Internship Report

Development of a controller implementing the energy-planning phase of Triana using the Flexible Power Application Infrastructure

Stergios Dagioglou
February 3-May 2, 2014

Student name: Stergios Dagioglou
Student number: 1347632

Name of company:
Dutch: *(Nederlandse Organisatie voor) Toegepast Natuurwetenschappelijk Onderzoek - TNO*
English: (*Netherlands Organisation for) Applied Scientific Research*

This internship was conducted during the period February 3rd – May 2nd 2014 at the facilities of TNO located in Groningen, the Netherlands and within the department of "Business Information Services – BIS".
A presentation on this internship was held at TNO, Groningen on Tuesday May 6th 2014.

Supervisor names: Wilco Wijbrandi, Dennis Krukkert

Supervisors at the University of Twente: Hermen Toersche, Johann Hurink

This internship is part of the student's thesis at the University of Twente, Faculty of Electrical Engineering, Mathematics and Computer Science (Dutch: EWI, Elektrotechniek, Wiskunde en Informatica), Department of Computer Science - CAES group (Computer Architecture for Embedded Systems) / Department of Applied Mathematics - DMMP group (Discrete Mathematics and Mathematical Programming).

Contents

Introduction – Problem Description

One of the challenges that electrical utilities face is to integrate renewable energy sources at large scale and also serve the demand load which increases substantially. An opportunity to administer that challenge is the development of technologies that would lead to the achievement of a flatter consumption demand profile by shifting the consumption from periods of high consumption to periods of lower consumption. This would decrease the cost of electricity and would also increase the reliability level regarding an uninterrupted supply of electrical power. The scientific area that comprises technologies which aim to the aforementioned shifting of the energy demand profile has been characterized by the term *Demand Side Management* (DSM). DSM is one of the aspects of the more general technological field that is developed the last years and which is described as *Smart Grid.* The smart grid is a system which uses modern technologies to create more efficient energy systems compared to the present ones. The smart grid usually combines decentralized energy production (often from renewable energy sources), smart metering, information technology, energy storage devices, load and generation monitoring and control and other modern technologies to produce and distribute electrical and thermal energy in the most efficient way possible [1].

Triana is a DSM methodology developed by the University of Twente. Its purpose is "*to investigate the possibilities of using ICT control systems to change the energy profiles of buildings. By properly changing the energy profile of buildings, the current electricity supply chain can be more efficient ".* [2] Triana consists of three different phases: Forecasting, Planning, Real-time control. These phases will be explained in more detail in the next pages.

Apart from Triana, a large number of other DSM methodologies have been developed and proposed by various institutions. The use of a DSM methodology by a consumer usually involves the installation of an Energy Management System (EMS) which includes software and hardware components that interact with the devices of a household. As the number of DSM methodologies increases, a need arose for the development of a common platform that would make it easier for the end user to change from a DSM system, installed to his/her home, to another. The *FlexiblePower Alliance Network* (FAN) developed an infrastructure to serve this need. The infrastructure is called *Flexible Power Application Infrastructure (FPAI)* and its aim is "*to create an interoperable platform that is able to connect to a variety of appliances and support a host of DSM approaches. This way the EMS hardware does not need to be changed when a consumer switches from one service to another*." [3]

The DSM methodology *Powermatcher* was chosen to be the first system to be implemented on top of FPAI. The research question of the current internship is whether an implementation of the Planning phase of Triana is possible using some the FPAI platform. The method followed to answer that question is the development of a software component that executes the Planning phase of Triana using FPAI followed by a series of simulations to test if the results of the algorithm are the desired-expected.

The first chapter of this report is dedicated to the explanation of how the subject of this internship is related to the master programme Sustainable Energy Technology, followed by the student. The chapters 2 and 3 give an overview of the technologies of Triana and FPAI. Chapter 4 presents the planning algorithm of Triana and which tools of FPAI are used to implement this algorithm. In chapter 5 a series of graphs is shown to examine if the results of the simulations run, are satisfying or not. Finally chapter 6 contains comments and recommendations about the research done during this internship and about further research that can be done related to this internship subject.

1) Relation of this internship to sustainability

Technology and sustainable development

The importance of developing sustainable technologies nowadays is profound due to a series of problems, related to human activities on the planet, like the increase of CO_2 emissions. In order for a technological development to be characterized as sustainable, it has to contribute to all dimensions of sustainability [Figure 1]: It has to be 1) environmentally responsible, 2) economically viable for both the producer of the technology and the users and 3) it must support the functioning of a society in general by e.g. providing employment seats and/or fulfilling a need of the society. In the next paragraphs it is made clear how the DSM technologies and generally Smart Grid technologies are related to the three sustainability dimensions.

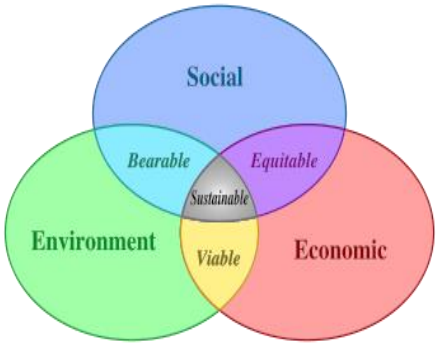


Figure 1: The three dimensions of sustainability,
Source: Course of Technology and Sustainable Development-UTwente, Lecture slides

Contribution to environmental aspects

CO_2 emission reduction

Electricity and heat generation and the transportation sector are the largest CO_2 emission producers [Figure 2].

As it is shown later in this report, the introduction of DSM technologies can lead to a larger integration of renewable energy sources and to a more efficient exploitation of them. Reducing the CO_2 emissions by decreasing the electricity produced by fossil fuels is a central environmental and energy-policy goal worldwide and the integration of renewables plays a decisive role towards this direction.

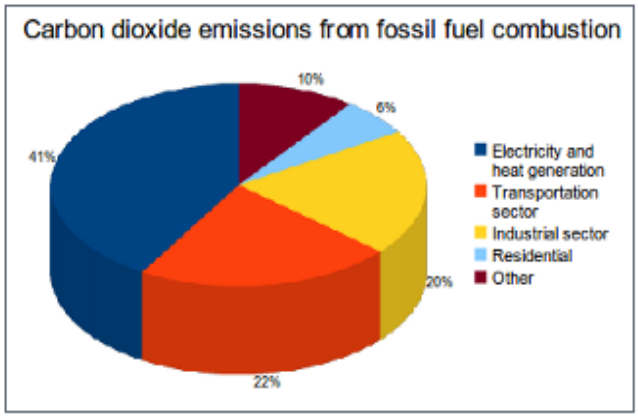


Figure 2: Source: CO2 Emissions from Fuel Combustion (2012), International Energy Agency

Moreover, a larger integration of renewables and the introduction of Electric Vehicles (EVs) will reduce the CO_2 emissions produced by transportation means. Another promising concept related to DSM and electric vehicles is Vehicle-to-Grid (V2G), according to which vehicles charge their batteries in low consumption-price hours and could offer it back to the grid when they are parked in peak hours. The substitution of heaters that use fossil fuels with heat pumps using electricity coming from sustainable sources is another way to reduce CO_2 emissions.

Energy consumption decrease by increasing the energy efficiency

Specific Smart Grid systems aim at the maximum possible exploitation of electricity produced locally mostly because of the economic benefits that bring to the end-users and also because most of this energy usually comes for renewable sources. By using energy produced at a local level we reduce the energy wasted due to transportation and distribution losses. These losses can also be decreased by a smarter control of the voltage levels and thus by eliminating the need of electric utilities to provide excessive voltage [4].

Contribution to financial aspects

Reducing the energy production and consumption cost

One important target of DSM systems is to reduce the peak electricity demand. During peak demand hours, energy sources are used that are more expensive compared to the sources used to cover the base and intermediate demand; thus decreasing peak demand directly saves money to electric utilities as less use of peaker plants is done. On its turn, this should have a positive impact to the consumer's electricity bill. DSM methodologies like Triana have also as a local goal the cost minimization regarding the energy consumed during the operation of a device.

The reduction of the energy cost can also be considered as contribution to the societal need for cheaper and thus affordable for more people, electricity.

Contribution to societal aspects

Creating new employment seats

Apart from the jobs related directly with the development of Smart Grids, like people working in the ICT sector or companies working in the smart meter industry and researchers, there is a significant number of jobs created indirectly. That includes installation technicians or people working in the renewable energy sector as a consequence of their growing integration.

Finally, it is reasonable that all benefits posed in this chapter, will increase with a parallel increase of the integration of DSM systems. FPAI is expected to play an important role towards the deployment of DSM systems in large scale as it provides an intermediate platform that makes communication between different devices and DSM systems possible.

2) Description of Triana

2.1) The structure and the three steps of Triana

In [5] a definition of Triana is given. According to that definition the goal of Triana is "*to monitor, control and optimize the domestic import/export pattern of electricity and to reach objectives which may incorporate local but also global goals*". A local goal could be the use of locally produced electricity and global goal could be to shift the peak demand load.

The term "local" could refer to the level of a neighborhood or a single building or a device and the term "global" could refer to a country, a city or a part of them. The necessity for communication between the local nodes (e.g. a building) and the global planner (e.g. a Distribution System Operator) lead to a tree structure the root of which is placed the global planner and the bottom-leaves of which are the local node controllers [Figure 3]. Information related to the electricity price flows from the global planner to the local nodes and information related to the electricity consumption profile of a device is sent from the devices to the global planner.

