Cyber-physical systems design methodology for the prediction of symptomatic events in chronic diseases

Kevin Henares, Complutense University of Madrid, Spain
Josué Pagán, Technical University of Madrid, Spain,
José L. Ayala, Complutense University of Madrid, Spain
Marina Zapater, Swiss Federal Institute of Technology Lausanne, Switzerland
José L. Risco-Martín, Complutense University of Madrid, Spain

Abstract

Prediction of symptomatic crises in chronic diseases allows doctor or patient to take important decisions before the symptoms occur, like the intake of drugs to avoid symptoms or the activation of alarms. Objective and accurate prediction is necessary to increase the effectiveness of the health system. Each disease is different in nature, but the prediction always starts with the development of a predictive model that takes discrete data sets as input and provide the trigger alarm, i.e., the prediction of the symptomatic crises inside certain prediction horizons, as an output. Cyber-Physical Systems (CPS) provide many smart features to address these software and physical processes, since these systems are designed with a set of distributed hardware, software and network components embedded in physical systems attached to humans. The implementation of these predictive devices relies on sensing, processing and networking. Recent advances in wireless sensor networks, medical sensors, and cloud computing are making CPS an excellent candidate for these predictive and proactive healthcare applications. Formal Modeling and Simulation (M&S) methods are a promising alternative to deal with design issues of these CPS. M&S provides a cost-effective approach to verify and validate the design and implementation details of complex real-time applications. In this chapter, we show a hardware-in-the-loop model-driven method, based on the Discrete Event Systems Specification (DEVS) to apply CPS system engineering to prevent symptomatic events in chronic diseases. Our approach is based on an incremental substitution of initial DEVS software models with actual hardware devices and software control with the same behavior. Consequently, the complete implementation of the CPS is performed using the same M&S context for the entire design cycle. The methodology is validated against one particular neurological disorder: the migraine. In this case, a CPS is formally designed using the proposed methodology to prevent migraine crises up to 30 minutes ahead of the symptomatic event.

Keywords

Predictive Modeling, Cyber-Physical System Design, Healthcare, Discrete Event Systems

1 Introduction

In this chapter we propose a robust methodology for predictive modeling and optimization applied to complex systems addressing symptomatic crises. The proposed methodology is not constrained by the data availability. This system consists of a framework to generate knowledge from multi-source data. The data can be collected from multiple and heterogeneous sources with questionable reliability. From the knowledge generation, we can predict and actuate a complex system (e.g. neurological diseases) without an analytical description. In the following pages, we describe a real case study: the migraine disease.

1.1 Predictive modeling in mobile Cloud computing and health

Over the last two decades, there has been an explosion of data generation using the Information and Communication Technologies (ICT) which has led to a new industrial era in many fields, such as agriculture, communications or health. This exponential growth of unprecedented knowledge generation requires Big Data analytic solutions.

The Internet of Things (IoT) embraces heterogeneous architectures, methodologies and elements of many different scenarios in vertical way, encompassing always-connected devices that acquire, process and transmit data. When these always connected devices meet Cloud computing, we call it Mobile Cloud Computing (MCC), and when applied to healthcare within the MCC framework, and it is called eHealth. It is also known as mobile health (mHealth) when eHealth uses mobile devices such as smartphones, wearable devices or tablets for the healthcare practice.

In an eHealth application three major elements of an MCC network can be distinguished: (i) a Wireless Body Sensor Network (WBSN), (ii) an intermediate element or gateway (such as a smartphone), and (iii) a big computing facility such as Data Centers. In WBSN, sensors are placed on the body surface and register physiological and environmental human conditions in an unobtrusive way. In the paradigm of mHealth, the patients are named as digital patients. Digital patients do telemonitoring, self tracking and self-diagnosis. The impact of eHealh applications is high, as it is projected to reach 30 billion Wearable Medical Devices by 2020, with a compounded annual growth rate of 42.9% between 2014 and 2019[1].

There are three major challenges in mHealth to make this growth happen smoothly: (i) perform an unobtrusive ambulatory data acquisition, (ii) provide

[1] `https://rockhealth.com/reports/the-future-of-biosensing-wearables/`

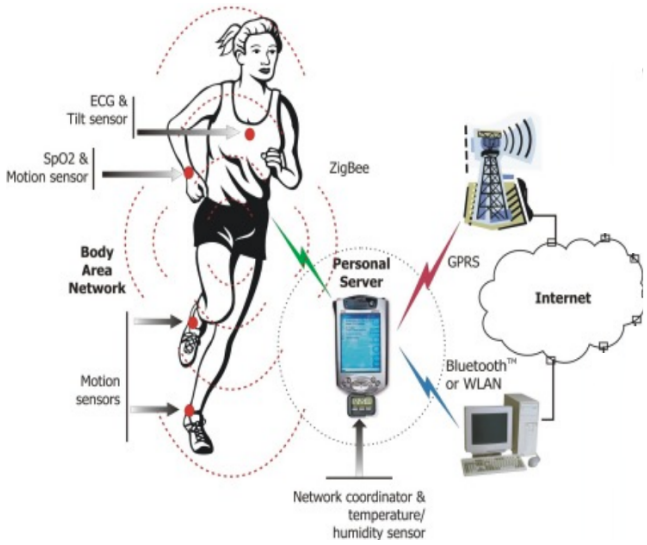


Figure 1: Example of a Mobile Cloud Computing network. Original Image: (*Security Enabled Pervasive Biometric Sensors*, 2018)

an accurate prediction, detection or diagnosis in real time, and (iii) increase the autonomy of monitoring devices.

In this chapter, we present a real case study gathering data from multiple sources and using real ambulatory devices. On the contrary, most of the current approaches in the literature use few variables, or data gathered from a database, or not monitored in an ambulatory fashion. In addition, we propose the use of a subjective and personalized pain scale, instead of traditional pain scales.

With this, we propose a prediction methodology of a complex system without an analytical description, for the migraine disease. In the literature, most prediction and detection problems focus on diseases with well described symptomatic responses, such as diabetes, arrhythmia or epilepsy. We present this predictive modeling methodology using a Model Based System Engineering (MBSE) framework. MBSE allows flexibility, modularity and fast prototyping. MBSE, combined with the use of the DEVS (Discrete Event System Specification) formalism allows us to study the behavior of a monitoring node to make predictions in real time prior to a final industrial implementation.

1.2 Energy efficiency in IoT

The form-factor of the technology of current batteries limits the autonomy of WBSNs (Vallejo, Recas, Del Valle, & Ayala, 2013). Despite the use of low power microcontrollers and more efficient wireless communication interfaces, there is still room for improvement of the energy efficiency in the monitoring devices and other elements of the network. However, the continous availability of the monitoring devices is still a challenge.

The computing capabilities of current smartphones have increased considerably in the recent years. These devices can perform complex computing tasks, and they neeed not be always connected to the backend—to the Data Center—offloading tasks to these facilities. However, this workload balancing has consequences, and this problem must be addressed from a holistic perspective.

Some efforts to address this issue are in practice through the implementation of new technologies such as Fog computing and Edge computing. These are the latest technologies that implement the workload offloading using small computing facilities closer to the data-source than to the Data Center. However, in this book chapter we are going to deal with the problems of a real case of impact of energy utlization in an MCC scenario. The main challenge of energy efficiency in MCC is the optimization of current infrastructures and high scale deployments. There are many approaches handling this problem in the state of the art. Major efforts have focused in (i) the data processing, (ii) the radio interface, and (iii) the workload balancing. Our contribution to these topics and presented in the following sections are:

- Data processing: Consumption aware proactive processing policies in the monitoring nodes, combined with optimized sensing and predictive model generation.
- Radio interface: trade-off between radio and processing power, and prediction accuracy of predictive models running in the monitoring devices.
- Workload balancing: holistic energy optimization in a real scenario with real nodes, solving a real problem and managing workload balancing.

1.3 The migraine disease

The migraine is one of the most disabling neurological diseases. It affects around 15% of the Europe population and 10% of the population worldwide (Stovner & Andree, 2010) leading to high economic costs for private and public health systems. In 2012, each migraine patient led to costs of € 1,222 per year in Europe (according to the study in (Linde et al., 2012)), but more recent studies report the average cost of migraine per patient per year to € 12,970 for patients with chronic migraine (more than 15 days with pain per month) and € 5,041 for patients with episodic migraine, where around 60% correspond to the loss of labor productivity (Research & de Sevilla, 2018).

Migraine is mostly hereditary. It is a social disease that affects more women than men. Currently there is no cure for the migraine and patients take the pills when they feel the pain and it is too late. Migraines not only are composed of pain phases, but are associated with a cascade of neurological processes. Some migraine sufferers experience symptoms that may occur from three days to hours before the pain starts (Giffin et al., 2003). These symptoms are called premonitory symptoms and they are subjective and unspecific: nausea, yawns, tearing, etc. Some patients also suffer from auras. Auras are objective and specific perceptual disturbances such as losing vision that occurs commonly

within 15-30 minutes before the onset of pain. The most efficient way to stop this process and avoid the pain is to take of specific drugs in advance. Therefore, the action mechanism of the medicine is able to block the symptoms before they appear. Even after the episodes, the migraine patients report hangovers, phases composed of the so-called postdromic symptoms. Some of them are fatigue, nausea or dizziness.

