POLITECNICO DI TORINO

Corso di Laurea in Ingegneria Informatica

Tesi di Laurea Magistrale

Design and Implementation of a Web based Performance Analysis application for a Non-Intrusive Load Monitoring System

Relatori
prof. Andrea Acquaviva
ing. Edoardo Patti

Candidato
Ivi Mehaj

Supervisore aziendale
Midori S.r.l
dott. ing. Christian Camarda

December 2017

Summary

Increasing demand for eletrical energy, has been followed by an emphasis on energy efficiency and putting in play strategies of energy conservation. Non-intrusive load monitoring techniques are based on disaggregating the original electrical signal collected from the eletrical panel of a building, without applying intrusive techniques such as putting smartplugs for each device. This is done with the scope of translating information gathered into meaningful knowledge for energy consumers by making them aware of their energy consumption.

Several obstacles stand in the way. Lack of well defined metrics to measure performance of NILM algorithms, make that newly created algorithms provide non-improving results, making the feedback generated not always effective. Objective of this thesis is to answer the pivotal research question which consists on measuring the NILM algoritm performance with help of Business Intelligence Solutions. A case study is taken into consideration at Midori, a startup company which operates in energy sector by providing detailed information about energy usage to consumers with the objective of triggering consumption saving strategies.

A general purpose multitier architecture addresses this stated problems.For providing an interactive web application the proposed solution is in the form a Single Page Application (SPA) and consists in a Dashboard prototype, by respecting the best practices and giving the feeling of a native-like application. Part of the process is looking through lens of the web application to find aspects of power traces to define a set of KPIs, such as total detection rate of devices, correctly assigned values during the disaggregation process per device, and the acceptable time interval range in which detection is considered correct. This metrics make easy to evaluate the algorithm performance in a quantitative way. Dashboard has been though as a tool of identifying the losses and tracking the mistakes during the developement phase of the NILM algorithm. Improvement gaps are noted to the NILM developer so he can take the necessary steps to improve the product. Feedback provided to developers aids by improving the converging times, by making the whole experience a less time consuming process.

Finally, the designed and implemented solution is valuated through user interaction and provided feedback from the stakeholders, gathered during unstructed interviews or weekly meetings.

Contents

Chapter 1

Introduction

1.1 Overview

Ensuring sustainable energy supply is one of the principal challenges the governments worldwide are facing. Added to that, the decreasing of availability of non-renawable energy sources like fossil fuels combined with effects of the climate changes have added complexity to this challenge. Several initiatives have been introduced to replace consumption of primary energy fuels with electric alternatives. However, this means that the demand for electrical energy will be increased and new approaches with an emphasis on energy efficiency and energy conservation need to be taken into consideration. One of the main aspects of energy conservation is understanding how energy is consumed and translating this knowledge into meaningful information for energy consumers, making them aware of their energy consumption. A great emphasis is made in introducing techniques that inspire behavioral conservation by the part of clients. Thus providing feedback to users indicating their energy usage and behavioral patterns regarding energy consumption is vital. But, a great deal of attention with bearing in mind that violating user's privacy and developing very intrusive techniques can have demetrial effect on the final consumer. In this context development of non-intrusive non-event-based load monitoring algorithms is not a trivial process. Furthermore, process is complicated by the lack of well defined, precised metrics for evaluating the performance of developed algorithm. In this thesis work, a web based application is created for performance evaluation of the on-progress developing Non-intrusive load monitoring algorithm taken from the context of a case study and tracking it's development progress. Scope of this thesis is to identify improvement gaps in NIALM algorithms and identify losses, mistakes or incorrect datas during the development process.

1.2 Problem Definition

Key elements of the research questions stated here serve to provide greater insight about the problems that this thesis tends to answer. *Recognition accuracy* is the most widely used performance evaluation metric for accessing the accuracy of learning algorithms. Most of the researchers base the results of their work by reporting the

performance of their system using accuracy metrics. However, due to incosistency in the definition of the accuracy for the non-intrusive load monitoring algorithms it is not possible to draw meaningful comparisong between reported research work[1],[2]. The overall accuracy is not a good indicator, particularly for the multi-class classificiation because of impaired with data unbalance issue. Other proposals have been made to add other accuracy measures for evaluationg performance: Detection accuracy, dissaggregation accuracy, and overall accuracy. Although combination of these factors can lead to higher accuracy, the diversity of accuracy metrics makes meaningful comparisons between different NIALM algorithms difficult. Thus defining the *performance measurement* as

> " On-going monitoring and reporting of algorithm accomplishment, particularly progress towards pre-established goals"

is a way in which organizations measure the quality of their services toward the objective of meeting their organizational goals. Apart from a common evaluation metric there was a lack of reference dataset on which the algorithm performance could be compared. It is quite obvious that the output of the load disaggregation algorithm is dependent on the source data, which varies either due to difference in the number of type of appliances used or due to the hardware used for extracting load signatures. For retrieving meaningful performance comparisons the availability of common datasets is crucial. Motivated by this, REED and BLUED datasets have been publicly available in order to facilitate the researchers in the development and evaluation of new load disggregation algorithms. Secondly, although NILM algorithms can provide accuracy about targeted homes, the *feedback* generated is not always effective. Translation between energy disaggregation techniques and higher saving in energy is not always feasible[3]. Results of evaluating whether existing energy disaggregation techniques provide power traces with sufficient fidelity to support the feedback techniques that we created, show that feedback accuracy is very low even while disaggregation accuracy is high. These results indicate a need to revisit the metrics by which disaggregation is evaluated [4]. This, because the metrics for gauging NILM algorithms are not consistent. As an end result, **newly created algorithms provide non-improving results**. The above results indicate that the disaggregation community needs to revisit the metrics by which it measures progress. Part of this process will be to look through the lens of applications to find the aspects of power traces that are most important. This defines also the scope of this thesis. After all, **"what you measure is what you get."** The third element is *Business Intelligence (BI)*, which includes a broad spectrum of processes and technologies to **turn data into information and information into knowledge.** Despite the fact that NILM techniques promise several useful use-cases, its broader applications have not been fully realized, reiterates in its conclusion that

> "Despite the major leaps forward in the NILM field, the energy disaggregation is by no means solved. State-of-the art algorithms still leave a lot of challenges when it comes to real-time disaggregation that is general enough to be deploy in any household"

Reference from past literatures enforce this view and demonstrate that exists very little indication on the role the utility companies in the Home Energy Management

Systems(HEMS) industry. But certain ideas invigorate that "utilies" need to set the pace and not take a passive stance in expanding the HEMS systems and setting the standards for user interaction. By exploring their full potential they can take central role to better take advantage of the full energy savings. In general, lack of the standardization of NILM performance metrics, development of efficient NILM benchmarking framework and availability of high-quality energy data set are critical for the advancement of energy disaggregation. In the light of the previous arguments, research question would be defined as

> **"How to measure the non-intrusive load monitoring algorithm performance with help of Business Intelligence Solutions?"**

The three defined key elements of the research questions are formally defined in chapter 2.

1.2.1 Motivation

In a nutshell, answering this top level questions described above in this thesis, it will be easier for a company to assess the performance of their developing algorithms, with its apparent benefits and and not so apparent costs. This thesis tends to accomplish this by firstly, identifying the added value of the Business Intelligence tools and solutions for the company in the ICT energetical sector. Secondly, how the performance of unsupervised non-intrusive load monitoring algorithms can be measured. As a mean of realizing this thing defining which KPI(Key Performance Indicators) are used for judging the internal performance of NILM algorithm is essential. A prototype for reporting performance of the algorithm is developed and as a final step the validation of the framework with stakeholders feedback is taken into consideration.

1.3 Aim of Work

The objective of this thesis is to design, develop and implement a web-based application in a form of a Single Page Application (SPA) targeted to non-intrusive load monitoring algorithm developers taken from a case study, in order to be able to evaluate performance of the algorithm in a quantitative way. The final application thought as Business Intelligence solution, provides greater insight about the on-development algorithm. The web application should be able to provide feedback to algorithm developers, by providing a set of comparable metrics in order to judge the efficiency of the algorithm with respect to certain defined goals.
To be able to answer the research question defined in the previous section, certain elements must be defined. First, the NILM performance measures in context of measuring the internal performance must be identified.Here a set of KPI for measuring the internal performance of the algorithm will be defined. Since there exists an ambiguity about which metrics should be used to evaluate the performance of the algorithm, in cooperation with other members, additional KPI are defined for

increasing the accuracy and determing the performance of a certain algorithm with respect to certain pre-defined goals. The KPIs are proposed with respect to the findings from the literature review and refined and integrated with results obtained from the case study. These KPIs which provide the backbone of the framework, will be private to the company and not available to the customers. The second element should be an improvement guideline. The intention is to simplify life for the NILM developer by providing feedback about the errors occured and discoverd during development and noted by the KPIs. In such way, the error detection rate improves in a less time consuming process. Realization of the proposed solution, will be realized in form of a dashboard which represents the Business Intelligence Solution. In addition to that a demystification of the SPA architecture as a tool for realizing the dashboard is provided. Its differences from the traditional web applications are made visible with the scope to reflect how the SPA architecture helps in better decision making in the overall picture of NILM as part of the Business Intelligence solution. Finally, the designed solution is valuated based on the feedback from the stakeholders, gathered from unstructured inteviews and meetings.

1.4 Thesis Layout

The main purpose of this chapter is to provide the structure of the thesis in the following chapters. chapter 2 gives a detailed info on the background and theroretical insight about non intrusive load monitoring algorithms and relevant Business Intelligence solutions in the ITC energetical sector and provides some state-of-art solutions. An analysis of an SPA as an implementation of the business solution and various technologies used in development and deployment of an SPA are also discussed. chapter 3 takes into a consideration a case study and portrays the overall architecture structure of NED, with a strong reference to the non-intrusive algorithm developed as one of the building blocks of the architecture. A detailed explanation of the architecture and its communication paradigm is provided. chapter 4 aims to explain why the need to design an evalution Business intelligence solutions to fill the void and why such tool is of great importance. The complete process for developing an SPA with best practices is included in this chapter. chapter 5 gives an insight of the implementation of the proposed framework via a Business Intelligence Dashboard. The application architecture is analysed in details and the steps undertaken to develop the features of the dashboad. Technical choices are explained in detail and also some of the drawbacks from the implementative point of view are also noted. chapter 6 explains the results obtained from the newly designed web application and the implication on the overall process of creating a new version of NILM algorithm. Also the mistakes noted during the development phase have been documented. Lastly, discussion about the limitations of the proposed approach, future work and conclusion is presented.

Chapter 2

Literature Review

In order to answer the pivotal research question stated in the previous chapter, we try to decompose it into the following mini objectives, by easing the process of assessing the performance of the developed NIALM algorithm to the developers group.

1. Identifying the added value of Business Intelligence (BI)
    - What is BI?
    - What is the added value for the business and how can it be measured?
    - How can BI solutions applied to track progress during development of NILM algorithms?
2. Defining how the performance of a NILM algorithm is measured
    - What KPI are used to measure the internal performance of an usupervised NILM algorithm?
    - What frameworks are available to measure the internal performance of NILM algorithms?
3. What is the current and the desired situation according to the company from the case study?
    - Provide a case study description
4. Proposed framework of KPIs for evaluation performance on NILM algorithms
    - What internal KPIs are used?
5. Developing a dashboard as a prototype of BI solution for reporting performance
    - Based on literature review and specific data from the case study a BI prototype is created
6. Validate framework and the prototype with stakeholder's feedback

This chapter lays the theoretical background for the research. Here the concept of Business Intelligence (BI) is explained and the unsupervised NILM performance measurement is discussed. The focus of this chapter is to fulfill the first two objectives stated above.

2.1 Research Methodology

This section provides an overview of the research methods used for literature findings in this thesis. For ensuring an unbiased literature review the Wolfswinkel model [5] is used for literature research. The criterias for literature selection are defined first, followed by literature search. Once an initial publication is retrieved, it is refined and additional publications are fetched from forward and backward citations. Some of the forward and backward citations are also added to the sample set and refined in the next iteration. This is an iterative process, which consists of filtering the duplicate studies. Forward citations are papers which use the paper as a source, while the backward citations are the publications that are referenced in the paper of interest. Iteration loop is exited after no new ideas on the subject are generated. After literature set is selected, it is analyzed and presented. During analysis of the set of literature, three different techniques are used. Open Coding consists in the fact that the researcher should continuously go back to the earlier read papers and/or excerpts; concepts(and/or categories with their belonging properties) might need to be changed due to potentially new insights gained in a later reading. Secondly, through this so-called axial coding (Step 4.2 of 2.1), the interrelations between categories and their subcategories (including their properties) are identified. These higher-order categories (which qualitative researchers sometimes describe as core or key) will eventually represent the main themes or patterns of the studies' findings in the data. Concepts can be part of categories or become (sub-)categories themselves. Thirdly, Selective coding (Step 4.3 of 2.1) is used to integrate and refine the categories that were identified. Although axial coding is about finding the relations between categories and their sub-categories, selective coding is the process of identifying and developing relations between the main categories. It is mainly during selective coding that the researcher theorizes and/or reconceptualizes in ways he or she sees fit the puzzle at hand. The essential task at this point is to develop a single reasoning with which one or a set of phenomena are potentially explained.When concepts and (sub-)categories begin to emerge, researchers need to be already engaged in so-called 'comparative analysis' – continuously comparing, relating and linking the identified categorizations with each other and the studied papers and excerpts. Preliminary results of analyzing the randomly selected texts/studies actually guide the consecutive reading and further analyzing of the remaining texts. This is called 'theoretical sampling'. The process is illustrated in 2.1. The Five-stage Grounded-Theory model is not used in its entirety but used as a guideline for the relevant literature research. The analyze stage which more strongly characterises the model is followed strictly. For collecting the initial set of literature several suggestions by the stakeholders were included, which is not formally defined in the model. Later in the design of the artifacts those part of the literature study which did not fit were removed. The objective of this thesis is not to review an extensive set of literature

Number	Task
1	*Define*
1.1	Define the criteria for inclusion/exclusion
1.2	Identify the fields of research
1.3	Determine the appropriate sources
1.4	Decide on the specific search terms
2	*Search*
2.1	Search
3	*Select*
3.1	Refine the sample
4	*Analyze*
4.1	Open Coding
4.2	Axial Coding
4.3	Selective Coding
5	*Present*
5.1	Represent and structure the content
5.2	Structure the article

Table 2.1. The five Stage Grounded Theory used for literature research

in the areas of interest, but rather present the state of the art in these areas with a focus on the main topic of the thesis.

2.2 Smart House and Smart Metering

The smart home concept was first introduced by the American Association of House Builders in 1984 [6]. The concept was a broad one applied in many different industries,thus different definitions were introduced. The energy sector associates Smart Home concept with the efficient provision, co-production and consumption of energy [7]. A smart meter is an electronic device that records consumption of electric energy in intervals of an hour or less and communicates that information back for monitoring and billing purposes. Different smart meter projects are funded and carried out by different organizations all around the world. FLEXMETER [8] is a smart metering project being developed and funded within the European Union's Horizon 2020 research and innovation program in collaboration with Politecnico di Torino. Because smart meters rely largely on communication between the utility and the household through the current electricity transmission system, it is not necessary to simultaneously implement large- cale infrastructure changes. In this way they provide an advantage from the costs point of view[9]. While it is important to plan comprehensive infrastructure improvement over the coming decade, these infrastructure projects will complement the benefits that smart meters offer both in the near future and for years to come. As mentioned above, the vision introduced, is that of world where tons of interconnected intelligent devices and networks serve human in an unobtrusive way. In this manner, the final objective is to provide information to the users by improving their energy consumption by merging the

technology domain with the economic domain.

2.2.1 Load Monitoring Concept

The goal of Appliance Load Monitoring (ALM) is to perform detailed energy sensing to provide detailed information on the breakdown of the energy spent. This would further enable the automated energy management systems to profile high energy consuming appliances, allowing them to devise energy conservation strategies such as re-scheduling of high power demanding operations for the off-peak times. Moreover, companies would be able to develop a better understanding of the relationship between appliances and their usage patterns. ALM is an essential prerequisite for providing energy feedback to the residential consumers, but it is equally beneficial for the industrial sector because of its applicability in fault detection and remote load monitoring services [1]. There are two existing major approaches to AML metering technologies for collecting energy consumption data:

- Intrusive Load Monitoring
- Non-Intrusive Load Monitoring

For an intrusive monitoring, the energy consumption of each appliance can be gathered from a small meter, installed into a power socket. The advantage of this approach is high accuracy of metering, but suffers from an inconvenient installation process and a high cost for a large-scale deployment. The other method is less intrusive and reduces the strain of a complex installation, as explained in detail in the next section.

