ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

# Προσομοίωση Έξυπνων Σπιτιών με Χρήση Τεχνολογίας Πρακτόρων

# Smart Homes Simulation using Agents Technology

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Της

**ΠΕΤΡΟΜΕΛΙΔΟΥ ΚΥΡΙΑΚΗ**

Βόλος, ΜΑΡΤΙΟΣ 2017

ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ
ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ
ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

# Προσομοίωση Έξυπνων Σπιτιών με Χρήση Τεχνολογίας Πρακτόρων

# Smart Homes Simulation using Agents Technology

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Της

**ΠΕΤΡΟΜΕΛΙΔΟΥ ΚΥΡΙΑΚΗ**

**Επιβλέποντες :**

**Τσομπανοπούλου Παναγιώτα**
Αναπληρώτρια Καθηγήτρια Π.Θ.

**Μποζάνης Παναγιώτης**
Καθηγητής Π.Θ.

Εγκρίθηκε από την διμελή εξεταστική επιτροπή την 1 Μαρτίου 2017

| | |
|---|---|
| *(Υπογραφή)* | *(Υπογραφή)* |
| …...................................... | .................................. |
| Αναπληρώτρια Καθηγήτρια Π.Θ. | Καθηγητής Π.Θ. |

*(Υπογραφή)*

...................................

**Πετρομελίδου Κυριακή**

Διπλωματούχος Μηχανικός Ηλεκτρονικών Υπολογιστών, Τηλεπικοινωνιών και Δικτύων του Τμήματος Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών, Πανεπιστημίου Θεσσαλίας

Dedicated to my sister.

## Acknowledgements

At the completion of this thesis and reaching at the end of my undergraduate studies, I would like to thank all the people who helped me to get here.

First of all, I would like to thank my supervisors Panagiota Tsompanopoulou and Panagiotis Bozanis for their support and the confidence they showed in me during my thesis elaboration and for the opportunity they gave me to be part of a project like this. In addition, I am grateful to Athanasios Fevgas and Dimitris Zimeris for their help and valuable advices through this project.

Finally, I would like to thank my family for their implicit support and invaluable assistance through the course of my studies and my friends who were beside me all these years.

# Περίληψη

Ο σημερινός τρόπος ζωής και οι παρούσες τεχνολογίες έχουν εμπνεύσει νέους τρόπους για την διαχείριση και την μείωση της κατανάλωσης ενέργειας. Διάφορες τεχνολογίες αυτοματισμού επιτρέπουν στον χρήστη να ελέγχει και να προγραμματίζει συνηθισμένες εργασίες στο σπίτι, χωρίς ωστόσο να είναι απαραίτητη πάντα η δικιά του παρέμβαση. Το έξυπνο σπίτι είναι ένα μοντέρνο σπίτι που επιτρέπει στους κατοίκους να έχουν άνεση υψηλού επιπέδου με την αποτελεσματική χρήση της ηλεκτρικής ενέργειας.

Ο στόχος της παρούσας διατριβής είναι να δείξει ότι με τη χρήση πολυπρακτορικών συστημάτων που σχετίζονται με τις ηλεκτρονικές συσκευές, είναι δυνατόν να επιτευχθεί η πιο αποτελεσματική χρήση της ενέργειας και να ελαχιστοποιηθεί το κόστος των λογαριασμών ηλεκτρικού ρεύματος. Παράλληλα, περιγράφουμε τα μοντέλα συμπεριφοράς των πρακτόρων που έχουν αναπτυχθεί με βάση την πεποίθηση, την επιθυμία και την πρόθεση τους, καθώς και ορισμένα από τα λειτουργικά χαρακτηριστικά τους.

Το Internet of Things (IoT) είναι ένα πανταχού παρόν δίκτυο που βασίζεται στο Internet. Είναι ωστόσο μια τεχνολογία η οποία παρουσιάζει χαρακτηριστικά που ενέχουν σοβαρούς κινδύνους για την ασφάλεια και την προστασία της ιδιωτικής ζωής. Στην παρούσα εργασία αναφέρουμε μερικές από τις απειλές που μπορεί να εμφανιστούν στα στρώματα της αρχιτεκτονικής ασφάλειας του Internet of Things.

Η ανταπόκριση στη ζήτηση (Demand Response) είναι ένα βασικό εργαλείο που χρησιμοποιείται για την ανάπτυξη σύγχρονων συστημάτων ηλεκτρικής ενέργειας και αγορών ηλεκτρικής ενέργειας. Υπάρχουν δύο τύποι ανταπόκρισης στη ζήτηση, Price-Directed and Incentive-Based demand response. Κάθε μία από αυτές τις κατηγορίες αποτελείται από διάφορα προγράμματα και μεθόδους. Επιπλέον, αναφέρουμε μερικά από τα οφέλη που προκύπτουν τόσο για τους συμμετέχοντες στην αγορά και όσο και για την βελτίωση της αποδοτικότητας των αγορών.

Σκοπός της παρούσας διπλωματικής εργασίας ήταν η ανάπτυξη μιας εφαρμογής που θα δώσει στον χρήστη τη δυνατότητα να προσομοιώσει το περιβάλλον του σπιτιού του, ρυθμίζοντας τον αριθμό των συσκευών που διαθέτει. Ανάλογα με την κατανάλωση κάθε συσκευής και τα σενάρια λειτουργίας, ο χρήστης είναι σε θέση να υπολογίσει το συνολικό κόστος. Προσομοιώσεις όπως αυτές πραγματοποιούνται στο NetLogo, μία γλώσσα προγραμματισμού που στηρίζεται στους πράκτορες και αποτελεί ένα ολοκληρωμένο περιβάλλον προσομοίωσης.

# Abstract

Current lifestyles and technologies have inspired new ways to manage and reduce energy consumption. Several home automation technologies allow users to control and schedule common home tasks, often without human intervention. Smart Home is a modern home that allows residents to have high-level comfort with effective use of electricity.

The objective of this thesis is to show that by using multi-agents systems (MAS) related to appliances, it is possible to provide the most efficient energy usage and minimize the cost of electricity bills. We also describe the belief, desire and intention (BDI) agent behavior models that have been developed, as well as some of the functional characteristics of the agents.

The Internet of Things (IoT) is a ubiquitous Internet-based network. That however, exhibits characteristics that pose considerable risks in security and privacy. In this thesis we mention some of the threats that may appear in the layers of the security architecture of IoT.

Demand response is a basic tool used to develop modern power systems and electricity markets. There are two types of demand response, Price-Directed and Incentive-Based demand response. Each of these categories is composed of several programs and methods. Moreover, we mention some of the benefits that demand response creates for market participants and for the efficiency of electric markets.

The aim of this thesis was to develop an application that will give the user the ability to simulate his home environment by selecting the number of the devices he has. Depending on the consumption of each device and the function scenarios the user is able to compute the total cost. Simulations like that are performed using NetLogo, an agent-based programming language and integrated modeling environment where each device is an agent.

## Contents

# List of Figures

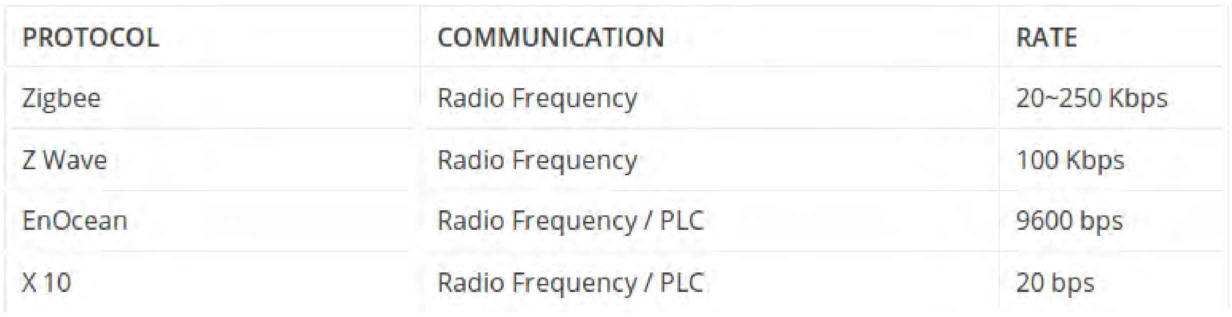
# 1. Smart Homes, Multi-Agent Systems, Internet of Things and Demand Response

## 1.1 Smart Homes

### 1.1.1 Definition

A smart home describes (Home automation or Smart Home also known as domotics or domotica) a residence which is equipped with the proper technology for the management and analysis of information that receives from the environment. The main purpose is to understand the demands of its residents and respond to them aiming at improving their lives concerning their convenience, security, safety even their entertainment. All the devices are controlled and connected with each other, i.e. the electronic devices coexist and communicate with each other. Regarding security issues; from the locking of the door, systems and surveillance video for the detection of any intruder to the washing machine and the coffee machine, all are connected wiredly, wirelessly as well as with other communication technologies. Smart houses and home automations are the evolution of building automation, heating, ventilation and air conditioning (HVAC) is the technology of indoor and vehicular environmental comfort, the security and fire systems are integrated and can be controlled by using a building management system (BMS) also known as a building automation system (BAS), which is a computer-based control system installed in buildings for the surveillance and control of the mechanical and electronic equipment. The main purpose of their designing was to combine the comfort of everyday life in the house with the idea of reducing the consumption of energy, time and money.

The interconnectivity of the home automation devices can be based on various protocols. Some of the most well-known protocols are the ZigBee, Z-Wave, EnOcean, X10. Each one of these protocols has specific characteristics, i.e. specific media, data transmission & reception rate and application (Figure 1) [1].



| PROTOCOL | COMMUNICATION | RATE |
|---|---|---|
| Zigbee | Radio Frequency | 20~250 Kbps |
| Z Wave | Radio Frequency | 100 Kbps |
| EnOcean | Radio Frequency / PLC | 9600 bps |
| X 10 | Radio Frequency / PLC | 20 bps |

**Figure 1** Characteristics of Protocols [1]

At smart houses, every connected unit is controlled by a central point with the use of a mobile phone, tablet making use of a specific application or a control panel. Some examples that make use of this technology in a house are the following [1]:

- heating, ventilation, air-conditioning
- lighting systems
- security systems
- intrusion detection and alarm systems
- gas detectors
- automations for the amenity of elderly people or people with special needs
- sound and image systems

**Figure 2** Example of connected units in a smart home

### 1.1.2 History and Development of Smart Homes

The first smart houses were ideas and not actual structures. For decades, science fiction has explored the idea of home automation. Smart homes, though, have been on surface for only a little period of time. At the following timetable, some inventions that led to smart homes of today, are listed [2] [4].

**1901 - 1920 – The invention of appliances –** An amazing achievement at the beginning of the 20th century is the invention of appliances, even though their initial form has nothing to do with what is called today 'smart'. In 1901 the first vacuum cleaner made its appearance which after a few years, in 1907, evolved into a more practical model. During the following two decades, the first refrigerators appeared, clothe dryers, washing machines, irons, toasters and many more creating a new era for the household.

**1950 – There Will Come Soft Rains –** The first appearance of a fully automated home in a sci-fi story by Ray Bradbury. In his story, he describes how he imagines one such home where the computers have the ability to take care of every single need of the daily life of a family even when they have already disappeared [3].

**1966 - 1967 – ECHO IV and the Kitchen Computer –** ECHO IV was the first smart device which had the potential to calculate the shopping list, check the temperature of the house as well as turn on and off the devices. The kitchen computer, appeared one year later and could save various recipes. Yet, none of the two devices managed to gain the public interest and get a foothold in the commerce as very few models had been sold.

**1970 – X10 –** X10 was the first communication protocol for the electric devices. This novelty played an important role to the history of smart houses by signaling the beginning of the house automation. Today X10, Z-Wave, ZigBee and others are still considered as the most usual protocols of home automation.

**1991 – Gerontechnology –** Gerontechnology combines gerontology and technology with the aim of making the elderly's life easier. For this reason a lot of research has been carried out in this field at Eidhoven University of Technology, in France and in Belgium as well as in the Netherlands. Through these researches the sense of home automation arose.

**1998 - Early 2000s – Smart Homes –** The smart houses and home automations began to gain grounds in popularity at the beginning of 2000. Smart houses suddenly became a more affordable option for the consumers. Domestic technologies, home networking and other gadgets began to appear on the shelves of the stores.

**Today – Smart Homes –** Today, smart homes focus on safety and how to secure a life more environmentally-friendly. They are sustainable and they try to ensure that unnecessary expenditure will occur. In the safety sector, they have the potential to notify the residents for possible intruders either they are in the house or out of it. Contemporary trends at home automation include distant control via a mobile phone or tablet, automate lights, automate regulation of the thermostat, control and programming of devices, notifications via mobile phones/emails/texts and many more.

During the last years, a development has been observed in the field of smart homes equally towards the aim of existence of such a system and its interaction with the user. It is necessary the smart home system offer to the user the potential to have the absolute control of every connected device, to program scenarios exactly as he wishes, to receive information in real time and get notified by automatic alarms in case an activity which takes place deviates from the programmed scenarios. Smart houses have to confront any difficulties that may arise due to the high expenses of such a technology and the installation of that as well as the anxiety for the assurance of the privacy so that they can become established in the commerce [6].

Home automations and control systems help the development of the environment in our house. New data such as saving energy and money, inhabit caring, amelioration of security start to be considered more important than entertainment and luxury. The designing of such systems has to be focused on the experiences and demands of the user.

The user’s interaction with the systems consists of various mechanisms which are based on natural impulses such as the language and the interfaces, and at the same time the consumers have the potential to apply full control to any device which is connected to the system. Therefore, a fully functional environment is created, which identifies everyday activities that the residents of the house do and with the proper actions it offers automatic assistance that is demanded. Technology today has reached such a level that the users are fully familiar with the direct results after each use either of the computer or of other portable devices such mobile phones and tablets. Concerning the smart homes, such eminent reactions demand systems with potentials of external communication 24/7, something which increases the complexity of the system as new demands appear constantly. Moreover, the current mobile and applications allow the users to determine their own settings (username, mechanism of ID control, way of communication, duty list, etc.) except for the typical configuration. Thus, the preferences of each user play a significant role to the designing of such an interactive system so that anyone could determine their own settings. At the moment, mobile and web applications prevail over the programs via a computer in the interaction of user and system [6].

The development of smart houses is accompanied with new suggestions in computer networking, embedded systems, wireless sensor networks increasing constantly the expectations for what a smart home system can offer [6].

- Identification of a number of vocal orders.
- Surveillance in real time through the use of systems for what is happening in the house.
- Identification of information that was collected and the execution of the corresponding demanding action.
- Surveillance of the home environment with sensors for the regulation of the temperature, humidity, lighting etc.
- Surveillance of the user's actions so that the system to be able to avert possible accidents or even to call help in such cases.
- Ability to give distant orders to the system.
- Certification of the identity of the user in order to avert the interaction with non-authorized users with the system.
- Programming of various actions by the user and execution in specific period of time.
- Notification of the user in case an unexpected incident occurs such as fire, air leak, etc.

By observing the residents' habits, the system will be able little by little to foresee if a device will be used in the future and to what extent. By this way, later the system will be in position of reducing the consumption of electric energy or water and as a result the cost will be reduced. Various algorithms and machine learning techniques process such data and demonstrate various scenarios on the way to use these devices with the aim of minimum consumption so that the user could opt for the one that regards as the most suitable to his needs and preferences [7].

## 1.2 Multi-Agent Systems

A smart house consists of a variety of embedded sensors/actuators, distributed/mobile computing units and energy harvesting devices. Yet, the existence of such an amount of devices has as a result the creation of a complex system with conflicting targets. One solution that is worth mentioning is the use of multi-agent technology for the designing and management of such systems. An agent is an independent hardware/software co-operation unit which is able to apprehend the existing circumstances every time and execute a series of predetermined actions with a specific target on behalf of another entity either human or computer programmed. A crucial advantage of agents is the potential of incorporating intelligence in various level. The agents can have different communication abilities and control according to their responsibilities.

The individual behavior of each agent is determined by their own functions, i.e. sensing, action, decision and database, as well as the role which has to play every time. A belief, desire and intention (BDI) based agent behavior generation method has developed. The user’s goal, tasks and contextual information can be converted into a set of beliefs, desire and intention models. The individual agent behavior is then systematically developed and based on BDI models. The belief-desire-intention software model, is a software model developed for programming intelligent agents. It provides a mechanism for separating the activity of selecting a plan (from a plan library or an external planner application) from the execution of currently active plans. Therefore, BDI agents are able to balance the time spent on deliberating about plans (choosing what to do) and executing those plans (doing it). Beliefs represent the information that each agent receives according to their own perception for the existing situation, in correlation with itself, the rest of the agents as well as the environment. Such information for instance could be its own situation and availability as the agents that cooperates with, i.e. the place, the time, the temperature and the reactions and preferences of the user. Yet, what an agent believes does not have to be necessarily true. Desires not only represent the information that have to do with which target has to be achieved but also with what the cost will be for this to be fulfilled. They depend on the beliefs of each agent as well as the user’s inputs.