Because of the pharmacokinetics of current drugs for treatment of migraine in the acute phase, premonitory symptoms and auras are not helpful to stop the pain sometimes, as it is difficult to estimate the onset of pain. Goadsby *et al.* in (Goadsby et al., 2008) demonstrate that the earlier the intake, the more effective the treatment. In addition, Hu *et al.* in (Hu, Raskin, Cowan, Markson, & Berger, 2002) demonstrate that specific migraine treatments, such as rizatriptan can abort the migraine within 30 minutes. Other specific treatments, such as sumatriptan, reduce this time to 10 minutes before the crisis starts.

It is known that there are changes in hemodynamic variables when a migraine occurs. Hemodynamic variables are regulated by the Autonomous Nervous System like body temperature, electrodermal activity (EDA), heart rate (HR) or oxygen saturation. We have already shown that it exists an evidence that these changes occur before the pain starts (Pagán et al., 2015), going further than the state of the art (Ordás et al., 2013; Porta-Etessam, Cuadrado, Rodríguez-Gómez, Valencia, & García-Ptacek, 2010). Using an ambulatory and unobtrusive WBSN: (i) the condition of the human body before, during and after the pain have been measured, (ii) a new method for pain objectification is proposed, and (iii) a study on the predictability of migraine attacks has been performed.

Since there was no mechanism that allows knowing the onset of pain, a prediction system becomes necessary. Predicting the onset of the attack will allow patients to act in advance in order to avoid the pain or reduce its intensity considerably. A case study analyzed the consequences of applying this predictive mechanisms to 2% of European migraine sufferers, concluding that would incur in savings of more than € 1272 million (taking into account 76% prediction accuracy) (Pagán, Zapater, & Ayala, 2018).

1.4 Modeling and Simulation in the design of CPS

We are now in the era of integrating previous well-studied embedded, real-time, and control systems in a large scale complex cyber-physical systems. One of the major challenges is how to integrate technologies developed by different communities into a coherent, robust, energy-aware CPS. This chapter tackles the design and implementation of current complex CPSs using different model-based methods, tools and methodologies, addressing objectives like performance, smooth integration of cyber and physical subsystems, robustness and energy consumption.

We describe the whole process of conceptualization, design, synthesis and analysis through the use of Model-Based System Engineering (MBSE) principles. This allows us a straightforward shared communication between all the

stakeholders involved in the design of the final product. Furthermore, the use of a M&S standard like Discrete Event Systems (DEVS) formalism (Zeigler, Praehofer, & Kim, 2000) facilitates the composition of models and the interaction with the physical system. Additionally, since DEVS separates the model from the simulator and there exist several simulation engines around the world, there is no need to build simulators, i.e., we must be focused only on the models.

The whole methodology, architecture and some results are detailed in the following sections.

2 General Architecture

In this section, the architecture and conception of different modules that compose the system is presented. First, the architecture is described as a conceptual design, followed by Section 3, describing the actual implementation. Overall, the proposed methodology is aware of on-board energy efficiency and workload-balancing energy efficiency in the whole MCC network.

The proposed methodology is designed and implemented through the named `Critical-Events Robust Prediction System (CERPS)`. CERPS is composed of three subsystems to compute predictions: (i) `Data Acquisition Systems` (`DASs`), (ii) `Robust Prediction Systems` (`RPSs`), and (iii) `Expert Decision System` (`EDS`). These systems perform predictions of critical events based on the processing of heterogeneous data collected from different kind of sources. These predictions are centralized in a single `EDS`, that will operate with the data collected by one or multiple `DASs`. In addition, depending on the nature of these data, an `RPS` can be included between a `DAS` and an `EDS`. Figure 2 shows a high-level distribution of models in the `CERPS` architecture.

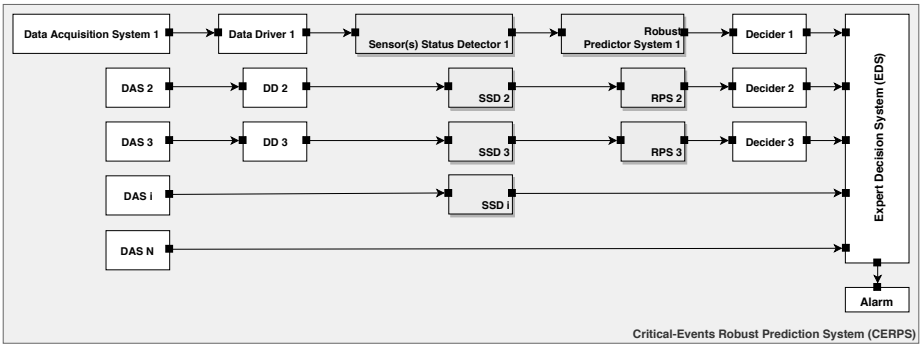


Figure 2: Design of the architecture of the system. `CERPS` is scalable and there might be as many DAS and RPS as needed.

In the following sections, each one of these subsystems is explained in detail.

2.1 Data Acquisition Systems (DAS)

The architecture and implementation of each `DAS` depends on the data source, and they are usually distributed in independent units. It should be noted that we are dealing with heterogeneous data sources, and because of this, some `DASs` are implemented as physical monitoring nodes, and others as remote web services. Moreover, the system can provide information from several sensors or services at the same time, combining its data, if necessary. Depending on the use that will be given to that data, they will be used for prediction purposes by one or multiple `RPSs` (or even none, driving directly the `EDS`).

However, the distributed nature of `DASs` imposes certain limits in the systems design work-flow, due to the need to correlate the data. This problem will be solved later, when necessary, in the `Data Drivers` section ahead.

2.2 Robust Prediction Systems (RPS)

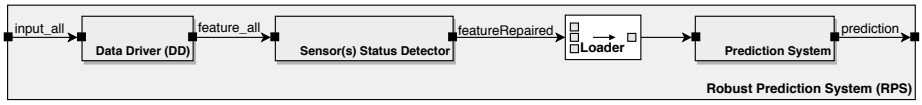


Figure 3: Elements that compose the `Robust Prediction System` architecture. `RPSs` are the most important subsystems, and they can be independent entities by themselves, out of the system.

`RPSs` are the most important component of the architecture and are composed of four subsystems: (i) a `Data Driver` (`DD`), (ii) a `Sensor Status Detector` (`SSD`), (iii) a `Prediction System` (`PS`), and (iv) a `Decider`. Among them, the `Prediction System` (`PS`) represents the core of `CERPS`.

2.2.1 Data Drivers (DD)

`Data Drivers` (`DDs`) pre-process the data obtained by `DASs`. They are composed of different modules, as shown in Figure 4. The `Splitter` generates a pair of *(variable, timestamp)* for each sample gathered from any data source used by the `DASs`. Each pair of values goes through a `Synchronizer` that validates the value of the timestamp and stores the value of the variable in a FIFOs buffer (one per variable). Then, some `Parser/ Feature Generator` module computes and correlates these data and generate features. It is important to notice that the number of generated features does not have to match the number of input variables: several variables can be computed into a single feature and several features can be obtained using a single variable.

These modules will vary depending on the type of the received data (qualitative or quantitative data). Quantitative data usually correspond to the measurements of physical magnitudes varying along time and sampled at a certain rate. Qualitative variables represent mostly non time-dependent information such as events or actions happening occasionally.

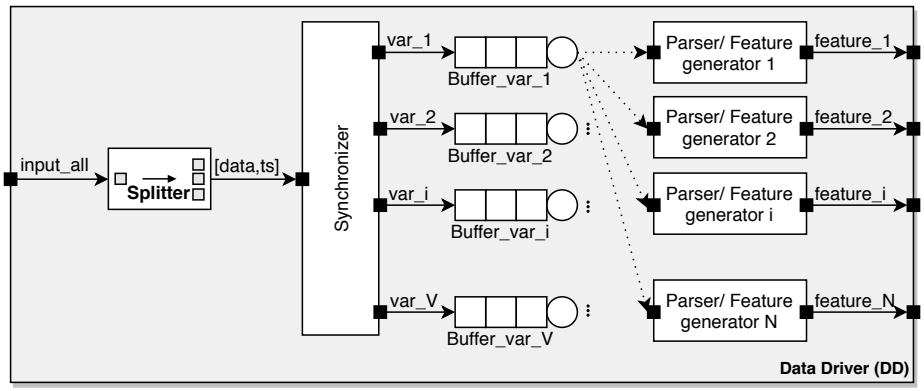


Figure 4: Example of `Data Driver` architecture. It is shown how one measured variable can lead to one or several features to be used in the subsequent modules.

Moreover, depending on the particular use case, `DDs` can be placed before or after `SSDs`. If they are placed before them, the input is a matrix of data and timestamps and the output an array of synchronized features (that may contain errors). Conversely, if they are placed after `SSDs`, the input is an error-free matrix of data and timestamps and the output a reliable array of synchronized features.