Non-Intrusive Load Monitoring

Non-Intrusive load monitoring (NILM), also called non-Intrusive appliance load monitoring (NIALM), developed in the 1980s at MIT [10] only needs a single meter and can dissaggregate the usage of power consumption of appliances according to different signatures of appliances and their states. Electrical loads exhibits a unique energy consumption pattern often termed as "load or appliance signatures", that enables the disaggregation algorithms to discern and recognize appliance operations from the aggregated load measurements. Thus a signature is a particular trait over the aggregated signal, that can be associated to a specific appliance [11]. Appliance identification is highly dependent on load signatures, which are further characterized by the appliance category. Usually NILM frameworks consist of several modules, discussed as depicted below:

- Data Acquisition Module
  Role is to acquire aggregated load measurement at an adequate rate so that distinctive patterns can be identified. A further classification of power meters can be as follow:

1. Low-Frequency Energy meters
2. High-Frequency Energy meters

- Future Extraction
  The next step after acquisition of data is to compute the power metrics. Normally as power metrics are used the active and reactive power. An event detection module detects the ON/OFF transition of appliances by analysing the changes in power levels. In the literature, several event detection methods have been proposed [12] and in order to characterize the detected events, steady-state and transient event-based feature extraction methods are developed. There has been a debate considering the use of either steady-state or transient based features extraction methods for load disaggregation as both of these approaches have their advantages and disadvantages. Bearing in mind the cost of the solution, the steady state methods seem to be a more feasible approach because it requires low-cost hardware. On the other hand, load disaggregation algorithms can incorporate transient features to improve the segregation of appliances with overlapping steady-state features, but at the cost of expensive hardware[1]. The functional description of steady-state and transient methods mentioned above is beyond the scope of this thesis.

- Load Identification
  The extracted appliance features are further analyzed by the load identification algorithm to identify appliance-specific states from the aggregated measurement. The majority of research work in NILM method is focused on supervised machine learning techniques that require labeled data for training the classifier. Supervised disaggregation methods for NILM can be divided into optimisation or pattern recognition based algorithms. Optimisation based methods deal with the task of load disaggregation as an optimisation problem. Problem with optimisation problem becomes more complex in the light of composite load disaggreation, as instead of one-to-one matching algorithm, now has to take into consideration possible combination of appliances present in the database, which could have generated the observed signal. Tentatives to use generic algorithms, but problem remain when unknown loads are pressent in the aggregated load data. Pattern Recognition methods are the ones most frequently used by researchers for load disaggregation. The appliance database contains multiple appliance specific features that are used to define structure and parameters of the recognition algorithm. In this technique which is a pattern recognition based approach, where each observed power measurement P(t) is matched with a combination of appliance power signals, present already in the database. Although various techniques based on supervised algorithms have been presented with reasonable performance, it is highly desirable for NILM systems to be installed in a target environment with a minimal setup cost as training requierement for the supervised load identification algorithms is expensive and laborious. They require individual appliance data for model training, prior to the system deployment, and thus their functioning depend on the user intervention and the a-priori knowledge of the power system parameters in which they are working is fundamental. For surpassing these inconveniences, unsupervised NILM techniques have been developed. These

approaches do not require individual appliance data and the models information is captured only using the aggregated load, without the user intervention. Furthermore, they are dependent on the number of appliances forming the aggregated load and capable of dynamically adapt to the power system changes over time (i.e.., addition, removal or substitution of appliance). The latter approach is the one chosen by the group of developers which has been taken under consideration in our case study, since the latter technique has a major drawback consisting in the presence of unknown loads complicates the optimisation problem, since the techniques tends to provide a solution based on the combination of known appliances. Requirement of training data for all devices is one the reason, why NED has adopted unsupervised method for load disaggregation, so that the need for annotated data can be eleminated. The unsupervised methods are non-event-based. They make use of unsupervised learning techniques and attempt to disaggregate load measurements directly without performing any sort of event detection. Load classification in the unsupervised techniquers aim to infer the appliances models having only the aggregate signal available. For this purpose, various clustering procedures try to discriminate between different signatures observed within the aggregated signal. In the classification phase, the cluster information is used for source matching when a signature is revealed within the aggregate signal. Other approaches rely on machine learning techniques for classification to model event matching. Solves the problem, using various blind source separation technique, in which no information about the sources is available. After the source signals are revealed, each method performs the specific appliance assignment.

- System Training
  This pre-learning phase is normally a prerequisite for NILM systems where the supervised disaggregation algorithm required adequate labeled data for learning the model parameters to perform the appliance recognition. Further they can be classified as on-line and off-line training methods. For on-line training, research use time window based methods for real-time detection and learning of appliance features. But, upon detection the manual labeling of the appliances is still challenging and complex since only the aggregated load values are observed instead of individual appliance measurements. Conversely the off-line training approach gathers the load measurents from the environment for a time period, such as specific days and applicances are labeled according to a pre-existing signature in the database. For easing the data annotation process, NED uses sub-metering process which requires installation of energy meters per appliance to record appliance-specific consumption. This comes at the cost of extra monetary costs, but also requires complex installation of sensors. The main reasons why this techniques is employed is to ensure, that system training is done in easier and more accurate way, but also as a mean of comparing the developed non-intrusive load monitoring algorithms with very accurate data retrieved from the sub-metering process. The whole process of building an appliance signature database and data annotation is a tedious process, facilitated by the human interaction and supervision.

2.3 Performance Measurement in NILM

It is currently practically impossible to compare any pair of dissagregation papers and algorithm implementations about NILM. Each paper uses a different dataset, different metrics, different pre-processing. As such, no feasible way to measure progress of NILM over time. There does not exists a certain methodology of any way for deciding which NILM approaches are the most promising and hence most deserving for future research and implementation. Thus far, NILM research is like a natural selection without selection[13]. Hard to replicate algorithms described in papers because little of the code is shared and often not full implementations details are included. Performing a performance evaluation of a NILM algorithm is difficult, because there is no standard evaluation procedure to apply [1]. This is supported even by [14], which indicates that a major obstacle to the release of finalized commercial unobtrusive energy disaggregation solutions is believed to be the almost complete *lack of formal methods to evaluate the performance of the underlying algorithms*, hence compromising the feasibility of the approach in real-world deployments. Even more challenging is comparison between different algorithms. This because composition and usage of appliances differences from time to time and from household to household. Performance of an algorithm depends on number of appliances running at the same time, the "noise" in the aggregated consumption data caused by unreported appliances, the performance metrics selected by the author, and the input parameters they choose to tune their algorithm [15]. Nevertheless, developing NILM algorithms and metrics holds its own challenges. For example, different algorithms have different inputs and training methods (e.g. event-based approaches require labeled transitions while non event-based approaches require individual appliance consumption data). Likewise, algorithms may produce different outputs, e.g., an event-based algorithm can produce a sequence of labeled transitions, while a nonevent based algorithm may produce estimates of the energy consumed by each appliance, thus supporting the importance of having mechanisms to facilitate the process of developing such algorithms.

2.3.1 Evaluation Metrics

[16] proposes a test bench and some quality measures to evaluate relevant parts of NILM system as a way to measure algorithm's performance. The analysis procedure for an algorithm consists in three stages:

1. Data generation
   NILM researchers need to describe in detail the data they are using to build models, train, and test their NILM algorithms [17]. Need to generate a set of ground truth events and signal data for voltage and current from several outlets. As a measurent can be considered the measured values of real power p(t) or real power and reactive power q(t). Samples are stored in an array, and also set of ground truth data telling us when($\vec{e}$) and which($\vec{a}$) appliance was turned on and off.

2. Algorithm Under Test (AUT)
   An event detection and/or classification algorithm is activated to put under

analysis the algorithm. Normally the structure of any NILM algorithm consists in two block:

- Event detection
  Tasked with the job of finding the switching events in measured signal.
- Appliance classification
  Assign an appliance ID or a label to each of the found events. Because each algorithm is implemented differently a lack of standardized interface to make comparisons easier is lacking.

3. Test Bench
   For judging the performance of the algorithm, a set of numbers indicating its quality should be implemented, named as *quality measures*.

Quality Measures

As defined above the Quality measures can be divided for the two building blocks of the NILM algorithms as

- **Quality Measures for Event Detection**
  Here the number of the events is fundamental. By appropriate ground truth measurements we get a list of true switching events:

$$\vec{e} = [e_1, ..., e_{N_e}]^T \tag{2.1}$$

  where e_i is the time stamp of the i-th event. Another $\vec{a}$ contains a vector of appliance ID for labeling purposes. After the event detection algorithm generates a list of detected event time stamps saved in vector:

$$\vec{\hat{e}} = [\hat{e_1}, ..., \hat{e}_{N_d}]^T \tag{2.2}$$

  where N_d is the number of detected events.

  – True Positives and True Positive Rate
    An event detected is considered as true positive $\hat{e}_j$ when:

$$|e_i - \hat{e}_j| < \Delta t_{tot}, \tag{2.3}$$

    with Δt_{tot} is the time tolerance. From the total number of detected events we get the number of true positives.

$$N_{TP} = \sum_{N_d} TP \leq N_e \tag{2.4}$$

    The true positive rate is the relation of number of true positives and true number of events (ground truth),

$$TPR = \frac{N_{TP}}{N_e} \tag{2.5}$$

As noted the value of TPS depends on the quality of AUT, which defines the desired point of measure and on quality of the ground truth. Bad ground truth data will result in bad algorithm performance by using this measure.

- False Negatives and False Negative Rate
  It may happen that the event detector misses an event, so for at least one $\vec{e}$ event for which no detected event $\hat{e}_j$ was found.

$$N_{FN} = N_e - N_{TP} \geq 0 \tag{2.6}$$

  , and the false negative rate as:

$$FNR = \frac{N_{FN}}{N_e} \tag{2.7}$$

  By the definiton is clear that each ground truth event is either a true positive, if it is found by the algorithm under test, or a false negative, if it is not found.

- False Positives and False Positive Rate
  Measured signals may contain edges, ramps or peaks that are not described by the ground truth data. Nevertheless, event detector finds events in these signals. These events are defined as *false positives (FP)*. So if not match found according to 2.3 then it is counted as false positve. The number of false positives is:

$$N_{FP} = N_d - N_{TP} \tag{2.8}$$

  with the corresponding false positive rate:

$$FPR = \frac{N_{FP}}{N_e} \tag{2.9}$$

  From the last equation it is possible to have a FPR $\geq 100\%$, depending on the signal's quality and sensitivity of AUT. The ideal event detector will show an FPR $= 0$.

- Quality Measures for Appliance Classification
  The measures regarding the appliance classification can be summed up as:

  - Number of appliance *events* correctly classified
    Appliance events are single events caused by an appliance. Turning on an appliance and turning it off again are two events. If we consider N_A the individual appliance and N_e be the total number of events(both on or off) that were caused by connected devices.Figure 2.1 shows an example of how appliance event and cycle is extracted from the signal. Also a vector of number of events for each appliance is computed by counting matching appliance for all true positives. From this we can compute the Appliance Event Classification Rate:

$$AECR = \frac{\sum \vec{\hat{n}}_e}{\sum \vec{n}_e} = \frac{\sum_{j=1}^{N_A} \hat{n}_{j,e}}{N_e} \tag{2.10}$$

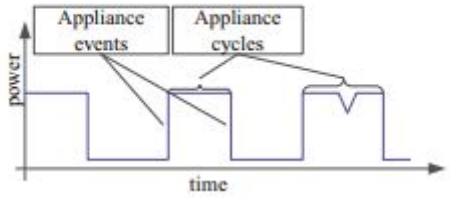

Figure 2.1. Definition of appliance classifier Stage

From the equation is evident that number of TPR (True Positive Rate) is the limit for AECR. Also, AECR can be computed for each appliance individually by:

$$AECR_j = \frac{\hat{n}_{j,e}}{n_{j,e}}, j = 1, ..., N_A \tag{2.11}$$

It would be wise to take individuals appliances's ground truth events as a basis. The AECR tells us how good a classifier for load signatures is on single event basis.

– Number of appliance *cycles* correctly classified
Appliance cycles are pairs of turn-on and turn-off events. For N_A as number of individual appliances and N_c the total number of appliance cycles (combination of on-off events) and $\vec{n_a}$ a vector which lists the number of cycles for each appliance. Number of appliance cycles classified correctly by the AUT is represented by the vector:

$$\vec{\hat{n_a}} = [\hat{n}_{1,a}\hat{n}_{2,a}\hat{n}_{N_A,a}]^T \tag{2.12}$$

From this the *appliance cycle classification rate* can be calculated as:

$$ACCR = \frac{\sum \vec{\hat{n}_a}}{\sum \vec{n_a}} = \frac{\sum_{j=1}^{N_A} \hat{n}_{j,a}}{N_c} \tag{2.13}$$

Note that the appliance classification rate is difficult for devices as televisions, washing machines which are turned on and off, but eventually there may be other events between the turn-on and turn-off due to their autonomous behavior. It is very important for an algorithm to not provide only correct event classification, but also to detect whole appliance cycle by providing information on the appliance's duration of usage and energy consumption.

– Energy correctly assigned Signal's total energy is defined as:

$$W = \frac{1}{f_s} \sum \vec{p} \tag{2.14}$$

with *power vector* $\vec{p}$ and the sampling rate f_s. The energy consumption for each appliance is represented by the vector $\vec{w_a}$.

$$\vec{w_a} = [w_1 w_2 w_{N_A}]^T \tag{2.15}$$

Note that the *detailed ground truth data* is required for computing the vector. The energy correctly assigned to each appliance by the algorithm is given in the vector $\hat{\vec{u}}_a$.

$$\hat{\vec{u}}_a = [\hat{w}_1 \hat{w}_2 \hat{w}_{N_A}]^T \tag{2.16}$$

The correctly assigned energy rate is given by:

$$AER = \frac{\sum \hat{\vec{u}}_a}{\sum \vec{w}_a} \tag{2.17}$$

Normally there is more energy from the signal recovered that covered by the ground truth data. Therefore the *total assigned energy rate* is"

$$TAER = \frac{\sum \hat{\vec{u}}_a}{W} \tag{2.18}$$

Depending on ground truth data, even this measure can be a weak one. This make the quality of data taken at ground truth of special importance, for achieving highest performance analysis possible for the algorithm under examination.

2.3.2 Measurement frameworks/tools

- NILMTK
  NILMTK is an open-source toolkit for energy disaggregation. It is written in Python as part of an open-source project. The main purpose of its creation is to make the life easier for the researchers to conduct high quality activity by:

  1. Providing a common set of pre-processing tools, benchmark NILM algorithms and disaggregation metrics to help in validating the performance of their algorithms in a standardised way.
  2. Developing, alongside with the community, a standardised way to publishing algorithm code
  3. Provide a standard dataset format and provide converters for other formats to the standard one.

NILMTK Overview

An overview of the NILMTK pipeline is shown 2.2. A set of pre-processing functions are provided to clean up imperfections or to re-sample data to a different timebase. Datasets are first converted to NILMTK's standard data format which is based on HDF5 file format. Users can invoke NILTK's dataset statistics functions to identify issues with their dataset or compute informative statistics. A number of benchmark algorithms are available which can be trained on disaggregated data and after models are inferred, aggregated data can be disaggregated. In the end, NILMTK provides a set o standard metrics function to compute the disaggregation performance [18].

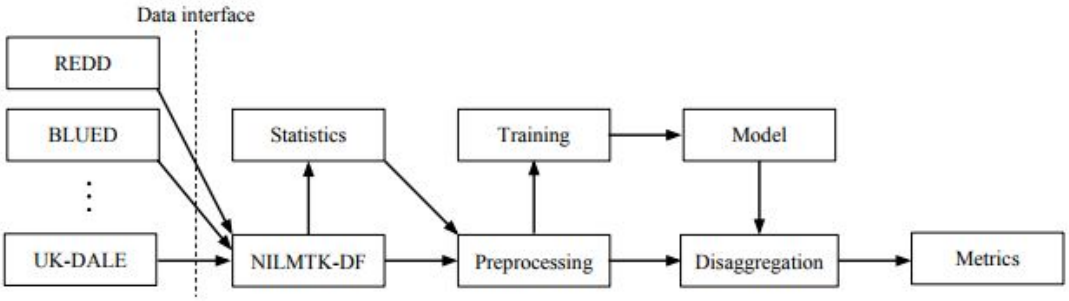

Figure 2.2. NILMTK Pipeline (Taken from [18])

Accuracy Metrics

Because of the diversity of application areas of energy disaggregation a range of accuracy metrics are required. NILMTK provides a set of metrics which combine general detection metrics and those specific to energy disaggregation.

1. Error in total energy assigned
   Difference between the total assigned energy and the actual energy consumed by appliance n over the entire data set
   $$|\sum_t y_t^{(n)} - \sum_t \hat{y}_t^n| \tag{2.19}$$

2. Fraction of total energy assigned correctly
   The overlap between fraction of energy assigned to each appliance and the actual fraction of energy consumed by each appliance over the data set.
   $$\sum_n min(\frac{\sum_n y_t^{(n)}}{\sum_{n,t} y_t^{(n)}}, \frac{\sum_n \hat{y}_t^{(n)}}{\sum_{n,t} \hat{y}_t^{(n)}}) \tag{2.20}$$

3. Normalized error in assigned power
   The sum of the differences between the assigned power and actual power of appliance n in each time slice t, normalized by appliance's total energy consumption.
   $$\frac{\sum_t |y_t^{(n)} - \hat{y}_t^{(n)}|}{\sum_t y_t^{(n)}} \tag{2.21}$$

4. RMS error in assigned power
   The root mean square error between the assigned power and actual power of appliance n in each time slite t.
   $$\sqrt{\frac{1}{T}\sum_t (y_t^{(n)} - \hat{y}_t^{(n)})^2} \tag{2.22}$$

5. Confusion matrix
   The number of times in which each of an applieance's states were either confused with every other state or correctly classified.

