MULTI AGENT SYSTEM BASED CONTROL MODELING FOR PERFORMANCE IMPROVEMENT IN CPS ENABLED BUILDINGS

by

Saurav Bhattarai

A thesis submitted to the Faculty and the Board of Trustees of the Colorado School of Mines in partial fulfillment of the requirements for the degree of Master of Science (Engineering).

Golden, Colorado

Date ______________________

Signed: ______________________
Saurav Bhattarai

Signed: ______________________
Dr. Marcelo Godoy Simões
Thesis Advisor

Golden, Colorado

Date ______________________

Signed: ______________________
Dr. Tyrone Vincent
Associate Professor and Interim Head
Department of Electrical Engineering and Computer Science

ABSTRACT

Energy usage in commercial and residential buildings accounts for nearly 40% of the total energy consumption in the US. Therefore, even small improvements in energy efficiency will provide a tremendous opportunity to reduce total energy consumption on a national scale. Advanced control systems based energy management of renewable and alternative energy can support improved energy efficiencies of modern buildings. The Smart Grid effort is expected to hasten the deployments of renewable energy resources in the US grid to wean off the use of fossil fuels for electricity generation and is aimed at increasing the reliability and security of the electricity grid. This initiative is also expected to impact the end users by providing them an avenue for active participation in demand response. This thesis proposes an advanced energy modeling and control based on Multi-Agent-Systems capable to support the Smart Grid characteristics as defined above and also to contribute for the technology of Cyber-Physical-Systems control for smart buildings.

A holistic approach considering many different systems in a commercial or residential building is required in order to incorporate decision making to achieve maximum energy efficiency. Multi-agent systems inherently support decentralized hierarchical control. Therefore, a multi-agent-system (MAS) based control mechanism is proposed in this work for energy management performance improvement control.

MAS based systems have been successfully applied in the control of non-linear dynamic, large scale distributed computing resources, where speed, reliability and scalability of such systems makes them very complex to be managed with standard control techniques.

In this thesis, mathematical models for thermal and electrical systems for a general area of a commercial building are presented. Methodologies for development of ad-

hoc MAS are developed and discussed in detail. Additionally, a MAS based control system for temperature control of is developed and the performance analyzed.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

floor	f
celing	c
internal walls/partitions	ip
external wall	w
internal air	ai
external air	ao
glazing	g
solar	s
plant (source)	p
internal heat gain	e
Temperature (°F)	T
surface area (ft^2)	A
thermal capacitance (BTU/°F)	C
fraction of solar radiation entering floor	p
heat(source depends on the index)	Q
thermal transmittance ($\text{W/ft}^2.°\text{F}$)	U

LIST OF ABBREVIATIONS

Cyber Physical Systems . . . . CPS

National Science Foundation . . . . NSF

Multi Agent Systems . . . . MAS

Digital Signal Processor . . . . DSP

Organization for Economic Co-operation and Development . . . . OECD

Heating, Ventilation and Air Conditioning . . . . HVAC

Simulation Problem Analysis and Research Kernel . . . . SPARK

Building Energy Simulation Tools . . . . BLAST

Artificial Intelligence . . . . AI

Artificial Immune System . . . . AIS

Hybrid Energy Systems . . . . HES

Building Energy Management Systems . . . . BEMS

National Renewable Energy Laboratory . . . . NREL

International Code Council ICC

Graphical User Interface . . . . GUI

American Society of Heating, Refrigerating and Air-Conditioning
Engineers . . . . ASHRAE

Markov Decision Process . . . . MDP

Lawrence Berkeley National Laboratory . . . . LBNL

United States Department of Energy . . . . USDOE

the QUick Energy Simulation Tool . . . . eQUEST

ACKNOWLEDGMENTS

I would like to thank the College of Engineering at the Colorado School of Mines as well as all the faculty members for their continued support. This thesis would not have been possible without the guidance and support provided by my advisor, Dr. Marcelo Simões. Thank you Dr. Simões for your continued interest, concertn, advice and assistance.

I am ever thankful to my family and their support throughout the course of this research. My father and mother have always supported me, encouraged me to work hard, celebrated my successes and cheered me up during rough times.

I would also like to thank my friends at the Colorado School of Mines, who always make me feel at home, and have become my family in the United States.

Thank you.

For my parents, Hare Ram and Indira.

CHAPTER 1

INTRODUCTION

With today's growing technological advances, it is logical to integrate such advances into our immediate world. The integration of computer-based technology with the physical world is known as Cyber Physical Systems (CPS). The National Science Foundation (NSF) defines cyber physical systems as those systems that are built from and depend upon the synergy of computational and physical components. The current market proliferation of low-cost, increased-capability sensors along with low-cost computational power and availability of wireless/wired networks have really pushed the usefulness and promise of CPS to new heights [1].

A building is a cohesive network of interacting physical systems that can benefit massively from the use of cyber technology in the control and management of these systems. Specifically, when viewed as an energy system with energy inputs and outputs, along with internal energy flows inside, a modern building presents an example of a deeply coupled system where energy usage, comfort and work are interrelated [2]. Implementing a CPS based modeling and control in a building requires understanding computing abstractions and concurrency. In this thesis, a multi agent based control system is developed for the control and energy management of a building. The approach presented in this thesis aims to overcome the problem of the lack of concurrency in computation present today, while reducing energy usage and maintaining user comfort.

1.1 Objectives

The objective of this thesis is to develop a mathematical model for the thermal system of a building, incorporating other systems present(electrical, sensing) within it, as well as to develop an intelligent control system for energy management that can function multiple tasks concurrently. Concurrency in the control system is desired because the physical systems of the building operate concurrently, and it only makes sense that the control system does too. This objective is realized by using a Multi Agent System (MAS) based computing architecture. MAS as the name suggests, are a collection of *agents* that have the ability to perform different computational tasks at the same time to achieve a system wide goal. Also, this thesis will analyze the energy efficiency benefits of using such a decentralized control scheme on a building.

1.2 Motivation

Control systems currently applied in buildings today for user comfort management and energy management do not address all the different systems and factors present in the building. Modern buildings do have energy management systems that do account for factors such as occupancy of building, and the actions of the occupants, they lack intelligent decision making processes to take full advantage of this information. Additionally, older buildings that have outdated controls cannot even take advantage of the new available computing resources, as well as new emerging technologies such as the smart-grid and renewable energy resources.

Intelligent energy management in commercial buildings in the US has potential to provide a large opportunity to reduce total energy consumption as well as to reduce pollution. Commercial buildings account for 38% of total primary energy usage in the United States and more than 53% of the total electricity usage [3]. Advanced modeling and control, associated with the flexibility of the smart-grid technology allow the integration of renewable energy, energy storage facilities, and customer participation in order to improve the overall efficiency [4].

Development of multi-agent based architectures has increased since the latter half of the 1980s [5]. A MAS approach is used in large, complex problems, with global goals and operating on local knowledge and possessing limited abilities [5]. MAS based controllers have been showing a lot of potential in systems requiring non-linear dynamic, large scale distributed computing resources [6] where speed, reliability and scalability of such distributed systems makes them ideal to be used to control a complex system such as a building. Other applications of MAS can be found in air traffic control [7], manufacturing, and robotics. One of the most important factors for a successful implementation of a MAS is availability of reliable, fast, secure communication. With increasing penetration of the internet based and other networked services, it is safe to assume that electronic communication will be available in buildings, especially of the commercial nature.

Additionally, the use and adaptation of MAS for control in power systems engineering has been explored [8, 9] and promising results have been observed, specifically in the area of micro-grid control [10]. With this increasing interest in using MAS for Smart Grid technology, using it for building control seems very feasible to maximize energy efficiency.

1.3 Scope

The scope of this thesis includes developing a simplified combined mathematical model of a commercial building for both thermal and electrical systems, and incorporating a MAS based energy management system to realise a CPS. Focus will be given to the application of a MAS based energy management system to the building model.

1.4 Background

A thorough literature review was performed for the preparation of this thesis. The literature review is organized into four main categories: 1) Cyber Physical Systems, 2) Energy usage in buildings, 3) Building Modeling, and 4) Multi Agent Systems.

1.4.1 Cyber Physical Systems

Cyber-physical systems are the synthesis of modern computation with existing physical processes, where embedded, networked computers monitor and control as well as affect the physical processes and vice versa[11]. Scholars that have been looking at the tight integration of computing with physical systems since the early 2000's have iterated that with proper design and implementation, the impact of CPS can be greater than that of the IT and internet revolution [1, 11, 12].

1.4.1.1 Applications of CPS

The use of microcontrollers and microprocessers in the control physical systems is not a new concept, they are known as *embedded systems* and have been in use since the 1960s [13]. These devices have historically been built for a single task, and generally do not have the ability to adapt to different circumstances. But the availability of inexpensive wide area networks with the ability to keep multiple devices connected with fast information flow between them introduces a lot of opportunities for enhancing these embedded systems for a state of the art CPS.

As is the focus of this thesis, there is a great potential for improving energy efficiency with the integration of CPS in a modern building. Even though energy efficiency is the focus, it should be noted that other factors in a building can also benefit a great deal from embedded technology such as security. There are many other areas where CPS can have a very big impact such as in transportation, where embedded intelligence can provide great benefits in terms of general safety on the roadways and efficiency. By embedding intelligence in vehicles, the possibility of a completely automated highway has been explored in [14]. The area of communications can benefit drastically if bandwidth requirements and availability can be negotiated by connected devices. Similarly, control of critical infrastructure such as the electric grid and the gas and water supply have a lot to gain from CPS.

1.4.1.2 Problems faced in CPS

There are two major problems related to implementing CPS in the areas aforementioned. Initially, the control of concurrent physical processes by non-concurrent computational models/methods. Previous embedded control systems did not require computer processes to run simultaneously as the systems and control algorithms were not complex. But, as systems become further complex, along with their control systems, simultaneous execution of code is required. In simpler control systems, if efficiency and speed were required, specialized processors such as Digital Signal Processors (DSPs) could be used, or the algorithm itself would be stripped of any extra, non-essential features that it might have contained [11]. But, with the increasing complexity requirements of today's systems, sacrificing functionality for speed is not desired. The problem is caused by current software paradigms rather than hardware limitations. Processors available in the market today consist of multiple cores, and are completely capable of running processes simultaneously. But today's mainstream software and software development platforms are focused on the age old sequential programming paradigm, which cannot take advantage of the multiple cores available in the hardware [15].

The second issue is related to the lack of concurrent computation,i.e. lack of accurate and predictable timing and thus proper real-time implementations. There is a misconception that real-time applications are fast applications does not help in their development [11]. Complex systems, such as the ones discussed before not only depend on the logical results of the computation, but also on the time at which these results are obtained. Current real-time applications are developed after rigorous testing, which is performed to find out exactly how long the application takes to run on a particular piece of hardware [16]. This translates into the application not being able to change or retro-fit in any way, essentially taking away the ability to adapt. CPS are most effective when they can learn and adapt to the environment that they are applied in, and the current inability to predict timing performance for adaptive CPS is a set back.

1.4.2 Buildings

Buildings are very complex structures composed of multiple interconnected systems and layers of abstraction. Large buildings consist of a vast network of sensors, along with a large thermal and electrical system. When considering large buildings, the occupancy and behaviors of the occupant cannot be ignored - this can be classified as the human network [17]. Figure 1.1 illustrates the different, interconnected networks present in a building. The outer ring denotes external factors; they affect the performance of the building, but are not contained within it. An advanced

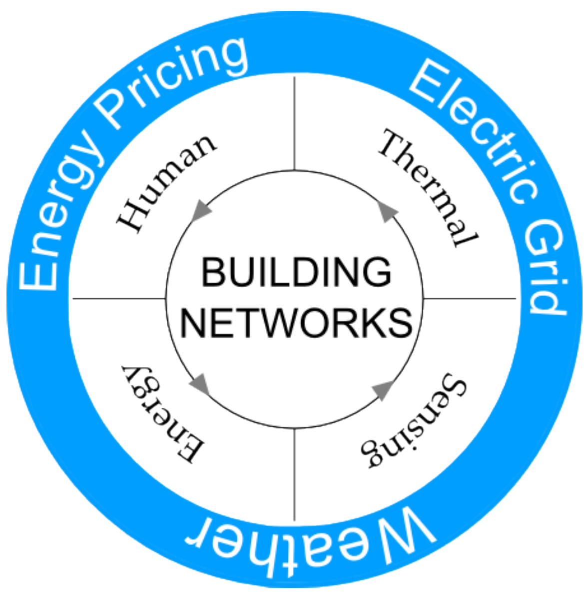


Figure 1.1: A building as a collection of interacting networks

energy management control system should integrate and fuse data based on thermal behavior, users occupancy, electric load, light, and some predictions based on scheduling of rooms and halls. In addition, modern buildings need to include on-site distributed generation technologies, demand response management as well as energy storage technologies.

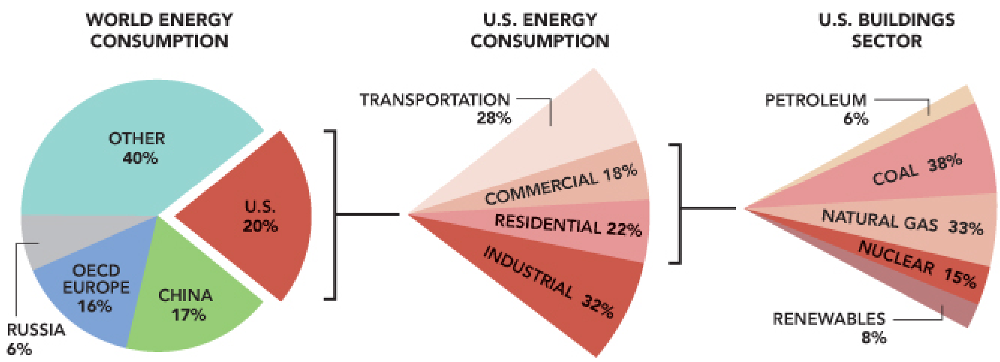
1.4.2.1 Energy Usage

Energy can be classified into three main industrial and commercial categories: i) Primary Energy, ii)Secondary Energy, and iii)Tertiary Energy. Primary energy is naturally occurring energy such as coal, oil, and natural gas. Secondary energy is transformed energy such as electricity and refined petroleum products. Tertiary energy is a further transformation from secondary energy; warmth, motion, and mechanical power are some examples of tertiary energy [3]. Tertiary energy is the most common form of energy that is utilized in buildings, and is the focus of this thesis.

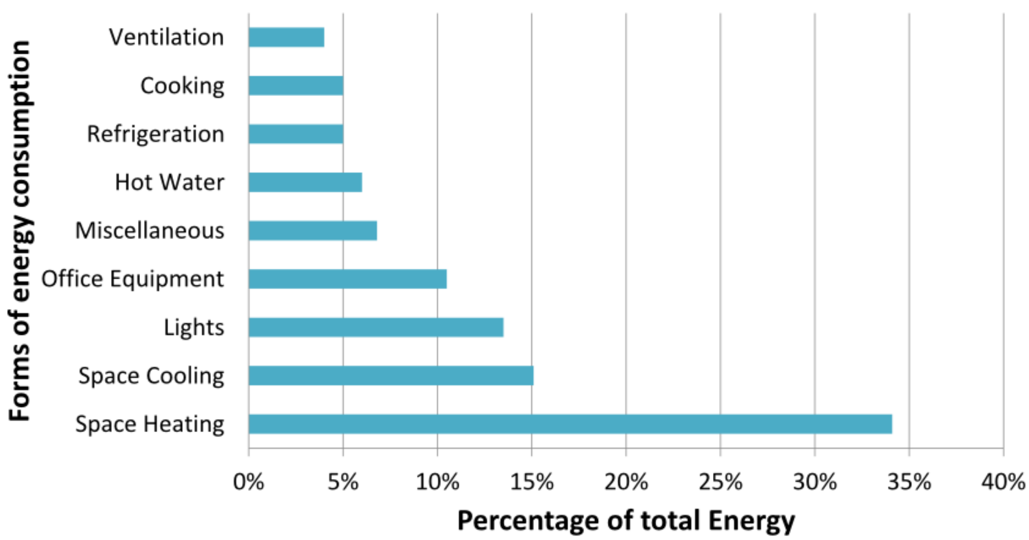
Data supports that 38% of total primary energy available in the Organization for Economic Co-operation and Development (OECD) countries is used in buildings (commercial and residential combined) [3], as well as accounting for 53% of the total electricity use. Figure 1.2(a) illustrates the distribution of energy usage in the world and energy supplied to US buildings. As one can see, the major contributors for energy in buildings are fossil fuel based sources (petroleum, coal, natural gas), which contribute to environmental pollution; buildings emit 38% of CO_2, 49% of SO_2, and 25% of NO_x [18]. Figure 1.2(b) illustrates the breakdown of the energy usage in a commercial building in the US. Most of the energy is attributed to low efficiency space heating devices. The use of these devices continues because of the comparatively low price of natural gas in the US in the last few years [19].

1.4.2.2 Energy Management Systems

Building energy management systems aim at the maintainance of user comfort by controlling the different aspects of a building such as the Heating, Ventilation and Air Conditioning (HVAC), lighting, and other electrical loads, while trying to keep an acceptable level of efficiency in energy conversion [22].



(a) Energy usage in the world and the United States [20].



(b) Energy usage in US commercial buildings [21]

Figure 1.2: World and US Energy Consumption Breakdown

1.4.2.3 Mathematical Modeling

In order to develop and analyze a control system, a model has to be created for the system that the control system has to act upon. A model can be an actual physical model or it can be a mathematical one. Mathematical models are better to manipulate compared to physical ones, and can be designed to closely mimic the actual system. In this thesis, the thermal behavior of a section of a building is modeled mathematically and used in computer simulations to perform analysis related to energy efficiency.

A mathematical description of thermal behavior of building systems are complex as they involve modeling of different interconnected subsystems, each containing non-linearities, long-time constants, and uncertainties [23]. Additionally, when developing a model for a building, external factors also have to be accounted for, mainly the weather plus utility interaction when a smart grid is available for the building to connect to.

The most common mathematical model in use today is the lumped capacitance model that was developed in 1982 by Lorenz and Masy [24] and further enhanced by Levermore in 1992 [25] and Tindale in 1993 [23]. The dynamic model for thermal building performance analysis presented in [23] is a cumulative work [24, 25], and has been widely used in the field of building simulation. This is the model that is used in this thesis and a detailed description is presented in Chapter 2.

There are different software tools available for building system modeling such as EnergyPlus, SPARK, and BLAST. These tools have incorporated many of the dynamic thermal models mentioned before, but since this thesis is concerned with developing a control system and not just a building model, using proprietary modeling software has been discarded as they do not offer flexibility in designing control systems, particularly with an advanced paradigm such as a MAS based control.

1.4.3 Multi Agent Systems

MAS theory is a relatively new programming paradigm evolved from the artificial intelligence (AI) theories introduced in the 1970s. In the field of AI, agents are are physical or virtual entities that intelligently interact in an environment by both perceiving and affecting the environment[26]. MAS extend this idea by introducing communication capabilities in the agents.

A formal definition of agents describes an agent *as an autonomous entity in an embedded environment that either solves problems by itself, or cooperates with other agents to find a solution. It has control over its internal state as well as its outputs and can run without external intervention*[5]. Massive amounts of data can be exchanged between agents due to the availability of fast communication channels. This allows the agents to complete global tasks that are not usually possible by the actions of a single agent alone. Agents have the ability to learn from past actions, or through communication with other agents. Agents have the ability to work with each other and interact with humans to monitor events and perform tasks[17]. Development of a multi-agent system is a methodological process. First, a problem has to be divided into sub-problems that can be solved by the individual, representative agents. Solutions to these sub-problems are then combined to change the current global state of the system through agent-to-agent coordination[27]. MAS are a distributed computing paradigm, i.e. instead of using a central computer to process inputs and make decisions, multiple, less powerful computing resources, dedicated to individual agents, achieve a complex goal.Agents can be designed to take global goal seeking actions as well as reactive actions. In large systems, emergency scenarios may arise where an agent needs to take a reactive action immediately without seeking information from other agents.

1.4.3.1 Application of MAS

MAS have stood out as a solution of choice when designing Complex Systems[1] [28] because of the ability to divide the problem formulation into parts, and solve them individually ultimately leading to the solution of the larger problem.

MAS have been implemented in different ways, with the majority of implementations conforming to the formal definitions. However, there are two recent implementations that have been seen in literature lately, and are gaining some popularity. First developing a MAS using the behavior of honey bees as their basis [29, 30]. In this implementation, the communication behavior of the agents are different than normal: the agents do not directly communicate with each other, but communicate through the environment they are present in. They learn about the actions of other agents by analyzing the environment. This allows agents to be added or removed from the environment without changing the system design. Another implementation of MAS is achieved by mimicking the human immune system, and creating an artificial immune system (AIS). In this implementation, the different parts of an immune system (antibody, antigen, paratope, epitope) are defined as agents and the interaction between these different parts of the immune system is the basis for the communication between the agents [31, 32]. The human immune system has properties of paratopes and epitopes combining to form new antibodies, essentially creating memory in the system. The AIS also has this functionality and is the major advantage of this implementation. The ability to learn (machine learning) is inherent in the design of the MAS.

Applications of MAS have been seen in a variety of fields, ranging from air traffic control [7] to applications in smart grid technology [8–10] and other power engineering technologies such as hybrid energy systems (HES) [33].. MAS have also been used to

[1] Complex Systems are characterized by large number of entities in interaction, exhibiting emergent behaviour.

develop BEMS as presented in [26], but it has a larger focus on the the development of the agents, and coordinating the behavior of the occupants.

1.4.3.2 MAS for Building Energy Management

As mentioned in previous sections, building structures are complex systems, which can benefit highly if MAS are used in the development of Building Energy Management Systems(BEMS). Maintaining energy efficiency and comfort are the major requirements for a building BEMS, while keeping into account external factors such as weather forecasts and electricity costs. Agents need to receive information from the sensors to detect changes in the environment (the building in this case) and modify to the environment through effectors (administering changes in the thermal and/or electrical system). As no major changes in the sensing or the thermal/electric system need to be made, agents can easily incorporate into legacy systems in building and can be validated for further advanced performance requirements [34].

1.5 Contributions of Thesis

The contributions of this thesis include:

1. Development of an adjustable building thermal model that considers all the major physical systems as well as external factors (weather conditions).

2. Development of an ad-hoc MAS design methodology derived from existing methodologies, but more specific for building energy management systems. This methodology facilitates design and specifications of agents at any user needs, without having to redesign the complete system, making the proposed system a friendly reusable design methodology for multi-agent-systems.