Intentions represent which action plans will be realized in order that desires could be achieved that the agent has been committed. The plans are a series of actions that the agent intends to realize in order to succeed in one or more of its intentions. One plan can include other plans or cover partially some others [7] [8] [9].

The behavior and interaction of the agents is controlled also by communication protocols, cooperative systems and management policies of the available resources. Through the communicative protocols, exchange of various types of information, service requests and reports are occurring as well as the update of the status in which every agent is. The agents have to define clearly what is the message they want to transfer, to wit which function has to be executed, find out the agents they will have to cooperate with and find the proper mechanisms in the system so that the function can be executed with success. The successful cooperation among the agents is ensured through the systems and policies which are in effect so as to avoid possible conflicts in their programming, their priority and management of available resources [8] [9].

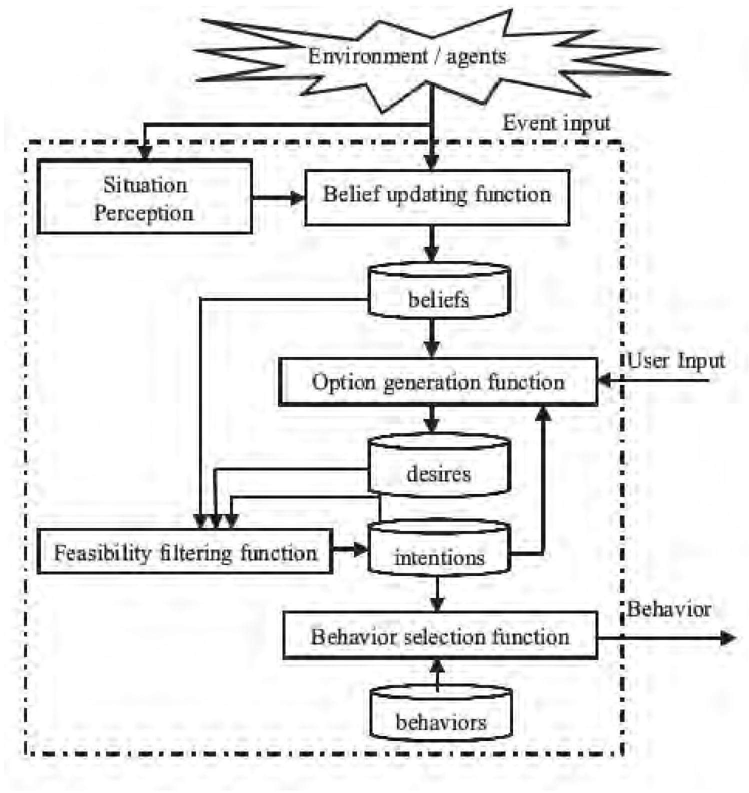


**Figure 3** BDI model individual agent behavior [7]

The functional characteristics of the agents are the following [8] [9] [10]:

- Autonomy: An agent is capable of taking initiatives and act without constant observation and eminent (human or other) interference. It has to be able to control to a certain extent its actions and its internal condition based on its beliefs, desires and intentions.
- Interactivity: An agent can interact with the environment and with other entities.
    - Reactivity: The potential to conceive the changes in the environment and react to them, if necessary
    - Pro-activeness: The ability to take initiatives and act, trying to satisfy the purpose of its function.
- Adaptability: The ability of the agent to perceive other agents around it. Various adaptation methods allow the agents to change their internal conditions according to their experience and environment.
- Goal-Oriented: An agent can choose how and when the existing demands will be fulfilled.
- Cooperativity: The potential of the agent to cooperate with other agents for the achievement of a mutual goal.
- Competitiveness: In cases of agents' dominance, an agent can compete with another agent or deny the supply of services to a rival agent.
- Temporal Continuity: An agent functions continuously, so it has to be well designed.
- Character: The potential of showing human characteristics, such as personality and emotional condition.
- Mobility: An agent can move from one environment to another without terminating its function.
- Flexibility: The functions of an agent are not predetermined. The agent can choose which functions it will execute and in which order according to the existing conditions.

- Adaptability: The agent can adapt to the possible changes of environment or to the preferences of the user based on previous experiences.
- Learning: An agent has to be able to learn from its reactions and interaction with the environment. The more effective the learning procedure, the smarter the agent becomes.

Ambient Intelligence (AmI) refers to electronic environments that are sensitive and responsive to the presence of people. It deals with a new world where computing devices are spread everywhere, allowing the human being to interact in physical world environments in an intelligent and unobtrusive way. These environments should be aware of the needs of people, customizing requirements and forecasting behaviors. AmI aims at the increase of comfort of the environment in which people live or work making use of available resources and services. Flexible and adaptive discovery of embedded systems must be leveraged for that. Therefore, mobile context-aware middlewares and agent systems are often seen as pivotal elements of AmI. More specifically, Multi-Agent Systems (MAS) are used in modeling scenarios which allow competition and communication. Agent-Oriented Software Engineering (AOSE) is a new software engineering set of including theories, research methods, postulates, for the application of optimal practices for the development of complex Multi-Agent Systems. According to AOSE, agents can meaningfully represent and simulate entities (e.g. devices), contexts or people emphasizing social capabilities (communication, cooperation, conflict resolution and negotiation) [5][6].

The limited availability of energy resources has led to the need to develop effectively measures for energy saving. The existence of complex and huge problems, which are internally spread or are solved only with the cooperation of spread elements, have proven that the use of a single agent is not enough. For this reason, multi-agent systems have been developed. A multi-agent system is an environment where a lot of agents can take action, either with one mutual, collaborative target working all together or with individual-rival ones, trying to dominate on the rest of agents. Multi-agent systems outclass the systems with one agent because of their spread nature; they are flexible, expandable and change easily in contrast to the architectures of one agent. An agent cannot gain extensive and specialized knowledge. A multi-agent system, because of its plenty and spread operations, can access more sources of knowledge at the same time [8] [9].

On these grounds the development of various multi-agent systems was based in which the agents communicate with each other and the residents in a house about how the minimum energy consumption is to be achieved. Multi-agent systems have been considered to provide sustainable energy for smart grid management. Multi-Agent Systems are very helpful in Home and Building Automation (HBA) due to their ubiquitous and distributed nature. For example, in Home Automation systems as composed by a collection of domotic objects and domotic agents, every agent is responsible for the solution of electrical power allocation problems, declares itself, detects other agents and interacts with them and handle possible conflicts among the rival agents. The agent-based solutions either require direct user intervention or support only elementary agent behaviors and basic interactions, lacking advanced service/resource characterization, discovery and composition [7].

The system of a house consists of sources which produce energy and electrical forces (appliances) which consume energy. This action comes from either national networks transfer/distribute of electric energy or alternative sources of energy (solar panels, wind generators, etc.) Yet, in order to predict the energy needs of every home, control and management systems have to be developed capable of handle the multiple sources of production. At the same time, a way to reduce the energy consumption of existing buildings has to be found. The need for structural self-adaptation leads us to multi-agent systems which allow the exchange of information between modules via message exchanges. The home automations have to be extensible so that devices to be added and removed at any time without doubting the function of the system. Progress in the field of Multi-Agent Systems, designed to solve spatially distributed and open problems, is likely to lead to a ”Multi-Agent Home Automation System” made up of agents embedded in different sources and loads all working together in order to solve the problem of energy consumption control in the home [11].

## 1.3 Internet Of Things

### 1.3.1 Definition

One of the factors that determine the economy of every country is the service consumption and appears at the price per service and cost. New suggestions are demonstrated continuously which concern the evolution of systems that are used in smart homes with the aim of saving energy. The optimization of energy resources is a way to confront such challenges. The energy consumption of every home is a small example of consumption of a neighborhood, a city and afterwards a whole country. A lot of researches have been conducted concerning the way the increasing demand of energy could be balanced with the most productive consumption and that is the reason why improvements to the efficiency of smart homes are realized continuously.

The sense of Internet of Things (IoT) is something that has to be mentioned as through this the control of the connected devices to the system of a smart home is accomplished. The term IoT was coined by Kevin Ashton a technology pioneer to describe the network connecting objects in the physical world to the Internet. He himself said [13]: *“In the twentieth century, computers were brains without senses—they only knew what we told them. In the twenty-first century, because of the Internet of Things, computers can sense things for themselves.”*

**Figure 4** Various interconnected devices can offer smart home services

It was a technological revolution that combined Internet characteristics and telecommunications network having simultaneously though its own features that made it differentiate. IoT has self-configuring capabilities based on standard communication protocols in which physical and virtual things have identity and physical characteristics and make use of intelligent interfaces in order to integrate in the information network. Actually, it is a universal network of connected objects which are addressable and connect people, data and things. This leads to a huge amount of devices which have the potential to gather and exchange information using embedded sensors. Contrary to IP addresses that can create a specific amount of about 4.3billion, IoT will allow the connection of approximately 25billion devices, from Smart phones and Smart watches to Self -Flying Pilotless planes, by 2020, all around the whole planet through any possible way of communication [12]. The Internet of Things is an immense network of connected “things”. The connection between people-people, people-things and things-things is allowed, with any-time, any-place and any-one connected technology [15] [17].

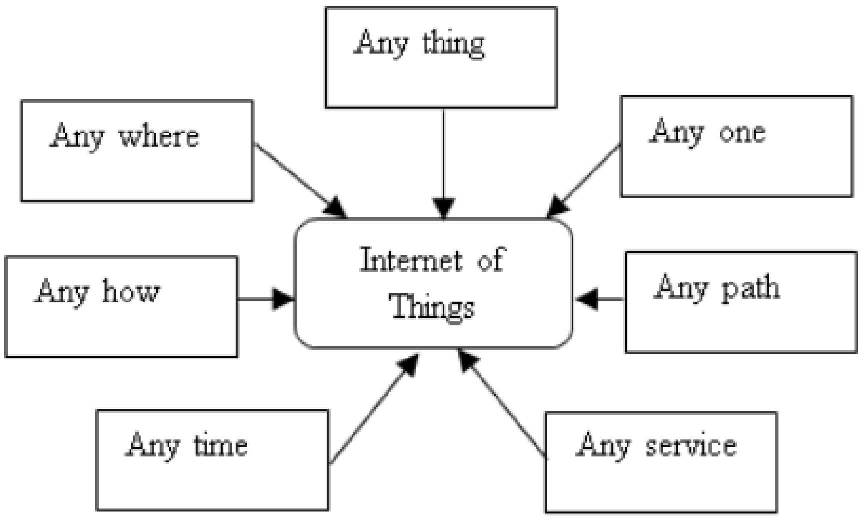


**Figure 5** “Any“ Paradigm in IoT [15]

### 1.3.2 Security and Privacy in IoT

The connectivity without any boundary can create huge opportunities and advantages in everyday activities such as the control and management of home devices, cars or even the equipment of a company as well as in the long run reduce of consuming energy and at the same time reduce of cost. Interconnected devices allow users to perform a various set of actions. Apparently, this facilitates everyday life. Therefore, the need for potential for management the increasing demands of data activity is augmented.

The Iot can be characterized as a technology with a lot of challenges in security systems. It is an Internet extension, which has a distributed, open-application architecture, and can be applied to vulnerable technologies owing to various loopholes. The Internet is by itself an unsecured technology, vulnerable to hackers’ breaches. Therefore, all the connected devices to it are led to confront the same danger.

Equivalent are the demands in terms of encryption and security of the data. Since, every IoT device is connected with each other, issues of privacy and data sharing arise as authenticity, confidentiality and integrity of data and services in IoT have to be ensured.

### 1.3.3 Security architecture of IoT

#### Perceptual Layer

It is the lowest layer of architecture of IoT. Here, nodes build an ad hoc network with a dynamic distribution for the collection of information. In this level, there are RFID tags, sensors, terminals, etc [17].

The security module in the perceptual layer maintains the logical security mechanism which consists of an encryption mechanism and a security algorithm. Various algorithms are applied to identity authentication mechanism, hide terminal identity and preserve data secrecy. Security can be categorized into terminal identity security and interactive data security. The first manages and destroys the terminal key of any terminal, if timeliness and legality are ensured. Interactive data security ensures that the data generated or forwarded by the terminal is not intercepted by unauthorized access. The perceptual layer has incomplete control of the behavior of the object. Every threat and weakness within security mechanism and nodes must be examined and every malicious behavior must be detected so the availability of the system can be ensured [15].

The main threats are [15] [16]:

- Physical capture: Device nodes are statically deployed, that’s why it is easy to be breaches by hackers putting in danger all the information that was collected.
- Brute force attack: The limited computation of the sensor nodes can endanger the information storage.
- Attack of the clones: The hardware structure of nodes is simple, so they can easily be deployed by the attacker.
- Impersonation: In distributed environments, authentication is difficult and so malicious nodes can use fake identity to attack.

• Routing Attack: The intermediate nodes can be used to attack the flow of the data causing a breach in network.
• Denial of Service (DoS) Attack: Nodes have finite processing ability and so attackers can use Denial of Service (DoS) methodology to stop services in network.

**Network Layer**

The next layer of architecture of IoT consists mainly of intercity nets and backbone network. Its main function is to transmit and process information from perceptual layer. Some of the main threats in this level, because of the strong closure of the backbone network, are malicious behaviors against right path topology and forwarding data, DoS attack. The authentication technology controls the integrity of node, which is responsible for the choice of a reliable route that the data is going to follow for which their integrity is ensured by security routing technologies. Security monitoring management platform is used to detect any malicious behavior [16].

**Middleware/Support layer**

The third layer functions as the connection between the Network Layer and the Application Layer, which will be brought up below. It offers services for the execution of basic actions such as Web service and application program interface. This layer ensures that the interaction of applications is legal [15].

The main threats are [15] [16]:

• DoS Attack: The service availability can be destroyed by hackers' breaches who can terminate the normal services of the network.
• Unauthorized Access: In an open architecture, with misconfiguration access control rights attacks can easily threaten the security by intruding the network.
• Data Attacks : These attacks are focused on data services.
• Session Attacks: The attacker can redo sessions to gain illegal access.

**Application Layer**

The highest layer of architecture of IoT is actually the interface through which services are allocated according to the demands of the user when they handle their devices. Through the use of standard web service protocols and service composition technologies, the cooperation between distributed systems and applications is achieved [15].

The main threats are [15] [16]:

- Privacy issues: The application of IoT is executed on common operating systems and hackers can hijack information.
- DoS Attack: These attacks function as in the rest of the layers. The intruder destroys the availability of varied services.
- Malicious Code: Attackers can upload malicious code through vulnerabilities in devices, which can spoil normal working
- Social Engineering: Attackers can easily obtain information from users.

### 1.3.4 Privacy in IoT

“Platform for Privacy Preferences” (P3P) provides standards for privacy rules and regulations and defines parameters of privacy based on the needs of personal information for running the services and the privacy requirements set by the user. The sheer amount of data that IoT devices can generate is staggering. This creates more entry points for hackers and leaves sensitive information vulnerable.

A lot of researches have been conducted in relation with the way consumers perceive the dangers that arise and the result is that a high percentage worries about their personal information in case they can be hacked and used in ignorance of them. The users must have full control of their personal data, be aware of which authority collects them, as well as where they are saved. Control on the diffusion of all such information is impossible with current techniques [14].

Non encoded data can give information like what moves are taking place in the house since they happen according to which device or devices are used. Various encoding techniques are applied to protect such data which are saved in the network and also their transfer from one network to another. There are plenty of encoding algorithms such as AES, SHA1, MD5, RSA, etc. Processors of high speed and memory of huge capacity are demanded so as to achieve the optimal performance of these algorithms. Respectively, modern Internet protocols such as TLS/SSL and IPSec for the sake of which the integral transfer of data is accomplished from one network to another demand higher processing power of that which the devices of such a system provide. Therefore, a big research needs to be conducted in the field of security and privacy [15].

## 1.4 Demand Response

### 1.4.1 Definition

Demand response has as main target the balance between the demand for power and the supply, by adjusting the demand for power instead of the supply. It refers to actions by customers that change their consumption (demand) of electricity in response to price signals. Electricity cannot be stored economically, the supply and demand for electricity must maintain balance in real time. Demand response programs offer the possibility to downsize the pressure on the transmission, by increasing the load when the power supply is more available or reduce the load at peak periods, without creating changes in the daily life of the consumers [19]. The consumers by this way have the potential to adapt their electricity usage in response to time-based rates or other forms of financial incentives. Methods in demand response include critical peak pricing, variable peak pricing, real time pricing, and critical peak rebates. Such programs can lower the cost of electricity in wholesale markets on a regular basis and thereby dampen the severity of price spikes in wholesale markets on extreme days. As consumers become more aware of their energy-usage patterns and are willing to invest in more energy-efficient appliances to lower their demand, energy efficiency can be achieved [20] [21].