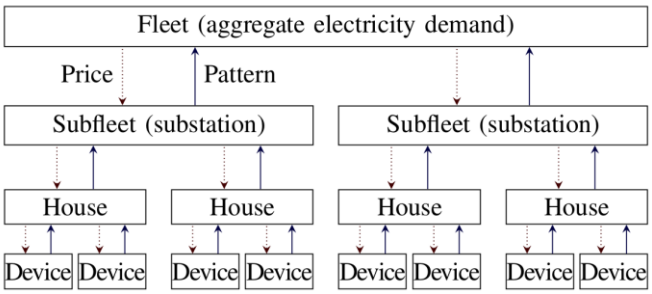


Figure 3: The tree structure of Triana

The three steps of Triana are: offline local forecasting, offline global planning and real-time local control.

During the first step, a forecast is made regarding different parameters related to the energy consumption of a device for a specific time horizon. This information determines the scheduling freedom of a device – the extent in which the consumption profile of a device can be shifted in time or altered in order to achieve a certain global goal. An example of a global objective could be to achieve a certain global consumption profile. This goal and the prediction done for every device in the first step, are the inputs for the second step – planning. During that step, the energy consumption of a device for every interval of a time horizon is defined, based on the global goal and the minimum energy cost for the device. If during the operation of a device based on the planning, an unacceptable mismatch is observed between the global plan and the actual given global energy profile, a new (re)planning phase begins, this time using forecasts

resulting from data obtained from the recent history of every device or from weather predictions for the next hours (if e.g. a prediction must be made for a heat producing device). In the end the solution that leads to the closest result regarding the planned goal is chosen or even a new (more conservative) global objective might be formed.

2.2) A comparison between Triana and Powermatcher

The *Powermatcher* is a Smart Grid technology developed by the Energy Centre of the Netherlands (ECN) and TNO.

Powermatcher is a real-time control mechanism aiming to balance the supply and demand (*SDM-Supply and Demand Matching*) in electricity grids which include a large number of *Distributed Energy Resources (DER)*. An auctioneer agent is responsible for calculating a balance price and distribute it to the devices-resources via a concentrator agent which connects a number of devices to the unique auctioneer agent [Figure 4]. The equilibrium price is found from the auctioneer agents after it has received a bid from every device aggregated by the concentrator agent. The price is formed based on the bids but is also based on an objective set by an objective agent.
An important difference between Powermatcher and Triana is that the first does not include an explicit forecasting and a planning process.

Powermatcher was FAN's launching application compatible with the FPAI.

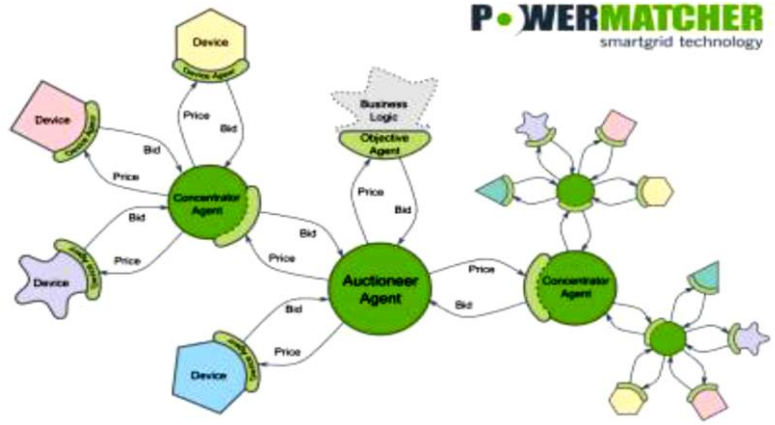


Figure 4: The structure of Powermatcher [6]

While a direct comparison between Triana and Powermatcher is not available yet, in [7] a comparative analysis between Triana and Intelligator is made. Intelligator is a system based on Powermatcher and developed by the Flemish Institute for Technological Research (VITO). Among others, simulations were run to see how Triana and Intelligator alter the aggregated demand of a number of houses that include loads of different nature, regarding their flexibility. The simulations showed that both methodologies have almost the same ability to reduce the peak consumption and Triana seemed to create a more even consumption profile, which is depicted in the next comparison graphs [Figures 5 and 6]. Other comparisons in [7] referred to the cost reduction from transportation and distribution losses and to CO_2 emission reduction.

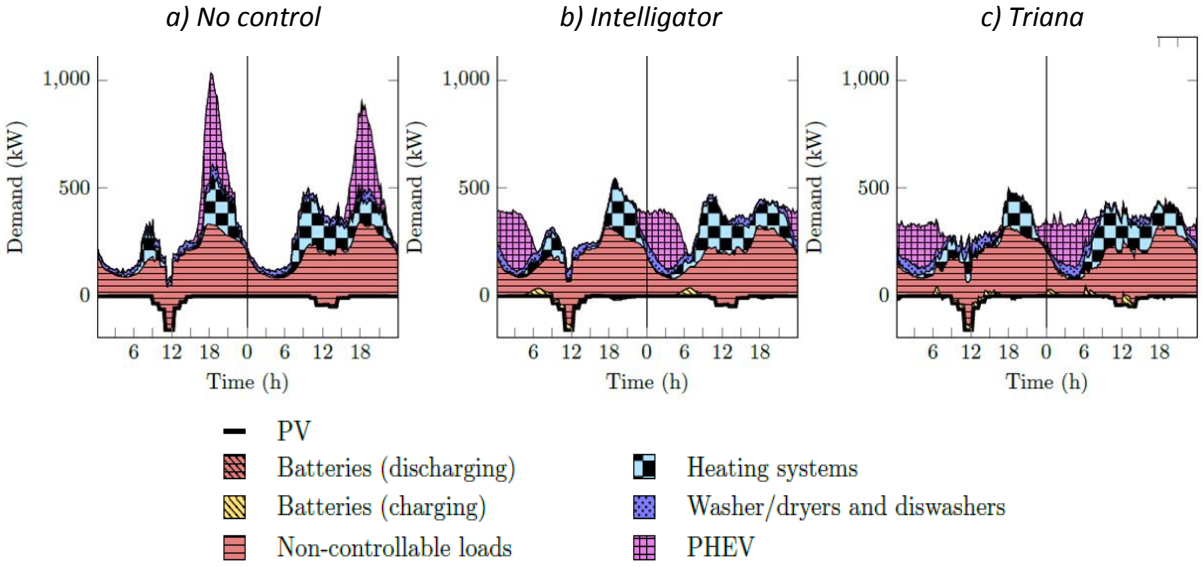


Figure 5: An aggregated energy profile with a) no control, b) control using Intelligator and c) control using Triana [7]

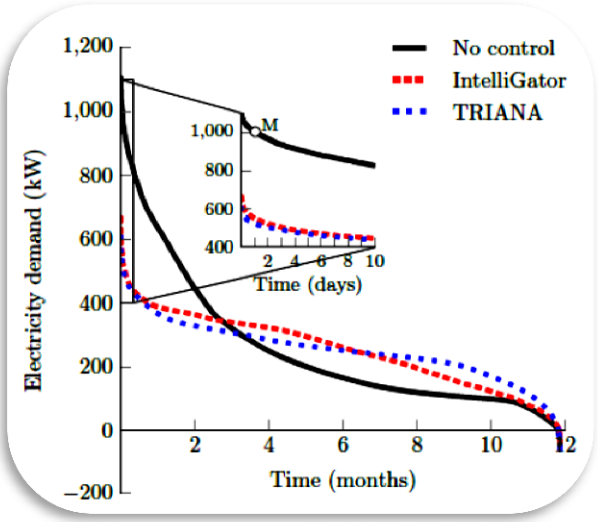


Figure 6: The load duration curves deriving from the figure 5 [7]

3) Flexible Power Application Infrastructure (FPAI)

The *Resource Abstraction Interface (RAI)* and the *Resource Abstraction Layer (RAL)*

As mentioned in the introduction, FPAI aims to the interoperability between energy applications on one side and devices on the other side. To achieve this interoperability, two corresponding abstraction levels have been developed: the *Resource Abstraction Interface (RAI)* and *Resource Abstraction Layer (RAL)*.

On its turn, the *RAL* consists of two different components specified for every device (a device is distinguished by the other devices via a unique identifier that is given): the *Resource Manager (RM)* and the *Resource Driver (RD)*. The *RM* is the connection link between the devices and an energy application. It receives data by the *RD* related to the operation of a device such as energy consumption, temperature etc. and also information regarding the preferences of the end – users. These inputs are translated by the *RM* to the energetic flexibility represented by the *Control Space* of a device. The RAI makes a separation of the devices to five categories according to the type of the flexibility these devices can offer [Figure 7]. Also for Triana a different algorithm has been deployed to exploit the flexibility offered by the different device categories.

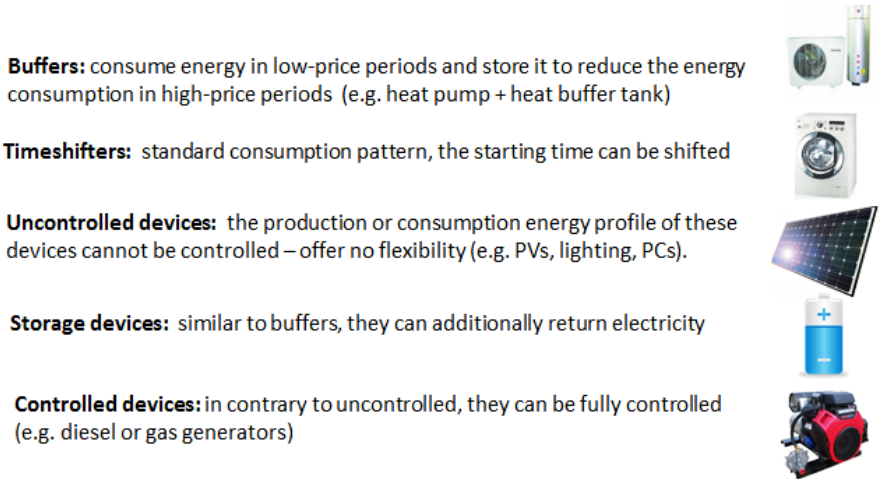


Figure 7: The five resource categories according to RAI

The Control Space of a device is an entity that must be understood by an energy application and after it is analysed according to the application's algorithm, it should result in an energy profile for the device for a future period - an *Allocation*. By energy application we refer to an application developed by a third party and which implements a DSM methodology (e.g. Powermatcher or Triana). Following the opposite direction, the RM receives the Allocation and is responsible to translate it into actions that should be applied by the RD. Figure 8 summarizes graphically the previous paragraphs.

