 3.0$. The target emits an energy K = 12 and the energy decay factor is 2. The probability of detection is computed using equation 7. The constraints to this problem are that cost has to be considered while deployment. The cost includes the cost of the sensor as well as the cost of deployment. Assumptions made here are that if the sensor is closer to the target its cost is higher. This allows the two constraints cost and detection to fight each other as they would be inversely related.

The tradeoff study is defined as:

Objective function:
Min ∑ Noise so that reading is as accurate as possible

Constraints -
Detect Target -∑ probabilities should be ≥ threshold
Cost constraints - Cost of Deployment ≤ budget

8.2.5 Problem Formulation:

Sensor distance matrix					
	1	**2**	**3**	**4**	**5**
1	0	5.157832	5	9.181756	5.713798
2	5.157832	0	9.567374	5	5
3	5	9.567374	0	12.21903	7.343005
4	9.181756	5	12.21903	0	5.068564
5	5.713798	5	7.343005	5.068564	0

Table 29 Matrix

The inter sensor distance is calculated to specify the range each sensor must have while the solver evaluates different sensor positions. This prevents one sensor to be placed on another or on the origin which would be the target position. The distance from the target is the geometric distance and the detection probability is derived from equation 1 while the noise is determined as a Gaussian distribution and a property of the sensor. The more the distance the greater the noise. But the closer the sensor the more the cost which is inversely proportional to the distance from the target.

Select Sensor	Sensor	Cost of Deployment	Sensor Position		Target Position		Distance from Target	Detection Probability	Noise
0	1	894.427191	10.0000	5.0000	0	0	11.18033989	0.096	0.065118504
0	2	1468.39129	6.7353	1.0069	0	0	6.810173874	0.258740758	0.087603843
1	3	707.1067812	10.0000	10.0000	0	0	14.14213562	0.06	0.060765169
1	4	4992.893219	1.7353	1.0000	0	0	2.002846759	2.991477924	0.244032767
0	5	1453.360017	4.2984	5.3728	0	0	6.880607613	0.253470641	0.086802406
2									

Decay Coefficient	2
Energy from target	12

Constraints

Budget	Cost of Sensor	<=	$
	5700		5700

Objective Function

False Alarm	False alert	<=	threshold
	0.304797936		0.187

	Sumdetect	=>	Threshold
	3.051477924		3

Figure 70 Trade off sheet

The select sensor column is a Boolean variable that allows for the selection of the sensor. The number of sensors selected can be set as a constraint as the amount of redundancy required. As the redundancy increases the performance improves as the detection is the sum of the individual sensor measurements, but the cost and likewise noise also increased. Linear programming is used to minimize the noise by varying the locations of the sensors and without violating the constraints of cost, inter-sensor range and integer values for selection. The results of the runs can be found in the appendix; the graph below summarizes the relation between cost, threshold for detection and the noise. These results were obtained by keeping range constant and limiting the sensor selection to 2.

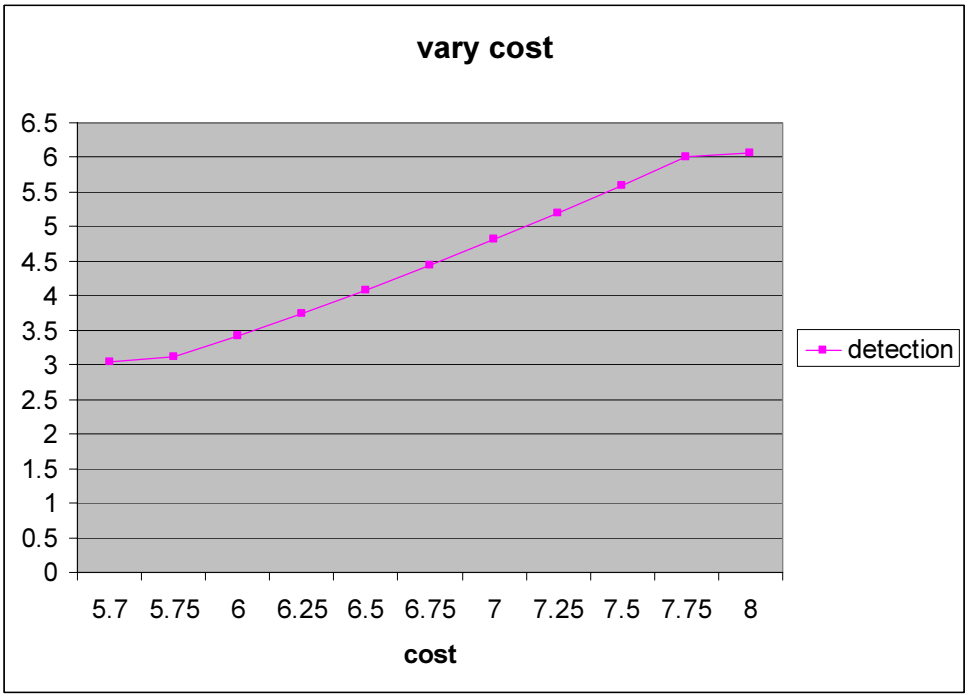


Figure 71 Detection vs. Cost

Since the detection improves as the sensors are placed closer to the target which is possible only at a higher cost the curve for detection vs. cost goes upwards with cost increase. While noise drops as cost increases as the sensors are closer to the target and give better performance.

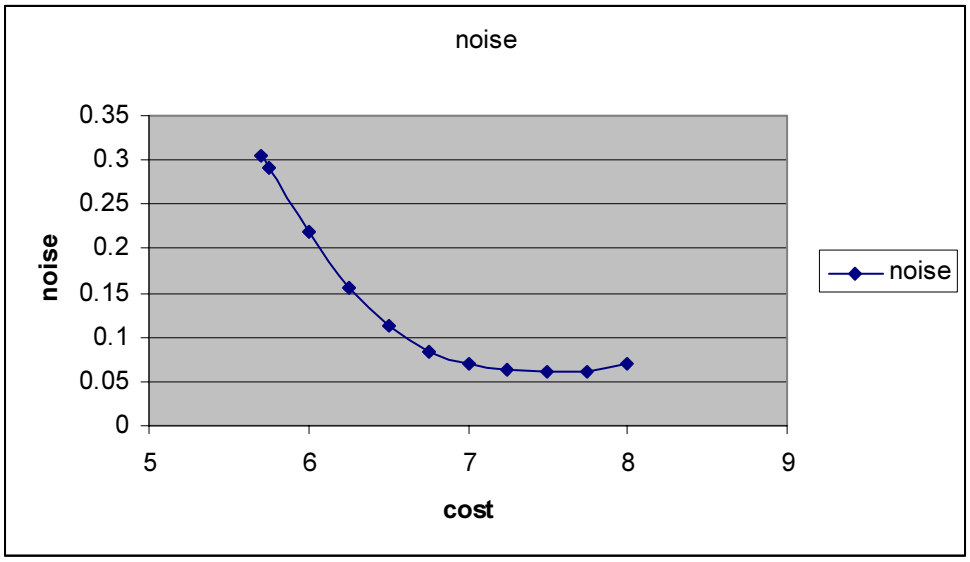


Figure 72 Noise vs. Cost

A second study was carried out by varying the number of sensors that should be used. Here the number of sensors was varied from 1 to 5 and the cost constraint allowed 4 and beyond sensors to be simultaneously introduced at the higher ends of the budget.

