Journal of Ambient Intelligence and Smart Environments 0 (2017) 1
IOS Press

Semantic Representation and Processing of Hypoglycemic Events Derived from Wearable Sensor Data

Jean-Paul Calbimonte [b,*], Jean-Eudes Ranvier [a], Fabien Dubosson [b], Karl Aberer [a]
[a] *Distributed Information Systems Laboratory, Ecole Polytechnique Fédérale de Lausanne (EPFL), Switzerland*
E-mail: firstname.lastname@epfl.ch
[b] *Institute of Information Systems, Universitiy of Applied Sciences and Arts Western Switzerland (HES-SO Valais-Wallis), Switzerland*
E-mail: firstname.lastname@hevs.ch

Abstract. Diabetes Type 1 is a metabolic disease which results in a lack of insulin production, causing high glucose levels in the blood. It is crucial for diabetic patients to balance this glucose level, and they depend on external substances to do so. In order to keep this level under control, they usually need to resort to invasive glucose control methods, such as taking a sample drop of blood from their finger and have it analyzed. Recently, other directions emerged to offer alternative ways to estimate glucose level, using indirect sensor measurements including ECG monitoring and other physiological parameters. This paper showcases a framework for inferring semantically annotated glycemic events on the patient, which leverages data from mobile wearable sensors deployed on a sport-belt. This work is part of the D1namo project for non-invasive diabetes monitoring, and focuses on the representation and query processing of the data produced by the wearable sensors, using semantic technologies and vocabularies that extend existing Web standards. Furthermore, this work shows how different layers of data, from raw measurements to complex events can be represented and linked in this framework, and experimental evidence is provided of how these layers can be efficiently exploited using an RDF Stream Processing engine.

Keywords: RDF streams, Diabetes, Sensors, Personalized health, Semantic Sensor Networks

1. Introduction

Hypoglycemia characterizes a state of low glucose level in the bloodstream. While for non-diabetic people this state is relatively rare due to adequate regulation of the glucose level, it can lead to life threatening effects for diabetic patients, ranging from headaches to judgment impairment and loss of consciousness. To help diabetic users regulate their glucose level, the standard method consists in collecting a drop of blood from the finger and analyze its glucose level using a glucometer. While this method is reliable as it is performed through a direct measurement, it is not very convenient and requires the user to pinch her finger for each new observation. Furthermore, this method does not allow for a continuous monitoring, but rather a sporadic sampling of the glucose level. Alternatively, continuous glucose monitoring can be achieved using an under-the-skin sensor which relays glucose information to an electronic receiver. This method has a granularity of a sample every few minutes. However, the position of the sensor makes it cumbersome for an extended usage, limiting its applicability. For this reason, alternative non-invasive techniques (i.e. not requiring the user to compromise her physical integrity) have been studied recently [7,31,3].

This paper describes the approach taken by the authors for representation and query processing of hypoglycemic events using semantic technologies and stan-

[*]

1876-1364/17/$35.00 © 2017 – IOS Press and the authors. All rights reserved

dards. This work is inscribed in the context of the D1namo project, a non-invasive approach to detect hypoglycemic events based on the continuous collection of sensed data from an off-the-shelf sensor sport-belt.

Existing approaches for monitoring patients with chronic diseases generally rely on wearable devices that generate raw data, with no or little additional information to make it possible for other applications to understand and interpret this data. The integration of applications and intelligent algorithms in personalized health require standard and semantically-rich data representations, which can be interpreted, consumed and integrated automatically. Furthermore, in scenarios such as the continuous monitoring of ECG signals and patient activity, this information is produced as a continuous stream of data, which needs to be processed online and under certain time constraints. For instance, consider the example of a diabetologist wanting to check activities of the user 1 hours before every hypoglycemic event, or checking ECG waveform when the breathing rate go higher than a certain threshold. Dedicated applications can embed these queries and rules, if they share a common understanding of the data produced by the devices, and of the events derived from them. While there have been previous works focused on event-based processing of rules in diabetes applications [4], there is still a need for well-principled and semantically rich data representation models that can be queried dynamically using declarative and standard languages.

To cope with these challenges, a semantic approach is proposed for representing hypoglycemic events, anomalous features, activities and energy expenditure, so that these annotated data can be ingested by a semantic complex event processor in a monitoring mobile platform. The main contributions of this paper can be summarized as follows:

(i) This paper provides detailed description of the semantics-aware modeling and processing techniques exploited in the D1namo framework. This approach allows to detect glycemic events using a non-invasive wearable device, while integrating semantic technologies to allow advanced stream processing about the user's condition.
(ii) A full description of the D1namo ontology is provided. It extends existing standards for representation of sensor data, provenance information and dataset annotations.
(iii) The authors show how the ontology is used to represent the data acquired by the wearable device, and the derived high-level events produced by the machine learning algorithms of the D1namo platform [26], which combine observations about physiological symptoms and energy expenditure in order to detect glycemic events.
(iv) The semantically enriched data model is used to feed an RDF stream processing engine, which is capable of executing continuous queries over the live data. The authors also provide experimental evidence of the efficiency of the query processing.

The remainder of the paper is structured as follows: In Section 2, the general approach of D1namo is presented, from data acquisition to event generation and prediction. Section 3 describes the semantic model for representing the D1namo sensors, observations, datasets and provenance information. Section 4 focuses on the execution of RDF stream processing queries over the data generated by the sensing devices and the processing modules. A general description of the current experimentation is presented in Section 5. A discussion about the results and existing challenges follows in Section 6 before a description of the related work in Section 7 and conclusion.

2. System overview

The driving idea behind the D1namo project is that hypoglycemic events can be detected by examining indirect symptoms that can be revealed by specific patterns in the ECG and other parameters. Given that the market already offers affordable sensing devices that measure these parameters, the D1namo approach has an enormous potential for self-monitoring and prevention for patients with this chronic condition.

More specifically, the D1namo system taps into the data generated by a Zephyr Bioharness [34] sensor belt worn by the user. The belt generates high frequency readings for accelerometers, breathing sensor and electrocardiogram (ECG). The pipeline followed by D1namo to go from raw sensor readings to the understanding of the user's blood glucose level is described in Figure 1 [26].

First, the data is acquired from the Bioharness sensor belt and preprocessed to reduce noise. It is then used to generate two semantic models for classifying hypoglycemic events, respectively based on physiological symptoms of hypoglycemia and energy expenditure.