Demand response technologies can be deployed at three distinct levels [19]:

- Industrial level: large manufacturing plants can adapt the production of electric energy in order to reduce the energy costs.
- Commercial level: main purpose is the reduce of energy costs through an automated management of air-conditioning and lighting systems.
- Residential level: with smart appliances linked to automated solutions in order to minimize the impact of demand response in daily life.

### 1.4.2 Categories Of Demand Response

According to Federal Energy Regulatory Commission (FERC) there are two types of demand response, which are highly interconnected and have as their aim the maintenance of reliability of the system. Each of these categories is composed of several programs for complementary purposes [21].

Price-Directed Demand Response includes dynamic pricing programs, in which the price of electricity is not fixed but varies in time. These programs lead electricity customers to move away from flat pricing which does not offer any price signals and encourage them to over-consume [20] [21].

Price-Directed Program [18] [20] [21]:

- Time-of-Use (TOU) Program: divides a day in multiple periods and sets the electricity price different for each period. High electricity demand means higher prices than normal and the opposite.
- Real Time Pricing (RTP) Program: includes periods with lower or higher than normal prices but they are not known in advance. Customers are typically notified of RTP prices on a day-ahead or hours-ahead basis.
- Critical Peak Pricing (CPP) Program: raises the price of electricity at peak periods so as to reduce demand.

Incentive-Based Demand Response Programs offer payments for customers' participation to reduce their loads at periods that system reliability is at risk. Some demand response programs penalize customers that fail to fulfill their contractual commitments [20] [21].

Incentive-Based Program [18] [20] [21]:

- Direct Load Control (DLC) & Interruptible Service (IS): electricity supplier or system operator is able to shut down or cycle a customer's electrical equipment on a short notice.
- Curtailable (CS) Service: users get rewarded for reducing their load to a certain period of time.
- Demand Bidding/Buy Back (DB)
- Emergency Demand Response Program (EDRP)
- Capacity Market Program (CAP)
- Ancillary Service (A/S) Markets

Demand Bidding, Emergency Demand Response, Capacity Market and Ancillary Service Markets programs offer the user the potential to receive a price per amount of load reduction according to their performance [20].

Demand Response programs use sensors that can perceive peak load problems and utilize automatic switching to divert or reduce power, removing the chance of overload and the resulting power failure. Advanced metering infrastructure expands the range of time-based rate programs that can be offered to consumers. Smart customer systems such as in-home displays or home-area-networks can make it easier for consumers to change their behavior and reduce peak period consumption from information on their power consumption and costs [22].

### 1.4.3 Benefits Of Demand Response

Demand response creates many benefits for market participants and for the efficiency of electric markets.

**Participant Benefits**

The consumers turn to demand response as they pursue to control and reduce their energy consumption. This offers them a lot of financial benefits as they are in the position to use less energy when prices are high or to shift usage to lower priced-hours. They also gain as far as the reliability is concerned that the system offers as the possibility of being involuntarily curtailed is lowered, incurring even higher costs [21].

**Market and System Benefits**

More efficient use of the electricity system provokes saving of the variable supply cost in the purchases(or in the commerce). In the long run, the need to build new generation, transmission, or distribution infrastructure eliminates offering operational and capital cost savings to distribution utilities. Eventually these savings may be passed onto most retail customers as bill savings [21].

**Additional Benefits**

Demand response provides additional choices and tools for customers to monetize their demand reductions and manage their electricity costs. Moreover, demand response offers the potential to the costumers as well as to the utilities to manage in a better way the dangers that may arise from the instability of prices and also from the unexpected accidents during the function of the system at critical times. Furthermore, demand response ensures the reliability and adequacy of the system by averting consequences of forced outages that impose financial costs and inconvenience on customers. Finally, the implementation of demand response contribution to reduced generation facility construction may provide environmental and aesthetic benefits [21].

# 2. NetLogo

## 2.1 NetLogo

NetLogo is an agent-based programming language and integrated modeling environment which is used for the simulation of both simple and more complex social and physical phenomena. It actually has a huge resonance to researching as well as educational purposes and it also covers a significant range of scientific fields such as economics, physics, chemistry, biology, psychology etc [26] [27].

## 2.2 History of NetLogo

NetLogo arises from the combination of StarLisp and Logo. *Lisp (StarLisp) programming language was created in 1986 by Cliff Lasser and Steve Omohundro. Its target was to consist an effective and at the same time of high level language for the programming of Connection Machine (CM) a series of massively parallel Supercomputers. The Logo language though, consists a programming language for educational purposes and it was created in 1980 by Seymour Papert.

That is how the name came out. The name “Logo” came out as it is one dialect of the Logo language and the word “Net” because its focus is on the interlinked nature of phenomena which could be modelized on that. It refers also to HubNet, a technology which enables the use of NetLogo for the realization of modeling one phenomenon in a classroom and as it was already mentioned above its design aimed at educational purposes [24] [26] [27].

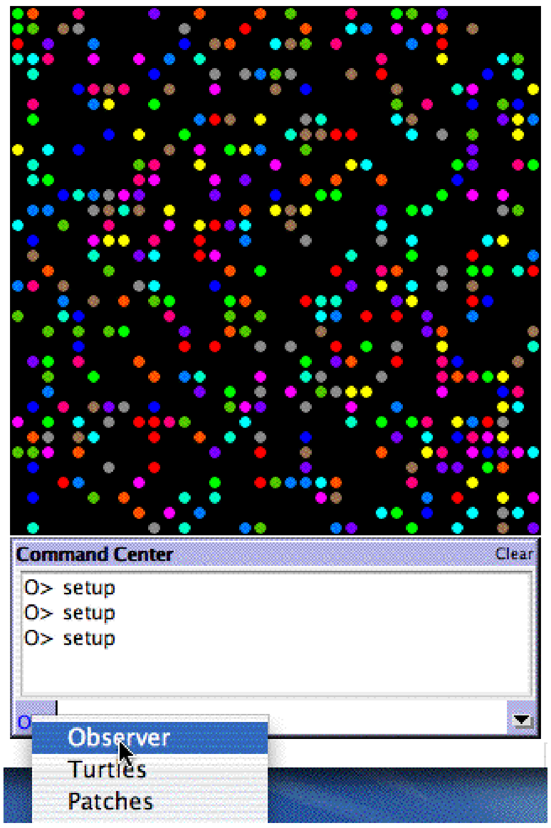
NetLogo is the next generation of another multi-agent model environment StarLogoT. Although the original design of StarLogoT, was made for using it in schools for educational purposes, it ended up becoming famous in various researchers. The version of NetLogo that appeared for the first time in 1999 by Uri Wilensky at the Center for Connected Learning and Computer-Based

Modeling, aimed at serving both areas and that is the reason why the renewed version includes plenty of extra features. NetLogo allows students to be part of a simulation observing the changes which take place under various circumstances. At the same time, it allows programmers to implement their own models. It is rather simple for the students and professors to use it as well as a powerful tool for scientific researches [24] [26] [27].

## 2.3 Language

Logo was especially known for the “turtles graphics”, in which the only turtle that could be used every time was moving on the screen leaving its traces behind. Therefore NetLogo inherited also the “turtle” with the difference that the user now has the potential to handle thousands of them. Generally, its world consists of agents, to wit objects that can execute specific orders. There are 4 types of agents [26] [27].

**Turtles:** Turtles are agents that move around in the world.
**Patches:** The world is two-dimensional and is divided up into a grid of patches. Each patch is a square piece of "ground" over which turtles can move.
**Links:** Links are agents that connect two turtles. These turtles are sometimes also called nodes.
**The observer:** There is only one observer that is looking out over the world of turtles, links and patches.



```
O> setup
O> setup
O> setup
```

**Figure 6** An example of agents in NetLogo

An agentset is a set of agents. All agents which are in an agentset must be of the same type (i.e. turtles, patches, or links). An agentset does not have any particular order. In fact, it is always in a random order. What is powerful about the agentset concept is that the user can construct agentsets that contain only some agents. Agentsets are responsible for much of NetLogo's expressive power. The NetLogo offers the potential to the user to define different "breeds" of turtles and links. Since the user defines them, he can make them behave as he wishes through the appropriate orders. All numbers in NetLogo are stored internally as double precision floating point numbers, as defined in the IEEE 754 standard [26] [27].

In NetLogo, there is a set of primitives that give the user the power to interact with outside files. There are two main modes when dealing with files: reading and writing. The difference is the direction of the flow of data. When the user is reading information from a file, the data that is stored in the file flows into user's model. On the other hand, writing allows data to flow out of user's model and into a File.

NetLogo is a “simulated parallel” environment. In parallel computing the programs should be carried out in a way that nondeterminism is avoided. The term nondeterminism, refers to programming languages which have the potential to determine to specific areas in the code choice points, i.e. alternative ways for which flow the rest of the code is about to follow. The programmer can determine specific alternative solutions as well as the program that is going to choose among them at run time. By this way, NetLogo is becoming even more usable for the beginners (or inexperienced) programmers [24] [25] [26].

## 2.4 NetLogo – 5.3.1

### 2.4.1 First Impression

The version of NetLogo - 5.3.1 is the latest version before NetLogo – 6.0.

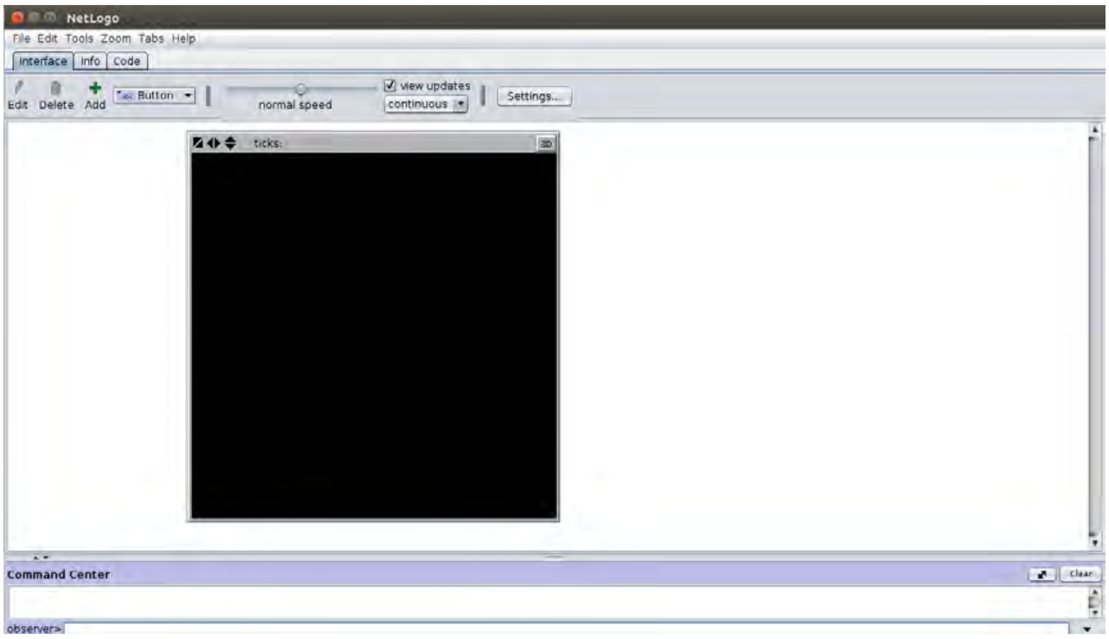


**Figure 7** NetLogo – 5.3.1

Figure 7 shows what the user sees when NetLogo opens. At the top of NetLogo's main window are three tabs labeled "Interface", "Info" and "Code". Only one tab at a time can be visible, but the user can switch between them by clicking on the tabs at the top of the window. Right below the row of tabs is a toolbar containing a row of controls. The controls available vary from tab to tab.

The Interface tab is where the user watch his model run. It also has tools that can be used to inspect and alter what is going on inside the model. When NetLogo opens the first time, the Interface tab is empty except for the view, where the turtles and patches appear, and the Command Center, which allows the user to issue NetLogo commands. The Info tab is where documentation on the model is found. The Code tab is where the code for the model is stored. Also the Command Center allows the user to issue commands directly, without adding them to the model's procedures. A command is an action for an agent to carry out, resulting in some effect. A reporter is instructions for computing a value, which the agent then "reports" to whoever asked it. Typically, a command name begins with a verb and most reporter names are nouns or noun phrases. In many NetLogo models, time passes in discrete steps, called "ticks". NetLogo includes a built-in tick counter so the user can keep track of how many ticks have passed. The current value of the tick counter is shown above the view [25].

## 2.4.2 Models Library

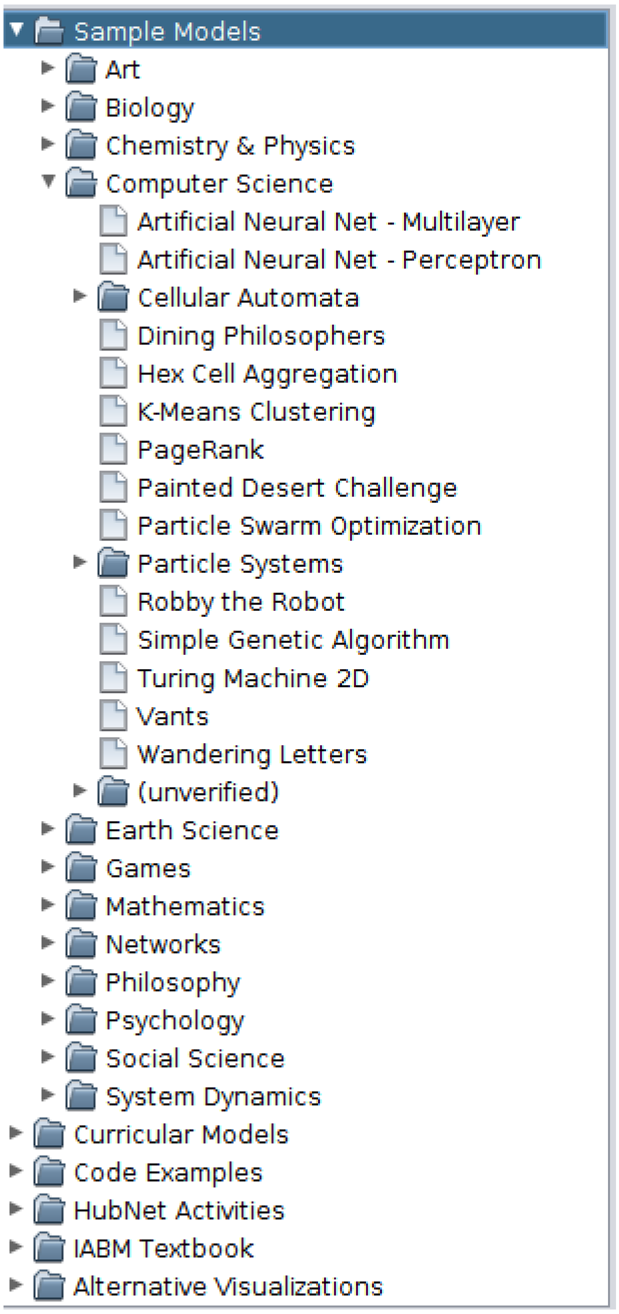


**Figure 8** Models Library of NetLogo

**Sample Models:** They are examples of coding and documentation practice.

**Curricular Models:** It contains groups of models that are intended to be used together in an educational setting as part of a curricular unit. The models also appear, sometimes in different form, in Sample Models [27].

**Code Examples:** They are not complete models, but short illustrations of particular features and coding techniques [27].