2.2.2 Sensor Status Detector (SSD)

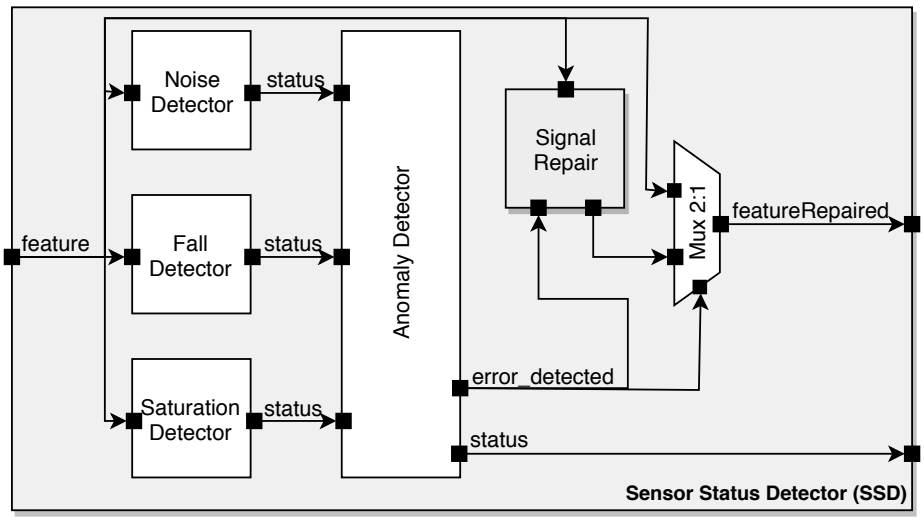


Figure 5: `Sensor Status Detector` architecture. It detects data errors and warns other modules to maintain the accuracy of the prediction.

Measurements in real scenarios are usually unreliable. They are susceptible to data loss and they can be exposed to multiple error sources. SSDs are responsible for providing robustness to the system. They check whether exist errors in the measured signals and generate alarms when something is wrong. Temporally, they repair the signals based on the statistics of buffered past data. Their components are shown in Figure 5.

While no error is detected, the output data correspond to the input signal and no alarms are triggered. Besides, this input signal, coming from the Data Driver or directly from a DAS, is analyzed by three error detectors (saturation, fall and noise detectors). All these components generate independent alarms. Their activation depends on simple thresholds or they can be the result of more complex procedures, like fuzzy logic algorithms. These three alarms are grouped in the Anomaly Detector component, which will raise a definitive alarm that will notify to the following components of the presence of errors in the signal. When this happens, the Signal Repair component is also activated.

Generating predictions when an error is detected would lead to low reliability, erroneous solutions and alarm failures. To avoid this situation, the Signal Repair component generates estimations of the data until the error is solved or enough time has passed for the estimations to stop being reliable. When this component is activated, the multiplexer drives its output to the main output of SSD.

Some solutions to the aforementioned errors would be the replacement of the sensors if they are damaged, or avoiding noisy environments or excessive movements. The operation of the Signal Repair module is based on the use of past samples to generate values that follow the trend that the signal carried before being erroneous. The algorithm applied will depend on the circumstances of the use case. Considering computing capabilities and accuracy of the repairing process, for those scenarios where the signal repairing process runs on devices with constrained computing capabilities, time-series algorithms are suitable to perform the recovery. Conversely, if it runs on a remote server with high processing capabilities a more complex algorithm can be used, such as Gaussian Process Machine Learning (GPML) algorithm. It is important to note that, depending on the algorithm used, it will not only use past samples of the signal managed by the current SSD, but external ones could be used—as an example, time-series models with exogenous inputs use past information from other features, whereas GPML only needs own past data—.

2.2.3 Prediction System (PS)

The Predictor is the component where predictions are computed. It is composed of three different subsystems: (i) the Sensor Dependent Model Selection System (SDMS2), (ii) a set of Predictors, and (iii) the Linear Combiner.

As aforementioned, when some error is detected in the SSDs, the Signal Repair system is activated. Nevertheless, as it is based on previous samples, this procedure is only reliable during certain time. After a while, the past data is not reliable enough to generate predictions and the damaged data being recov-

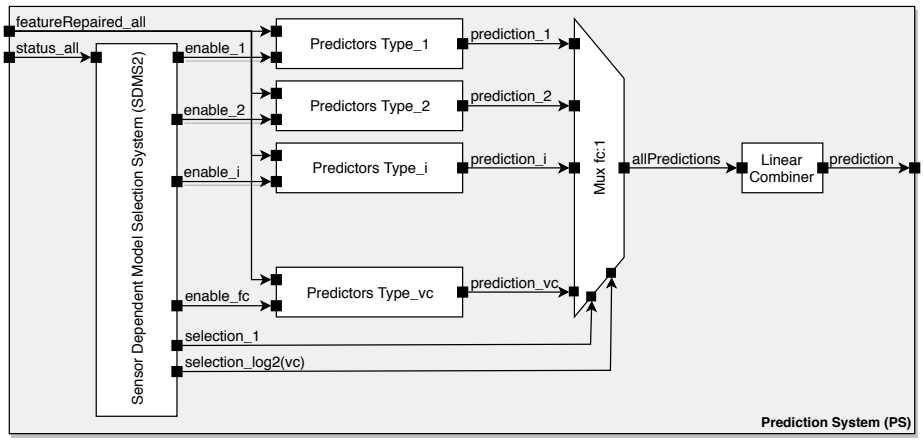


Figure 6: `Predictor` architecture. SDMS2 is the core of this module. It detects the damaged features and selects the suitable Predictor block.

ered must be discarded. In these situations, with a missing input the prediction models do not work. However, the presented methodology takes this into account and it defines different prediction models for each subset of inputs. In this way, when one input data is discarded, other predictive model is activated using only the remaining variables to generate the predictions. Thus, this procedure, as the previous repairing of the signals, increases the overall robustness of the system. With this, the system does not stop but keeps computing predictions maintaining some level of accuracy in the prediction.

Consequently, there will be as many `Predictors` components as the number of supported combinations of active features. The one that is to be used is elected by the `SDMS2`, that activates the corresponding `Predictor` component based on the alarm signals triggered by the `SSDs`. In the worst case, when all V variables (or features) but one are damaged there would be $VC = \sum_{i=1}^{V} \binom{V}{i}$ possible variable combinations to create predictive models. The generation of accurate predictions with one or two features is neither common nor accurate in these cases, and they are not considered as well. Thus, usually a subset $vc \subset VC$ is selected, generating vc `Predictor` modules.

Each `Predictors` component includes one or more predictive models. These predictive models calculate their output h steps ahead (seconds, minutes...), being h the prediction horizon (*i.e.* the time between declaration of a hypothetical event and the event itself). This allows to include models that consider different prediction horizons and combine them. Moreover, several types of predictive algorithms can be included in a `Predictors` component. There are different suitable algorithms that operate over time series data, such as state-space algorithms, time series analysis, artificial neural networks, or grammatical evolution.

In previous studies, it has been demonstrated that a combination of predictions (generated by different predictive models) provide more accurate results

and larger prediction horizons (Pagán et al., 2015). This combination is done in the `Linear Combiner`, that weights the predictions of the active `Predictors` component and generate a single result that will be passed to the `Decider`.

2.2.4 Decider

The `Decider` gets the unified predictions generated by `RPS` and uses them to activate a *local_alarm* signal that will be received by the `Expert Decision System`. Depending on the case, it will use either the last prediction or the last N predictions (accumulated in a buffer). Figure 7 shows the architecture of the `Decider`.

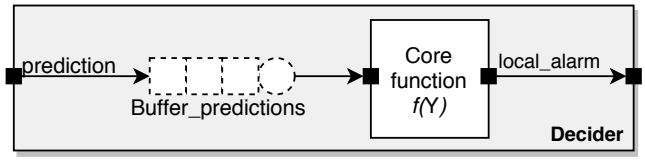
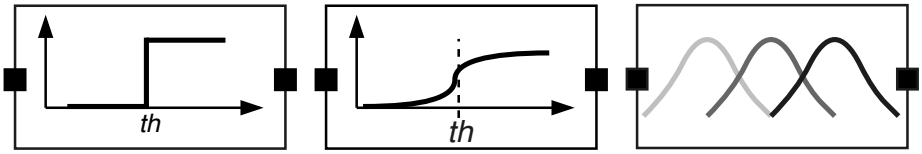


Figure 7: `Decider` architecture. This model generates local alarms based on a core function.

Several mathematical functions can be used to raise the alarm (see Figure 8). Some of them are (i) binary threshold decider, where the local alarm raises when the current prediction or a weighted average value of buffered predictions exceeds a threshold (Figure 8a), (ii) a general case of sigmoid function where the alarm is a softer version of the binary one (Figure 8b), and (iii) a fuzzy logic function that represents the alarm as a result of the fuzzification of the current prediction or individual past predictions (see Figure 8c).



(a) Binary function **(b)** Sigmoid function **(c)** Fuzzy logic

Figure 8: Three examples of core functions for the Decider module.

In systems with only one `DAS` and `RPS`, the output generated by the `Decider` corresponds to the final alarm signal that notifies the critical event. In more complex systems with several `Deciders` their signals go to `EDS`, the last module of the system that generates the final decision.

2.3 Expert Decision System (EDS)

The `EDS` module is the one that triggers the final alarms in `CERPS`. It is a computer-based system and is fed with both the alarm signals of the `Deciders`

and outputs of previous modules (as shown in Figure 2). As these inputs can be affected by problems as data loss or unlabeled data, the automatic generation of decision algorithms is not suitable. Instead, *Active Learning* (AL) algorithms are used. It is a semi-supervised machine learning that interacts with the user (or other information sources) by providing the output when special cases occur (as the arrival of new unlabeled data). It can operate with a greater variety of algorithms, such as Support Vector Machine algorithms, decision-tree based algorithms and Adaptive Neuro-Fuzzy Inference System. In the case of the `CERPS`, this decision model determines the occurrence of new critical events.

