mHealth and wearable devices

THEORY AND PRACTICE

Martin Úbl

2022

Table of contents

Introduction and context

Diabetes

- Type 1, 2 and others

We focus on Type 1 Diabetes

- Autoimmune
- Manifests in early childhood
- Insulin treatment
- Currently requires a substantial amount of wearable electronics

https://diabetes.zcu.cz/

Diabetes treatment

Insulin (Type 1, newly even for type 2)

- Insulin pen
- Insulin pump
  - Subcutaneous
  - Intradermal

Antidiabetic drugs (Type 2)

Insulin dosing

Bolus

- Manual

Basal

- Manual
- Automatic
  - How?

Measurement

Glucose concentration

- In blood
  - Glucometer
  - Sporadic
- Subcutaneously
  - CGM sensor
  - „continuous“

Typical „setup“

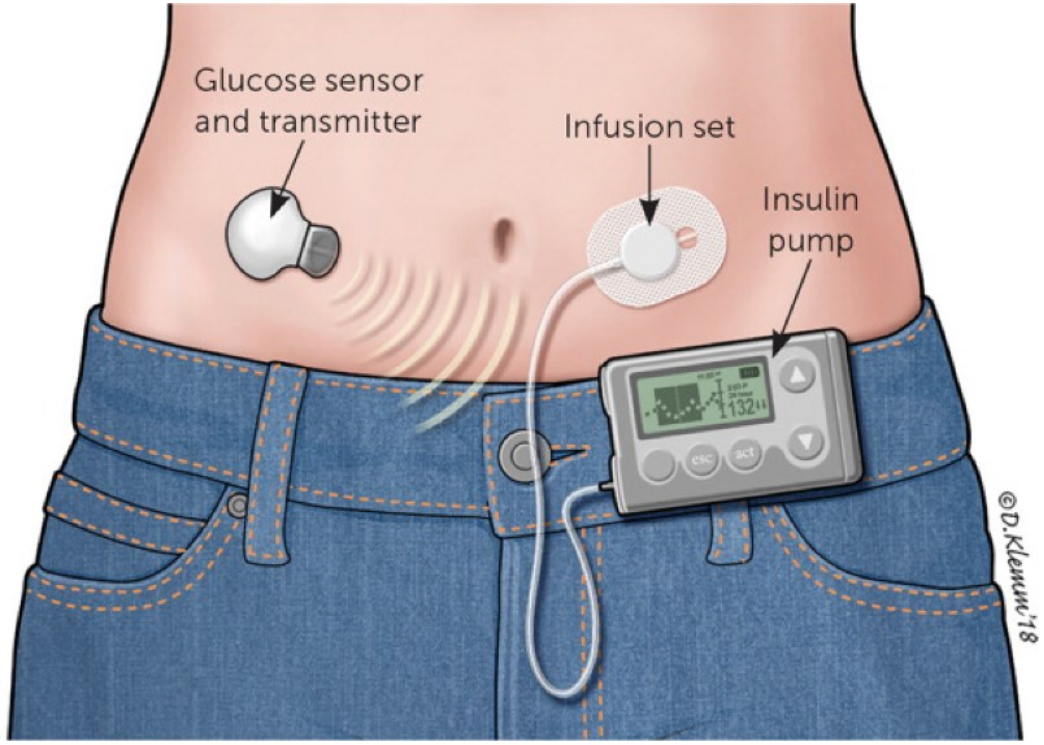
Sensor

Insulin pump

Infusion set

Controller (device)