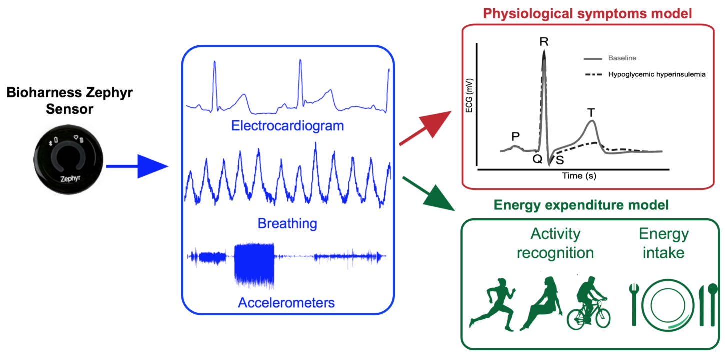

Fig. 1. *Data processing models in D1namo.* The data is acquired from an off-the-shelf sensor belt and preprocessed to reduce noise. It is then used to generate two semantic models for classifying hypoglycemic events, respectively based on physiological symptoms of hypoglycemia and energy expenditure.

Physiological model (ECG)	**Energy expenditure model**
HB fiducial points location	Heart rate
HB fiducial points amplitude	Breathing rate
ST segment shape	Vector Magnitude Unit (VMU)
QTc interval	Energy intake
	Insulin intake

Table 1

List of the features used in the two models defined for classifying hypoglycemic events.

2.1. *Preprocessing.*

The Bioharness captures multiple physiological signals from the patient. However, the different signals are collected with an important amount of noise, which varies from one parameter to the other. While the accelerometer readings are relatively clean, the breathing signal (which measures the extension and compression of the thorax of the patient) is subject to a high frequency white noise, and is also moderately affected by the re-adjustment of the belt by the patient. A low pass filter isolates the oscillations due to the breathing of the user while removing the noise. As for the ECG signal, it presents variable noise: When the user stands still, the noise is very limited and the shape of the ECG is clearly distinguishable. However, under moderate and heavy activity, the ECG becomes very noisy due to artifacts generated by muscle contraction as well as displacement of the belt's electrodes on the skin. Since the noise of the ECG signal is correlated with the movement of the user, a normalized least mean squares filter is applied to the ECG signal. For this, the accelerometer signal is used as interference signal, therefore making use of the correlation between the two signals to mitigate the impact of the noise.

2.2. *Extracting physiological features.*

Once the noise is attenuated, the features that are used by the two models to detect hypoglycemic events (Table 1) are extracted.

The *physiological model* relies essentially on ECG features. This is based on the findings described in [19], where the authors explain how hypoglycemia alters ventricular repolarisation, therefore influencing the ECG of the user. Based on this assertion, fiducial points[1] of the ECG are extracted to be used as features for training the model. In order to extract these points, the approach taken by Yazdani et al. [33] is leveraged. This approach analyses the mathematical morphology of the signals. Top-hat and bottom-hat transforms will accentuate peaks and valley in the ECG signal, and averaging the both allows to isolate the QRS segments efficiently.

The *energy expenditure model* takes as input the heart and breathing rates, the energy intake and the insulin intake. The heart rate of the user is extracted at the same time as the QRS complex. The breathing rate is computed as the local maximums of the breathing signal preprocessed as described in the previous section. The VMU feature is computed as the magnitude of the vector having for coordinates at time t the values of the 3-axis accelerometer of the sensor belt. The energy intake is obtained based on the log of meals and snacks entered by the users. This information is semantically enhanced by querying the Fitbit food database[2]. This database allows for a seman-

[1] Key points that characterize a heart beat: identified by the P, Q, R, S, T labels and can be seen in Figure 1

[2] `http://dev.fitbit.com/docs/food-logging/`

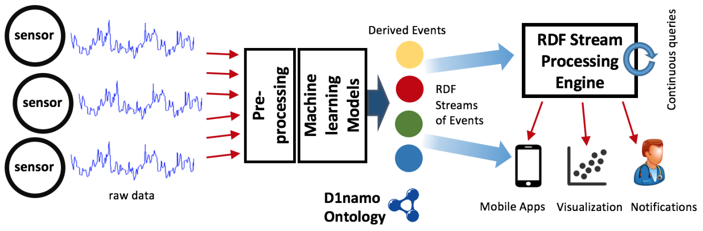

Fig. 2. *Generation of semantic events.* The events are generated as a result of the machine learning models derived from the raw data of the sensors. The D1namo ontology is used to encode these events and produce streams of RDF, which can be directly used by heterogeneous applications, or indirectly through an RSP engine that executes complex continuous queries. In the D1namo use case these queries are used to encode notifications and rules for the medical personnel.

tic annotation of the intake and activity events, which can be represented in terms of an ontology. This includes the event context, meal cat. Two types of insulin are considered, the long acting insulin which is release in the bloodstream over up to 42 hours, and the fast insulin lasting a couple of hours. A model is built to characterize the amount of insulin that is available in the blood stream at a certain time. Finally, the three quantities described above are combine to evaluate the glucose available in the blood stream.

2.3. Model learning.

Once the two sets of features presented in Table 1 are extracted, they can be used to train in parallel two models. The physiological approach is used for a classification model differentiating heart beat in normal glycemic conditions and heart beats under hypoglycemia. On the other hand, the energy expenditure approach yields a regression which can be used to estimate the glucose level of the user at time t, allowing a finer observation of glycemic events.

Once the model is learned, the different detected events can be encoded in a semantically rich representation that can be exploited, processed and integrated with other data sources. One of the main standards for data modeling and interchange with rich semantics support on the Web is RDF [28]. In order to use this standard for generating streams of high-level events, a vocabulary or ontology that describes the domain of interest is required. While the general approach of high-level events is domain independent, for the concrete diabetes domain a dedicated D1namo ontology was designed (see Section 3). Once the stream of RDF events encoded with this ontology are produced, they can be fed to an RDF stream processing (RSP) engine, capable of executing complex event queries. Finally, end-user applications can connect to the continuous results produced by the RSP engine, including mobile personalized health apps, monitoring visualizations, or notifications for physicians and nurses. This ecosystem of raw sensor data to semantically enriched events is depicted in Figure 2.