3 Software Model and Physical Implementation

As mentioned in previous sections, the `CERPS` architecture is applied to the migraine disease. Specifically, the objective of the presented system is to predict migraine episodes with enough lead time. To do that, migraine patients are monitored in an ambulatory way, measuring four hemodynamic variables: skin temperature (TEMP), electrodermal activity (EDA), heart rate (HR) and oxygen saturation (SpO2). An example of a 6 hours monitoring phase is shown in Figure 9.

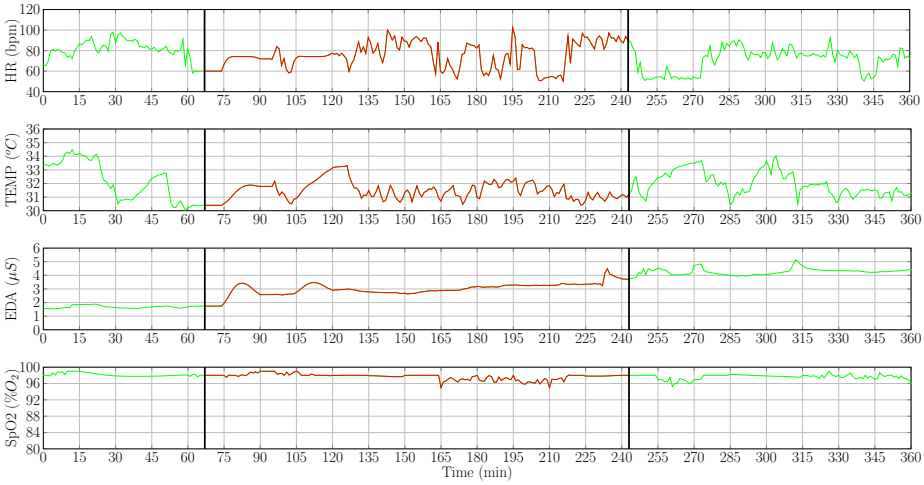


Figure 9: Hemodynamic variables after synchronization and preprocessing during a migraine episode (red curve between vertical bars).

To predict the symptomatic crisis, the first step is to generate a model of the migraine pain. To do this, an adjustment process of the registered subjective pain curve was carried out during the experiments. It is known that the pain rises faster than it recesses, so, the symptomatic curve has been modeled as two semi-Gaussian curves, as they fit the patient's subjective response. In addition to the discrete annotations of pain evolution, patients also indicate two time-points during the migraine attack. The first time-point indicates the onset of the

pain when detected, and the second time-point indicates the end of pain. With all this information, two semi-Gaussian curves can be generated, as shown in Figure 10. $\{(\mu_1, \sigma_1), (\mu_2, \sigma_2)\}$ are the two semi-Gaussian's parameters necessary to define a symptomatic curve. The symptomatic curve includes the pain period, as it reflects some changes in the migraine process. An example of the resulting function is shown in Figure 10, using actual data as well.

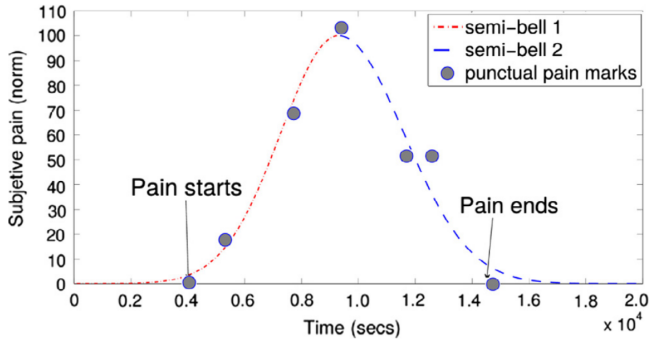


Figure 10: Modeling of subjective pain evolution curve using real data. The pain has been described as two semi-Gaussian curves.

Now, we shall apply CERPS to this real environment. Following the best practices of the MBSE paradigm, the CERPS-Migraine system is simulated through a software model. Next, after validation, it is implemented in a physical device.

As a first step, we introduce a fine-grained migraine predictive model. This relation is expressed in Equation (1).

$$\hat{y}[k + \Delta t] = f\{TEMP[k - p_1], EDA[k - p_2], HR[k - p_3], SpO2[k - p_4]\}, \quad (1)$$

In (1), y is the signal that predicts the migraine pain in the future horizon $k + \Delta t$. This predictive model is a function f of the hemodynamic variables stated above, measured in a past windows defined by a parameter p_i. f is a function that can be implemented in different ways. In our research we have tested several predictive modeling alternatives. State-space models and Grammatical Evolution (GE) showed best results, and in the remaining of the chapter, we will present the implementation using both. State-space models relate changes in hemodynamic variables through an immeasurable state of the pain using matrices (Pagán et al., 2015), and GE predicts directly the pain value by means of mathematical functions applied over the hemodynamic variables (Pagán, Risco-Martín, Moya, & Ayala, 2016).

Figure 11 shows the block diagram of the actual implementation of the migraine prediction system. The following two Sections provide details on how both the CERPS DEVS software model and the corresponding CERPS physical implementation are tackled.

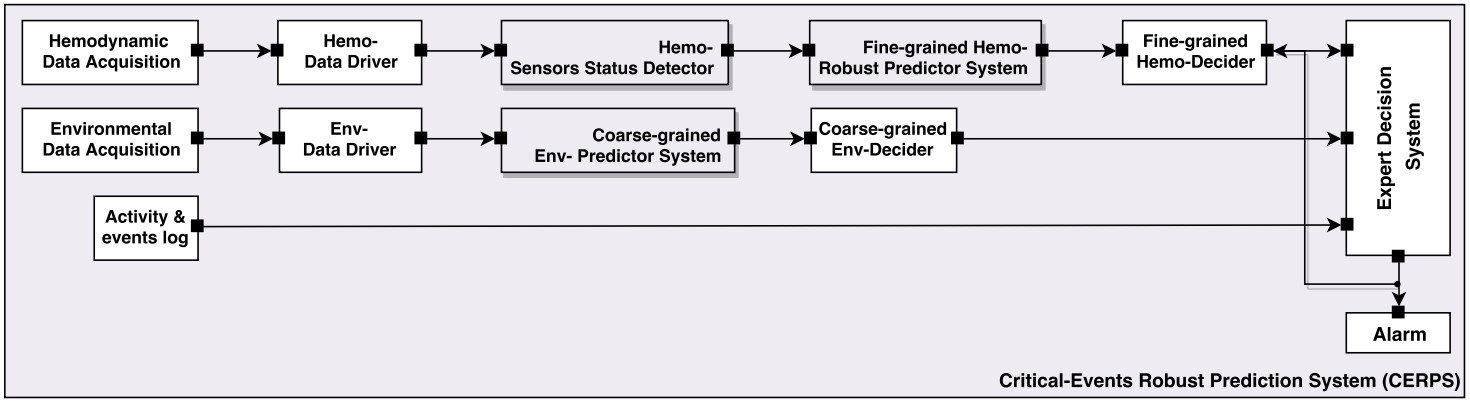


Figure 11: Overview of the implementation of the whole system architecture. There are three types of data sources: two of them perform fine and coarse predictions, while the third one serves as prediction support to the Expert Decision System.

3.1 Software model

Figure 12 shows the details of the application of the CERPS architecture to a DEVS software model of this predictive system (Pagán et al., 2017). Next, after validating the system, it is adapted to a VHDL implementation (Henares et al., 2018). Although certain decisions must be made to adapt it, the process is remarkably simplified due to the modular nature shared by both DEVS and VHDL.

A general view of the DEVS simulation can be seen in Figure 12. This refers only to the coarse-grained migraine prediction using hemodynamic variables. As it can be seen, it is composed of 4 `DASs`, 4 `DDs`, 4 `SSDs`, a synchronizer, a predictor and a decider. To analyze the operation of the system an additional module (`EFgt`) was added to the design. An extra module is used to provide the synchronization of the real pain values of each episode, that will be compared with the generated predictions in the `EFgt` module.

One `DAS` were added by each monitored biometric variable. In the simulation, past data of migraine patients were used to replace the data entry flow that would correspond to the sensors. Also, 4 extra modules were added between the `DASs` and `DDs` to simulate the errors that can appear due to its use (`ErrorInductors`).

The four `DDs` read their inputs and introduce them into the system, associating a timestamp to each sample. The resulting pairs are directed to the `SSD` modules, to repair the possible errors in the signals. Saturation, fall and noise errors are detected and are temporary replaced by an estimation. In this case the repairing of the signal is done using a GPML procedure. The average fitting achieved using GPML ranges from 73.4% for SpO2, to 93.2% for EDA.

In a software simulation, it is suitable to run GPML for signal recovery. However, the implementation in real monitoring nodes is computationally expensive for signal repair in real time. In those cases, autoregressive (ARX) model are used. ARX models are time-series models with low computation requirements. GPML signal recovery needs only its own past data. On the contrary, ARX models use past data from the remaining variables, so if more than one signal is damaged, the system will not be able to recover any of them. Figure 13 shows how the signal recovery works using both alternatives applied over an HR signal with errors.

