Framework for Controlling Smart Homes

Miguel Lopes
Lisbon, Portugal
miguel.s.lopes@tecnico.ulisboa.pt

ABSTRACT

Nowadays, with the constant technological evolution, there are more intelligent devices and accessories capable of offering a better quality of life to the people, particularly in their houses. However, these solutions have some problems. Their focus is on the individual control of devices and not on the definition of global behaviors for the house involving multiple devices with multiple rules and conditions. The recent evolution of the Internet of Things has also been applied to homes and from there have come various products such as Amazon Echo and Apple Home, that offer sophistication as they allow voice commands. But, these solutions simply control devices directly, allowing them to be switched on, turned off or just apply a few simple configurations. In order to overcome these problems, we developed an application, based on DomoBus system, which allows us to offer a new degree of definition of complex and sophisticated behaviors for housing. With this you can create automation blocks that define generic behaviors. These behaviors can, then, be instantiated in concrete devices through the application developed. When the automation blocks are created a XML file is generated with all the information necessary, the instantiation data is also written to an XML file which data can be used by the DomoBus supervisor for applying the automation block in a concrete house. After reaching the last version of our application, we make some tests to see if the application achieved is useful and solve the problems identified in the other solutions. The results are presented in this document and allow us to confirm that the application generated the desires results.

Keywords
Internet of Things, DomoBus, automation block, behaviors

INTRODUCTION

The great technological evolution that has taken place in the last years, has lead to an increasing number of people that increases their comfort and security requirements, with that the home automation and the control of smart homes has undergone a great evolution in the last years. [1]

Home automation concept is more and more present in our lives as being a technology responsible for controlling all the resources of a house, involving requirements such as comfort, security and communications, with the objective to improve the way people live. The number of intelligent accessories that surround us increases day after day. With that the necessity to create systems to control that accessories increases too. However, these systems have their problems and disadvantages. These systems aim to control individual accessories and not in the definition of global behaviors to the house, that involves multiple accessories, with multiple rules and conditions. They only limited to control directly the accessories by switched them on, off or make some simple configurations.

In order to fulfill the necessities for controlling a smart home, we propose a new system for controlling smart homes, using the DomoBus system. Thus, we aim to have a set of behaviors, created by a user that knows the system, so that behaviors can be used by a common people applying them to a set of concrete devices in a concrete division of a house.

The main objective of these work is the possibility to define complex behaviors with generic automation blocks using the DomoBus system. It's intended to offer mechanisms that simplifies the definition of these behaviors, involving multiple devices and multiple conditions according to the characteristics of a dwelling. It should be possible contemplate some functionalities, such as, be possible to edit the definitions of an automation block already created and change the devices that are connected to the inputs and outputs. Thus, with this is possible to contemplate automations and personalize them, switch them on and off easily. For that is necessary define models of generics behaviors through automation blocks with generics inputs and outputs that can be instantiated with concrete devices in a concrete division of a dwelling.

To fulfill our objectives, we create an application, in JAVA, which allow us to create automation blocks with all the fields necessaries to generate behaviors. These fields are inputs, outputs, timers and a state machine. With this application is also possible to get all the information of a house, such as the devices of the house, how many floors and divisions the house have, and which are the devices of the house separated by divisions. After the creation of a automation block it is possible to instantiate the inputs and outputs with concrete devices. In the end of that a XML file is generated with the data of the automation block defined, and another XML file is defined with the data of the instantiation made.

With this system we have tried to solve the problems described above regarding the interfaces that control smart homes, as well to increase the ease with which automation is created in home automation, making the life of population more comfortable and simple.

The rost of this paper is organized as follows: in Section "Related Work", we provide a overall viewing of actual solutions for home automation; in Section "Solution

Description" we talk about our system and how with works; in Section "Implementation" we provide information about how we do our system; in Section "Tests and Results" we provide some examples about how our system answer to some behaviors and what are the results of this tests; in the last section, "Conclusion" we give a glimpse of what we made and about the improves that we can make in the future, to make our system better.

RELATED WORK

Much work has been done regarding the home automation. With the evolution of the technology and the Internet of Things the companies try to go along that evolution and create news solutions (accessories and applications) to control smart homes. In the next sub-sections, we give a glimpse about these solutions and how if they can answer positively to define complex behaviors.