3. This thesis provides detailed descriptions and definitions of agents for diverse behaviors through the developed MAS design methodology . This thesis also discusses and shows how to implement the designed MAS in dynamic simulation

and interfacing with specific software for multi-agent-systems, integrating all the simulation environment inside Matlab, Simulink and JADE.

4. This research shows how to compare energy efficiencies of the MAS based control system with traditional control and provides some benchmarks for future improvements.

While performing the tasks listed above, some of the work has been peer reviewed and published. The development of the agents and their performance is the main focus in [35]. The implementation of the complete MAS based BEMS and the energy efficiency comparisons with non-MAS based BEMS is the main focus in [36]. Also, a collaborative paper is currently being worked on with fellow researchers in the Universitè de Technologie de Belfort-Montbèliard concerning the development of MAS for building energy management using an artificial immune system model for system design.

1.6 Organization of Thesis

This thesis is focused on the development of a MAS based energy management system for a building structure that can improve efficiency and comfort for the occupants.

In Chapter 2, the physical properties of an office building system are defined, and a general solution for mathematical modeling of the building is discussed. In order to extend the model to any size or configuration of any building, a GUI based program is developed to provide state space matrices for different building configurations.

Chapter 3 discusses the design of a complete multi agent system. Being such a new paradigm in computer science and engineering, there is no proven methodology of design yet. Therefore, Chapter 3 focuses on developing an ad-hoc MAS design methodology based on existing methods.

In Chapter 4, the physical building model is implemented in Simulink and multiple tests are run for validation. Following the physical model development, the multi agent system is developed using the methodology proposed in Chapter 3. The details of implementation of a MAS in software, and the control of the model developed in Chapter 2 are further discussed in this chapter.

Chapter 5 presents some case studies with different types of MAS control system implementation. The case studies range from simple reactive control, to more complex such as machine learning enabled concepts. The performance of the building model as well as the control system are analyzed and compared with each other.

CHAPTER 2
SYSTEM DEVELOPMENT: MATHEMATICAL BUILDING MODEL

For the application of a MAS based control to a physical buildling, there is a need to develop a physical model of a building. A mathematical model of a physical will help in testing and simulation of the system as well as any control strategies that might be applied to it. This chapter focuses on developing a mathematical model for the thermal aspect of a building structure. A state-space matrix representation of the thermal system of a building is developed, which can easily accomodate any size, structure buildling, and is easily implemented in a computer aided simulation enviornment.

2.1 Physical Building

To order to develop a mathematical model of a building structure, we need to define the physical attributes of a building. This includes defining the location of the building, the size, orientation as well as the material used in the construction of the building. For this work, the location of the building is to be Golden, Colorado. This choice was made because Colorado School of Mines is located here, but more importantly, thanks to the National Renewable Energy Laboratory(NREL) being located in Golden, there is abundant hour-by-hour weather data available [37]. . The building itself is chosen to be a medium sized office building with about 50 employees. The United States government recommended standard for office space is 200ft^2 of usable space per person, which also translates to 230ft^2 of rent able space per person [38]. Abiding by this standard, the size of the building is calculated to be (50)*(230)= 11,500ft^2. The building is defined as being three stories, so there is approximately 3800ft^2 of rent able space per floor. The height of each floor is defined to be 10ft, which surpasses the minimum height of 7.6ft set by the International Code

Council(ICC) in [39]. This translates into a total volume of (30)*(3800)=115,000ft^3 of air inside the office building.

2.2 Mathematical Model

The mathematical representation of the thermal characteristics of a building proposed by Lorenz and Masy in [24] and then amended by others in [25] and [40] is used frequently among modelers and the same is used in this thesis. This model is based on the assumption that a building consists of an outer shell containing internal elements. The outer shell is in thermal contact with the outer world. The equation governing the heat balance at the system boundary is give in Equation 2.1.

$$\phi_h(t) + \phi_s(t) = \phi_t(t) + \phi_c(t) \tag{2.1}$$

In Equation 2.1, ϕ_h is the heat supplied by the plant (HVAC) as well as any other sources of energy present within the building; ϕ_s is the heat gained by solar radiation that enters into the building through the windows directly, or through diffusion; ϕ_t is the total heat loss in the building through external contact and thermal resistances of materials inside the building; ϕ_c is the capacity of the elements in the building to retain heat. Using the heat balance equation above, and thinking of elements causing

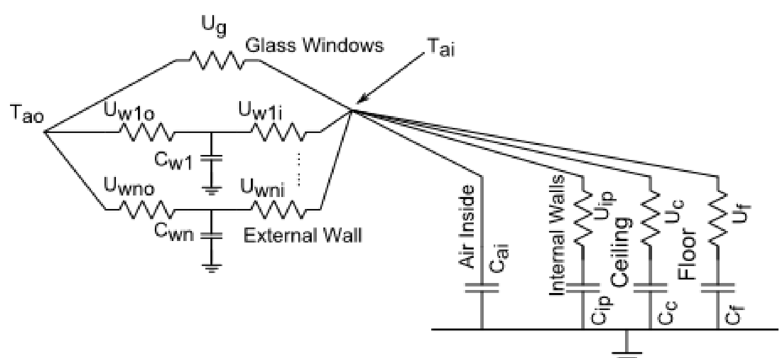


Figure 2.1: Analogous Electric Circuit for Lumped Capacitance Thermal Model

loss due to their thermal resistances as resistors, and the thermal retention capacity of the materials as capacitors, a lumped capacitance model can be developed and the system can be thought of in terms of an electric circuit. Figure 2.1 is an example of how a a thermal model can be viewed as an electric circuit. The lumped capacitance model of the thermal system of the building can now be represented by a set of differential equations, each for the different parts of the building.

$$\frac{dT_f}{dt} = \frac{A_f}{C_f}\left[\frac{pQ_s}{A_f} + U_f(T_{ai} - T_f)\right] \tag{2.2}$$

$$\frac{dT_c}{dt} = \frac{A_c}{C_c}\left[U_c(T_{ai} - T_c)\right] \tag{2.3}$$

$$\frac{dT_{ip}}{dt} = \frac{A_{ip}}{C_{ip}}\left[\frac{(1-p)Q_s}{A_{ip}} + U_{ip}(T_{ai} - T_{ip})\right] \tag{2.4}$$

$$\frac{dT_w}{dt} = \frac{A_w}{C_w}\left[U_{wi}(T_{ai} - T_w) + U_{wo}(T_{ao} - T_w)\right] \tag{2.5}$$

$$\begin{aligned}\frac{dT_{ai}}{dt} &= \frac{1}{C_a}Q_p + Q_e + (A_gU_g + U_v)(T_{ao} - T_{ai}) \\ &\quad + A_wU_{wi}(T_w - T_{ai})A_fU_f(T_f - T_{ai}) \\ &\quad + A_cU_c(T_c - T_{ai}) + A_{ip}U_{ip}(T_{ip} - T_{ai})\end{aligned} \tag{2.6}$$

Table 2.1: Description of variables and indices used in 2.2 through 2.6

Indices	**Description**
f	floor
c	celing
ip	internal walls/partitions
w	external wall
ai	internal air
ao	external air
g	glazing
s	solar
p	plant (source)
e	internal heat gain
Variables	**Description**
T	Temperature (°F)
A	surface area (ft^2)
C	thermal capacitance (BTU/°F)
p	fraction of solar radiation entering floor
Q	heat(source depends on the index)
U	thermal transmittance (W/ft^2.°F)

Equations 2.2 to 2.6 represent the thermal characteristics of the floor, ceiling, internal walls, external walls, and the air inside of a building respectively. The description of the variables and subscripts used are presented in Table 2.1.

Earlier, an analogy between an electrical circuit and thermal behavior was drawn, and terms such as thermal resistance and thermal capacitance were used. To understand thermal resistance better, it has to be understood that just as electrical resistance is associated with the conduction of electricity, thermal resistance is associated with the conduction of heat. If we go by the general definition of resistance being the ratio of a driving potential to the corresponding transfer rate, and with the help of the heat flow rate equation (Equation 2.7), the thermal resistance for conduction can be defined as in Equation 2.8.

$$q = \frac{kA}{L}(T_1 - T_2) \tag{2.7}$$

$$R_t = \frac{T_{s,1} - T_{s,2}}{q_x} = \frac{L}{kA} \tag{2.8}$$

Similarly for electrical resistance for conduction, using Ohm's law, we have 2.9.

$$R_e = \frac{E_{s,1} - E_{s,2}}{I} = \frac{L}{\sigma A} \tag{2.9}$$

The analogy between Equation 2.8 and Equation 2.9 is obvious. This derivation of thermal resistance was for a structure such as a wall in a building, but a thermal resistance can also be present because of heat transfer due to convection at a surface. If we define q using Newton's law of cooling (Equation 2.10),

$$q = hA(T_s - T_{\text{inf}}) \tag{2.10}$$

we arrive at the expression in 2.11, for the thermal resistance due to convection.

$$R_{t,conv} = \frac{T_s - T_{\text{inf}}}{q} = \frac{1}{hA} \tag{2.11}$$

A graphical representation of thermal resistance as compared to electrical resistance is presented in Figure 2.2. Since the conduction and convection resistances are in series in a thermal circuit, the total thermal resistance can be represented as in Equation 2.12.

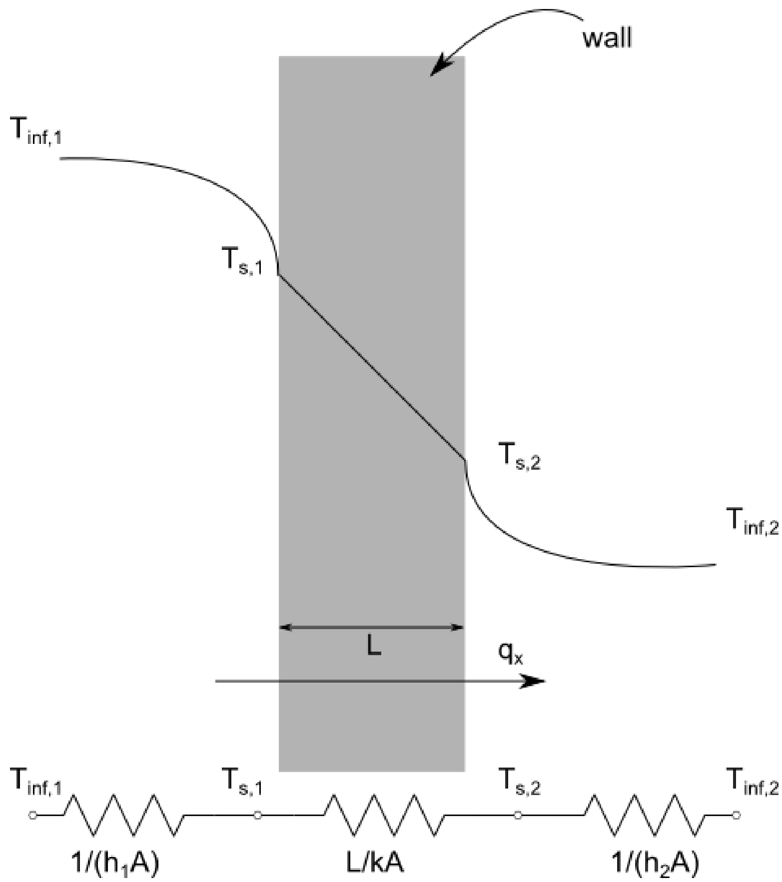


Figure 2.2: Temperature distribution and equivalent thermal circuit for heat transfer through a plane wall

$$R_{tot} = \frac{1}{h_1 A} + \frac{L}{kA} + \frac{1}{h_2 A} \tag{2.12}$$

Having drawn an analogy between thermal resistance and electrical resistance, another analogy between the thermal and electrical systems which makes the model easier to understand is thermal potential. The thermal potential, which can be viewed as temperature of a point, is fixed in steady state heat transfer, while it varies in time in transient heat transfer or heat storage.

Another term that was used to build the analogous circuit shown in Figure 2.1, was *thermal capacitance*. During transient heat transfer, the temperature of the materials changes with time. Thermal capacitance, which can be viewed as heat capacity, is the capacity of the thermal component in question to retain or store heat. The units for heat capacity are J/$^{\circ}F$. If the thermal component (wall, floor, ceiling) is made of homogeneous material, the thermal mass is the product of the mass of the material and its specific heat capacity. If the thermal component is heterogeneous (which is normally the case), the sum of the specific heat capacities are used for simplicity.

In the context of building design, thermal mass provides "inertia" against temperature fluctuations. If the outdoor temperature fluctuates considerably throughout the day, placing a large thermal mass in the insulated portion of the building will aid in preventing similar fluctuations indoors. It should be noted that this is different than a material's *insulative value*, which reduces the thermal conductivity, in this case of a building.

To create a computational model for the system for simulation purposes, a state space representation is desired. Equations 2.2 to 2.6 can be stacked using the state space notation in Equation 2.13, where $\dot{x}$ is a vector of the derivatives of temperatures, A and B are the matrices of coefficients, x is a vector of states and u is the input vector.

$$\dot{x} = Ax + Bu \tag{2.13}$$

This results in the complete state space representation presented in equation 2.14.

$$\begin{bmatrix} \dot{T}_w \\ \dot{T}_f \\ \dot{T}_c \\ \dot{T}_{ip} \\ \dot{T}_{ai} \end{bmatrix} = \begin{bmatrix} \frac{-A_w}{C_w}[U_{i_w} + U_{o_w}] & 0 & 0 & 0 & \frac{A_w U_{i_w}}{C_w} \\ 0 & \frac{-A_f U_f}{C_f} & 0 & 0 & \frac{-A_f U_f}{C_f} \\ 0 & 0 & \frac{-A_c U_c}{C_c} & 0 & \frac{-A_c U_c}{C_c} \\ 0 & 0 & 0 & \frac{-A_{ip} U_{ip}}{C_{ip}} & \frac{-A_{ip} U_{ip}}{C_{ip}} \\ \frac{-A_w U_w}{C_a} & \frac{-A_f U_f}{C_a} & \frac{-A_c U_c}{C_a} & \frac{-A_{ip} U_{ip}}{C_a} & (\cdots) \end{bmatrix} \times \begin{bmatrix} T_w \\ T_f \\ T_c \\ T_{ip} \\ T_{ai} \end{bmatrix} \cdots$$
$$+ \begin{bmatrix} 0 & 0 & 0 & \frac{A_w U_{ow}}{C_w} \\ 0 & 0 & \frac{p}{C_f} & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{(1-p)}{C_{ip}} & 0 \\ \frac{1}{C_a} & \frac{1}{C_a} & 0 & \frac{A_g U_g + U_v}{C_a} \end{bmatrix} \times \begin{bmatrix} Q_p \\ Q_e \\ Q_s \\ T_{ao} \end{bmatrix} \tag{2.14}$$

This state-space representation of the system can be used to develop a computational model for simulation in the MATLAB/Simulink environment, which is discussed in more detail in Chapter 3.

When developing a mathematical model for a physical system such as a building, scalability has to be taken into account: buildings can be of different shapes and sizes and additionally made from different materials that have different thermal properties. The model presented here has the capability of doing exactly that. The major parts of the building that affect the thermal performance of the building have been included in Equations 2.5 and 2.6. Since the model is based on the idea of lumped capacitance and every element represented as a series combination of resistors and capacitors, if need arises to add another element to the model, it can be easily done. Also, in terms of size scalability, every representation of a physical element of the building is dependent on area (A), so the model can be sized according to the physical dimensions of the building. As can be seen in Equation 2.14, the matrix A is only populated in the main diagonal as well as the last row and column. This trend was seen to continue when another element, with different properties was added to the model; the matrix would grow in the direction of the main diagonal. Having recognized this pattern, a MATLAB script was written which would take inputs from the user as to how many elements are to be considered for modeling and generate matrices A and B accordingly. Since the symbolic representation of the matrix is not of great use during analysis, coupled with the lack of symbolic math abilities of MATLAB, the script would also require inputs of all the variables/constants described in Table 2.1, to generate numerical matrices. This graphical user interface (GUI) developed for easy input to the script, and easy viewing of the output matrices can be seen in Figure 2.3. The building chosen for modeling in this thesis as described in Section 2.1 is quite large (11,800ft^2). There are a lot of possible layouts of this building, and calculating the different areas and keeping track of all the different materials used will be a tedious

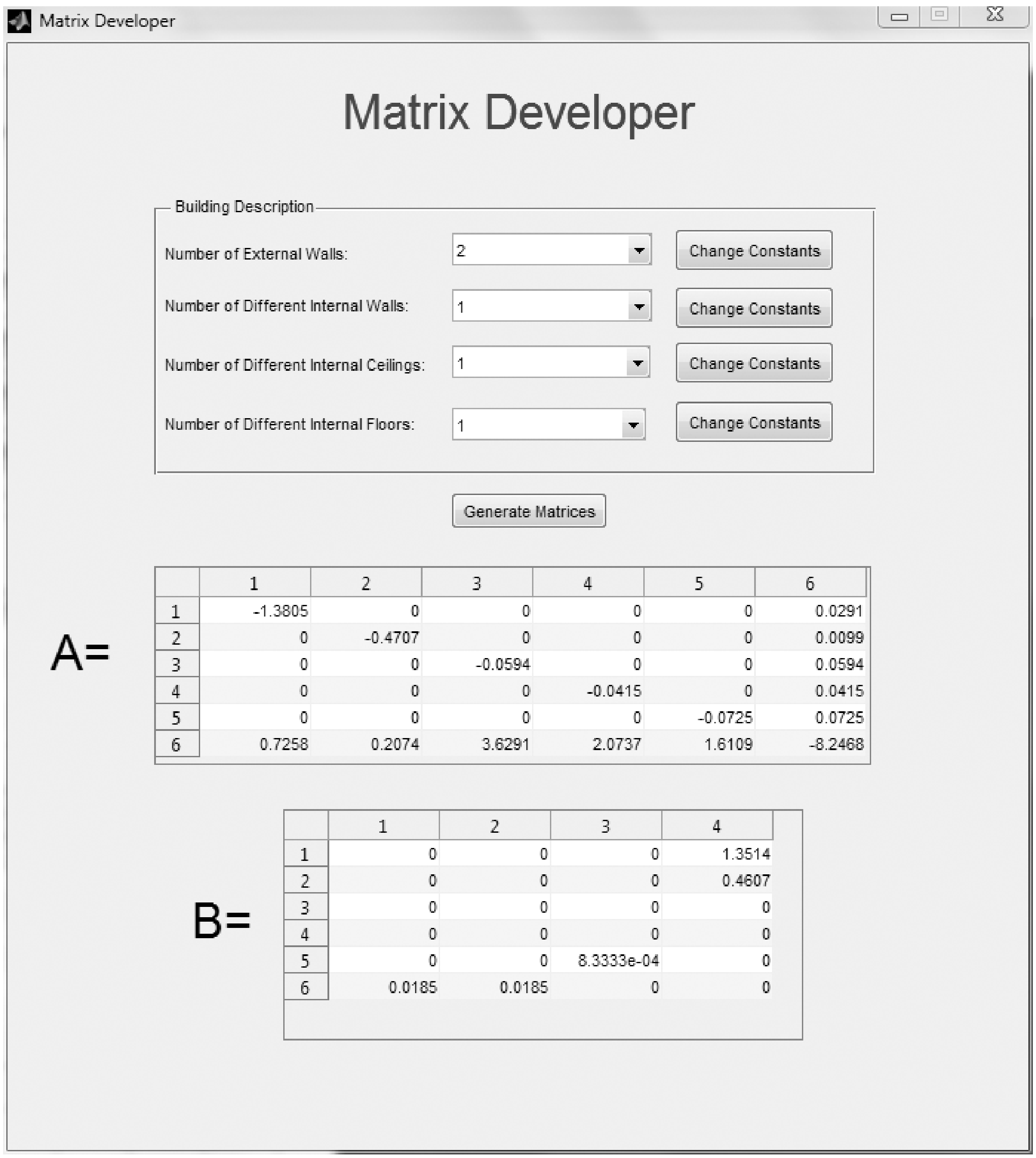


Figure 2.3: GUI for development of state-space matrices

task. Due to the generality of the modeling solution presented before, a smaller area of the building can be considered and simulated. The floor plan of the considered area is presented in Figure 2.4. The configuration presented consists of three rooms, of which one is a kitchen. Room 1 is modeled as a typical reception/outer office area consisting of office equipment such as printers, a photocopy machine, and computers. Room 2 is a typical single person office consisting of furniture and a single computer. The kitchen area contains a refrigerator, microwave and a coffee maker. Table 2.2 summarizes the contents of each individual room.