The outputs of the four `SSD` are provided as input to the synchronizer. The `Sync` module synchronizes and buffers the data for simultaneously supplying the values for the four biometric variables to the coupled model `EFgt`. The pain value is also grouped with the other variables, if available. After this informative module, the synchronized information is processed in the `Predictor` module. In this case, it uses state-space models to generate predictions (solved using N4SID method) and it counts 5 different model sets (Pagán et al., 2015). One of them is intended to operate in the ideal situation, when all the variables are free of errors. The other four are trained to generate prediction using variable sets with a damaged signal (combinations of three operating variables). Hence, reliable predictions cannot be generated with two or fewer biometric variables.

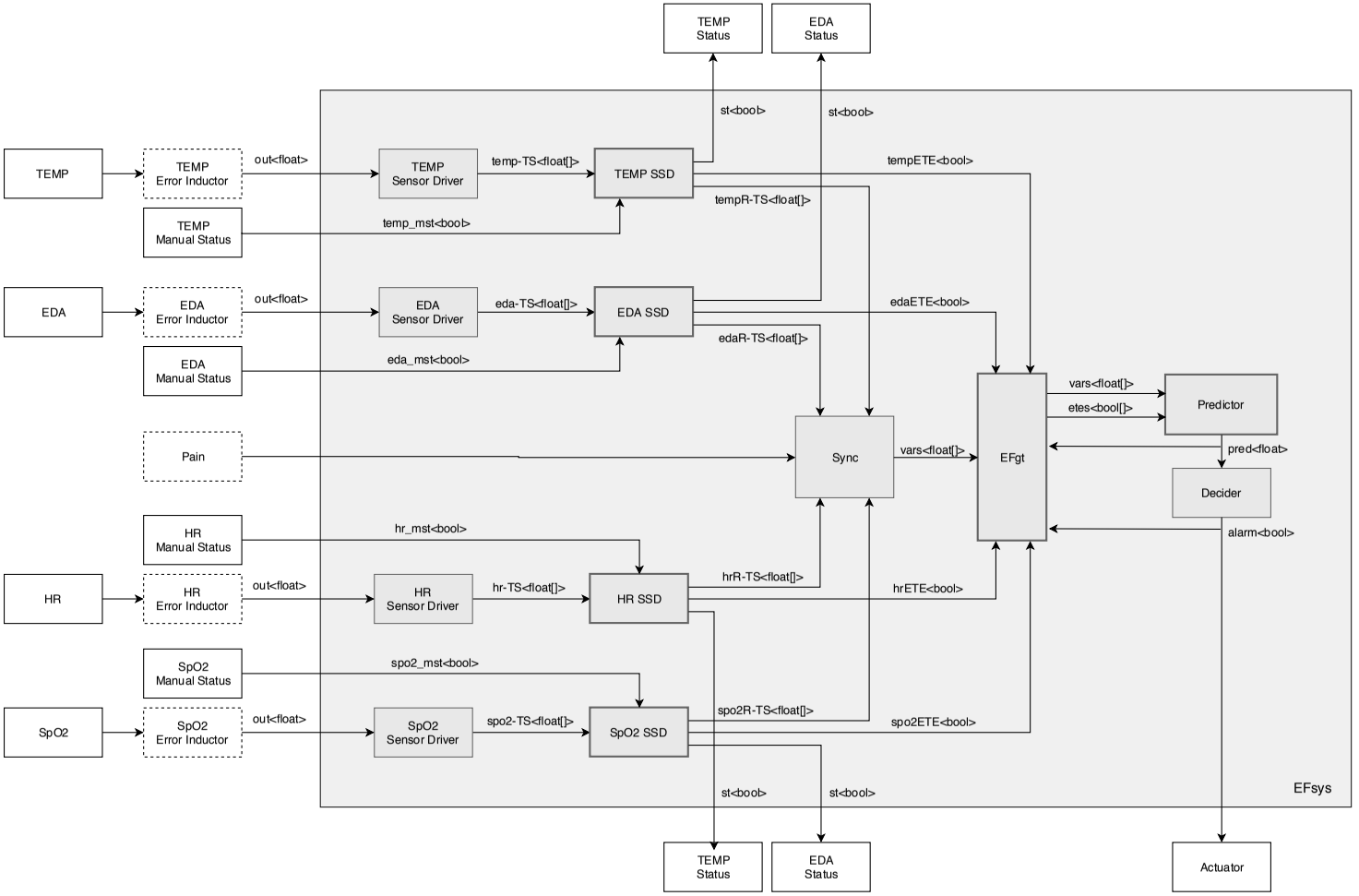


Figure 12: Root component of the migraine pruning system implemented in an FPGA with VHDL.

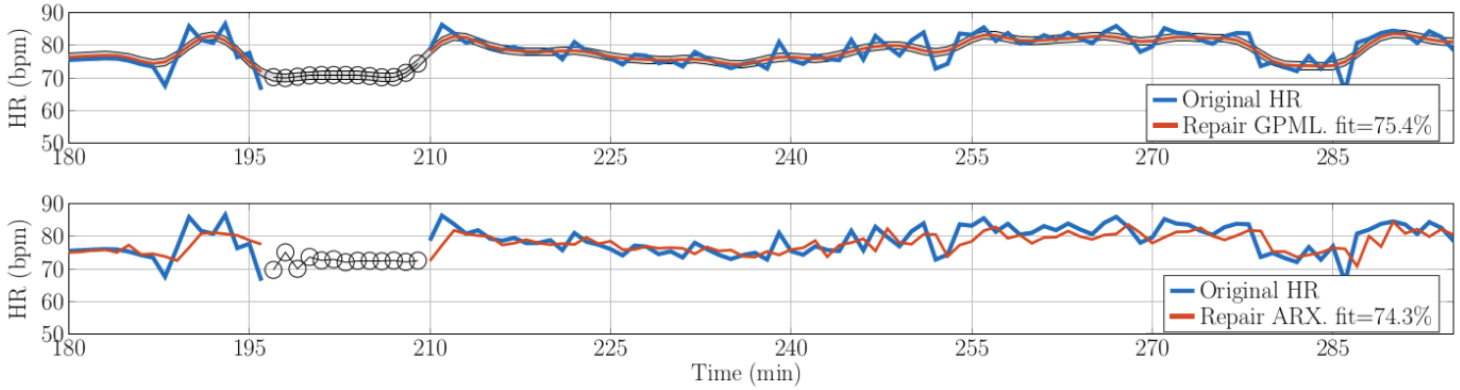


Figure 13: Example of signal repair using Gaussian process machine learning (GPML) and time series algorithms (ARX) for the HR signal

Each model set contains three N4SID models trained to generate predictions with three different time horizons. This was made to increase the effectiveness of the combined prediction, generated by a `Linear combiner` module. In this case, this module simply averages the three predictions (but depending on the case it can be suitable to weight them).

Finally, the output of the `Predictor` is sent to the `Decider`. This module is in charge of activating the alarm signal based on a binary function. The used threshold value is 32. This represents a 50% probability of the maximum pain level (see Figure 10). As it is the only decider of the system, there is no need of an `EDS` and this signal corresponds to the final alarm.

Once the software model is tested and validated (see (Pagán et al., 2017)), it is implemented in the hardware. As we will see, there are some modifications that must be done to the software model, because of some peculiarities of the physical implementation and the target device.

3.2 Physical implementation

As seen before, simulation adds savings in terms of reduced implementation costs, time, or human and material resources. Simulation is the natural step in MBSE design, prior to a physical implementation because it accelerates error debugging phase and it allows the verification of the methodology. Before an actual device is implemented in hardware, a hardware/software (HW/SW) co-simulation that includes hardware-in-the-loop (HIL) is used. This will ensure that the system works in presence of actual hardware sensor failures and physical actuators, and triggers alarms accurately, as predicted by the simulation system. In this section, the changes and the considerations made in the process of translating the DEVS model into a VHDL implementation using an FPGA are discussed. The root component of this implementation is shown in Figure 14.

The first consideration to be made relates to the `DASs`. As this implementation works with real sensors, some interfaces are needed to link them to the real system. Both the TEMP and EDA sensors are analogical and need an analog-to-digital converter. Specifically, a Pmod AD2 has been selected. It counts with 4 channels, 12 bits of precision and communicates through the Inter-Integrated Circuit (I2C) protocol. The remaining biometric variables, HR and SpO2, are read through a OEM III platform[2] (by NONIN ®) using serial communication. It has several operating modes. The selected device streams one integer measure of each one of the variables each second.

