Software Engineering

Module: Model Based Software Engineering

Modelica Projects - Insulin Pump

AY 2020/21

Enrico Tronci
Computer Science Department, Sapienza University of Rome
Via Salaria 113 - 00198 Roma - Italy

tronci@di.uniroma1.it

Version: 2020-12-29

1 Outline

All running examples on your textbook can be used as Modelica projects. The *insulin pump* is one of those.

This document outlines possible requirements for the *insulin pump* project. This is just a possibility and you may freely explore different approaches. If in doubt, just send me an email: tronci@di.uniroma1.it.

2 Insulin Pump

As usual we should identify the following four main components:

- Environment.
- System model.
- Functional requirements.
- Non-functional requirements.

2.1 Environment

As for the insulin pump, the environment consists of the food ingested by the patient. As usual this can be modelled as a Markov Chain. For example, a diligent patient may ingest a moderate amount of carbohydrates every 8 hours or so, whereas a *greedy* patient may eat more and perhaps more frequently.

2.2 System model

Our system consists of at least two components: the patient and the pump (i.e., the software controlling the pump in our setting).

2.3 Patient model

The patient model takes as input the insulin dose provided by the pump and the food intake from the environment.

The patient model output is the blood glucose concentration.

Of course it is not our goal here to elaborate a model linking insulin, food and glucose. We can just use a model from the literature. You may use the one from the recent paper:

Roberto Visentin, Claudio Cobelli, Chiara Dalla Man. *The Padova Type 2 Diabetes Simulator from Triple-Tracer Single Meal Studies: In Silico Trials also possible in Rare but Not-So-Rare Individuals.* Diabetes Technology and Therapeutics, 2020.

This paper is included in this directory.

This is a paper modelling type 2 diabetes, whereas insulin pumps presently are mainly used for type 1 diabetes (indeed, it is much harder to certify software for type 2 diabetes). However the model above is self-contained and contains all elements we are interested in.

The appendix in the paper contains the equations for the model. The food intake is modelled through the variable *Dose* in equations A6, A7, A8. This connects the patient model to the environment.

Thus, the environment will define how the variable *Dose* changes. For example you may model 3 or 5 meals per day.

Of course, you do not need to enter the details of the model. It suffices to model inside Modelica the equations in the paper.

You should use the Modelica keyword `equation` (no `algorithm` in the patient model).

Parameters for the equations are in the tables in the above paper.

Accordingly, the patient will look something like:

```
...
input Real Dose;
input Real insulin;
output Real glucose;
...
equation

der(...) = ....
der(...) = ...
```

2.4 Pump model

The pump model (actually the model for the control software) takes as input the glucose level from the patient and returns as output the amount of insulin to be injected (which, in turn, will go as input to the patient model).

For the pump you may use rules as those in the book, or any strategy that you like. You may use an approach similar to the one shown during our classes to optimise the parameters of the control strategy. You may use a sampling time of 5 minutes.

Accordingly, the pump will look something like:

```
...

input Real glucose;
output Real insulin;
...

parameter Real T = 300;  // (we measure time in seconds, 5 minutes are 300 seconds)

algorithm

when sample(0, T) then

insulin :=
if (glucose > 110)
   then
       pre(insulin)*1.1;
   else
       if (pre(insulin) < 90)
       then
       pre(insulin)*0.9;
       else
       pre(insulin);

end when;
```

2.5 Functional requirements

A few examples of functional requirements.

- *Safety*: glucose should never drop below 50 mg/dL (hypoglycemia, very dangerous).
- *Liveness*: glucose should stay as close as possible to 100 mg/dL.

2.6 Non-functional requirements

A few examples of non-functional requirements.

- The amount of insulin injected should be minimised.

- The control software sampling time should be maximised (5 minutes will do, but a strategy working with a sampling time of 10 minutes would be even better).

3 Final remarks

Start model development from the interfaces. So, just develop models that take inputs and output and just return dummy numbers. This will allow you to test if the models are properly connected. Then, you will focus on developing correct models for each single component.