**HubNet Activities:** They are for use with the HubNet participatory simulation architecture.

### 2.4.3 Shapes

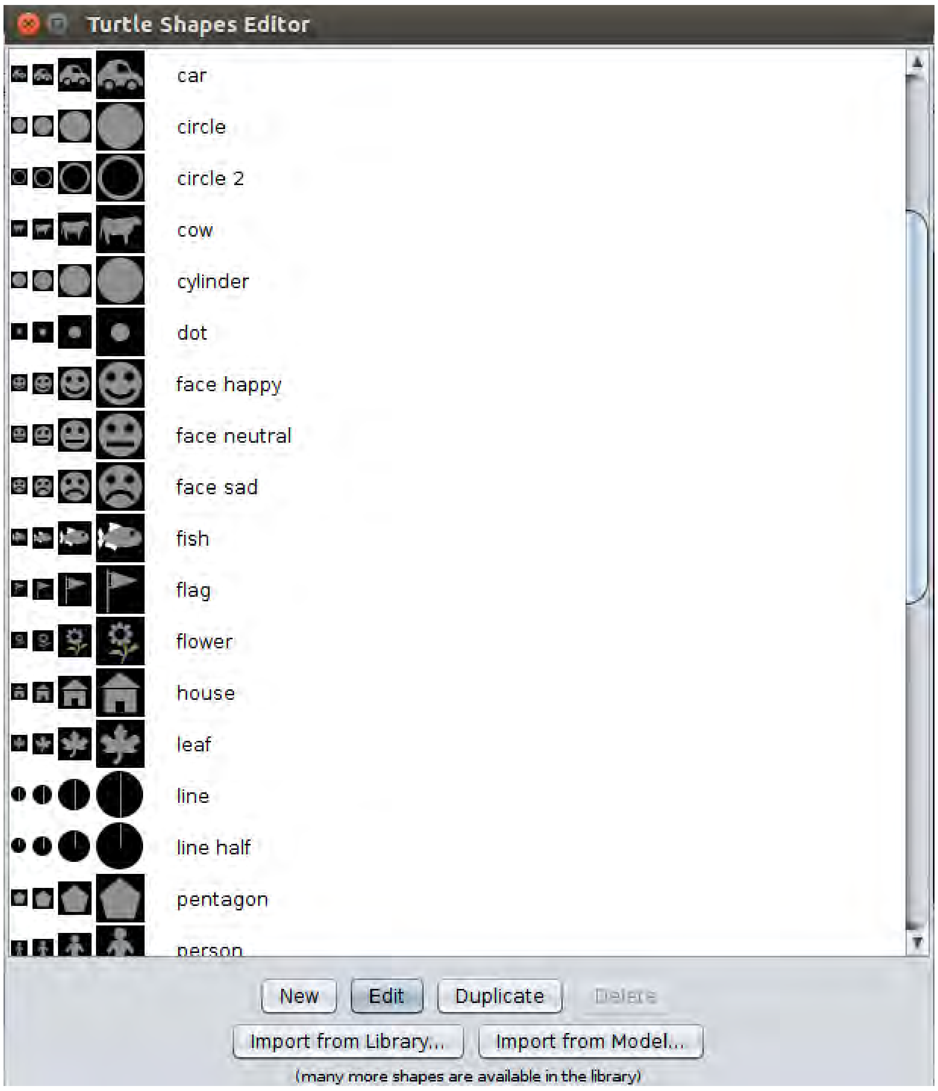


**Figure 9** Some of the turtles shapes

The Turtle and Link Shape Editors (Figure 9) allows the user to create and save turtle and link designs. NetLogo uses fully scalable and rotatable vector shapes, which means the user can create designs by combining basic geometric elements, which can appear on-screen in any size or orientation. Many more turtle shapes are available by using the Import from library... and Import from model… buttons. By default there is only one Link shape in a model, a single straight line with a simple arrowhead (if the link happens to be directed) [25].

## 2.4.4 Interface Tab

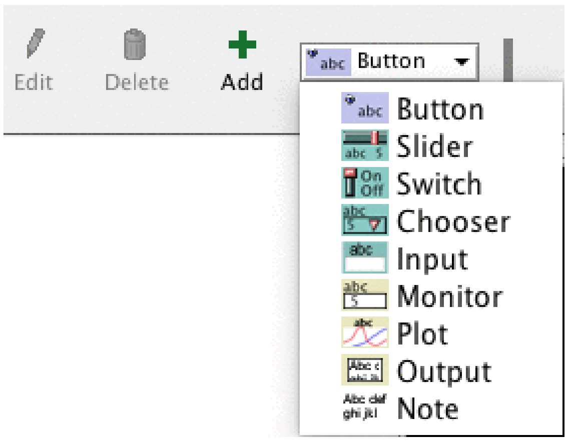


**Figure 10** Toolbar on Interface Tab

The toolbar on the Interface tab contains buttons that let the user edit, delete, and create items in the Interface tab and a menu that let the user select different interface items (such as buttons and sliders) (Figure 10) [25] [27].

**Buttons:** A button is either once or forever. When the user clicks on a once button, it executes its instructions once. The forever button executes the instructions over and over, until the user clicks on the button again to stop the action.

**Slider:** Sliders are global variables, which are accessible by all agents. They are used in models as a quick way to change a variable without having to recode the procedure every time. Instead, the user moves the slider to a value and observes what happens in the model.

**Switch:** Switches are a visual representation for a true/false global variable. The user has the option to set the variable to either on (true) or off (false) by flipping the switch.

**Chooser:** Choosers let the user choose a value for a global variable from a list of choices, presented in a drop down menu. The choices may be strings, numbers, booleans, or lists.

**Input:** Input Boxes are global variables that contain strings or numbers. The model author chooses what types of values the user can enter. Input boxes can be set to check the syntax of a string for commands or reporters. Number input boxes read any type of constant number reporter which allows a more open way to express numbers than a slider. Color input boxes offer a NetLogo color chooser.

**Monitor:** Monitors display the value of any reporter. The reporter could be a variable, a complex reporter, or a call to a reporter procedure. Monitors automatically update several times per second.

**Plot:** Plots show data the model is generating.

**Output:** The output area is a scrolling area of text which can be used to create a log of activity in the model. A model may only have one output area.

**Note:** Notes lets the user add informative text labels to the Interface tab. The contents of notes do not change as the model runs.

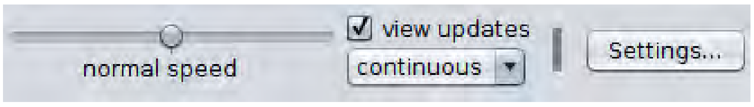
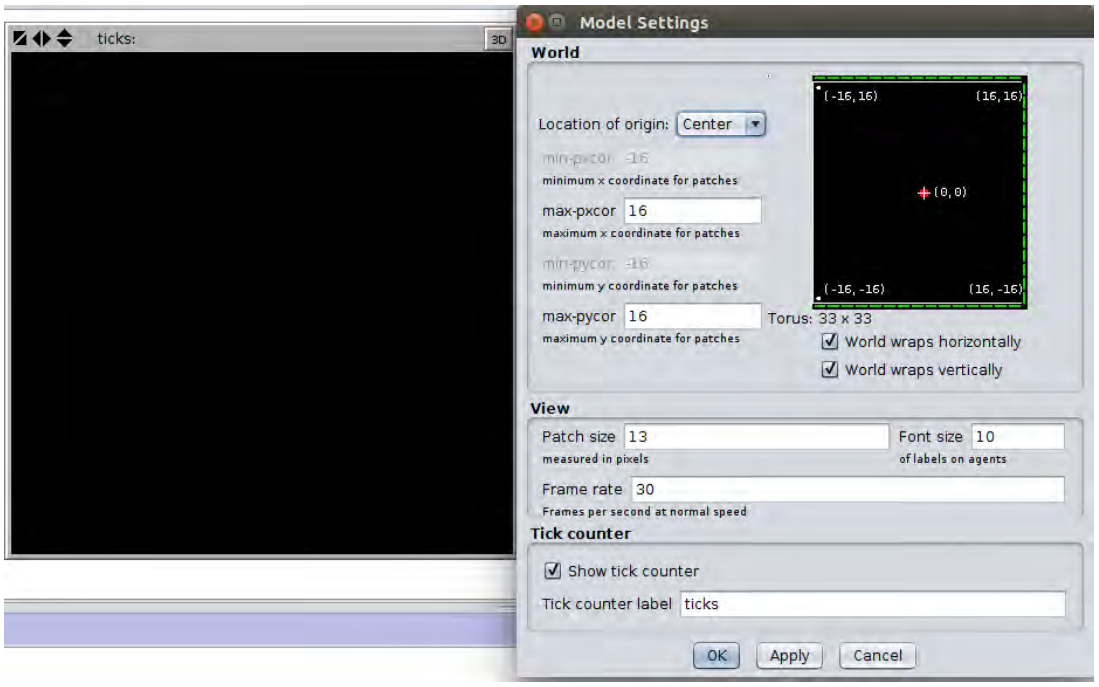


**Figure 11** Toolbar on Interface Tab

The slider lets the user control how fast the model runs. The view updates checkbox controls whether view updates happen at all. The "Settings..." button allows the user to change model settings (Figure 11) [25].

By pressing the "Settings..." button in the Interface toolbar opens a window as shown in Figure 12.



Model Settings

World

Location of origin: Center

min-pxcor -16
minimum x coordinate for patches

max-pxcor 16
maximum x coordinate for patches

min-pycor -16
minimum y coordinate for patches

max-pycor 16
maximum y coordinate for patches

Torus: 33 x 33

☑ World wraps horizontally

☑ World wraps vertically

View

Patch size 13
measured in pixels

Font size 10
of labels on agents

Frame rate 30
Frames per second at normal speed

Tick counter

☑ Show tick counter

Tick counter label ticks

**Figure 12** Model Settings

The large black square in the Interface tab is the 2D view. It is a visual representation of NetLogo world of turtles and patches. Initially it is all black because the patches are black and there are no turtles yet [25].

There are three groups of settings world, view, and ticks counter. World settings affect the properties of the world that the turtles live in. View and tick counter settings only affect the appearance, changing them will not affect the outcome of the model. The world settings allow the user to define the boundaries and topology of the world. The topology of the NetLogo world has four potential values, torus, box, vertical cylinder, or horizontal cylinder. At the top of the left side of the world panel he can choose a location for the origin of the world either "Center", "Corner", "Edge", or "Custom". By default the world has a center configuration where (0,0) is at the center of the world. Also below the preview there are two checkboxes, the world wrap settings. These allow the user to control the topology of the world. The view settings allow the user to customize the look of the view without changing the world. Tick counter settings control the appearance of the tick counter which is visible (or not) in the view control strip [25].

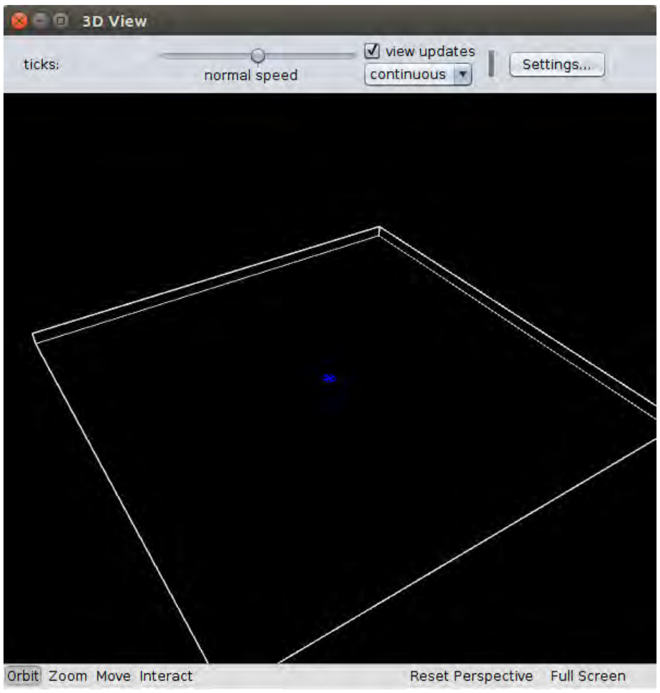
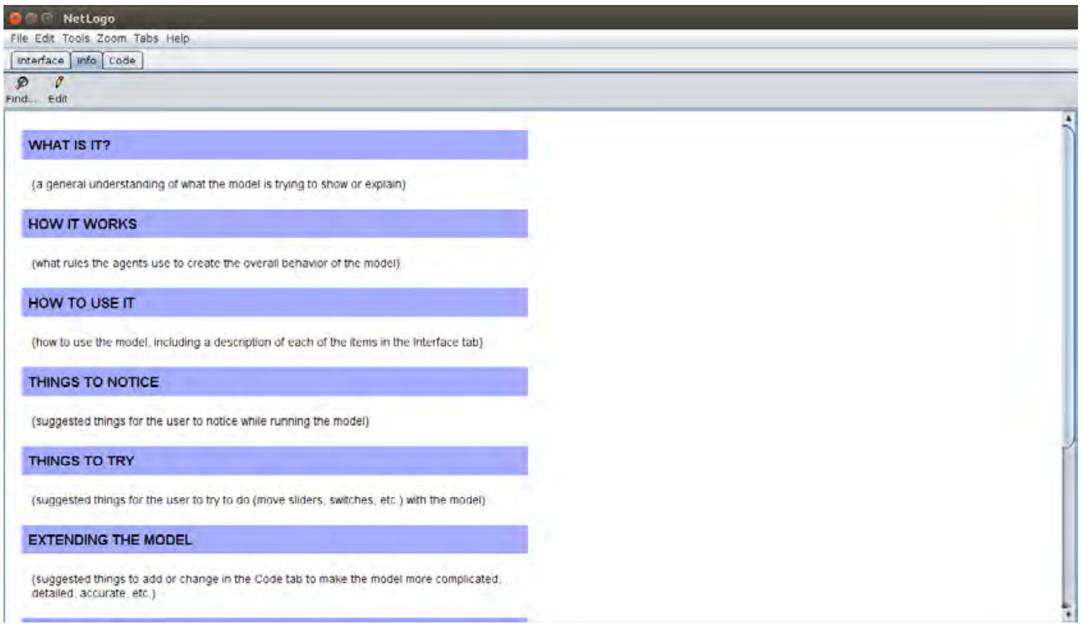


**Figure 13** 3D View of the world

The user can open the 3D View, an alternate visual representation of the world [25].

### 2.4.5 Info Tab



WHAT IS IT?

(a general understanding of what the model is trying to show or explain)

HOW IT WORKS

(what rules the agents use to create the overall behavior of the model)

HOW TO USE IT

(how to use the model, including a description of each of the items in the Interface tab)

THINGS TO NOTICE

(suggested things for the user to notice while running the model)

THINGS TO TRY

(suggested things for the user to try to do (move sliders, switches, etc.) with the model)

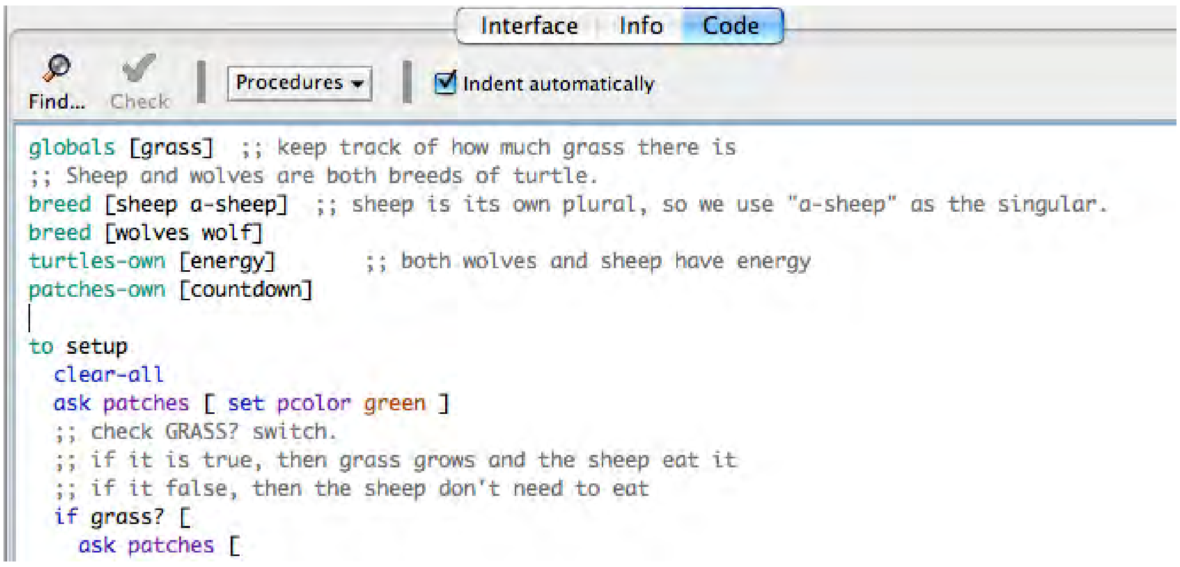
EXTENDING THE MODEL

(suggested things to add or change in the Code tab to make the model more complicated, detailed, accurate, etc.)

**Figure 14** Info Tab in NetLogo – 5.3.1

The Info tab provides an introduction to a model. It explains what system is being modeled, how the model was created, and how to use it. It may also suggest things to explore and ways to extend the model (Figure 14) [26].