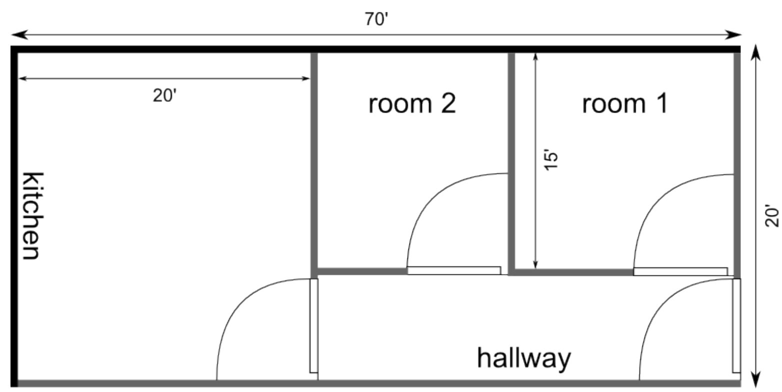


Figure 2.4: Floorplan of the section of building to be modeled

The building chosen for modeling in this thesis as described in Section 2.1 is quite large (11,800ft^2). There are a lot of possible layouts of this building, and calculating the different areas and keeping track of all the different materials used will be a tedious task. Due to the generality of the modeling solution presented before, a smaller area of the building can be considered and simulated. The floor plan of the considered area is presented in Figure 2.4. In the figure, the dark black lines of the structure represent external walls and the gray lines are representative of the internal walls/partitions of the office. As an exercise to reiterate the claim made earlier about the adaptability of the modeling solution, it is assumed that the two external walls are made of different materials, hence changing the thermal behavior of the building. It is also assumed that the materials used for the internal partitions, ceiling, and floor are consistent throughout the area considered. The different external walls translate to the addition of a differential equation in the lumped capacitance model of the building; the walls will be referred to with $w1$ and $w2$ indices. The additional equation is identical to Equation 2.5, with the index w replaced with $w2$.

$$\frac{dT_{w2}}{dt} = \frac{A_{w2}}{C_{w2}} \left[U_{w2i}(T_{ai} - T_{w2}) + U_{w2o}(T_{ao} - T_{w2})\right] \tag{2.15}$$

Using Equation 2.13, and adding 2.15, we reach the following state space equation:

$$\begin{bmatrix} \dot{T}_{w1} \\ \dot{T}_{w2} \\ \dot{T}_f \\ \dot{T}_c \\ \dot{T}_{ip} \\ \dot{T}_{ai} \end{bmatrix} = \begin{bmatrix} \frac{-A_{w1}}{C_{w1}}\left[U_{i_{w1}+o_{w1}}\right] & 0 & 0 & 0 & 0 & \frac{A_{w1}U_{i_{w1}}}{C_{w1}} \\ 0 & \frac{-A_{w2}}{C_{w2}}\left[U_{i_{w2}+o_{w2}}\right] & 0 & 0 & 0 & \frac{A_{w2}U_{i_{w2}}}{C_{w2}} \\ 0 & 0 & \frac{-A_fU_f}{C_f} & 0 & 0 & \frac{-A_fU_f}{C_f} \\ 0 & 0 & 0 & \frac{-A_cU_c}{C_c} & 0 & \frac{-A_cU_c}{C_c} \\ 0 & 0 & 0 & 0 & \frac{-A_{ip}U_{ip}}{C_{ip}} & \frac{-A_{ip}U_{ip}}{C_{ip}} \\ \frac{-A_{w1}U_{w1}}{C_a} & \frac{-A_{w2}U_{w2}}{C_a} & \frac{-A_fU_f}{C_a} & \frac{-A_cU_c}{C_a} & \frac{-A_{ip}U_{ip}}{C_a} & \cdots \end{bmatrix} \cdots$$

$$\times \begin{bmatrix} T_{w1} \\ T_{w2} \\ T_f \\ T_c \\ T_{ip} \\ T_{ai} \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & \frac{A_{w1}U_{o_{w1}}}{C_{w1}} \\ 0 & 0 & 0 & \frac{A_{w2}U_{o_{w2}}}{C_{w2}} \\ 0 & 0 & \frac{p}{C_f} & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{(1-p)}{C_{ip}} & 0 \\ \frac{1}{C_a} & \frac{1}{C_a} & 0 & \frac{A_gU_g+U_v}{C_a} \end{bmatrix} \times \begin{bmatrix} Q_p \\ Q_e \\ Q_s \\ T_{ao} \end{bmatrix} \tag{2.16}$$

As mentioned before, only the main diagonal and the last row and column are populated in the A matrix. The next step in the design process is to define the characteristics of the given area and calculate the constants.

The configuration presented consists of three rooms, of which one is a kitchen. Room 1 is modeled as a typical reception/outer office area consisting of office equipment such as printers, a photocopy machine, and computers. Room 2 is a typical single person office consisting of furniture and a single computer. The kitchen area contains a refrigerator, microwave and a coffee maker. Table 2.2 summarizes the contents of each individual room. The thermal properties of the furniture are ignored in

Table 2.3: Efficiency values for different electric loads

Type of Load	Load	Efficiency
Lighting	1400W[2]	90%
Computers	320W	85%
Office Equipment (active)	900W	85%
Office Equipment (standby)	20W	85%
Refrigerator	50W	70%
Microwave	1000W	65%
Coffee Maker	1000W	65%

this model, as their effect would be very insignificant. If required, equations for the furniture can be developed without a lot of effort. The electronic equipment listed in Table 2.2 are used in the model as they generate heat, and affect the thermal performance of the building. Efficiency figures for the different electrical components are used to calculate the amount of heat generated. Efficiency data collected for typical Energy Star certified devices from [26, 41–43] is presented in Table 2.3.

Table 2.2: Summary of the contents of each room.

Room	**Contents**
Room 1	Furniture
	2 Computers
	1 Printer
	1 Photocopier
Room 2	Furniture
	1 Computer
	1 Printer
Kitchen	Furniture
	1 Refrigerator
	1 Microwave Oven
	1 Coffee Maker
	1 Electric Kettle

Table 2.4: Thermal Resistance and Capacitance values

Building Parameter	Area(ft^2)	U(W/ft^2.°F)	U_i	U_o	C(BTU/°F)
External Wall 1 (w_1)	700	-	0.056	2.60	1346.8
External Wall 1 (w_2)	200	-	0.056	2.60	1128.6
Floor (f)	1400	0.14	-	-	3300
Ceiling (c)	1400	0.08	-	-	2700
Internal Walls (ip)	1450	0.06	-	-	1200

To complete the model for the building area, data for the materials used in the construction was required. It was assumed that the building's exterior walls were concrete walls with a single layer of insulation inside, and the internal walls were metal frame partitions, that did not include insulation material. The American Society of Heating, Refrigerating and Air-Conditioning Engineers(ASHRAE) handbook [44] contains detailed tables, as well as calculation methods for obtaining the values for thermal transmittance (U), and the thermal capacitance (C). Details of the calculation of U values are presented in Appendix A. A summary of the values that were calculated, and to be used in the model are presented in Table 2.4 along with the values for the areas of different components. The areas are calculated based on the dimensions presented in Figure 2.4. Using these values, and the aforementioned MATLAB GUI/script(Figure 2.3), the A and B matrices were calculated.

$$A = \begin{bmatrix} -1.38 & 0 & 0 & 0 & 0 & 0.03 \\ 0 & -0.47 & 0 & 0 & 0 & 0.01 \\ 0 & 0 & -0.06 & 0 & 0 & 0.06 \\ 0 & 0 & 0 & -0.042 & 0 & 0.04 \\ 0 & 0 & 0 & 0 & -0.07 & 0.07 \\ 0.73 & 0.21 & 3.66 & 2.07 & 1.61 & -8.25 \end{bmatrix} \tag{2.17}$$

$$B = \begin{bmatrix} 0 & 0 & 0 & 1.3514 \\ 0 & 0 & 0 & 0.4607 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0.0008 & 0 \\ 0.0185 & 0.0185 & 0 & 0 \end{bmatrix} \tag{2.18}$$

So, the state space equation in Equation 2.19 looks like Equation 2.16.

$$
\begin{aligned}
\begin{bmatrix} \dot{T_{w1}} \\ \dot{T_{w2}} \\ \dot{T_f} \\ \dot{T_c} \\ \dot{T_{ip}} \\ \dot{T_{ai}} \end{bmatrix}
&=
\begin{bmatrix}
-1.38 & 0 & 0 & 0 & 0 & 0.03 \\
0 & -0.47 & 0 & 0 & 0 & 0.01 \\
0 & 0 & -0.06 & 0 & 0 & 0.06 \\
0 & 0 & 0 & -0.042 & 0 & 0.04 \\
0 & 0 & 0 & 0 & -0.07 & 0.07 \\
0.73 & 0.21 & 3.66 & 2.07 & 1.61 & -8.25
\end{bmatrix} \cdots \\
&\times
\begin{bmatrix} T_{w1} \\ T_{w2} \\ T_f \\ T_c \\ T_{ip} \\ T_{ai} \end{bmatrix}
+
\begin{bmatrix}
0 & 0 & 0 & 1.3514 \\
0 & 0 & 0 & 0.4607 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0.0008 & 0 \\
0.0185 & 0.0185 & 0 & 0
\end{bmatrix}
\times
\begin{bmatrix} Q_p \\ Q_e \\ Q_s \\ T_{ao} \end{bmatrix}
\end{aligned}
\tag{2.19}
$$

As mentioned before, there are four inputs to the system: Q_p,Q_e, Q_s, and T_ao, which are respectively the plant capacity, internal heat gains, solar glazing, and the outside air temperature. Using guidelines provided in [45], the plant capacity(Q_p) for heating was selected to be 30,000 BTU/hr. For the internal heat gains(Q_e), only the heat gains from the electric equipment listed inr̃eftab:electrical˙loads were used. For the sake of simplicity, and lack of available data, Q_s is neglected. The temperature data, T_ao is hourly data collected for Golden, CO by NREL [37]. For simulation, the data from the month of January and November was used.

With all the mathematical modeling of a building supporting a transient computer simulation as presented in this Chapter, it is possible to design a MAS based control system.

CHAPTER 3

SYSTEM DEVELOPMENT: MAS DESIGN METHODOLOGY

In this chapter, the methodology for the design of multi agent systems will be discussed. Multi agent systems are a fairly new technology of computer programming, system modeling, and control, but a formal procedure for system design is still not available. It is very difficult to have a unique methodology and some ad-hoc guidelines are proposed in this chapter, followed by a conceptual description of a design approach that is applicable for a building management system. Multi agent systems have provided a new perspective at distributed systems, and the development of more robust intelligent applications [46]. However, the traditional methods of software design, from planning to implementation, are not encompassing for the MAS paradigm. The main reason is that object-oriented- programming (OOP), is the closest paradigm in terms of functionality, but it still does not completely capture the potential and flexibility of MAS. On the other hand, the similarity between those two platforms allows for the design of MAS to use OOP design methodology as a base. This chapter discusses some of the MAS design methodologies that use OOP design methodologies as a first approach towards a procedure of formal design.

3.1 MAS Modeling Methodologies

With the increasing use of MAS in different fields, and several applications, new MAS design methodologies have been developed, either as an evolution of an existing one or sometimes as a new concept. Some of the common current methodologies present are the GAIA methodology [47], Belief-Desire-Intention (BDI) methodology [48] and Responsibility Driven Design (RDD) [49]. The following sections present an overview of each methodology along with their salient features.

3.1.1 GAIA Methodology

The Gaia methodology is an attempt to define a complete and general methodology that it is specifically tailored to the analysis and design of MAS. Gaia is a general methodology that supports both levels of the individual agent structure and the agent society in the MAS development process. In this methodology, MAS are viewed as a system composed of a number of autonomous interactive agents that live in an organized society in which each agent plays one or more specific roles [47]. Gaia defines the structure of a MAS in terms of a role model, based on the roles that agents have to play within the MAS and the interacting protocols between such different roles. Roles are defined with the following attributes: (i) responsibilities, (ii) permissions, (iii) activities and (iv) protocols.

Responsibilities are a key attribute related to a role that determines the functionality of an agent. Responsibilities are of two types:

- Liveness properties the role has to add something good to the system, and safety properties the role must prevent and disallow that something bad happens to the system. Liveness describes the tasks that an agent must fulfill given certain environmental conditions and safety ensures that an acceptable state of affairs is maintained during the execution cycle. In order to realize responsibilities, a role has a set of permissions.

- Permissions represent what the role is allowed to do and in particular, which information resources it is allowed to access. The activities are tasks that an agent performs without interacting with other agents. Finally, protocols are the specific patterns of interaction: what type of messages are sent? What frequency they are sent at, and how fast? [50].

3.1.2 Belief-Desire-Intention(BDI) Methodology

The BDI methodology for designing MAS is based on the belief-desire-intention model of human practical reasoning developed by Michael Bratman [51]. This is a model of human psychology used for explaining future-directed intentions. It has been used in computer science to model and develop algorithms in artificial intelligence.

In Object Oriented Programming (OOP) design, one can divide a system in three different models that are simultaneously developed to achieve a complete description of the system by three instances: *(i) the object model, (ii) the dynamic model, and (iii) the functional model*. The object model captures information about objects within the system such as their data structure, relationships and the operations that they perform. The dynamic model describes the states, transitions, events, actions and the interactions of the different objects, whereas the functional model describes the flow of data during system activity. The BDI methodology of MAS design is loosely based on this architecture, with the agents of the MAS being analogous to objects.

While designing MAS using this methodology, two viewpoints are taken in agent development: an external viewpoint, and an internal viewpoint. The external viewpoint deals with decomposing the system into agents, characterized by their purpose, responsibilities, the services they provide, and the interaction requirements between them. To streamline this process, the external viewpoint is further broken up into two parts: the agent model, and the interaction model [52]. The agent model describes the existence of agents in the system: how many instances of the same agent are present, when they are initiated, terminated. The interaction model deals with the responsibilities of the agent, the services, and the relationships between the agents. With the overall system of agents designed using the external viewpoint, the internal viewpoint deals with functionality of the individual agents and their decision making processes. This internal viewpoint is divided into three parts according to the BDI model. In the internal viewpoint, we have the Belief model, Goal model, and Plan model. The belief model describes the information about the environment the agent is in and its internal state. It also has information about the actions it may perform. The goal model as the name suggests describes the goals of the agent, as well as the events to which it can respond to. The plan model describes the plans that an agent may possibly employ to achieve its goals or respond to events it perceives [52].

3.1.3 Responsibility Driven Design (RDD) Methodology

The responsibility driven design (RDD) methodology is also based on the principles of object, dynamic, and functional models of OOP system design. The design procedure in RDD is divided into three different parts, analogous to OOP system design, but with the following realizations:

1. Agents can be more complex than objects.

2. Agents can be autonomous; they can act on their own behalf by following their goals, and can choose what events to react to.

3. Communication in a MAS is not only based on simple messages, but are more detailed and there are different message types and dialogues. There are also conversations between agents with respect to specific contexts.

With these differences between agents and objects as above, the RDD methodology proposes three models to design the complete MAS system:

1. The Agent Model contains the complete description of the agent, including the behavior of the agent in different situations, both reactive and proactive. This model is analogous to the internal viewpoint of the BDI methodology described in the section above.

2. The Organization Model contains information about the relationships between the agents. These can be relationships between similar agents that have inherited properties between them or between two completely different agents. The

relationships are not to be confused with actual communication between the agents.

3. The Cooperation Model contains information about the interaction, cooperation among agents. This is comparable to the dynamic model of OOP.

A clear advantage of using one design methodology over another for MAS design has not been discovered; all the methodologies are valid, but it seems they were all proposed with a specific MAS in mind, and were later generalized to become a design process for all MAS. Even though the methodologies mentioned above are different from each other in certain respects, there are common themes between them:

1. The complete system has to be analyzed, and the environment has to be designed for the deployment of agents.

2. Agents have to be defined according to the requirements of the system, and the information available from the environment, along with the relationships between the agents.

3. After agents and their relationships have been defined, designed, the appropriate communication requirements, ranging from infrastructure to protocol have to be chosen and designed.

With such common themes between the different design methodologies, the following section describes the design process used in developing the MAS for building energy management, in this thesis.

3.2 Model

For the design of the MAS model, a three level design technique was used. In order to keep the naming schemes used in the different methodologies mentioned before, the steps are in this thesis referred to *(i) the environment model, (ii) the agent model,* *and the (iii) communication model.* Those different models, and their development, along with examples pertaining to building energy management system will now be discussed.

3.2.1 Environment Model

The development of the environment model is a part of the analysis phase of MAS development. The main focus of this model is centered around the following specifications:

- What are the goals of the system ?
- What are the overall inputs/outputs of the system ?
- What will the system offer to the agents ?
- Can goals be divided to sub-goals ?
- What are the tasks required to fulfill goals ?

In developing the model used in this thesis, the overall system has to be looked at, and before setting out to design a MAS, the governing overall question, of 'what are the requirements?' has to be answered. Once it is known what the MAS has to achieve, the system can be analyzed further to decide which inputs will be important and what are the system outputs. For example, in order to design a MAS for the energy management system of a building, the goals of the system will be

- to increase energy efficiency,
- reduce energy related costs,
- and maintain comfort for the users of the building.

For such a system, some examples of inputs would be weather information, real-time energy pricing information. Outputs of such a system would be the control of the different physical systems of a building such as the HVAC, and lights. The building energy management system will be able to offer various sensor information such as temperature, humidity, and occupancy, to the agents as well as controllable physical devices. The goals mentioned before are larger goals of the entire system, which can be divided into smaller goals to tackle the problems in a more organized fashion. For example, the goal of reducing energy cost will have sub-goals such as *find if utility pricing has changed*. Further decomposing goals, arrives at tasks; an example of a task is *turn off unused lights*. There are differences between tasks and sub-goals: tasks are concerned with actions, where sub-goals are concerned with decision- making, which can be aided by tasks.

3.2.2 Agent Model

After defining the environment model design, goals, sub-goals, and certain tasks along with the overall inputs/outputs of the system as previously discussed, it is important to define how the agent deals with information available to it from the system, this is defined by agent model.

The agents are chosen to complete the tasks as described in the environment model, while working together with other agents to achieve sub-goals and as a consequence achieve the overall goals. The inputs and the outputs of the system are handled by individual agents. For example, the input of energy pricing is modeled as a pricing agent, whose sole task is to provide the pricing information to the system/other agents whenever required. The structure of the individual agents consist of three different parts: *inputs, behavior, outputs.*

3.2.2.1 Inputs

The inputs of the agent are composed of the information and data received to perform tasks. The inputs to the agents can be from other agents, or from the external environment. For example, the pricing agent described earlier, receives input from the external environment: the utility company. Another agent, the electric agent, deals with the control of the physical electric system of the building. The electric agent receives input from other agents, and one of them is the pricing agent. While designing individual agents, the input part of the structure should be designed by answering the following questions:

1. What inputs are required for agent to complete it's task ?

2. Where do those input come from ?

    - The environment ?

    - Other agents ?

3.2.2.2 Behavior

The next step in this design procedure is to develop the decision- making process. The behavior of the agent, i.e. how the inputs to the agent will be processed in order to perform actions and provide output must be defined. Agent behavior ranges from performing actions as simple decisions dependent on the input, to further advanced decisions that require interaction with multiple inputs in conjunction with a complex algorithm. Simple behaviors can be governed by IF-THEN logic rules, i.e. performing actions when inputs are received from the environment (rule base). The agents generally provide fast responses to the system [53]. Sensors of the system are also modeled as agents, and have simple behaviors. The agent that provides outside temperature data has outside temperature as its input from different sensors, or from databases on the Internet, and its behavior is to average those different input values. Advanced behaviors require more complicated decision making techniques compared to simple logic rules.

In the methodology adopted in this work, advanced behaviors are modeled similar to the approach of the belief model (BDI methodology). An agent can be envisioned as being in a certain state at any given time, and has the ability to move between these states depending on changes in the environment, or any other input it might receive. Depending on how many states are available, which is dependent on what the agent can perceive in the environment, and what state it is currently in, coupled with the input information, the agent will have different preferences of moving between states.

In MAS theory, a common simplifying assumption is that an agent's preferences are captured by a utility function [6]. It is analogous to the *von Neumann-Morgenstern utility function* discussed in game theory [54]. Such utility function assigns a index to how much a particular agent likes a particular state. If S is a set of all the states in the environment the agent is in, and the utility function of an agent i is given by:

$$u_i : S \to \mathbb{R} \tag{3.1}$$

When the agent has to change states from a state s to a state s', it calculates the expected utility of moving to this new state by taking a certain action a. This expected utility is calculated by first defining the probability of the agent reaching state s' from state s using the action a: $P(s, a, s')$. This probability has to be defined/calculated/learned because in real world situations, change of state might not happen exactly as planned. In an environment where S is a set of all states, the expected utility of an agent's move is given in Equation 3.2.

$$E[u_i, s, a] = \sum_{s' \in S} P(s, a, s') u_i(s') \tag{3.2}$$

A decision making process that involves uncertainty can be optimized by using Markov Decision Processes (MDP) [26]. In the case of an agent, one can assume the choice of the new state to depend on the agent's current state and the agent's action. At each state, choice of action and consequently the probability of transitioning to the next state are stochastically controlled. A reward function is also incorporated into the system. This function rewards the agent when the most optimal state is reached.

It is necessary to define a policy for every agent; the agent has to always be aware of the maximum available utility and reward. This awareness and search of the maximum expected utility of an agent can be viewed as its search for an *optimal policy*, and can be denoted mathematically as in Equation 3.3.

$$\pi_i^*(s) = \arg \sum_{a \in A} E\left[u_i, s, a\right] \tag{3.3}$$

The optimal policy presented in Equation 3.3 is missing one key aspect: rewards, both positive and negative. A reward system, containing a maximum reward of one, and a discounted reward awarded to the agent (between 0 and 1) for every state change solves problems such as the agent moving unnecessarily between states to obtain maximum rewards. With a discounted reward system, the agent will not move between states unnecessarily because it will decrease its maximum expected utility in Equation 3.3. This can be represented mathematically (Equation 3.4), and defined as the real utility that an agent receives for being in state s.

$$u(s) = r(s) + \gamma \max \sum_{s'} P\left(s, a, s'\right) u(s') \tag{3.4}$$

Here, $r(s)$ is the reward the agent receives for getting to state s, and γ is the discount factor for movement between states.P is the probability of movement between states as described earlier in Equation 3.2. Equation 3.4 is also known as the *Bellman Equation* [6]. It represents a MDP definition, as the agent's utility depends not only on its immediate rewards but also on its future discounted rewards. To solve for $u(s)$, we use the idea of value iteration, and arrive at the *Bellman update* equation:

$$u^{t+1}(s) \leftarrow r(s) + \gamma \max \sum_{s'} P(s, a, s') u^t(s') \tag{3.5}$$

The solution to Equation 3.5 will give us the policies of the agent, including the optimal policy. Using a value iteration algorithm, the optimal policy for the thermal agent is calculated in the next section.

For the energy management system designed in this thesis, a thermal agent is needed for handling the HVAC. The thermal agent has been designed with 4 states and 3 possible actions as outlined in Table 3.1.

Table 3.1: States and Actions for Thermal Agent

State	**Description**
1	Ideal Temperature
2	Too Cold
3	Too Hot
4	Intermediate
Actions	**Description**
1	Decrease furnace strength
2	Increase furnace strength
3	Do Nothing

Next, the transition functions (probability) for the movement between states for the agent were defined. A non-deterministic model is used as is the case in the real world. A certain action in a certain state might not always cause the agent to go to a particular state, even though it is very likely to. For instance, the transition function for movement to state 4 from state 1 by taking action 1, which translates to the probability of going to the state of *intermediate* from the state of *ideal* when furnace strength is decreased, is 0.7. A 0.3 probability is assigned to the possibility of the agent going directly to a state of *too cold*. Table 3.2 lists some of the other transition functions defined for the thermal agent, and Figure 3.1 presents all the probabilities assigned.