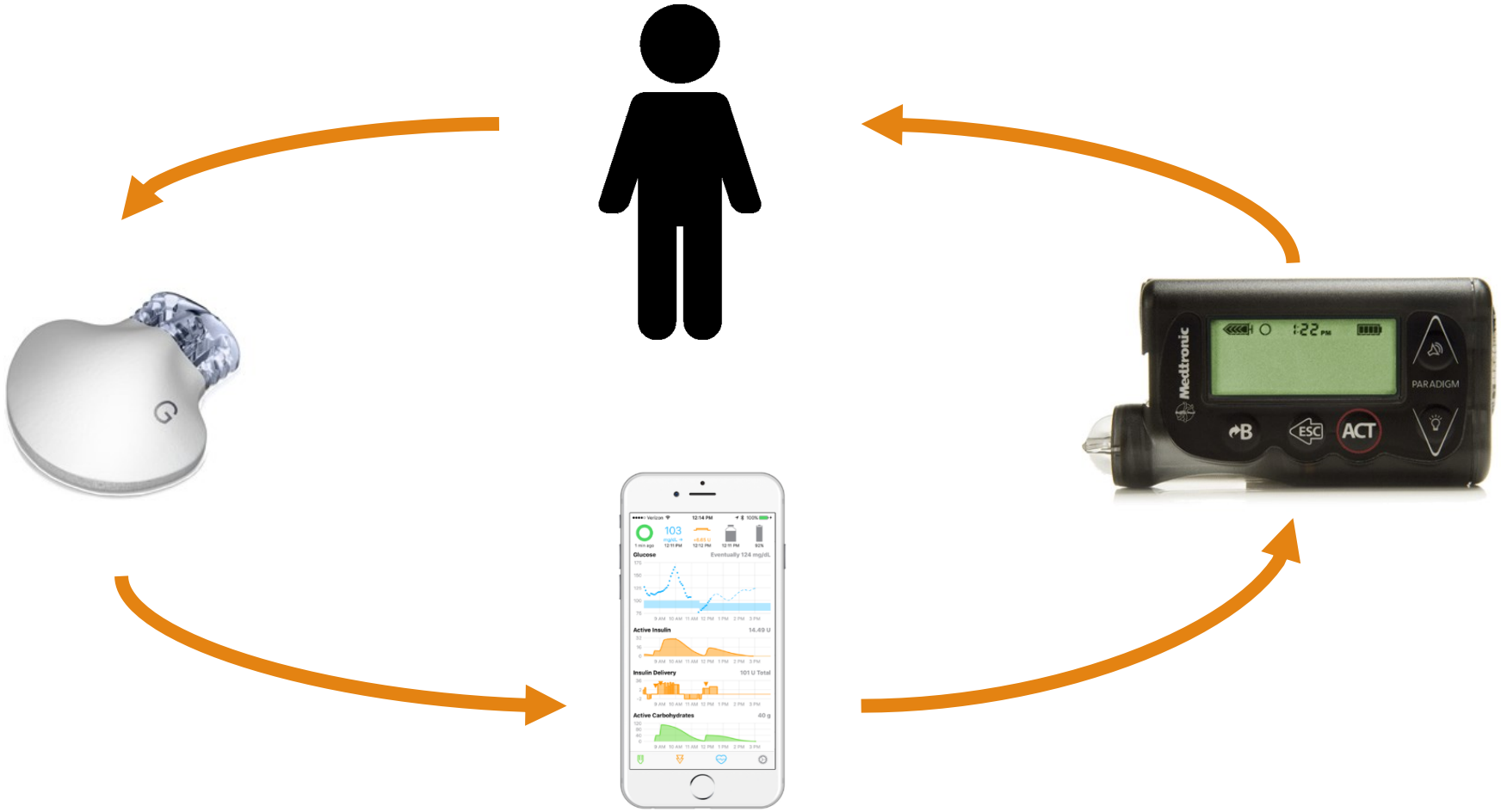
- Optional
- Resides between the pump and the sensor
- Mobile phone, smart watch



Source: https://diabeteson.com/technical-devices-that-improve-risk-factors-care-and-quality-of-life/

Discussion: Lots of wearable devices, how does it influence the patient mental state (e.g., in children patients)?

Closed control loop → artificial pancreas



Controller development

Certified

- „the correct and safe one“

DIY

- „the immediatelly deployed one“

DIY

Last 15-20 years

Patients themselves develop a treatment loop

- „Gluing together“ a number of components
- Algorithm prototyping

Is not a subject of certification

Risks vs. advantages?

DIY in diabetes treatment

OpenAPS

- open-source, JavaScript, Python
- *oref0* algorithm

AndroidAPS

- open-source, Java
- Runs the oref0 algorithm (JavaScript)

Loop

- iOS variant, Objective-C (later Swift)