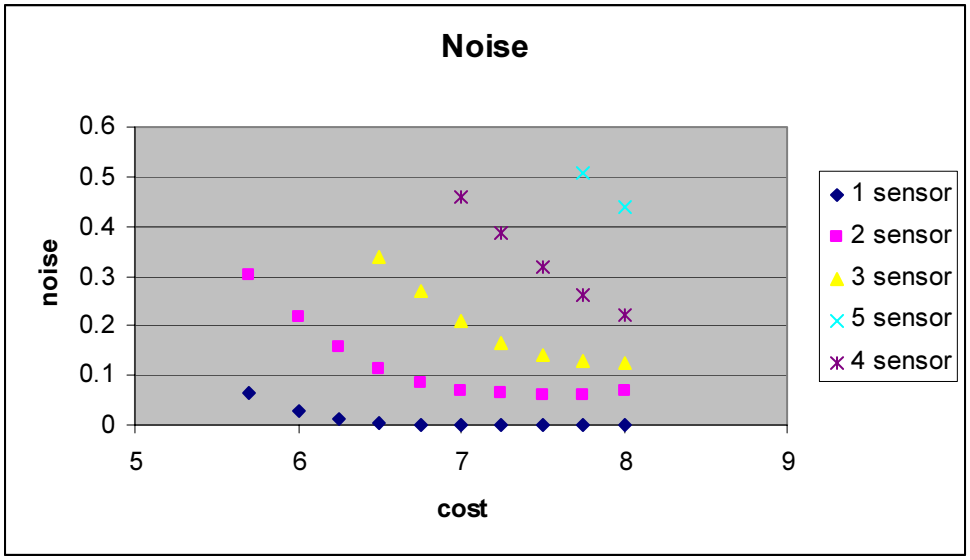


Figure 73 Noise vs. Cost -Vary number of sensors

As the chart indicates noise levels reduce considerably with just a single sensor as it would e placed closer to the target. However as the number of sensors increases the

cost does increase too but the performance deteriorates. So in this case having more number of sensors is not a good tradeoff. The higher curves show the poor performance as compared to the lower ones viz the ones limited to 1 and 2 sensors. The graph of threshold vs. cost attests the statement above.

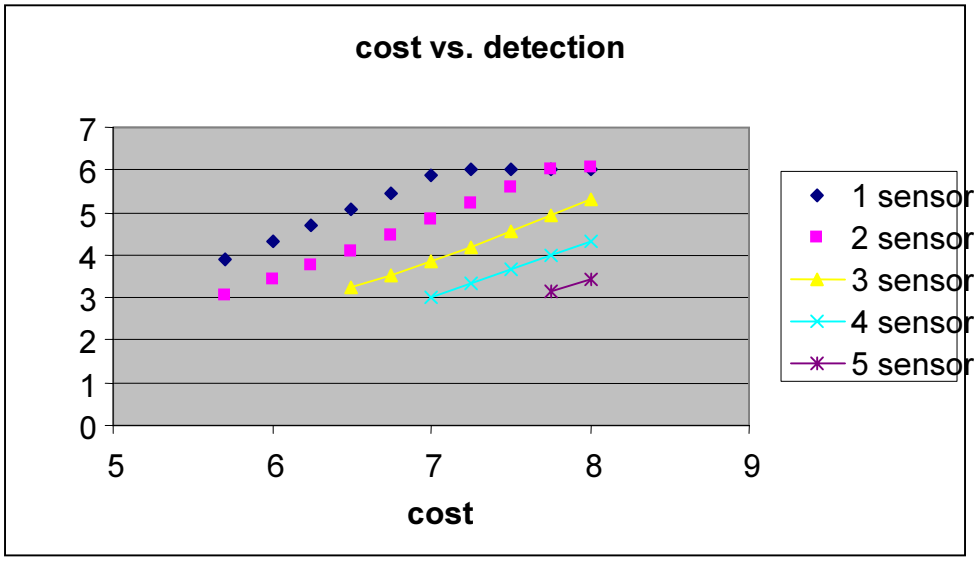


Figure 74 Detection vs. Cost -Vary number of sensors

8.2.6 Conclusion

The tradeoff study can be formulated with any assumptions and many other constraints. However for the given problem keeping in mind redundancy as a major criteria and cost the next the curve for 2 sensors is selected as the optimal operating curve. Two sensors assure redundancy good performance at a low cost. The selection of the Pareto point on the performance graphs is now trivial. From the graph of cost vs. noise we see that the point that can be considered is

Point No	*Noise*	*Cost*	*Detection*
1	0.08414	6.75	4.442
2	0.069	7	4.82
3	*0.06367*	*7.25*	*5.197*

4	*0.0615*	*7.5*	*5.59*
5	*0.0609*	*7.75*	*6.0123*
6	0.069	8	6.06

Table 30 Conclusion

Point # 5 gives a very good performance at a higher price while point number 4 is a good tradeoff between the two conflicting requirements and hence that becomes our Pareto point. This point is not the optimal point for the design but for the given data and constraints this is the most feasible point that satisfies all constraints.

Extending this solution to combining logic using sensor probabilities;

Consider the case when an entry is attempted at C, there are two sensors combined to explain the type of entry. Say sensor S_1 is good at detecting an entry alone but not the type of entry (forced/authorized) whereas sensor S_2 can indicate the type of entry but has overall poor detection capabilities. Consider a single state **x** of the system can take on one of two values:

. x_1: authorized entry has been attempted.

. x_2: unauthorized entry has been attempted

. x_3: no entry has been attempted

Sensor1 observes **x** and returns three possible values:

. z_1: Observation of an authorized entry.

. z_2: Observation of an unauthorized entry.

.z_3: No entry observed.

The sensor model for sensor 1 is described by the likelihood matrix $P_1(z \mid x)$:

	z_1	z_2	z_3
x_1	0.45	0.45	0.1
x_2	0.45	0.45	0.1
x_3	0.1	0.1	0.8

For a fixed state, it describes the probability of a particular observation being made (the rows of the matrix). For an observation it describes a probability distribution over the values of true state (the columns) and is then the Likelihood Function Ë(x).