X10, KNX and LonWorks

This tree systems are very similar. The X10 system is a home automation protocol that uses the electrical network of an house to send basic commands to devices, for example, switch on/off an device or increase/decrease the bright of a lamp. It's a system simple and intuitive to use, have a small cost to apply and could be used by very accessories. In addition to being able to define these basic operations it's possible to define behaviors through specific controllers. With these controllers and through a interface it's possible to receive schedules and schedule devices to turn on and off.

The KNX system is a technology that can be applied to houses and van control various devices such as, illumination devices, heaters, blinds and security devices. This technology can use various communication means and needs a own software to configure the devices, this software is the ETS [2]. This solution has a elevate cost in comparation with X10, however it gives more functionalities. In other hand it installation is complex and is not intended to be configured by a common user without knowledge in KNX area, it needs a technic to make the installation. To and behaviors to devices the ETS system is needed and has a additional cost to KNX. Only a user familiarized with ETS can make the configuration of behaviors for devices. It's a non flexible system, because if the user wants to make a small alteration on devices behaviors needs to use ETS and call to a technician to made the changes.

The LonWorks technology is a home automation system to control smart homes, that communicate between a network such as electrical network, radiofrequencies or optic fiber. It's a system equivalent to KNX and in general has the same problems.

IFTTT

The IFTTT (if this, then that) it's a framework that can be applied to all devices connected to a network [3]. It's available as mobile application or website. The main objective is to a user automatize is applications or websites and automatize is devices if they are compatible with the system. Actually this system supports 360 services like Android and iOS sytems and applications like Facebook, Google Drive or Youtube [3] [4].

To use the IFTTT system the user needs to log in the application and generate the automations that he wants, these automations are based on actions that triggers other actions for example: Switch on a lamp if the user receives a notification from Facebook. Thus, it's possible to control the behaviors by the application and personalize these behaviors since devices and services are supported by the system [3].

Amazon Echo

The Amazon Echo is a device that uses voice recognition commands to execute the actions asked by the users and control smart devices [5]. This device has integrated microphones which does the voice recognition. The Echo device is always turned on waiting to receive instructions, but it's disabled, to enable it the user needs to spell the word "Alexa" at beginning of ever order. The voice processing is done on Amazon cloud it's not done locally. When the device is active, the users can ask it "How is the weather today?" or "What is the traffic in my street?" or just "Switch on the lights of my room". The device answers the questions by consulting a huge data base connected to the internet [5].

The device allow the user to control some devices, individually. It's not possible to define behaviors that are done because previous behaviors, for example, it's not possible to open the blinds if the luminosity of some division is under 10%. To be possible to do these actions the Echo needs to connect to IFTTT online service. This is a way of defining these behaviors but is add-on to Echo and it's needed to follow a guide to implement these behaviors.

The Amazon stores on his database the voice commands used by users in order to answer quickly on future orders if they are the same that the asked before. If the Amazon stores only the voice commands it will be no problem, but the Amazon also stores some voice detection moments before Echo being enable. Thus, in the database could also be stored some conversations of the user completely apart of voice commands to devices and this could put at risk the security of the user, being also a case of invasion of privacy [6]. The device also stores his localization in order to being more efficient to answer the user requests, but stores this localization is also a problem of privacy.

Discussion

After the analysis of this solution we can verify that the definition of complex behaviors it's not possible, even though Echo being a recent solution. To realize any action the user needs to ask the device as if it were a remote command. Being one system that uses voice recognition

could have some problems with that, because sometimes de voice recognition may not work as well.

Overall Balance

Taking into account the researching done, it is possible to verify that none of the solutions described presents all the functionalities of the solution that its pretends to develop.

In the explored solutions it is possible to define behaviors, but these are basics and simple, it is not possible to define rules by a combinatory logic using states machines. Some of the explored solutions are not flexible, because they are not portable and it can not be applied to all devices.

DESCRIPTION OF THE SOLUTION

The definition of automations for smart homes, where these are consistent and easy to use by the users is no easy task, and as we can see by the analysis made on last section there are a few solutions that explore this capacity. Considering this, we decided to develop this strand through automation blocks. All the development is inserted in the context of DomoBus system. The automation blocks will help the users to define automations to their homes that can be applied in other dwellings.

In the next sub-section is described the development method to do this automation blocks, the necessary requirements for it correctly work and the functionalities that the automation blocks offer.

