Software Requirements Specification

for

Diabetic Patient Monitoring

Version 2.0 approved

Prepared by:
Christopher Fong
Koenrad Macbride
Yosep Kim
Andrew Padilla
Wun Woo

Medtronic / Alex Zhong

April 14, 2018

Revision History

Name	Date	Reason For Changes	Version
Koenrad MacBride	9/28/17	Document Creation	1.0
Koenrad MacBride	10/12/17	Section 1	1.1
Christopher Fong	10/31/17	Architecture Design	1.2
Yosep Kim, Koenrad MacBride Christopher Fong	12/1/17	Section 2, 3, 4, 5, 6	2.0

1. Introduction

1.1 Purpose

This document will outline in detail the software architecture and design for the Diabetic Patients Monitoring (DPM). This document will provide several views of the system's design in order to facilitate communication and understanding of the system. It intends to capture and convey the significant architectural and design decisions that have been made for the DPM.

1.2 Intended Audience and Reading Suggestions

This document is written on a technical level to address Medtronic's technical department and CSULA Computer Science department.

1.3 Product Scope

This document provides the architecture and design of the GLP. It will show how the design will accomplish the functional and non-functional requirements detailed in the Software Design Document (SDD).

1.4 Definitions, Acronyms, and Abbreviations

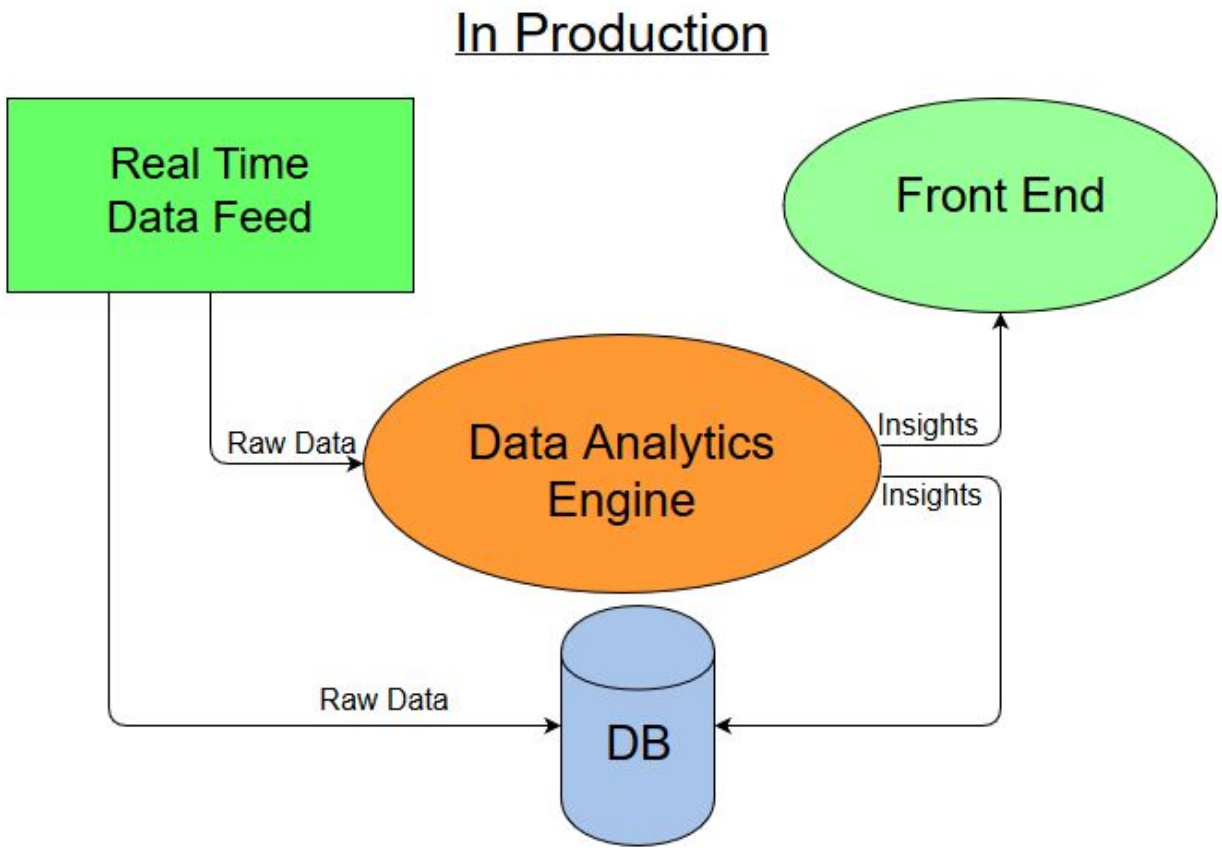
Refer to **Appendix A**.