6. True positives, False positives, False negatives, True negatives
   The number of time slices in which appliance n was either correctly classified as being in on TP, classified as being on while it was actually off (TP), classified as being on while it was actually off (FP), classified as being off while it was actually on (FN) and correctly classified as being off (TN).

$$TP^{(n)} = \sum_t AND(x_t^{(n)} = on, \hat{x}_t^{(n)} = on) \tag{2.23}$$

$$FP^{(n)} = \sum_t AND(x_t^{(n)} = off, \hat{x}_t^{(n)} = on) \tag{2.24}$$

$$FN^{(n)} = \sum_t AND(x_t^{(n)} = on, \hat{x}_t^{(n)} = off) \tag{2.25}$$

$$TN^{(n)} = \sum_t AND(x_t^{(n)} = off, \hat{x}_t^{(n)} = off) \tag{2.26}$$

7. True/False positive rate
   The fraction of time slices in which an appliance was correctly predicted to be on that it was actually on (TPR), and the fraction of time slices in which the appliance was incorrectly predicted to be on that it was actually off (FRP)

$$TPR = \frac{TP}{(TP + FN)} \tag{2.27}$$

$$FPR = \frac{FP}{(TP + TN)} \tag{2.28}$$

8. Precison, Recall
   The fraction of time slices in which an appliance was correctly predicted to on that it was actually of (Precision), and the fraction of time slices in which the appliance was correctly predicted to be off that i was actually on (Recall)

$$Precision = \frac{TP}{(TP + FP)} \tag{2.29}$$

$$Recall = \frac{TP}{(TP + FN)} \tag{2.30}$$

9. F-score
   The harmonic mean of precision and recall

$$F - score = 2\frac{Precision * Recall}{Precision + Recall} \tag{2.31}$$

10. Hamming loss
    The total information lost when appliances are incorrectly classified over the data set.

$$HammingLoss = \frac{1}{T}\sum_t \frac{1}{N}\sum_n XOR(x_t^{(n)}, \hat{x}_t^{(n)}) \tag{2.32}$$

- NILM-Eval

Overview

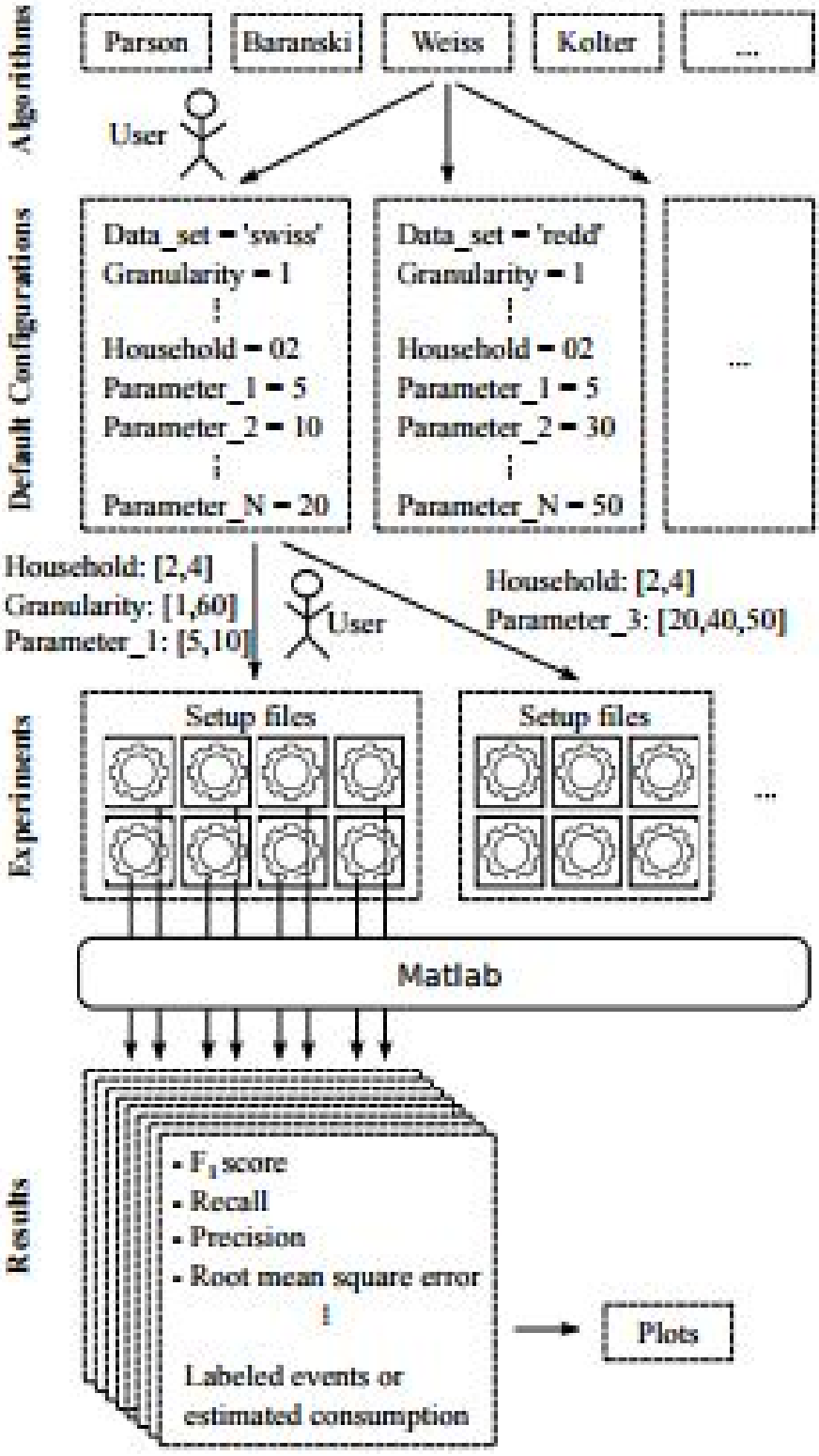

Figure 2.3. NILM-Eval Pipeline

It is a NILM toolkit written in MATLAB for evaluating NILM algorithms in different scenariois to gain a comprehensive view of their performance. Certain parameters as multiple data sets, households, time periods, and data granularities are encapsulated in configuration phase. By encapsulating those parameters in configurations, it allows a user to evaluate his algorithm on a new data set, and to fine-tune configurations to improve the performance of an algorithm in a new setting. Figure 2.3 shows an overview of the framework pipeline. As input the user provides the implementation of an algorithm and specifies the *configurations*. This gives the opportunity to adapt it to the corresponding data set. For each combination of parameters specified by the user, NILM-Eval creates *experiments*, creating a setup file, which then serves as input for the evaluation system. For each configuration, MATLAB creates a new instance, so NILM-Eval scales well (e.g., by running it on a computing cluster). As the end result, it provides for each experiment:

- Value of each performance metrics supported by the algorithm
- Estimated consumption of each appliance or alternatively labeled events
- Plots illustrating the results

Accuracy Metrics

1. Root mean Square error

$$RMSE = \sqrt{\frac{1}{T}\sum_{t}(y_t^{(n)} - \hat{y}_t^{(n)})^2} \tag{2.33}$$

   $y_t^{(n)}$ denotes the actual electricity consumption of n at time t, $\hat{y}_t^{(n)}$ corresponds to n's inferred electricity consumption at time t, and T corresponds to the number of time steps.

2. Deviation

$$Dev = \frac{|\sum_{t=1}^{T} y_t^{(n)} - \sum_{t=1}^{T} \hat{y}_t^{(n)}|}{\sum_{t=1}^{T} y_t^{(n)}} \tag{2.34}$$

   Is defined as the deviation of inferred electricity consumption from the actual electricity consumption over a period of time.

3. True Positives (TP), false positives (FP) and false negatives (FN) An appliance-specific threshold θ is defined. If $y_t^{(n)}$, $\hat{y}_t^{(n)} > \theta$ we consider a true positive, if $y_t^{(n)} < \theta < \hat{y}_t^{(n)}$ we consider it a false positive. Finally if $\hat{y}_t^{(n)} < \theta < y_t^{(n)}$ a false negative.

4. Precision and Recall Precision is defined as:

$$PRC = \frac{TP}{TP + FP} \tag{2.35}$$

   , while Recall as:

$$PRC = \frac{TP}{TP + FN} \tag{2.36}$$

5. F1 Score Calculated as:

$$F_1 = 2 * \frac{PRC * RCL}{PRC + RCL} \tag{2.37}$$

2.4 Business Intelligence

In this section a definition of the Business Intelligence(BI) will be provided, followed by description of common BI technologies. Advantages of using the BI in general and the added value of BI specifically for the energetical sector, with a focus on NILM context are discussed in the latter part.

2.4.1 Definition

Busines Intelligence (BI) software is a collection of decision support technologies for the enterprise level aimed at enabling knowledge to workers such as executives, managers and analysts to make better and faster decisions. While enterprises today collect data at finer granularity, businesses are leveraging their data asset by deploying and experimenting more sophisticated data analysis techniques to drive business decisions.
The term BI is first defined by Luhn[19] in an IBM journal, but no common definition of the term has been agreed upon. Different leading vendors and prominent authors have conducted different studies and have come with different definitions of the term. A comparison of the definitions [20] reveals that they generally fall into three main categories, namely:

- Management aspect
  Focus on the process of gathering data from internal and external sources and on analysing them in order to generate relevant information for improved decision making.

- Technological aspect
  Focus on the tools and technologies that allow the recording, recovery, manipulation and analysis of information.

- Product aspect
  Describe BI as the emerging result/product of in-depth analysis of detailed business data as well as analysis practices using BI tools.

In the end description by [21] is used because it spans over the three categories, and because the focus in technology aspect is more dominant.

> "The processes, technologies, and tools needed to turn data into information, information into knowledge and knowledge into plans that drive profitable business action. BI encompasses data warehousing, business analytics tools and content/knowledge management"

2.4.2 The BI technology

Various BI technologies exist that elaborate the process steps of converting data into information and information into knowledge. [22] in his paper, provides the overview of a typical architecture of and end-to-end BI solution. BI solutions provide a sequential processing of data to transform it into relevant and useful information. It consists of five main elements:

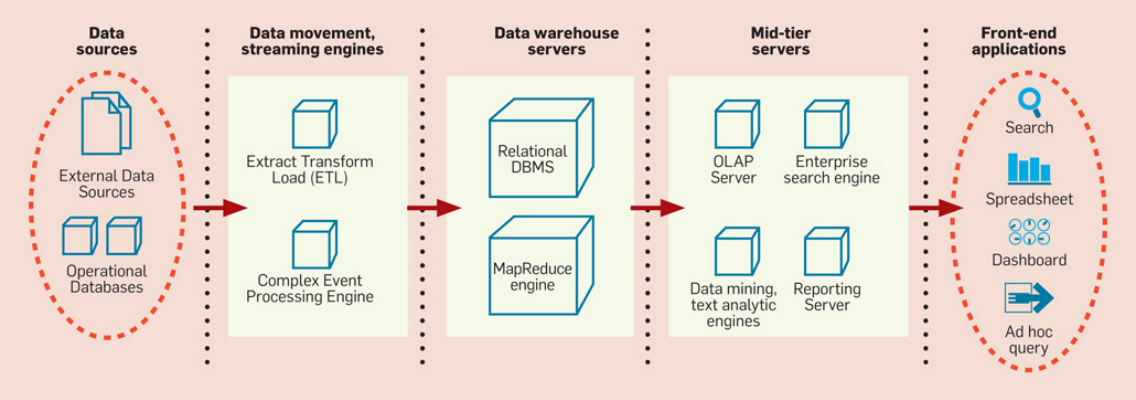

Figure 2.4. Business Intelligence architecture (Taken from [22])

- Data Source
  BI operates with data which come over from different sources. Typically, they come from operational databases across departments of an organization, but also data from external parties can be included. Enterprise Resource Planning (ERP) system is an example of a data source.

- Data movement and streaming engines
  Since there is a variety of sources, they contain data of varying quality, inconsistent representation and formats. Thus the problem of integrating, cleansing and standardizing these data is not trivial. Also, efficient and scalable data loading is imperative for BI. These back-end technologies for preparing the data are called Extract-Transform-Load(ETL) tools. They assist in discovering data quality problems and facilitate the process of loading large amount of data into warehouses. As will be mentioned later, data quality is of great importance in the process of determing the added value of the proposed BI solution. If data presented to the final user is wrong, incomplete or inconsistent may hinder the whole process of decision making. Specialized engines like Complex Event Processing (CEP) have emerged to support near real time BI tasks. In several cases, it is extremely beneficial to have real time access to data that is visualized in such a mode to support fast decisions.

- Data Warehouse servers
  Operational data is loaded into repositories called *data warehouse* that are managed by one or more warehouse servers. Subject oriented means that the data warehouse is designed around the main subject that concern business, in

order to facilitate analyzing of data. Integrated refers to the fact that data originates from different sources. Time variant describes the evolution over time and is not focused just on the recent data. In the core of data warehouses lays the concept of *multidimensional model*. This model is created to support data analysis. The data in this model is classified as a "fact" which is associated with a numerical value or as being a "dimension" that describes the facts and are mostly textual. Dimensions are used for selection of data and grouping of data at a desired level of detail. Dimension is organised in hierarchy of levels, where each level represents a level of detail that is of interest to analyze. On the other hand, facts are objects that represent the subject of the desired analysis. Every fact has a certain granularity, determined by the levels from which its dimension values are gathered.
Normally engines choice for storing and querying warehouse data are divided into two groups. The first group is represented by the Relational database management systems (RDBMS). New data structues, optimizations, and query processing techniques have developed for executing complex SQL queries over large data as a key requirement for BI. Need to architect low-cost data platforms that support larger data volume have given rise to the *"Big Data"* challenge. Big data challenge pushes to create new platforms that support much larger data volume than the typical RDBMSs. Such platforms belong to the second group and are based on MapReduce paradigm [22].

- Mid-tier servers
  Data warehouse servers are complemented by a number of mid-tier servers, with the function of providing specialized functionalities for different BI scenarios. Online analytic processing (OLAP) servers efficiently expose the *multidimensional* view of data and give the opportunity for common BI operations as filtering, aggregation, drill-down. This way the multidimensional model is presented directly to the users, and performing directly the mentioned operations. Newer "in-memory BI" engines are exploiting large main memory sizes to improve performance of multidimensional queries. Several other mid-tier servers can be included in one of these categories:

  – Reporting servers
    Enable definition, efficient execution and rendering of reports. Importance of text data such as product reviews and emails have seen a increase in importance of these mid-tier servers.

  – Enterprise search engines
    Support keyword search paradigm over text and structured data in the warehouse.

  – Data mining engines
    Enable in-depth analysis of data that goes beyond that OLAP or other reporting servers can provide. Also the ability to create predictive models makes them very valuable.

  – Text analytic engines
    Have the ability to inspect large amount of text data and extract valuable information that would have required large and significant manual labour.

- Front-end applications
  These applications are used explicitly by the business users to make decisions. There are several popular front-end applications through which user perform BI tasks: spreadsheets, engine enterprise portals, performance management applications that enable decision makers to track key performance indicators of the business using visual dashboards. The latter will be on the front focus of this thesis. The more flexible the tool the more enables dynamic exploration of patterns. Reason why dashboard is chosen is because it could be highly dynamic by allowing the user to drill down to the data of particular KPIs and provide different level of aggregation. Added to that the fact with minimal time and effort we can speed up the process of taking decisions and apply efficient actions in response.

In addition, there are other BI technologies such as Web analytics, which enables understanding of how visitors to a company's Web site interact with the pages. For example which landing pages are likely to encourage the visitor to make a purchase. Likewise, vertical packaged applications such as customer relationship management (CRM) are widely used. These applications often support built-in analytics, for example, a CRM application might provide functionality to segment customers into those most likely and least likely to repurchase a particular product. Today, it is difficult to find a successful enterprise that has not leveraged BI technology for their business. Another nascent but important area is mobile BI that presents opportunities for enabling novel and rich BI applications for knowledge workers on mobile devices. The following technologies will not be elaborated any further as they are not included as part of this thesis.

2.4.3 BI added value for the business

According to [23] the value of the BI is created by acting on the information and the knowledge provided to the organization, otherwise the BI in itself has no real value, with the information integrated into a decision in order for its value to be assesed. Two are the main purposes for measurement: Determing the value of BI and managing the BI process. According to literature the expected benefits of the first are to cost-justify BI services and demostrate the actual effects of the BI and also increasing credibility of BI as a managerial tool. For the latter, the main goal is the continuous improment of the BI products and services.