DomoBus System

The DomoBus is a system of home automation, composed by modules of control and modules of supervision where they communicate with each other between a network and allow to interact and cooperate with each other. In this system, all the devices connected are considered generic entities characterized by a set of properties. At any time, the system could consult the value of these properties or edit the value of them [7].The DomoBus has a specification language defined in XML. Through this specification it is possible to define a structure of a house and the devices characteristics that compose the house, it allows us to know the follow information:

- Number of floors;
- Number of divisions;
- All devices;
- Devices type;
- Devices separated by division;
- Identification of the users of the dwelling.

Control Level

In this level, the modules used are typically small boards based in microprocessors, which that connect directly to sensors and actuators. The sensors are all the devices that could answer to a stimulus like a movement or a bright alteration. The actuators are all the elements of the system which can interact physically like switches. These modules can control various devices, each module could have different function. One can control 11 switches and other can control 5 lamps. Therefore, can be as modulo control modules as you want in a DomoBus system [8].

Supervision Level

In this level, the supervision modules are responsible for the control and supervision of the system. These modules are typically Raspberry Pi, PCs, tablets or smartphones, and there can be as many supervision modules as you want. These modules receive information by control modules, process this information according to the programmed rules and to the respective behavior, and then send to the control modules the appropriated commands to produce the correct action to the devices.

Supervisor

The Supervisor has a preponderant role in this system. In a dwelling, can exist since one supervisor to control all home to one supervisor per division, where this depends on the number of devices of a dwelling. Each device is associated to a supervisor to whom transmits all information such as when change is state, turn on, turn off. Whenever supervisor wants it can access to devices information. At this moment the supervisor supports tree mechanisms that can executes actions, they are:

- Rules – They are like “if, then, else” rules and can be as complex as we want.
- Time sequences – Here timers are used to initiate actions. When a time sequence is enable a period of time can be waited in order to execute a list of actions after this period.
- Schedules – With this mechanism the system is able to program cyclic actions, or actions that occur in specific instants. The schedules can correspond to actions that occurs, repeatedly, diary, weekly, monthly or yearly. In case to be diary it is possible to define the period of the day when user wants to execute the action. When the actions are weekly it’s possible to select the days that the user wants to execute the action. An example of use for this mechanism is the garden watering.

At this moment the supervisor is programmed used a set of commands, that can be read by a file or inserted interactively on the command line. The mechanisms associate to the supervisor can be edited, deleted, enable or disabled at any time.

Although all these advantages the installation of and programing of a DomoBus System proves to be complex. It is necessary to define the house specifications in a XML file, configure all the sensors and actuators and associate them to a supervisor. Adding to that, it’s needed to program all the automations wanted to the devices, and this process in

particularly can be more complex. These tasks are not within reach for all the common users of a DomoBus system. Thus, to avoid all these complexity, we propose that the system starts to contemplate **automation blocks**. In this approach a user has access to automation blocks previously defined and only need to apply them to the devices that he wants, simplified this process a lot, because the automation blocks only need to be defined one time. The automation blocks are generic and can be applied in concrete at any dwelling.

Solution Proposal

The existing mechanisms in supervisor allow us to define rules and generate behaviors wanted by the users. It can be used a combinatorial logic (the final result only depends in the actual state of the system) or a sequential logic, where the results depends on the previous events. Although this mechanism being very powerful, as it allows to define various behaviors to a dwelling, its hard to use, especially when we want to implement complex behaviors with multiple sensors and actuators.

With that, the propose solution answer to two problems that exists on DomoBus system actually, these problems are:

- Complexity on the installation and programing of a DomoBus system to a specific dwelling;
- It takes a long time and is difficulty to do the programing process when are involved complex behaviors with various actuators and sensors.

An automation block is composed by inputs, outputs and timers, in which each as the following characteristics:

- Inputs – It could exist multiple inputs in an automation block. In these inputs will be instantiated the sensors and actuators. Each input only can be off one type, for example if an input is for a movement sensor, in this input only can be instantiated movement sensors. It can be instantiated to an input as much sensors or actuators as wanted. The inputs communicate to the system when a property of one connect device is changed, for example if a sensor detected movement it should send this information to the system.
- Timers – An automation block can include one or more timers. The timers are used for behaviors that need to count periods of time.
- Outputs – In the outputs we connect the devices on which we intend to act. It can exist multiple outputs on a automation block. If the behavior is to switch on the lights on one division of the house, all the lamps can be connected to the output if they are all the same type.