3. Semantic Representation: the D1namo Ontology

One of the key challenges in the context of D1namo is to coherently integrate different sensor data sources, which can be combined in order to generate higher-level data. This is the case, for instance, for the generation of activity-detection events, which in fact are produced from the combination of different sensor streams: accelerometer, breathing rate, etc. While it would be possible to engineer this integration in an ad-hoc fashion, this solution is costly in the long term, and leads to poor maintainability. Furthermore, it requires system integration developers to have dataset-specific information in order to fuse the data correctly. As an alternative, semantic approach is taken, based on an ontology-based model that uses existing Web standards for knowledge representation. More specifically, the D1namo ontology[3] is propose. Specified in OWL [16], this ontology is in turn based on well known standard vocabularies for representing sensor data and observations. Furthermore, this section describes how the D1namo datasets can be annotated using semantic constructs, enhancing discoverability and self-describability. The provenance annotations are also included, which make it possible to trace how a certain dataset was produced, which methods were used and which original datasets were processed.

[3]D1namo ontology: `https://github.com/jpcik/d1namo/blob/master/d1namo.owl`

3.1. Sensor Description in D1namo

Metadata plays an essential role in order to allow discoverability and self-description of sensor datasets. In the case of D1namo, the sensor metadata can be structured at different levels: one can consider the Bio-Harness device as a sensor that captures multiple properties, or instead, as a sensor system that is composed of different internal sensing devices, each measuring a particular type of signal. Choosing one or the other structures depends on the granularity that is required for the metadata. For example, if individual accuracy and calibration specifications are needed for each device, then a fine grained description would be more appropriate. Furthermore, this metadata includes not only information about the sensor and its characteristics, but also meta-information about the generated datasets and how they can be accessed and exploited. The usage of existing vocabularies is critical, and the adoption of standards is recommended, specially if the data is expected to be used across different applications.

Taking this into consideration, the D1namo ontology extends the widely used Semantic Sensor Network ontology (SSN-O [9]), which is described in the standard OWL language and is extensible for different domains of use. The ontology includes the definitions of sensors that can be physical devices, but also *virtual sensors*. The latter may refer to software entities that produce observations based on other sensor data. In this way,different layers of sensors can be created, starting from those that capture raw data, to higher-level events that aggregate and process observations from lower layers. For instance, one can consider the excerpt of the D1namo ontology in Figure 3, showing some of the sensor classes and subclasses. Under the SSN-O Sensor class one can find both devices (e.g. accelerometers, ECG sensors), virtual sensors (e.g. activity monitors) and human-as-sensors, which refer to observations that patients can perform themselves about their condition. This includes, for example, annotations about their general health, food ingestion, how they slept, etc.

Notice that these are generic sensor classes, and more concrete sensor classes can be defined, e.g. the Zephyr Bioharness device used in the D1namo project. We illustrate this in the following code snippet[4]. The `d1na:ZephyrBioharness` sensor class defines the characteristics of all the particular devices of this kind. For instance, Listing 1 specifies that this class is subclass of a breathing rate and ECG sensor. Then it is possible to specify that a particular device (e.g. identified as `:bioharness_1` is an instance of the defined sensor class, and it is capable of observing breathing rate and ECG wave forms.

```
d1na:ZephyrBioharness a owl:Class;
  rdf:subClassOf d1na:BreathingSensor;
  rdf:subClassOf d1na:ECGSensor.

:bioharness_1 a d1na:ZephyrBioharness;
  ssn:observes d1na:breathing_rate;
  ssn:observes d1na:ecg_waveform.
```

Listing 1: Example of the Bioharness sensor description in RDF using the SSN-O and D1namo ontologies.

It is also possible to further establish restrictions on the sensor characteristics. For instance, the Zephyr Bioharness, being also a breathing sensor, should be restricted in such a way that it observes breathing properties (e.g. breathing rate, peak, etc). This is exemplified in the following ontology excerpt.

```
:BreathingSensor a owl:Class ;
  rdfs:subClassOf ssn:SensingDevice;
  rdfs:subClassOf [ rdf:type owl:Restriction ;
                    owl:onProperty ssn:observes;
                    owl:someValuesFrom d1na:Breathing ] .
```

Listing 2: Defining restrictions for the observed properties of a sensor.

In a slightly more complex scenario, measurement capabilities, such as the sensor frequency, can be represented. The SSN-O ontology provides the `ssn:Frequency` class that can be extended, and used to provide a restriction specified as a measurement capability, as it is shown in the snippet below:

```
:ZephyrBioharness a owl:Class ;
  rdfs:subClassOf [
    rdf:type owl:Restriction ;
    owl:onProperty ssn:hasMeasurementCapability ;
    owl:allValuesFrom :BioharnessBreathingRateCapability ] .

:BioharnessBreathingRateCapability a owl:Class ;
  rdfs:subClassOf ssn:MeasurementCapability;
  rdfs:subClassOf [
    rdf:type owl:Restriction ;
    owl:onProperty ssn:hasMeasurementProperty ;
    owl:someValuesFrom :BioharnessBreathingRateFrequency] ] .

:BioharnessBreathingRateFrequency a owl:Class ;
  rdfs:subClassOf ssn:Frequency .
```

Listing 3: Restricting the frequency specification for the Bioharness sensor using SSN-O.

Finally, once these restrictions are defined, concrete values can be defined, e.g. 25 Hertz frequency for the

[4]For brevity, prefixes are used in all RDF turtle code snippets, which can be expanded by looking at the ontology sources.

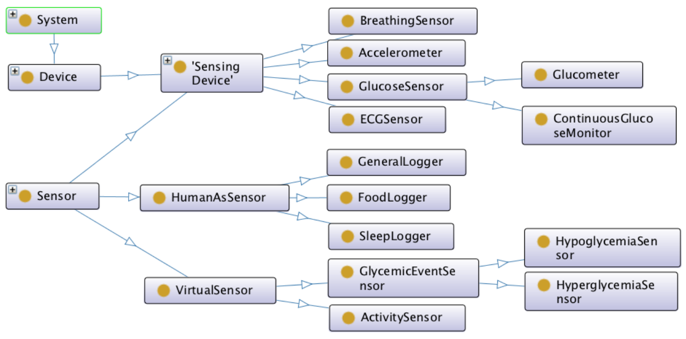

Fig. 3. *D1namo Ontology.* This excerpt shows the different types of sensors and the hierarchy that is built for the D1namo project. Sensors include physical devices, such as an Accelerometer; virtual sensors, such as the machine learning models, and human-as-sensors, i.e. annotations provided by the patient themselves.