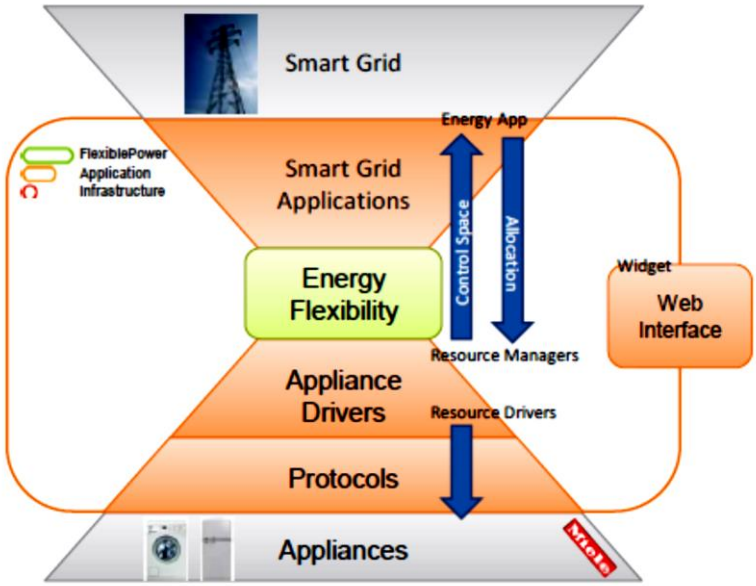


Figure 8: The FPAI architecture [8]

Next, more details about the Buffer and Time Shifter Control Space are presented.

BufferControlSpace

Buffers can temporarily store an amount of energy during specific periods in order to use it in periods when consuming or storing energy is more expensive or generally less beneficial. Examples of Buffer devices are fridges, freezers and heat pumps accompanied by a heat buffer. The RAI of the FPAI uses a number of attributes to describe the energy flexibility of a buffer which are listed below.

- *totalCapacity*: it indicates the total amount of energy that a buffer can store or generate (in kWh in our problem).
- *stateOfCharge*: this attribute shows how much of the *totalCapacity* of a buffer is currently used.
- *targetTime* and *targetStateOfCharge*: when a user wants to have a specific amount of energy (e.g. hot water) available at a specific time (*targetTime*) then the buffer should have a specific amount of energy stored (*targetStateOfCharge*) at that time.
- *selfDischarge*: denotes the losses a buffer has due to its finite insulation (in watts in our problem).
- *minOnPeriod* and *minOffPeriod*: the buffers must operate for a period at least equal to *minOnPeriod* and once they are turned off they should remain turned off for a at least *minOffPeriod*.
- *chargeSpeed*: is the power that a buffer consumes. A number of single power steps as well as a range within the heat pump can operate is defined using the PowerConstraintList class (in kW in our problem).

TimeshifterControlSpace

Devices that belong to this group can shift their operation for a specific period of time. The TimeshifterControlSpace is defined by the next three properties:

- *energyProfile*: the energy profile of a device, visualized by a bar graph, each bar having a height representing the amount of energy and a width representing the time in which this amount of energy will be consumed or produced [3]
- *startAfter*: after that timestamp the device can be started
- *startBefore*: the device must be started before that timestamp

Allocation

The Allocation is the output produced by an energy application and it describes how the flexibility of a device should be used. Two of the most important parameters that shape an Allocation are the:

- *energyProfile* (as described in the previous paragraph)

- *startTime*: defines the start time of the *energyProfile*

A difference between an Allocation and an energy profile formed by Triana can be mentioned. The difference is that Triana uses discrete time intervals (of the same duration) for each of those intervals an energy value is defined that shows the energy consumed during the interval. In contrary, the Allocation contains an *energyProfile,* defined by a list of energy/duration tuples where the duration is not equal for all elements of the list. More details about how an Allocation is formed by Triana are given to the next chapter.

4) Implementing the Planning phase of Triana using FPAI

4.1) The Planning phase of Triana

Planning is the second phase of Triana. The Planning process takes into account the forecasts done in the first phase and a global objective to determine the energy profile of each device for a time horizon, by performing a cost minimization for the device. A common time horizon that is used for the planning phase is 24 hours split in intervals of 15 minutes. The implementation of the planning phase is executed for Buffer devices, for Timeshifters and Uncontrolled devices.

Buffer devices

An example of a Buffer device is a heat pump connected to a heat buffer. A heat pump is a device that uses electricity to produce heat and a heat buffer is a device that can store energy in the form of hot water. When the heat demand of a house is covered using a heat pump and a heat buffer, the flexibility is created by storing hot water in the buffer to be used in periods of high demand-electricity prices.

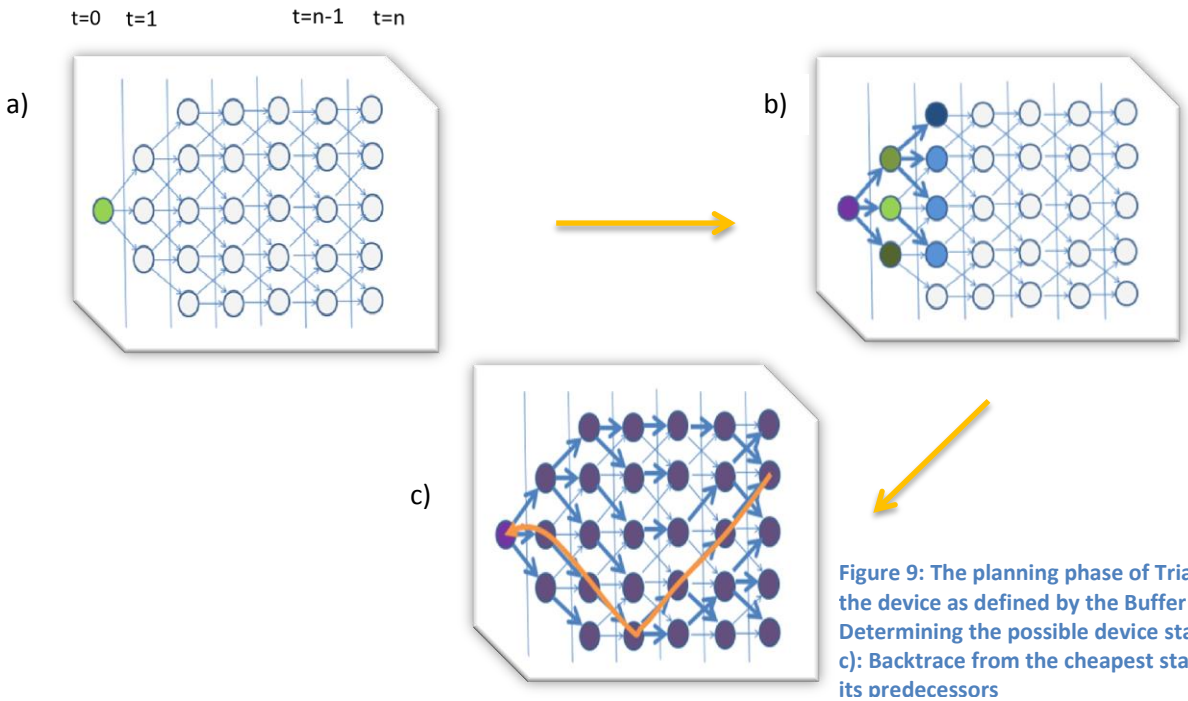


Figure 9: The planning phase of Triana. a): The initial state of the device as defined by the Buffer Control Space. b): Determining the possible device states for all time intervals. c): Backtrace from the cheapest state of the last interval to its predecessors

The Triana States and the price-revising technique

This section presents how the energy price for each buffer device is determined. The energy price is calculated for every time interval and for each device separately. In Figure 9, every circle represents a State of a buffer device. The State contains information about the SoC of the buffer, the energy consumed during the transition to this State form the previous, the total cost of until that interval, the charge speeds of the buffer (heat pump power modes), the total capacity of the buffer and a reference to a State useful for the back tracing done in Figure 9-c). The number of new States that will be created by a State depends on the number of the buffer charge speeds and by the SoC of the State. The total cost is defined as the total cost before reaching this State plus a cost calculated by:

$$cost = \underbrace{energy\_consumption(t) * energy\_price(t)}_{\text{energy cost}} + \underbrace{\beta * energy\_consumption(t)^2}_{\text{local cost}}, 0.1 \leq \beta \leq 0.5$$

A different energy price is calculated during every iteration of the algorithm for all devices. The price revising in Triana is based on the aggregated energy profile, the global objective (in our case a flat profile equal to the average consumption) and a probability that differs for every device. Since it is not realistic to achieve a totally flat aggregated energy profile, we use an upper and a lower bound used as criteria for revising the price of a device. Those boundaries have a distance of ±10% with respect from the average consumption. More specifically after an aggregation of all profiles has been made by the global planner, each controller receives that profile and calculates the distance of the energy consumption value of every interval from the upper and lower bound. Then a random number is generated by the controller and if the distance mentioned is larger than the probability, then the price changes according to a price difference defined in the controller. The price difference is positive if the distance from the upper bound is positive or negative if the distance from the lower bound is negative.