To be possible to define behaviors through an automation block its necessary to contemplate some logic, in order to generate some rules and that the results are subsequently passed to the outputs. There are two types of logic implemented in the automation blocks:

1. Combinatory logic, implemented by the DomoBus system, used through “if, then, else” conditions.
2. Sequential logic implemented by the automation blocks. It allows to define complex behaviors through the use of state machines. It this logic it is possible to react in function of the actual states of the inputs devices and react by previous states in order to create sophisticated behaviors.

The automation blocks will be created according to the behavior wanted by the user, taking in consideration the inputs and the outputs. So, each automation block will be for one specific behavior, so it can be a lot of automation blocks. These blocks are generic, once they are not for a concrete house, but to any dwelling. These automation blocks are defined through a application, developed in JAVA. To create these blocks the user, need to define the behaviors that we wants for a programmer familiarized with the system create the automation block.

In order to create a automation block through the application developed, a set of steps need to be followed:

1. First its necessary to create a automation block by giving to the application a name for that automation block.
2. Then, the inputs are created, in which are defined the type of inputs and all the components needed to an input such as the logic comparator, the compare value (if needed) and the name of the input.
3. After that the timers can be created, only if the behavior need timers.
4. Posteriorly the outputs are created. On here the type of the output is defined.
5. Finally, the internal logic is created. If the logic wanted is combinatory only one state is created, and all the condition are done on this state. If the logic pretended are sequential, can exist various states and the correspondent condition form one state to another. Each state has a transition condition, when each condition has one or more actions where is choose the output and the value applied to it. An action can also be applied to a timer (enable or disable) or it can also be applied to change from one state to another.

After a automation block creation, the programmer goes to the instantiation stage. On this a XML file with the house specification is read by the application and the devices of the house are ready to be connected to the inputs and outputs.

As the DomoBus system works with XML files, the XML is also choosing to store the information for the automation blocks. All the information such as inputs, outpus, timers or states are stored in this XML document and are able to be consulted to be applied on the supervisor. The instantiation

also produces a XML file with the information of the connected devices to the inputs and outputs. This file could be used by the supervisor also.

IMPLEMETATION

The application built for create automation blocks was developed with JAVA language, using the JFrame package in order to help to have a graphic interface with a simple design. For the application has the knowledge of the dwelling specifications we made the parse of the XML file that are used to specify any dwelling. The application is sub-divided into 5 pages, such as:

- Devices;
- Devices by division;
- Automation Block Editor;
- Instantiation;
- House Model;

In the next sub-sections these pages/functionalities are described.

At this stage the main objective was to has a functional prototype in order to show the concept of creating automation blocks, the focus is not on the visual aspect, but on the functionalities.

Devices

On this page of the application the devices read by the parse of the application are presented in a text area by pressing a button. The parse of the XML file some dwelling is made by using the tags. The devices have some data that are stored in the class od the devices, this data is: Access Level; Device Name; Device Type; Device Division.

There is an array of devices created to be easier to access the information when needed. With this functionality we can see what the devices are present in one dwelling.

Device by Division

At the second page of the application the devices are showed the devices according to their divisions. The user must select the division that he wants to see the devices, for that, the divisions are showed in a ComboBox (Drop down list), and the user select the division that he wants, and the devices of this division are printed in one text area.

The devices as all the information in the XML that specifies the house. The divisions of the house is written to the ComboBox by scrolling the array of the divisions of the house. Then when the division are selected, the array of the devices are scrolled and if the device is from the division select it is printed in the text area.

House Model

At this page the characteristics of the dwelling are showed up, these characteristics are:

- Name of the house and the identification number. These data are collected by the parse of the XML file referring to the house specification and are showed in a text box.
- Name of the floors existing in the house. This data is collected by the same way, the XML file are parsed, the data is saved in the Floors class in an array. Then the array is scrolled, and the data is printed.
- Name of the divisions and name of the floor of these divisions. The parse of the XML file put the data wanted in the respective classes, then the application access to that classes, scroll the array of the divisions and print the divisions and the respective floor in the text area.

This page is only to see the model of the house, when can be see the floors, divisions and the house identification.

Automation Block Editor

At this page the automation blocks can be created, this page is more complex in terms of programing code, once is needed to store all the data inserted by the user that are creating the automation block, edit that information, if needed, and at the end create the XML file with the automation block specification. It is needed to prevent errors when the block is being created.