1.5 References

- DPM Software Design Document.

2. Overall Description

The scope of the project is to make a back end platform for MedTronic to use for predicting the continuous glucose levels of type II diabetes patients. A basic user interface to test all functionality will also be included. Data analytics and data science algorithms will be used to predict the continuous glucose level of patients based on their activity level.



2.1 Product Perspective

The DPM will be used in the future as the back end of a larger system to provide predictions and personalized insights into individual patient's blood glucose levels. Real time data feeds from glucose sensors and wearable fitness trackers will be processed and used by the GLP. These predictions and insights will then be displayed to the patient to help them manage their diabetes.

2.2 Product Functions

The DPM will take a data stream and process the data in real time. Machine learning algorithms will be used to make predictions and insights. These predictions and insights will be displayed on a user interface for the end user.

2.3 User Classes and Characteristics

The users of DPM back end will be the future Medtronic developers assigned to the project. The users of the future front end will be patients and doctors.

2.4 Operating Environment

The real time data stream will be handled by Apache Kafka, which will send the data stream to Apache Storm for data analysis using python data analysis libraries. Storm will send the analysis to a front end for viewing by the end-user. Each module is a micro-service and can be run independently from each other micro-service. These microservices can be easily scaled.

2.5 Design and Implementation Constraints

Design implementation constraints include:

- All use of data must be HIPAA compliant
- Must have continuous network access
- Sufficient Computing Power to carry out tasks in real time

2.6 Apportioning of Requirements

- Implementation of the prediction engine in TensorFlow
- Responsive front-end web interface

3. External Interface Requirements

Again, this section is a high level description for the non-technical people who may be reading this document. Everything in this section is to detail how your software interacts with any external interfaces, whether these are other software or even hardware interfaces.

3.1 User Interfaces

DPM framework does not include a front end, however in future development the users will interact with the outputs generated by DPM.

3.2 Hardware Interfaces

DPM does not require any hardware interfaces.

3.3 Software Interfaces

Name	**Source**
FitBit API	Medtronic
SG Data Stream	Medtronic
Amazon Web Services (AWS)	https://aws.amazon.com/

3.4 Communications Interfaces

As DPM is using the micro-services architecture, each service has its own Restful API to interact with the other micro-services. These API’s will be utilized in over a local neterwork.

4. Requirements Specification

Technologies Used		
Name	**Version**	**Source**
Apache Kafka	1.0.0	http://kafka.apache.org/
Apache Storm	1.1.1	http://storm.apache.org
Apache Cassandra	3.11.1	http://cassandra.apache.org/
Python	2.7.14	http://python.org/
Flask	0.12.2	http://flask.pocoo.org/
Flask Gentelella	1.4	https://github.com/afourmy/flask-gentelella
Charts.js (gentelella)	2.0	https://www.chartjs.org/
Amazon Web Services	----	https://aws.amazon.com/
TensorFlow	r1.4	https://www.tensorflow.org/
JavaScript	1.8.5	https://www.javascript.com/
SciKit-Learn	0.19.1	http://scikit-learn.org/
Pandas	0.210	https://pandas.pydata.org/
HTML5	5.2	https://www.w3.org/TR/html5/