Planning for a fleet of devices

During the planning of the operation of a fleet of Buffers that cover the domestic heat demand of different houses, an initial price vector equal for all devices is sent by a Global Planner to every local controller (a so called Triana Controller) responsible for implementing the Planning for the device with which it is associated. This initial price and all prices in the algorithm are just artificial prices that correspond to the electricity price. The local controller executes its Planning algorithm taking into account the heat demand of the house and other parameters like e.g. the normalized state of charge of the heat buffer which must always stay between 0 (empty buffer) and 1 (full buffer). The energy profiles of the Buffers that result from the planning are aggregated by the Global Planner to observe in which extent the global objective is achieved. In our case the global objective is to reach an aggregated profile equal to the average consumption of all devices. In every iteration the final tasks of the global planner is to calculate the mismatch and also send a command to local controllers to calculate a new energy price for the device they

control. This process is repeated until we reach a convergence of the mismatch to a certain value. Then, the profile shaped for every device is the one that is theoretically returned to the RM who is responsible to translate into actions to be executed by the RD [Figure 9]. An overview of these steps is given below.

1. Initiate-create Triana controllers (executed by the Global Planner)
2. Receive the control space and the heat demand (for Buffers) for the planning horizon. As soon as a Triana controller is initiated, it reads the Control Space of the device.
3. Receive Initial Price values. Triana controllers are responsible for "reading" the initial price vector.
4. Execute Triana's Planning algorithm
5. Return the energy profile of the controlled device
6. Update the price vector based on the aggregated energy profiles and a price-revision technique
7. Repeat 3-6 for a number of iterations (e.g. until the mismatch is adequately small)
8. The simulation ends by returning an energy profile to the resource manager

Triana Controller

Finally it is worth to mention that although the price revision demands the aggregation of all devices' energy profiles by the global planner, the task for calculating the new price (which is different for every device after the 1st iteration) is placed locally to the Triana controller of the device (number 6).

The technique described is called Iterative Distributed Dynamic Programming – IDDP [9].

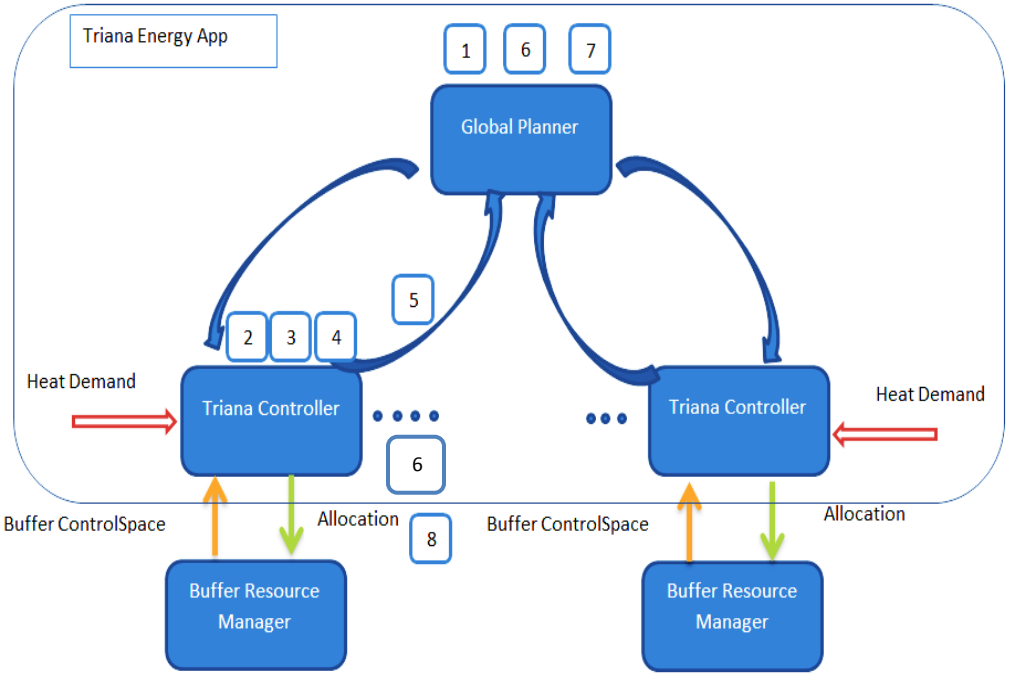


Figure 10: The algorithm steps labeled according to the description given above

The mathematical technique used during local planning is Dynamic Programming. The goal is to explore all possible states of a device at a specific interval so as in the end to choose the route of states that lead to the minimum cost of the device, where cost refers both to the energy consumption cost of the device and to a local cost factor which indirectly imposes an extra penalty to the states with high energy consumption [Figure 9]. The states include information about the energy consumption and the cost of a device when it operates at all possible modes.

Time Shifters and Uncontrolled devices

Time Shifters

The algorithm developed for planning the operation of Time Shifter devices differs in some points to the algorithm developed for the Buffers. Those differences are explained in this section.

As described in the previous chapter, the Time Shifter Control Space is determined by the attributes *startAfter*, *startBefore* and *energyProfile*. The first two variables are objects of the Java class *Date which* shows how many milliseconds have elapsed from the 1st January 1970 and define an exact date and time. The *energyProfile* consists of pairs of values one of them showing an amount of energy consumption and the other showing the duration in which that amount of energy is consumed.

Therefore, those attributes had to be translated in variables that fit to the algorithm of Triana which uses the time interval notion to specify a timestamp during the planning horizon. Java provides a method that calculates the number of seconds (or hours or minutes) of a Date object that passed from the beginning of the current day. By dividing that number with the length of a time interval in seconds and by rounding that number, we can define how many intervals have passed from the beginning of the planning horizon. By doing that for the variables startAfter and startBefore, we define them as integer numbers that correspond to the number of time interval they belong during the planning horizon.

Additionally, the energy profile as defined in RAI had to be translated in an energy profile which has an energy value for every interval. RAI contains a method that can split an energy profile to sub profiles. Each of them, is defined by an offset time (or the starting time of the sub profile) and the duration of the sub profile. In our case the duration is equal to a time interval. The offset is an integer multiple of the interval length. Another method of RAI is able to calculate the energy consumed during the sub profile, which is now the energy value of the interval.

In Figure 11 we can see the form of a Time Shifter energy profile as defined in RAI and how it is split in sub profiles each one having a duration equal to the time interval. RAI defines a continuous energy profile. This profile is defined by pairs of energy consumption and the duration in which this energy was consumed. For example during the minutes 4-38 (duration of 35 minutes) the Time Shifter consumed 0.7 kWh and this is defined by the command:

```
EnergyProfile.create().add(Measure.valueOf(35,NonSI.MINUTE),Measure.valueOf(0.7,NonSI.KWH))
```

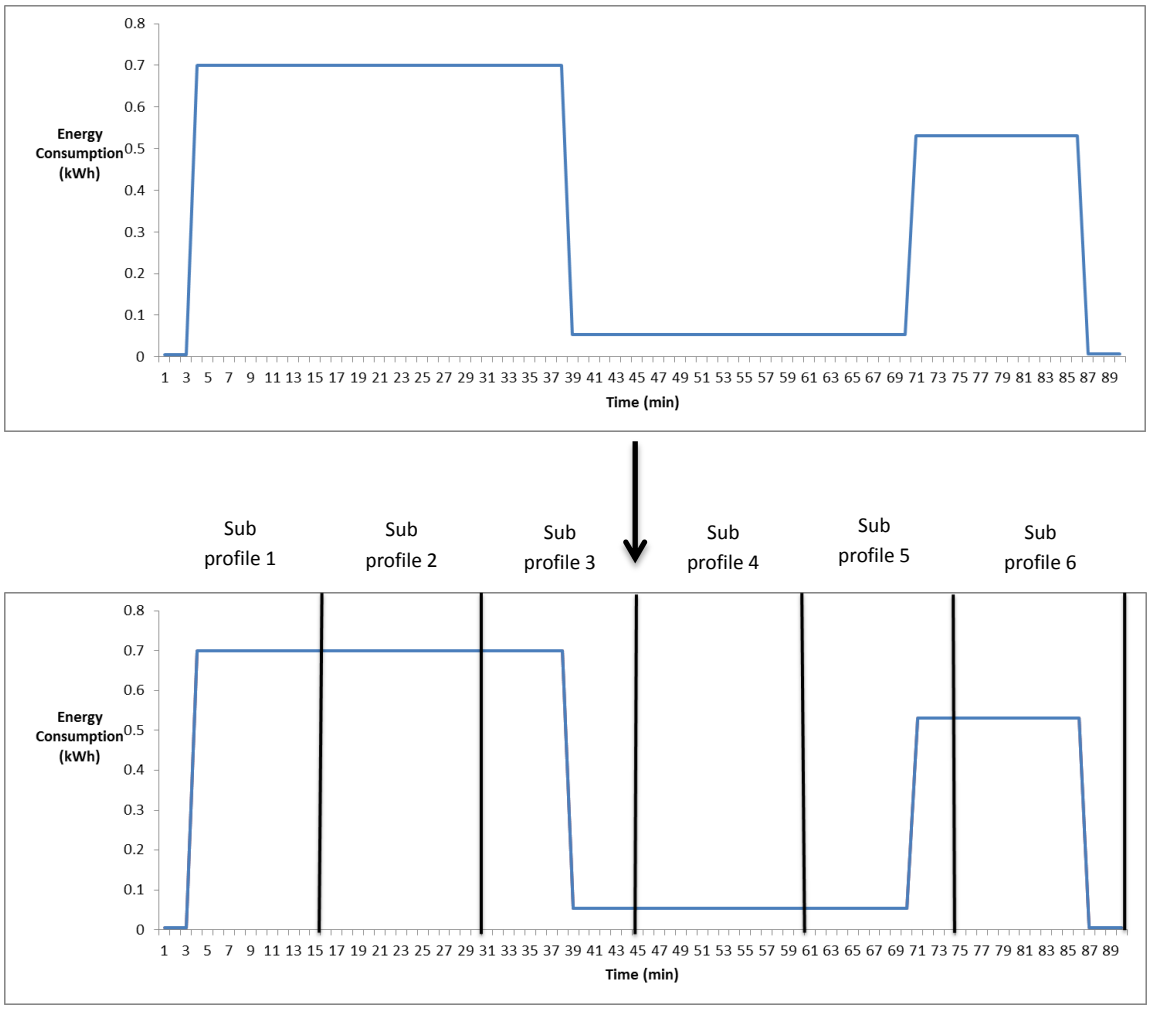


Figure 11: The energy profile of a time shifter as defined by RAI (above) and the energy profile split in discrete sub profiles with a duration equal to the time interval (15 min)