The main steps to create one automation block through the application are described in the next sub-sections. As mentioned above, we need to create the automation block, the inputs, timer's outputs and the states.

Automation Block creation

In this page, at the beginning only one button is showed, this button allow us to create a automation block.At the beginning of the creation of the automation block it is needed to create it by giving a name to the block. When the block is created the name is added to a ComboBox. It can be select to edit (complete the definition by adding inputs, outputs, timer's and states). There is an array that has all the automation blocks created, this array is of the type of the class automation block.

After creating the automation block it can be edited, and when the button to edit the automation block is clicked, a new menu appears where it can be done some functionalities. In that menu it is possible to create or edit inputs created, it's possible to create or edit outputs already created, it's possible to create new timer´s or edit them, and it's possible to create states, add transitions to that state or add actions to that transitions. At this menu is also showed how many inputs

does the actual block have, how many outputs, states and timer's.

In this menu we need to create a input first.

Inputs creation

When the button to create a input is pressed a new menu appears. This menu has 9 entrances, as showed on Figure 1 they are:

- Name – In this field is inserted the name of the input.
- Value Type – There are two types of value types, enumerated or scalar. The enumerated are used to sensors that only give a set of values (yes/no, detected/non-detected), the scalar is used to sensors that give various values such as a temperature sensor, it can gives values form 0 to 50 for example.
- Value Type Name – In this field can be selected how the value type is, in the case of the Figure 1, the percentage is selected.

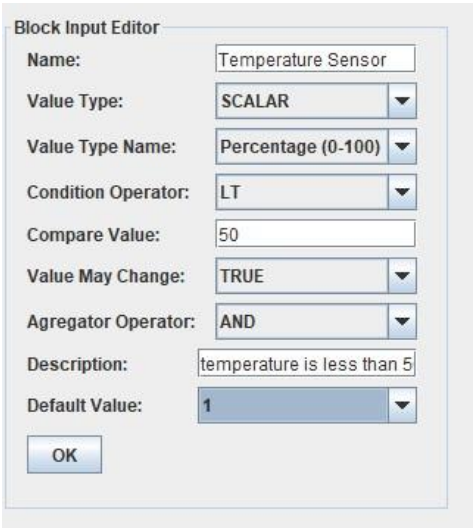

Figure 1: Menu for creating or editing an input, with an example of a Temperature Sensor.

- Condition Operator and Compare Value – These fields will make the action to be executed. In this case, if the temperature is LT (Less than) 50 some action should be done. In other words, here is defined the behavior of the input.
- Aggregator Operator – This field is used if is instantiated more than one device to the input and combine its values. For the example showed on Figure 1 if there are 3 temperature sensors and we choose the aggregator operator the input will intend to execute an action if all these 3 sensors have the value less than 50.
- Description – At this field a small description of the behavior of the input should be write.
- Default Value – In this field the value 0 or 1 should be select if we want the input to be disable or enable, respectively.

If we want to edit these values we can choose the input by his name and press the button edit.

Outputs

After creating the inputs, the outputs can be created and then, after the button for create a new output being clicked a new menu appears. There are 3 fields to complete, they are:

- Name – In this field the name of the generic output should be written.
- Value Type and Value Type Name – These fields are similar to the ones at the inputs definition. But, here it is needed to select the type of the output. If we want to turn on a brightness lamp we should select the scalar type and the value type as a percentage.

Then a new output is created, if we want to edit its values we should press the edit button and select the output that we want to edit by selecting the right one in the ComboBox. As the outputs is being created they are being added to the ComboBox. The outputs created are also added to the outputs array like the inputs. These arrays are into the automation block array.

Timer's

If a timer is needed, we can create one by pressing the button "Create New" referenced to the timer. After pressing this button, a new menu appears where we can insert the timer name and the time associated to them. It is also possible to edit that time by selecting the timer that we want to edit in the ComboBox.

States

The creation of states is the last step to finish the automation block creation. As mentioned above each state has one transition and each transition can have multiple actions. Each position of the block array has associated a array of states and each state has associated a array of transitions and each transition has associated a array of actions.

To create a state it is needed to click the button "Create New" associated to the state. After this button being clicked a new menu appears. At this menu we can add transitions to the state, add actions to the transitions created or edit the transitions and the actions already created. To create a state it only needs to write the state name. After that it is needed to create the transition for that state, for that we need to complete some fields :

- Name – The name of the transition should be inserted on this field.
- Logic Expression – In this field a logic expression should be added to the transition. A logic expression is composed by a logic operator, an input and a value 0 or 1 (false or true). If in this state we want to make an action when the value if the input is true we should put in this field, for example, "I0 EQ 1".