As in this case only two `DAS`are there, we need Pmod AD2 converter and NONIN OEM III module for each of them. The system also counts two `DD`. Since in this case real sensor readings must be made, they have the additional task of interacting through suitable communication protocols. The first `DD`, that reads the TEMP and EDA values using the I2C protocol, has to send reading requests three times per second (sample rate of 3Hz). The second `DD`, that reads

[2]OEM-III: `http://www.nonin.com/OEM-III-Module` (accessed December 2018)

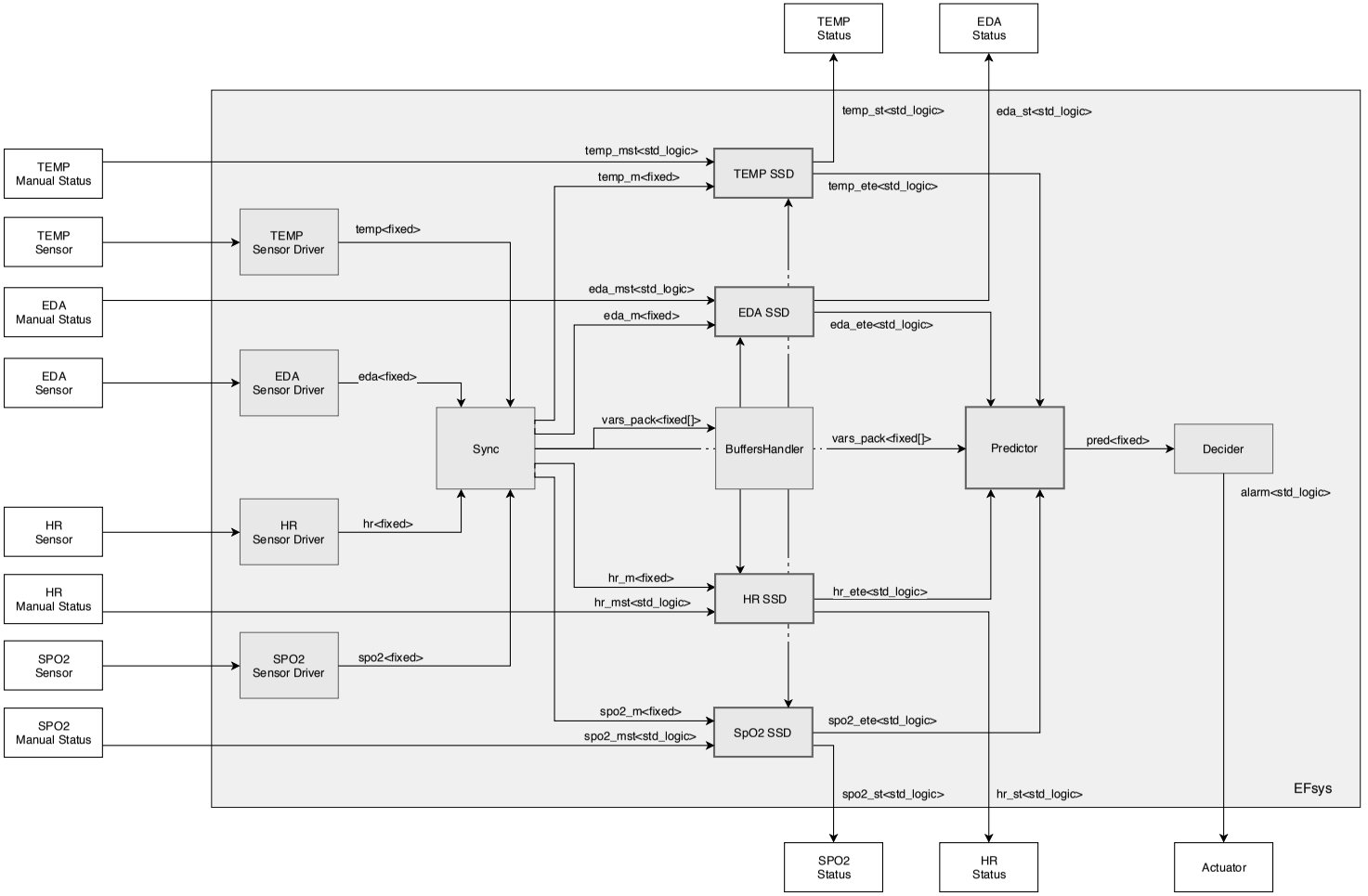


Figure 14: Root component of the migraine pruning system implemented in an FPGA with VHDL.

HR and SpO2 values using serial communication, keeps waiting and read the data sent by the OEM III module (one data pack per second). Moreover, both modules must adapt the data format to match them to the data types used by the system.

As stated above, the `Signal repair` module operates in hardware based on an Auto Regressive model with eXogenous inputs (ARX). The evaluation of these models is more efficient and also offers acceptable results, but has the disadvantage that each module needs previous information of all the other variables of the system. For this reason, an additional module (`BuffersHandler`) was added to group that information and offer them as input to all the `SSD` modules of the system (see Figure 13).

Moreover, since the data obtained from the sensors have different sample rates, `SSDs` are placed after the synchronizer module. This causes the buffer to store the data by the minutes. In this way, the input of the ARX modules is already synchronized and consist of variables of the same data rate.

As there are no large timing requirements in the generation of predictions (since a data set is processed per minute) the `Predictors` modules are joined into a single module. It loads the suitable models from a separately deployed memory module and generates three predictions in a sequential way. This process is controlled by the `SDSM2` module, with requests the predictions one by one, which are accumulated in the `Linear Combiner` as they are generated. This change allows the reuse of FPGA components and results in an optimized use of resources.

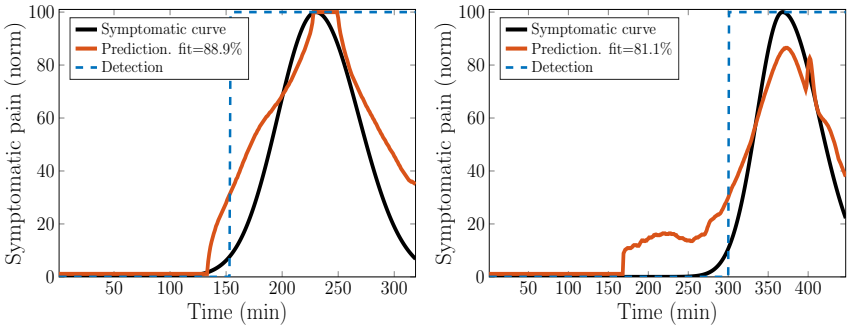
Models are trained to predict migraines 30 minutes ahead. A migraine event is detected when the *fit* metric reaches a 70% of accuracy and the average prediction rate (taking into account the availability of the sensors) is 76%, with a low rate of false positives.

The `Decider` module operates in the same way as in the SW DEVS simulation.

Figure 15 shows two prediction results and the corresponding alarms using N4SID models when all the sensors are available (Figure 15a), and when one sensor fails (Figure 15b) and how `SDMS2` actuates changing the predictive model used.

As floating-point numbers require higher processing in FPGAs (unless specialized modules are available), fixed-point numbers are used in this design. Fixed-point numbers assign a fixed number of bits for both the whole and the decimal part. Consequently, there is a precision loss that must be controlled to avoid introducing significant errors in the obtained results. In this way, two measures are taken: (i) values read from sensors are stored in data types with several decimals, enough to avoid precision loss, and (ii) the intermediate data types to perform the operations needed for signal repairing and prediction generation are analyzed to choose a suitable decimal length. For this, the RMSE is calculated with a range of decimal lengths and it was selected the minimum length with a RMSE lower than 2%. Results can be seen in Figure 16.

Once the feasibility of the procedure and methodology has been demonstrated, the next section shows how to make it energy efficient and how to



(a) Prediction using all sensors.

(b) Prediction using EDA-HR-SpO2 only after the `SDMS2` actuates.

Figure 15: Test results: symptomatic periods for one of the trained patients.

tackle scalability issues.

4 Energy consumption and scalability issues

This section deals with the energy consumption of the proposed system, as well as the scalability issues when the migraine detector is integrated into the health system.

4.1 Energy consumption

The MCC scenario must be economically rewarding. In our implementation, we considered two interfaces where we can actuate: (i) from the WBSN to the gateway or smartphone, and (ii) from the gateway to the high computing facility or Data Center. Energy efficiency in the first interface means battery savings in the monitoring nodes and smartphones; on the other hand, energy efficiency in the second interface means saving money due to the reduction of the electricity bill of the power-hungry big computing facilities.

The use of wearable monitoring devices is becoming increasingly popular. For that reason, more and more research teams are focusing their efforts in reducing their power consumption to enlarge their battery life. Some of the common approaches are based on creating more efficient architectures (Braojos Lopez & Atienza, 2016), develop new on-data processing techniques (Ghasemzadeh, Amini, Saeedi, & Sarrafzadeh, 2015) or apply pre-processing techniques such as compressed sensing (Braojos et al., 2014). In this section, we present a methodology to optimize the energy consumption in the first of the interfaces (from the WBSN to the gateway), focused on the monitoring devices for prediction of migraine attacks in real time. This methodology is focused on the optimization of two main sources of energy consumption: (i) reducing the complexity of the processing in the physical device (our FPGA in this case), that

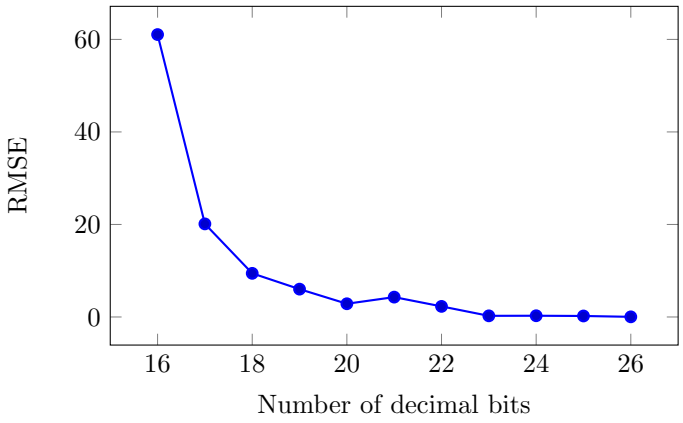


Figure 16: RMSE of the predictions depending on the number of decimal fixed bits used in the variables that hold the intermediate calculations in the `Predictor` module (compared with the results of the simulator, using floating-data types).