During this conversion of a RAI energy profile to a Triana energy profile, a control is made that ensures that the operation of a Time Shifter doesn't end after the end of the planning horizon. That control is done using a variable representing the time interval that corresponds to the starting time of the operation. If the operation time (also calculated in intervals) added to the starting time interval exceeds the planning horizon, then a Triana energy profile cannot be determined.

During Planning, given a price profile and a time shifter profile we calculate where the energy profile should be "placed" during the planning horizon in order to get the minimum cost. In other words, given a price profile for the planning horizon we define the starting time of the device by finding the cost when the device starts at any valid time interval.

In the next figure, three possible positions of a Time Shifter (in our case it was a hypothetical profile of a washing machine) profile are shown. The profile has a duration of 90 minutes that corresponds to 6 intervals. Only case 2) shows an acceptable position. Case 1) has a starting time before startAfter. Case's 3) ending time, slightly exceeds the planning horizon (begins at time interval 92). Case 2) also represents the optimum solution regarding the energy cost as the price vector has the minimum price values during these specific intervals.

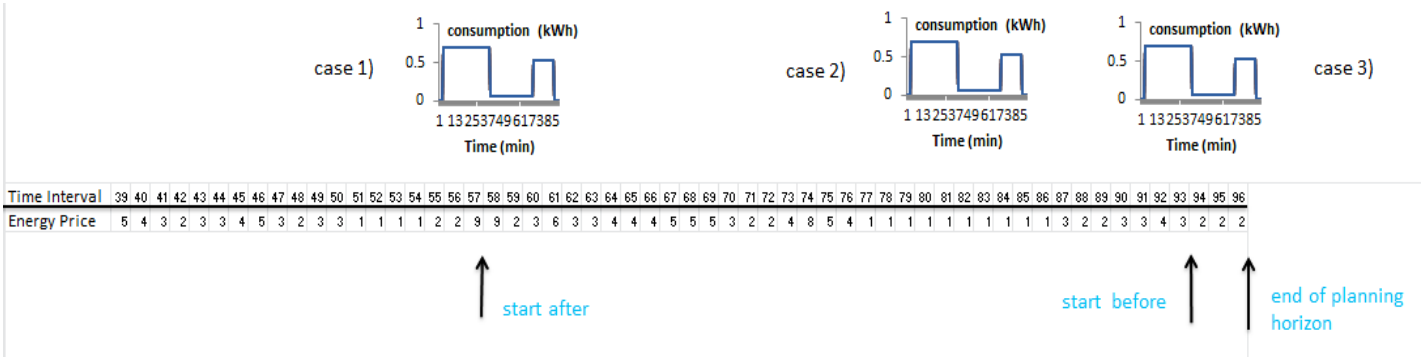


Figure 12: Placing an energy profile in the optimum position of the planning horizon

Uncontrolled devices

The implementation of the Planning phase for Uncontrolled devices is quite simpler compared to Buffer and Time Shifter devices.

The energy profile that belongs to an uncontrolled device is filled with energy values that correspond to the production or consumption of such a device. No alterations can be done to the profile of an uncontrolled device. Its planning process returns to the Global Planner the energy profile that is shaped as described. In our case an installation of PVs represented an uncontrolled device. The production values during a winter day were used to define its profile.

Finally it should be mentioned that the price-revising scheme used for Time Shifters is the same with the Buffer devices, whereas the price-revising is not applied for Uncontrolled devices as there is no optimum cost solution that can be found for them. Of course they influence indirectly the price for the other devices with their contribution to the aggregated profile.

4.2) Implementing the planning phase of Triana using the FPAI tools

The RAI provides all device-specific properties that are necessary for the implementation of Triana's planning phase. These properties in the case of Buffers are the state of charge of the buffer, the capacity of the heat buffer and the power modulation steps with which it can be charged (charge speeds). What is more, the Buffer Control Space as provided by the RAI, offers information for the self-discharge of the heat buffer or a target state of charge as well as other properties (listed in chapter 3) that were not necessary for the implementation of the planning phase of Triana but could be very useful for the implementation of other smart grid methodologies that will use the FPAI in the future.

Another incompatibility between FPAI and Triana is that FPAI uses in its energy calculations only electricity properties. Thus all Triana attributes that expressed energy in the form of heat had to be converted in electricity.

For example although Triana uses the Coefficient of Performance (COP) of a heat pump to calculate the SoC of a heat buffer based on the heat stored in it and the heat demand, in the current implementation the COP was not used and instead we took into account the COP to convert the heat demand of a house to electricity demand. Auxiliary heat devices were also not taken into account. Auxiliary devices are used in Triana to cover the heat demands in the extreme case where the heat stored in the buffer and the energy consumed by the heat pump are not enough to cover the heat demand during one interval. If auxiliary devices are used, an extra penalty is posed to the total energy cost.

Measurable and ConstraintList

The *Measurable* interface makes calculations that include measurable properties more convenient. With Measurable we can express any measurable property in a SI or a non SI unit; any multiplication factor can also be added before the unit. For example, in order to define that the capacity of a heat buffer is 10 kWh we use the command:

```
Measurable<Energy> totalCapacity=Measure.valueOf(10, NonSI.KWH);
```

In order to determine all possible charge speeds of a heat buffer a class ConstraintList has been developed. ConstraintList can define a series of single values or a range between which a heat pump could operate. For example the following constraint list represents a device which can consume 1000 watts and any amount between 2000 and 3000 watts.

```
ConstraintList<Power> cl =
ConstraintList.create(SI.WATT).addSingle(1000) .addRange(2000, 3000).build;
```

4.3) Comments on the Class diagram of the code developed

In the appendix of this document the Class diagram of the code developed is given in the form of Unified Model Language (UML). A few remarks are put in this section to justify this specific structure of the Class diagram.

The Timeseries super class and its sub-classes

The Timeseries super class uses the planning horizon and the time interval length to calculate the number of time intervals of the planning horizon. Then, an array with size equal to the number of intervals is defined. The sub classes *EnergyProfile*, *PriceProfile* and *HeatDemand*, need such an array to store a value for each time interval. For example an EnergyProfile object stores a value in every array index that represents the consumption (or production) of a device.

The aggregated energy profile is also an object of the same type holding in each array position the aggregated energy consumption of an interval. Similarly a PriceProfile object needs such an array to store the energy price values of each interval and a HeatDemand object the heat demand during each interval. In the Timeseries class two methods are written that are used to read from and write to CSV files in order to read necessary data for our algorithm (e.g. the heat demand profiles of the houses) or to extract data from the simulations run.

The State and TimeInterval classes

These two classes are only used in the planning implementation for Buffer devices. As mentioned in previous section a State object contains all information that specify the characteristics of the heat buffer (SoC, buffer capacity, charge speeds) but also methods and variables used to calculate the energy consumption during a transition from a State to another and the corresponding cost of this transition. Each TimeInterval object is composed of a number of States. A method makes sure that when two States have the same SoC, only the one with the minimum cost is stored. A TimeInterval is also responsible for finding the State of an interval with the minimum total cost. This is used during the planning to find the cheapest State of the last time interval. That is the state we use, to find its predecessors that finally determine the optimum energy profile (this energy profile resulted to the minimum total cost). The usage of these two classes made it easier to split different tasks, which are necessary to determine the desired energy profile, to more than one components-objects thus improving the structure and readability of the code and making it easier to add extensions to it.

The TrianaController Interface, its implementations and the GlobalPlanner

The TrianaController interface contains three methods. One of them is the execution of planning for a single device and all classes that implement the interface have a different version of this method as described in the previous sections of this chapter. The two other methods are only implemented by the Buffer and the Time Shifter Controller. These two methods are responsible for creating an initial price profile and to updating the price profile for the next iteration.

The TrianaController implementations receive the Control Space of a device typically by the Resource Manager and execute the Triana algorithm using that information in order to return an Allocation – the energy profile of the device. The Resource Manager is the connecting link between the device and an energy application. Similarly to the BufferTrianaController, a Buffer Agent is responsible for exploiting the Control Space of a buffer device in Powermatcher's implementation.