A second sensor which makes the same 3 observations as the first sensor but whose likelihood matrix is P_2 (z_2|x) is described by

	z_1	z_2	z_3
x_1	0.45	0.1	0.45
x_2	0.1	0.45	0.45
x_3	0.45	0.45	0.1

whereas Sensor S_1 is good at indicating an entry has been attempted; sensor S_2 indicates the type of entry and has overall poorer detection capabilities.

With a uniform prior, when an observation $z=z_1$ is made the posterior is the first column of the likelihood matrix of sensor 1 and 2

$$P(x|z_1) = (0.45, 0.1, 0.45)$$

Overall likelihood function for the combined information from both the sensors is

$$P_{12}(z_1, z2|x) = P_1(z_1|x)\, P2(z_2|x)$$

Observe z1=z1 and z2=z1 and assuming a uniform prior the posterior is

$$P(x|\, z1, z1) = \alpha\, P12(z1, z1|x) = \alpha\, P1(z1|x)*P2(z1|x)$$

Taking log

$$= (L(z_1|x)) + L(z_1|x) + C$$
$$= (-0.7985,-0.7985,-2.3026) + (-0.7985,-2.3026,-0.7985) + C$$

= (-1.597,-3.1011,-3.1011) +C
= (-0.3680,-1.8721,-1.8721)

Where the constant C=1.299 is found through normalization (which in this case requires the anti-logs sum to one)

Here sensor S_2 adds information to help discern the type of entry into the area of interest and significantly adds value to information provided by sensor S_1.

For the example considered the strategy devised is to use sensor fusion to cover the map while keeping cost low and accuracy high. We combine information from two inexpensive sensors that give a rough estimate of the position and information from highly accurate expensive sensors to correctly locate the target. The tradeoff is as follows

Objective function

Minimize error

By changing *Location of expensive sensors*

Constraints

Cost < Budget
Detection => threshold
Inter-sensor distance => unit

For the inexpensive sensors; we have two sensors which measure position of the target. The sensor measurement has a given confidence. For the sensors the center, or mean, of the distribution is the estimated location of the object and the standard deviations along the major and minor axes of the distribution correspond to estimates of the uncertainty (or noise) in the observation along each axis. The distribution

corresponds to the conditional probability that the object is in any location, given the observation.

For S_1 (sensor 1); the probability of estimation of S_1 in the x-axis is 88% and that in the y-direction is 40%. Sensor S_2 has an estimation of 50% and 95% respectively.

$\sigma_{x1} = 2$ and $\sigma_{y1} = 3$

For S_2 (sensor 2)

$\sigma_{x2} = 3$ and $\sigma_{y2} = 1$

Provided two observations are independent and drawn from normal distributions, the observations can be merged into an improved estimate by multiplying the distributions.

- Sensor measurements: z_1 z_2 with covariance matrices P_1 P_2 from two different sensors
- **Solution:** Optimal estimate x with minimal combined covariance matrix P:

$$\boxed{\begin{aligned} \vec{x} &= \frac{P_2}{P_1 + P_2}\vec{z}_1 + \frac{P_1}{P_1 + P_2}\vec{z}_2; \\ P &= \frac{P_2}{P_1 + P_2}P_1; \end{aligned}}$$

Based on this equation for a target position of (2, 5) we get an estimate of (2.876, 4.6875) with a deviation of σ (1.2, 0.75) Combining this information with that obtained from the expensive sensors (see section on Sensor fusion for tradeoff with sensors); we get a revised estimate of the error in K by adjusting the previous error $\sigma_x^{2-}(t_k)$ with $\sigma_{z(t_k)}^2$ obtained from the second set of sensors.

$$K = \frac{\sigma_x^{2-}(t_k)}{\sigma_x^{2-}(t_k) + \sigma_{z(t_k)}^2};$$

The resulting estimate is now adjusted with estimate from the expensive sensors and values obtained from the expensive sensors.

$$\boxed{\begin{aligned} \hat{x}(t_k) &= \hat{x}^-(t_k) + K \cdot \left[z(t_k) - \hat{x}^-(t_k) \right] \\ \sigma_x^2(t_k) &= \left[1 - K \right] \cdot \sigma_x^{2-}(t_k) \end{aligned}}$$

Based on this equation for a target position of (2, 5) we get a corrected estimate of (2.0015, 4.99913) with a deviation of σ (0.002069, 0.00207)

Initially the optimum solution of the location is found for target position (0, 0). After which a sensitivity analysis was done to see which of the positions satisfies all target position from $n_x = 0 \ldots 10$ and $n_y = 0 \ldots 10$

The tradeoff is solved for all target positions by maintaining some positions constant and reducing the degree of freedom, until a feasible set of locations is reached that gives the best solution for the given problem requirements and constraints. This set is (1.083, 6.153), (5.74, 4.32), (2, 1), (5.028, 9.273), (10, 10) while maintaining cost at 13172.00517 $ within the limit of 15000 $.

location	cost	select	sx	sy	energy	confidence
4.574430873	1600.512194	1	1.0830742	6.153409847	0.573465252	0.144216559
3.58915172	1392.250234	1	5.7359904	4.323008056	0.931531643	0.225428814
1.414213562	4472.135955	1	2	1	6	0.00013383
7.549986051	948.0523222	0	5.0277343	9.272591202	0.21051786	0.080454643
10.63014581	707.1067812	1	10	10	0.10619469	0.066391396
	8172.005165	**4**			***7.611191585***	***0.4361706***

Figure 75 Expensive Sensor

P-stdev (old)	combined	1.2	0.75
S1measurement		5.34	1.02
S2measurement		2.9	3.69
S1estimate		2.136	0.765
S2 estimate		1.74	0.9225
estimate (old)	combined	3.876	1.6875
correction		3.001386589	1.999209319
New P		0.001899437	0.001897634
		0.001386589	-0.000790681