Before the value iteration algorithm was applied to calculate the policy of the agent, a stopping point was defined. The evolution between the states will stop depending on the maximum change δ allowed in utility values of the states. The relationship between δ and the discount factor γ is given in 3.6, where ϵ is the error tolerance.

$$\max\delta = \frac{\epsilon(1-\gamma)}{\gamma} \tag{3.6}$$

Table 3.2: Some transition probabilities for thermal agent

s	a	s'	T(s,a,s')
1	1	2	0.3
1	1	4	0.7
1	2	3	0.3
1	2	4	0.7
1	3	1	1

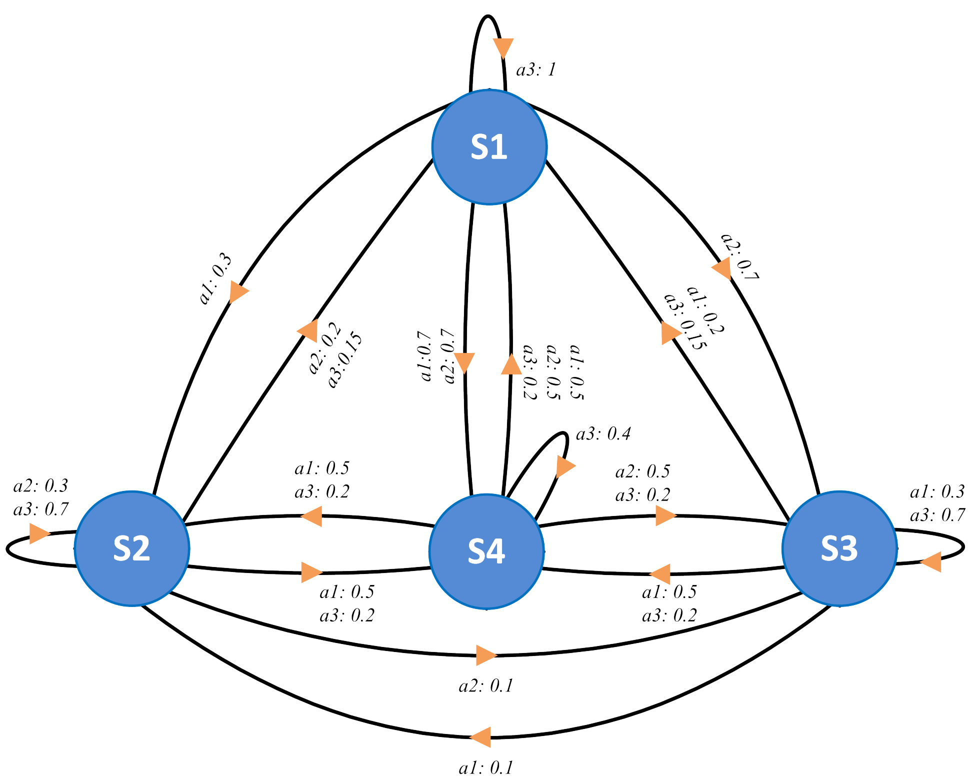


Figure 3.1: Transition probabilities for thermal agent

The initial values for $u(s)$ were set to zero, a maximum allowed error of 5% was specified along with a discount factor of 0.5(the agent is penalized 50% of it's reward every time it moves between states). This gives us a maximum δ of 0.05 (from Equation 3.6). The results of the value iteration algorithm is presented in Table 3.3.

Table 3.3: Utility values through each iteration

	u(s)			
Iteration	1	2	3	4
0	0	0	0	0
1	1	0	0	0
2	1.5	0.1	0.1	0.25
3	1.75	0.22	0.22	0.4
4	1.875	0.2990	0.2990	0.4925
5	1.9375	0.3458	0.3458	0.5435
6	1.9688	0.3716	0.3716	0.5708

The algorithm stops after six iterations because the change in utility values of state 1 between iteration 5 and 6 is $1.9688 - 1.9375 = 0.0313$, which is below the maximum δ of 0.05. This is shown graphically in Figure 3.2.

Another scenario was considered where the discount factor was prescribed to 0.7 (agent is penalized only 30% per move between states), which changes the maximum allowed δ to 0.0214. The same value iteration algorithm was applied, and the results are presented in Figure 3.3.

As can be seen, the utility function converges (goes below the maximum allowed *delta*) after 12 iterations. According to our definition of the problem before, this was expected. As there is less penalty for moving between the states, the agent is going to try to maximize utility but considering more moves than in the first case, where the penalty for movement was higher.

With the final utility values obtained, the policy of the agent can be calculated using Equation 3.3. The policy of the thermal agent is presented in Table 3.4.

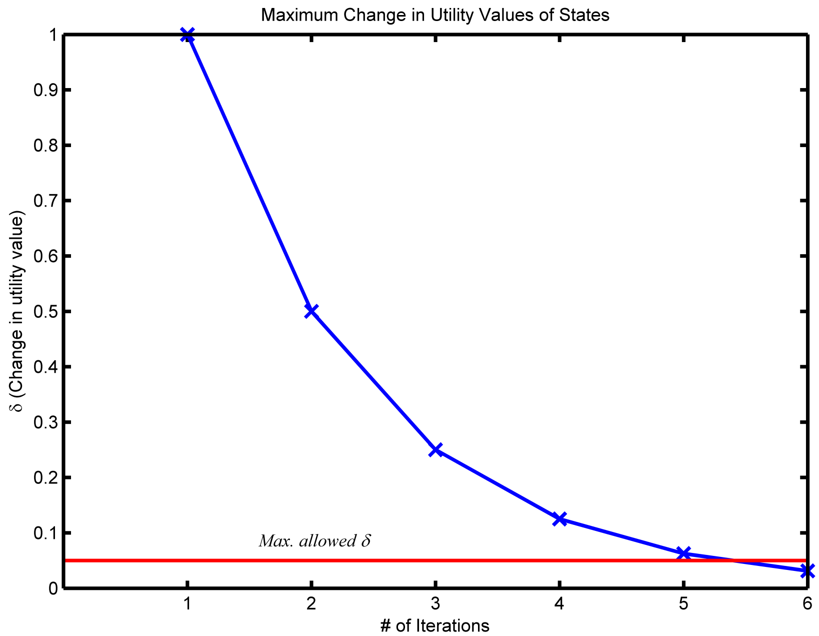


Figure 3.2: Change in δ through iterations (ϵ=5%, γ=0.5)

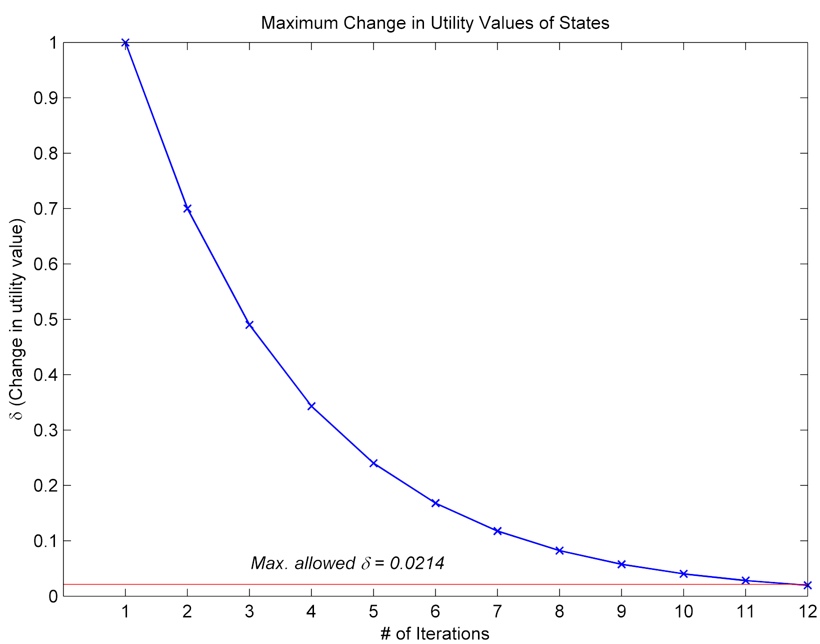


Figure 3.3: Change in δ through iterations (ϵ=5%, γ=0.7)

Table 3.4: Thermal Agent Policy

State	Preferred Action
1	3
2	2
3	1
4	1

The results translates to the agent doing nothing (Action 3) while in ideal temperature state (State 1), and would increase temperature (Action 2) when in State 2. This decision that has been reached via the iterative solution, is exactly what an ad-hoc solution would suggest. This validates the value iteration algorithm used for solving the bellman update equation, and in turn calculating the policy of an agent.

The value iteration algorithm has been used to solve the Bellman update equation and hence calculate the policy of an agent. As seen in Figure 3.2 and Figure 3.3, the agent performs as expected, and an optimal policy is chosen. However, the optimal policy that is reached is dependent on the different states of the agent and the actions possible for it to take. It does not depend on actions or states of other agents that are present in a multi agent system. To incorporate the effect of other agents in the system, and to calculate an optimal policy based on that, a few different techniques can be used.

The most simplest technique would be to incorporate the effect of the agents into the transition values (T). In a simple case where there are few agents, with a relatively low number of states and actions, this is feasible. However, as the agents get complex and increase in number, the calculation of transition functions incorporating the effect of all the agents present in the environment becomes extremely difficult, if not impossible. Also, it was defined earlier that agents are intelligent, and have the ability to learn. With this definition, it becomes even more difficult to calculate the agent policy using the above mentioned value iteration algorithm as the policies will have to be changed because of what has been learned by the agent over time, or also because of just different inputs it has started receiving. Machine learning algorithms can overcome this problem, and help in building multi-agent systems where each of the agents learn how to do a specific action.

Machine learning algorithms for multi agent systems are defined as *the algorithms that increase the ability of an agent to match a set of inputs to the corresponding outputs* [55]. If we assume there is an input/output pair {a,b} out of a set of such pairs E, where a $\in$A, and b$\in$B, where A is the set of inputs and B is the set of outputs, the learning function must map A$\rightarrow$B for every input/output pair {a,b}.

For any problem, multiple learning algorithms will correctly map every element in set A to B, but some algorithms will be more efficient than others. Given the same set of input/output pairs, two learning algorithms can learn to perfectly classify them but will have learned different mapping functions. This effect is called the *induction bias* or a learning algorithm [6]. This *induction bias* is what causes some algorithms to perform better than others in certain sets. But, also as defined by Wolpert and Macready in [56] in the *no free lunch theorem* [56], there is no one learning algorithm that will out perform others all the time.

In a MAS, there is no fixed set of input/output pairs {a,b} as discussed earlier; because the agents are always in different states, and taking different actions, the input/output pairs keep on changing. As seen in earlier sections, an agent in a multi-agent application might not know the reward that it will receive for the different actions, and a random action must be taken to explore the world, to find the best payoff and find the preferred action.

The *Q learning* algorithm developed by Watkins and Dayan [57] is a popular learning algorithm for use in an agent based system. It is used to find the optimal policy of a agent, similar to the value iteration algorithm. The concept of the algorithms are very similar except for the major difference in learning. The value iteration algorithm has no learning abilities, but *Q learning* introduces two new concepts: the learning rate, and the exploration rate. The learning rate is a measure of how much emphasis is put on new rewards, compared to the learned values, whereas the exploration rate is required to make sure the agent considers other actions, and the rewards associated, without converging to a solution too quickly. These values are both normalized to be in between 0 and 1. The problem associated with the Q learning algorithm is the same as what was found with the value iteration algorithm: only one agent is considered at a time. In the Q learning algorithm, only one agent can learn at a time, and collective learning is not possible.

A way to incorporate actions and states of multiple agents is to reach an equilibrium point different than that desired by the Q learning and the value iteration algorithms. A *Nash Equilibrium* can be defined for a MAS. In game theory, *Nash equilibrium* is a solution concept of a game involving two or more players, in which each player is assumed to know the equilibrium strategies of the other players, and no player has anything to gain by changing only his own strategy unilaterally [54]. In the case of a MAS, the Nash equilibrium point is the set of policies for a set of agents, such that no one particular agent, acting alone, will gain anything by changing its policy from the equilibrium point. It has been shown in [54] that a Nash equilibrium point always exists. An algorithm that can achieve a Nash equilibrium is presented in Appendix C.

3.2.2.3 Output

With the inputs and the behavior of the agents that makes decisions based on the inputs designed, the next step is to analyze the outputs of the agent. After the decision is made from the behavior section of the agent, what are the outputs and how are they to be presented in the system. The output of the agent deals with the answers to the following questions:

1. What are the outputs of the agent ?
2. How is the output seen by the system ?
   - Direct action on the physical system ?
   - Information provided to another agent(s)?

For example, the output of a comfort agent in a building, which decides what is the current level of user comfort in the building, has an output of comfort level. There is no direct action onto the physical system, but instead, the information about the current comfort level is provided as input to other agents that might require it in the system. On the other hand, a thermal agent that controls the building furnace will modulate the physical furnace strength output i.e. the amount of Btu/hour required by the furnace.

3.2.3 Communication Model

With the agent model completed in the design methodology, the next step is to look at both the environment and the agent models, and the relationships present in them, agent-agent relationships and agent-environment relationships. The communication model of the design methodology is concerned with the flow of data between the agents and the environment to maintain these relationships. Both the inputs and the outputs of the agent model have to be re-visited to decide on which data is to be transmitted. In communication, care needs to be taken to maximize the communication with the least bandwidth. This leads to making decisions answering the following questions:

1. What will be the frequency of communication between agents/environment ?
2. What communication speed is required for agent functionality ?
3. What type of data needs to be communicated ?

The communication model is more concerned with the implementation of the MAS, and the answers to the questions above will help in selecting the communication protocol, technology and the content of the messages that are passed between the different agents and the environment.

As a conclusion for this chapter, the flowchart in Figure 3.4 presents the MAS design methodology used in the thesis. As discussed before, the analysis phase begins with the environment model, where the system is studied, and specifications are set. The analysis phase continues on to the Agent model, where goals and sub-goals are studied further, and broken down into tasks. These recognition of these tasks eases the transistion to the design phase of the MAS design methodology. Here, the individual agents are defined: inputs, behaviors, and outputs. The behavior of the agent is crucial in agent development, and different algorithms, techniques are studied and selected according to the behavior of the agent. The final part of MAS development is the communication model, where according to the inputs, behaviors, and outputs of the agent, the communication requirements are studied, and appropriate communication protocols are selected for use in the MAS.

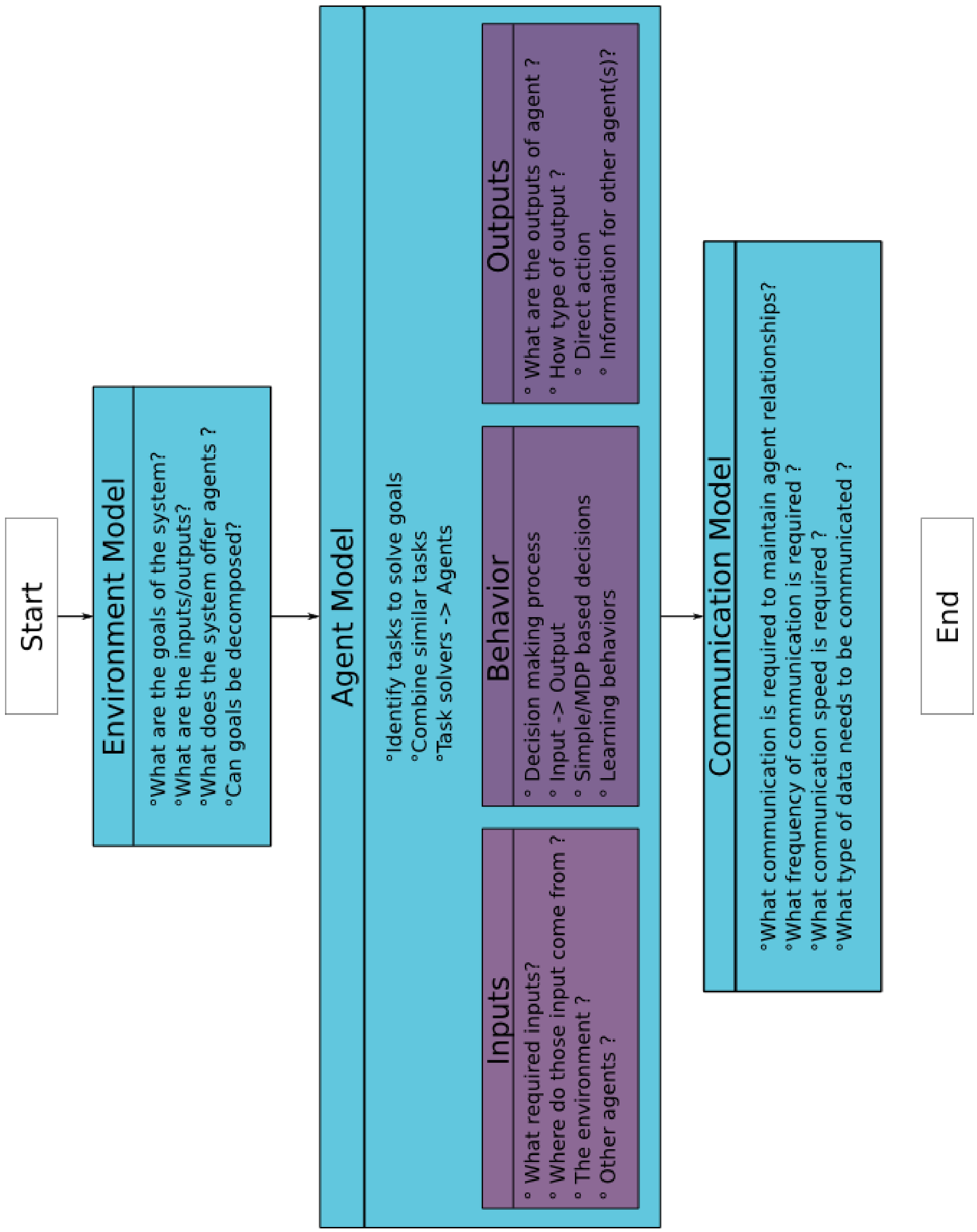


Figure 3.4: Flow Chart Outlining the MAS Design Methodology Used in Thesis

CHAPTER 4

IMPLEMENTATION: THERMAL MODEL AND MAS

In this chapter, a building model has been developed and using a MATLAB and Simulink based environment in order to allow full-fledged co-simulation with a multi-agent based control. The MAS methodology outlined in Chapter 3 will be used to design and implement a MAS based temperature control of a proposed commercial building.

4.1 Building Model

The mathematical model for the thermal performance of a section of a commercial office building was developed Section 2.2. An equivalent thermal circuit, analogous to an electric circuit was developed, and so were the corresponding differential equations. Using traditional mathematical techniques, the differential equations have been stacked using State Space notation, and the following matrices were obtained (Equations 4.1 and 4.2).

$$A = \begin{bmatrix} -1.38 & 0 & 0 & 0 & 0 & 0.03 \\ 0 & -0.47 & 0 & 0 & 0 & 0.01 \\ 0 & 0 & -0.06 & 0 & 0 & 0.06 \\ 0 & 0 & 0 & -0.042 & 0 & 0.04 \\ 0 & 0 & 0 & 0 & -0.07 & 0.07 \\ 0.73 & 0.21 & 3.66 & 2.07 & 1.61 & -8.25 \end{bmatrix} \tag{4.1}$$

$$B = \begin{bmatrix} 0 & 0 & 0 & 1.3514 \\ 0 & 0 & 0 & 0.4607 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0.0008 & 0 \\ 0.0185 & 0.0185 & 0 & 0 \end{bmatrix} \tag{4.2}$$

For a complete state space solution, we also require two more matrices: C and D. Matrix C is the output matrix, and for the purpose of this thesis, we are only concerned with the temperature of the air inside the room (T_{ai}), as opposed to the other states that were defined in Section 2.2. Since only the last state of all the states is required, C can be defined as:

$$C = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.3}$$

The matrix D is a feed-forward matrix, and since in our system we do not have direct feed-through or feed-forward, the D matrix is a zero matrix.

$$D = \begin{bmatrix} 0 & 0 & 0 & 0 \end{bmatrix} \tag{4.4}$$

4.1.1 Open Loop Test

With the State Space matrices defined, the *State Space Model* block of Simulink was used to create a prescribed model of the building. The model has four inputs as discussed in the previous chapter:

1. heat from solar glazing,

2. heat gained internally in the building,

3. outside temperature,

4. and the building heating system.

The output of the system (from the definition of the C matrix) is the inside air temperature. To test the operation of the mode shown in Figure 4.1, a very simple open-loop system was designed in Simulink for simulation.

For this test, the external temperature is kept at a constant 65°F, and the heat gain from solar glazing and the internal heat gains are neglected. The furnace capacity, as discussed in Chapter 2 is selected to be 30,000 BTU/hr. The output of the system is plotted, and presented in Figure 4.2 (the temperature of the air inside the building T_{ai}).

From the figure, one can see the indoor temperature rises to $70°F$ in less than half an hour. Given that the outdoor temperature was set to a $65°F$, and all the internal temperatures were initialized to 32°F, this is verified as a reasonable result.

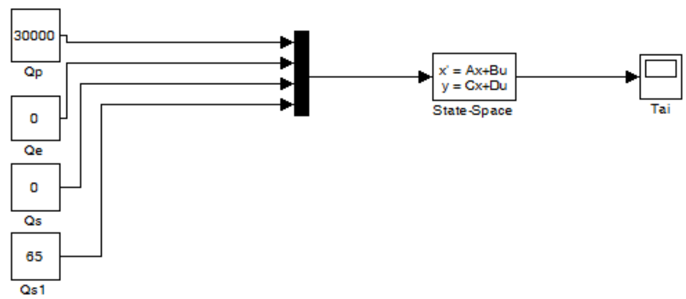


Figure 4.1: Simulink Model for Open Loop Test

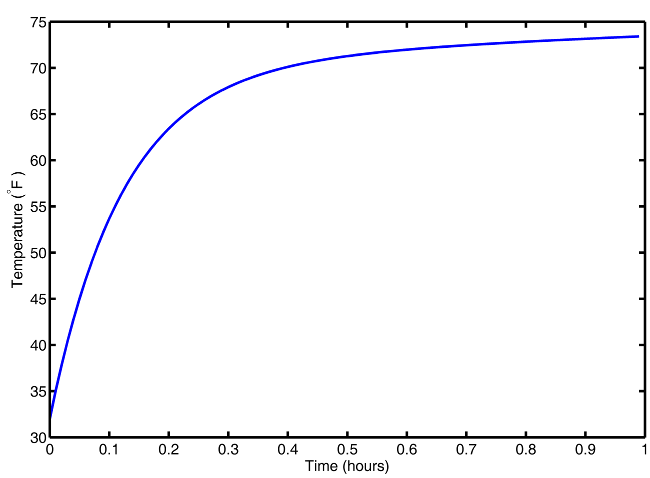


Figure 4.2: Results of an open loop simulation.