The GlobalPlanner has a list of TrianaControllers. The first action of the GlobalPlanner is to define what kind and how many devices it will have under control defining the same time the data that need to receive concerning the heat demand (relevant to the Buffer Triana Controllers) and the PV production (relevant to the Uncontrolled Triana Controllers) . When the process of Planning for all devices begins, the Buffer and Time Shifter devices receive an initial price profile and then they execute their planning algorithms. Finally the Global Planner

calculates the total mismatch and gives a command to the Triana Controllers to update their price profiles given the aggregated energy profile.

5) Simulations – Results

A number of simulations was executed to verify if the implementation of Triana's planning phase using the FPAI resulted to the expected energy profiles for a fleet of devices.

Next, a series of graphs is shown for simulations that included 20 houses, each one equipped with a heat pump and a heat buffer that cover the heat demand of the house. Real data that represent the heat demand of a Dutch household for a winter day (collected in 2007) were used. The initial SoC of the heat buffers was set to 0.5, the buffer capacity was set equal to 10 kWh_el (or 30 kWh_th with a COP=3) and the planning horizon was 24h, split in time intervals of 15min.

In Figure 13 the heat demand of one house is shown. The abscissa of every point is the time interval and the ordinate the energy consumed during that interval.

The heat demand of each house has a profile with high peaks at certain hours of the day and zero demand during other periods [Figure 12]. That would lead to a corresponding electricity profile if heat pumps consumed energy according to the heat demand.

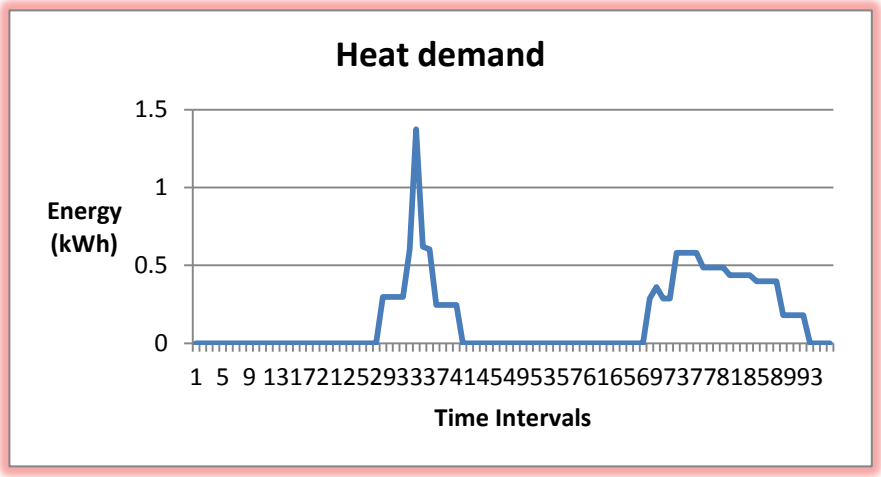


Figure 13: Heat demand of one house

The next graphs show the aggregated energy profile after the algorithm is executed for 2 [Figure 15] and 30 iterations [Figure 16]. The red line shows the average consumption which is also our global objective. The squared mismatch shows in which extent the objective is achieved and is calculated by:

$$Squared\ Mismatch = \sum_{i=1}^{N} (E(i) - E_{av})^2$$

N, the number of time intervals of the planning horizon

$E(i)$, the aggregated energy consumption of all devices at interval i

E_{av}, the average energy consumption of the aggregated profile (a targeted-flat profile)

In case we set a lower and upper bound for the objective of 10% [a representation is given in Figure 14], the mismatch would be:

$$Mismatch = \sum_{i=1}^{N} Mi \ , \ \ Mi = \begin{cases} 0.9E_{av} - E(i) & if \ \ E(i) < 0.9E_{av} \\ E(i) - 1.1E_{av} & if \ E(i) > 1.1E_{av} \\ 0 \end{cases}$$

If those boundaries are not included to the revision of the price vector, then the price tends to appear a strong oscillation, namely having very large values during intervals of high consumption and zero value during intervals of low consumption.

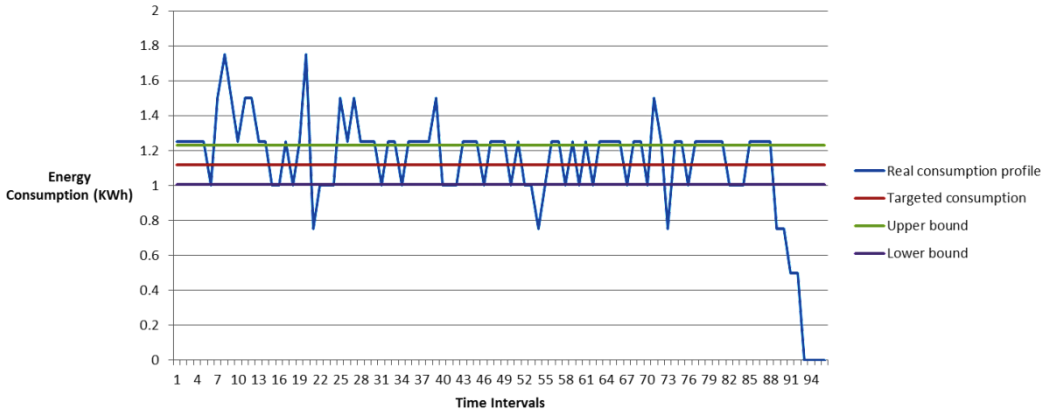


Figure 14: The objective and the boundaries

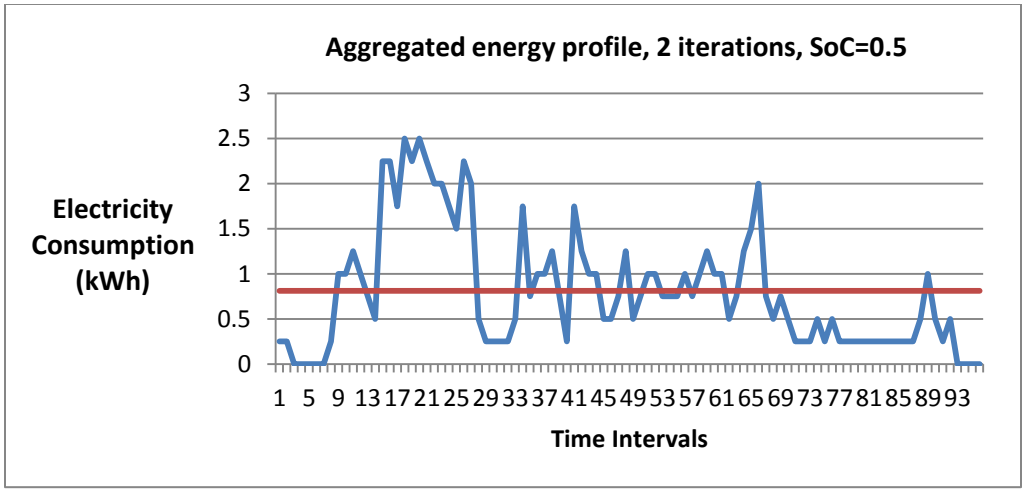


Figure 15: The aggregated energy profile after 2 algorithm iterations

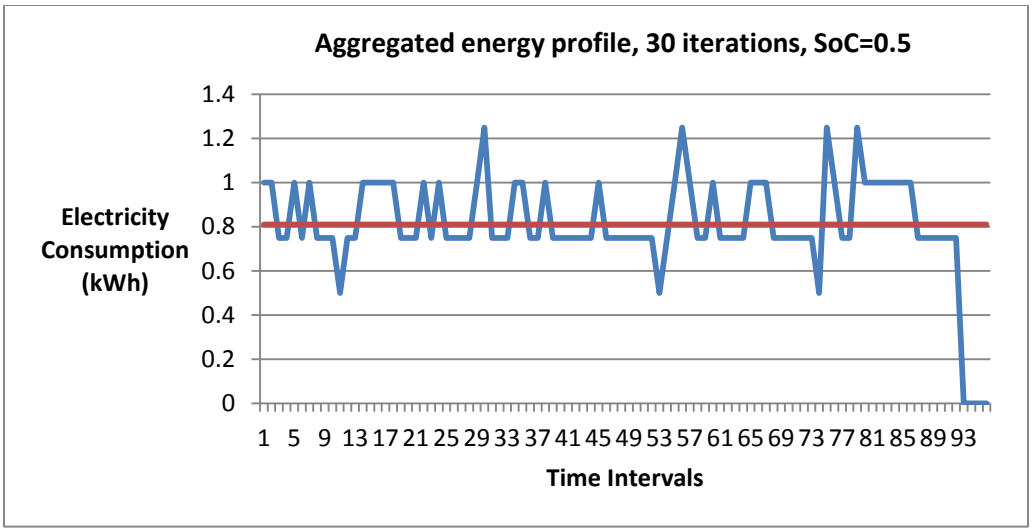


Figure 16: The aggregated energy profile after 30 algorithm iterations

A convergence of the mismatch was observed after approximately the 10th iteration as can be seen from Figure 17 which represents the mismatch after every algorithm iteration. That means in other words, that an insignificant improvement of the aggregated profile can be made after the 10th iteration of the algorithm.