After create a transition we should add an action to that transition, this action will be executed when the logic expression off the transition is verified.

An action has 4 fields, they are:

- Name – In this field the action name should be inserted.
- Action Type – In this field is selected the where we want to execute the action, it could be to an output, to a timer(start/cancel) or to another state (change state).
- Action Object – This field change according to the choice made on "Action Type" field. If a output has selected in this field will appears all the outputs created. If a timer is selected in this field appear all timers created, the same applies for states.
- Value – In this field we can choose the value to apply to the output if it is a scalar we can define the value according to the output type, or just select on or off if it is a enumerated for example. In this field we can also choose the state that we can change with that action.

After the state is complete, the automation block definition is finished, and we should confirm that. When we want to confirm a button "Close App" should be clicked and then a XML file with the automation block data is created. This file as all the information such as: Block Name; Block inputs data like value type, aggregator operands, compare value and the other fields mentioned above; Block outputs with all these values; Block state with all the states created, all the transitions and actions associated to them.

When the automation block is created the concrete devices can be instantiated to them.

Instantiation

At this page of the application the user can choose which automation block wants to instantiate, by selecting the block name in the ComboBox. When the automation block is selected two menus appear: one to instantiate the inputs; one to instantiate the outputs.

To instantiate the inputs, the user should select what is the input associated to the automation block selected that he wants to connect the concrete devices. After selecting the input by one ComboBox a new menu appears. In the new menu there are tree ComboBoxes:

- One to choose the division that the user wants to instantiate the input and consequently the behavior.
- The other ComboBox show all the devices of the chosen division, and the user should select what device he wants to connect to the select input.
- The last one show the properties of that device. The user should select what property he wants to use in that input. For example, if the device is a movement sensor the property select should be the movement.

After these selections are made the user should click the "Connect" button and the instantiation is done. One input can have as much devices connected has the user wants, but the type of the devices should be the same of the defined in the input.

To instantiate outputs, the same logic is used. The user needs to select the output to be instantiated, then select the division, the device and the property. One output can be connected to a lot of devices of the same type.

After instantiating all the devices to the inputs and outputs a XML file is created. This file has the instantiation data getting by scrolling the arrays where the data was stored. With this file, the devices and properties are mentioned by number such as the blocks identification. This is done for the file is simply used by the supervisor and apply this to a concrete house.

EVALUATION

In order to test our system, we run try to implement some behaviors and instantiate them with concrete devices with a dwelling specification file. The more complex behavior tested is presented in Figure 2.

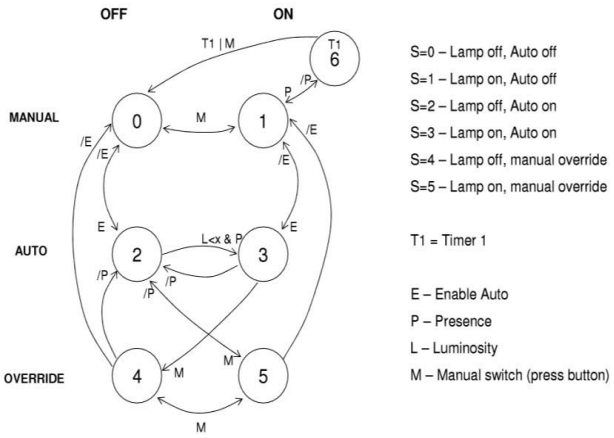

Figure 2: State machine done for one complex example using 7 states and 19 transitions.

This example could be applied in any division of a determined dwelling. It consists in the controlling the illumination of one division using two modes:

- Automatic Mode – The illumination of the division is enable taking into account the result from two sensors, one presence sensor and one luminosity sensor.
- Manual Mode – The illumination is enable when a manual switch is pressed. This mode override to the automatic mode, when a switch is pressed to turn off the light, the illumination is disable despite the results of the automatic mode.

The inputs are these sensors motioned above and can be connected to them as much concrete devices as wanted. The outputs are the lamps.