Figure 76 Inexpensive sensor

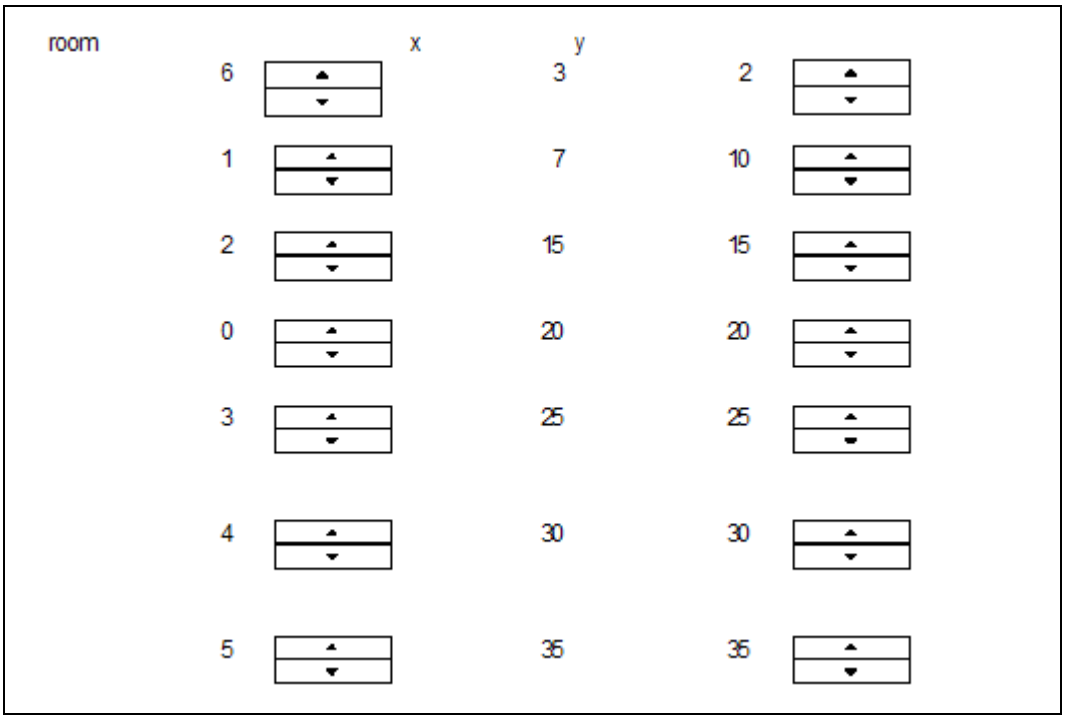
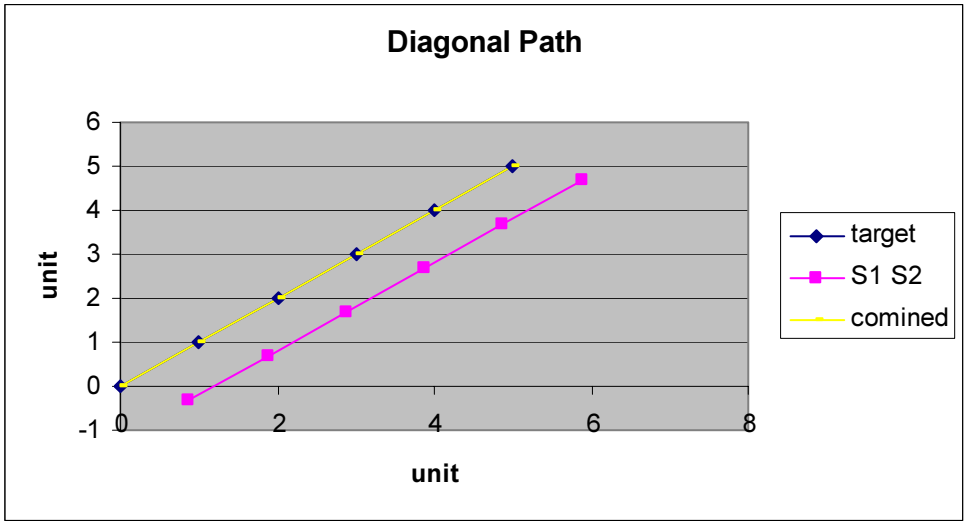


Figure 77 Buttons to maneuver in the area

Note: In the graph below there exists an error between the sensor output and the actual location. However the error is small of the order of 10^{-3} and cannot be effectively depicted on this scale.



8.2 Card Reader system

8.2.1 Measures of Effectiveness

An Intelligent Sensor Network for the protection of a building serves to protect the building from any untoward incident. It should thwart any possible threat to the system. Though the potential threats to such a system are numerous only one type of threat (Unauthorized access) is considered in this section. The Measure of Effectiveness for such a system should include the system reliability and accuracy. It should also include the level of redundancy incorporated in the system to reduce potential danger due to a sub system outage or malfunctioning. Since protection of system is the chief concern the cost analysis takes a lower priority in the analysis.

8.2.2 Performance characteristic

- **Minimizing the Cost:** The system should provide maximum benefits such as tamper proof environment resistant, built in back up power supplies etc at minimum cost.
- **Maximizing the reliability** would concentrate on features of the system such as tamper proof, environment resistance. It would also depend on the redundancy built into the system
- **Minimizing the time** required to clear a single employee: This is a direct measure of the speed of the system. If the system takes considerable time in clearing one employee the idea might not be feasible

8.2.3 Decision Variables

- **Time taken by system look up:** The time taken by the system is the service rate of the readers. This, in the current study is a choice between two manufacturers of card readers.
- **Number of card readers:** This variable decides the cost, system reliability and also the time taken to clear an employee. If the number of readers is more than the queue lengths are smaller but the reliability is lower and the cost is higher.
- **Number of engineers:** This variable decides the cost of the system

8.2.4 Formulation:

1. Minimize the cost

   The cost is decided by:

- Cost of The reader and scanner
- Choice between 2 makes (satisfying requirements) X1, X2 (Boolean variables giving choice between the two makes) Cost of backup power supply: Redundant supply available choices two makes Y1 and Y2 (Boolean variables giving choice between the two makes)
- Cost of software for networking capabilities; K1&K2(Boolean variables giving choice between the two makes)
- Cost of engineers developing database

  W= Number of engineers deployed. Assume that engineers are paid @ 25$/hour. Number of hours= 4

Total Cost =R* (X1D1+X2D2) + (Y1*DB1+Y2*DB2+K1*X1+K2*X2+ 25* 4* W
where
R= number of card Readers

2. Maximize Reliability

   Reliability of card reader = $R_{cr,}$ Reliability of scanner= $R_{scan,}$ Reliability of card= R_{card} . Reliability of tamper proof covering= $R_{tap,}$ Reliability on adverse environmental conditions= R_{env}

Reliability (maximize)

Reliability= $(R_{card})^{(R*X)} * (R_{battery})^{(R*Y_j)} * (R_{comm})^{(R*K_j)}$ (11)

➔reliability is the product of independent component reliabilities.

with X_i =Boolean variable giving manufacturer; i= 1,2; Y_j =Boolean variable giving manufacturer; j= 1,2; With k_i =Boolean variable giving manufacturer; i= 1,2

3. Minimize Time,

Minimize Time taken to clear one employee

$$0.5*(ca^2+cp^2) *(\rho)^ ((sqrt (2*R+2)-1)) /(\mu*(1-(\rho))*R)< N1 \qquad (12)$$