### 2.4.6 Code Tab



```
globals [grass]  ;; keep track of how much grass there is
;; Sheep and wolves are both breeds of turtle.
breed [sheep a-sheep]  ;; sheep is its own plural, so we use "a-sheep" as the singular.
breed [wolves wolf]
turtles-own [energy]       ;; both wolves and sheep have energy
patches-own [countdown]

to setup
  clear-all
  ask patches [ set pcolor green ]
  ;; check GRASS? switch.
  ;; if it is true, then grass grows and the sheep eat it
  ;; if it false, then the sheep don't need to eat
  if grass? [
    ask patches [
```

**Figure 15** Code Tab and an example of code in NetLogo – 5.3.1

The Code tab contains the complete code for the model (Figure 15). Language elements are automatically color-coded so the code's structure is more clearly visible. To determine if the code has any errors, the user may press the "Check" button. Switching tabs also causes the code to be checked. If there is an error in the code, the part of the code that contains the error will be highlighted an error message will appear (Figure 16) [25].

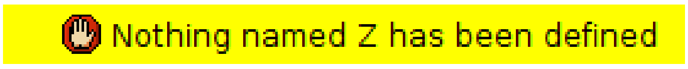

Nothing named Z has been defined

**Figure 16** An example of an error message

## 2.5 Implementation

NetLogo is written mostly in Scala, with some parts in Java and runs on the Java Virtual Machine. NetLogo Web is a version that runs on JavaScript, instead of the JVM, so models may be run in a web browser. It does not have all features of the desktop version. The platform JAVA on which was written makes possible the use of independent math libraries.

NetLogo is free and open source software, anyone can download it for free and build models without restriction.

NetLogo is a hybrid compiler/interpreter. The compiler follows a tailormade way according to a procedure of annotation without having the code interrupted. As a result the compilation process becomes even more efficient.

NetLogo was a closed platform now it is becoming extensible. Since NetLogo 2.0.1, there is an application programmer's interface (API) for extensions so that users can add new elements to the language by implementing them directly in Java. Agents that make sounds and music using Java's MIDI capabilities, or communicate with remote computers, and many other things [26] [27].

## 2.6 Extensibility

NetLogo is a tool that serves educational and researching purposes. Its code demonstrates a distinctive convenience in writing even by a beginner programmer as well as in reading by someone completely inexperienced. This obviously has as a result the simulation models to be built and becoming understood easily allowing in this way the researchers to communicate with each other, to compare and verify their results [26] [27].

In computer programming, an application programming interface (API) is a set of defined methods of communication between various software components. The NetLogo environment includes a programming language, a complier, an interpreter, a syntax highlighting editor, an interface builder and many more. There is the danger though some of the user’s needs not to be able to be served with none of these components. Yet, NetLogo is extensible and offers an API for extensions so that the user can have the potential to add new elements to the language by implementing them in JAVA [26] [27].

# 3. Implementation of the application in NetLogo

## 3.1 In general

The development of home automation takes place at the same time with a growing consumer awareness of the need to consume energy responsibly and the potential to use new technologies to control household expenditure. Concerning the consumers, the major value of the smart homes is that they have the potential to control and manage the interconnected devices of their house via a simple interface no matter where they are. The main target is a better quality of life and more energy efficient living that will bring with it some financial savings for households. For the utilities, companies that provide household with energy supply have the opportunity to widen their service offering and extend their relationship with householders.

The application which was implemented in terms of this thesis is a multi-agent application that aims at offering to its users/consumers the potential to control the electricity consumption of their homes. The user can simulate the house environment, regulate the amount of the devices, the consumption of each device as well as create scenarios for the time that each device will operate.

## 3.2 Why NetLogo

Multi-agents systems have been proven a really powerful tool for modeling and understanding phenomena in many fields like economics, physics, chemistry, social sciences etc. Systems and models that need to be analyzed are becoming more and more complex. Some of the agent platforms that can be used in simulating real-life study cases are JADE, Repast, MASON, NetLogo, Swarm, JACK etc. JADE still remains the most popular platform.

However, NetLogo is one of the most popular platforms that can be used for specific applications domain like artificial life and behavioral observation. A specialized platform means that it was designed for that domain and it is efficient. It actually has a huge resonance to researching as well as educational purposes. It is rather simple for the students and professors to use it as well as a powerful tool for scientific researches. Moreover NetLogo is extensible and offers an API for extensions so the user can add new elements to the language by implementing them directly in Java. It is written mostly in Scala and Java and that means that makes possible the use of independent math libraries as well as many other ready models libraries that exist in NetLogo and can be used in simulations. Finally it is a free and open software and anyone can build models without restriction [26] [27].

## 3.3 Description of the application

Initially the types of devices that exist in the simulation environment and the user can make use of, are separated into 5 categories: Lights, Kitchen, Living Room, Bathroom and Other Devices.

**Lights:**

- Fluorescent Lights (20W)
- Led Lights (15W)

**Kitchen:**

- Fridge (100W)
- Stove (4kW)
- Dishwasher (2kW)
- Toaster (15.kW)
- Coffee Machine (1.5kW)

**Living Room:**

- Air condition (1kW)
- Television (150W)

**Bathroom:**

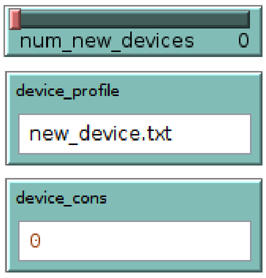
- Washing Machine (2kW)

- Water Heater (4kW)

**Other Devices:**

- Iron (2kW)
- Computer (100W)
- Vacuum Cleaner (2kW)

Every device is regarded as an agent in NetLogo. For each type of device, a different color and design has been defined for its representation in the world of NetLogo. The user has the potential to choose the amount of devices he wishes for the simulation of his house. Apart from the already existing devices, the user can add a new type of device which does not belong to any of the above categories. Yet, he has to declare to the interface the consumption of this type of device, as well as create the relevant function scenario in text file (Figure 17).



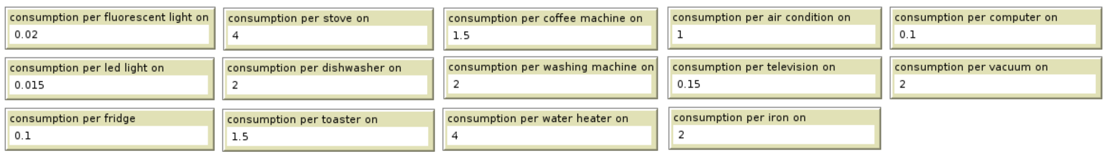
num_new_devices 0

device_profile
new_device.txt

device_cons
0

**Figure 17** Amount of new devices, function scenario, consumption

At the same time the user in the interface can keep track of the consumption of every device in Watt, as they have been defined (Figure 18), but also he can give the operating data of every device in text files through the scenarios that they had created.



consumption per fluorescent light on
0.02

consumption per stove on
4

consumption per coffee machine on
1.5

consumption per air condition on
1

consumption per computer on
0.1

consumption per led light on
0.015

consumption per dishwasher on
2

consumption per washing machine on
2

consumption per television on
0.15

consumption per vacuum on
2

consumption per fridge
0.1

consumption per toaster on
1.5

consumption per water heater on
4

consumption per iron on
2

**Figure 18** Consumption (kW) of the devices that exist in the application

Finally, the user can choose the pricing policy he wishes for the calculation of the cost either a fixed electricity price or interchanges in price per day, per hour or per 10 minutes (per tick).

Having given this data, the user now can go on with the simulation. The setup button creates the whole amount of devices, to wit the agents, and it represents them in the world. Respectively, the locate_devices button places the agents in the world according to the dimensions that have been determined for each type of device. Finally, the get_measurements button makes all the necessary calculations for the cost that the user can also see in the interface (Figure 19). During this process, the user can watch in the world which of the devices he had selected are activated at any time depending on the scenario he had created.

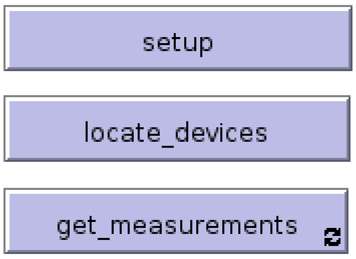


**Figure 19** Buttons for the execution of the process of the application

Finally, in the interface various plots are created which show the consumption of energy for each category of devices, the overall consumption and the cost. The x axis in plots symbolizes the instant moment whereas the y axis the consumption (kW) according to the consumption of every type of device, or the cost (€).

## 3.4 Cost Calculation

For the cost calculation we assume:

- N: number of each device type
- n: the identity of each appliance

- $k_i^{(n)}$: the function of each device, for i = [1, 2, ..., m]
- $k_i^{(n)} \in$ {0,1, ... 100}: the percentage of power consumed during the time period we have defined
- m: the count of 10-minutes periods in the scenario that used to simulate
- $cons^{(n)}$: max consumption of each appliance in kW

Thus, the first type that occurs is the following:

$$A^{(n)} = \frac{1}{6} \sum_{i=1}^{m} \left(\frac{k_i^{(n)}}{100} * cons^{(n)}\right) \quad \text{(kWh)}$$

where,
$A^{(n)}$: the total amount of kWh resulting from data for the n-device.

The sum divided by 6, because measurements are made every 10 minutes and we need to convert the data to an hour. Simultaneously the division by 100 in the sum is because the operation of each device is in percentage.

When there is a fixed price (€ / kWh), the calculations of the cost would be very simple:

$$B^{(n)} = A^{(n)} \cdot price \quad \text{(€)}$$

where,
price: fixed price

$B^{(n)}$: current cost for the particular device (n)

Therefore, total cost over all devices is equal to:

Total Cost = $\sum_{n=1}^{N} B^{(n)} = \sum_{n=1}^{N} price * \frac{1}{6} * \sum_{i=1}^{m} \left(\frac{k_i^{(n)}}{100} cons^{(n)}\right)$ (€)

In case the electricity value varies, like the examples we simulated in this thesis, type occurs with the same logic as above but vary a little.

$$TotalCost = \sum_{n=1}^{N} \frac{1}{6} * \sum_{i=1}^{m} \frac{k_i^{(n)}}{100} * cons^{(n)} * price_i \quad (€)$$

where,

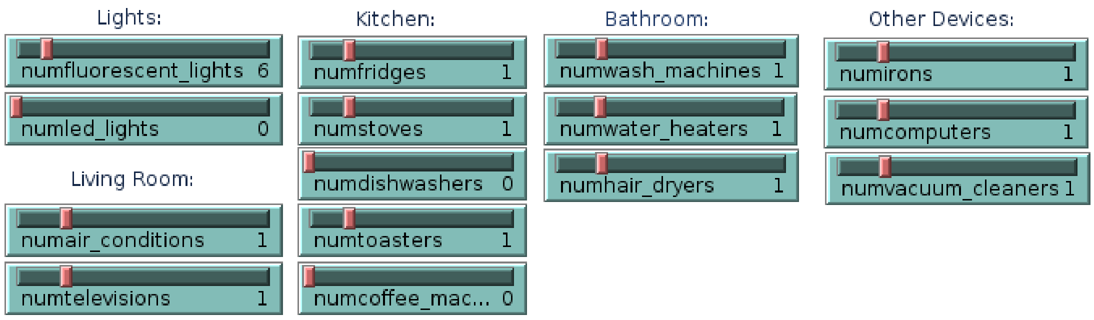
$price_i$ : electricity price for the $i^{th}$ 10-minutes period.

## 3.5 Simulation of scenarios

### 3.5.1 Daily life of a student

The first simulation example that we are going to analyze refers to the everyday life of a student for 7 days in the summer period.

We assume that a student house has the following devices (Figure 20):



Lights:
numfluorescent_lights 6
numled_lights 0

Living Room:
numair_conditions 1
numtelevisions 1

Kitchen:
numfridges 1
numstoves 1
numdishwashers 0
numtoasters 1
numcoffee_mac... 0

Bathroom:
numwash_machines 1
numwater_heaters 1
numhair_dryers 1

Other Devices:
numirons 1
numcomputers 1
numvacuum_cleaners 1

**Figure 20** Amount of devices that have been regulated with the sliders

At the same time, scenarios for 7 days with measurements per 10 minutes have been created in advance for the function of each device in text files. The measure for every day starts at midnight. The scenarios have been created based on the needs and habits a student has, i.e. more usage of computer and TV contrary to other appliances such as the washing machine, the oven and the iron. It is also observed an intense usage of the air condition since we have assumed that it is a summer.

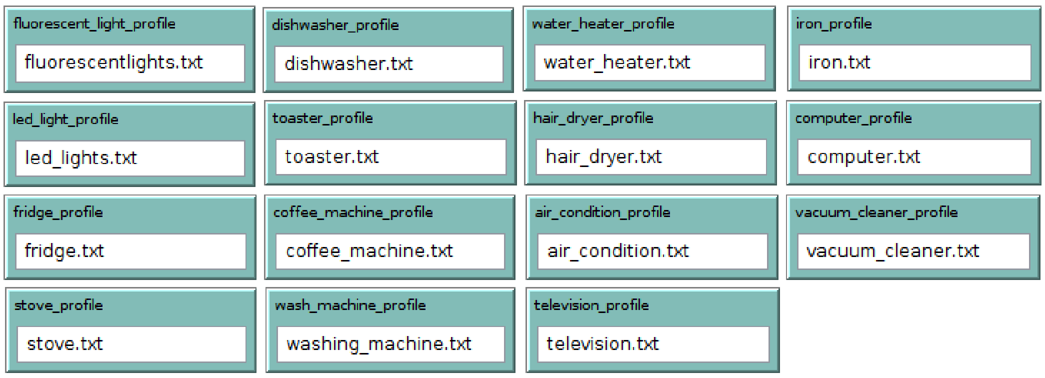

fluorescent_light_profile: fluorescentlights.txt
dishwasher_profile: dishwasher.txt
water_heater_profile: water_heater.txt
iron_profile: iron.txt
led_light_profile: led_lights.txt
toaster_profile: toaster.txt
hair_dryer_profile: hair_dryer.txt
computer_profile: computer.txt
fridge_profile: fridge.txt
coffee_machine_profile: coffee_machine.txt
air_condition_profile: air_condition.txt
vacuum_cleaner_profile: vacuum_cleaner.txt
stove_profile: stove.txt
wash_machine_profile: washing_machine.txt
television_profile: television.txt

**Figure 21** Functioning scenarios of devices

We also assume that the domestic invoice includes the following electricity prices:

- 23:00 – 07:00 the night invoice is in force with charge 0.0661 €/kWh
- the rest of the day the charge is 0.0946 €/kWh

The archive that includes the billing data is also given through the interface (Figure 22).

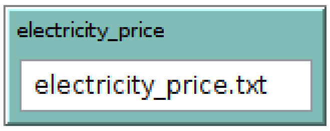

electricity_price: electricity_price.txt

**Figure 22** Archive with the electricity prices

The pricing policy chosen for this example is that with the interchanges of the electricity price per hour (Figure 23).

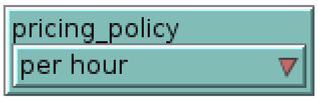

pricing_policy: per hour

**Figure 23** Pricing policy per hour

Pressing the setup button the agents/devices are created. The locate_devices button places them in the world in specific dimensions in the area. Every device has its own color and shape (Figure 24).