leads to reduce the number of clock cycles required to execute the code and, (ii) reducing the consumption of peripherals using the minimum number of sensors. This is a multi-objective optimization problem, for which the multi-objective function can be formulated as follows:

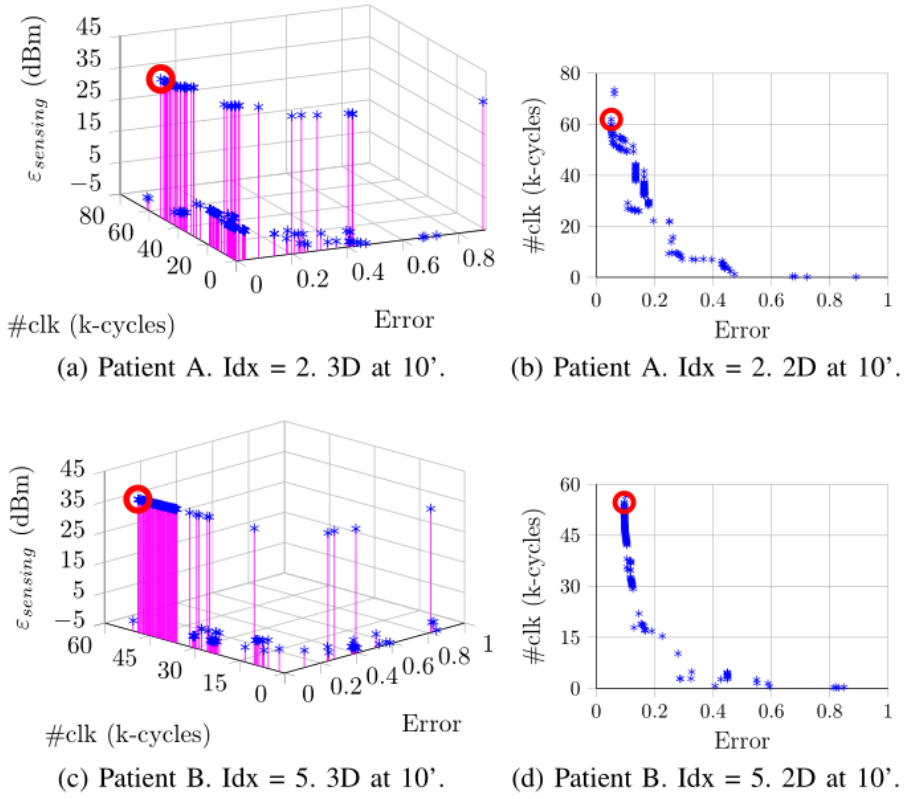
$$\min\left(-fit, \#clk, E_{sensing}\right) \tag{2}$$

To minimize the multi-objective function, we use the developed migraine system with different GE predictive models m_i. Each m_i is defined as a mathematical expression. When the system is working with m_i, we extract three optimization objectives. These objectives, shown in (2), are:

1. the accuracy or *fit* of the predicted values for m_i (the *fit* is related to the normalized RMSE),
2. the number of clock cycles $\#clk$ that the predictive model m_i takes to be computed in the FPGA, and
3. the energy consumption $E_{sensing}$ of the sensors used.

The optimization process is based on the Non-dominated Sorting Genetic Algorithm II (NSGA-II) (Deb et al., 2002), that generates a set of non-dominated solutions (usually called Pareto front). The optimization is performed calculating the number of cycles consumed by the FPGA to compute the model m_i and the energy consumed by the sensors. The first parameter can be easily calculated taking into account the complexity of m_i, the second one is calculated taking into account the energy consumption of the sensors. In our tests, the skin surface temperature (TEMP) uses a thermistor (0.32mJ / .4.9dBm), the electrodermal activity (EDA) uses two differential electrodes (0.32mJ / -4.9dBm),

the electrocardiogram (ECG) to extract the Heart Rate (HR) uses two leads with one reference (396mJ / 26dBm), and the blood oxygen saturation (SpO2) uses an 8000R SpO2 sensor.



(a) Patient A. Idx = 2. 3D at 10'. (b) Patient A. Idx = 2. 2D at 10'.

(c) Patient B. Idx = 5. 3D at 10'. (d) Patient B. Idx = 5. 2D at 10'.

Figure 17: 3D and 2D views of the Pareto Fronts, result of the optimization process.

With these parameters, several models of energy consumption of the MCC environment are obtained. As a result, we obtain two tri-dimensional pareto fronts. The solutions, since they are non-dominated solutions, have the same goodness. The ones with the maximum *fit* are selected. Figure 17 depicts 3D and 2D projections of the results obtained in the pareto fronts for two patients. They are globally convex, tending to minimize the error, the number of clock cycles and the energy consumed by the operation of the sensors. The chosen solutions are painted with a red circle. This methodology reflects savings up to 90% in the execution time. This directly reduces the energy consumption of the system, at the expense of barely degrading the accuracy of the models.

4.2 Scalability issues

Large scale population monitoring systems in MCC scenarios are getting closer to reality. Smart cities combine ICT services and ICT resources to improve the urban environment and improve the citizen's life. In this context, there are examples of research areas that are applying monitoring and model generation to optimize processes and improve techniques. In the sports area, for instance, the monitoring is being used to generate predictions in team sports (Groh et al., 2014). In the medical area, multiple projects also follow this trend. Some applications are: remote diagnosis, disease alarms generation (Alemdar & Ersoy, 2010), prediction of atrial fibrillation (Milosevic et al., 2014), etc. Although these examples focus on specific cases, larger monitoring networks are a reality nowadays. Nevertheless, these networks must be supported by an adequate architecture that supports their progressive extension and improvement.

This section explores the hypothetical deployment of a large-scale monitoring network centered in the migraine prediction, covering the 2% of European migraine sufferers. It is done extrapolating real data and simulation results of previous studies. As a result, it is calculated that the benefits of implant that network would report savings of € 1272 million due to the benefits of the migraine prediction.

The proposed network consists of three main parts: the sensing nodes, the coordinator and the Data Centers. Each one of the three network elements can operate in various modes. Sensing nodes can collect, transmit and process data. The controller can receive, transmit and process data, and perform predictions, if necessary. The Data Center can process data and perform predictions. The energy efficiency policies take these possibilities into account to minimize the power consumption of the whole system. For the sake of clarity, among all combinations of scenarios,the five considered below are more significant from the energy perspective, shown in Table 1.

Table 1: Five scenarios for the workload balancing policies.

	Sensor device	Coordinator	Data Center
SC1	Collect + transmit data	Receive + transmit data	Process data + perform predictions
SC2	Collect + transmit data	Receive + process + transmit data	Perform predictions
SC3	Collect + transmit data	Receive + process data + perform prediction + transmit data	-
SC4	Collect + process + transmit data	Receive + transmit data	Perform predictions
SC5	Collect + process + transmit data	Receive data + perform predictions + transmit data	-

In these scenarios, the sensor nodes can act in two different modes:

- Streaming mode (SC1, SC2, SC3): they collect the raw information from sensors and relay it to the controller. S1 transmit ECG signal immediately, S2 transmit raw data from the OEM-III devices every second.
- Processing mode (SC4, SC5): HR is calculated from the ECG signal and it is transmitted every minute. SpO2 data are extracted from the OEM-III device and transmitted once a minute as well.

The Data Center that has been considered to develop the use case is composed of (i) a High-Performance Computing (HPC) cluster to train and validate

the models, and (ii) a virtual Cloud computing cluster for online prediction.

All the online and offline tasks are characterized in terms of power and performance, taking into account the different network devices where they are executed. Offline tasks are data processing (in GPML), and model training and validation. Online tasks include data processing (in GPML), and online prediction. Model training and validation are CPU and memory intensive, whereas runtime prediction is a light-weight process (that can be performed by controllers).

In runtime processes, several additional policies are used to minimize the Data Center energy consumption. When the coordinators off-load computation from the Data Center, the numbers of VMs are reduced. This policy, when the processing charge decay, is combined with server turn-off policies and cooling optimization. Because of these techniques, the power usage in the Data Center is drastically reduced.

Further details about the Data Center features and the tasks to be performed are available in (Pagán et al., 2018). With these features, HPC and Cloud clusters can handle up to 1,393,649 migraine patients (2% of the migraine sufferers in Europe). Its inclusion in the network is designed to be a gradual process. Four stages are planned, with rates of 50%, 25%, 15% and 10% of the total included patients. Total evaluation period is 10 weeks.