Bioharness breathing rate, as in Listing 4. The approach presented here, relies on existing vocabularies for declaring quantities[5] and units of measurement.

```
:bioharness_breathingRateCapability a owl:NamedIndividual ,
  :BioharnessBreathingRateCapability ;
  ssn:hasMeasurementProperty :bioharness_br_frequency .

:bioharness_br_frequency a owl:NamedIndividual ,
  :BioharnessBreathingRateFrequency ;
  qu:unit unit:hertz ;
  qu:numericalValue 25 .
```

Listing 4: Bioharness breathing rate frequency of 25 herz.

A visual example of the definition of the measurement capabilities and properties is provided in Figure 4. This definition focuses on the frequency and response time . Other properties such as accuracy, latency, etc. can be defined similarly, following the recommendations of SSN-O.

3.2. *Sensor Observations*

The D1namo ontology can also be used to annotate the observation generated by the wearable sensors. The framework extends mainly the SSN-O module that deals with observations, and the context in which they are produced. The main extensions are driven by the different observed properties that the sensors capture, i.e. in the case of D1namo ECG, breathing rate, accelerometer, etc. So for each of these properties, a class was created, that represents observations of this kind (e.g. `d1na:ECGObservation` defines

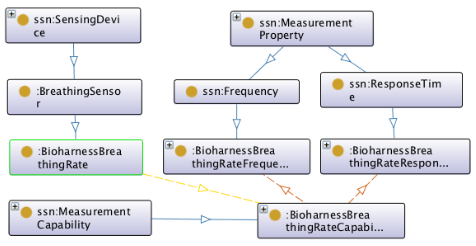
ssn:SensingDevice
ssn:MeasurementProperty
:BreathingSensor
ssn:Frequency
ssn:ResponseTime
:BioharnessBreathingRate
:BioharnessBreathingRateFreque...
:BioharnessBreathingRateRespon...
ssn:MeasurementCapability
:BioharnessBreathingRateCapabi...

Fig. 4. *D1namo Ontology.* This excerpt shows how the different capabilities and measurement specifications are specified for the D1namo sensors, e.g. the frequency and response time of the sensors.

the class of observations about ECG). In the same way, a subclass of `ssn:ObservationValue` is created for each different types of properties. As an example, the following snippet shows an RDF representation of such an observation and its corresponding observation value. It includes important information including the feature of interest (what is observed), the sensor that performs the action, the timestamp, the unit of measurement, etc.

```
:ecg_obs1 a d1na:ECGObservation ;
      ssn:observedProperty d1na:ecg_waveform ;
      ssn:featureOfInterest d1na:skin ;
      ssn:observedBy :bioharness_belt_1 ;
      ssn:observationResult :ecg_obsvalue_1 ;
      time:inXSDDateTime "2016-03-31T23:20:00"^^xsd:datetime .

:ecg_obsvalue_1 a d1na:ECGObservationValue ;
      qu:numericalValue "345.00"^^xsd:float ;
      qu:unit :millivolt .
```

Listing 5: Example of an ECG observation in RDF using the D1namo ontology.

[5]Ontology for Quantity Kinds and Units `http://purl.oclc.org/NET/ssnx/qu/qu`

3.3. Dataset descriptions

The D1namo project produces a data flow in which intermediate datasets can potentially be reused for other purposes, including additional symptomatic analysis, data processing audits, or even for totally different research purposes. In order to do so,the different datasets produced during the dataflow can be systematically annotated, starting from the raw datasets produced by the Bioharness device, the data cleaning process and the complex machine learning models described previously. The DCAT vocabulary[6], which allows defining catalogs and datasets, is used for this purpose. A catalog may include several datasets, e.g. one dataset can represent one or more time series produced by a sensor. A dataset can also group several sensor readings, especially if these are related to the same observed property. These aggregations can be configured as virtual sensors. As an example, consider an ECG waveform dataset (Listing 6) produced by a virtual sensor that refines data from raw ECG signals from the BioHarness sensor. Furthermore the dataset metadata can be combined with the SSN ontology, in order to include specific information about the sensor, such as measurement capabilities, calibration, etc. .

```
:ecg_patient1_2015-06-06 a dcat:Dataset;
      dct:title "ECG waveform data";
      dcat:theme d1na:ecg_waveform;
      dct:issued "2015-16-06"^^xsd:date;
      dct:publisher :LSIR-EPFL;
      ssn:isProducedBy :bioharness_1.
```

Listing 6: Dataset description.

The key feature of the annotated datasets is the possibility to represent provenance information among them. For instance Listing 7 shows how the high level hypoglycemic events predicted by the physiological model is annotated as a derived dataset from the processed sensor data. The PROV-O[7] ontology is used, as recommended by W3C when interchanging and representing provenance information. It is based on four main concepts: *Entities*, which are physical or digital entities, i.e. the datasets in our case. *Activities*, which are actions that happen to an entity, i.e. transformation, processing, generation, modification, etc. *Agents*, which perform or are responsible of the activities upon the entities. The example below represents a hypoglycemia event dataset as an entity, which is generated by an activity, described as a physiological model performed by the D1namo machine learning algorithms.

```
:hypoglycemic_events_patient1_2015-06-06
  a prov:Entity;
  prov:wasGeneratedBy :physiologicalModel_patient1_2015-06-06;
  prov:wasDerivedFrom :ecg_patient1_2015-06-06;
  prov:wasDerivedFrom :breathingRate_patient1_2015-06-06;
  prov:wasAttributedTo :d1namoML.

:d1namoML a prov:Agent;
  prov:actedOnBehalfOf :LSIR-EPFL.

:physiologicalModel_patient1_2015-06-06
  a prov:Activity, d1na:PhysiologicalSymptomsModel;
  prov:used :ecg_patient1_2015-06-06;
  prov:used :breathingRate_patient1_2015-06-06;
  prov:wasAssociatedWith :d1namoML;
  prov:endedAtTime "2015-06-07T02:02:02Z"^^xsd:dateTime.
```

Listing 7: Dataset description.

4. RDF Stream Processing in D1namo

One of the main challenges for managing the dataflow in D1namo is the streaming nature of the sensor data produced by the Bioharness belt. While traditional RDF and Linked Data standards and technologies have shown to be effective for addressing heterogeneity problems in static or semi-static datasets, for highly dynamic streams it is not always appropriate. To fill this gap, RDF Stream Processing (RSP[8]) extensions have been introduced in recent years, including both data models, query languages and processing engines for streaming data [2,20,5].