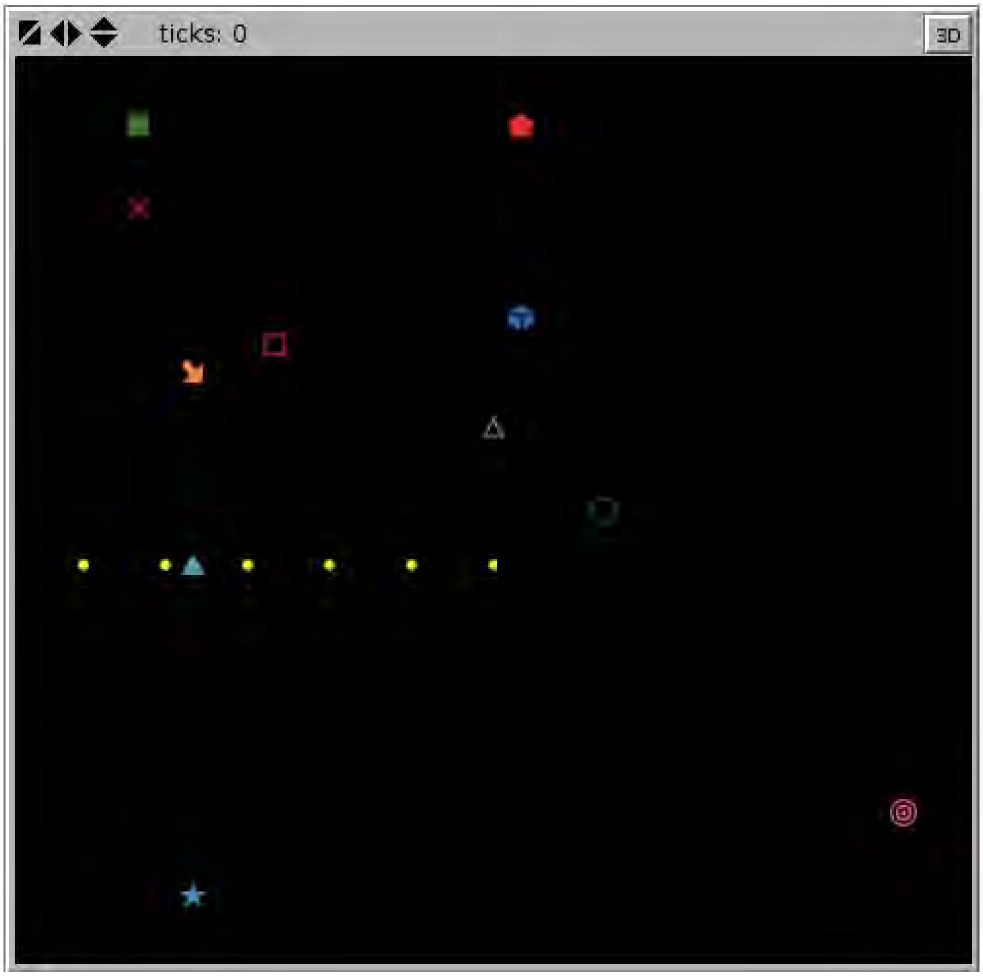


**Figure 24** Representation of devices in the World

Finally, the get_measurements button is the one that will give the final results. During the execution of the orders for the calculation of the final cost that the consumer will have to pay, we observe that in the world, the devices appear and disappear according to the functioning data that have been given (On or Off). Also, the user is able to check whenever he wants, the current consumption (KW) of the devices (Figure 25) and the current price as well (Figure 26).

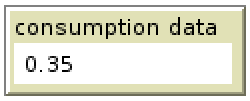

consumption data
0.35

**Figure 25** An example of the current consumption (kW) of the devices

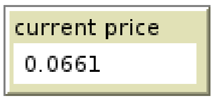

current price
0.0661

**Figure 26** An example of the current price of kWh

As soon as this procedure in the interface is over, the user has the potential to see the consumption of every type of device and the cost, in the plots which are created as well as the overall consumption (W) and the final cost (Figures 27-36).

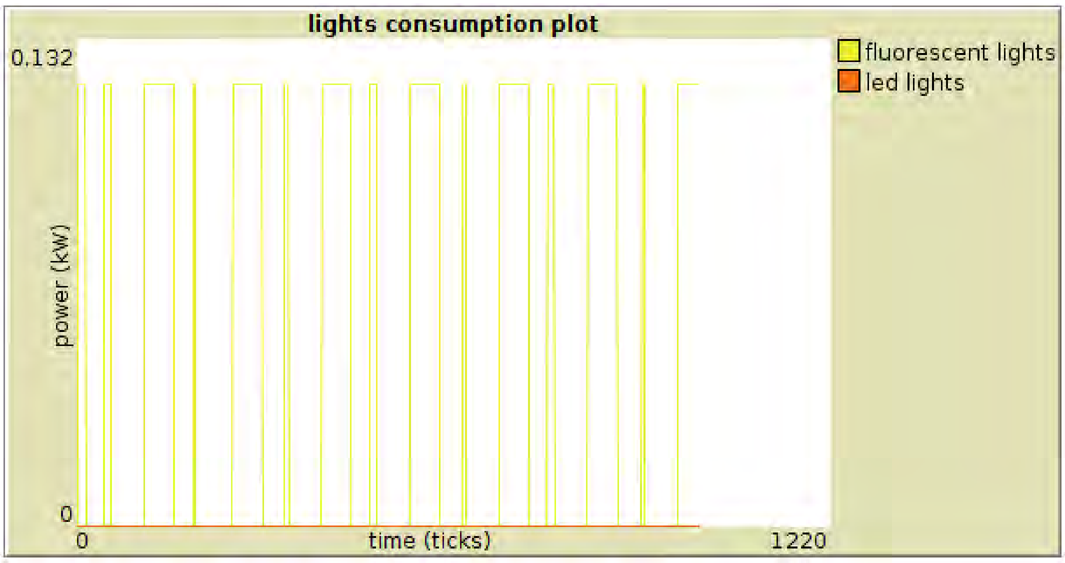


**Figure 27** Lights power consumption plot

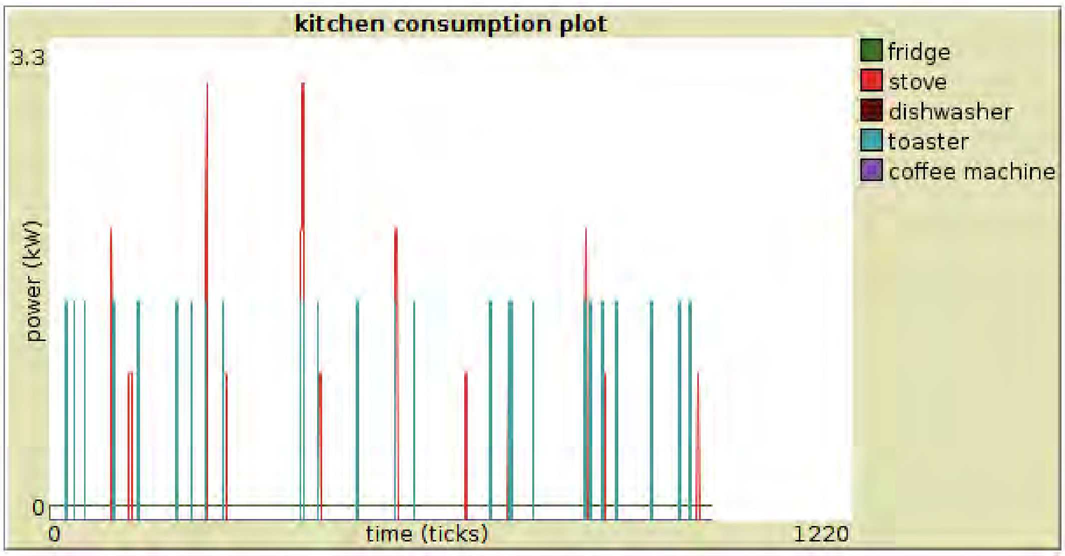


**Figure 28** Kitchen power consumption plot

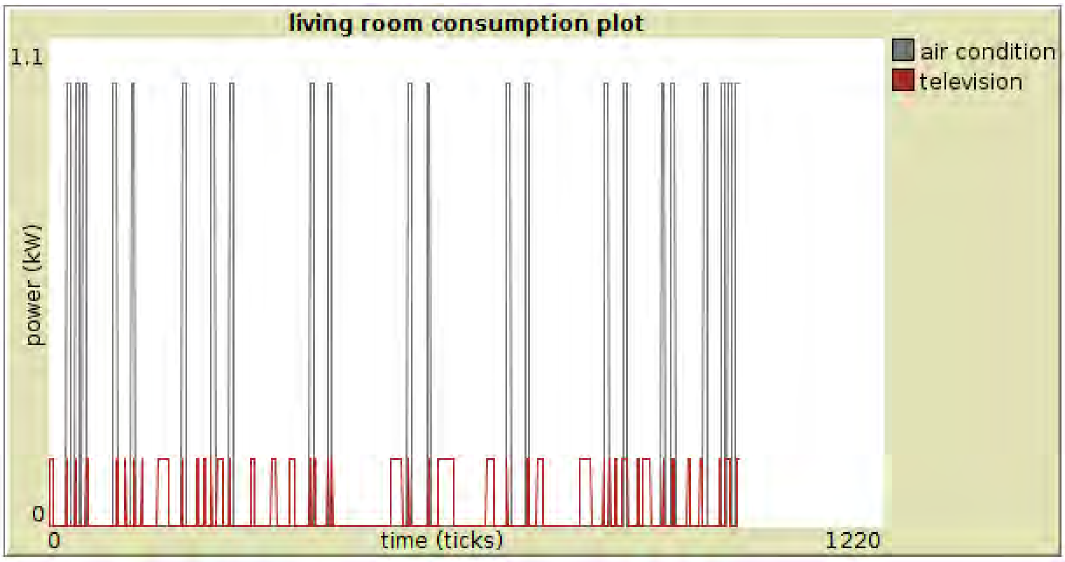


**Figure 29** Living Room power consumption plot

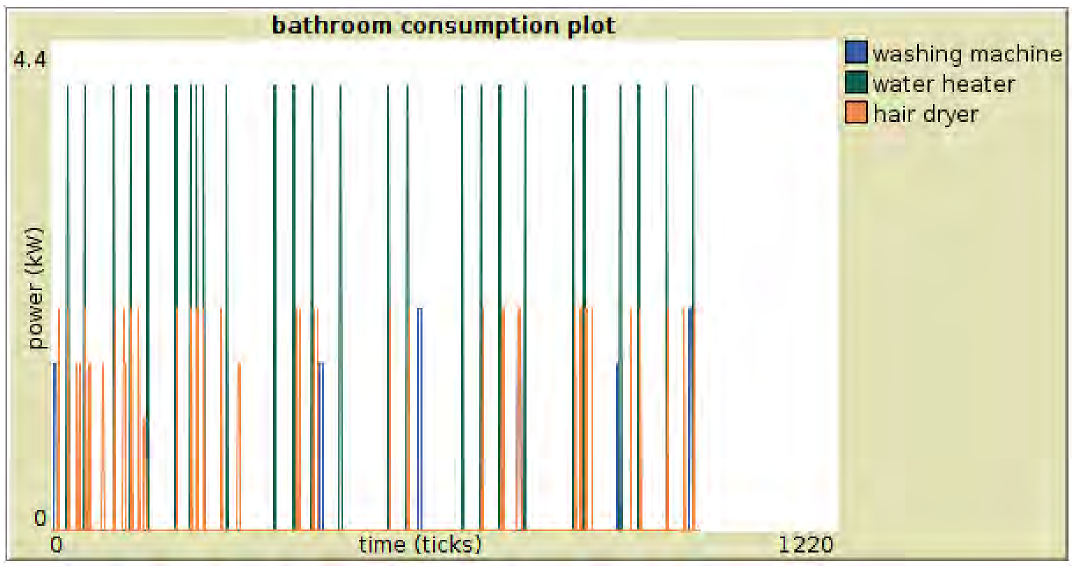


**Figure 30** Bathroom power consumption plot

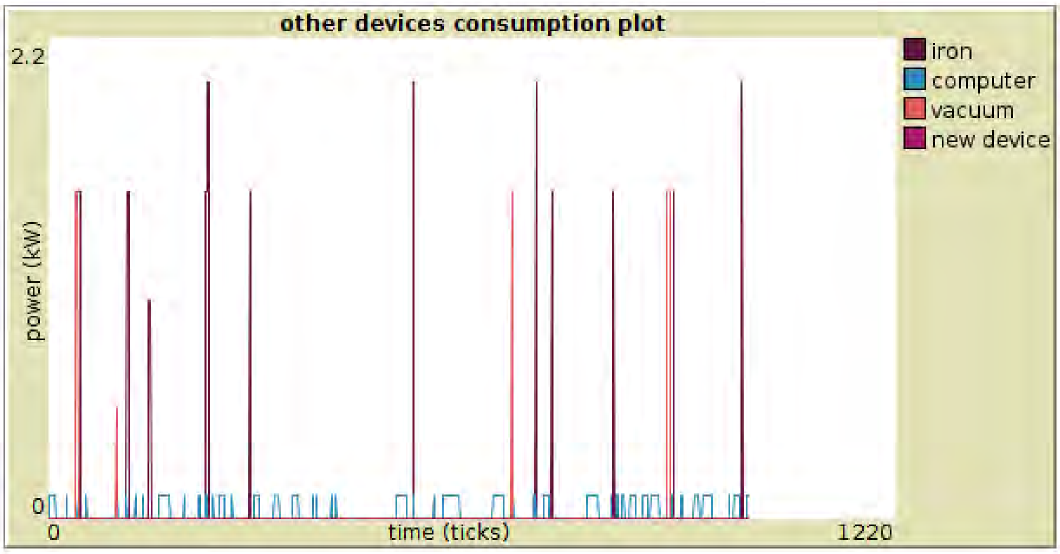


**Figure 31** Others Devices power consumption plot

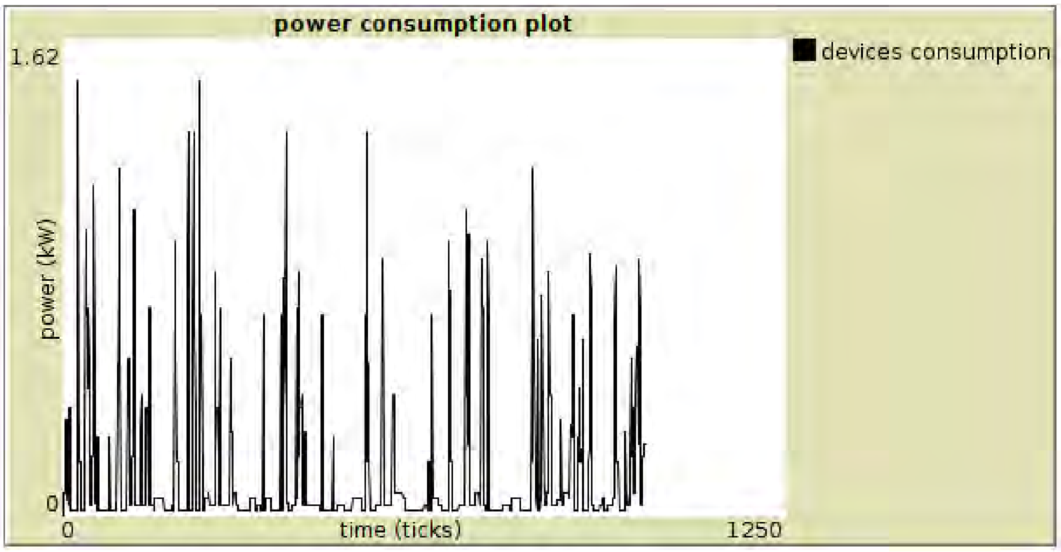


**Figure 32** Devices power consumption plot

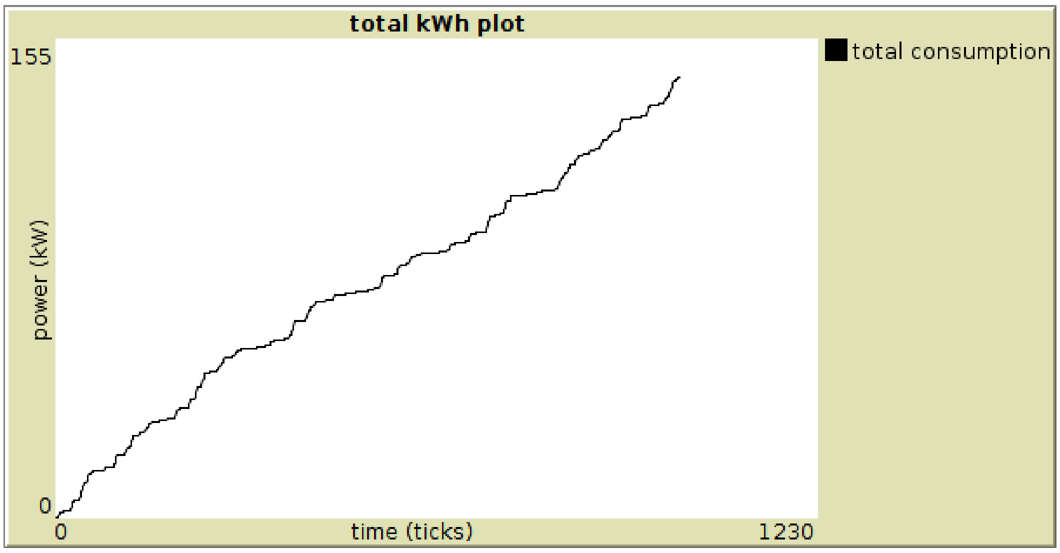


**Figure 33** Total kWh plot

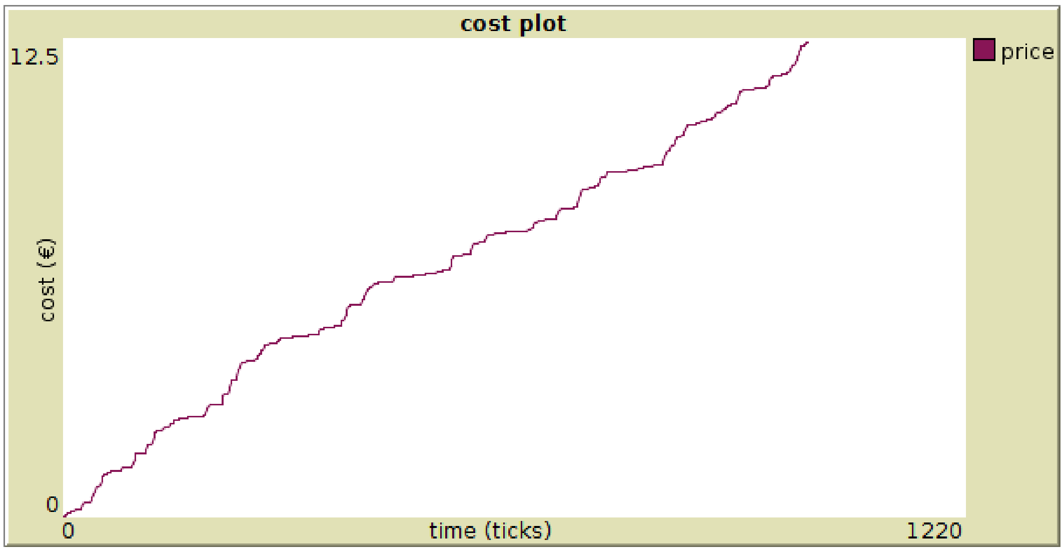
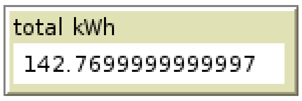
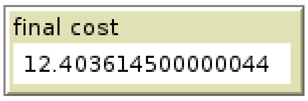