4.1 Functional Requirements

1.2 - Real Time Data Feed

- 1.2.2 - Cassandra Database shall be used to store incoming data
- 1.2.3 - Apache Kafka shall receive data stream from fitbit
- 1.2.4 - Apache Kafka shall receive data stream from SG
- 1.2.5 - Apache Kafka shall create a producer for streaming to Apache Storm
- 1.2.6 - Apache Storm shall map data in parallel to the stream
- 1.2.7 - Apache Storm shall send processed data to Feature Extraction Engine (1.3)

1.3 - Feature Extraction Engine
    1.3.1 - Data shall be received from Apache Storm.
    1.3.1 - Python script shall analyze and sort data
    1.3.2 - Pandas shall handle features of real time data
1.4 - Insight Engine
    1.4.1 - Python script shall find features to develop insights from the real time data
    1.4.2 - Pandas shall handle good values for correlations of different features.
    1.4.3 - Pandas shall handle data in real time to show any reduction in glucose values.
1.5 - Prediction Engine
    1.5.1 - Data shall be received from Apache Storm.
    1.5.2 - RNN shall be implemented using TensorFlow to create predictions
1.6 - Front End
    1.6.1 - Html shall show webpage
    1.6.2 - Javascript shall show charts
    1.6.3 - Data is coming from the Prediction Engine

4.2 External Interface Requirements

Name	**Purpose**	**Source**	**Destination**
FitBit Data	Activity data for predictions	FitBit API	Apache Kafka (1.2)
SG Data	Sensor Glucose data for predictions	Medtronic	Apache Kafka (1.2)

1.1 - Static Fitbit Data
    1.1.2 - non-duplicated data stream and in real time
1.2 - Real time data stream of Sensor Glucose

4.3 Logical Database Requirements

CGM Data

- SG: Sensor glucose values which are readings collected from the sensor every five minutes.
- SGdt: Time stamps for sensor glucose in MATLAB datetime*
- Mbg: Blood glucose readings taken from a finger stick (generally readings occur when patient calibrates sensor or takes an insulin shot called bolus)
- mbgdt: Timestamps for blood glucose in MATLAB datetime
- bolus_delivered_U: Units of insulin delivered

- bolus_dt: Time of bolus delivery in MATLAB datetime

FITBIT Data

- CaloriesOut: Calories expended at minute resolution
- CaloriesOutDt: Timestamps for calories expended in MATLAB datetime
- Steps: Steps taken by user at minute resolution
- Stepsdt: Timestamps for calories expended in MATLAB datetime
- Level: degree of activity (0 - sedentary; 1 - lightly active; 2 - fairly active; 3 - very active.)
- Mets: Metabolic equivalents which is a measure of energy expenditure
- Dt: Timestamps corresponding to mets in MATLAB datetime
- Heartrate: Heart rate at minute/seconds resolution
- Dt: Timestamp for heartrate in MATLAB datetime
- Duration: Duration of sleep in minutes
- Efficiency: (time asleep) / (Total time in Bed – Minutes to fall asleep) * 100
- IsMainSleep: Boolean variable to denote whether it was a main sleep event or a nap
- MinutesAfterWakeUp: Minutes after wake up spent in bed
- MinutesAsleep: Total minutes of sleep in the sleep event
- MinutesAwake: Minutes awake during the sleep event
- MinutesToFallAsleep: Time take in minutes to fall asleep
- RestlessCount: Count of times when user moves during the sleep
- RestlessDuration: Total number of minutes the user moves during the sleep
- StartTime: Start of sleep event in MATLAB datetime
- TimeInBed: Total time spent in bed during the sleep event

Demographic

- AgeDemographic
- YearsOnInsulin
- AgeOnset: Age during onset of diabetes
- Sex
- Diagnosis: ICD-9 code. Generally diabetes related codes start with 250

4.4 Design Constraints

Specify design constraints that can be imposed by other standards, hardware limitations, etc. This should be a more technical description of the overview given in section 2.5.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

DPM requires substantial CPU performance for real time machine learning.