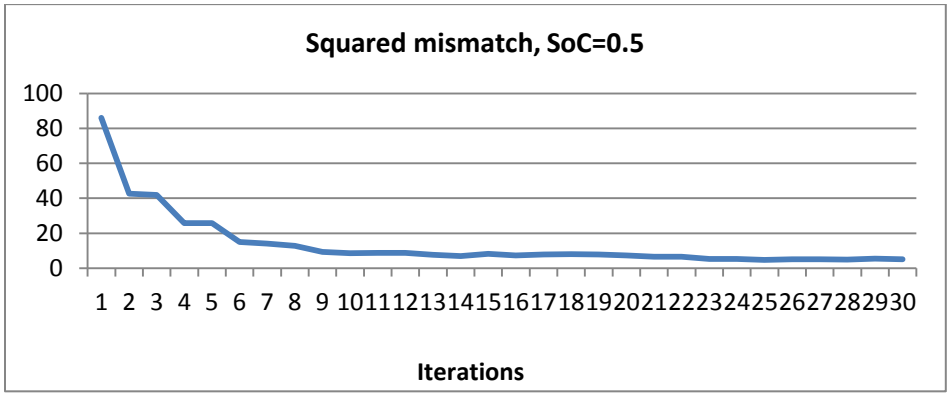


Figure 17: The squared mismatch as a function of the iterations run

Figure 18 also shows that the aggregated energy profile of the 10th, 20th and 30th iteration look very similar. More specifically, important consumption peaks found in the profiles after the 2nd and 5th iteration, are eliminated after the 10th iteration

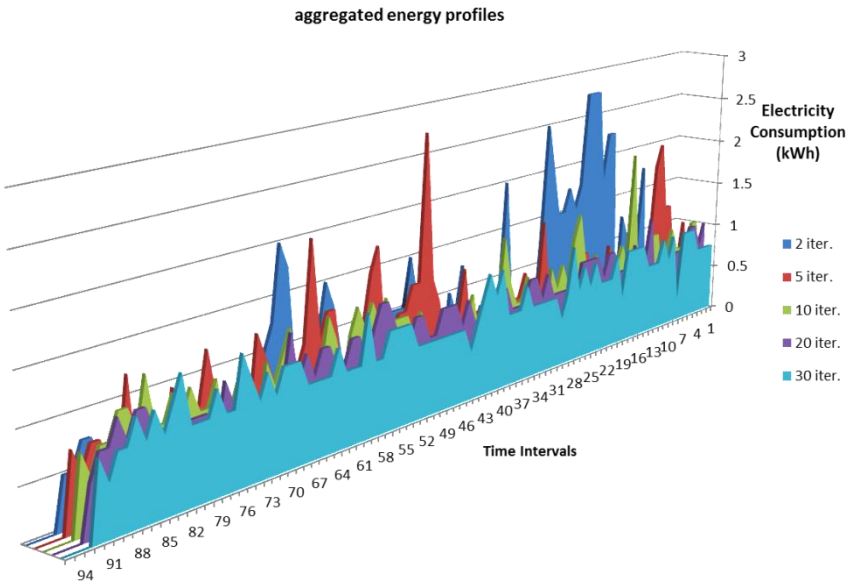


Figure 18: The aggregated electricity consumption for a different number of iterations

When reducing the size of the buffer by 50% a larger squared mismatch (M_2) was observed as well as a larger average electricity consumption. The latter was expected as the initially stored energy in the buffer was half and thus more energy needed to be consumed by the grid to cover the heat demand. The mismatch was also expected to be larger as less flexibility is offered to achieve the objective. The results are summarized in table 1.

Buffer Size (kWh)	5	10
Eav (kWh)	1.27	0.81
M_2 (30 iterations)	15.38	4.96
M_2 (2 iterations)	148.7598	42.71
Peak Consumption (kWh)	2.5	1.25

Table 1: Comparison of the simulations with 5 and 10 kWh buffers.

Further simulations included the existence of electricity production by a PV installation of 70kW_p. In that case the expected behavior of the heat pumps (this simulation included 10 houses) would be to consume larger amounts of energy when the production from the PVs was also larger [Figure 19]. The data for the PVs were also received during winter (December 2012).

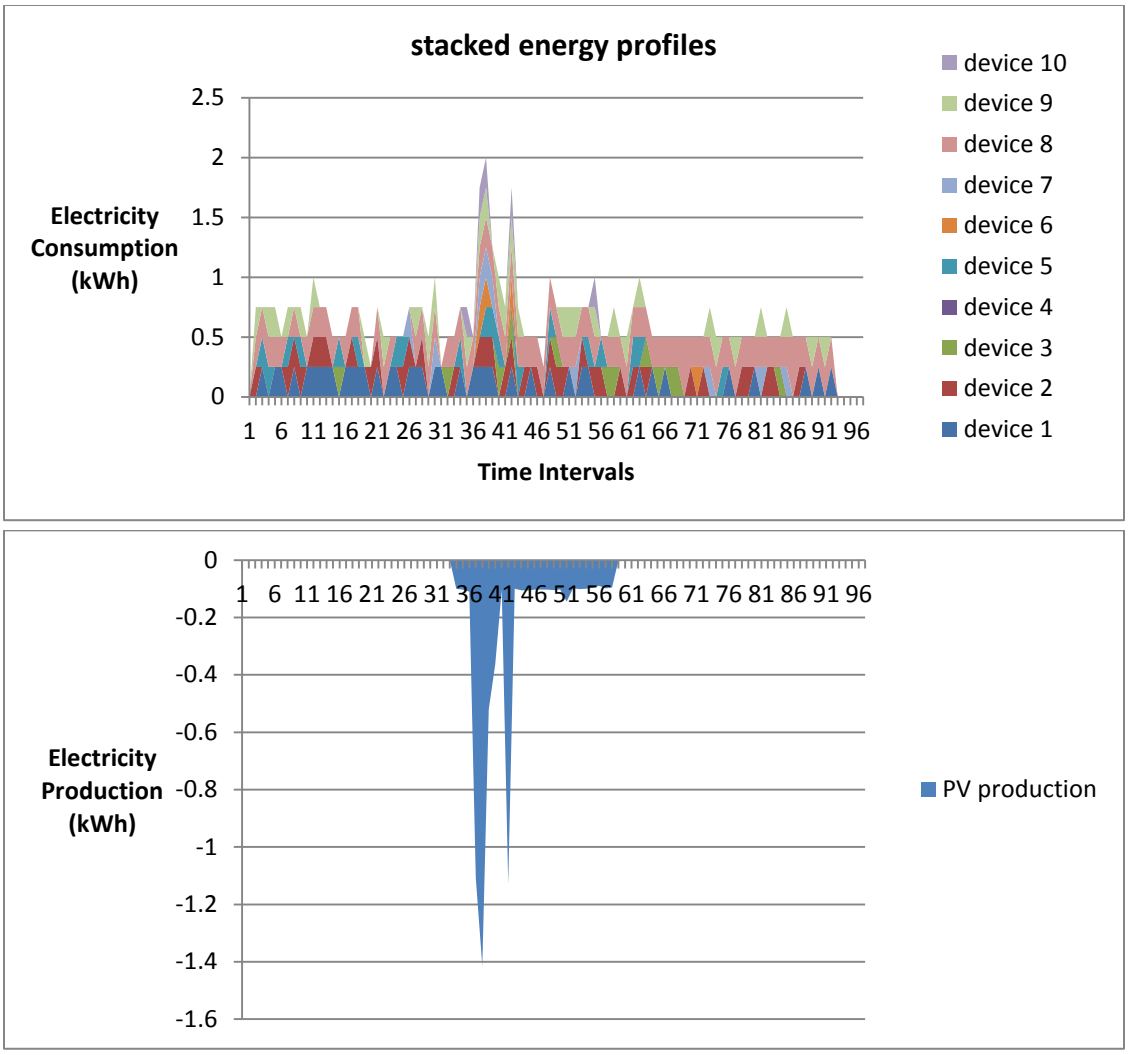


Figure 19: The stacked profiles of 10 heat pumps and the PV production

Finally [Figure 20] simulations were done for households that were equipped with both a device of Timeshifter type and a Buffer device; in this case the Time Shifter was represented by a (theoretical) energy profile of a washing machine. The squared mismatch for this case is depicted in Figure 21.

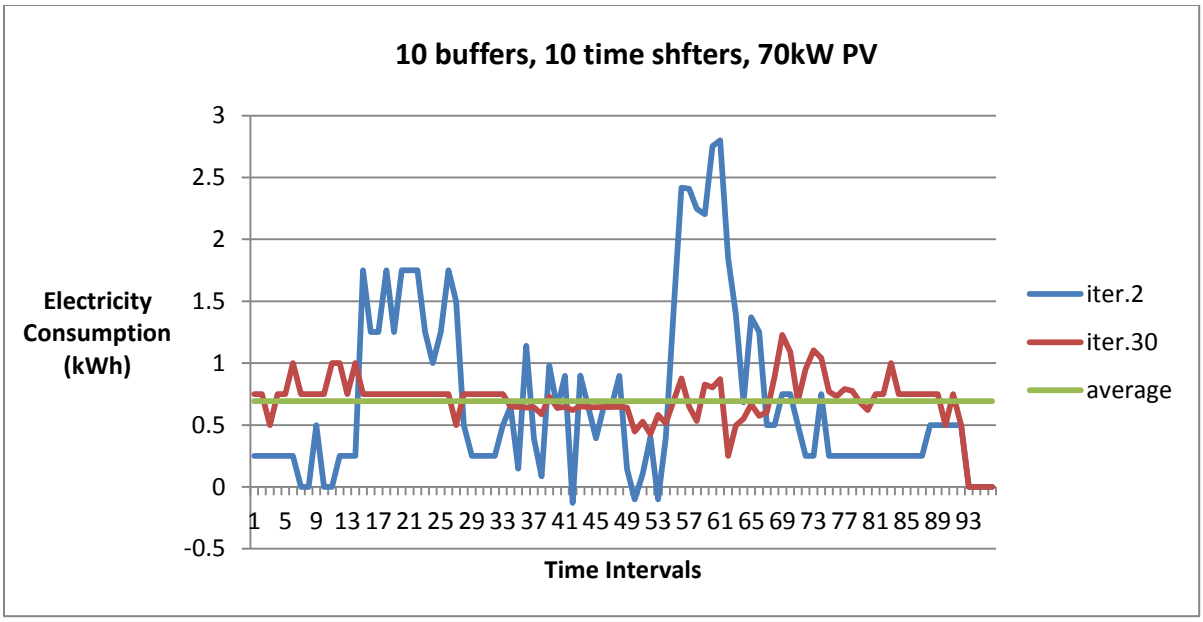


Figure 20: The aggregated energy profile resulting from for 10 buffers, 10 washing machines and 70 kW from PVs