Determing the added value of of BI solution to a business is a challenging task. This is due to the fact that business benefits are often difficult to measure especially in certain parts of the organization which are not related directly to the income properties and intagible with other factors [24]. Measuring the value of IT remains a complex task due to a lack of understanding of the processes responsible for realizing benefits. The way BI is applied in different companies can vary a lot. Case-specific issues are likely to have an effect on how BI can be measured. Also measuring the benefits of BI is not as simple as measuring the costs. There may be a time lag between the production of the BI and the financial gain. Many of the benefits realized as a result of BI initiatives are non-financial and often intangible, such as

timely delivery of information and improved product or service quality, and whilst they should lead to financial, measurable benefits, a time lag may make measurement difficult[23]. Also measuring the monetary value by calculating the return on investment (ROI) is also trivial, because of the output of the BI process, which is some kind of processed information. Therefore, measurement in practice is quite difficult. Nevertheless, several measurement models have been proposed. [25] proposed an extended model consisting of five processes that are used for the evaluation of the BI solutions, as depicted in 2.5. The first process is the *IT alignment process* which

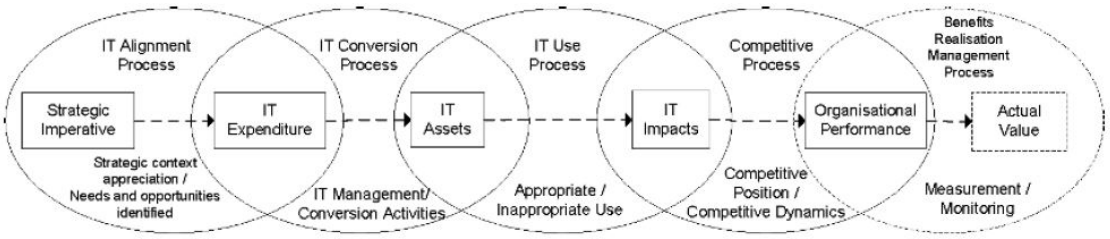

Figure 2.5. How IT creates business value - a process model

includes tasks as identification of opportunities and development of an IT stategy, aligned with business goals. Secondly, the *IT conversion process* is used to address acquisition of BI products and services and their deployment capability. Thirdly, the *Use process* represents the activities that are necessary to ensure that BI assets are used appropriately. Impact of activitites in this process may result in better organizational impacts such as new products, improved business processes and better decision making. Fourth, the *IT Competitive process* focusses on the benefit of the organization from the implementation of the newly improved products and services. Finally, the *Benefits Realization Management process* makes the prioritization of BI requests simpler and more effective by linking priority with the expected business benefits. The conclusion from this proposed model is that:

> "Realization of business value from BI is highly dependent on activities that occur in five stages of the process model - from the alignment of the BI strategy with that of the organization, through to the way in which the business benefits of BI are measured"

A strong partnership established between the business and the IT BI teams is fundamental for success. Althougth the model helps in highlighting the benefits, the measurement of individual benefits is still challenging due to the delayed and indirect nature of the benefits. Also the overlapping of tasks between consecutive processes leads to difficulties in assessment of the added value.

2.4.4 Application of BI in ICT energy sector

BI is applied in every industry including ICT and energy industry. The ICT energy provider companies can take full advantages of all BI technologies including reporting tools, data visualization and data mining tools. Usage of such tools is reflected into a

reactive time for preparation of reports, guarantees faster and more direct access to the information needed for decision-making, analyses the flow of the business across the services and clients. Specifically for the ICT energy sector it can lead to service improvement by detailed analyses and reports about the processes involved in the realization of a particular service. Also, it can help in improving the organizational support functions by providing support for their decision making processes.

From the management level point of view, BI technologies are used to create dashboard reports on KPIs. For each metric, a target can be set and benchmarked and alerts can be triggered if any of the KPIs falls below the desired threshold. Having an overview of the organizational performance, the management will have a better understanding of the current capabilities of the company, and make better strategic decisions in the future. From the financial point of view, BI could be used to carry out budgetary analysis. This might include analysis of confrontation between the budgeted and the actual expenditures. By using BI solutions in human resources functions an integrate view of the workforce can be provided.

2.5 Single Page Application

2.5.1 SPA Key Components

All Single Page Applications(SPA) have two key components which are present in all newly developed SPAs like *AJAX* and *REST*, while the rest are SPA-specific.

AJAX

AJAX is the main technology used nowdays to make rich client applications. It gives the possibility to an application to update the DOM data without refreshing the whole page, in contrary to traditional web applications which force users to wait until a response from the server is ready when transitioning to a new view. Capability to allow user to interact with the page during the process, showing progress and status, highly improves the user experience with this kind of applications [26]. The whole technology is based on the most known core components like:

- HTML
- CSS
- DOM
- XMLHttpRequest object
- JavaScript

From a global point of view these components can be described as follows: HTML and CSS occupy the rendering process, DOM is responsible for dynamic displaying and interaction while XMLHttpRequest for data exchange with JavaScript as the

glue for the whole logic. Today, mostly all important web browsers support the above mentioned components. Applications based on this technology are created based on state changes, rather than keeping sessions on the server and rendering static pages according to the session status. Second advantage relies on the reduced amount of data exchanged between the client and the server. From the asynchronous nature of AJAX, interactivity on the client side is increased by performing several operations at once.

REST

Representational State Transfer (REST) represents an architectural style used in conjuction with AJAX applications. The main purpose of REST is to make application data to be accessible in components known as resources. More specifically the REST architectural style can be descibed by five features:

- Resource Identification
  All resources must be mapped to an Uniform Resource Identifier (URI)
- Uniform Interface
  Resources are accessed by the same interface, typically HTTP
- Self-Describing Messages
  REST uses well-defined resource format like XML or JSON for data exchange.
- Hypermedia Driving Application State
  Clients which consume REST service must follow links found in the resources.
- Stateless Interaction
  Request are self-contained, thus containg all required information to be included in a single request.

Web Applications communicate with REST services via standard HTTP protocol. Since the HTTP protocol defines several sematics for the operations called *verbs*. Verbs have the property tha implies whether they can or not change the resource's state or data and provide idempotency, which means multiple requests do not cause an effect which is different from sending a single one. The main advantage of using REST services is the fact that allow loose coupling of the server and client, like proving a REST service to be consumed not only by a web application but also from by native mobile and desktop application.

2.5.2 Design Principles

This section lists the reasons why a Single Page Application architecture style is a good choice for implementing the desired dashboard mentioned in the requirements.

- Native-like Application
  The ability of a SPA to revamp the structure of DOM tree and manipulate

the model to be incorporated with the view without a page refresh gives the impression of a native-like application while using them. The current views remain responsive and the end-user can perform action while other events take place. It gives the user the impression that the applications is reacting faster and improves the user experience.

- Separation of Concerns
  This software engineering concept is used to provide modularity for a given application. By proving modularity, the principle forces the use of a pattern that decomposes software behaviour in encapsulated units. In this way we can group together logically related elements, which result in low coupling, reduced code complexity. The aim is decomposing the application into its respective units and making each unit responsible for all of its functionalities. In SPA this is done by applying the MVC pattern. MVC pattern is a common software architecture pattern that separates the visual data representation from the underlying model. Several derivatives of MVC have emerged like MVVM(Model-View-View-Model) which is MVC used in conjuction with two-way data binding. Thus implementing these patterns requires to include component that define templates, data models and controllers. Templates are in plain HTML writen in a separate file or inside JavaScript components. Data Models are typically written in pure JavaScript objects or extends skeleton objects provided by the framework. Controller pattern is defined by the framework, with the main purpose of acting as a glue between the Views and Data Models. In other words, the model is the representation of the data model fetched from the server side, views are templates combined with data model to genereate meaningful front-end with the help of DOM, while controllers help bring the business logic to front-end of the application and bind everything together.

- Responsive and Reliable
  SPA controller is not responsible only for data validation but also deals with the business logic implemented at client-side. The business logic resides in the SPA controller, making interactions with the server-side minimalistic. In this way the application is responsive to user interaction. In addition the interaction that takes place with the server-side is also kept asynchronous in order to avoid any delay in response. The data model is stored in the browser's storage and synchronized with the server when necessary.

- Cross-platform
  Browsers are the most effective way to interact with a web application. They do not require special instructions to be executed in different systems and any prerequisite to be installed on the system. SPA has everything it needs just present with a help of a browser.

- Easy Deployment
  Effort needed to deploy an SPA is minimal. Javascript is available in all browsers and hence attaching the appropriate Javascript files with the application would make application executable on any browser.

- Testing
  Modularization of the SPA comes with an added advantage that these individual chunks are written in such manner that they are loosely coupled, and hence, unit testing is easier compared with traditional web applications which binds data, view, and logic together. End-to-end testing frameworks are also available to favor SPA, as unit testing in itself is not enough.

- Routing
  Module which delegates requests from predefined URIs to proper functions. Requested URI is assigned a Controller and passed parameters from the URI. Since finding resources in web is based on URIs, this is essential in a newly designed application and different from traditional web applications since they do not allow to save application state in an URI. Routing is non-trivial since a "history" feature has to be implemented, which provides the option to go back and retain the previous visisted history.

Chapter 3

NED Technical Architecture

In this chapter the mini objective O4 defined in the problem definition section 3 will be addressed with the help of a case study. Case studies are appropriate where there is limited amount of existing knowledge and when a phenomenon is broad and complex. A single case study facilitiates an in-depth investigation. The main selection criteria for the case study is conform with the one proposed by [27], standing as an exemplary case in a wider group of cases. The case study represents a company which operates in energy sector by providing detail information about energy usage to its consumers and helping in activating consumption saving strategies.

3.1 Company Presentation

Midori is a startup company founded in Turin, Italy, at the end of 2011 and hosted at the Innovative Companies Incubator of the Polytechnic University of Turin. The company operates on the ICT energetic sector, more specific in the energetic efficiency field and maintains a profile which stands at the mix of technological and social interaction. It has emerged in the energy market thanks to the proposal of an innovative smart metering tool for both companies and residential users.

For tackling the challenging problem of eletrical energy conservation in today's world, one of most popular solution to these problems has been represented by the monitoring of the residential and commercial energy consumption by exploiting Non-Intrusive Appliance Load Monitoring (NIALM or more commonly NILM). It represents a low cost technology based on the analysis of the signal at the main electricity control panel, that aims, from the point of view of the users, to provide a tool that allows the knowledge of the energy consumption of each domestic appliance and, from the point of view of the utilities, to consolidate a technology that permits the monitoring of the customer's behaviour and consequently to cope with energy waste, by designing a more efficient distribution plan.

3.2 Architecture Overview

NED is a state-of-the-art *load monitoring technology* designed with aim of offering a detailed feedback on appliance-level to the end user. It consists of an hardware and software part: the hardware is represented from the metering device, which is needed to be installed at the electrical panel level, while the software part is represented from the algorithm used for appliance recognition. The fact of using the power consumption as a key feature for the disaggregation process requires a Data Acquisition Unit. The data acquisition unit (DAU) is equipped with a meter that is able to record current and voltage at low sampling frequency rate (1 Hz) and its task is to detect the most significant switching events. Previous version of architecture included also measuring the voltage at the electrical control panel. This measurement hampers a little bit the concept of "Non-intrusiveness" since requires a more intrusive procedure becaused for its realization it is needed the access to the power system through a socket, done by the intervention of an electrician. Surveys made

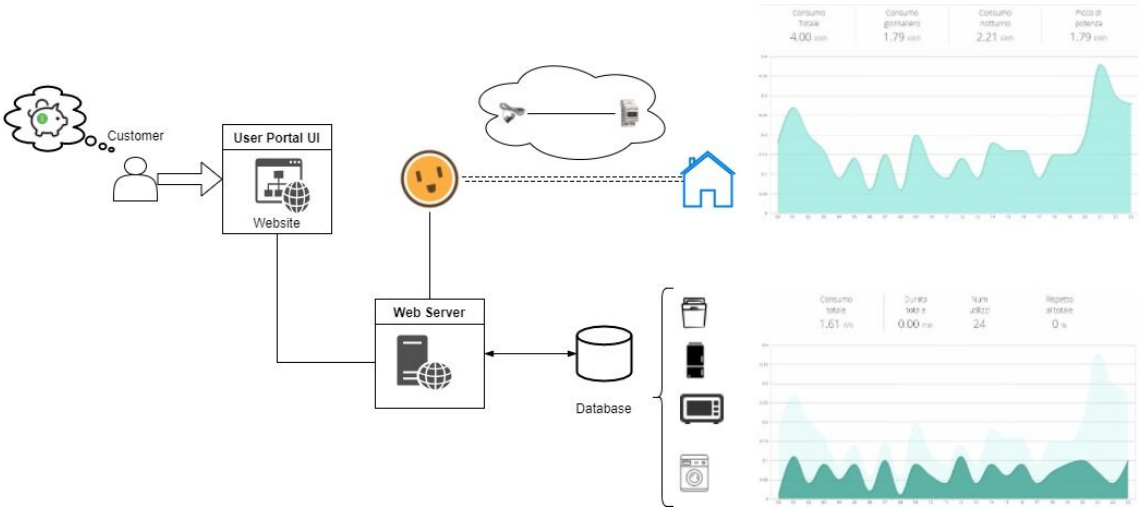

Figure 3.1. NED Architecture

but also external factors are included when the final architecture is to be defined completely. Two aspects which make NILM an interesting choice for management of energy efficiency are possibility to have a "low-cost" and "Non-Intrusive" technology. The importance of the market in the choices of a company is important: it represents an external factor which has to be taken into consideration. Even though from the engineering point of view it can be efficient, it has to suit consumers' needs. While measuring the current consumption does not pose any difficulty, since applying only a current clamp on the electrical wire that goes into the electrical control panel is more than enough. The DAU is equipped with a high frequency meter. The high frequency meter does not transmit continuosly the current measuement to the central server but implements a microprocessor locally which computes the harmonic features and outputs with a rate of 1 Hz. The architecture intends to use current and its harmonics for determination of the signature space, instead of using real and reactive power, with the aim of simplifying the architecture to allow a reduction of the costs and the intrusiveness of the first approach. The real power

$P_{(t)}$ and the current contain similar information. While the reactive power $Q_{(t)}$ can be extracted easily from the analysis of the harmonics of the current. A detailed view of the architecture is shown in 3.1

3.3 Current situation

At the beginning of the work, at Midori did not have any previous tool for measuring the performance of the NED algorithm and also tracking its progress in correlation with pre-defined goals. The whole process for determining the accuracy of the data was a labour process done by the developers with the help of Excel spreadsheets. This brings several disadvantages as high debugging time spent for tracking mistakes encounter during the disaggregation process. Also using Excel spreedsheets or CSV files for keeping track of previous errors or setting the newly defined targets it is not a trivial process, which most of the time results in poor maintability.

3.4 Desired situation

The initiative behind the idea to create a web-based application in the form of a Dashboard for measuring the performance of the under development algorithm was to improve efficiency with respect to the developed NILM algorithm as the fundamental block of company's product which was NED. The strategic goal of the project are to achieve higher accuracy in identifying losses and also improve the identified improvement gaps. As a direct consequence of the newly developed tool is to improve drastically the convergence times for achieving objectives in minimal time with respect to the older techniques which were used precedently. Midori stongly believes that a web-based platform for measuring performance of its algorithm will result in significant benefit for its developers for improving the above mentioned benefits, in spite of extra costs for developing the new application. The desired situation would be to have a long-term tool for measuring the effectiveness of the algorithm. Preparation of *ground truth* data for comparison is also an important aspect to be considered in the designing phase. Also of high importance is the tracking of the progress done during a certain interval of time, for providing to the manager a panorama of the progress done based on the last improvements.

Chapter 4

Analysis and Design

When we divided the pivotal question into chunks, one of the pieces was defining a prototype which represents the added business solution to the stated problem. Business solution in this thesis is defined in form a Dashboard. This chapter gives an insight of the technologies which can be used in the analysis phase for building the required web application. Despite the diffusion and popularity of *traditional* web applications they suffer from an inferior interactivity and responsiveness compared to desktop and native applications. Prior to single page architecture, classic web applications required a refresh of the entire interface due to multi-page design pattern [28]. Single Page Applications aims to enhance the user experience of web applications by improving the UI responsiveness and interaction. The proposed solution is based on single page application architecture. The components that define the architecture are discussed in this chapter and the motivation behind certain decisions taken during the design phase. The proposed solution follows the pattern of a *thick client* and a thin web server.

4.1 Novelty of the proposed solution

The proposed solution hightlights the added value of Business Intelligence and how can it be measured in the overall panorama of a business by providing a support for better and faster decisions. For this purpose the Business Intelligence solution is presented in the form of the Dashboard, which is represented by a full stack application. It is necessary to represent the feedback that this tool generates in the optic of quantitative indicators. For this instance, it is necessary to look through the lens of disaggregation to find the aspect of power traces that are most important and define a set of internal KPIs, which will facilitate the work of developers when comparing two different versions of a developing algorithm. The KPIs used for evaluation have been extracted from a two-phase processs. First they are retrieved during the literature research process, with a special focus on NILM accuracy metrics. In a further step, the retrieved metrics are adjusted to be integrated with those taken from a case study, in order to facilitate an in-depth investigation. Case study represents a company, Midori, which operates in energy sector by providing detailed information about energy usage to its costumers and helping in activating consumption saving

strategies. Their main product, NED, is a load monitoring technology designed with aim of offering a detailed feedback on appliance-level to the end user. The KPIs follow a top-down approach, dy defining a numerical index on top as a mean of measure of the overall quality of the NILM algorithm version from a qualitative point of view. The overall index is calculated as a numerical value by taking into account the following accuracy metrics. In our case, the data gathered during the sub-metering process is used to validate the quality of disaggregation process. The KPIs defined are named Duration Detection Accuracy Index and Consumption Accuracy Index. The first metric determines if the disaggregation is in accordance, especially the time duration, with those reported by the sub-metering process. By comparing the start time and end time retrieved from disaggreation with those collected from the sub-metering process, a comparison on duration accuracy is available. In a further step, we can set an acceptable range of minutes to detect the deviation from those reported from ground-truth data. In the second metric, we go more in depth, by checking the disaggregated value of those retrieved values who have satisfied the first condition. Again this value is confronted within a user-defined range of the value reported from the ground-truth data. If it is included in the range, the disaggregation is considered as correct. The combination of both metrics provides a way in which the disaggregation quality of a particular version is furnished to the developer by the mean of an index.