5.2 Safety Requirements

DPM must be in compliance with HIPAA standards for patient identity confidentiality. DPM can must not prescribe or suggest any kind of medical action to the end-user. All predictions and insights are purely informational, and shall be stated as such in the web interface.

5.3 Security Requirements

Security best practices shall be used in all micro-service design.

5.4 Software Quality Attributes

Using the micro-services architecture, each

5.5 Business Rules

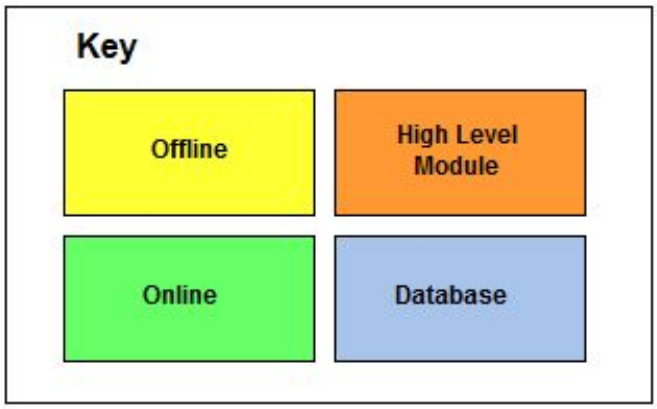
The team shall use SCRUM style development