4.1. CQELS Continuous Evaluation

Of these RSP engines, CQELS [20] provides a simple model based on timestamped triples, on which continuous windowed queries can be evaluated. CQELS is one of the most stable and efficient engines for RDF stream processing, compared to the existing alternatives [21]. CQELS queries are evaluated against live sensor data as soon as it is available, using a language that extends the standard SPARQL [13]. In the context of D1namo, CQELS can be used at different levels in order to get live results and notifications based on the datasets produced by the data pipeline. As it is depicted in Figure 5, raw data produced by the sensors could be queried even before the data cleaning process is effectuated, or after the processed data has

[6]DCAT: `https://www.w3.org/TR/vocab-dcat/`
[7]PROV-O ontology: `https://www.w3.org/TR/prov-o/`
[8]RDF Stream Processing W3C CG `https://www.w3.org/community/rsp/`

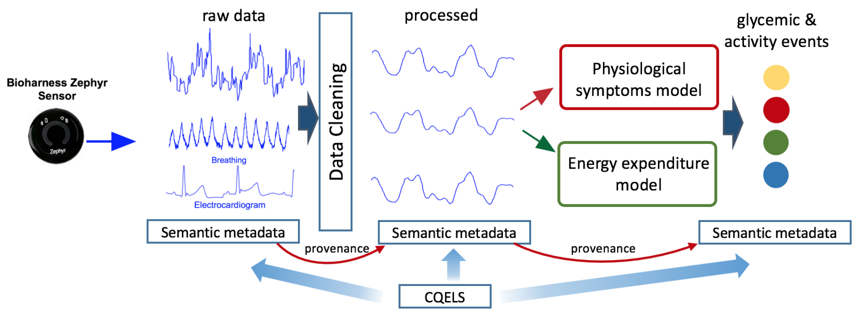

Fig. 5. *D1namo data flow*, starting from the raw data acquisition, pre-processing and model learning, to the generation of actionable events. All datasets produced during these stages are represented using semantic metadata, as described in Section 3, and CQELS can register continuous queries to get results as soon as the data is available.

been transformed into actionable events (e.g. hypoglycemic events). CQELS is capable of consuming multiple streams, and is able to answer to several continuous queries registered simultaneously. Thanks to the data model described in Section 3, CQELS can be used at any of these stages using a unified and coherent semantic model.

For example, Listing 8 represents an excerpt of a stream of breathing rate observations, represented using the D1namo ontology described in Section 3. For brevity the timestamp is represented as a `[t]` annotation.

```
[t] :sensor_p5/obs/breathing_rate_20150501T053300491
  a d1na:BreathingRateObservation
[t] :sensor_p5/obs/breathing_rate_20150501T053300491
  ssn:observedBy :sensor_p5
[t] :sensor_p5/obs/breathing_rate_20150501T053300491
  ssn:observedProperty d1na:breathing_rate
[t] :sensor_p5/obs/breathing_rate_20150501T053300491
  ssn:featureOfInterest d1na:skin
[t] :sensor_p5/obs/breathing_rate_20150501T053300491
  time:inXSDDateTime "2015-05-01T05:33:00.491+02:00"
[t] :sensor_p5/obs/breathing_rate_20150501T053300491
  ssn:observationResult
  :sensor_p5/obs/breathing_rate_20150501T053300491/obsvalue
[t] :sensor_p5/obs/breathing_rate_20150501T053300491/obsvalue
  a d1na:BreathingRateObservationValue
[t] :sensor_p5/obs/breathing_rate_20150501T053300491/obsvalue
  qu:numericalValue 3798.0
```

Listing 8: Example of a stream of observations.

Notice that these streams are not stored by CQELS but they instead flow through the system, and it discards the unnecessary old values when they are no longer needed for query evaluation. A query in CQELS is specified declaratively, extending SPARQL with window operators. An example CQLES query that requests ECG waveform values and observations in the last 100 ms is specified in Listing 9. The main addition in CQELS is the STREAM construct, which references in this case a D1namo stream identified by a URI, and includes a time range window of 100 ms.

```
PREFIX d1na:<http://hevs.ch/aislab/vocab/d1namo#>
PREFIX qu:<http://purl.oclc.org/NET/ssnx/qu#>
PREFIX ssn:<http://purl.oclc.org/NET/ssnx/ssn#>

SELECT ?obs ?val
WHERE {
  STREAM <http://hevs.ch/aislab/d1namo/stream> [RANGE 100ms]
{
    ?obs ssn:observedProperty d1na:breathing_rate .
    ?obs ssn:observationResult ?obsval .
    ?obsval qu:numericalValue ?val .
  }
}
```

Listing 9: CQELS query for the D1namo breathing rate observations.

Although the CQELS engine provides a ready-to-use API, it needed to be further adapted to the rest of the platform, and a simple programming interface developed in Scala was added. This interface is available as open-source code in Github[9]. The Scala code snippet below shows how a CQELS engine is instantiated, starts consuming a stream, registers a query, and finally obtains results continuously. A CQELS engine instance can be created with default parameters as:

```
  val cqels=new CqelsEngine
```

Then, different streams can be launched. In this example, a time series stream is initiated with a constant rate of 10 ms. Any custom stream can be engineered by implementing an RDF stream interface.

```
  val stream1=new StreamTimeSeries(cqels,props,10)
```

Finally, a query can be registered to the system, and a listener will be called each time there are results coming from the engine. For brevity, the full query string is omitted.

[9]Github repo: `https://github.com/jpcik/d1namo`

```
  val q=s"""SELECT ?obs """
  val listener=cqels.selectListener(getResults)
  cqels.registerSelectQuery(q,listener)
```

Notice that the listener receives a method as an input. This method will be invoked every time a result is produced. As an example, the method below simply prints the variable mappings to the standard output.

```
  def getResults(data:Map[String,Any]):Unit={
    println("mappings: "+data.mkString)
  }
```

4.2. RSP Query Evaluation: Discussion

The previous sections described how live sensor data can be processed using CQELS, although these queries have certain limitations in terms of what can be expressed with their respective languages (e.g. limited inference support, sequencing, federation, etc.). Continuous queries in this context can be most useful for providing notification based on the observed properties, not so much on the raw data but more on the complex event that are generated after the machine learning models have been executed, i.e. for processing the glycemic events, or the activity recognition results. As an example, the excerpt below shows two sample Hypoglycemic and Systolic observation events that could be appended to the stream.

```
hypo1 a :HypoEvent;
      :observedAt "2016-03-03T20:30:31";
      :hasValue 45.3.
syst1 a :SystolicObs;
      :observedAt "2016-03-03T20:30:31";
      :hasValue 145.
```

This type of data would require more complex data pattern queries that exploit the temporal dimension of the data. One application of such a query could be to find hypoglycemia events happening before a systolic observation that goes beyond a certain threshold. The current capabilities of RSP engines such as CQELS do not allow for this expressiveness, which requires the so-called Complex Event Processing operators, e.g. sequencing and time pattern matching. Using a CEP-enabled RDF Stream query processor, such as the preliminary work described in CQELS-CEP [10], rules could be evaluated on the incoming events, e.g. sequences over a sliding window, as on the example below encoded as a query:

```
SELECT ?h1,?sys
FROM NAMED WINDOW :win ON ex:eventStream [RANGE  1h]
WHERE {
  WINDOW :win {
    SEQ({?h1 a :HypoEvent},
        {?h2 a :SystolicObs;
             :hasValue ?sys.
         FILTER (?sys>140)} )
  }
}
```

Although this type of CEP queries have not been integrated yet in the system, mainly because of the unavailability of a reliable system that implements them, there is a latent potential for future works on the topic, integrating classical stream processing and CEP techniques.