4.1.1.1 Initial Conditions

The performance of the building thermal system in simulation is directly related to the initial conditions. There are no specific initial conditions of a building and thus the initialization value of 32°F was adopted in this study. The effect on performance due to different initial conditions have also been investigated. In Figure 4.3, it can

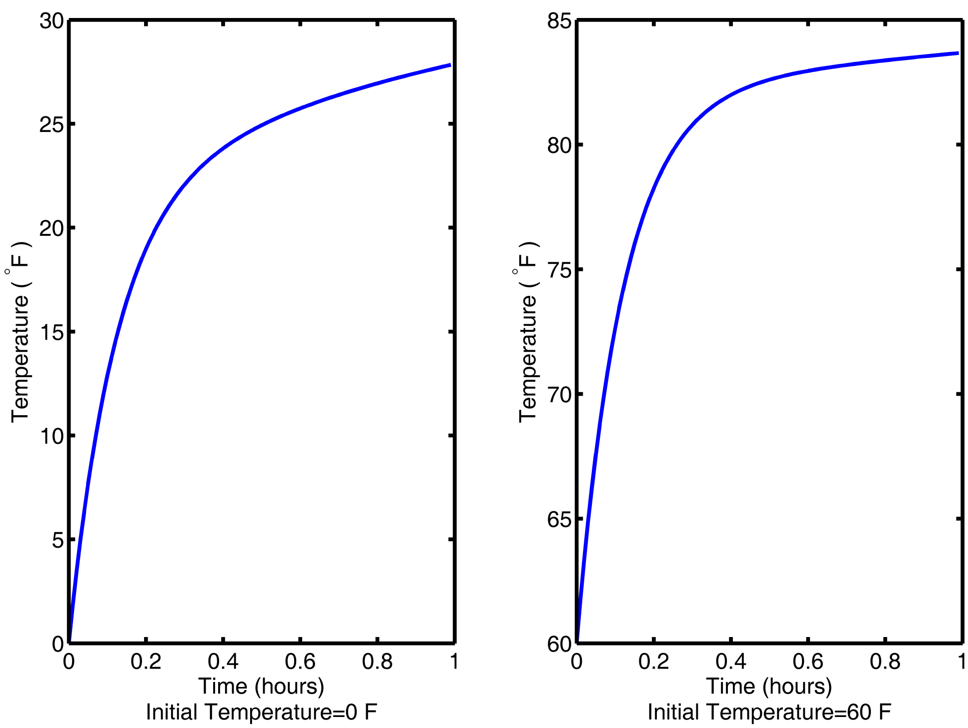


Figure 4.3: Open loop performance with different initial conditions

be clearly seen that the rise time is affected by the different initial conditions. This is a very intuitive result: if the building is already warm, it will rise to a set higher temperature faster than if it were at a very cold initially. However, it should be noted that the envelop of the curves in both Figure 4.2 and Figure 4.3 remain the same. Moreover, it is very unlikely that once a building has been built and is in operation, that the initial condition will have to be dealt with many times. Thus, the change in performance in the open loop simulation is noted, but is neglected, as it is a condition not likely to occur in real-life situations. In addition, because the development of the control for this thesis is more concerned about the energy usage in a building in the longer run compared to the first couple of hours of operation. As a result, the performance of the building model looks promising.

4.1.2 PI Control

Before implementing a MAS based control scheme for the building area developed earlier, a simple PI based control was applied to the building model and the performance was analyzed. The set point for temperature was set at 72°F for the working hours (between 8 AM and 5 PM) and otherwise at 65°F. The furnace and the sensors were modeled as time delays in the system as there will always be a lag in sensing as well as in furnace operation. The simulink model is shown in Figure 4.4. For the sake

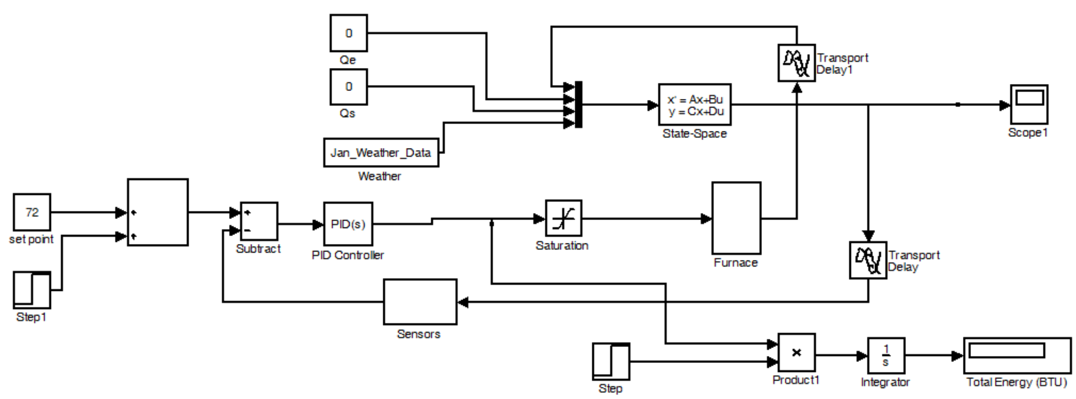
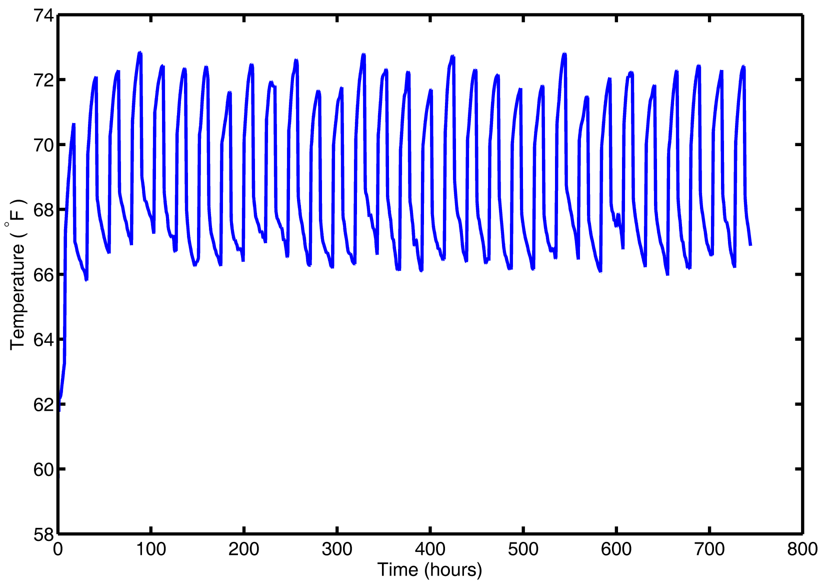


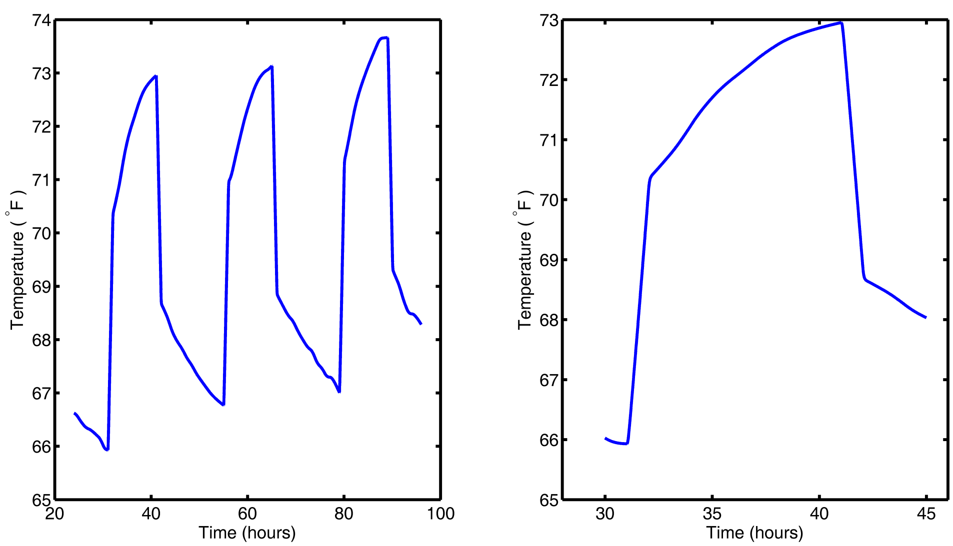
Figure 4.4: PI based temperature control

of simplicity, the heat gain from solar glazing and the internal temperature gains are neglected. But, unlike the open loop test in Section 4.1.1, the outside temperature was not set to a constant value. Hourly temperature values for the month of January, collected by NREL [37] was used as the input temperature. The simulation was run for the entire month of January, and results can be seen in Figure 4.5.

Looking at just Figure 4.5(a), it is tempting to invalidate the performance of the system, because it seems the range of the internal temperature to be large. But,



(a) Internal Air Temperature (T_{ai}) change under PI Control, for January (744 hours)



(b) Internal Air Temperature (T_{ai}) change under PI Control, between hour 30 and 45

Figure 4.5: Internal Air Temperature change under PI Control

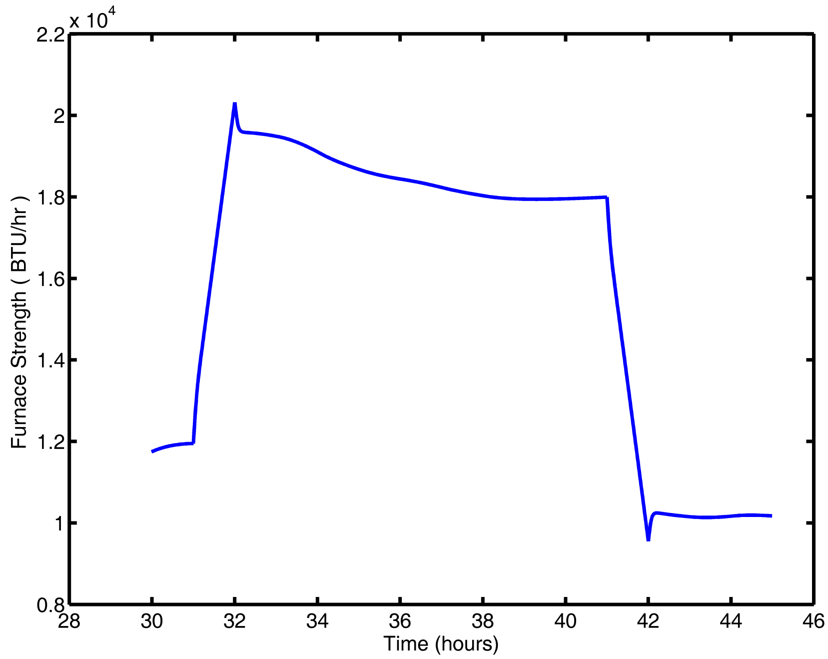
since Figure 4.5(a) is the result for the entire month, and there are different set points for different times of the day, a closer look has to be taken, as in Figure 4.5(b). It is noticeable here that the temperature range is not that high, and control is quite good. Hour 30 corresponds to 6 AM as it is assumed that hour 0 is midnight. So from 6 AM to 7 AM, the temperature is to be kept around 65°F and after that to increase to 72°F, until the end of the day, corresponding to hour 41, and then decrease to 65°F again till the start of the next work day.

4.1.3 Energy Use

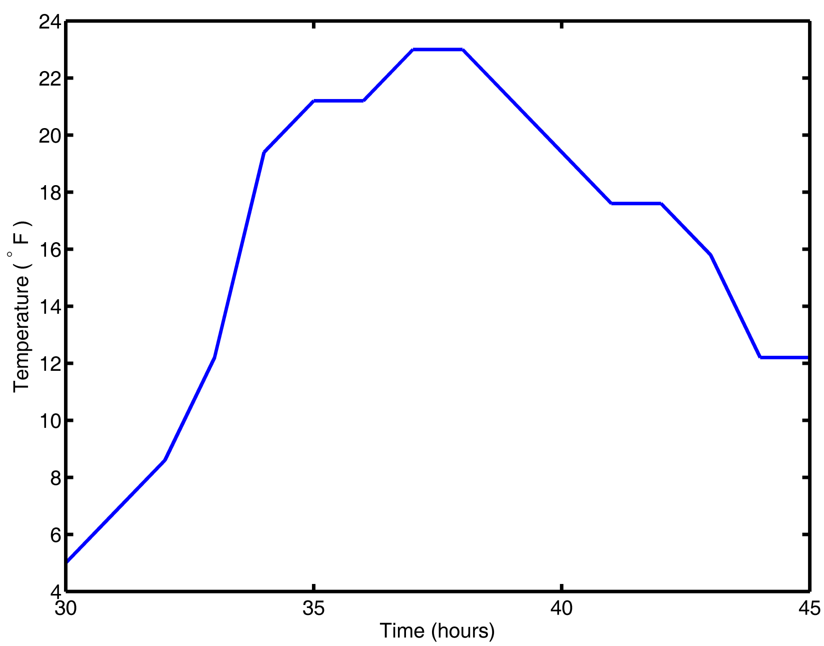
It is important to calculate the energy use (in BTU). In the PI control scheme in Figure 4.4, the output of the *PID controller* block of SIMULINK is the furnace control, i.e the level of the furnace strength. If this furnace strength is integrated over a time period, the total energy usage in that particular time period is obtained (furnace strength is in BTU/hr, so the integral gives the BTU or energy). The furnace signal for hours 30 and 45 is shown in Figure 4.6(a), along with the external temperatures during that period in Figure 4.6(b). It can be observed that the furnace strength decreases as the external temperature is increasing, which is consistent with the physical conditions. The chosen time period here was a during a very cold period in Golden, CO, as the input data suggests. The temperature does not go above 30°F for a number of days, and so drastic changes in furnace control are not seen. But in terms of the operation of the model, it is as expected. The furnace strength does not cross the maximum value of 30,000 BTU, that was specified in 2. The integral of this signal, for the entire month (744 hours) yields an energy usage of $5.44e^6$ BTU for the entire month.

4.1.3.1 Comparisons

To validate the above calculated monthly energy consumption of $5.44e^6$ BTU, a building of similar dimensions and material characteristics was designed in an energy



(a) Furnace Signal for hours 30 to 45



(b) External Weather for hours 30 to 45

Figure 4.6: Furnace Signal and External Weather for hours 30 to 45

simulation tool developed by the Lawrence Berkeley National Laboratory (LBNL) and the United States Department of Energy (USDOE). This software, the QUick Energy Simulation Tool (eQUEST), allows users to design and perform detailed analysis of today's state-of-the-art building design technologies using today's most sophisticated building energy use simulation techniques [58]. The interface is very intuitive where one can see The 3D model of the designed building, and the different energy usage information obtained is presented in appendix D. For direct comparison with the energy usage data calculated earlier, Figure 4.7 is very useful. The eQUEST software was provided with the same weather information, and similar usage patterns as before.

The energy used for only space heating is 6.94e^6 BTU, which is higher than the

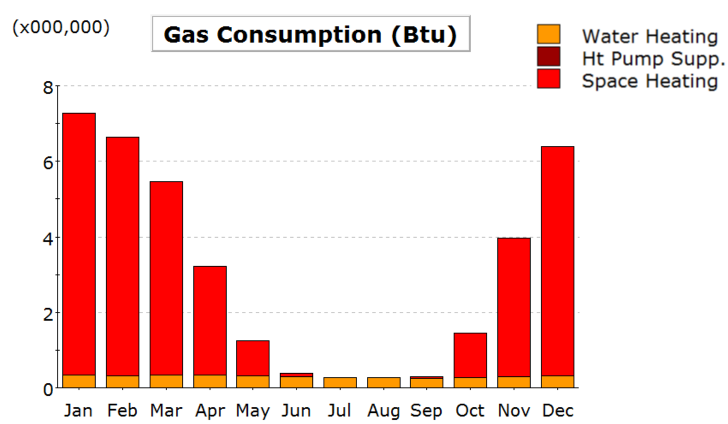


Figure 4.7: Energy usage of building designed in eQUEST

values obtained from simulation. This discrepancy is not because the PI control used in this thesis is a more efficient control system, but because of the inability to replicate exact building parameters used in the modeling methodology in eQUEST. Materials having exactly the same characteristics as used in the model developed in Chapter 2 were not available in eQUEST. But, something that showed to be useful was to observe that the order or magnitude is the same in both cases,validating the model developed in Chapter 2.

4.2 MAS Design

With the building model constructed and implemented, the next step is to develop the MAS to control the temperature in the area modeled. In order to have a complete MAS design, a methodology has to be followed. For this work, the methodology for MAS design was developed in Chapter 3. Following the flow chart, presented in Figure 3.4, we follow the following steps to design a MAS for temperature control in the building area.

4.2.1 Environment Model

As described in Chapter 3, the development of the environment model of the MAS focuses on answers to the following questions:

- What are the goals of the system ?
- What are the overall inputs/outputs of the system ?
- What will the system offer to the agents ?
- Can goals be divided to sub-goals ?
- What are the tasks required to fulfill goals ?

For the environment model of the MAS we are developing to control the temperature of the building, the goal of the system is to maintain the temperature of the area of the building in concern at a comfortable level for the users. Inputs are essential to keep the temperature at a comfortable level; the inputs to our system are: *weather information (external temperature), human occupancy information, current temperature (internal temperature), sources of heat*. The main output of the system is the *control of the furnace of the building*. The various sensor data available is the overall system's offering to the agents.

The overall goal can be divided into some sub-goals. For example, a sub-goal can be: *how much heat is produced from office equipment?*. Similarly an example of a task is to *control the furnace of the building.*

With this general idea of a thermal system environment of a building present, the next, and most important, step is to develop the agent model.

4.2.2 Agent Model

After identifying goals, sub-goals, tasks and the inputs/outputs of the system, the different agents can be developed. The main goal of the system of maintaining thermal comfort has three major sub-goals:

1. Control the thermal system of the building, making informed decisions using the inputs available.

2. Calculate the extra heat in the system from electrical appliances

3. Know what the current comfort level is.

With these three main sub-goals in mind, three major agents were chosen: *the thermal agent, the electric agent, and the comfort agent.*

The electric agent, is responsible for providing data to the other agents, specifically the thermal agent, about the amount of heat produced by the electric devices and appliances in the building area. The inputs to this agent are device usage characteristics throughout the day, as all the devices are not in use all day, as well as the efficiency data of the different appliances, which is required to calculate the heat produced (the assumption of energy is wasted solely as heat is made). In this case, the efficiency data presented in Table 2.3 is used as input, along with typical occupancy behavior from the eQUEST simulation software. The behavior of the electric agent is very straight forward, and is an algebraic behavior, where the inputs are used mathematically to calculate the output. The output of the electric agent is a single BTU/hr value that is provided to the thermal agent.

The thermal agent, is concerned with the thermal performance of the building. In this example, the thermal agent, instead of being interconnected with an actual physical system, is connected with the mathematical model for the thermal system developed before. There are many inputs to this agent: weather and occupancy information, current comfort level, and other heat sources except the furnace. The information about other heat sources are available from the electric agent as discussed before. The other inputs are available from the sensors in the building, which are modeled as agents as well, and will be discussed shortly.

The behavior for the thermal agent is more complicated than the electric agent, as the agent has to decide what comfort level is selected, assessing the different inputs to make a decision on which actions to take. The Markov Decision Process that is described in Section 3.2.2.2 is used for the decision making in the thermal agent. The procedure for developing the MDP is the same as described before, and the policy is calculated. Instead of using arbitrary values for the transition values as in the example before, the transition values are calculated using the open loop response of the thermal model presented in Figure 4.2.

The output of the system is a change-in-the-internal-temperature, as well as the control command to the furnace. The change in the internal temperature will act as an input to the internal temperature sensing agent, and the control of the furnace will be a direct action to a physical system of the building.

The comfort agent is responsible for keeping track of the current comfort level in the building. The input to this agent is the current average temperature, and defined comfort levels. The temperature input is received through sensors, and the defined comfort levels are preset to 72°F for active building hours, and 65°F otherwise. The behavior of this agent is a simple IF-THEN rule base statement, where the temperature is compared with the comfort level temperature, and the comfort level is decided. The output of the agent is the current comfort level, which will act as the input to the thermal agent.

The temperature sensor (both external and internal) are also modeled as agents, but they have very simple models, as the inputs is the data that they obtain, which is passed to the behavior section of the agent. The behavior depends on the number of a type of sensor present; if there are multiple external temperature sensors present, the average is taken, and provided as output, otherwise a direct output=input relationship is maintained by the behavior section of the agent.

4.2.3 Communication Model

The communication model is, as described in Chapter 3, is dependent on the answers to the following questions:

1. What will be the frequency of communication between agents/environment ?
2. What communication speed is required for agent functionality ?
3. What type of data needs to be communicated ?

For this example, the frequency of communication has to be high between the sensor agents and the comfort agent, as maintaining the thermal comfort is the main goal of the MAS and the sensors are the way to find out what the comfort level is currently. The frequency of communication between the electric and heating agent can be flexible, as it heat produced by electric appliances does not change very fast, and follows a pattern. The speed is of concern when this MAS has to be implemented in a actual physical environment, but now, since it is simulation, the speed of communication is not an issue. Most of data to be communicated is numerical or text data. As this example is relatively simple, complex data structures for communication do not need to be created.

This section concludes the design of the MAS for maintaining thermal comfort in a building. Next, a discussion on the actual software implementation of the MAS is discussed.

4.3 MAS Implementation in Software

There are several MAS development platforms available, such as NetLogo [59], which has been developed by researchers at Northwestern University. It is a very useful MAS simulation environment, using the Logo programming language, which does not allow it to use many new libraries available in other programming languages such as C, C++, Java, and Python. Logo, is, however, an easy language to learn and makes NetLogo an ideal MAS implementation platform for quick simulation studies. The Java Agent DEveleopment Framework (JADE) is another such MAS development platform, which as the name suggests, is based on the Java programming language. Java is a widely used programming language for different purposes. There is a lot of support, as well as a diverse pre-written libraries, are readily available. Java has built-in, robust multi-threading libraries available to use, and JADE takes full advantage of this feature. Also, for the previously mentioned problem of lack of socket programming in MATLAB, this is not present in Java, and consequently not available in JADE as well. Therefore, the author of this thesis decided to choose JADE as the MAS implementation platform mainly because of their programming and software robustness.

By choosing a different platform than MATLAB for the development of the control system, the problem of interconnecting the building model (developed in Simulink) and the MAS control arises. To overcome this problem, the possibility of developing the building model completely in Java was explored, but due to the rigorous programming required, compared to the relatively easy implementation in Simulink, the idea has been dropped (but maybe it could be explored by a computer science focused research). Using the previously mentioned MEX files and the Python interface, the development of a middleware to connect the JADE models to the Simulink models was also explored, but because of the extensive low level programming knowledge required, and the very difficult task of keeping track of simulation time between the two platforms, this idea has been also left.