4.2 SPA Software Process

A light weight software process can be used to develop an SPA that gives opportunity to measure in quantitave way the ability of a NILM algorithm to detect the correct devices.The software process is indicated in figure 4.1 and includes all the below activities:

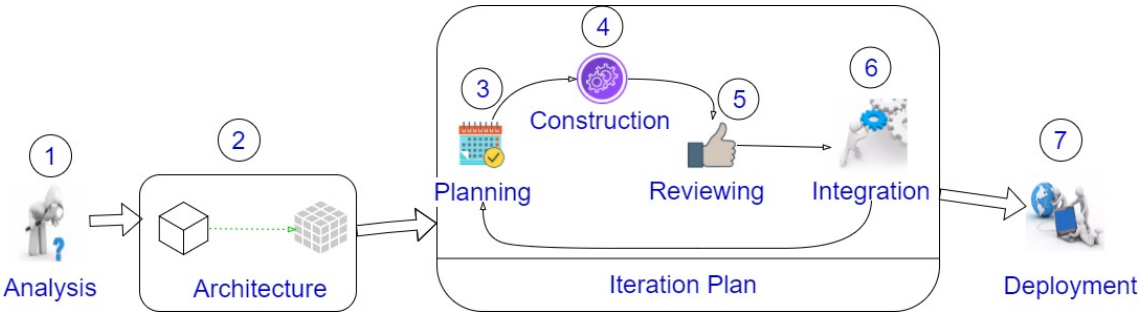

Figure 4.1. Sofware Development Process

1. Analysis
   Idea is understanding the bigger picture of the project and identifying the objectives of the project. Here the business requirements are gathered in this phase. It facilitates the process by breaking the complex ideas into smaller but significant chunks. The solution targets developers and technicians in the field of non-intrusive load monitoring algorithm, in particular those taken from the

case study, and aims to provide a tool which enables them with the capability to assess performance of on-the-hand NILM algorithm, with its apparent benefits and not so apparent costs. The system is thought to be available in all platforms that support a browser, making in this way the web application a viable option. As mentioned before during the literature review is important that the input to be available to the system. As an input the data gathered during the submetering process which provide the ground truth data will be used to valuate the algorithm version quality. Also as input will serve the disaggregated data saved in DB. The confrontration between them will provide the result which will elaborate in a more detail way the performance of the NILM algorithm. In this way, the application can be divided in several chunks and apply multiple scenarios and behaviors expected from an SPA. Scenarios are documented on the basis of the clients' clarifications and discussions about the project. Developer identifies the features and the components from these discussions and generates a number of scenarios that these components or features are expected to perform.

2. Architecture
   After retrieving the user requirement from the analysis phase it is important to define which are the building blocks of the proposed solution. Architecture analysis takes into consideration all the functional and non-functional requirement of the user expressed in the previous phase when defining the building blocks. Here the conceptual components are defined, to be implemented in a latter stage. Componets are created in based of their functionality.

3. Iteration Plan
   On receiving the system design, the work is divided into modules and units and actual coding is started. In this the phase the code is procuded, so it is the longest phase of the software development life cycle. Iteration plan is a fine-grained plan with a time-sequenced set of activities and tasks, with assigned resources, containing task dependencies for the iteration. It also involves the redesign and implementation of a task from the specification list and analysis of the current version of the system. It helps in identifying problems and faulty assumptions at peridiodic intervals. The major goals during this phase are:

   (a) Select/revise scenario set
   Choosing at each iteration the scenarios to be implemented or adding or modifying exististing scenarios to incorporate in the project specification.

   (b) Set iteration goal
   Are the iteration goals to be achieved before the current iteration ends and it includes following goals.

   (c) Schedule work
   Role of the artifact is to map the component from the architecture into an iteration map. The iteration map consists of number of iterations identified as per the schedule and based on the amount of work. The map can be modified by adding more time in the schedule or adding more iteration, extending the project deadline.

(d) Construction
Construction is the activity which involves implementing the high level design using a low-level design, coding and unit testing. The test cases are written based on the scenarios and then, code is written in order to pass them. In development of the SPA, the components are unit tested with Jasmine. Jasmine allows the developer to desribe test cases directly from the scenarios. Jasmine provides functions to help with structuring user-defined tests and also making assertions. AngularJS provides the ngMock module to facilitate mocking of the test cases. This module is used for injecting and mocking the services for unit testing.

(e) Review
Review stage helps in examing the test coverage for the recent iteration completed. If the test coverage is not sufficient, bugs are discovered and are fixed using the above mentioned approaches of development. Code review happens after every iteration and the bugs raised in the review are fixed in the next iteration. The number of defects that occurred in the project listed on the review list is checked to see if matches the historical defect count. If matches, then the developer can confirm that new changes did not introduce any new defects. The test code is checked for correctness and coverage.

(f) Integration
For integrating the existing application with the new changes made in the recent iteration, a regression testing is performed. This kind of testing makes sure that the application functionality remains intact even after the integration. Jasmine is used for performing the regression testing for developing the SPA. In case bugs are found, they are removed in the successive iteration.

(g) Post Mortem
A brief phase to help the developer to prepare for the next iteration. The activities performed in this phase are:

i. Baseline the production code in version control on the local or remote server

ii. Baseline the test code in version control on the local or remote server

iii. Analyze the activities performed in the previous iteration and the correctness of them

iv. Revisit the estimation and planning of the project

4. Deployment Code complete marks the end of the project development. It includes a non-trivial amount of final testing and verification of production code. Once the application is verified to be correct, deploying the application is next step. An SPA deployment involves building the application for production level and pushing it to the production server, if required.

4.3 Specifications

In this section the specifications for the web applications are presented with the scope for a latter implementation. The requirements here express what the system is supposed to accomplish. The functional requirements specify particular results which need to be obtained by the system. In more detail:

- The proposed system should be able to provide a quantitative comparison between the latest versions of the under-development algorithm
- System should provide dynamic and changeable inputs on the duration and consumption when confronting two different versions of the algorithm
- The system must provide an overall panorama of the errors generated during and after the disaggregation process
- System should contain the data gathered during the sub-metering process and provide the ability to user to include other ground-truth data via unstructured interviews, using smartmeters or other means
- System should provide a detailed view of the disaggregation process for a particular house in a particular day, by specifying the detected devices after disaggregation, the timestamps for each detected device and the measured consumption.
- System should check for abnormal behaviour from appliance detected during usage, such as excessive usage detection of an appliance

Functional requirements are supported by non-functional requirements. Non-functional requirements specify the overall characteristics such as having a native-like application, reliable application, modularization for better unit testing and easy deployment of the software. The plan for implementing non-functional requirements is detailed in the next chapter, System Architecture.

The web application, thought in the form of a Single Page Application in its core functionalities will be a dashboard application which should allow the group of NILM developer to monitor and gauge the efficiency of the newly created NILM-algorithm. The principal objective is to help developers in finding eventual bugs and errors retrieved from previous phases of NILM algorithm developement in the fastest way possible. In additon to that a second functionality will be to gauge the improvement or deteriorating with respect to previous versions or with a predefined benchmark. Confrontation with other previous versions gives the developer the opportunity to detect the strengths and weakness of the under consideration version. Furthermore the application should be thought also as a tool to visualize daily graphic for a particular house under investigation for more in-depth analysis. Most common and recent revealed errors should be notified to the user via a notification system. The application has no strict requirement for the platform or performance, since it is used merely as an internal tool by the Midori developers. For this reason the tool advised was thought in the form of a web application.

4.4 System Architecture

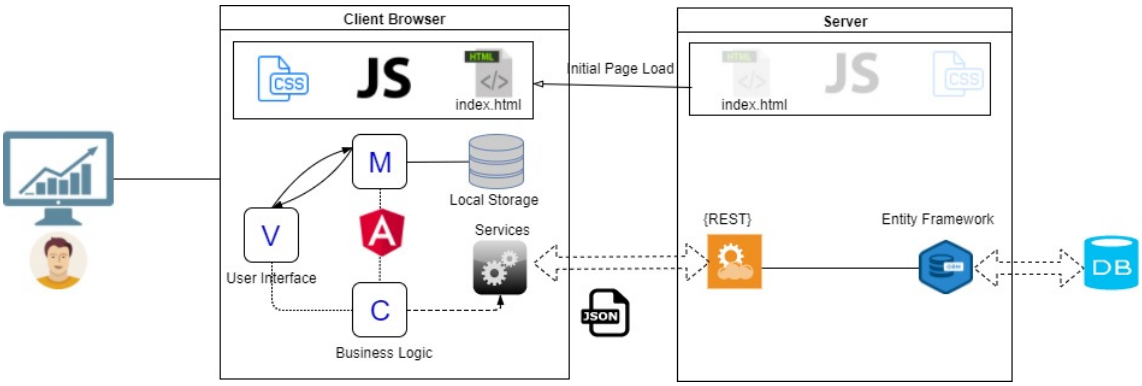

Figure 4.2. Proposed Software Architecture

In this phase the system and software design is prepared from the requirement specifications which were studied in the precedent phase. System Design helps in specifying the hardware and system requirements and also helps in defining the overall system architecture. The system design specifications server as input for the next phase of the model. In the architecture phase, the main objective is to assimilate the information gathered during the proceeding phase and identify the components and features and their functionality. In this phase the conceptual components are divided into actual components. For developing an SPA, a preference is given to a light weight process to facilitate a rapid development of the application. AngularJS promotes the idea of Behavioral Driven Development (BDD), which focusses on how to implement specific behavior in the application. It eases the process of converting user specifications to code that must be written to satisfy their needs. Initially, a spec is written by the developer for testing. The specification consists of the expected behavior from the application and BDD uses whole sentence for describing a specification. The developer then write enough code to pass the test cases written in the specification. The process of the BDD inherits its traits from TDD and unit testing, and forces techniques to write test cases before developing the code to pass them. In our case, it is also important that we gather from user stories when developing the specifics. A possible architecture is proposed in Figure 4.2, which depicts the fundamental blocks of the architecture.

4.4.1 Client

The client is a modern browser which has support for HTML5 and has Javascript enabled. The browser will display a SPA (Single Page Application) to the users. The SPA contains a graphical user interface which have different areas in the interface that displays some type of content. The 4.3 shows an illustration of the SPA. The black arrows represent the HTTP-communication that works in a synchronous fashion. The initial communication takes place when the client initates a request for certain resources to the server, in this case is the SPA and its dependencies. The load of the SPA into the client is a *one-time* transaction. All the necessary

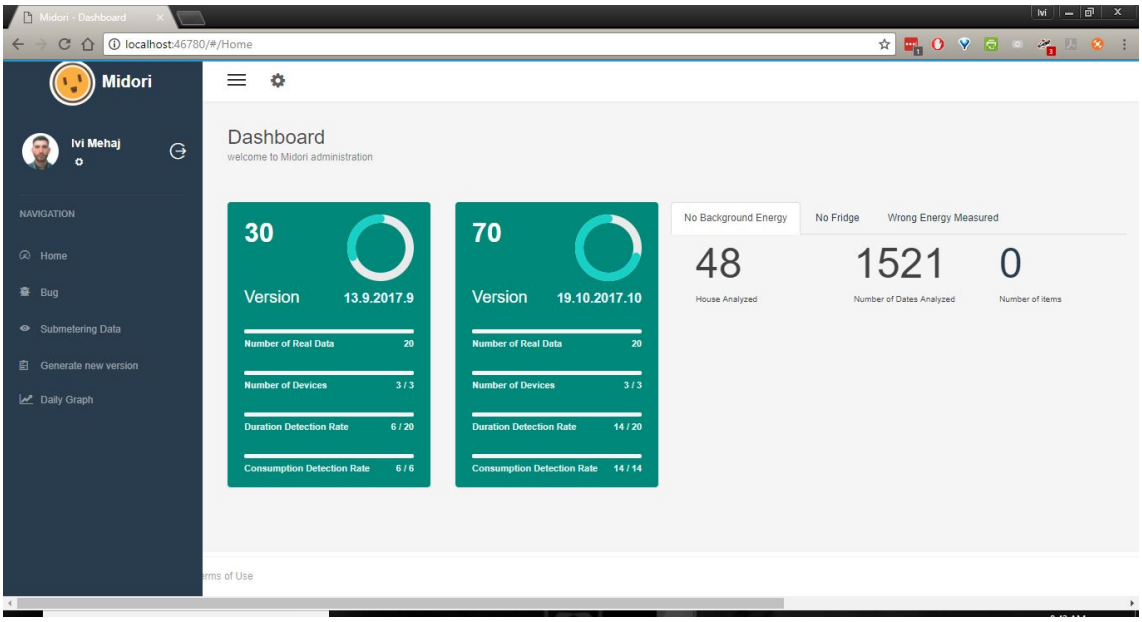

Figure 4.3. Client User Interface

resources needed for the user to run the SPA will be downloaded once, but the successive interactions with application will be faster since the user interface is interactable from a local storage. Since the content of the web pages might be varying, to retrieve new content we can query the web services for specific parts we want to show or update. This process happens behind the curtains with the help of Javascript. This type of asynchronous communication is illustrated by the dotted lines which communicate with the Web Services attached to the web server. The asynchronous nature of the communication allows that communication can also happen directly to the web services of the architecture without the need of the web server isf the information needed to be retrieved is directly from the database. The relationship between the client and the server is a thick client and thin server, where most of the logic is being pushed to the client and leaving a minimal but only that logic that satisfies the requisite of disturbing the service provided by the REST Services. After defining the overall architecture of the application, several choices are neccesary to be taken in accordance to the specifications defined in section **??**.

4.4.2 Javascript Framework

Maturity of Javascript, HTML5 and CSS3 in the recent years have given the browsers the omnipresent in most of the devices in the world and be used as a platform that supports all major applications. For developing an SPA, several Javascript Frameworks exist often with very little differences between them. The choice here for the right Javascript framework was made to take into consideration several factors:

AngularJS Features

- Routing
  In SPA routing plays a vital role in order to bind the correct models to Views and using the defined controllers and in that respect standard routing given by the AngularJS framework this is not quite adequate. For this reason an AngularJS library called AngularUIRouter has been used. Compared to router included in AngularJS, it provides more functionalities to build more complex application. For a better usage experience View are nested into each other, and this library helpg by implementing nested and parallel Views around *states* rather than *routes*. States are like routes, only that the associated URL is optional.

- Modularization
  The ability to divide the application into logical modules is essential in every programming language. AngularJS emphasis very much reusability and testability which are very releated to the concept of modularization. In more depth, AngularJS framework approaches modularization via a technique known as *Dependency Injection*. This technique is not a standard in SPA, but rather an internal implementation used by AngularJS. Its functioning is based in defining modules as functions tht are given a name, which are then parsed by AngularJS. This technique makes it effortless to write new modules in a way that developers know which other modules they need.

- Data Flow
  As mentioned in the previous chapter, AngularJS implments two-way data binding. The binding of data between the controller and the View in the application is made by the AngularJS $scope service. Templates that are paired with a Controller are given a scope, which is accessible by both: template and the controllers. Thus, a click event in a template result in triggering an action defined in Controller. Controler-View pairs create child scopes to parent scopes. Most hard-coded data is written in the templates directly. Then all the essential content is downloaded via the REST API using *http* service component and propagated to the templates from the controllers.

- UI Components
  UI components in AngularJS are called "*directives*". They are controller-view couplings that can be given a custom HTML name. Directives are an interesting technique to achieve modularity and testability. They are in place with the concept of "separation of Concerns" defined in **??**. Almost all UI components used more than once, are directives.

AngularJS has many components that work in harmony with each other such as directives, templates, repeaters, modules, controllers and much more. These section discusses theese components as fundamental core for the developing a SPA

AngularJS Building Blocks

- Templates

Template is an HTML web page which consists of AngularJS directives and artifacts. In AngularJS is a combination of directives, expressions, filters and controls that combine with HTML to form the view.

- Expressions
  Expressions are AngularJS way of incorporating Javascript-like code snippets into templates by using {{expression}} syntax to produce an outcome such as data binding and function executiuon. AngularJS expression do not run on the global window variable of the browser. Instead they are valuated in the $scope variable and do not have access to all functionalities such as function declaration.

- Repeaters
  Web application usually have to display items or collection of elements that users interact with. Repeaters in AngularJS provide a way to iterate over a collection or array of objects including all the data types. A known directive in *ng-repeat* which loops over a collection and display it on the view without any user-defined DOM manipulation.

- $scope
  This is a fundamental block in AngularJs since is the link between the Controller and the HTML view. During template linking phase the directives set up *$watch* expressions on the scope. The *$watch* expression allow the directives to be notified of property changes, which allow the directive to render the updated value to the DOM. Staying loyal to the concept of separation of concerns, the $scope acts as glue between the controllers, directives and DOM. Each of these components has a reference to its respective scope. AngularJS uses the idea of scope hierarchy in which multiple scopes can be created within an AngularJS application, even under the same controller. Thus, defining different behaviors to parent attribute on the HTML view and another to the child attributes and elements. When the child scope finishes to exists, the parent scope value would be used as a fallback.

- Modules
  Modules are an abstraction of a container in which application components such as controllers, services, directives reside. In AngularJS does not exists a main method to initialize the application as traditional web applications do. Instead it provides an *angular.module()* function to create, register and call the modules in the application. It uses a declarative way to specify the bootstrapping of the application. The benefit of using a declarative way is the fact it is easier to develop and understand by the developer, easier to utilize the modules as independent piece of software and makes unit-testing and end-to-end testing revelant to particular modules.

- Controllers
  AngularJS uses a controller as a javascript function that has an entry point in the web application. When developer attaches the directive *ng-controller* in the HTML template, when the application starts bootstrapping, the controller attaches the scope with a set of state and behaviours. It is considered a good

practice to use the controller *only for the business logic*, and not manipulate the DOM directly, but rather use directives or service for DOM manipulation. In this way we keep the application loosely coupled.