5. Experimentation

This section describes the different experiments that have been performed in order to validate the use of CQELS for processing RDF streams of the data generated by the Bioharness sensor. It essentially focuses on the output throughput of the system, i.e. the number of results per unit of time. The throughput is a relevant property for evaluation purposes, as it gives a clear idea of the scalability and efficiency of the querying system. The different parameters for the experiments were varied, including different input rates, multiple queries, and simultaneousness. The experiments are driven by the different requirements of the D1namo project. First, events need to be represented using a standard machine-readable data model (i.e. RDF), available as a stream. Then, the system should support multiple streams simultaneously, given that several devices produce different datasets (e.g. ECG, breathing rate, etc) which need to be consumed and combined at the same time. Furthermore, these streams need to run at different speed, ranging from few entries per day (e.g. exercise logs) to thousands per second (e.g. ECG). Finally, it is needed to run several concurrent queries over the RDF streams, and executed continuously, for instance for generating alerts and notification on the live data. The full set of queries is available in the Github repository. All the experiments were performed on an Intel Core i7 3.1 GHz, 16 GB.

The throughput of the system has been the main metric in these experiments, as it provides an idea of the load that the system can sustain. In the first experiment, the throughput of a single query (queries follow a pattern similar to Listing 9) is compared with the ideal throughput that would be produced under perfect conditions. As it can be seen in the results in Figure 6, for lower and middle rates the CQELS evaluation fits

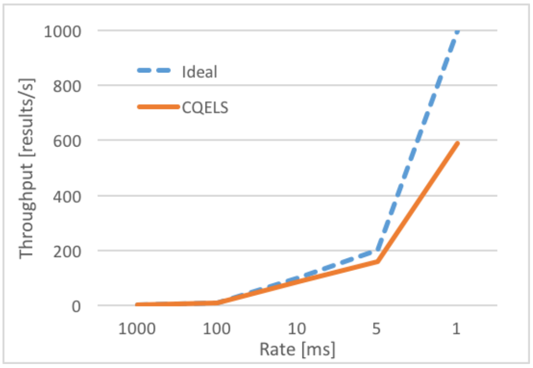

Fig. 6. Execution of a CQELS query. Throughput compared with the ideal behavior (maximum possible throughput). The CQELS execution follow the ideal behavior closely, except for very high streaming rates (1ms).

with the ideal conditions, and only degrades significantly for very high rates, e.g. 1 ms.

However, a query can also consume multiple sensor data sources, each from different observed properties. The next evaluation was to compare the throughput of a stream that includes only one observed property, and another stream that points to several different properties. As it can be seen in Figure 7, there are no major differences except for very high input rates (1 ms). Notice that the figure displayed the results throughput, and in most cases this implies a much larger input dataset.

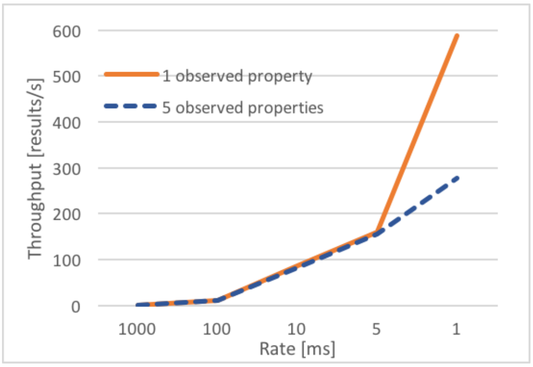

Fig. 7. Adding observed properties to the stream fed to CQELS. Adding the different feature properties used in D1namo, e.g. breathing rate, peak, ECG waveform, accelerometer, etc. The throughput is not affected by adding these properties for high rate up to 5 ms.

Next, the authors analyzed the results after registering multiple queries within CQELS. Depending on the number of continuous queries (all of them of the same complexity), the engine can work differently, until it reaches saturation. As we can see in Figure 8, 1, 5, 20, 50 simultaneous queries were evaluated. Logically, more queries produce more results, increasing the overall throughput. However, it can be seen that the throughput starts deteriorating for high sensor rates, and this applies to most of the query settings. This deterioration is more visible for larger number of simultaneous queries (e.g. 50), although this is not really a problem for D1namo.

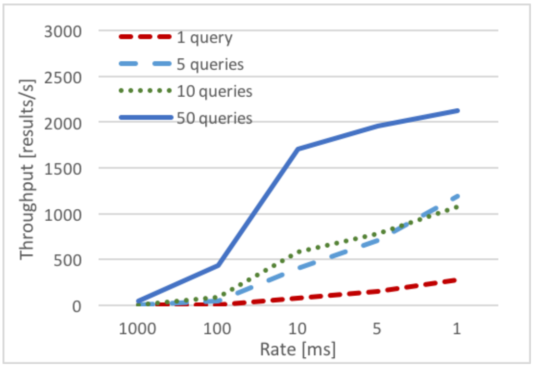

Fig. 8. Adding multiple queries to CQLES. The system is able to respond to simultaneous queries, even if the throughput slightly decreases.

Finally, Figure 9 shows how a series of streams can be fed to the same CQELS engine. Again for low-medium input rates, and high number of streams will negatively impact the result throughput. Different input rates were tried while using 5, 25, 50 ad 100 sensor streams to feed the system. The system suffers a noticeable deterioration of the throughput when the number of streams reaches the first hundred.