8.2.5 Mapping temporal behavior: Queuing models

The table below records the activity in a building. We use this data to get the inter arrival times for employees. Typically as in any other building the activity is marked by peaks when people arrive in the morning and leave in the evening. In between these peaks are those characterized by lunch breaks or early leavings. This data is a sample for a day, averaged from data collected over a month and is used for this study

Hour	# of people	Arrival Rate
600-700	75	0.020833333
700-800	80	0.022222222
800-900	64	0.017777778
900-100	14	0.003888889
1100-1200	15	0.004166667
1200-1300	48	0.013333333
1300-1400	65	0.018055556
1400-1500	45	0.0125
1500-1600	10	0.002777778
1600-1700	20	0.005555556
1700-1800	70	0.019444444
1800-1900	60	0.016666667
1900-2000	34	0.009444444
Mean	46.1538462	0.012820513
Stdev	25.09929	0.006972025

Table 31 Activity in a building

Mean arrival rate= 0.012820513

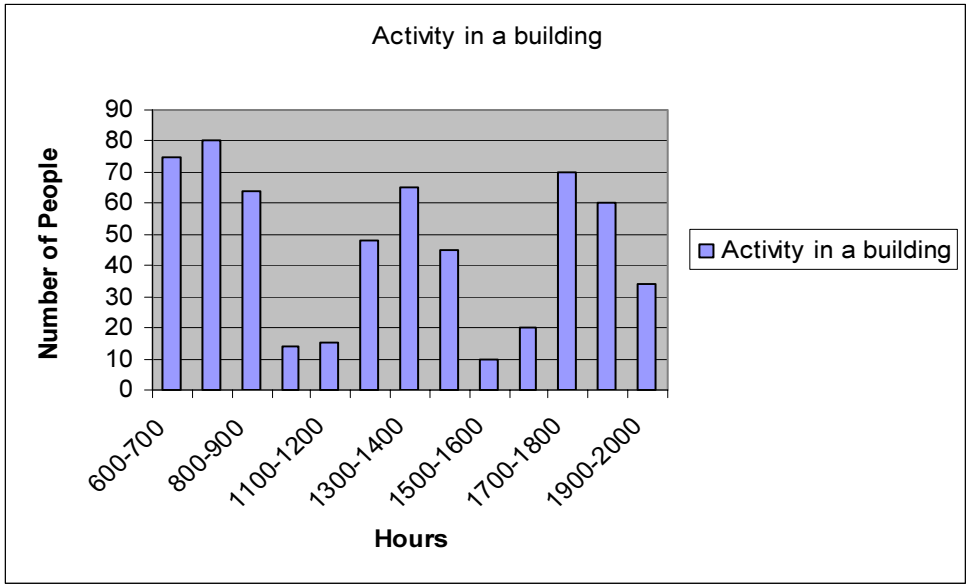


Figure 78 Histogram

Having modeled the activity of the building and having arrived at an inter arrival time we need to develop a model that will capture the interarrival behavior and the number of servers that need to be installed to cater to the employees in order to minimize waiting times. There are two major factors in the system:

- Cost of providing service: cannot afford many idle servers.
- Cost of employee waiting time: employee will have to wait in longer queues

what we study here is a tradeoff between these two factors.

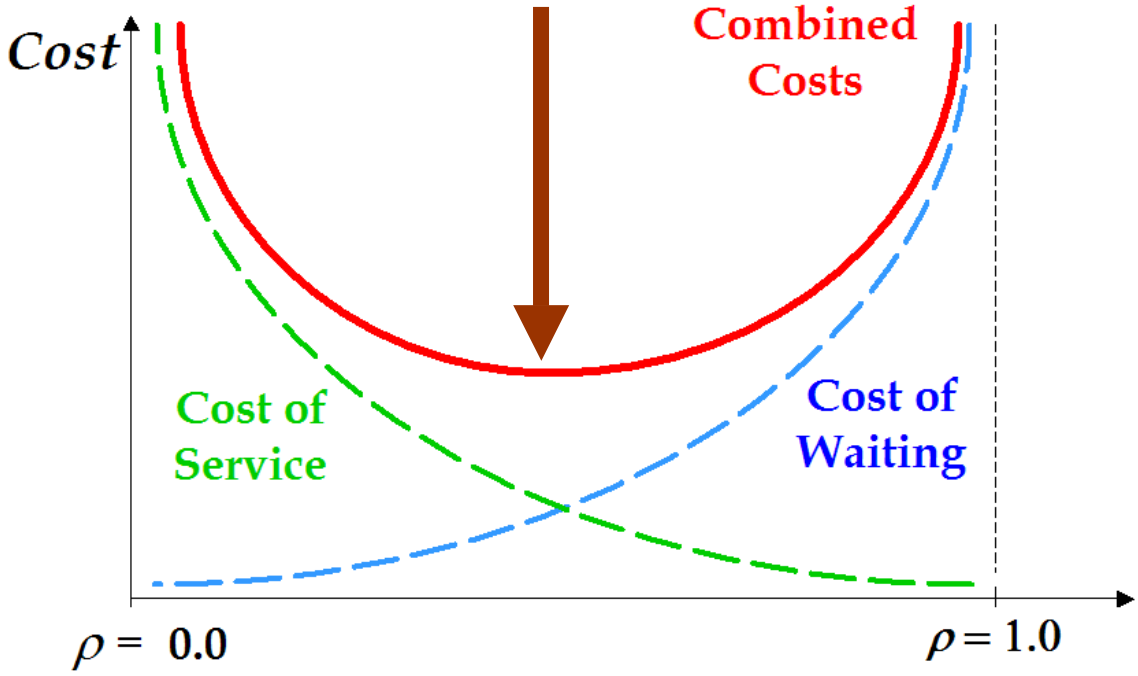


Figure 79 Tradeoff point

The figure shows the tradeoff between two conflicting requirements and the point that serves as an effective compromise between the two requirements. This point is not the optimal point but a point that provides *a feasible* solution which satisfies all the requirements. For the temporal behavior queuing theory is used to characterize the behavior of the card system. A parallel is drawn between the clients as the employees trying to gain access and the card readers as the server.

Characteristics of Queue models[32]

1. Calling population
    - infinite population: leads to simpler model,
    - Finite population: arrival rate is affected by the number of employees already in the system.

2. System capacity
    - The number of employees that can be in the queue or under service.
    - An infinite capacity means no customer will exit prematurely.
3. Arrival process
    - For infinite population, arrival process is defined by the interarrival times of successive customers
    - Arrivals can be scheduled or at random times
4. Queue behavior describes how the customer behaves while in the queue waiting balking - leave when they see the line is too long; renege - leave after being in the queue for too long; jockey - move from one queue to another
5. Queue discipline
    - FIFO - first in first out (most common)
    - FILO - first in last out (stack)
    - SIRO - service in random order
    - SPT - shortest processing time first
    - PR - service based on priority
6. Service Times
    - random: mainly modeled by using exponential distribution or truncated normal distribution (truncate at 0).
    - Constant
    - Service mechanism describes how the servers are configured.