- Components
  Components are treated as a special form of directive. Unlike directives, they use the method *angular.component()* to attain functionality of a directive and controller. It helps creating HTML elements which are independent and encapsulated. *Angular.module()* is used in an application for creating a component and the resultant component has the ability to attach a controller which can be used for adding data and the view to the $scope. With different components and with their corresponding data and view, AngularJS can generate a modular application. The idea of using a component-based model is enforced because :

  1. Ability to isolate scopes of different components allowing application to be divided in meaningful chunks
  2. Easier upgrading to future versions
  3. Simplicity respect to HTML tag

- Services
  Consists of shared piece of code that exists in a separate file for providing access to a resource or component that is added to the controller using dependency injection. They are not provided to the controllers right away, but are **lazily instantiated**, meaning that the object created by the controller to access the service is injected when needed and during the bootstrapping of the application. These are singleton objects, meaning only one copy of the instance is created during the lifetime of the application.

- Routing
  angular-route is a built-in service provider used for the navigation of the application. It uses hash-bang(#) or HTML5 pushState for routing the application and provides a way to navigate from one state to another. A route in a SPA consists of the path to a service along the with route parameter.

- Event Handlers
  To handle events in a SPA, AngularJS prefers using special directives already implented such as ng-Click. These directives are used to add special behaviour to the DOM elements and are detected by AngularJS by executing the handler code. Developers can implement a directive to handle specific events based on the application.

AngularJS is a javascript based framework used for developing dynamic web applications. It provides the developer with the ability to create reusable HTML directives and components which can be attached to DOM structure without any page reload. It provides

- Data Binding
  Defines the process of establishing a connection betwen the business logic and presentation layer. The two types of data binding processes exists:

  - one-way
    In One-way the View is updated to reflect the Model.
  - two-way
    Introduced by AngularJS, it allows data to flow in both directions: changes in View can update the Model, but also changes in Model update the View. This way any change to the data model is automatically reflected in the view and vice versa. In practice, when several concurrent components are updating the same view, thus change in one model updates the View but also propagate to other components. This powerful technique helps a lot in removing from programmer the task of creating event-handlers to update the View.

- Dependency Injection
  It is the ability to allow various components in AngularJS environment to be created and delegated based on their requirement in the application. AngularJS provides a system which controls the management of the components and can inject various components such as services, directives without caring about their interdependence.

- Dynamic Routing
  This is a module responsible for delegating requests on the basis of a URI to a certain controller which holds the functionality to perform an appropriate action. Controller uses the URI for accessing resources from the server side.

- Directives
  Directives are HTML markers which have the ability to add new behaviour to the existing DOM. AngularJS has an HTML compiler which parses the HTML and AngularjS directives in form of an elements and attributes. Once compiled the associated directive gets attached with the DOM. A lot of built-in directives exist such as ngApp, ngBing or ngController etc. They have their respective functionality as they provide ways to bind data, view and controller together.

Besides the advantages, several disadvantages are also present, such as

- Support for Javascript libraries
  A rich SPA is developed using AngularJS in conjuction with other Javascript libraries to enhance the user experience. AngularJS makes it tough for developers to use directly these libraries and inject with existing AngularJS Services and components. For these reasons, the developer has to explicitly create custom service to make library usable in the AngularJS environment.

- Understanding AngularJS lifecycle
  The learning curve for AngularJS is not a simple one, but rather a rigid one. Developers have difficulties to absorbe easily the concept of AngularJS life cycle and the role of $scope and $watch variable in it. Unaccurate use of watchers in the controller to handle events, can be followed by a decrease in performance of the SPA and also lead to memory leaks. Below is given a brief

explanation on how AngularJS lifecycle is composed. AngularJS applications are detected by the HTML compiler by traversing the DOM and locating AngularJS based tags in the HTML templates. As mentioned before, AngularJS puts a great emphasis on the idea of modularizing the application and separating business logic in the controllers from the HTML templates. Those two are bind together during application lifecycle. The three major phases (depicted

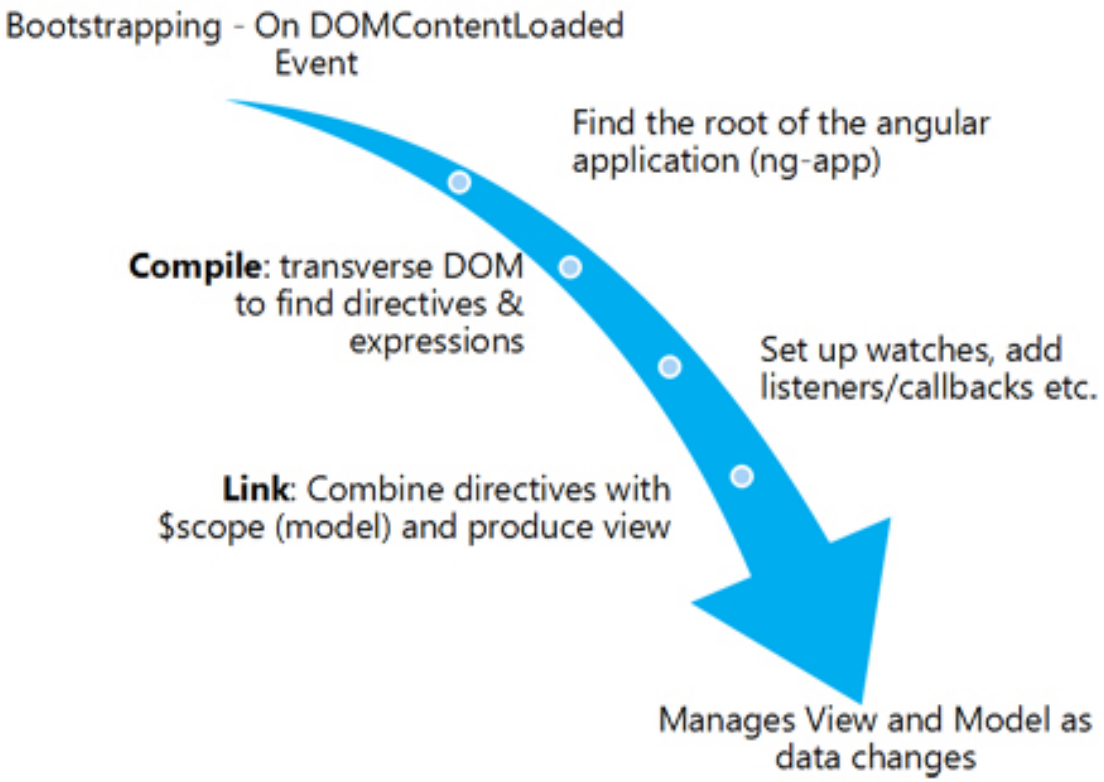

Figure 4.4. AngularJS lifecycle

in 4.4) involved in the lifecycle are:

1. Bootstrap
   This is the first stage of the application lifecycle and begins when the JavaScript library is included at page load. It tries to utilize jQuery library, if present for running the AngularJS framework. Otherwise, it uses an in-built JQLite, which is a compact version of jQuery in the AngularJS framework. Both libraries play a pivotal role in AngularJS because they bring library functions written with best practices in mind. AngularJS initializes itself with necessary components upon the **DOMContentLoaded** event or if the **document.readyState** is set to "complete". Final step will be initializing your application via the *ng-app* directive. Module is loaded and the dependencies are injected into application and respective modules and are available for module usage.

2. Compilation
   Once the application is loaded successfully, the root of the application is detected and the second phase starts. The compilation phase is marked when a *static DOM* is loaded in the browser. Feature of this phase is the two step compilation. In first phase the static DOM is traversed in the background. All the collected directives from the first step, which might inlcude Angular built-in or custom directives are linked with *the scope* objects to produce dynamic view. This linking results in a dynamic view which is driven by the model. During this phase the static DOM is replaced with a **dynamic DOM** that represents the Angular parsed view.
3. Runtime Data Binding
   In the runtime phase, the application completes loading while the services are loaded when requested by the controller. The application persists in this phase until is reloaded or navigated away from. During runtime the application is synchronized with the scope. Any changes in the scope are reflected in the view, and any changes in the view are directly updated in the scope, making the scope the single source of data for the view. Once closed, the scopes and the views of the application are destroyed with it.

4.4.3 Server

The web server is a program made with the intention of serving via HTTP protocol the requested resouces to the clients in accordance to the client-server model. The server is not designed to be CPU-intensive, but rather I/O intensive to handle some user requests and provide a better interaction. Since one the design choices was the separation of the application from the web server, the heavy computations are prefered to be maintaned away from the web server, with the exception of logic verification and request handling. Our server is bound to any type of server type IIS, this making it a general purpose web server. All web Servers need an HTTP-listener and a request handler.

Data Services

Role of the Server in the SPA architecture is to provide Data Service to the client via RESTful service to the client. Its endpoints expose the RESTful services to the client via the HTTP verbs to make it easy to perform the necessary operations. It is beneficiary from the point of view that reduces the number of HTTP calls from the browser to server to retrieve data, in expense that datasize sent from server to client is increased.

Security

But the fundamental task of the server is providing security. You cannot rely on the client for any type of true security. It is in the server side where the real security

checks should occur. Only the view content that authenticated and authorized user can be see is sent down to the browser. It is important to provide security on server side as browser development tools make it super easy to tweak HTML, CSS and JavaScript. It is important to validate on the server-side to ensure that a user has given role/permission and can perform a specific action. Authentication is part of the fundamental 3A in computer security along with Authorization and Accounting. In our ASP.NET MVC 5 application the prefered authentication is the Token, which works like sessions and can be stored within the cookies. Since HTTP protocol is a stateless protocol, tokens are attached to each request. Normally, the token is signed by server and if the signature is validated correctly, the user is granted access.

Chapter 5

Implementation Choices

The chapter illustrates the ideas presented thus far, by demonstrating a single page application called "Midori Dashboard". The client-side of the application prototype is written in Javascript, using the AngularJS framework where the server-side business logic is written in C# according to the architecture proposed in the previous chapter. It is an SPA intented to engineers at Midori for measuring in quantitative way the performance on non-intrusive load monitoring algorithms. The final objective of this SPA is to help the employees to compare diffeent versions of created NILM, for better bug tracking and faster analysis with respect to previous Excel spreadsheets to keep track of the progress, in accordance with specifications defined. This chapter details the core functionalities of the application and elaborates some parts considered as crucial for a well functional application.

5.1 Development Environment

All of the features of the SPA were developed using **Microsoft Visual Studio** development environment. The chosen version was *Community Version 2015*. For the client side the **AngularJS 1.6** version was used, available in the Visual Studio. The server used for the application was **IIS Express** which is a development Server provided by the Visual Studio (version 2015). The RESTful APIs were written using the **ASP.NET Web API**. Web API handles the low-level transport details of HTTP protocol, while exposing the HTTP programming model. These APIs are used to provide various resources to the client application. The chosen language was **C# (version Visual C# 2015)** provided by the .NET framework. For managing the data a Relational DataBase Management System in form of **SQL Server 2015** was already in place.

5.2 Development Tools

Efficient development for a Single Page Application require tools which are different from that of a traditional web application. Such tools help to automatize the development process and also are in line with good software engineering principles.

Continuous integration is a development practice that requires from developers to integrate code into a shared repository several times during the project life cycle. By integrating the application at the iteration boundary, developers can detect errors quickly and locate them more easily. Major benefits of implementing continuos development and integration are as follows:

1. Automate testing and building process of application
2. Increasing test coverage by continuously commiting to repository
3. Provide visibility to other developers across the teams.Determining what went wrong if the build is not successful.

Tools used are:

1. **TortoiseSVN**
   For commiting the recent developments and continuining with good practices of continuous integration, a version control system is necessary. Subversion is an open source version control system, which has had success in the recent years, and since it was an open source it was adopted for our purposes. TortoiseSVN is an Apache Subversion (SVN) client, implemented as a Windows shell extension. It's intuitive and easy to use, since it doesn't require the Subversion command line client to run.
2. **NuGet Package manager v3.4.4**
   For installing the necessary plugins for correct functioning the **NuGet Package manager v3.4.4** tool is deployed. In this way is useful for developing an SPA and allows a continuos development and integration for the project.
3. **Grunt**
   Grunt is a Javascript task runner, a tool used to automatically perform frequently taks such as minification, compilation, unit testing, build process. It uses a command-line interface to run custom tasks defined in a file called *Gruntfile*. Grunt has a large ecosystem and has several plugins developed by other developers with best practices in mind. Gruntfile is a template based approach to define mechanism to perform tasks that are repetitive in SPA development, such as unit testing using Jasmine, generating *"dist"* folder for building production level application, compiling application and deploying on the production server. The script allows to watch and track files constantly and perform a certain number of actions to achieve the desired outcome.
4. **npm**
   npm known as Node Package Manager, which is a package manager for javascript.Its purpose is to make it easy for JavaScript developers to share and reuse code, and it makes it easy to update the code that you're sharing to solve particular problems. These bits of reusable code are called packages or modules. A package is just a directory with one or more files in it, along with a file called *"package.json"* that contains metadata about the package. A typical application, such as a website, will depend on dozens or hundreds

of packages. npm is command-line tool which is bundled with *Node.js*. So, in this way it has a dependency only on Node.js, if you have it installed, then you already have npm.

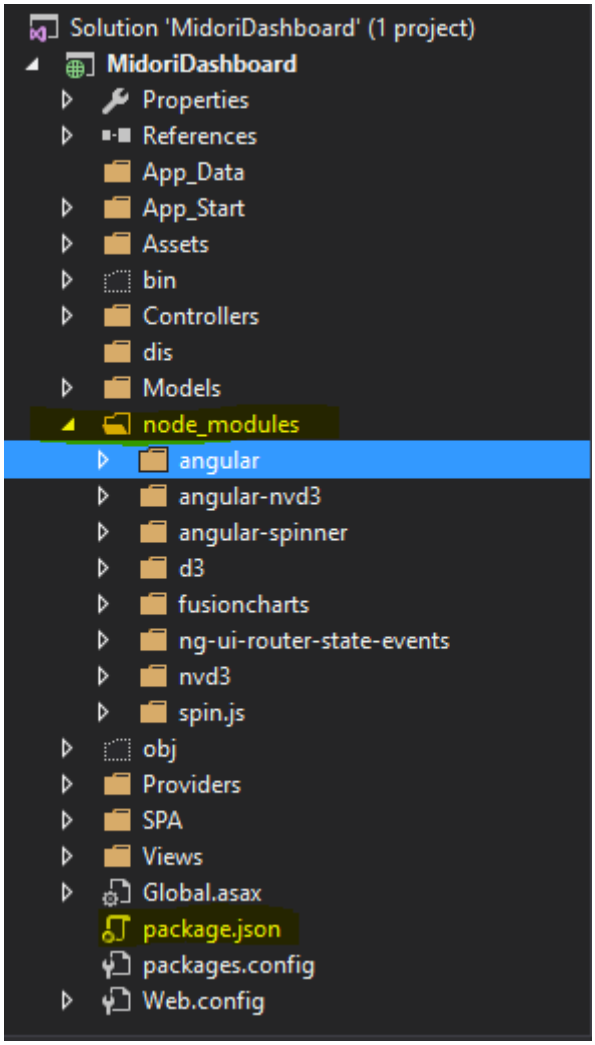

Figure 5.1. Node Modules used in the project

5.3 Application Structure

The mentioned above features as dependency injection, separation of concerns, components and testing are an integral of AngularJS. They help the application to remain modular in design. An SPA must follow these design patterns and concepts during the development process. These concepts are adopted in the process for modularization and the directory structure of the application plays an integral role. When it comes to directory structure of the SPA, all the components could exists in a single javascript file under one folder and the application would still initialize and run properly. From an operation system point of view, running the Javascript code present in single file has advantage respect to multiple files, because multiple

files can slow down the process of executing the code because of the fragmentation in files. But from a developer, point of view, keeping the Javascript code in single file would be disadvantageous in several ways:

- Code maintainability
  There is a lack of logically compartmentalized application where every component is easily located and edited with a single file.
- Debugging
  Debuggin the code is a challenge if the application is not modular in design. The developer has to spend a lot of time locating the appropriate section of the code.
- Testing
  Single javascript file offer a tight coupling in features, which renders very difficult to perform unit testing without affecting other sections of the code.
- Scalability
  Introduction of new features is difficult as there is one file only to work with. Even the version control suffers, especially if a team of developers are working on the same piece of code.