It has been found through literature review that a middleware, similar to the one described in the previous paragraph, has been developed by Charles Robinson at the University of York [60]. This middleware, has been named as MACSimJx. It performs exactly what is required for the control implementation required in this thesis. It provides two-way communication between the Simulink model and the control system, while perfectly keeping track of simulation time. When running a simulation, time is very important, as it dictates the sequence of events, and is vital in calculating process timings and other timing data. When a software simulation invloves two different platforms, like we have in this work, the same simulation time has to be maintained between all the platforms. It it extremely important for the building simulation in SIMULINK and the control system in JADE share the same time, otherwise the connection between the two platforms will be useless. All the JADE functionalities are available throughout the system, as well the Simulink capabilities. Figure 4.8 graphically shows the interaction between Simulink and the JADE agents. The MACSimJx

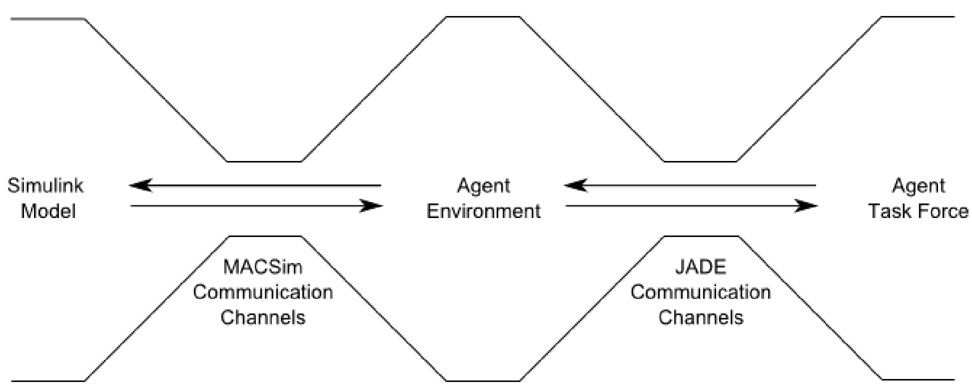


Figure 4.8: The Interconnection of Simulink and JADE using MACSim

middleware uses the Multi-Agent Control for Simulink (MACSim) engine, which is a medium through which a program for implementing agent designs developed in C/C++ or Java might pass data to and from Simulink. It uses a client/server architecture: the MACSim client runs within Simulink (as a S-Function block) and the server runs as a standalone software, which has open socket connections for multithreaded MAS implementation software such as JADE. MACSimJx is specifically designed to make the connection to JADE easier. The Agent Task Force in Figure 4.8 is another name for the collection of agents and their interactions. A screen shot of the initialization of MACSimJx, along with Matlab and JADE is shown in Figure 4.9.

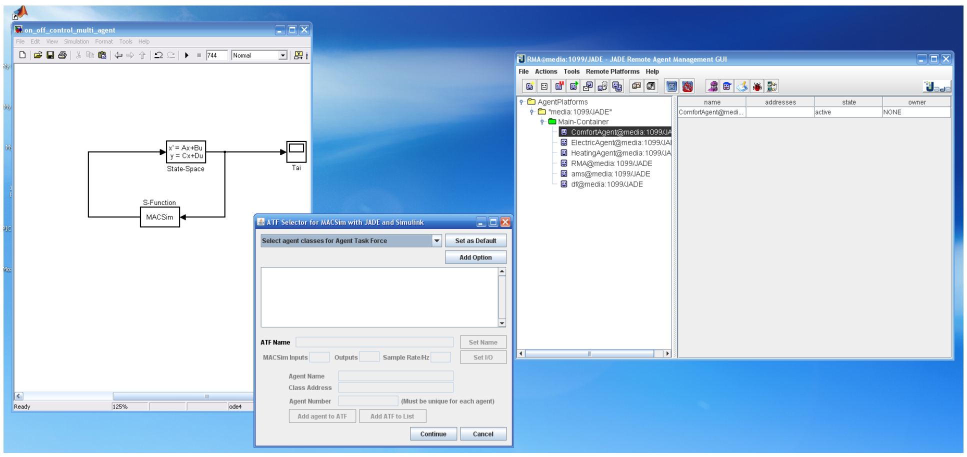


Figure 4.9: A screen shot of the initialization of MACSimJx

4.3.1 Agent Implementation in JADE

With the agents designed and ready, JADE is used for implementation. JADE has a system of agent implementation that is very easy to understand and learn. The programming language used is JAVA, and most of the common agent functionalities and behaviors are provided as in built library functions.

4.3.1.1 Behaviors

JADE makes the development of agents easier by separating the two most important aspects of an intelligent agent: the agent behavior class, and agent communication class. As the name suggests, the behavior class of an agent in JADE is concerned with the actions of the agent. The behavior class is further divided into three main categories:

- One-shot behaviors (jade.core.behaviours.OneShotBehaviour)

- Cyclic behaviors (jade.core.behaviours.CyclicBehaviour)
- Generic behaviors (jade.core.behaviours.SequentialBehaviour)

The names of the different categories are self explanatory. Using these different behavior categories, the algorithms mentioned in 4.2.2 can be implemented. The Markov decision process algorithms such as the value iteration algorithm is implemented using the generic behavior class present in JADE. One-shot behaviors are generally used for emergency situations, where a single action needs to be taken once when a certain input is received or when the environment the agent is in changes.

4.3.1.2 Communication

JADE implements a *asynchronous message passing* communication paradigm. In this paradigm, every agent has a quasi mailbox, which is also known as the agent message queue [61]. In this queue the JADE run time posts messages that are sent by the other agents. Whenever a message is posted in the queue, the receiving agent is notified. The timing to read and use the message can be dictated in the design of the agent. Since a protocol has to be used to keep messages standard across agents, and avoid miscommunication, JADE employs the famous Agent Communication Language (ACL). Syntax for the ACL language is very intuitive and documentation easily available, making implementation very easy. In the simulation for this work, the messages being passed by agents are not very complex, and generally consist of plain text. For these purposesl ACL is appropriate, as it accomodate these messages, and the functionality is in-built in JADE, making it rather easy to implement.

With the mathematical and computational modeling and implementation of the different models conducted as discussed in this chapter, the implementation of MAS using Matlab/Simulink and MACSim has been outlined in this chapter. In the next chapter, the control implementation of the proposed MAS is combined with the developed thermal model of the building and several case studies will support their performance improvement.

CHAPTER 5
RESULTS AND DISCUSSION

In this chapter, results for the implementation of temperature control for a commercial building, as supported by model developed in Chapter 2 and using the MAS control strategy developed in Chapters 3 and 4 is presented and discussed. Various scenarios for building control are evaluated, and the performance in terms of energy usage in the building area plus the computational performance of the control system are discussed. The first couple of scenarios deals with simple control strategies using MAS architecture, and analysis is done to compare it with the traditional control methods presented in Chapter 4 in order to corroborate the proposed design methodology in Chapters 3 and 4. The performance of different MAS strategies are compared and, eventually, an advanced event scheduling MAS strategy is implemented, involving multiple thermal zones of the building, showing how the energy usage can be improved with some knowledge of occupancy and some degree of user interface capability provided by the event scheduling MAS control.

5.1 Case 1: Simple Reactive MAS Control

In order to test the MAS based control, the PI control presented in Section 4.1.2 is replaced with a MACSimJX client block (represented by a s-function) in Simulink. The complete Simulink model is shown in Figure 5.1. The Simulink model although

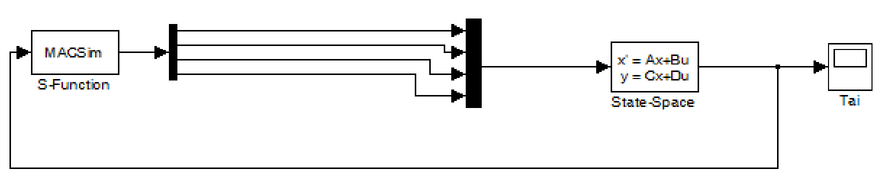


Figure 5.1: Complete Simulink Model using MAS Control from MACSimJX

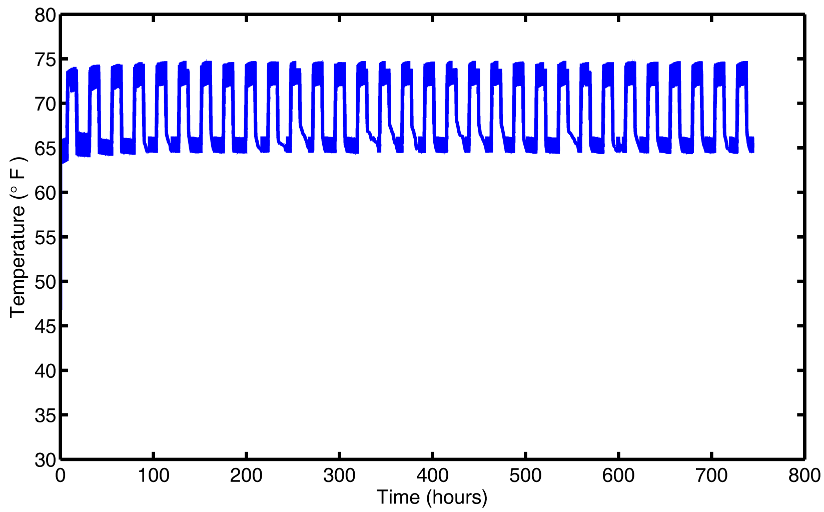
looks very simple, has all the sophisticated control algorithms and decision making is embedded into the MACSim S-Function. The state space model of the building requires 4 inputs (weather information, heat gain from solar glazing, heat gains from internal sources, heat from furnace), and the output of the MACSim block has 4 corresponding output variables. The MACSim block seen in Figure above contains the functions to run the MACSim server, and connect with JADE; it is actually the complete MAS with all the agents, and their behaviors.

For this scenario, a simple bang-bang control for the furnace was used. The weather data was the same as used before in Section 4.1.2. As mentioned in Section 4.2.2, the electric agent's behavior was programmed to by cyclic with a typical electric demand for a typical office setup. The electric agent's output is connected directly to the state space block's Qe input. For this simple case, the fuzzy logic based comfort agent was not used, specific temperatures of 72°F and 65°F were used as described in Section 4.1.2. The results obtained for temperature control for this scenario is presented in Figure 5.2(a) and Figure 5.2(b).

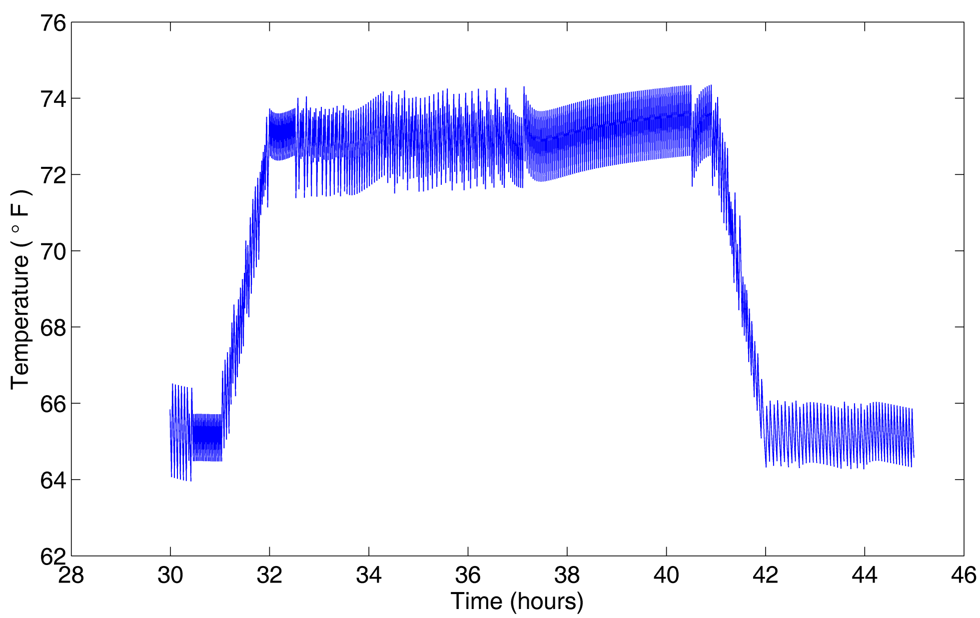
Comparing the results in Figure 5.2 to Figure 4.5, the overall response of the system variables(time, T_{ai}) are seen to be similar, but with a lot of noise present in the MAS based output. This noise is present because of the on/off based furnace control, and rigid set points for the temperature in the comfort agent. This noise results in comparatively fast fluctuations in the indoor air temperature, which may not be desirable in terms of comfort. Even though the average of the temperature over time will result in the desired temperature, the constant fluctuation will make the building users very uncomfortable.

5.1.1 Energy Usage

To calculate the energy usage, the concept of integrating the furnace signal introduced in section 4.1.3 is used. Performing the integral for the furnace signal for the entire month of January, gives 5.128×10^6 BTU. This value is very close to the



(a) Internal Air Temperature (T_{ai}) change under simple MAS control, for January (744 hours)



(b) Internal Air Temperature (T_{ai}) change under simple MAS control, between hour 30 and 45

Figure 5.2: Internal Air Temperature change under simple MAS control

value calculated using the PI control. Even though this total energy usage value is lower than that calculated under PI control, it is only due to the nature of the on/off control used, and is not an advantage introduced by a MAS. The furnace signal for hours 30 to 45 is presented in Figure 5.3. A value of 1 corresponds to the furnace being on and 0 to it being completely off. While integrating, the ON value was taken to be 30000 as that is the full strength of the furnace (30,000 BTU/hr).

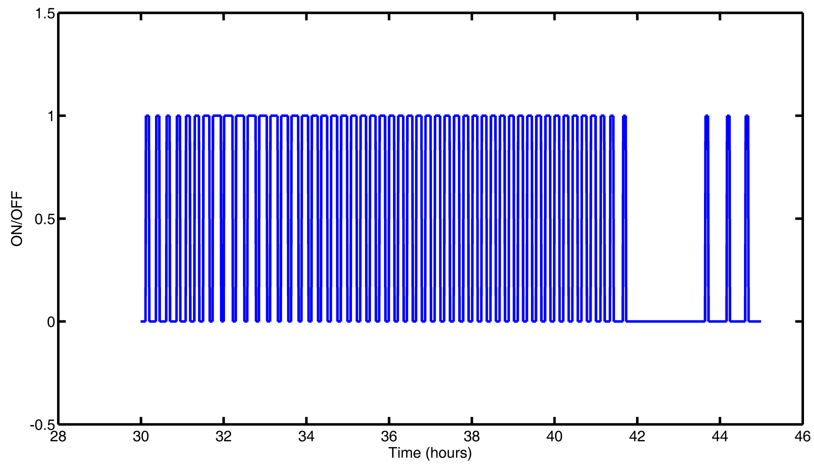


Figure 5.3: Furnace signal between hour 30 and 45

5.1.2 Communication

JADE has a pre-installed agent named as *Agent Sniffer*, which allows for the sniffing, or viewing of the communication between the agents as the agents operate. The sniffer agent has been activated for some cycles to observe the communication between the three agents in the simulation. A screenshot of the sniffer agent is presented in Figure 5.4 The *df* agent that is seen in the figure is *directory facilitator* and it acts as a yellow pages for the enviornment, having all the communication information available, and also acts as the agent that supplies environment values (in this case the sensor information from the building model). Hence, we can see the comfort agent requesting information the df agent, which in this case would be temperature information or T_ai from the model. The red lines indicate when an agent

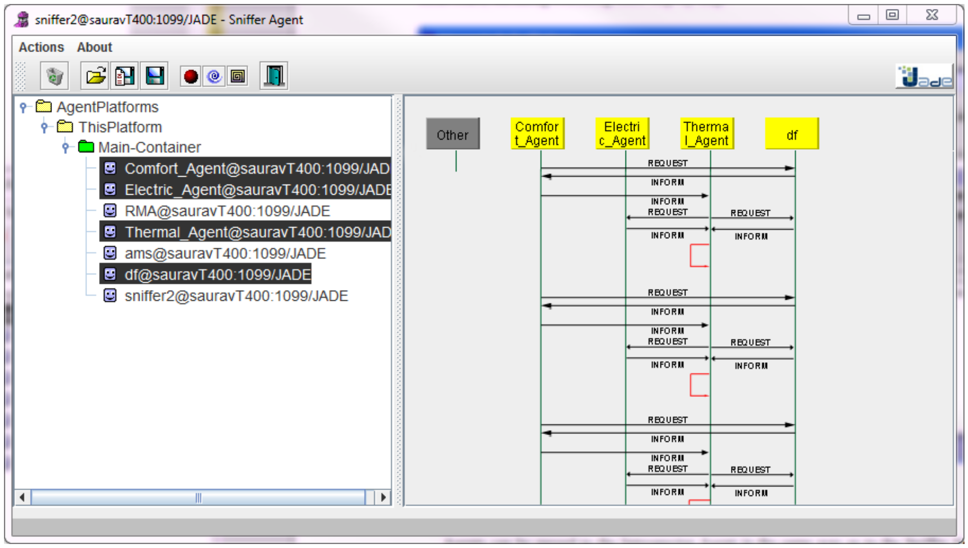


Figure 5.4: The sniffer agent detailing agent communication for case 1

takes an action, so the red loop for the thermal agent represents the thermal agent controlling the furnace.

Just as expected, the comfort agent first requests temperature information, and when it is informed about it, and it calculates that the comfort level has changed, it informs the thermal agent, which at that time requests information from both the df and the electric agent for weather, and electric demand data. once that is received, it takes appropriate action on the furnace, either turning it on or off.

5.2 Case 2: Fuzzy logic based comfort agent

A fuzzy logic based agent has been implemented in order to prevent fluctuations in the internal temperature of the building. The behavior of the comfort agent had to be changed in order to incorporate a range and minimize the rapid fluctuations (chattering) in the furnace signal control. The design methodology presented in Chapter 3 has been followed, but no changes in the environment were made; the only modification was in the behavior section of the comfort agent. The inputs and the outputs of the comfort agent were to remain the same as before.

So, to define the new comfort levels, fuzzy logic membership functions were used. Figure 5.5 details the membership function for the output. As discussed in Chapter 3 and 4 four regions are defined for the temperature level, and consequently the comfort level: too cold (TC), ideal(ID), too hot(TH), and intermediate. Two membership functions were used for the intermediate state (INC, INW). The entire

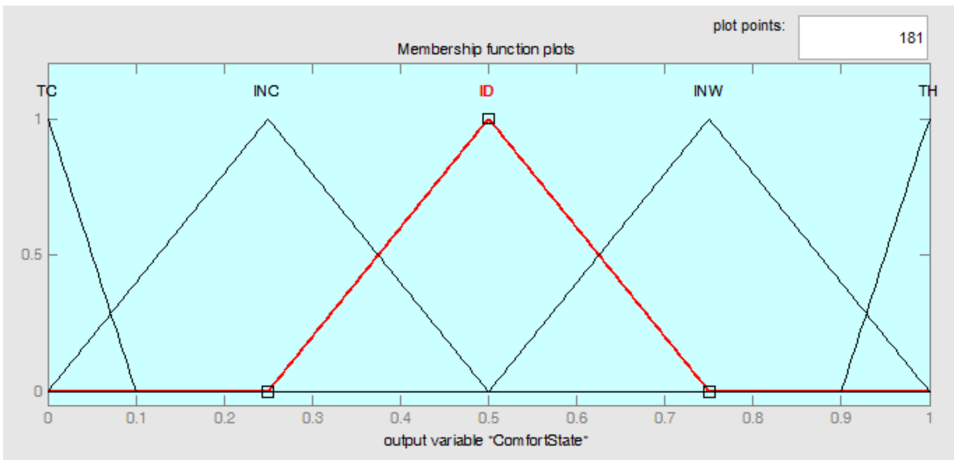
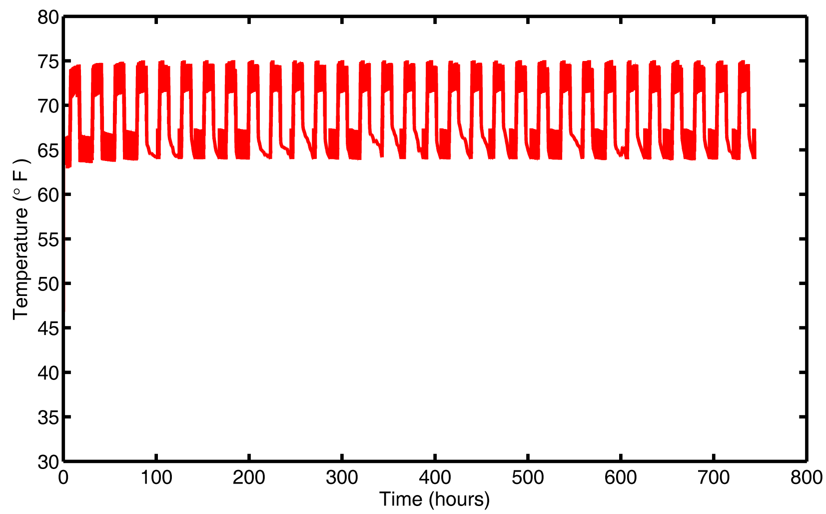


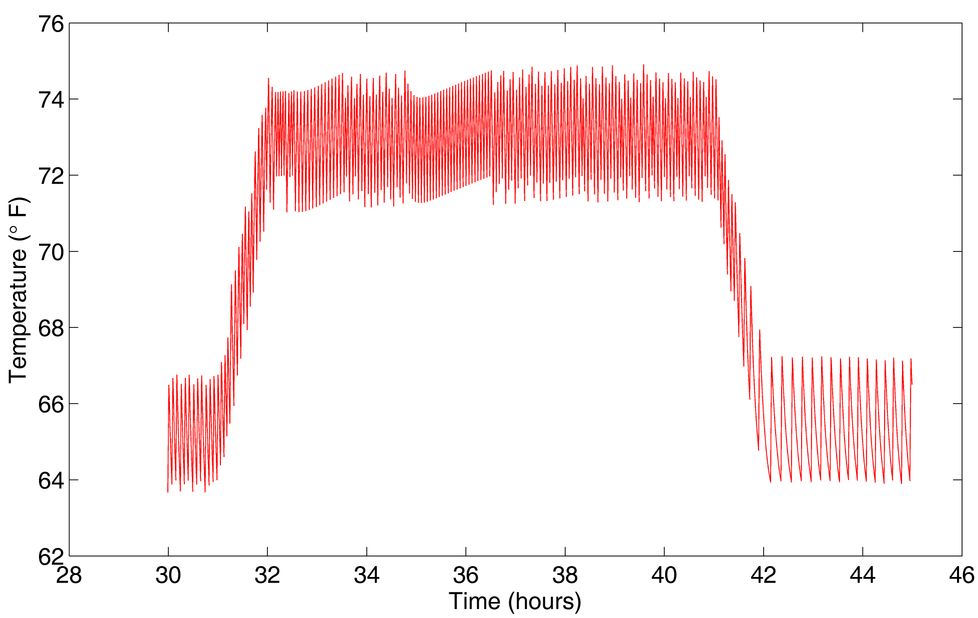
Figure 5.5: Output Membership Functions for Fuzzy Logic Behavior of Comfort Agent

fuzzy logic subsystem in the comfort agent, including the implementation of the mamdani type fuzzy inference system was done using the *jFuzzyLogic* open-source library available for Java [62]. The jFuzzyLogic library is very easy to use for someone who is familiar with the Fuzzy Logic Toolbox in MATLAB. The developers of this toolbox were inspired by the MATLAB toolbox, and have tried to make every functionality from the toolbox available for the Java environment. For implementing the mamdani type inference system, a set of rules to map the input membership functions to the output membership functions were defined. There is a detailed and complete fuzzy model, plus the set of rules in the CD-ROM accompanying this thesis.The following results were obtained, with the modified fuzzy based behavior of the comfort agent (Figure 5.6).