**Figure 34** Cost Plot


total kWh

142.7699999999997

**Figure 35** Total kW


final cost

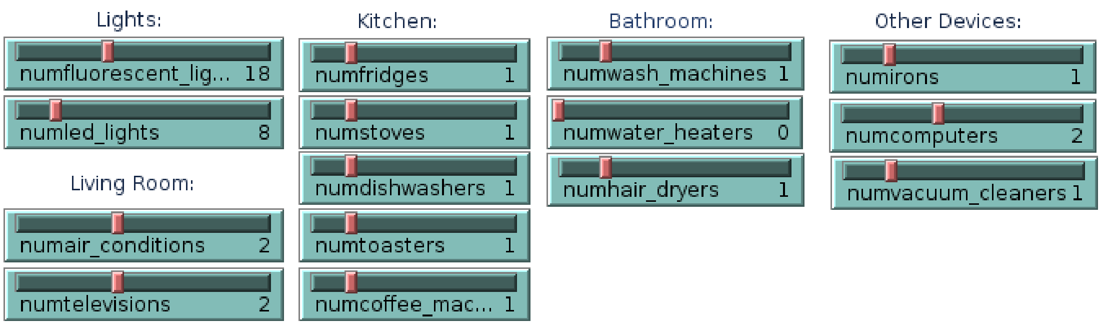
12.403614500000044

**Figure 36** Final Cost (€)

## 3.5.2 Daily life of a Family

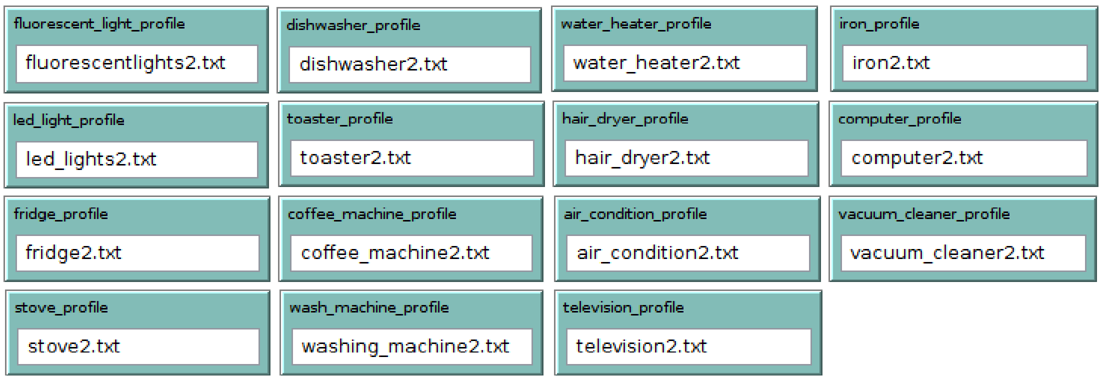
The second simulation example concerns a four-member family, in a period of 7 days in the autumn period.

We assume that such a house, for the specific example, includes the following devices (Figure 37):



| Lights: | Kitchen: | Bathroom: | Other Devices: |
|---|---|---|---|
| numfluorescent_lig... 18 | numfridges 1 | numwash_machines 1 | numirons 1 |
| numled_lights 8 | numstoves 1 | numwater_heaters 0 | numcomputers 2 |
| **Living Room:** | numdishwashers 1 | numhair_dryers 1 | numvacuum_cleaners 1 |
| numair_conditions 2 | numtoasters 1 | | |
| numtelevisions 2 | numcoffee_mac... 1 | | |

**Figure 37** Amount of devices that have been regulated with the sliders



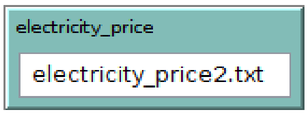
| | | | |
|---|---|---|---|
| fluorescent_light_profile: fluorescentlights2.txt | dishwasher_profile: dishwasher2.txt | water_heater_profile: water_heater2.txt | iron_profile: iron2.txt |
| led_light_profile: led_lights2.txt | toaster_profile: toaster2.txt | hair_dryer_profile: hair_dryer2.txt | computer_profile: computer2.txt |
| fridge_profile: fridge2.txt | coffee_machine_profile: coffee_machine2.txt | air_condition_profile: air_condition2.txt | vacuum_cleaner_profile: vacuum_cleaner2.txt |
| stove_profile: stove2.txt | wash_machine_profile: washing_machine2.txt | television_profile: television2.txt | |

**Figure 38** Functioning scenarios of devices

The scenarios for 7 days with measurements per 10 minutes for the function of each device have been created in text files (Figure 38). The measurement for each day starts at midnight. The scenarios created are based on the daily life of a family which consists of 4 members. For instance, a great usage of devices such as washing machine, oven and dish washer. The scenario is based on an autumn week and that is the reason that there is no usage of the air conditioning system, as well as the hypothesis that the house does not have a boiler for the heating of water.

Moreover, according to the domestic invoice the electricity price is defined as 0.0946 €/kWh.

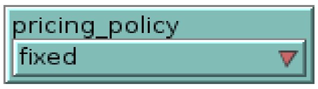
The text archive that includes the billing data is given also via the interface (Figure 39).


electricity_price

electricity_price2.txt

**Figure 39** Archive with the electricity prices

The pricing policy that has been chosen for this example concerns a fixed electricity price (Figure 40).


pricing_policy

fixed

**Figure 40** Pricing policy fixed

Next step, the setup button and then the locate_devices button for the creation of agents for the simulation of the devices in the world (Figure 41).

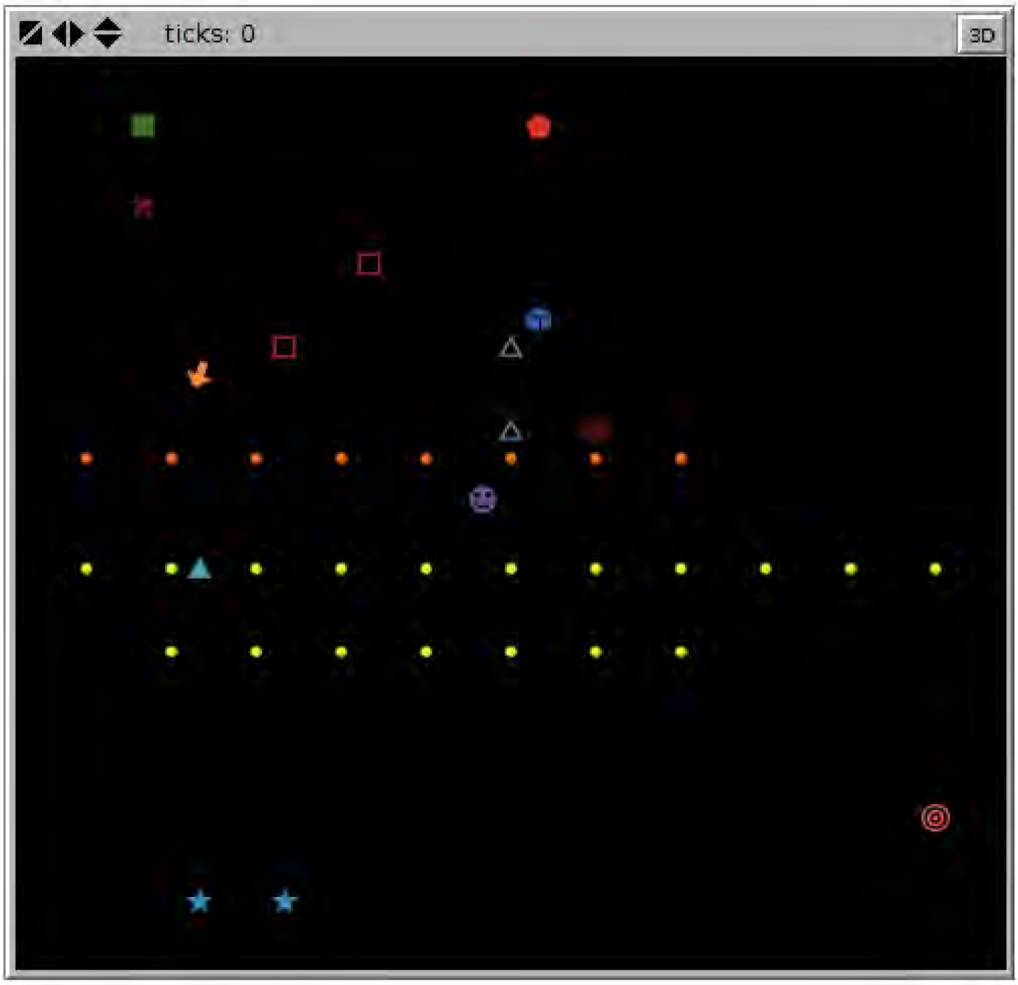


**Figure 41** Representation of devices in the World

Once more, the user is able to check whenever he wants, the current consumption (kW) of the devices (Figure 42) and the current price as well (Figure 43).

consumption data
1.8

**Figure 42** An example of the current consumption (kW) of the devices

current price
0.0946

**Figure 43** An example of the current price of kWh

The get_measurements button will create relevant plots in the interface, for the overall consumption and the cost right after the process of the data that had been given (Figures 44-53). Once more, it is observed that with the increase of the ticks in the world, the devices appear and disappear according to the functioning data that had been given (On or Off).