Because of the above mentioned factors is important and a good practice to set a logical division of code for developing the SPA. 5.2 shows the structure of our SPA. The directory structure presented offeres a modular way to develop an SPA following the best practices from AngularJS. It presents a structure suitable for a full stack application which involves development of both client and the server. The application structure consists of several various components:

1. SPA
   Here resides all components that are directly related to the features development of an SPA. The directory is composed of the main pillars which AngularJS supports such as Views and Models. Also the available services are dedicated a folder. Each of the components has in each folder its relative starting with its name and followed by the folder name such as *component-Controller.js* and the view such as *component.htm*.
   - componentController.js
     This module is the main controller for the component which helps setting up the scope variable and performs the necessary data binding.
   - component.html This file acts as a partial template which uses the controller to manipulate the DOM and attach behavior to SPA. The view is created as per the directives used in the template along with the controller logic and data model associates with the $scope.
2. components
   This sub-directory is used for housing components which are reusable by the SPA. It follows the same structure as any other component or directive. For example, since an SPA is composed from a header, a footer and a navigation

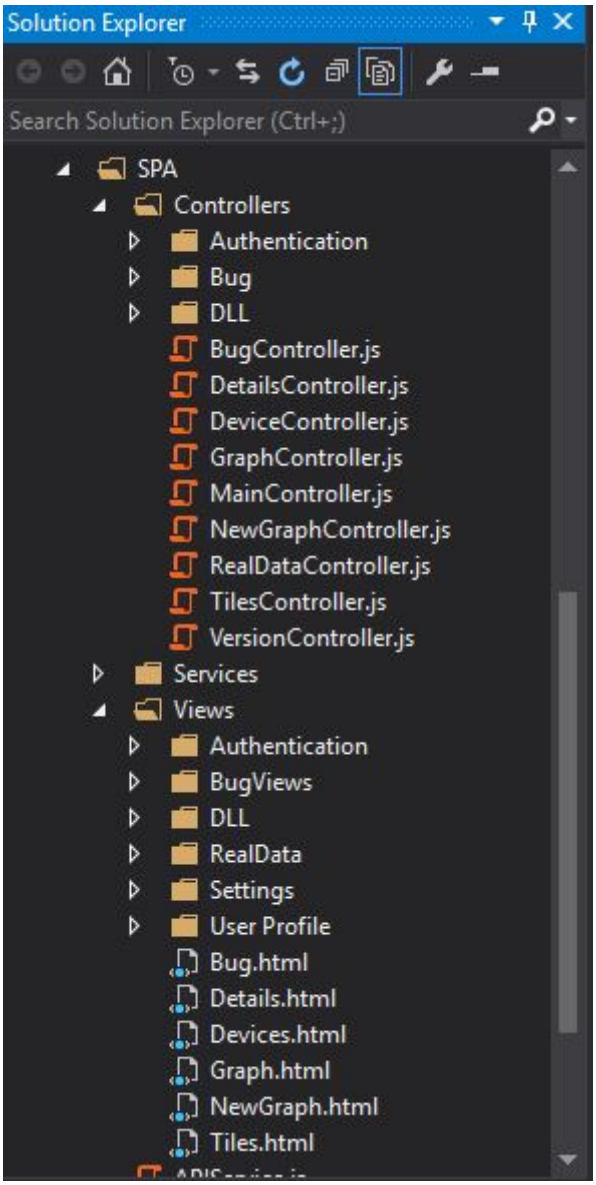

Figure 5.2. SPA Project Structure

feature they are included in every view and listed as reusable components in the client side.

3. assets
   The structure of the assets depends on the quantity of assets used in an SPA. Normally at production level, an asset-heavy will rely on content delivery network for faster retrieval of assets such as images, fonts, and icons. But, for the purpose of our application, the assets folder is sufficient to place all the assets needed such as images, fonts and icons.

4. index.html
   The *index.html* file is identified by the application as the shell of the application which becomes the entry point into the SPA when loaded into the browser.

It consists of the relative path of all necessary files for an SPA and helps bootstrapping the application with the help of the AngularJS framework. Loading begins when a user makes a request to the application root level using the browser URL and an API request is made to the server. Server responds with path to index.html in the client, loading the initial view of SPA and retrieving the business logic and data model in the browser.

5.4 Implementation Process

The development of the dashboard was based on the process illustrated in figure 4.1. The first two phases were described in detail in chapter**??**. For understanding the development process completely, the implementation process need to be explained. The methodology used here is adopted from the agile development methodology and tends to work on a particular feature during a sprint. The lifecycle implementation for each feature has in common:

1. Project Plan
   Project planning involved estimation which played an important role in the development of the application. The objective of planning was to provide the overall effort required for the complete project. All the application features were identified during the project planning.

2. Iteration Plan
   An iteration plan is a fine-grained plan with a time-sequenced set of activities and tasks, with assigned resources, containing task dependencies for the iteration. We used iteration map distribuiting the tasks based on the available time for each calendar day. The objective of dividing the tasks in such way to make sure that one feature would be completed in single iteration. If the feature needed more time for completion, we pushed id to the next iteration by changing the iteration map and accommodating it with other proposed tasks. In order to adjust the change in iteration plan, we either added more time in the calendar for the next iteration or added another iteration based on the backlog generated at the end of the previous iteration.

3. Construction
   At initial stage, the project was set up with the directory structure similar with that of the 5.2 and each feature of the SPA was constructed in the client/SPA folder. In order to begin development of the Midori Dashboard, we started with development of the project overview feature. It involved developing the index.html shelll document to bootstrap the SPA. A shell of AngulaJS consists of a special directive called *ng-app*. This directive normally is attached with the body tag of HTML to recognize an AngularJS application. The *Authentication* directory contains the complete authentication feature. It consists of the controller, the view and the testing file. We used all the necessary test cases based on the user stories discovered in the analysis of the components involved in development of the feature. The construction of the SPA component involves programming AngularJS based template, controller and route. The

components were built after writing the test cases in *component.spec.js*. As mentioned above all the components have a *View* and *Controller* in common which serves as the glue between the View and the Model.

(a) component.html
We construct the template as if we were constructing a static web page. This helped in mocking the user experience of the feature. Once the mock up was ready, we identified the template sections that required data model or event handling and used various AngularJS components or concepts to construct the application.

(b) componentController.js
The controller in an SPA is responsible for handling the client business logic and is continuosly developed along with the template to fulfill the feature requirement. While programming the controller and the template, we continuosly tested the functionality to pass the test cases written for the feature. Once, all the test cases were passed, we got the confidence to mark the feature for completion. After constructing these files to completion, we reviewd the test code for its coverage and made sure that the feature was tested thoroughly. When the user after has been logged in, the application redirects the user to the tiles feature. The routing also makes sure that the transition from one state to another state happens without page refresh. All the other features were constructed in the same way and we made sure that we adhere to the best practices of SPA development mentioned above. With each feature, wiith its own route, we made sure that the application developed follows the single page architecture and gives an impression of a native-like application.

4. Review
It is of high importance to get the feedback from the stakeholders about the desired functionality of the developed component. Near the end of each iteration the feature is shown to the stakeholders to verify if the functionality it provides is aligned with that specified in the requirements by the stakeholders.

5. Post mortem
At the end of each iteration, we understand the difficulties during the current iteration and prepared for the next iteration. If there were some backlog from previous iteration, then we changed next iteration plan and with existing tasks to meet the project deadlines. If during the construction, new features or components were discovered, then we discussed about the features during the weekly meeting and added them to the plan for the upcoming iterations.

6. Deployment
In order to deploy the application, the application server had to be executed in the operating system. The application's *dist* folder was prepared for distribution using **"grunt build"** command in the developed environment. Building is a one-time process for producing the final applicaiton using the grunt task builder and once built, an SPA was available for distribution in the dist folder. It was downloaded from a common repository for utilization. After installing

the prerequisistes and cloning the repository, the application can be used with any browser present on the operating system.

5.5 Features

The scope of the application is to make clear the user the ability to confront different versions of the developed NILM algorithms and enhance user ability to fast track bugs and the progress of the certain algorithm.

5.5.1 Routing

```
var configFunction = function ($stateProvider, $routeProvider, $locationProvider, $urlRouterProvider, $httpProvider

    $stateProvider
        .state('Home', ...)
        .state('Bug', {
            url: '/Bug',
            templateUrl: 'SPA/Views/Bug.html',
            controller: BugController,
            data: {
                requireLogin: true
            }
        })
        .state('Devices', ...)
        .state('Details', ...)
        .state('Versions', ...)
        .state('Bug.1', ...)
        .state('Bug.2', ...)
        .state('Bug.3', ...)
        .state('Bug.4', ...)
        .state('Bug.5', ...)
        .state('RealData', ...)
        .state('Login', ...)
        .state('DLL', ...)
        .state('TimeIntervals', ...);
```

Figure 5.3. Routing using UI-Router

Since the SPA application is separated from the Web server, and we do not ask Web server for content, we need to set up a routing strategy inside the application to know which content is required to be retrieved from the Web server. All this internal routing is done inside the SPA application, in the main file which is located at the root folder of the solution. When the client locates to one of the paths, it will request a View defined by the *templateUrl* parameter. Views are masked by the paths and the full path of the view is not displayed to the client. Althought the action depend on *hashchange* event listener, # symbol can be ommited from the path if the client browser has HTML5 support by using the HTML5-mode in the application provider *$locationProvider*, which stores the application linking paths. This gives the impression of a traditional *desktop application*, but the content is updated dynamically. In our application we use the **UI-Router** for routing. UI-Router is a routing framework for AngularJS. Influenced by the core angular route *$route* and the Ember Route, UI-has become the standard choice for routing app in AngularJS(1.x). It provides a different approach from ngRoute in that it changes

your application views based on state of the application and not just the route URL. This way the views and routes are not tied down just to the URL, but you can change parts of your site even if the URL does not change. This way all the states, routing, and views are handled in just one *.config()*, which helps when using a top-down view as in our application. Figure 5.3 shows there the states using UI-Router are declared in our application.

5.5.2 Authentication

A modern Web application need to have an authentication handler. OAuth stands for OpenAuthentication and relies on concept of authentication via third-party applications to grant access to your service. Different authentication mechanisms in this library take the name of *stategies*. In this application we are using a standard strategy called local-strategy. This strategy verifies users only with credentials which are already present in the database. User sends info to Web application, which by itself sends the user information to the REST Web service for authentication. If authentication is successful, Passport will dispatch a signed JSON Web Token (JWT) to the client. After signed, it is distributed to the client sa it can be saved in local storage for further interactions with the web application. Later on the client uses the service to provide authenticity toward the server.The whole process is done by the authentication service. This service takes care of all logic involving authentication on client-side from logging to removing the token after client signs out.

5.5.3 Dashboard

This feature of the application is the entry point for the developer to interact with the application. It is responsible for giving an overall view of the last two existing version of the developed NILM algorithms and an "on-the-fly" comparison of the best from the two represented by an index. The calculated index takes into consideration the overall Detection rate as specified in literature as the first step. Another step is the found result correspond to a certain timespan limit set in an acceptable way by the developers. The third and final step is controlling if the measured disaggregation energy corresponds to a certain acceptable range which will classify the current disaggregation as an accurate one. Besides that, general statistics are included in the page, as number of houses taken into consideration and latest date of the released DLL. In the main page an overall estimation of the developed algorithms is provided, but also a top-down approach is available to get more details of the confrontation. The approach tends to divide the final calculations per device, by giving a detail information an disaggregation for each measured device. Another ulterior steps is finding the details for a certain house and certain day of a device which shows some unstable state. Also A notification system is available in the right top corner of the screen with the aim to notify the users when a new version of the DLL(Dynamic Link Library) is available or if one of the user has reported some kind of error or uncertainty. In order to retrieve information about an existing feature of

the dashboard, the developer can search the project using the filter option available on the view.

5.5.4 Bug Section

This section is an artifact which helps the developer tracking defects attained in the process. The defect is recorded whenever the developer encounters an anomaly in the NILM algorithm, especially those related with disaggregation process. The errors are divided in four major groups like:

- Background Energy errors
  This kind of errors are generated when the disaggregation process has not worked correctly and the energy
- No Fridge Error
  Midori developers consider the Fridge as an indispensible device and for this reason for every house is considered as a device which is almost present in every house taken into consideration and where the algorithm is being tested. When the dashboard notifies for this kind of error, means that the disaggregated data do not correspond to a fridge.
- Time Errors
  This category includes error which are not anticipated from the developers based on the consumption time, specific for each device. For example, it is improbable that a microwave has had a consumption which has lasted for more than 30 minutes. The necessary steps are taken to verify if there is an error during the disaggregation from the algorithm, or real consumption was not predicted.

In the all above mentioned cases the developer provides the necessary details to notify other users. The details include the type of defect, injected in which activity, removed in which activity, and a brief description about the defect.

5.5.5 Real Data Section

Although NILM represent a non intrusive way to measure the energy consumption, for its better functionality it need the *ground-truth data* for measuring its performance. For this purpose a little bit of "intrusiveness is included" for having data which is considered "real" and with which the disaggregated data is compared for measuring the degree of the performance. Although it represent a deviation from the general principle of non-intrusiveness it is a necessary step for measuring the the quality of under-development NILM algorithm. The ability to upload this file is provided in the application. As soon as the file is uploaded, the data are inserted into the database by using CRUD operations. The upload of the data from sub-metering serves as ground-truth data, which will be focal point in which the comparison will be made. The accepted format is that of a CSV file, with all the data fetched not only from the smarplugs also from interviews and those provided by other partners (Telecom Italia).

5.6 Dependencies/Libraries

During the implementation phase, several UI and utility libraries have been used.

- Bootstrap
  One of the most famous front-end frameworks, it is UI library used for developing responsive websites. An important feature of Bootstrap is that can easily and efficiently scale your websites and applications with a single code base, from phones to tablets to desktops with CSS media queries.

- jQuery
  jQuery is a fast, small, and feature-rich JavaScript library. It makes things like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler with an easy-to-use API that works across a multitude of browsers. AngularJS first checks the included version of jQuery, otherwise it includes an own lightweight version called *jQLite*.

- jQuery countTo
  jQuery countTo is a jQuery plugin that will count up (or down) to a target number at a specified speed, rendered within an HTML DOM element. This element is used to display the index calculated for measuring the quality of the newly developed NILM algorithm.

- jQuery Knob
  jQuery Knob is a javascript library for the graphical manipulation of the images which is canvas based. No png or jpg sprites are included in this library.

- Font Awesome
  Font Awesome gives you scalable vector icons that can instantly be customized — size, color, drop shadow, and anything that can be done with the power of CSS.

- Highcharts
  Highcharts is a javascript charting engine which helps in setting up interactive charts in web pages. In our AngularJS application we use the **highcharts-ng** directive, which is an AngularJS directive for Highcharts.

- ng-bs-daterangepicker This is an AngularJS directive used for Dan Grossman's bootstrap-daterangepicker.

- **Plotly**
  Built on top of d3.js and stack.gl, plotly.js is a high-level, declarative charting library. plotly.js ships with 20 chart types, including 3D charts, statistical graphs, and SVG maps. In our application we use the angular directive of this library called *angular-plotly*

5.7 Server-side implementation

ASP.NET is a server-side web application framework, part of the more general .NET framework, with the scope of developing web sites, web applications and web services. It allows programmers to write code using the supported .NET language as C#, F# or Visual Basic. More specifically, ASP.NET MVC web application framework is used since it implements the MVC pattern, thus building as a composition of three roles: Model, View, Controller. This composition helps the programmer in organizing the structures and interaction of users with the web application. The static routing is set for what is available to be displayed for the clients connecting. After the client connects, the server will serve the index.html page which corresponds to the SPA as depicted in line . After all the routing and conditions are met, the Web Server listens for incoming connections in its public/external IP at the specified port. In addition to the client application, the system consists of the following component: The main application logic is implemented on the client side, and a REST API is used as an interface to send and retireve data between the client and the server. The main functionalities it provides are: The application retrieves the collected consumption for each device and apartment from the database through the API. After retrieving the necessary data it posts updates to the dashboard to update the values of detection for the algorithms for a better decision making process.

1. RESTful Services
   It is a module of the Web Server in the multi-tier architecture. Since we have opted for a thin server, the proposed Web Service is a lightweight one with REST. The show the flow model when an incoming request from the client to the web server for consuming web service is made. The request handler takes the request and checks if the request is valid. Two checks are performed:

   - Request requires certain permissions
     Since the request is available only to authenticated users, the web service will retrieve information restricted only to authenticated users. The level of permission is defined by the token attached to each request from the client. When the data is processed, it is passed to the part of the Web Service that encapsulates the information to a specified format or encoding, which in our case is JSON, preferred due to lowered overhead and readability.

   The REST webservices created in our application follow the standard creation procedure provided by the ASP .NET framework which starts with /api/-controller/api. The routes in which the Web service is using to dispatch the information are configured in *WebApiRoute.config* file. From the authentication section, we do HTTP posts to */api/login*. The controllers which handle the bussiness logic of the web service, are included and passed along with HTTP verbs in the web service. Controllers contain the schema information which filters the requests. Figure 5.4 shows where in the project structure the implemeted REST APIs rest.

2. Entity Framework
   Entity is an ORM(Object Relational Mapping) software tool which is part of

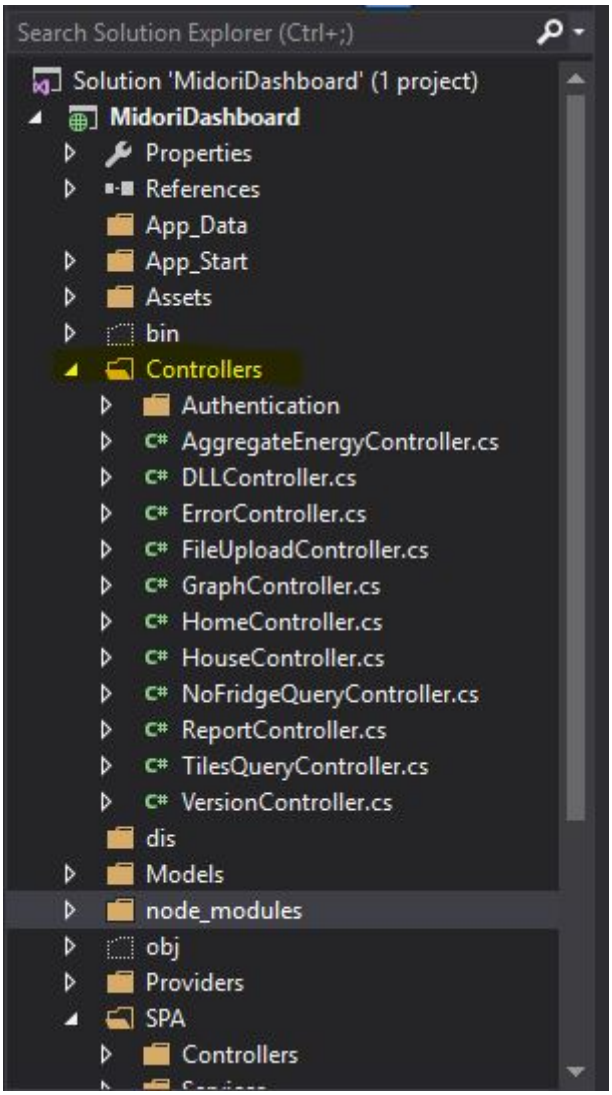

Figure 5.4. Implemented REST APIs

the ADO.NET framework. It facilitates the work of developers by enabling to work with data in the form of domain-specific objects and properties, without having to concern themselves with the underlying database tables and columns where this data is stored. In this way, the developers can work at a higher level of abstraction when they deal with data, and can create and maintain data-oriented applications with less code than in traditional applications.