The fluctuations have decreased when compared to the previous results, and has increased the temperature tolerance of the building. This is because there is a proper



(a) Internal Air Temperature (T_{ai}) change for January (744 hours)



(b) Internal Air Temperature (T_{ai}) between hour 30 and 45

Figure 5.6: Internal Air Temperature change under MAS control with fuzzy based control agent

range owing to the membership functions, rather than the set points of 72°F and 65°F as before, as allowed by the fuzzy membership. The increase in tolerance can be seen in Figure 5.7. In addition, observing the furnace signal(in Figure 5.8), shows less

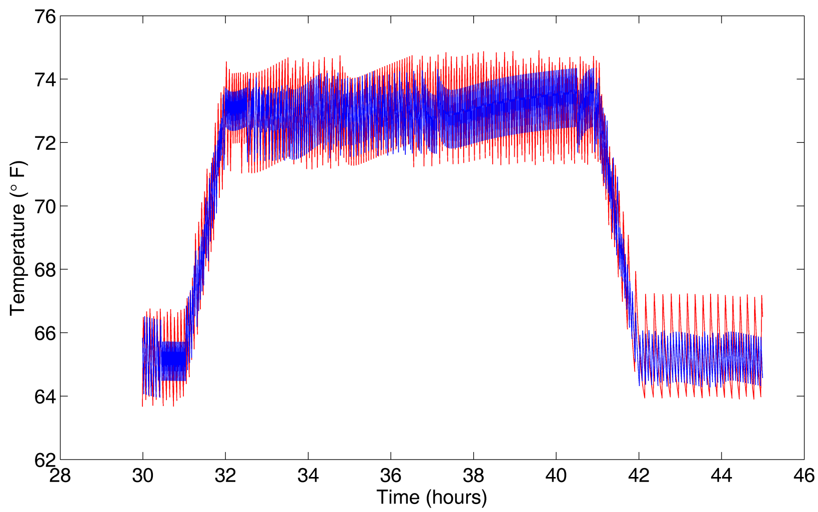


Figure 5.7: Internal Air Temperature with and without fuzzy based behavior

fluctuations in the actual furnace signal, which in turn results in better performance and improved reliability (since there are less cycles of turning on and off).

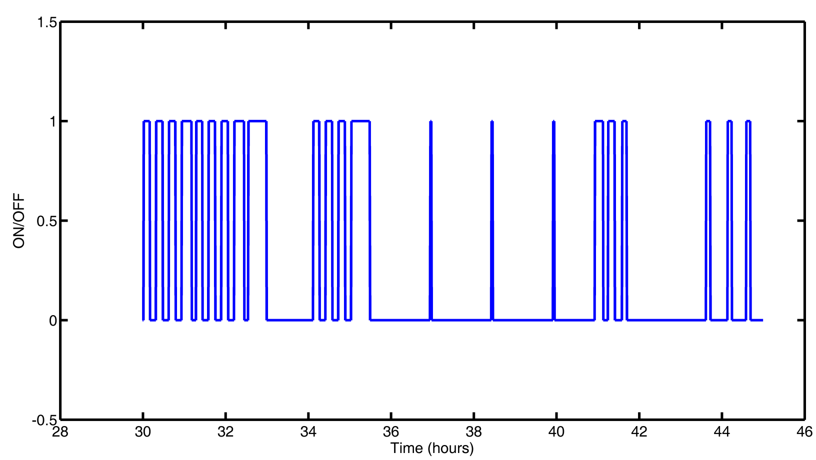


Figure 5.8: Furnace signal when fuzzy based comfort agent is used

5.2.1 Energy Usage

As explained previously, the energy usage can be calculated by integrating the furnace signal over time. For the fuzzy logic enabled MAS control system, the total energy usage for the month of January was calculated to be 4.981×10^6BTU. This value can be compared to the energy usage value in the previous case as the only change in the current case was the implementation of the fuzzy logic system. Since the furnace is not switching constantly like previously, the energy usage is lower by (2.87%). This decrease in energy usage is possible because of the advanced performance attributed to fuzzy logic behavior in the comfort agent.

5.2.2 Communication

The sniffer agent in Jade makes it easy to visualize agent communication. The sniffer agent for this case is presented in Figure 5.9. Although there is more commu-

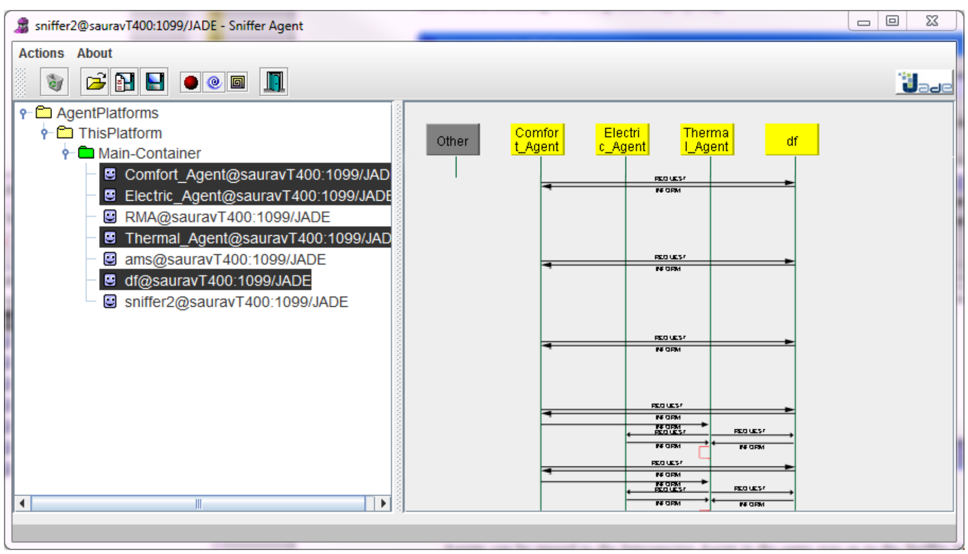


Figure 5.9: The sniffer agent detailing agent communication for case 2

nication between the df and comfort agent about temperature, only two actions on the thermal agent are seen. This corresponds with the graphs observed earlier on the switching of the furnace and the temperature control. Due to the fuzzy membership functions, change in the comfort level is not so frequent as it was in Case 1, showing some improved performance of the system operation.

5.3 Case 3: Revising MAS with new agent

In order to demonstrate that the proposed design methodology for MAS described in Chapter 3 can be used in redesigning an existing MAS without complete system overhaul, a new goal is added to the system: *scheduling meetings for building users while minimizing energy usage.* In this case, the following assumptions are made: that there are 2 meeting rooms available, in two separate parts of the building, with different thermal characteristics. This requires that the thermal system of the new meeting room in a different part of the building be modeled. Figure 5.10 shows the floor plan of the other part of the building considered in this case. In this section of

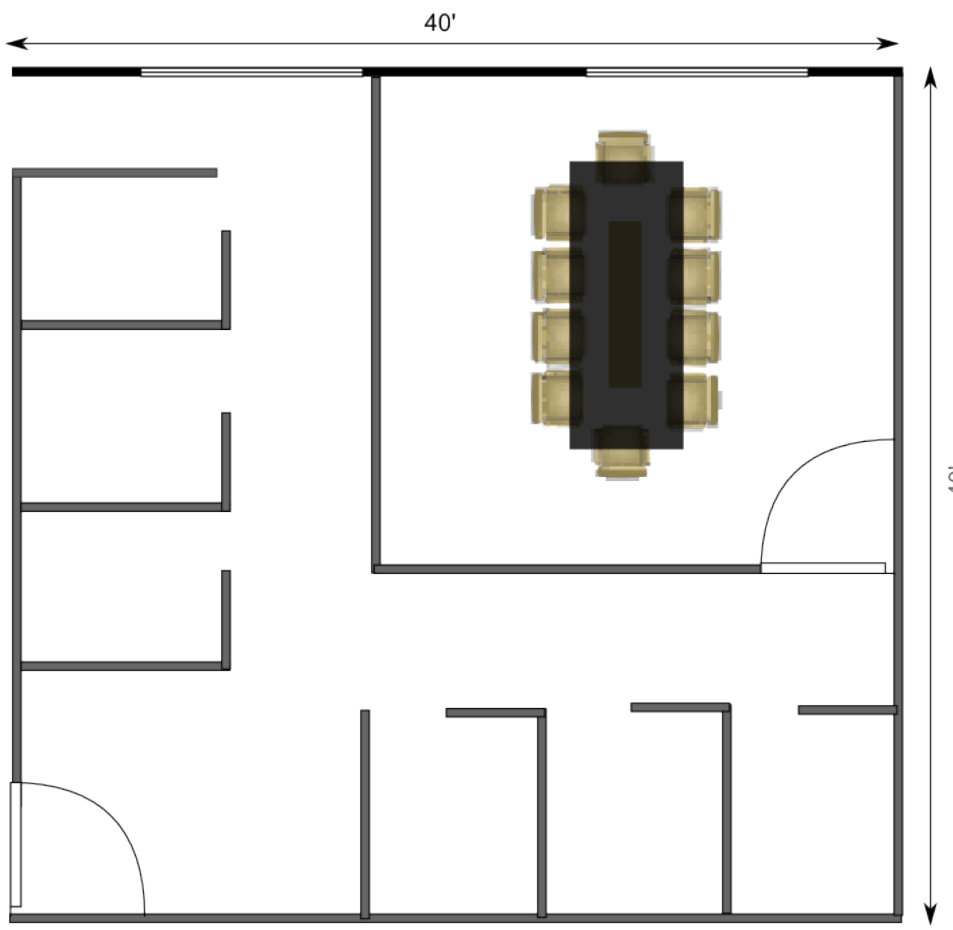


Figure 5.10: Floorplan of a different section of the building

the buildling, there are mainly cubicles, as well as a conference room. There is no kitchen as the previous area, and office equipment is limited to personal workstations of the building users in that area. The occupancy information of this area is also provided. The matrix developer script discussed in 2 is used to find the A and B matrices (Equations 5.1 and 5.2).

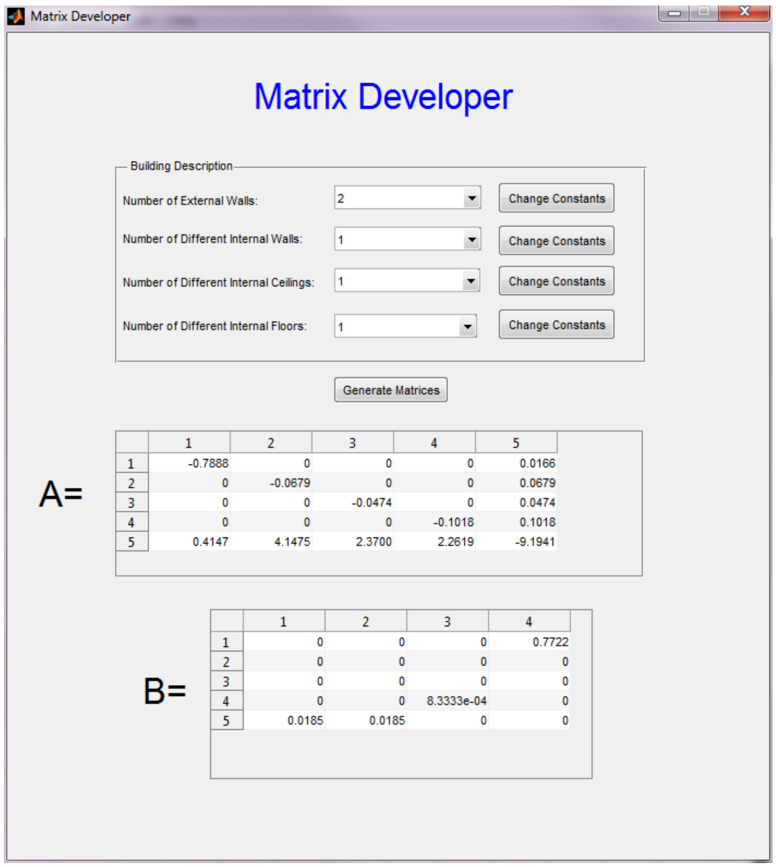


Figure 5.11: Matrix Developer to Calculate A and B

$$A = \begin{bmatrix} 0.7888 & 0 & 0 & 0 & 0.0166 \\ 0 & -0.0679 & 0 & 0 & 0.0679 \\ 0 & 0 & -0.0474 & 0 & 0.0474 \\ 0 & 0 & 0 & -0.1018 & 0.1018 \\ 0.4147 & 4.1475 & 2.3700 & 2.2619 & -9.1941 \end{bmatrix} \tag{5.1}$$

$$B = \begin{bmatrix} 0 & 0 & 0 & 1.3514 \\ 0 & 0 & 0 & 0.07722 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0.0008 & 0 \\ 0.0185 & 0.0185 & 0 & 0 \end{bmatrix} \tag{5.2}$$

With the physical system designed, we follow the MAS design methodology to complete the design.

5.3.1 Environment Model

A new goal has been added to the system: Schedule meetings for building users while minimizing energy usage. The physical system has also changed, as there is a new section of the building; which increases the amount of inputs available. Even though the characteristics of the inputs are the same, there are a set available, the same inputs of sensor data of the new building area are also available. Since there are two different areas, and the new goal has to deal with both of them, a new sub-goal is defined: compare energy usage between two zones. This can be further divided into a task: calculate energy usage in zone. With new goals, sub-goals, and tasks defined, the agent model can be developed.

5.3.2 Agent Model

The addition of a new building area to the model requires new agents to be created. Before considering the new goals and tasks, just adding the new building area to the physical model will require an additional instances of the same agents developed before. So, there will be a comfort, thermal, electric agent for this new zone, that are separate from the ones created before, but are functionally identical. This reiterates a common property of a MAS that there is a possibility of a lot of instances of similar agents, just like objects from classes in a OOP paradigm.

The next step is to address the new goals and the related tasks of the system along with the new inputs to the system such energy usage information of both zones. The major agent that needs to be added to address the new goal is meeting agent .The meeting agent will be responsible for the scheduling of the meeting according to the energy usage. The inputs to the agent will be a meeting time and energy usage at said time in both zones. The meeting time is a user input; the building user who wants to

Table 5.1: Zone selection according to meeting times

Time	Zone
9:00 AM	Zone 2
11:00 AM	Zone 1
1:00 PM	Zone 1/Zone 2
3:00 PM	Zone 2

schedule a meeting will provide this information. The energy usage information is a result of calculations from other agents, namely the thermal agents of the respective zones. The output of this agent is information back to the user who provided the input. The output is either 'Zone 1' or 'Zone 2', depending on the decision according to the behavior.

5.3.3 Communication Model

The communication model remains mostly the same, with just some added communication between the new agents. No change in the frequency or speed of the communication is required.

With this setup for the building zones, and the design for the new agent, the following results were observed:

The meetings at 9AM and 3PM were assigned to Zone 2 and one at 11AM to Zone 1, but the meeting at 1PM was never fixed at either zone, every trial would produce a either Zone 1 or 2, it was not consistent. Further analysis showed that while scheduling at 1PM, the agent stayed in the *no concensus* state and because of the probability assigned from there for no action, it would move to either *Zone 1* or *Zone 2* arbitrarily. Without further information, there was definite incentive for the agent to move to a particular state.

The reason that it required more informatoin was because at around 1PM, the energy use for maintining comfort at either location is calculated to be very close to each other, and since no more information that could be provided would break this tie, this problem of choosing the two zones arbitrarily was found.

At 11AM, total energy consumed was 4.2×10^5 BTU in the two zones combined with the meeting happening in Zone 1. Forcing the meeting to be in Zone 2, instead of 1, and calculating the total energy consumption resulted in 4.168×10^5 BTU, which could have been the result if the MAS was not used and the meeting zone selected randomly. This results in an energy savings of 0.8%.

5.4 Network Performance

One of the advantages of using a MAS is the ability to have distributed computing resources. This feature requires the use of networks to connect all the computing resources. For the experimentation performed in this thesis, in all the presented cases, two simple network scenarios were conducted:

1. Connect the agents to a large, campus-wide, network at Colorado School of Mines,
2. and connect the agents to a small, local network with only the agent communication as traffic on the network.

The lag in information between sending/receiving has been the main observation in this section. In a large network environment, the time for information relay was 0.15ms (average) , with some worst case scenarios the lag reached up to 6ms. For a vast majority of the time, the lag time stayed around the 0.15ms mark, but there were some events with very poor performance. In a smaller, local network, the time was very consistent and averaged 0.10ms. For a continuously network performance of observation test for 2 hours period there was no faults and no spikes of large . With these results in mind, it is possible to confirm that the proposed MAS control can be implemented in any building network (on its own subnet), and it seems that is feasible that the MAS can run on the same network, with the internet or any other data network.

CHAPTER 6
CONCLUSIONS

As computer aided technologies have been supporting a rapid growth in most of the applications in the world, it seems very plausible to further integrate such technologies in physical systems. The work in this thesis identified the problems that are present in performing such integration, and developed as well as analyzed one possible solution using Multi Agent Systems to overcome challenges presented by modern smart buildings.

The work conducted in this thesis was split into two major parts: 1) to develop a mathematical model for a building, allowing control methodologies to be applied and interactively simulated, and 2) to design and implement a multi-agent based control system in order to improve the computational performance of energy management control of buildings.

6.1 Thesis Contributions

The literature review of Chapter 1 gave a complete overview of the state of art for Cyber-Physical Systems, some of their applications as well as the problems associated with their use in real world situations. Background information of building systems, and the ideas involved in their mathematical representation were also provided. Eventually, an introduction to a relatively new computational paradigm for modeling and control i.e. Multi-Agent Systems, has been provided, and the use of such a system for use in buildings has been discussed, in order to support the contributions for this thesis.

In Chapter 2, the physical properties of an office building system were defined, and a general solution for mathematical modeling of the building was discussed. Using a general solution, a section of the building was selected and represented in state-space form using a lumped parameters model. In order to extend the model to any size or configuration of any building, a GUI based program was developed to provide state space matrices for different building configurations.

Chapter 3 dealt with designing a complete multi agent system. Being such a new paradigm in computer science and engineering, there is no proven methodology of design yet. Therefore, one of the contributions in this thesis is to analyze how MAS can be designed based on to object oriented programming. A review of the most prominent existing design methodologies was presented, followed by the discussion of an ad-hoc methodology that might be best for the current research. This showed to be very useful and the flexibility of this methodology was discussed in detail, using examples of a complete building energy management system.

In Chapter 4, the physical building model was implemented in Simulink and was successfully validated, by running an open loop test, as well as a closed loop PI control. The performance of the thermal system of the building, mainly temperature control and total energy usage, was analyzed and compared with available energy simulation tools. Following the physical model development, the multi agent system was developed using the methodology proposed in Chapter 3. Detailed discussions of each step in the design process were provided. Implementation details, including choice of software, classes used in the programming of the agents, and overall system configuration parameters were discussed.

Chapter 5 presented some case studies with different types of MAS control system implementation. The case studies ranged from simple reactive control, to more complex such as machine learning enabled concepts. The performance of the building model as well as the control system were analyzed and compared with each other, and also with other forms of control, and the flexibility of redesign procedure, as outlined in Chapter 3 was applied in order to incorporate a fuzzy behavior in the comfort agent, showing improved performance and supporting the proposed design procedure.

6.1.1 Conclusions

Based on findings supported by the previous chapters, it can be accepted that a MAS based solution for a Cyber Physical application definitely holds great potentials and promises. While it might not be applicable for every CPS case, it is definitely applicable for a smart building, and makes an improved decision making process possible. Using MAS allows the use of advanced computational methods such as artificial intelligence (involving fuzzy logic systems, and machine learning) and distributed computing. The case studies presented in this work were not emphasized in showing that a MAS is more energy efficient than any other control/computing paradigm. But it has supported that an overall paradigm shift of distributed control can be implemented. It is expected that when a complete building energy management system is implemented using MAS, it can also be a more energy efficient solution than standard control, particularly in long term range, and this full-fledged design can be subject of further research work.

6.2 Future Work

There are various avenues for future work that are available from this thesis. Some of them are listed below:

1. A more extensive model of a building can be developed incorporating zonal heating, and much deeper integration of the electrical system of the building. The model also needs a more robust model of the furnace of the building, compared to the time delay approach used in this work. Cooling of the building should also be considered as that is a large user of energy. The thermal agent can be expanded to include other thermal systems that might be present in a building such as a boiler or a chiller.

2. The electrical agent, which only acts as a demand curve supplying agent currently, can be expanded for an energy dispatch agent, which can communicate with the heating system. This would also require models of renewable energy sources present in the building to be modeled.

3. With further development of the agents, newer algorithms for learning, bargaining and negotiation can be explored for multi agent interaction.

One of the main aspects of a cyber-enabled physical system is to implement the cyber aspect to control or enhance the physical aspect. In this thesis, no real interaction between the cyber and physical systems were presented (no real world, hardware implementation was done). There is a lot of potential future work in the real world implementation of MAS for CPS. With computational power becoming faster and less expensive plus, robust communication infrastructure easily available, the future of cyber enabled physical systems looks very promising.

REFERENCES CITED

[1] Ragunathan Rajkumar, Insup Lee, John Stankovic, and Lui Sha. Cyber-Physical Systems : The Next Computing Revolution. In *Design Automation Conference*, page 1, Anaheim, California, 2010.

[2] Jan Kleissl and Yuvraj Agarwal. Cyber-Physical Energy Systems : Focus on Smart Buildings. In *Design Automation Conference*, pages 1–3, Anaheim, California, 2010.

[3] L. D. Danny Harvey. *Energy Efficiency and the Demand for Energy Services*. The Cromwell Press Group., London, 1 edition, 2010. ISBN 978-1-84407-912-4.