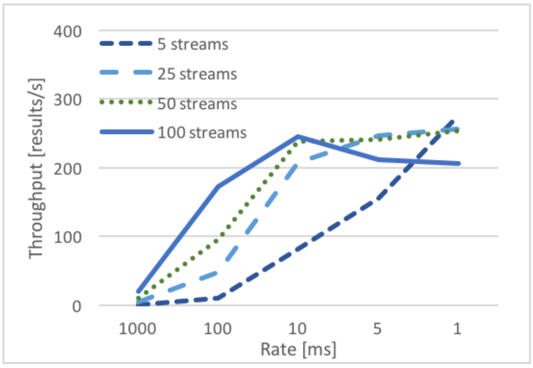

Fig. 9. Adding several streams to the pipeline is sustained by the system. For very high rates of 5ms and 1 ms the system throughput degrades, specially for a large number of concurrent streams.

In summary, the CQELS engine can be successfully used to represent and implement queries over the

D1namo data, reaching close to optimal performance for low and mid rates. Even when the system is under heavy load, it is able to continue under lower capacity, although it will differ more and more from the ideal behavior, as soon as the rates increase.

6. Discussion

As it has been seen, the D1namo ontology can be used not only for the representation of metadata, but also for large data streams of events encoded as RDF triples. The experimental results show that these streams can be consumed and processed under very high load conditions, i.e. up to tens of thousands of data items per second. The requirements of velocity for stream processing of events in the D1namo project range from sporadic acquisition (e.g. food intake) to intensive monitoring (e.g. ECG signals). The ability to answer continuous queries over these streams using extensions of standard languages such as SPARQL, is an objective advantage not only for interoperability, but also for discovery and formulation of notification rules. This may enable in the future the integration of autonomous intelligent applications that feed from the RDF streams of events, thanks to the inherent semantics embedded on the data. Moreover, these semantic representations can be used to feed event processing rules specialized in the health domain, such as in [4].

As a concrete application and validation of the whole approach, a close collaboration has been established with the medical staff at the Hospital of Riviera-Chablais, in order to collect data from diabetes type 1 patients. The study counts with the approval of the local ethical committee board. The acquired data contains the signal acquired by the sensor belt for 4 days, but also glucose readings coming from a continuous glucose monitoring system Ipro 2 (Medtronic) which collects a glucose measurement per 5 minutes. These measurements are used as a ground truth for the training and evaluation of the models. Insulin and food intakes were also collected as manual annotations from the participants. The long process of patient recruitment and gathering of statistically significant volumes of data will provide us with the means to validate the whole D1namo approach in the near future, even though the data processing framework and the semantic representation and query evaluation have already been shown to be effective for the types and amounts of data collected.

7. Related work

Recent years have seen the development of non-invasive methods aiming at replacing the drop of blood to evaluate glucose level. Recently, Google introduced a smart lense [29] in order to sense glucose level from tear fluid. In [7,8,3], the authors leverage signals such as accelerometer data and heart rate in order to refine the insulin dosage provided by pumps and artificial pancreas. For the same purpose, in [32], the author uses energy expenditure and galvanic skin response to estimate the food intake along with glucose measurements. Closer to our concerns, [31] integrates accelerometers, heat-flux sensors, thermistors, ECG electrodes and galvanic skin response sensor to predict glucose level, physical activity and energy expenditure. Other systems that integrate mobile devices and wearables for diabetes patient data management include [12,18,17,22].

Related work also exists on related tasks such as activity detection using wearables [11,1], although these are in most cases not integrated with ECG or other indicators that can lead to detection of glycemic events.

Other approaches have taken totally different directions, aiming at the design of systems that specialize on food intake analysis for risk prevention in diabetes patients [24,30].

While there have also been proposals for ontology-based activity detection [6,27], to the best of our knowledge semantics-based approaches are so far unexplored for combining physiological and energy expenditure data in a glucose estimation system. Other general health monitoring works include [23,15], although they do not consider streams of semantically enriched events, or the use of RDF stream processing engines. More specifically, for diabetes there have also been previous attempts that used ontologies for represent diabetic patient records and observations [25,14], although none of these approaches considers live data processing to detect patterns of interest.

8. Conclusion and Future Work

This paper presents the semantic representation and processing aspects of D1namo, a non-invasive approach to detect glycemic events from mobile sensor data. It describes a physiological and an energy expenditure model to classify these events, and how these events, raw data and provenance information can be modeled using standards-driven ontologies, and queried using extensions for RDF stream processing. To complete this approach, a validation of the inferences described above will be performed with glucose readings from diabetes type 1 patients. This will require the completion of the formulation of the physiological and energy expenditure models, expressed using the semantic model detailed in this paper. The patient studies is expected to bring light into the advantages and potentially the practical issues of this approach. An additional future work will be to include the detection algorithm into a mobile platform that will exploit the semantically enriched data through a complex event processor, providing alerts and recommendations.

References

[1] Fahd Albinali, Stephen Intille, William Haskell, and Mary Rosenberger. Using wearable activity type detection to improve physical activity energy expenditure estimation. In *Proceedings of the 12th ACM international conference on Ubiquitous computing*, pages 311–320. ACM, 2010.

[2] Davide Francesco Barbieri, Daniele Braga, Stefano Ceri, Emanuele Della Valle, and Michael Grossniklaus. C-SPARQL: SPARQL for continuous querying. In *Proc. 18th International Conference on World Wide Web*, pages 1061–1062. 2009.

[3] Marc D Breton et al. Adding heart rate signal to a control-to-range artificial pancreas system improves the protection against hypoglycemia during exercise in type 1 diabetes. *Diabetes technology & therapeutics*, 16(8):506–511, 2014.

[4] Albert Brugués, Stefano Bromuri, Michael Barry, Óscar Jiménez del Toro, Maciej R. Mazurkiewicz, Przemyslaw Kardas, Josep Pegueroles, and Michael Schumacher. Processing diabetes mellitus composite events in magpie. *Journal of Medical Systems*, 40(2):44, 2015.

[5] Jean-Paul Calbimonte, Oscar Corcho, and Alasdair J. G. Gray. Enabling ontology-based access to streaming data sources. In *Proc. 9th International Semantic Web Conference ISWC*, pages 96–111. 2010.

[6] Liming Chen and Chris Nugent. Ontology-based activity recognition in intelligent pervasive environments. *Intl. Journal of Web Information Systems*, 5(4):410–430, 2009.

[7] Simon Cichosz et al. Combining information of autonomic modulation and cgm measurements enables prediction and improves detection of spontaneous hypoglycemic events. *J. Diabetes science and technology*, 2014.