3. Database Layer
   Database is used as a persistent data store. REST API work as an interface to communicate with database to extract/post data.

Chapter 6

Results and Conclusion

Final chapter represent the results obtained from this thesis, some of the conclusion take from the work done and what some future work might be on the same subject.

6.1 Results

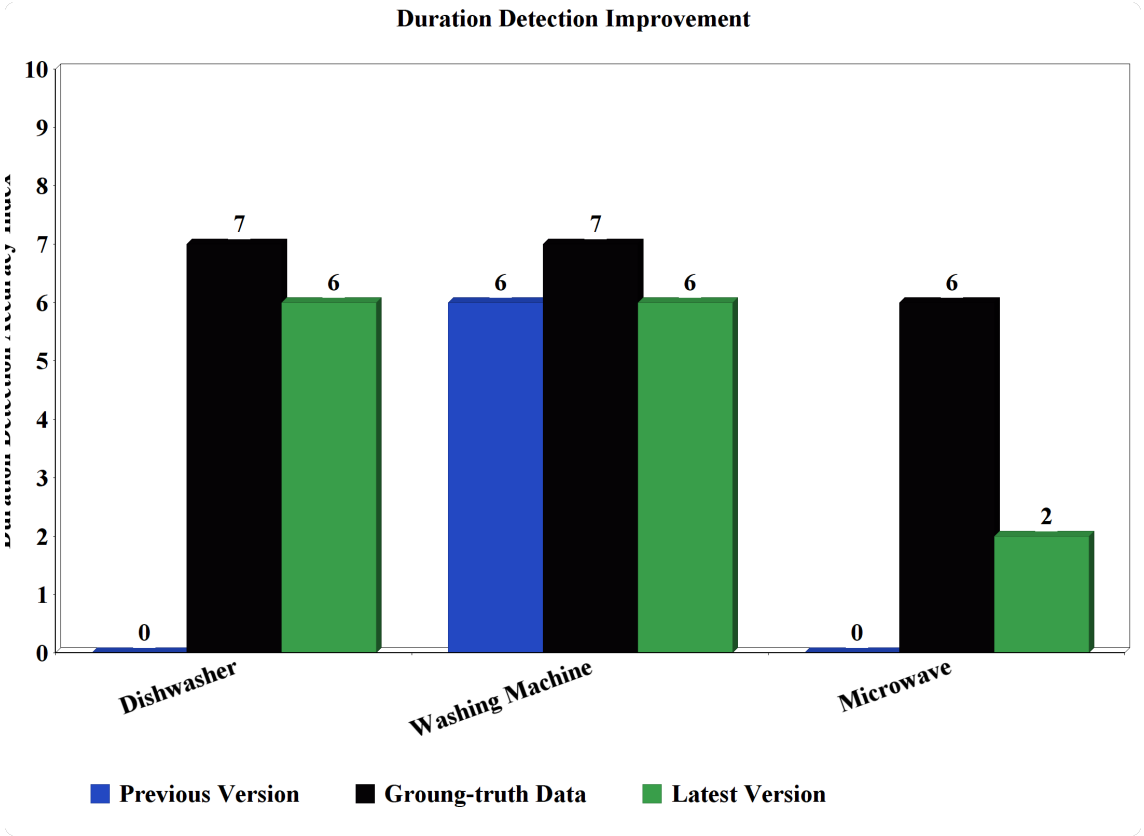

Figure 6.1. Duration Detection Improvement

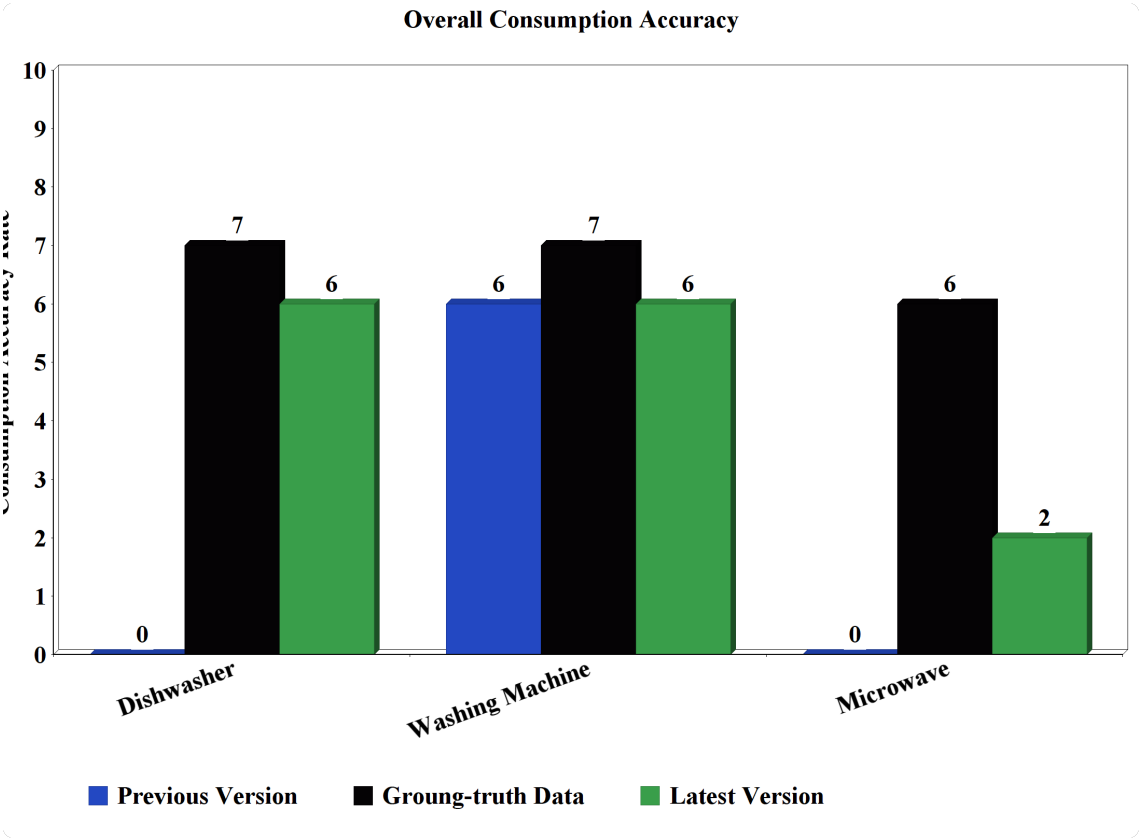

Figure 6.2. Overall Consumption Accuracy

6.2 Conclusion

In order to note the improvement from version to version of a NILM algorithm , it is necessary to collect ground-truth data as an indicator of quality with which each version is confronted. The ground-truth data was collected with help of submetering device such as smart-meters installed in three different devices: Dishwasher, Microwave and Washing Machine. The gata gathered by the submetering process are reported in table 6.1. They were installed in a test house and result were gathered during a week in March 2017 (23 March - 30March). For calculation of the KPIs defined in the thesis the acceptable duration timelength was set to **5 min**, while the power consumption value should not change more than **1%** from the value reported from the ground-truth data.Figure 6.1 is an evidence which demonstrates that the web application is able provide to user information about the improvement gaps which exists in a version of the algorithm. The first bar represent the Duration Detection Accurary Index of the previous version, before implementing the web application. After noting the results to the NILM developers, neccesary improvements were made on the under-developent NILM algorithm. The later version performs better in detecting devices such as Microwave or Dishwasher reported by ground-truth data inside a certain time range. This is illustrated by the third bar in the figure. The Consumption Accuracy Index metric provides results which are shown in figure 6.2. As noted this index provides the same results which means that if detected correctly the disaggregated value is within the desired range. Future work

would consist in collecting a larger amount of ground-truth which has undergone a refining process for providing even better feedback and a greater insight on the under-developing NILM algorithm.

Device	**Date**	**Start Time**	**End Time**	**Duration**	**Consumption**
Washing Machine	2017-03-23	09:35:00	11:52:00	02:17:00	633
Washing Machine	2017-03-23	18:42:00	19:47:00	01:05:00	619
Washing Machine	2017-03-24	17:06:00	19:05:00	01:59:00	350
Washing Machine	2017-03-25	08:40:00	10:26:00	01:46:00	825
Washing Machine	2017-03-29	14:33:00	16:54:00	02:21:00	650
Washing Machine	2017-04-01	10:05:00	11:52:00	01:47:00	506
Washing Machine	2017-04-02	09:33:00	12:37:00	03:04:00	859
Microwave	2017-03-21	19:01:00	19:03:00	00:02:00	13
Microwave	2017-03-21	19:11:00	19:13:00	02:04:00	8
Microwave	2017-03-24	21:49:00	21:51:00	00:02:00	30
Microwave	2017-03-31	18:04:00	18:06:00	00:02:00	34
Microwave	2017-04-01	11:35:00	11:37:00	00:02:00	25
Microwave	2017-04-02	18:16:00	18:18:00	00:02:00	18
Dishwasher	2017-03-21	20:39:00	22:43:00	02:04:00	1432
Dishwasher	2017-03-23	20:07:00	21:54:00	01:47:00	905
Dishwasher	2017-03-24	21:10:00	22:59:00	01:49:00	1441
Dishwasher	2017-03-26	12:31:00	14:16:00	01:45:00	885
Dishwasher	2017-03-27	18:52:00	20:40:00	01:48:00	910
Dishwasher	2017-03-30	19:23:00	21:09:00	01:46:00	926
Dishwasher	2017-04-01	13:00:00	14:47:00	01:47:00	924

Table 6.1. Ground-truth data table

This thesis explored the added value of monitoring tool in the field of non intrusive load monitoring algorithms used for disaggregation of eletrical energy in a house. While examining different ways to provide , a Single Page Application was found to be a suitable way to providethe desired service with respect to more traditional web applications. SPA offer a native-like experience with respect to traditional applications. As browser never reloads the page to navigate, it requires less amount of networking and computing bandwidth as comporared to traditional applications. AngularJS was chosen as the front end framework since it encourages some very good practices as modularization, separation of concerns, dependency injection and BDD with unit testing. These reasons helped in making the SPA easier to develop and deploy for utilization. The proposed solution takes advantage from implementing and agile proces and a light weight software development process by allowing to continously to integrate and deploy SPA. Using a browser and its capability of using Javascript allows that the developed SPA to reside completely on the browser and only uses server for data synchronization.

6.3 Future Work

Althought the application, is developed with the intentions of integrating the best practices of AngularJS, there are certain enhancements that can provide a developer with a better experience. Some areas of improvement are:

1. The visualizations of the graph can be enhanced by providing the users the ability to add tags or make notes which can be useful in the near future. In this way then can get better idea about the reasons why in certain moments, certain decisions were taken and improving them by making wiser decisions about the whole process

2. Adding a real time notification system, when a new DLL version of the NILM algorithm is available

3. Using more *refine* ground-truth data, for a more deep and rigorous analysis of the impact of the tool in the NILM evaluation field

Bibliography

[1] Ahmed Zoha, Alexander Gluhak, Muhammad Ali Imran, and Sutharshan Rajasegarar. Non-intrusive load monitoring approaches for disaggregated energy sensing: A survey. *Sensors*, 12(12):16838–16866, 2012.

[2] Michael Zeifman and Kurt Roth. Nonintrusive appliance load monitoring: Review and outlook. *IEEE transactions on Consumer Electronics*, 57(1), 2011.

[3] Nipun Batra, Amarjeet Singh, and Kamin Whitehouse. If you measure it, can you improve it? exploring the value of energy disaggregation. In *Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments*, pages 191–200. ACM, 2015.

[4] Nipun Batra. Non intrusive load monitoring: Systems, metrics and use cases. In *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems*, pages 501–502. ACM, 2015.

[5] Joost F Wolfswinkel, Elfi Furtmueller, and Celeste PM Wilderom. Using grounded theory as a method for rigorously reviewing literature. *European journal of information systems*, 22(1):45–55, 2013.

[6] Richard Harper. Inside the smart home: Ideas, possibilities and methods. *Inside the smart home*, pages 1–13, 2003.

[7] Anna Fensel, Slobodanka Tomic, Vikash Kumar, Milan Stefanovic, Sergey V Aleshin, and Dmitry O Novikov. Sesame-s: Semantic smart home system for energy efficiency. *Informatik Spektrum*, 36(1):46–57, 2013.

[8] Flexible smart metering architecture for multiple energy vectors. http://flexmeter.polito.it/index.php/project//.

[9] Brendan Cook, Jerrome Gazzano, Zeynep Gunay, Lucas Hiller, Sakshi Mahajan, Aynur Taskan, and Samra Vilogorac. The smart meter and a smarter consumer: quantifying the benefits of smart meter implementation in the united states. *Chemistry Central Journal*, 6(S1):S5, 2012.

[10] George William Hart. Nonintrusive appliance load monitoring. *Proceedings of the IEEE*, 80(12):1870–1891, 1992.

[11] Roberto Bonfigli, Stefano Squartini, Marco Fagiani, and Francesco Piazza. Unsupervised algorithms for non-intrusive load monitoring: An up-to-date overview. In *Environment and Electrical Engineering (EEEIC), 2015 IEEE 15th International Conference on*, pages 1175–1180. IEEE, 2015.

[12] Leslie K Norford and Steven B Leeb. Non-intrusive electrical load monitoring in commercial buildings based on steady-state and transient load-detection algorithms. *Energy and Buildings*, 24(1):51–64, 1996.

[13] Jack Daniel Kelly. Disaggregation of domestic smart meter energy data, 2016.

[14] Lucas Pereira and Nuno J Nunes. Towards systematic performance evaluation of non-intrusive load monitoring algorithms and systems. In *Sustainable Internet and ICT for Sustainability (SustainIT), 2015*, pages 1–3. IEEE, 2015.

[15] Christian Beckel, Wilhelm Kleiminger, Romano Cicchetti, Thorsten Staake, and Silvia Santini. The eco data set and the performance of non-intrusive load monitoring algorithms. In *Proceedings of the 1st ACM Conference on Embedded Systems for Energy-Efficient Buildings*, pages 80–89. ACM, 2014.

[16] Philipp Klein, Jean Merckle, Dirk Benyoucef, and Thomas Bier. Test bench and quality measures for non-intrusive load monitoring algorithms. In *Industrial Electronics Society, IECON 2013-39th Annual Conference of the IEEE*, pages 5006–5011. IEEE, 2013.

[17] Stephen Makonin and Fred Popowich. Nonintrusive load monitoring (nilm) performance evaluation. *Energy Efficiency*, 8(4):809–814, 2015.

[18] Nipun Batra, Jack Kelly, Oliver Parson, Haimonti Dutta, William Knottenbelt, Alex Rogers, Amarjeet Singh, and Mani Srivastava. Nilmtk: an open source toolkit for non-intrusive load monitoring. In *Proceedings of the 5th international conference on Future energy systems*, pages 265–276. ACM, 2014.

[19] Hans Peter Luhn. A business intelligence system. *IBM Journal of Research and Development*, 2(4):314–319, 1958.

[20] Timothy Chee, Lee-Kwun Chan, Min-Hooi Chuah, Chee-Sok Tan, Siew-Fan Wong, and William Yeoh. Business intelligence systems: state-of-the-art review and contemporary applications. In *Symposium on Progress in Information & Communication Technology*, volume 2, pages 16–30, 2009.

[21] Larissa Moss and Steve Hoberman. The importance of data modeling as a foundation for business insight. *Design*, 2004.

[22] Surajit Chaudhuri, Umeshwar Dayal, and Vivek Narasayya. An overview of business intelligence technology. *Communications of the ACM*, 54(8):88–98, 2011.

[23] Antti Lönnqvist and Virpi Pirttimäki. The measurement of business intelligence. *Information systems management*, 23(1):32, 2006.

[24] Mahesh S Raisinghani. *Business Intelligence in the Digital Economy: Opportunities, Limitations and Risks: Opportunities, Limitations and Risks.* Igi Global, 2003.

[25] Derek Smith and Maria Crossland. Realizing the value of business intelligence. *Advances in Information Systems Research, Education and Practice*, pages 163–174, 2008.

[26] Jin Yuping. Research and application of ajax technology in web development. In *Electronics, Computer and Applications, 2014 IEEE Workshop on*, pages 256–260. IEEE, 2014.

[27] Robert K Yin. Case study research: Design and methods fourth edition. *Los Angeles and London: SAGE*, 2009.

[28] Ali Mesbah and Arie Van Deursen. Invariant-based automatic testing of ajax user interfaces. In *Software Engineering, 2009. ICSE 2009. IEEE 31st International Conference on*, pages 210–220. IEEE, 2009.

List of Figures

List of Tables