- Parallel - multiple servers are operating and take customer in from the same queue.
- Serial - customers have to go through a series of servers before completion of service
- combinations of parallel and serial.

Characteristics of the model also include values for

·λ arrival rate (in customers per time unit)

·m service rate of one server (in service/transaction per time unit)

Performance metrics

·ρ average utilization factor, percentage the server is busy.

·L_q average length of queue

·L average number of employees in the system

·W_q average waiting time in queue

·W average time spent in the system

·P_n Probability of n employees in the system

Utilization i.e. the fraction of time the server is busy is defined as a ratio of the arrival rate to the service rate

Utilization = ρ=arrival rate/service rate **(13)**

For this study we assume a G/G/ k model which is a model that has a general arrival distribution a general service pattern and has multiple servers. We do this to characterize the arrival distribution as normal as the curve can be approximated to a normal distribution.

Assumptions

General interarrival time distribution with mean **m** and std. dev. = s_a

General service time distribution with mean **m** and std. dev. = s_p

Multiple servers (k)

First-come-first-served (FCFS)

The equation below is the model used to calculate the waiting time of the employee when there are k readers in the building.[33]

Average waiting times (approximate) (14)

$$W_q \cong \left(\frac{c_a^2 + c_p^2}{2}\right)\frac{\rho^{\sqrt{2(k+1)}-1}}{\mu \cdot k(1-\rho)} \qquad c = \frac{s}{m}$$

As per the model the waiting time increases with square of arrival or service time variation, it decreases as the inverse of the number of servers. In the study:

Service rate is defined as time taken for server to complete one transaction= 1sec in this case. The service rates provided by 2 manufacturers are 1 and 2.5 secs. If we assume that the total time taken to clear one employee is 15 secs including employee fumbling and blundering then it takes 15 secs for Manufacturer 1 to clear an

employee and 17.5 seconds for manufacturer 2. In this case the utilization factor is 0.019

Tradeoff study:

Objective function
Min Cost

Constraints
Time<N1
Reliability >N2
All parts (power supply, card, reader, communication) should be bought from one manufacturer

The model was run by varying time and reliability. Reliability was varied only between two points 80 and 90% as a probability lower than this is unacceptable. From the table below the Pareto point is identified as #7 and # 8. Since # 7 satisfies both the requirements selection in this case is easy. During the runs certain infeasible solutions were also recorded as indicated with an *. These points provide a solution with a constraint violation that of resulting in a fractional value for the reader.

Runs	Constraint(<Time &>Rel)	x1*y1*k1 R	C		W=Wq+1/mu	T	Re
1	<2.2&80	1	1	2.9	15.53339	0.53339	0.9702
2	<2&90	1	1	2.9	15.53339	0.53339	0.9702
3	<1&80	1	1	2.9	15.53339	0.53339	0.9702
4	<1&90	1	1	2.9	15.53339	0.53339	0.9702
5	<0.5&80	1	2	5.6	15.11337	0.113368	0.9413
6	<0.5&90	1	2	5.6	15.11337	0.113368	0.9413
7	**<0.05**	**1**	**3**	**8.3**	**15.039**	**0.039**	**0.91**
8	**<0.03**	**1**	**4**	**11**	**15.01645**	**0.01645**	**0.886**
9	***<0.03& 90 ****	1	***3.27***	***9.05***	15.03	***0.03***	***0.9***
10	***<0.02&90****	1	***3.48***	***9.6***	15.025	***0.025***	***0.9***
11	<0.02&80	1	4	11	15.0165	0.0165	0.886
12	<0.01&80	1	5	13.7	15.0079	0.0079	0.86
13	***<0.01&90****	1	***4.666***	***12.8***	15.0099	***0.0099***	***0.87***
14	***<0.005&90****	1	***3.48***	***9.6***	15.025	***0.025***	***0.9***
15	<0.005&80	1	6	16.4	15.00416	0.00416	0.83
16	<0.003&80	1	7	19.1	15.0023	0.0023	0.8
17	***<0.001&80****	1	***7.38***	***20.11***	15.0019	***0.0019***	***0.8***

Table 32 Result

**Infeasible*
Possible solution point 7 and 8

The graph chalks out the relationship between the requirements. As is seen from the first graph the relationship of cost vs. time is one which gives the best result at the highest cost. The cost though has to be kept lower hence a point that ensures the waiting time is low and the cost is reasonable has to be picked. Points 7 & 8 are high lightened on the plots as these serve as effective solutions for the given constraints.