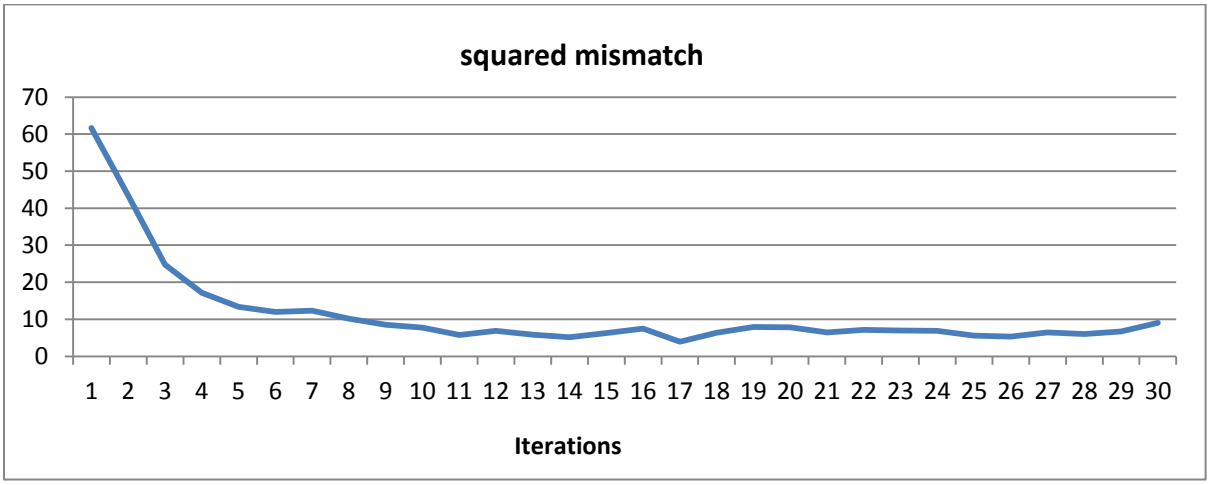


Figure 21: The squared mismatch for the case of Figure 18

A general comment that should be made for all graphs presented is that the appearance of a large consumption decrease during the final time intervals exists because the heat demand for these intervals can be covered by the stored energy of the buffer; that makes the electricity consumption by the heat pumps unnecessary. In a typical Triana simulation this masked by the rolling horizon nature of the optimization i.e. executions are made every 6 hours for a horizon of 24 hours.

6) Summary – Recommendations

The purpose of this internship was to explore the possibilities of designing and implementing a controller that executes the planning phase of Triana using the FPAI framework. The usefulness for both the FPAI team of TNO and the research group of UT was to have an indication whether the FPAI framework provides the tools for creating the desired interoperability for DSM systems, since there was only one DSM implementation based on FPAI, that of Powermatcher.

It was found that FPAI can provide all information regarding the flexibility of a device that the Planning phase of Triana demands.

This internship was focused on the implementation of the planning algorithm for the devices of Buffer Control Space primarily as well as of Time Shifter Control Space and Uncontrolled Control Space.

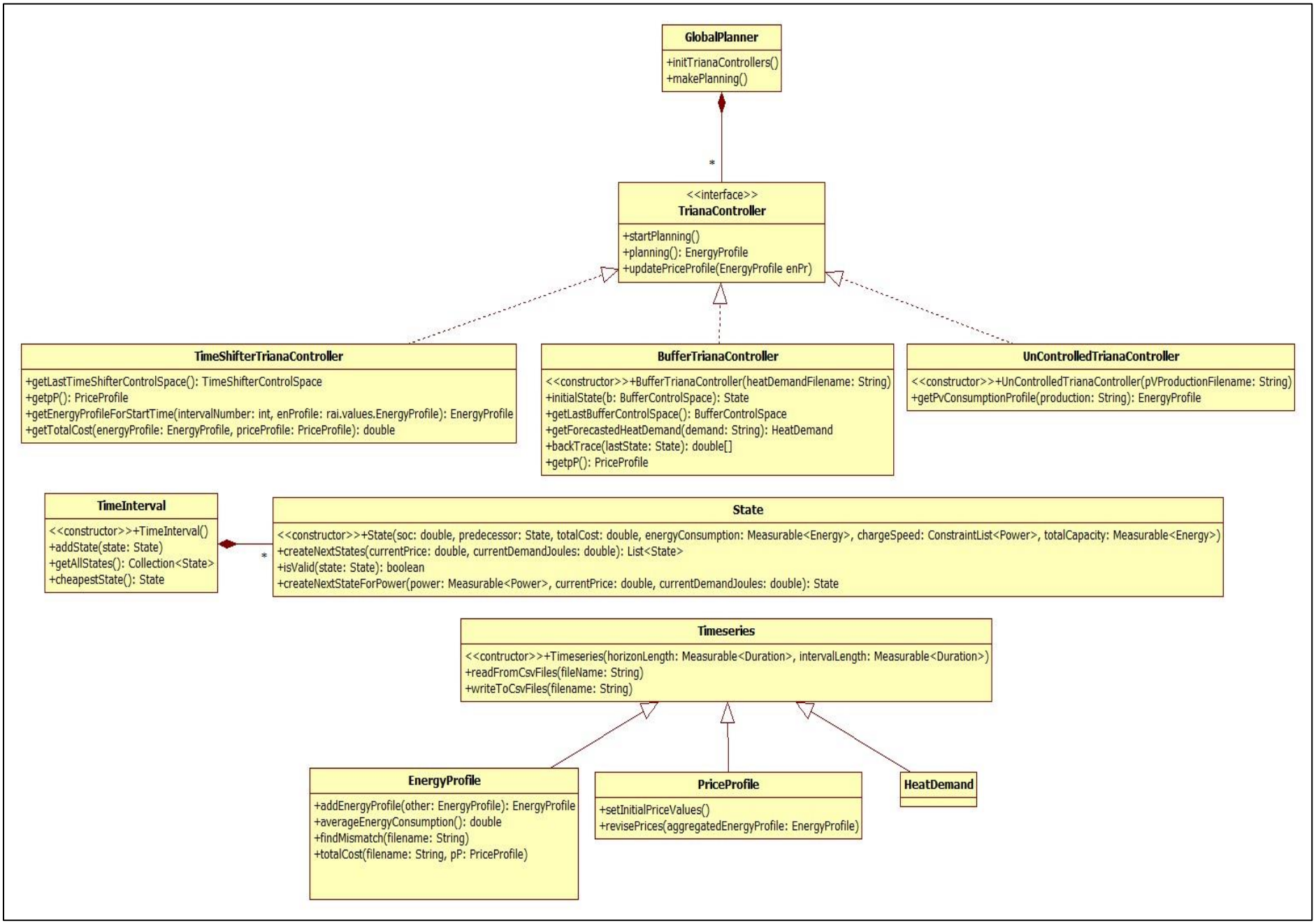
Further research can be done to make a quantitative comparison between the implementations of Powermatcher and Triana when implemented using the FPAI framework.

Finally it is important to mention that for a full implementation of Triana's algorithm based on FPAI, additional components need to be developed for the execution of the Forecasting and the Real-time Control phases, which could also be subjects for future internships and/or master theses.

Acknowledgements

At this point I heartily want to thank Wilco Wijbrandi and Dennis Krukkert whose advice and cooperation, during my presence at TNO for the completion of this internship, was essential.

Appendix – Code UML



References

[1] Stergios Dagioglou, “A literature review on energy control methodologies”, Capita Selecta Assignment, University of Twente, 2013

[2] Vincent Bakker. “TRIANA: a control strategy for Smart Grids”, Ph.D. Thesis, University of Twente, 2011

[3] High Level Functional Specification of the FlexiblePower Application Infrastructure, FAN General Documents, Version 0.4, January 2013

[4] Environmental Defense Fund, “U.S. Smart Grid Finding new ways to cut carbon and create jobs”, URL (May, 2014):
http://www.edf.org/sites/default/files/smart-grid-cut-carbon-create-jobs.pdf

[5] A. Molderink, “On the three-step control methodology for Smart-Grids”, Ph.D. Thesis, University of Twente, 2011

[6] J.K. Kok, B. Roossien, P.A. MacDougall, O.P. Pruissen, G. Venekamp, I.G. Kamphuis, J.A.W Laarakkers, and C.J. Warmer, “Dynamic Pricing by Scalable Energy Management Systems - Field Experiences and Simulation Results using PowerMatcher”, *IEEE Power and Energy Society General Meeting 2012*, IEEE, 2012.

[7] F.N. Claessen, B. Claessens, M.P.F. Hommelberg, A. Molderink, V. Bakker, H.A. Toersche, M.A. van den Broek, “Comparative analysis of smart grid control systems using the Flex Street model”,2011, Elsevier-Renewable Energy Journal

[8] Flexible Power Application Infrastructure, Developer Tutorial, FAN, November 2013

[9] M.Bosman, “Planning in Smart Grids”, Ph.D. Thesis, University of Twente,2012