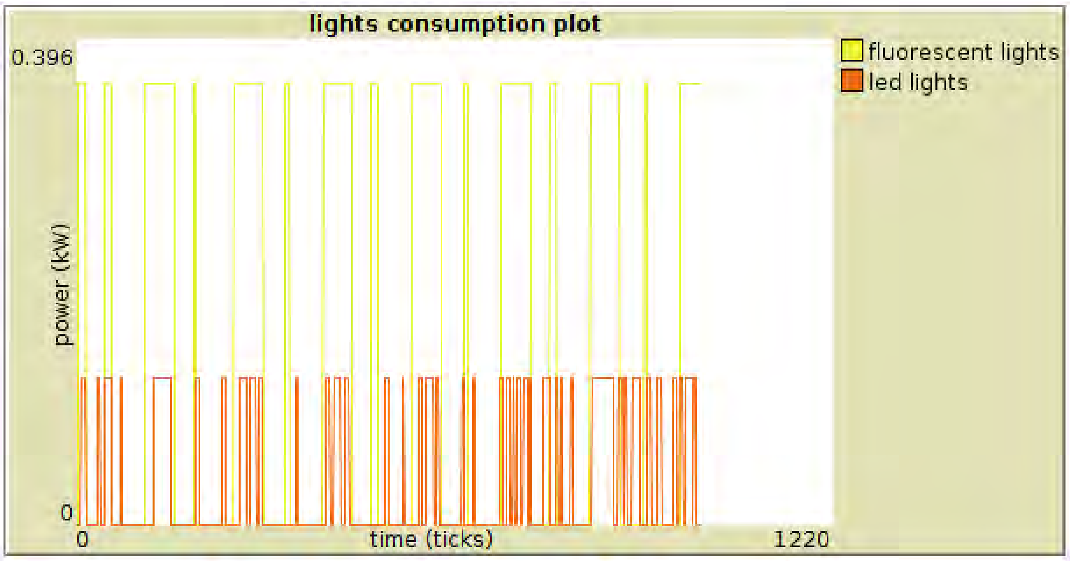


**Figure 44** Lights power consumption plot

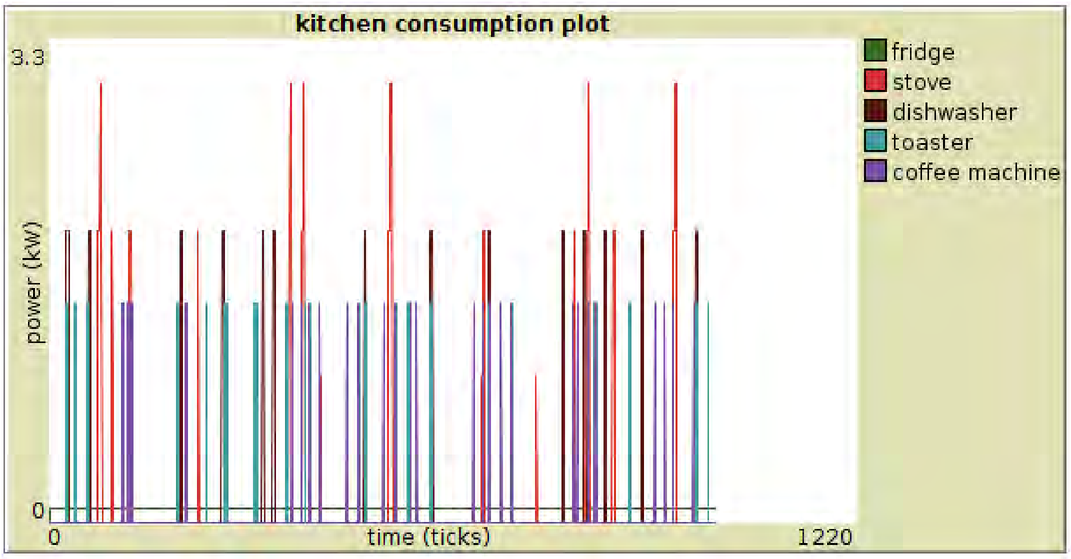


**Figure 45** Kitchen power consumption plot

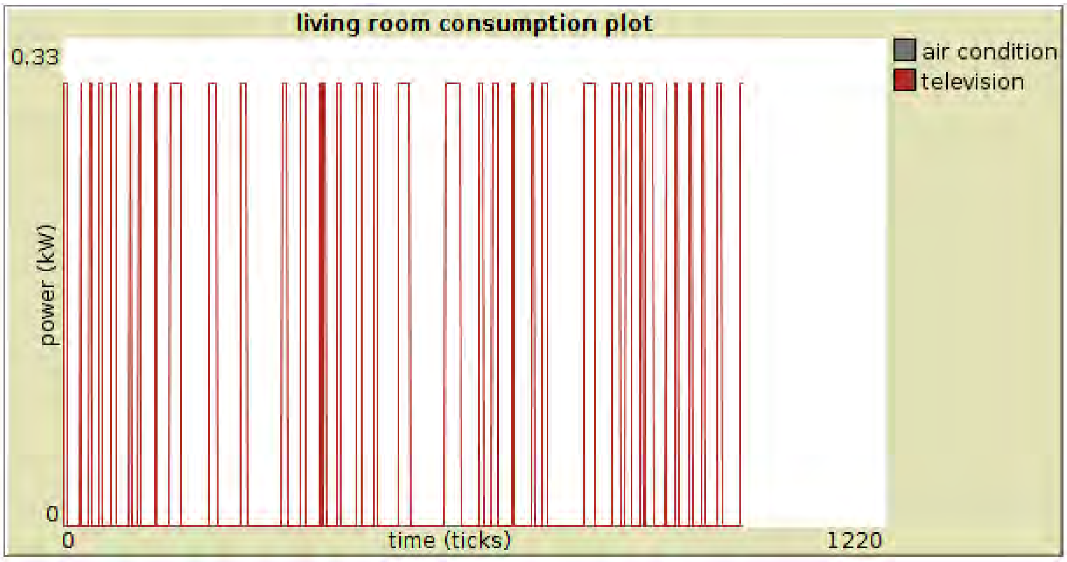


**Figure 46** Living Room power consumption plot

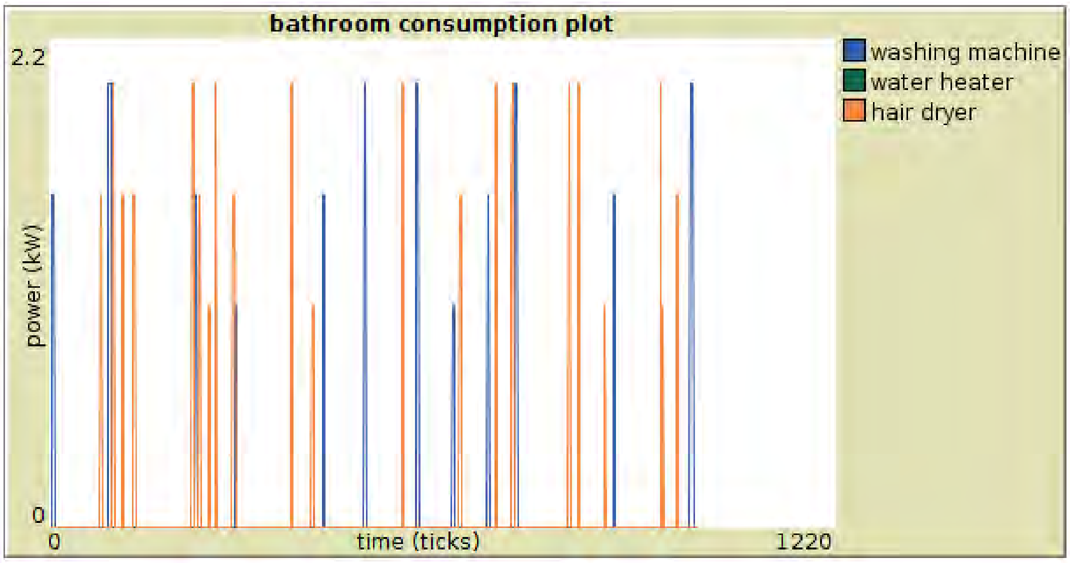


**Figure 47** Bathroom power consumption plot

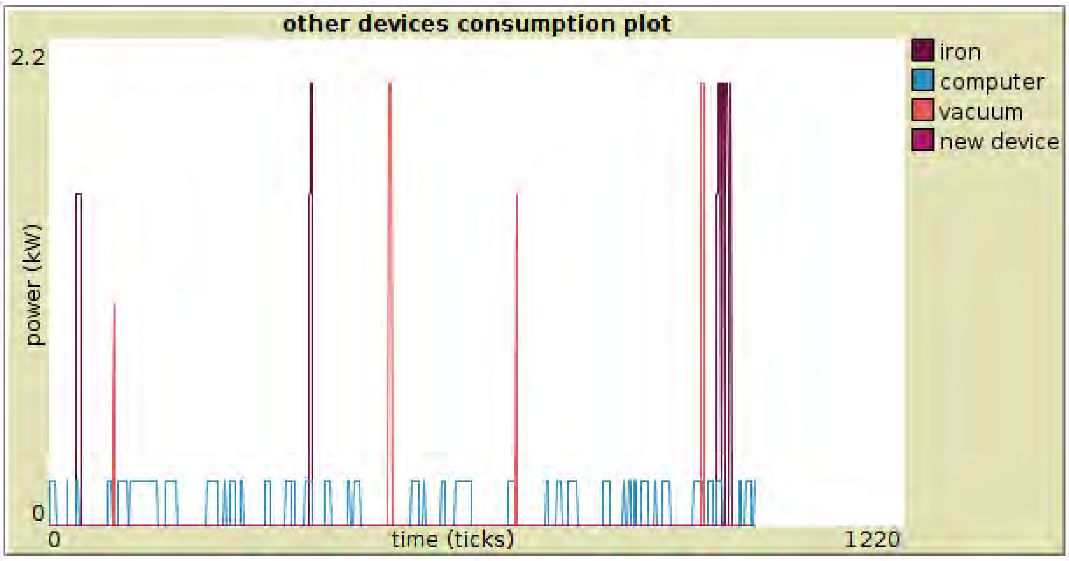


**Figure 48** Other Devices power consumption plot

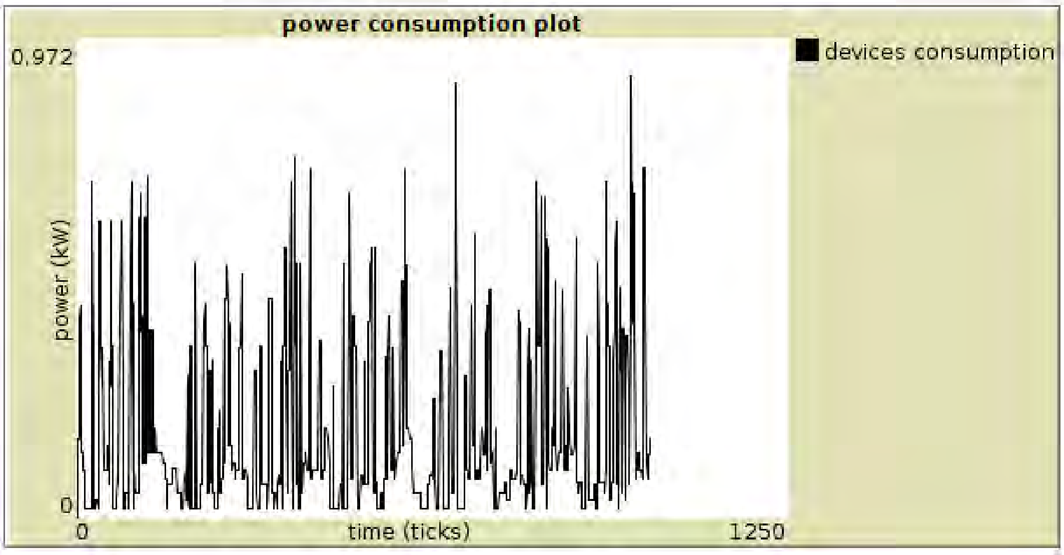


**Figure 49** Devices power consumption plot

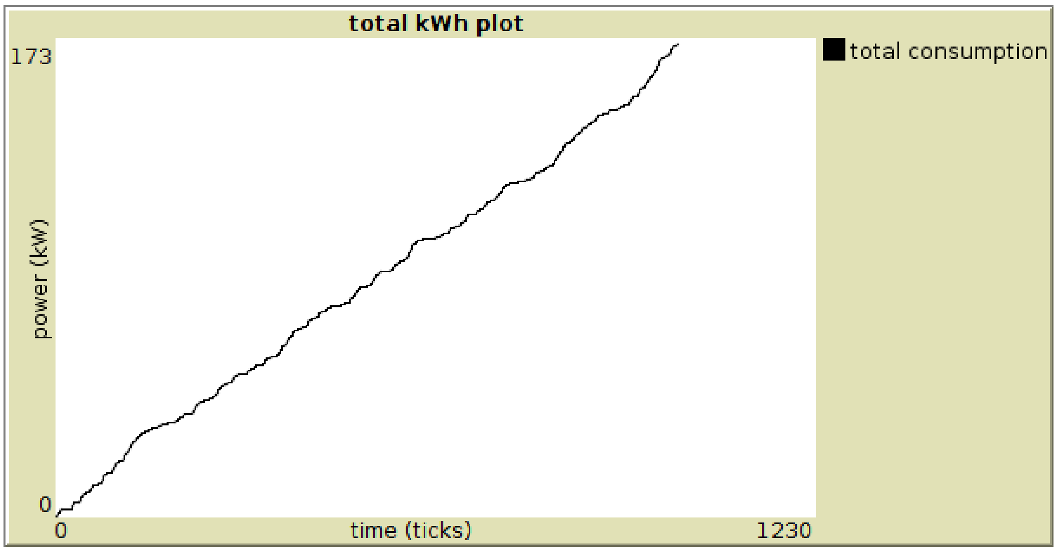


**Figure 50** Total kWh plot

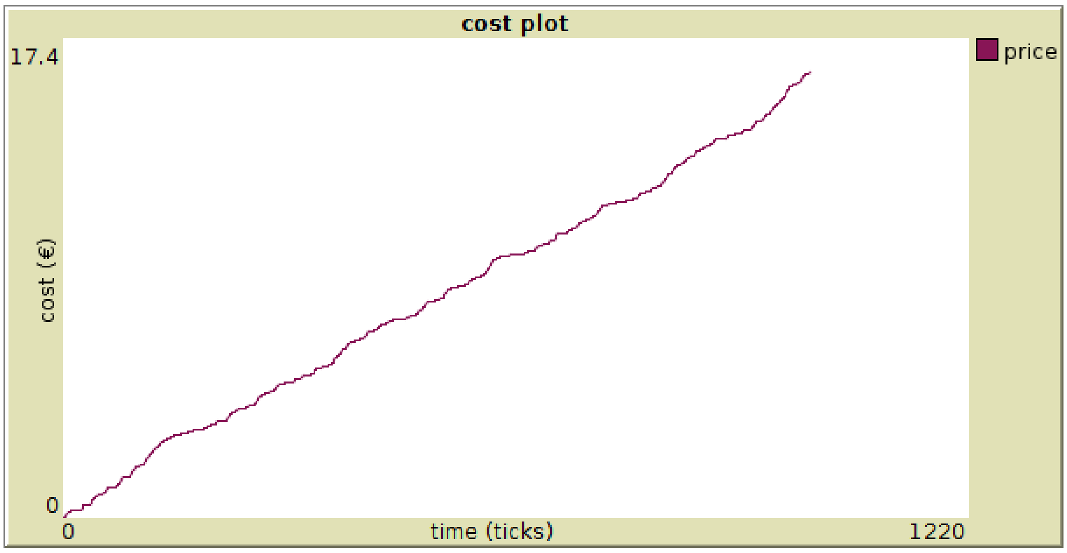


**Figure 51** Cost Plot

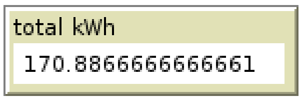

total kWh
170.8866666666661

**Figure 52** Total kWh

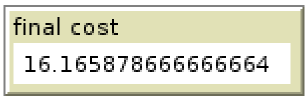

final cost
16.165878666666664

**Figure 53** Final Cost (€)

# 4. Conclusion

## 4.1 Summary

The control of electric power consuming, as well as the minimizing of cost are of the most important needs nowadays and that is why a huge progress has been observed in smart homes. On these grounds the actualization of the application this thesis was based. The target was to offer the user the potential to simulate their house environment by regulating the devices he has as well as to estimate the overall cost according to the consumption of each device and the functioning time. In this way, the user is able to create various operational scenarios of their devices and opt for the one that is the most cost-efficient to him and at the same time to respond to their needs and everyday habits.

## 4.2 Future Work

The fundamental aim of this thesis was the development of an application in NetLogo, which would allow the user to simulate their home environment with the devices they have, so as to be able to monitor the consumption of electric power and also the overall cost. Yet, several future studies for the progress of this application could take place.

Initially, more device categories could be created which in their turn would include a larger amount of other devices. The aim is the creation of an even more realistic simulation for the coverage of possible needs of the user, that we did not consider in first place at the design of the application.

In addition the application can easily be enhanced with new features, based on the same logic as before for new requirements of user. An interesting addition would be the possibility to read all the data the program needs from a csv file, for example with what devices the home is equipped, the function scenarios and then the program itself adjust those information to interface tab.

Moreover, for the estimation of the cost, the acceptance that the price of kWh can take one or two prices, a differentiation of the price during the night hours according to the current scenario has occurred. Nevertheless, at the domestic invoices that Public Power Corporation S.A. (PPC) offers for example as the provider of electricity, the charge varies according to the rank of the overall consumption. To wit, the price (€/kWh) increases as the amount of kWh increase. For instance, in such an invoice the charge for 0-2000kWh is 0.0946€/kWh, whereas for >2000kWh 0.10252€/kWh. At the same time, the cost is estimated without taking into account the rest of the charges in the bill (VAT, various other charges/credits, municipal charges, etc). Therefore, another future development of the application would be the modification of the estimation way of the cost for the creation of an effective model for predicting prices, so that the user could see the overall amount of money that they have to pay as it would appear in the electricity bill.

One more noteworthy improvement that has to be reported would be that the application could have the potential to manipulate the house devices and not only to control their consumption. A kind of intelligence to be created in the house based on the collected data and to have actuators that will be able to manipulate the whole home system. The actuators are designed so that they can make decisions on their own, when is required, according to the needs of the system and also the residents', based on the data that had been collected.

Simultaneously, in terms of this idea, that the application has to acquire knowledge on the daily life and needs of the user, another possible development could be the potential to create the application itself possible operational scenarios of the devices for the user's benefit. For example, for devices such as the washing machine, it could suggest which operational hours will bring about a lower cost and at the same time to have the potential to program the operation without the intervention of the user, as long as the user agrees with this scenario. On the contrary, on what concerns devices such as the iron or the oven it could be more flexible, to wit, to suggest once more scenarios that bring about the lower cost of operation but according to the needs and everyday life of the user as their involvement in this case is essential.

In conclusion, various operational scenarios of devices have been used for 7 days with measurements per 10 minutes. These scenarios are given to the interface in text files and were written by hand with assumptions for the user’s needs. Nevertheless, it would be rather useful to have a script on which the user could have the potential to regulate which hours of the day the devices operate and to what percentage (0-100%) and an archive to be created at once without the need for the user to create it every time.

# Bibliography

[1] http://cctvinstitute.co.uk/smart-home/

[2] http://www.iotevolutionworld.com/m2m/articles/376816-history-smart-homes.htm

[3] https://en.wikipedia.org/wiki/There_Will_Come_Soft_Rains_(short_story)

[4] http://mashable.com/2015/01/08/smart-home-tech-ces/

[5] Semantic-based Smart Homes: a Multi-AgentApproach by Giuseppe Loseto, Floriano Scioscia, Michele Ruta, Eugenio Di Sciascio

[6] Towards the Evolution of Smart Home Enviroments: A Survey by A. Bejarano, M. Jimeno, A. Salazar, P. Wightman

[7] A Multi-Agent-Based Intelligent Sensor and Actuator Network Design for Smart House and Home Automation by Qingquan Sun, Weihong Yu, Nikolai Kochurov, Qi Hao, Fei Hu

[8] Ανάπτυξη πλατφόρμας Πολυπρακτορικού Συστήματος για την λήψη αποφάσεων στο Χρηματιστήριο Ενέργειας by Χρυσόπουλος Αντώνιος

[9] ezHome – Σύστημα Προσομοίωσης και Ελέγχου Έξυπνου Σπιτιού με χρήση Πρακτόρων Λογισμικού by Σπανουδάκης Εμμανουήλ

[10] What is an Agent? Technical Report by James Ingham

[11] Advantages of MAS for the resolution of a power management problem in smart homes by S. Abras, S. Pesty, S. Ploix, M. Jacomino

[12] http://cctvinstitute.co.uk/iot/

[13] http://www.smithsonianmag.com/innovation/kevin-ashton-describes-the-Internet-of-things-180953749/

[14] The Internet of Things: A survey by Luigi Atzori, Antonio Iera, Giacomo Morabito

[15] Internet of Things (IoT): A Survey on Privacy Issues and Security by Vikas B O

[16] Security Architecture of the Internet of Things Oriented to Perceptual Layer by Weizhe Zhang, Baosheng Qu

[17] Research on the architecture of Internet of things by Miao Wu, Ting-Jie Lu, Fei-Yang Ling, Jing Sun, Hui-Ying Du

[18] Optimum Time of Use program proposal for Iranian Power Systems by H.

Aalami, G. Reza Yousefi
[19] http://www.iea.org/topics/electricity/subtopics/demandresponse/
[20] Review of barriers to the introduction of residential demand response: a case study in the Netherlands, by J. Weck, J. van Hoof and M. van Sark
[21] Assessment of Demand Response and Advanced Metering, Federal Energy Regulatory Commission
[22] https://energy.gov/oe/services/technology-development/smart-grid/demand-response
[23] http://ccl.northwestern.edu/NetLogo/docs/
[24] https://ccl.northwestern.edu/NetLogo/docs/faq.html
[25] https://ccl.northwestern.edu/NetLogo/docs/interface.html#tabs
[26] NetLogo: A Simple Environment for Modeling Complexity by Seth Tisueseth, Uri Wilensky
[27] A Survey of Agent Platforms, by K. Kravari, N. Bassiliades