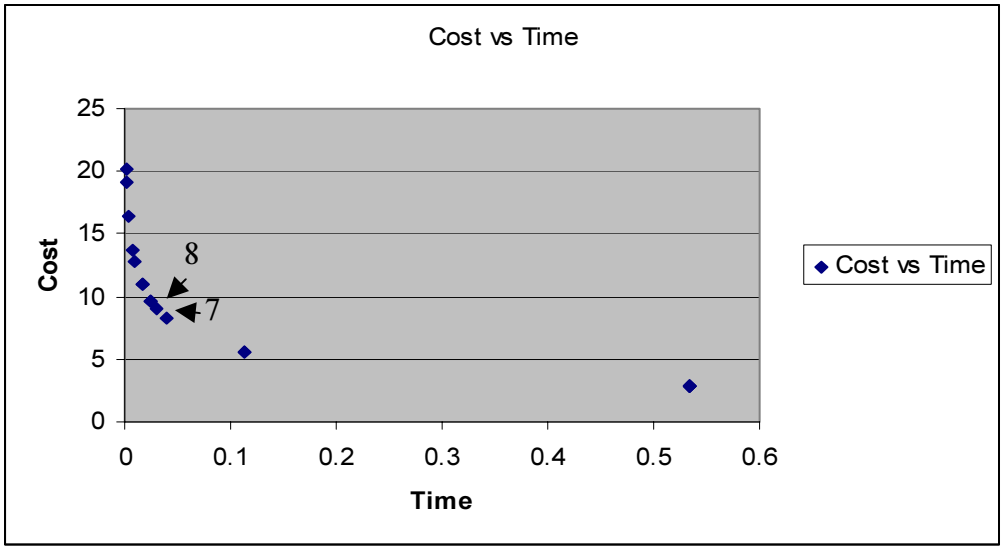


Figure 80 cost vs. Time

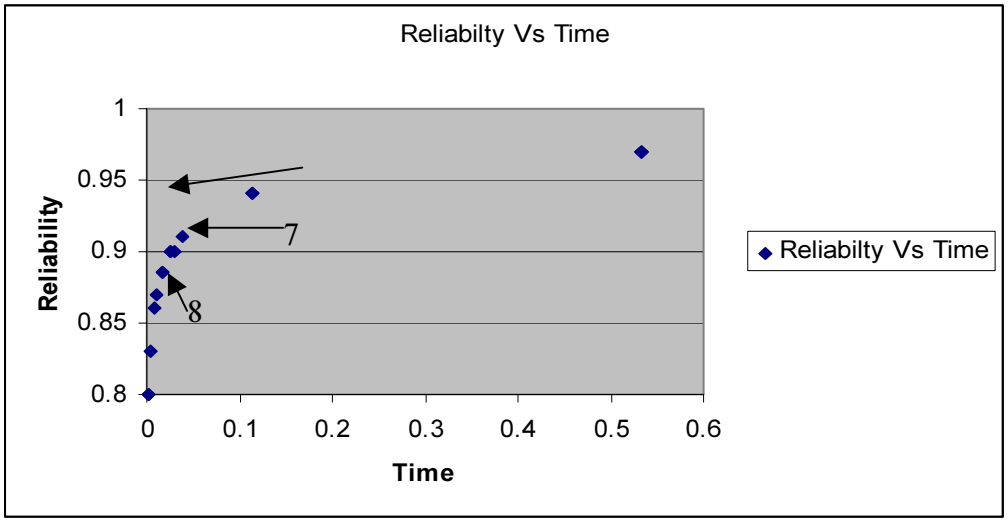


Figure 81 reliability vs. Time

To further help the analysis a plot of analysis vs. time and reliability vs. cost is plotted the

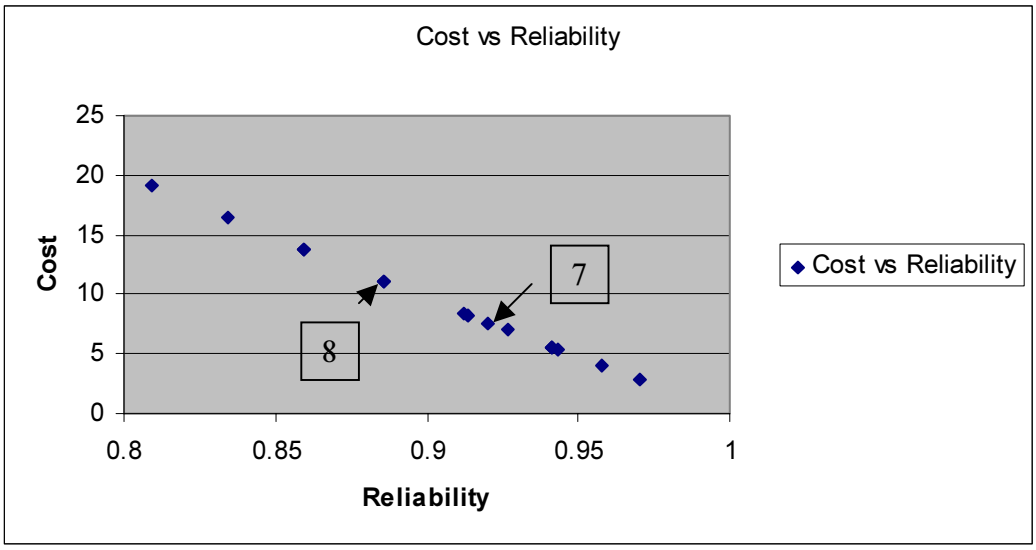


Figure 82 cost vs. reliability

8.2.5 Conclusion

The Pareto point is decided on the basis that it gives the best results for all three objectives. Since reliability is an important criterion in access control system it is important to maintain reliability at least< 0.90. Also a system that doesn't give a reliability of 0.9 might incur additional losses in repair and downtime. The Cost has to be kept low, however for an identification system cost is not the most important criteria. Also the number of readers in a system should be such that the queue waiting times are within specifications. Keeping these conditions in mind the Pareto point of time <0.05&reliability >90 giving a result of Number of readers: R= 3; Cost= 8.3; Time= 0.039; Reliability= 0.91 is the best solution for the given data.

Chapter 9 Conclusion

The thesis thus supplies a framework that can be adapted while designing a security system for a building. A systematic step-by-step approach plugs in different aspects of the behavior that can be used as templates for future system.

The reusability component is maintained by not using specifics thereby rendering the system useless for future work. The static and dynamic models can be used as guidelines for further expansion and in depth design of a real complex system. The tradeoff study summarizes a technique that can be used in multi criteria decision making. A technique which is mathematical not given to heuristics.

UML provides an exceptionally good framework to envision the system piecewise and can be extended to aUML easily using some of the standard notations and if there is a need t model proactive and reactive behavior.

This thesis is an attempt to develop a method for modeling complex real life systems using UML.

Appendices

Standards Used:

ANSI/ISO/IEC 7811-3 specifies in detail the location of embossed characters on an ID-1 card, and Part 4 specifies the location of magnetic stripes. As illustrated in Figure, two areas for embossing are specified. The first, whose center line is 21.42 mm above the bottom edge of the card, or just below the center line of the card, allows for up to 19 card identification number numerals to be embossed. Just below this is an additional area of approximately 14.53 mm by 66.04 mm in which 4 rows of 27 characters each can be used to form a name and address field. This is offset at least 2.41 mm from the bottom of the card and 7.65 mm from the left edge; the embossed characters are raised toward the front side of the card. If a magnetic stripe is included on the card, it is found near the top, on the back side of the card. The specifications state that the magnetic stripe and the embossing may not overlap.

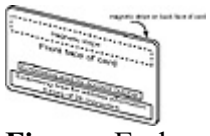


Figure: Embossing and magnetic stripe locations.

Two variants of magnetic stripes can be found on ID-1 identification cards; the form and location of these are defined in ANSI/ISO/IEC 7811-4 (for read-only tracks) and Part 5 (for read/write tracks). One of these is 6.35 mm tall by 79.76 mm wide, positioned no more than 5.54 mm from the top edge of the card and on the back face of the card. This magnetic stripe supports two recording tracks, each of which is intended to be a read-only track.

The Business Model for Identification Cards

By following the ISO standards through several interconnected specifications for identification cards, it is possible to go beyond just the description of physical and electronic characteristics of the card. They have arrived at a business model from which inferences can be made regarding how cards will be manufactured, what groups will actually distribute the cards to end users, and some of the operations to be performed by the end users of the identification cards. For example, the ANSI/ISO/IEC 7811-1 specification defines two terms reflecting the “distribution state” of a card:

- *Unused card*—A card that has been embossed with all the characters required for its intended purpose but has not been issued.
- *Return card*—An embossed card after it has been issued to the cardholder and returned for the purpose of testing.