[8] Simon Cichosz et al. A novel algorithm for prediction and detection of hypoglycemia based on continuous glucose monitoring and heart rate variability in patients with type 1 diabetes. *J. diabetes science and technology*, 2014.

[9] Michael Compton, Payam Barnaghi, Luis Bermudez, Raul García-Castro, Oscar Corcho, Simon Cox, John Graybeal, Manfred Hauswirth, Cory Henson, Arthur Herzog, Vincent Huang, Krzysztof Janowicz, W. David Kelsey, Danh Le Phuoc, Laurent Lefort, and et al. The SSN ontology of the W3C semantic sensor network incubator group. *J. Web Semantics*, 17:25–32, 2012.

[10] Minh Dao-Tran et al. Towards enriching cqels with complex event processing and path navigation. In *HiDeSt*, pages 2–14, 2015.

[11] Miikka Ermes, Juha Pärkkä, Jani Mäntyjärvi, and Ilkka Korhonen. Detection of daily activities and sports with wearable sensors in controlled and uncontrolled conditions. *IEEE Transactions on Information Technology in Biomedicine*, 12(1):20–26, 2008.

[12] Eleni I Georga, Vasilios C Protopappas, Christos V Bellos, and Dimitrios I Fotiadis. Wearable systems and mobile applications for diabetes disease management. *Health and Technology*, 4(2):101–112, 2014.

[13] Steve Harris and Andy Seaborne. SPARQL 1.1 Query Language. http://www.w3.org/TR/sparql11-query/, 2013.

[14] Widhy Hayuhardhika, Nugraha Putra, Riyanarto Sarno, Mohamad Sidiq, et al. Weighted ontology and weighted tree similarity algorithm for diagnosing diabetes mellitus. In *Computer, Control, Informatics and Its Applications (IC3INA), 2013 International Conference on*, pages 267–272. IEEE, 2013.

[15] Mark Hennessy, Chris Oentojo, and Steven Ray. A framework and ontology for mobile sensor platforms in home health management. In *Engineering of Mobile-Enabled Systems (MOBS), 2013 1st International Workshop on the*, pages 31–35. IEEE, 2013.

[16] Pascal Hitzler, Markus Krötzsch, Bijan Parsia, Peter F Patel-Schneider, and Sebastian Rudolph. Owl 2 web ontology language primer. *W3C recommendation*, 27(1):123, 2009.

[17] Yu Jin Kim, Sang Youl Rhee, Jong Kyu Byun, So Young Park, Soo Min Hong, Sang Ouk Chin, Suk Chon, Seungjoon Oh, Jeong-taek Woo, Sung Woon Kim, et al. A smartphone application significantly improved diabetes self-care activities with high user satisfaction. *Diabetes & metabolism journal*, 39(3):207–217, 2015.

[18] Morwenna Kirwan, Corneel Vandelanotte, Andrew Fenning, and Mitch J Duncan. Diabetes self-management smartphone application for adults with type 1 diabetes: randomized controlled trial. *Journal of medical Internet research*, 15(11):e235, 2013.

[19] Tomi Laitinen et al. Electrocardiographic alterations during hyperinsulinemic hypoglycemia in healthy subjects. *Annals of Noninvasive Electrocardiology*, 13(2):97–105, 2008.

[20] D. Le-Phuoc, M. Dao-Tran, J. Xavier Parreira, and M. Hauswirth. A native and adaptive approach for unified processing of linked streams and linked data. In *Proc. 10th International Semantic Web Conference ISWC*, pages 370–388. 2011.

[21] Danh Le-Phuoc, Minh Dao-Tran, Minh-Duc Pham, Peter Boncz, Thomas Eiter, and Michael Fink. Linked stream data processing engines: Facts and figures. In *International Semantic Web Conference*, pages 300–312. Springer, 2012.

[22] Lucy Mackillop, Lise Loerup, Katy Bartlett, Andrew Farmer, Oliver J Gibson, Jane E Hirst, Yvonne Kenworthy, Dev A Kevat, Jonathan C Levy, and Lionel Tarassenko. Development of a real-time smartphone solution for the management of women with or at high risk of gestational diabetes. *Journal of diabetes science and technology*, 8(6):1105–1114, 2014.

[23] Federica Paganelli and Dino Giuli. An ontology-based context model for home health monitoring and alerting in chronic patient care networks. In *AINA Workshops (2)*, pages 838–845, 2007.

[24] Sebastian Paßler and Wolf-Joachim Fischer. Food intake activity detection using a wearable microphone system. In *Intelligent Environments (IE), 2011 7th International Conference on*, pages 298–301. IEEE, 2011.

[25] Alireza Rahimi, Siaw-Teng Liaw, Jane Taggart, Pradeep Ray, and Hairong Yu. Validating an ontology-based algorithm to identify patients with type 2 diabetes mellitus in electronic health records. *International journal of medical informatics*, 83(10):768–778, 2014.

[26] Jean-Eudes Ranvier, Fabien Dubosson, Jean-Paul Calbimonte, and Karl Aberer. Detection of hypoglycemic events through wearable sensors. In *Proc. International Workshop on Semantic Web Technologies for Mobile and PErvasive Environments SEMPER*. CEUR-WS, 2016.

[27] Daniele Riboni et al. Is ontology-based activity recognition really effective? In *PERCOM Workshops*, pages 427–431, 2011.

[28] Guus Schreiber and Yves Raimond. Rdf 1.1 primer. *W3C Working Group Note*, 2014.

[29] Melanie Senior. Novartis signs up for google smart lens. *Nature biotechnology*, 32(9):856–856, 2014.

[30] Geeta Shroff, Asim Smailagic, and Daniel P Siewiorek. Wearable context-aware food recognition for calorie monitoring. In *2008 12th IEEE International Symposium on Wearable Computers*, pages 119–120. IEEE, 2008.

[31] Sandra I Sobel et al. Accuracy of a novel noninvasive multisensor technology to estimate glucose in diabetic subjects during dynamic conditions. *J. of diabetes science and technology*, 8(1):54–63, 2014.

[32] Kamuran Turksoy et al. Multivariable adaptive closed-loop control of an artificial pancreas without meal and activity announcement. *Diabetes technology & therapeutics*, 15(5):386–400, 2013.

[33] Sasan Yazdani and Jean-Marc Vesin. Adaptive mathematical morphology for qrs fiducial points detection in the ecg. In *Computing in Cardiology Conference*, pages 725–728, 2014.

[34] Zephyr. Bioharness specification document, http://www.zephyranywhere.com/products/bioharness-3, 2016.