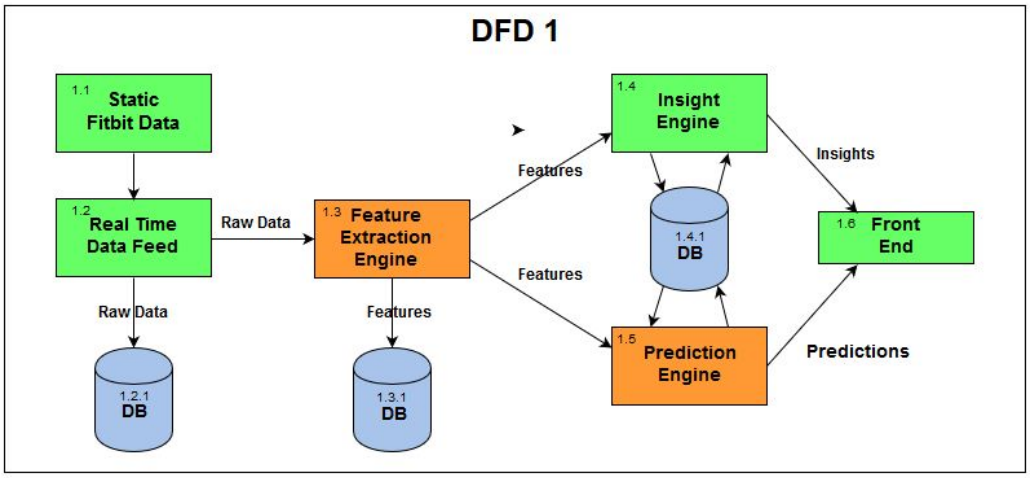
6. Other Requirements

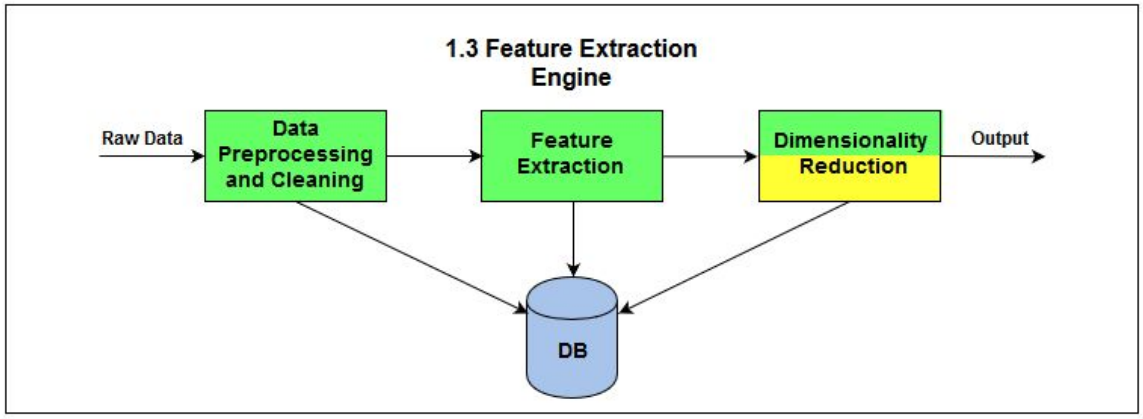
Appendix A: Glossary

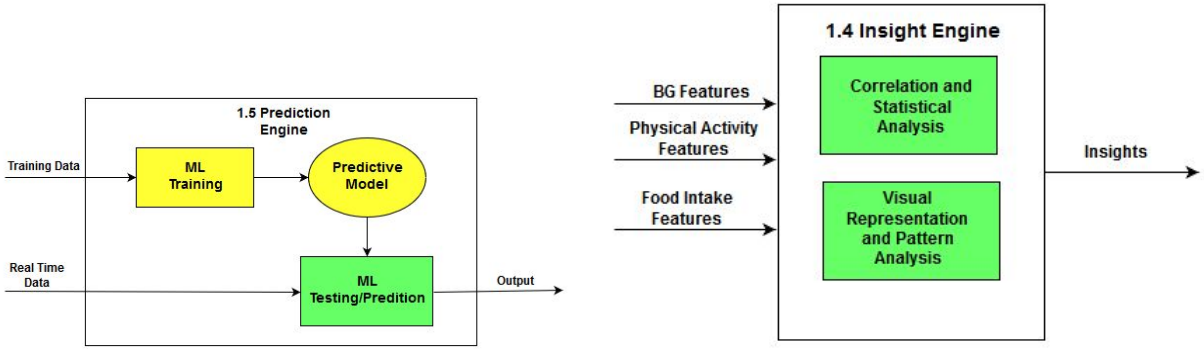
DPM	Diabetic Patients Monitoring
SDD	Software Design Document
BG	Blood Glucose Level
CGM	Continuous Glucose Monitoring
SG	Sensor Glucose Data from Real Time Glucose Monitor
RNN	Recurrent Neural Network
API	Application Programming Interface

Appendix B: Analysis Models




DFD 1
1.1 Static Fitbit Data
1.2 Real Time Data Feed
Raw Data
1.3 Feature Extraction Engine
1.4 Insight Engine
Features
Features
1.4.1 DB
Insights
1.6 Front End
Raw Data
Features
1.2.1 DB
1.3.1 DB
1.5 Prediction Engine
Predictions


1.3 Feature Extraction Engine
Raw Data
Data Preprocessing and Cleaning
Feature Extraction
Dimensionality Reduction
Output
DB


1.5 Prediction Engine
Training Data
ML Training
Predictive Model
Real Time Data
ML Testing/Predition
Output
1.4 Insight Engine
BG Features
Physical Activity Features
Food Intake Features
Correlation and Statistical Analysis
Visual Representation and Pattern Analysis
Insights

Appendix C: To Be Determined List

Modules	**Technology**
Insight Engine	Python, Pandas, Cassandra
Prediction Engine	TensorFlow, Scikit Learn, Python Flask API, Cassandra
Front End	HTML, Javascript