As we can see in the Figure 2 the state machine presented has 7 states and 19 transitions between them. The manual mode and the automatic mode is represented such as the states when the illumination is ON and OFF. To explain the operation of this state machine some states behavior will be explain:

- State 0 – At this state the illumination is disable and the used mode is the manual. As showed in Figure 2 only one transition go from this state to state 1. This transition is nominated by "M", when a manual switch is pressed the illumination is turned on and the actual state is the state 1.
- State 1- At this state the mode is the manual and the illumination is on. This state has two transitions to other states. One transition is when a switch is pressed, the lights are turned off and the actual state will be the state 0. In this state, state 1, if the light is turned on is because someone is at the division. So the other transition is when the movement sensor stops to detect presence, and the actual state is the state 6. At this state the lights are on and is waited a period of time (T1). If during this period a presence is detected the actual state will be state 1, but if its not detected any presence and the time expires, the illumination are disable and the actual state is the state 0.
- State 2 – At this state the automatic mode is enable and the lights are off. According to this state transitions, can occur tree behaviors: The automatic mode can be disable and the actual state will be state 0; A manual switch can be pressed and override the automatic mode, turn on the lights and the actual state will be state 5; The last behavior was done when the luminosity sensor has a value minor than X and the movement sensor detected presence, if that occurs the lights are turned on and the actual state will be the state 3.

With this example we can prove that our system can produce complex behaviors. The movement sensors and luminosity sensors can be instantiating with concrete devices, a XML file for the automation block and for the instantiation data will be created as well. This example take time to do, but when it is done it can be applied to any dwelling at any divisions with these inputs and outputs.

CONCLUSION

Nowadays the automations to dwellings are being more and more. People in general like these automations because they offer comfort, security and some flexibility in what they offer. However, not all these automations can offer the option to define complex behaviors to a house. Taking these into account and after various searching we don't find any solution that offer the degree of functionality pretended. To fulfill these restriction, in this paper we propose an approach that use automations blocks to define behaviors as much complex as wanted, that can use states machines and timer's.

The system realized allow to create automation blocks that can be, afterwards, applied as many times as wanted in the dwellings that we want. These blocks are flexible, because their inputs could be with no device connected, with one device connected or multiple devices connected of the same type.

Thus, to prove the solution and simplify the generation and using of automation blocks a application was developed in JAVA which allows to read an XML file with a specification of an house including its devices and allow to create a automation block through a application that helps to create this block without making mistakes. The definition of an automation block is a complex task, so its assumed that who uses the application has knowledge about the concept of automation blocks and its functionalities. At the end we want that the blocks be simply to use. This definition is complex and hard to do, but after that it is simply to use the blocks and to instatiate them to concrete devices.

In conclusion, the evaluation tests made to the system, prove that our solution satisfies the defined objectives. A very complex example was tested that involves 7 states and 19 transition and the produce results allow us to show that our system gave support to the initial objectives. In the future it is intending to evaluate the system with some people with previous knowledge of an automation block to use the system and create a automation block and applied it to a concrete home.

REFERENCES

[1] V. Miori e D. Russo, “Domotic evolution towards the Iot - International Conference on Advanced Information Networking and Applications Workshops,” 2014.

[2] “ETS5: one tool for ALL media,” [Online]. Available: https://www.knx.org/media/docs/Flyers/ETS5-Flyer/ETS5-Flyer_en.pdf. [Acedido em 12 October 2017].

[3] E. Betters, “What is IFTTT and how does it work?,” Pocket-Lint, [Online]. Available: http://www.pocket-lint.com/news/130082-what-is-ifttt-and-how-does-it-work. [Acedido em October 2017].

[4] “IFTTT Services,” IFTTT, [Online]. Available: https://ifttt.com/search/services . [Acedido em 11 October 2017].

[5] P. Dempsey, “TEARDOWN CONSUMER ELECTRONICS, Burnt the Fire Phone, Amazon plays a cautious game with its new smart speaker,” 201.

[6] J. Glascock, “Police want Amazon Echo to help solve a murder,” 2016. [Online]. Available: https://www.csmonitor.com/USA/Justice/2016/1229/Police-want-Amazon-Echo-to-help-solve-a-murder.-Should-it. [Acedido em March 2017].

[7] R. Nunes, “Domotics/Home Automation, Slides da unidade curricular de Ambientes Inteligentes do Instituto Superior Técnico, Campus Alameda,” Lisboa, 2016.

[8] R. Nunes, “DomoBus - A new Approach to Home Automation,” Instituto Superior Técnico - Lisboa, 2003.