[4] Marija D. Ilic, Le Xie, Usman a. Khan, and José M. F. Moura. Modeling of Future CyberPhysical Energy Systems for Distributed Sensing and Control. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, 40(4):825–838, July 2010. ISSN 1083-4427. doi: 10.1109/TSMCA.2010.2048026. URL `http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5473027`.

[5] Charles R Robinson, Peter Mendham, and Tim Clarke. MACSimJX : A Tool for Enabling Agent Modelling with Simulink Using JADE. *Structure*, 4(3):1–7, 2010.

[6] Jose M Vidal. *Fundamentals of Multiagent Systems with NetLogo Examples*. 2010.

[7] M Pechoucek, D Sislak, D Pavlicek, and M Uller. Autonomous agents for air-traffic deconfliction. *Proceedings of the fifth inernational joint conference on Autonomous agents and multiagents of the ACM*, pages 1498–1505, 2006. URL `http://doi.acm.org/10.1145/1160633.1160925`.

[8] Stephen D J Mcarthur, Senior Member, Euan M Davidson, Victoria M Catterson, Aris L Dimeas, Student Member, Nikos D Hatziargyriou, and Ferdinanda Ponci. Multi-Agent Systems for Power Engineering Applications Part I : Concepts , Approaches , and Technical Challenges. *Power*, 22(4):1743–1752, 2007.

[9] Stephen D J Mcarthur, Senior Member, Euan M Davidson, Victoria M Catterson, Aris L Dimeas, Student Member, Nikos D Hatziargyriou, and Ferdinanda Ponci. Multi-Agent Systems for Power Engineering Applications Part II : Technologies , Standards , and Tools for Building Multi-agent Systems. *Power*, 22(4):1753–1759, 2007.

[10] Aris L Dimeas, Student Member, Nikos D Hatziargyriou, and Senior Member. Operation of a Multiagent System for Microgrid Control. *Power*, 20(3):1447–1455, 2005.

[11] Edward A Lee. Cyber-Physical Systems - Are Computing Foundations Adequate ? In *NSF Workshop On Cyber-Physical Systems*, Austin, Texas, 2006. UC Berkeley. URL `http://ptolemy.eecs.berkeley.edu/publications/papers/06/CPSPositionPaper/Lee_CPS_PositionPaper.pdf`.

[12] John A Stankovic, Insup Lee, and Aloysius Mok. Opportunities and Obligations for Physical Computing Systems. 38(11):23–31, 2005.

[13] Steve Heath. *Embedded Systems Design*. Newnes, Burlington, MA, 2 edition, 1997. ISBN 0750655461.

[14] Jean Michel Contet, Franck Gechter, Pablo Gruer, and Abder Koukam. Multiagent System Model for Vehicle Platooning with Merge and Split Capabilities. In *International Conference on Automation, Robotics and Applications*, Palmerston North, New Zealand, 2006.

[15] Herb Sutter and James Larus. Software and the Concurrency Revolution. *ACM Queue*, pages 54–62, September 2005.

[16] J. A. Stankovic. Misconceptions about real-time computing: a serious problem for next-generation systems. *Computer*, pages 10–19, October 1988.

[17] Robert Braun, Bill Hoff, Dinesh Mehta, Kevin Moore, Marcelo Godoy Simoes, and Siddharth Suryanarayanan. CPS: Medium: Cyber-Enabled Efficient Energy Management of Structures (CEEMS). Technical report, National Science Foundation, 2009.

[18] Peng Zhao. *A cyber physical system enabled efficient building energy management system through a multi-agent decision making control methodology*. PhD thesis, Colorado School of Mines, 2010.

[19] Peter C. Balash and Kenneth C. Kern. Natural Gas and Electricity Costs and Impacts on Industry. Technical report, DOE, NETL, 2008.

[20] DOE. Buildings Energy Data Book. Technical report, Department of Energy, 2009.

[21] Peng Zhao, Marcelo Godoy Simoes, and Siddharth Suryanarayanan. A conceptual scheme for cyber-physical systems based energy managment in building structures. In *IEEE/IAS International COnference on Industry Applications*, Houston, TX, 2010.

[22] B.L. Capehart, Wayne C Turner, and W.J. Kennedy. *Guide to Energy Management*. Fairmont Press, 6th edition, 2008.

[23] N Mendes, G H C Oliveira, H X Araújo, and L S Coelho. A MATLAB-BASED SIMULATION TOOL FOR BUILDING THERMAL PERFORMANCE ANALYSIS. pages 855–862, 2003.

[24] F Lorenz and G Masy. *Methode d'evaluation de l'economie d'energie apportee par l'intermittence de chauffage dans les batiments. Traitement par differences finies d'un model a deux constantes de temps*. Report no. gm820130-01, University de Liege, 1982.

[25] G.J. Levermore. *Buildling Energy Management Systems*. E & FN Spon, London, 1 edition, 1992.

[26] L. Klein, Jun-young Kwak, Geoffrey Kavulya, Farrokh Jazizadeh, Burcin Becerik-Gerber, and Milind Tambe Pradeep Varakantham. Coordinating occupant behavior for building energy and comforrt management using multi-agent systems. *Automation in Construction*, 2011.

[27] A.I. Dounis. Artificial Intelligence for energy conservation in buildings. *Advances in Building Energy Research*, 4:267–299, 2010.

[28] Sebastian A. Rodriguez. *From Analysis to Design of Holonic Multi-Agent Systems: A Framework, Methodological guidelines and applications*. PhD thesis, Universite de Technologie de Belfort-Montbeliard, 2010.

[29] Satchidananda Dehuri, Sung-Bae Cho, and Alok Kumar Jagadev. Honey Bee Behavior: A Multi-agent Approach for Multiple Campaigns Assignment Problem. *2008 International Conference on Information Technology*, pages 24–29, December 2008. doi: 10.1109/ICIT.2008.14. URL `http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4731292`.

[30] Nyree Lemmens, Steven De Jong, and Karl Tuyls. Bee behaviour in multi-agent systems : A bee foraging algorithm. In *ADAPTIVE AND LEARNING AGENTS AND MULTI-AGENT SYSTEMS (ALAMAS)*, 2007.

[31] V Hilaire, A Koukam, F Lauri, M Cossentino, and M Cirrincione. An Energy Mangement Solution based on Artificial Immune Systems and Agents. *Advances in Complex Systems*, pages 1–22, 2011.

[32] N. K. Jerne. Towards a network theory of the immune system. *Annales d'immunologie*, 125C(1-2):373–389, January 1974. ISSN 0300-4910. URL `http://view.ncbi.nlm.nih.gov/pubmed/4142565`.

[33] Jeremy Lagorse, Marcelo Godoy Simoes, and Abdellatif Maraoui. A Multiagent Fuzzy-Logic-Based Energy Management of Hybrid Systems. *IEEE Transactions on Industry Applications*, 45(6):2123–2129, 2009.

[34] P Maes. Agents that reduce Work and Information Overload. *Communications of the ACM*, 1994.

[35] M Godoy Simões and Saurav Bhattarai. Multi Agent Based Energy Management Control for Commercial Buildings. In *IEEE/IAS International Conference on Industry Applications*, Orlando, FL, 2011. ISBN 9781424495009.

[36] Saurav Bhattarai and Marcelo Godoy Simoes. Improving Energy Efficiency of Cyber Physical Systems using Multi Agent Based Control. In *IEEE/IAS International Conference on Industry Applications*, 2012.

[37] NREL. Weather Data Golden-NREL 724666 (TMY3). URL `http://apps1.eere.energy.gov/buildings/energyplus/weatherdata/4_north_and_central_america_wmo_region_4/1_usa/USA_CO_Golden-NREL.724666_TMY3.zip`.

[38] Division ofOffice of Real Property Innovative Workplaces. Real Property Performance Results. Technical Report December, United States GSA Office of Governmentwide Policy, 2011. URL `http://www.gsa.gov/graphics/ogp/spaceuse_2002_R2P52_0Z5RDZ-i34K-pR.pdf`.

[39] International Code Council. International Building Code. Technical report, International Code Council, 2012. URL `http://publicecodes.citation.com/icod/ibc/2012/icod_ibc_2012_12_sec008_par001.htm`.

[40] G G J Achterbosch, P P G De Jong, S F Van Der Meulen, and J Verberne. The Development of a Convenient Thermal Dynamic Building Model. *System*, 8:183 – 196, 1985.

[41] ENERGY STAR. ENERGY STAR Program Requirements for Computers. Technical report, 2009. URL http://www.energystar.gov/ia/partners/prod_development/revisions/downloads/computer/Version5.0_Computer_Spec.pdf.

[42] ENERGY STAR. ENERGY STAR Qualified Imaging Equipment Typical Electricity Consumption Test Procedure. Technical report, 2010. URL http://www.eu-energystar.org/downloads/specifications/20060422/TECTestProcedure4-21-06.pdf.

[43] KENMORE. Kenmore Refrigerators. URL http://www.kenmore.com/shc/s/p_10154_12604_04679304000P?vName=Kitchen&cName=Refrigerators+&+Freezers&sName=Top+freezer+Refrigerators&prdNo=1&blockNo=1&blockType=L1.

[44] ASHRAE. ASHRAE Fundamentals, 1997.

[45] Arthur Bell. *HVAC Equations, Data, and Rules of Thumb*. McGraw-Hill, 2nd edition, 2007.

[46] Scott DeLoach. Multiagent systems engineering: A methodology and language for designing agent systems. *Agent-Oriented Information Systems*, 1999.

[47] Pavlos Moraitis, Eleftheria Petraki, and Nikolaos I. Spanoudakis. Engineering jade agents with the gaia methodology. In *Agent Technologies, Infrastructures, Tools, and Applications for e-Services, volume 2592 of Lecture Notes in Computer Science*, pages 77–91. Springer-Verlag, 2003.

[48] J Buford.

[49] Michael Wooldridge, Nicholas R. Jennings, and David Kinny. A methodology for agent-oriented analysis and design, 1999.

[50] Michael Wooldridge, Nicholas R. Jennings, and David Kinny. The gaia methodology for agent-oriented analysis and design. *Journal of Autonomous Agents and Multi-Agent Systems*, 3:285–312, 2000.

[51] Michael Bratman. *Intention, Plans, and Practical Reason*. Cambridge University Press, Cambridge, UK, 1 edition, 1999.

[52] David Kinny and Michael Georgeff. Modelling and design of multi-agent systems. In *INTELLIGENT AGENTS III (LNAI)*, pages 1–20. Springer-Verlag, 1997.

[53] Benjamin Blunier Abdellatif Miraoui Abderrafa Koukam. Robin Roche, Fabrice Lauri. *Multi-Agent Technology for Power System Control*. 2012.

[54] Martin J. Osborne and Ariel Rubinstein. *A Course in Game Theory*. MIT, 1994.

[55] T.M Mitchell. *Machine Learning*. McGraw-Hill, 1997.

[56] D. H Wolpert and W.G Macready. No free lunch theorems for search. Technical report, Santa Fe Institute, 1995.

[57] Peter Dayan Christopher Watkins. Q-learning. 1992.

[58] DOE2. eQUEST. URL `www.doe2.com/equest/`.

[59] Northwestern University. NetLogo User Manual, 2012. URL `http://ccl.northwestern.edu/netlogo/docs/`.

[60] Charles R Robinson. *Decentralised Data Fusion Using Agents*. Phd thesis, The University of York, 2008.

[61] Giovanni Caire. Jade Programming for beginners. Technical report, Telecom Italia, 2009.

[62] Pablo Cingolani. Open Source Fuzzy Logic library and FCL language implementation. URL `http://jfuzzylogic.sourceforge.net/html/about.html`.

APPENDIX A - THERMAL RESISTANCE AND CAPACITANCE CALCULATIONS

A.1 Thermal Resistance

Thermal resistances of common materials are provided in the ASHRAE handbook [44], but mathematical calculations need to be done to calculate values for different thermal characteristics such as thermal resistance and capacitance. Calculation of thermal resistance for a wall consisting of different materials is presented next.

The wall shown in Figure A.1 is a typical external wall of a commercial building, made with insulated concrete blocks. The various parts of the wall are labeled in the figure. Since there are different materials with different thermal properties used in the wall construction, thermal property tables will not have all the values for the entire wall. For this calculation, some assumptions are made, which can be changed according to requirements, as they only are to do with the dimensions of the wall. Assumptions:

- web thickness of block = 25mm
- face shell thickness = 30mm
- block dimensions194mm × 194mm × 395mm

The equation for calculating overall thermal resistance is:

$$R_{T(av)} = R_i + R_f + \left(\frac{a_w}{R_w} + fraca_c R_c\right)^{-1} + R_o \quad \text{(A.1)}$$

where $R_{T(av)}$ = overall thermal resistance

R_i = thermal resistance of inside still air

R_o = thermal resistance of outside air

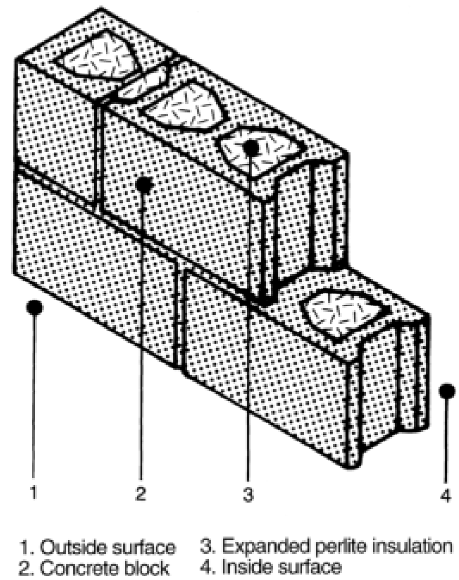


Figure A.1: Insulated Concrete Block for External Wall Construction

R_f = thermal resistance of face shells

R_c = thermal resistance of cores between face shells

R_w = thermal resistance of webs between face shells

a_w = fraction of total area represented by webs of blocks

a_c = fraction of total area represented by cores of blocks

All the thermal resistances mentioned above can be found in Table 4 in Section 25.5 of [44] . With those values, the calculation becomes simply the following for thermal resistance, and its reciprocal (the U-factor).

$$R_{T(av)} = 0.12 + 0.045 + \frac{(0.091 \times 2.60)}{(0.81 \times 0.91) + (0.19 \times 2.60)} + 0.03 = 0.612\mathrm{K}\dot{\mathrm{m}}^2/\mathrm{W} \quad (A.2)$$

$$U_{av} = \frac{1}{0.612} = 1.63W/(K\dot{m}^2) \quad (A.3)$$

APPENDIX B - AGENT DETAILS

B.1 Thermal agent MDP Parameters

Table B.1: Transition probabilities for thermal agent

s	a	s'	T(s,a,s')
1	1	2	0.3
1	1	4	0.7
1	2	3	0.3
1	2	4	0.7
1	3	1	1
2	2	3	0.1
2	2	4	0.4
2	2	2	0.3
2	2	1	0.2
2	3	1	0.15
2	3	2	0.7
2	3	4	0.2
2	3	3	0.05
3	1	2	0.1
3	1	4	0.5
3	1	3	0.3
3	1	1	0.1
3	3	1	0.1
3	3	4	0.2
3	3	3	0.7
4	1	1	0.5
4	1	2	0.5
4	2	1	0.5
4	2	3	0.5
4	3	1	0.1
4	3	2	0.4
4	3	3	0.1
4	3	4	0.4

APPENDIX C - LEARNING ALGORITHMS

C.1 Q Learning

The *Q learning* algorithm developed by Watkins and Dayanc̃iteWatkins1992 is a popular learning algorithm for use in an agent based system. It is used to find the optimal policy of a agent, similar to the value iteration algorithm. The concept of the algorithms are very similar except for the major difference in learning. The value iteration algorithm has no learning abilities, but *Q learning* introduces two new concepts: the learning rate, and the exploration rate. The learning rate is a measure of how much emphasis is put on new rewards, compared to the learned values, whereas the exploration rate is required to make sure the agent considers other actions, and the rewards associated, without converging to a solution too quickly. These values are both normalized to be in between 0 and 1. The problem associated with the Q learning algorithm is the same as what was found with the value iteration algorithm: only one agent is considered at a time. In the Q learning algorithm, only one agent can learn at a time, and collective learning is not possible.

Q-Learning
$\forall_s \forall_a Q(s,a) \leftarrow 0; \lambda \leftarrow 1; \epsilon \leftarrow 1$
$s \leftarrow$current state

if RAND() $< \epsilon$ **then**
    $a \leftarrow randomaction$
else
    $a \leftarrow \arg\, max_a Q(s,a)$
end if
Take action a

11: Receive reward r

12: $s' \leftarrow$ current state $Q(s,a) \leftarrow \lambda(r + \gamma \max_{a'} Q(s',a')) + (1-\lambda)Q(s,a)$ $\lambda \leftarrow .99\lambda$

13: $\epsilon \leftarrow .98\epsilon$

14: goto 2

C.2 NASH-Q Learning

The NASH-Q Learning algorithm is presented below:

1: $t \leftarrow 0$

2: $s \leftarrow$ current state $\forall_{s \in S} \forall_{j \leftarrow 1,\cdots,n} \forall_{a_j \in A_j} Q_j^t(s, a_1, \cdots, a_n) \leftarrow 0$

3: Choose action a_i^t

4: $s \leftarrow s'$

5: Observe $r_1^t, \cdots, r_n^t; a_1^t, \cdots, a_n^t; s'$

6: **for** $j \leftarrow 1, \cdots, n$ **do**

7: $\quad Q_j^{t+1}(s, a_1, \cdots, a_n) \leftarrow (1-\lambda^t) Q_j^t(s, a_1, \cdots, a_n) + \lambda^t(r_j^t + \gamma \mathrm{Nash} Q_j^t(s'))$

8: **end for**

9: $t \leftarrow t+1$

10: goto 4

In the above algorithm, $NashQ_j^t(s') = Q_j^t(s', \pi_1(s') \cdots \pi_n(s'))$ and $\pi_1(s') \cdots \pi_n(s')$ are the Nash equilibrium points calculated from the Q values from the Q algorithm mentioned earlier. The Nash equilibrium point for agents is a set of policies $\pi_1(s') \cdots \pi_n(s')$ such that for all $s \in S$ and $i = 1, \cdots, n$,

$$\forall_{\pi_i} \in \textstyle\prod_i V_i(s, \pi_1, \cdots, \pi_n) \geq v_i(s, \pi_1, \cdots, \pi_i, \cdots, \pi_n)$$

where $v_i(s, \pi_1, \cdots, \pi_i, \cdots, \pi_n)$ is the total rewards a particular agent can expect to receive.

	Construction Name	Specification Method	Absorptance	Roughness	U Value (Btu/h-ft2-°F)	Wall Parameters	Layers
1	EWall Construction	Layers Input	0.600	4	0.080	- undefined -	EWall Cons Layers
2	Roof Construction	Layers Input	0.600	1	0.043	- undefined -	Roof Cons Layers
3	Ceilg Construction	Layers Input	0.700	3	0.514	- undefined -	Ceilg Cons Layers
4	IWall Construction	U-Value Input	0.700	3	2.700	- undefined -	n/a
5	IFlr Construction	Layers Input	0.700	3	0.813	- undefined -	IFlr Cons Layers
6	UFCons (G.NE1.U2)	Layers Input	0.700	3	0.124	- undefined -	UFLyrs (G.NE1.U2)
7	UFCons (G.SW2.U3)	Layers Input	0.700	3	0.114	- undefined -	UFLyrs (G.SW2.U3)
8	Sgl Lyr Unins Mtl Door	U-Value Input	0.700	3	2.080	- undefined -	n/a

Figure D.1: Construction Details

	Layers Name	Inside Film Resistance (R-val) (h-ft2-°F/Btu)	Material 1	Thickness 1 (ft)	Material 2	Thickness 2 (ft)	Material 3	Thickness 3 (ft)	Material 4
1	EWall Cons Layers	0.680	Plywd 5/8in (PW04)	0.052	Insul Bd 3/4in (IN6	0.063	EWall Cons Mat 2 (:	n/a	GypBd 1/2in (GP01
2	Roof Cons Layers	0.680	Blt-Up Roof 3/8in (E	0.031	Polyurethane 3in (I	0.250	Plywd 5/8in (PW04)	0.052	Roof Cons Mat 4 (2
3	Ceilg Cons Layers	0.680	AcousTile 1/2in (AC	0.042		n/a		n/a	
4	IFlr Cons Layers	0.680	Conc HW 140lb 6in	0.500	Linoleum Tile (LT01	n/a		n/a	
5	UFLyrs (G.NE1.U2)	0.680	UFMat (G.NE1.U2.M	n/a	Light Soil, Damp 12	1.000	Conc HW 140lb 6in	0.500	Linoleum Tile (LT01
6	UFLyrs (G.SW2.U3)	0.680	UFMat (G.SW2.U3.M	n/a	Light Soil, Damp 12	1.000	Conc HW 140lb 6in	0.500	Linoleum Tile (LT01

Figure D.2: Details of Layers of Construction Material

	Material Name	Specification Method	Thickness (ft)	Conductivity (Btu/h-ft-°F)	Density (lb/ft3)	Specific Heat (Btu/lb-°F)	Resistance (h-ft2-°F/Btu)
1	EWall Cons Mat 2 (8.6)	Resistance	n/a	n/a	n/a	n/a	8.600
2	Roof Cons Mat 4 (2.8)	Resistance	n/a	n/a	n/a	n/a	2.800
3	UFMat (G.NE1.U2.M1)	Resistance	n/a	n/a	n/a	n/a	4.857
4	UFMat (G.SW2.U3.M1)	Resistance	n/a	n/a	n/a	n/a	5.523
5	Plywd 5/8in (PW04)	Properties	0.052	0.0667	34.00	0.290	n/a
6	Insul Bd 3/4in (IN62)	Properties	0.063	0.0316	18.00	0.310	n/a
7	GypBd 1/2in (GP01)	Properties	0.042	0.0926	50.00	0.200	n/a
8	Blt-Up Roof 3/8in (BR01)	Properties	0.031	0.0939	70.00	0.350	n/a
9	Polyurethane 3in (IN46)	Properties	0.250	0.0133	1.50	0.380	n/a
10	AcousTile 1/2in (AC02)	Properties	0.042	0.0330	18.00	0.320	n/a
11	Conc HW 140lb 6in (HF-C	Properties	0.500	1.0000	140.00	0.200	n/a
12	Linoleum Tile (LT01)	Resistance	n/a	n/a	n/a	n/a	0.050
13	Light Soil, Damp 12in	Properties	1.000	0.5000	100.00	0.250	n/a

Figure D.3: Thermal Properties of Materials