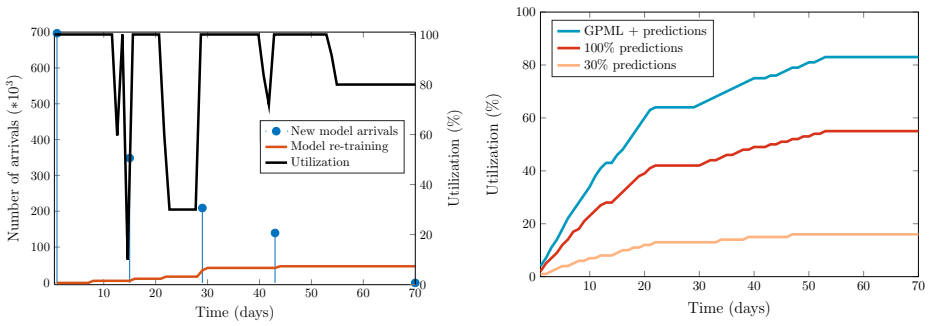
To analyze the differences derived from the use of controllers as intermediate elements, cases S4 and S5 have been studied. Specifically, the following variations are considered:

- SC4 (baseline): the coordinator simply forward computation to the Data Centers (there is no workload off-loading). The off-line phase is performed in the HPC cluster, and the online phase in the virtualized cluster.
- SC4 (optimized): energy minimization techniques are applied in the Data Center, but no workload off-loading policies are applied.
- SC5, 100% prediction: coordinators nodes perform data preprocessing (*i.e.*, GPML), but all the predictions are generated in the virtualized Cloud. The off-line phase is performed in the HPC cluster.
- SC5, 30% prediction: coordinators execute both GPML and 70% of the predictions. The remaining 30% are computed in the virtualized cluster.

Figure 18a shows the utilization ratio of the proposed Data Center as new patients are introduced. Blue dots represent the gradual inclusion of patients (50%, 25%, 15% and 10%). Red line corresponds to models that need to be re-trained.

Figure 18b reflects the utilization of the virtualized clusters in the different scenarios. It can be seen how this charge decreases significantly as we off-load computation to the coordinators. This combined with the turn-off policies at the Data Center, results in a drastic reduction of power consumption.

As can be seen, MCC (and IoT in general) poses important challenges because of the large volumes of data that need to be gathered and analyzed.



(a) HPC Data Center. **(b)** Cloud cluster.

Figure 18: Utilization of the HPC Data Center and Cloud cluster.

Among all the MCC possible applications, population monitoring in eHealth define important constraints and demands energy minimization strategies to develop massive healthcare solutions in massive population scenarios. In this section we have shown a multilevel approach to reduce energy consumption of nodes, ensuring acceptable levels of scalability.

5 Conclusion

The recent advances in wireless sensor networks, medical sensors, and cloud computing are making CPS an excellent candidate to design predictive and proactive healthcare applications. In this chapter, we show a hardware-in-the-loop model-driven method, based on the DEVS formalism to apply system engineering on CPS oriented to prevent symptomatic events in chronic diseases or complex systems without an analytical description.

The architecture of the system has been presented. The derived methodology can be applied to chronic diseases cases where the prediction of symptomatic events is needed. Through its use, the resulting system significantly increases its robustness. On the one hand, temporary errors in the signal are detected and restored. On the other hand, the proposed system allows having several model sets and select the suitable one in real-time. In this way, different prediction algorithms and horizon times can be used, weighting them, if necessary, to improve the prediction quality.

As an example, the proposed architecture was applied to a migraine prediction system. The system is firstly developed in a DEVS M&S context, that allows us to validate the design. Next, it was transformed into a VHDL implementation, taking advantage of the modularity of both the DEVS formalism and the VHDL language.

In addition, several techniques and policies for reducing energy have been presented. First, the impact of the algorithms and sensors used in the sensing nodes have been analyzed, reflecting savings up to 90% of energy consumption.

Also, some techniques related to the management of both the Data Center and the virtualized servers have been discussed. The application of all these points in the developed workflow can result in a significant reduction in terms of energy consumption.

Finally, the deployment of a hypothetical large-scale network extending the migraine prediction case has been discussed. The different network elements and tasks are categorized and used to study the utilization ratio of the infrastructures, comparing approaches with and without intermediate coordinators.

We have shown how MBSE-based techniques and the use of appropriate M&S standard facilitates all the phases of the CPS design and the evaluation of objectives like reliability, performance, robustness, and energy consumption. This covers the full straightforward design method for CPSs, from the first conception, to the hardware design, taking care of the energy consumption and final massive deployment.

References

Alemdar, H., & Ersoy, C. (2010). Wireless sensor networks for healthcare: A survey. *Computer networks*, *54*(15), 2688–2710.

Braojos, R., Mamaghanian, H., Junior, A. D., Ansaloni, G., Atienza, D., Rincón, F. J., & Murali, S. (2014). Ultra-low power design of wearable cardiac monitoring systems. In *Proceedings of the 51st annual design automation conference* (pp. 1–6).

Braojos Lopez, R., & Atienza, D. (2016). An ultra-low power nvm-based multicore architecture for embedded bio signal processing. In *Ict-energy conference 2016.*

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation*, *6*(2), 182–197.

Ghasemzadeh, H., Amini, N., Saeedi, R., & Sarrafzadeh, M. (2015). Power-aware computing in wearable sensor networks: An optimal feature selection. *IEEE Transactions on Mobile Computing*, *14*(4), 800–812.

Giffin, N., Ruggiero, L., Lipton, R., Silberstein, S., Tvedskov, J., Olesen, J., ... Macrae, A. (2003). Premonitory symptoms in migraine an electronic diary study. *Neurology*, *60*(6), 935–940.

Goadsby, P., Zanchin, G., Geraud, G. a., De Klippel, N., Diaz-Insa, S., Gobel, H., ... Fortea, J. (2008). Early vs. non-early intervention in acute migraine—'act when mild (awm)'. a double-blind, placebo-controlled trial of almotriptan. *Cephalalgia*, *28*(4), 383–391.

Groh, B. H., Reinfelder, S. J., Streicher, M. N., Taraben, A., & Eskofier, B. M. (2014). Movement prediction in rowing using a dynamic time warping based stroke detection. In *Intelligent sensors, sensor networks and information processing (issnip), 2014 ieee ninth international conference on* (pp. 1–6).

Henares, K., Pagán, J., Ayala, J. L., & Risco-Martín, J. L. (2018). Advanced migraine prediction hardware system. In *Proceedings of the 50th computer simulation conference* (pp. 7:1–7:12). San Diego, CA, USA: Society for Computer Simulation International. Retrieved from `http://dl.acm.org/citation.cfm?id=3275382.3275389`

Hu, X. H., Raskin, N. H., Cowan, R., Markson, L. E., & Berger, M. L. (2002). Treatment of migraine with rizatriptan: when to take the medication. *Headache: The Journal of Head and Face Pain*, *42*(1), 16–20.

Linde, M., Gustavsson, A., Stovner, L., Steiner, T., Barré, J., Katsarava, Z., ... others (2012). The cost of headache disorders in europe: the eurolight project. *European journal of neurology*, *19*(5), 703–711.

Milosevic, J., Dittrich, A., Ferrante, A., Malek, M., Quiros, C. R., Braojos, R., ... Atienza, D. (2014). Risk assessment of atrial fibrillation: a failure prediction approach. In *Computing in cardiology conference (cinc), 2014* (pp. 801–804).

Ordás, C. M., Cuadrado, M. L., Rodríguez-Cambrón, A. B., Casas-Limón, J., del Prado, N., & Porta-Etessam, J. (2013). Increase in body temperature during migraine attacks. *Pain medicine*, *14*(8), 1260–1264.

Pagán, J., Orbe, D., Irene, M., Gago, A., Sobrado, M., Risco-Martín, J. L., ... Ayala, J. L. (2015). Robust and accurate modeling approaches for migraine per-patient prediction from ambulatory data. *Sensors*, *15*(7), 15419–15442.

Pagán, J., Risco-Martín, J. L., Moya, J. M., & Ayala, J. L. (2016). Grammatical evolutionary techniques for prompt migraine prediction. In *Proceedings of the genetic and evolutionary computation conference 2016* (pp. 973–980).

Pagán, J., Zapater, M., & Ayala, J. L. (2018). Power transmission and workload balancing policies in ehealth mobile cloud computing scenarios. *Future Generation Computer Systems*, *78*, 587–601.

Pagán, J., Moya, J. M., Risco-Martín, J. L., & Ayala, J. L. (2017). Advanced migraine prediction simulation system. In *Proceedings of the 2017 summer simulation multiconference (summersim 2017)*.

Porta-Etessam, J., Cuadrado, M. L., Rodríguez-Gómez, O., Valencia, C., & García-Ptacek, S. (2010). Hypothermia during migraine attacks. *Cephalalgia*, *30*(11), 1406–1407.

Research, H. . T., & de Sevilla, U. (2018). Impacto y situación de la migraña en españa: Atlas 2018. In (pp. 1–183). Editorial Universidad de Sevilla.

Security enabled pervasive biometric sensors. (2018). `http://www.ofsrc.ul.ie/index.php/research/3-security-enabled-pervasive-biometric-sensors`. ([Online; accessed 28-Dec-2018])

Stovner, L. J., & Andree, C. (2010). Prevalence of headache in europe: a review for the eurolight project. *The journal of headache and pain*, *11*(4), 289.

Vallejo, M., Recas, J., Del Valle, P. G., & Ayala, J. L. (2013). Accurate human tissue characterization for energy-efficient wireless on-body communications. *Sensors*, *13*(6), 7546–7569.

Zeigler, B. P., Praehofer, H., & Kim, T. G. (2000). *Theory of Modeling and*

Simulation. Integrating Discrete Event and Continuous Complex Dynamic Systems (2nd ed.). Academic Press.