ANSI/ISO/IEC 7811-2 further defines similar states for magnetic stripe cards:

• *Unused unencoded card*—A card possessing all components required for its intended purpose that has not been subjected to any personalization or testing operation. The card has been stored in a clean environment without more than 48-hour exposure to daylight at temperatures between 5 degrees C and 30 degrees C and humidity between 10% and 90%, without experiencing thermal shock.
• *Unused encoded card*—An unused, unencoded card that has only been encoded with all the data required for its intended purpose (for example, magnetic encoding, embossing, electronic encoding).
• *Returned card*—An embossed or encoded card after it has been issued to the cardholder and returned for the purpose of testing.
DIN ISO 7810 "Identification cards"

ANSI/ISO/IEC 7812:
"Identification of Issuers—Part 1: Numbering System" further develops the business model by establishing a standard for the card identification number, which is displayed in embossed characters on the front face of an ID-1 card. The card identification number, which may be up to 19 characters long, is subdivided into three components:
• Issuer identification number—A six-digit component that includes the following:
• Major industry identifier—A one-digit indicator of the industry designation of the card issuer; it is one of the following:
0—Tag reserved to indicate new industry assignments
1—Airlines
2—Airlines and other future industry assignments
3—Travel and entertainment
4—Banking/financial
5—Banking/financial
6—Merchandizing and banking
7—Petroleum
8—Telecommunications and other future industry assignments
9—For assignment by national standards bodies
• Issuer identifier—A five-digit number associated with the specific issuing organization.
• Individual account identification number—A variable-length component up to 12 digits maximum.
• Check digit—A cross-check number that is calculated from all the previous digits in the identification number according to an algorithm called the Luhn formula, which is defined in an appendix of ANSI/ISO/IEC 7812.
The path toward standards-based specification of a general business mode (for financial transactions) becomes very explicit with ISO/IEC 7813: Identification Cards—Financial Transaction Cards. This specification does not consider any new technical areas, but makes a strict enumeration of the standards that must be adhered to in order to call a card a financial transaction card.
ISO/IEC 7813 specifies the content of the two read-only tracks of a magnetic strip included on the card. This augments the content definition for ISO 4909 for the read/write track. The end result is a complete description of both the technical

characteristics and the information content of cards suitable to support financial transactions, all rooted in international standards and acceptable for worldwide deployment.

References

[1] UML executable architectures of the CDRF architecture-Steve Bernier

[2] Introduction to UML http://www.omg.org/gettingstarted/what_is_uml.htm

[3] Functional Requirements and Use cases www.bredemeyer.com/use_cases.htm

[4] Booch, G., I. Jacobson and J. Rumbaugh, The Unified Modeling Language User Guide. Addison-Wesley, 1999, pp. 219-241.

[5] Christerson, Magnus, "From Use Cases to Components", Rose Architect, 5/99. http://www.rosearchitect.com/cgi-bin/viewprint.pl

[6] Cockburn, Alistair, "Structuring Use Cases with Goals", Journal of Object- Oriented Programming, Sep-Oct, 1997 and Nov-Dec, 1997

[7] Cockburn, Alistair, "Basic Use Case Template", Oct .1998.

[8] Coleman, Derek, "A Use Case Template: Draft for discussion", Fusion Newsletter, April 1998. http://www.hpl.hp.com/fusion/md_newsletters.html

[9] Constantine, Larry. "What Do Users Want? Engineering usability into software"

[10] Malan, R. and D. Bredemeyer, "Functional Requirements and Use Cases", (functreq.pdf, 39k) June 1999.

[10] UML Specification. http://www.rational.com/uml/index.jtmp

[11] UML Class Diagrams – Bernhard Bauer

[12] Understanding UML

[13] www.wam.umd.edu/~rajshree/project

[14] Identix LLC. www.identix.com

[15] Cogent www.cogent.com

[16] Biometric group www.biometricgroup.com

[17] Wireless sensor network framework - Jose' M. Perotti

[18] Standards http://www.biometricgroup.com/reports/rpt_standards.html

[19] Wireless Sensor Networks for Habitat Monitoring-Alan Mainwaring Joseph Polastre Robert Szewczyk David Culler, John Anderson

[20] Special Projects Office (SPO) Chemical and Biological Defense Systems Robert Gibbs

[21] Lecture notes Introduction to Sensors-Dr Marc Madou

[22] Description of Industrial grade Chemical and Biological Sensors.

[23] IEEE standard standards.ieee.org/wireless/overview.html

[24] Sensor Fusion by Albert Tebo, OE Reports 164 - August 1997

[25] Confusion Matrix www2.cs.uregina.ca/~hamilton/courses/831/notes/

[26] UML class diagrams revisited in the context of agent based systems-Bernhard Bauer

[27] Sensor Deployment Strategy for Target Detection by Thomas Clouqueur, Veradej Phipatanasuphorn

[28] R. R. Brooks and S. S. Iyengar. Multi-Sensor Fusion Fundamentals and Applications with Software. Prentice Hall, 1998.
[29] P. Varshney. Distributed Detection and Data Fusion. Springer-Verlag New-York, 1996.

[30] T. Clouqueur, P. Ramanathan, K. K. Saluja, and K.-C. Wang. Value-fusion versus decision-fusion forfault-tolerance in collaborative target detection in sensor networks. In Proceedings of Fourth InternationalConference on Information Fusion, Aug. 2001.
[31] Queueing models by Stephen. R. Lawrence www.csupomona.edu/~hco/ManagementScience/ 09QueueingTheory.ppt
[32] Power Management in Wireless Sensors – Zhiyuan Ren
[33] Infrastructure tradeoffs for sensor networks – Sameer Tilak
[34] Multiagent Systems – M .Vidal
[35] An Environment Description Language – Antonio Chella
[36] Extending UML to modeling and design of Multi-Agent Systems- Krishna Kavi
[37] A generic agent architecture for multiagent systems – Paul Buhler
[38] On Demand Power Management – Rong Zheng
[39] Intelligent sensor Management – Mark Perillo, Wendi Heinzelman
[40] Power Efficient Data Management – J.P. Lynch
[41] Modeling agents with UML-J. Lara
[42] Agent UML website- www.aUML.org
[43] Distributed sensor fusion for object position estimation – Ashley Stroupe, Martin, T. Balch
[44] Sensor fusion using Kalman Filter – M. Spengler
[45] Sensor fusion system – Bjoern Griesbach
[46] Introduction to Space syntax - F.D.Neiman
[47] Sensor Fusion – Laurence Saul