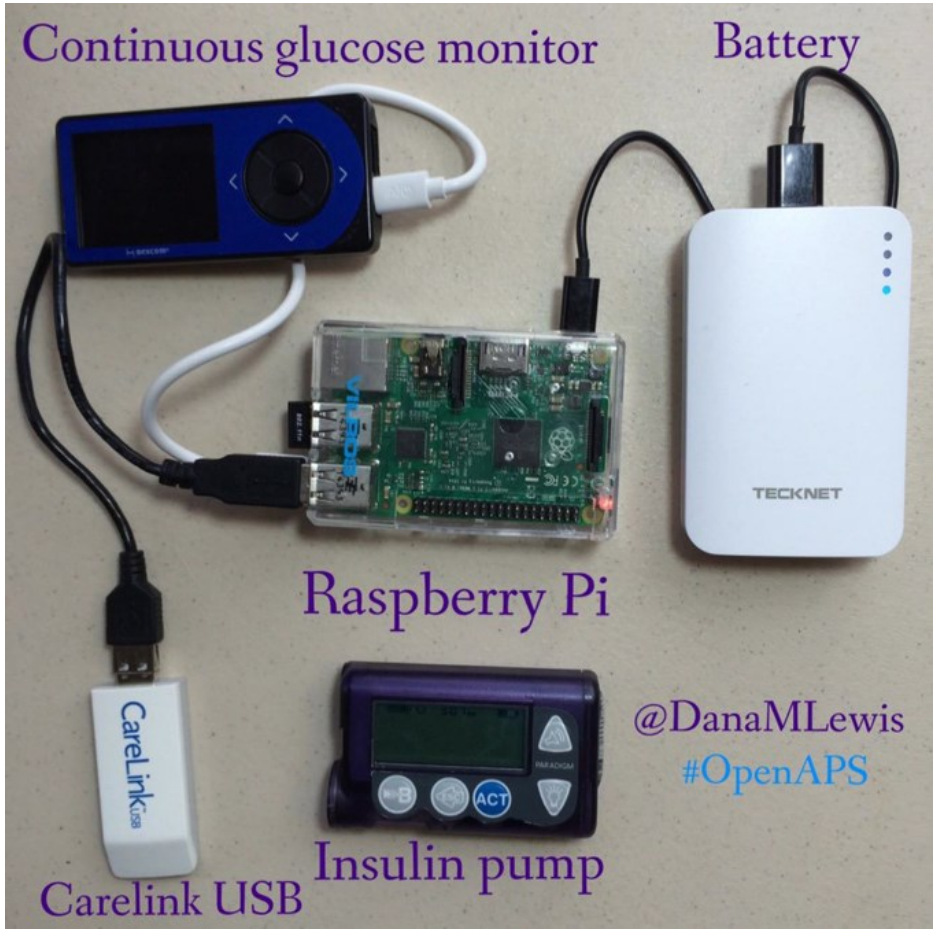
OpenAPS

mHealth?

Wearable electronics?

Safety?

- Old, deprecated devices
- JavaScript
- What if it fails?



OpenAPS - failure

How do we detect and/or solve a failure?

OpenAPS code does not look like a safe code…

```
// 38 is an xDrip error state that usually indicates sensor failure
// all other BG values between 11 and 37 mg/dL reflect non-error-code BG values, so we should zero temp for those
if (bg <= 10 || bg === 38 || noise >= 3) {  //Dexcom is in ??? mode or calibrating, or xDrip reports high noise
    rT.reason = "CGM is calibrating, in ??? state, or noise is high";
}
if (minAgo > 12 || minAgo < -5) { // Dexcom data is too old, or way in the future
    rT.reason = "If current system time "+systemTime+" is correct, then BG data is too old. The last BG data was read "+minAgo+"m ago at "+bgTime;
// if BG is too old/noisy, or is changing less than 1 mg/dL/5m for 45m, cancel any high temps and shorten any long zero temps
} else if ( bg > 60 && glucose_status == 0 && glucose_status.short_avgdelta > -1 && glucose_status.short_avgdelta < 1 && glucose_status.long_avgdelta > -1 &
    if ( glucose_status.last_cal && glucose_status.last_cal < 3 ) {
        rT.reason = "CGM was just calibrated";
    } else {
        rT.reason = "Error: CGM data is unchanged for the past ~45m";
    }
}
if (bg <= 10 || bg === 38 || noise >= 3 || minAgo > 12 || minAgo < -5 || ( bg > 60 && glucose_status == 0 && glucose_status.short_avgdelta > -1 && glucose_s
```

Console logging

No attempt of recovery after failure

```
try {
    iobArray.forEach(function(iobTick) {
    …80 lines of code…
} catch (e) {
    console.error("Problem with iobArray.  Optional feature Advanced Meal Assist disabled");
}
```

OpenAPS – will not fail?

„OpenAPS cannot fail“

- Really?
- Statement supported by „tens of thousands of run-time“

We can partially avoid failure by verification

- OpenAPS has not been verified
- To be verifiable, the code needs to be prepared for it
  - „spaghetti“ code of appx. 1600 lines of JavaScript certainly does not look like it is prepared

```
rT.predBGs = {};
IOBpredBGs.forEach(function(p, i, theArray) {
    theArray[i] = round(Math.min(401,Math.max(39,p)));
});
for (var i=IOBpredBGs.length-1; i > 12; i--) {
    if (IOBpredBGs[i-1] !== IOBpredBGs[i]) { break; }
    else { IOBpredBGs.pop(); }
}
rT.predBGs.IOB = IOBpredBGs;
lastIOBpredBG=round(IOBpredBGs[IOBpredBGs.length-1]);
ZTpredBGs.forEach(function(p, i, theArray) {
    theArray[i] = round(Math.min(401,Math.max(39,p)));
});
for (i=ZTpredBGs.length-1; i > 6; i--) {
    // stop displaying ZTpredBGs once they're rising and above target
    if (ZTpredBGs[i-1] >= ZTpredBGs[i] || ZTpredBGs[i] <= target_bg) { break; }
    else { ZTpredBGs.pop(); }
}
rT.predBGs.ZT = ZTpredBGs;
lastZTpredBG=round(ZTpredBGs[ZTpredBGs.length-1]);
if (meal_data.mealCOB > 0) {
    aCOBpredBGs.forEach(function(p, i, theArray) {
        theArray[i] = round(Math.min(401,Math.max(39,p)));
    });
    for (i=aCOBpredBGs.length-1; i > 12; i--) {
        if (aCOBpredBGs[i-1] !== aCOBpredBGs[i]) { break; }
        else { aCOBpredBGs.pop(); }
    }
}
if (meal_data.mealCOB > 0 && ( ci > 0 || remainingCIpeak > 0 )) {
    COBpredBGs.forEach(function(p, i, theArray) {
        theArray[i] = round(Math.min(401,Math.max(39,p)));
    });
    for (i=COBpredBGs.length-1; i > 12; i--) {
```

AndroidAPS

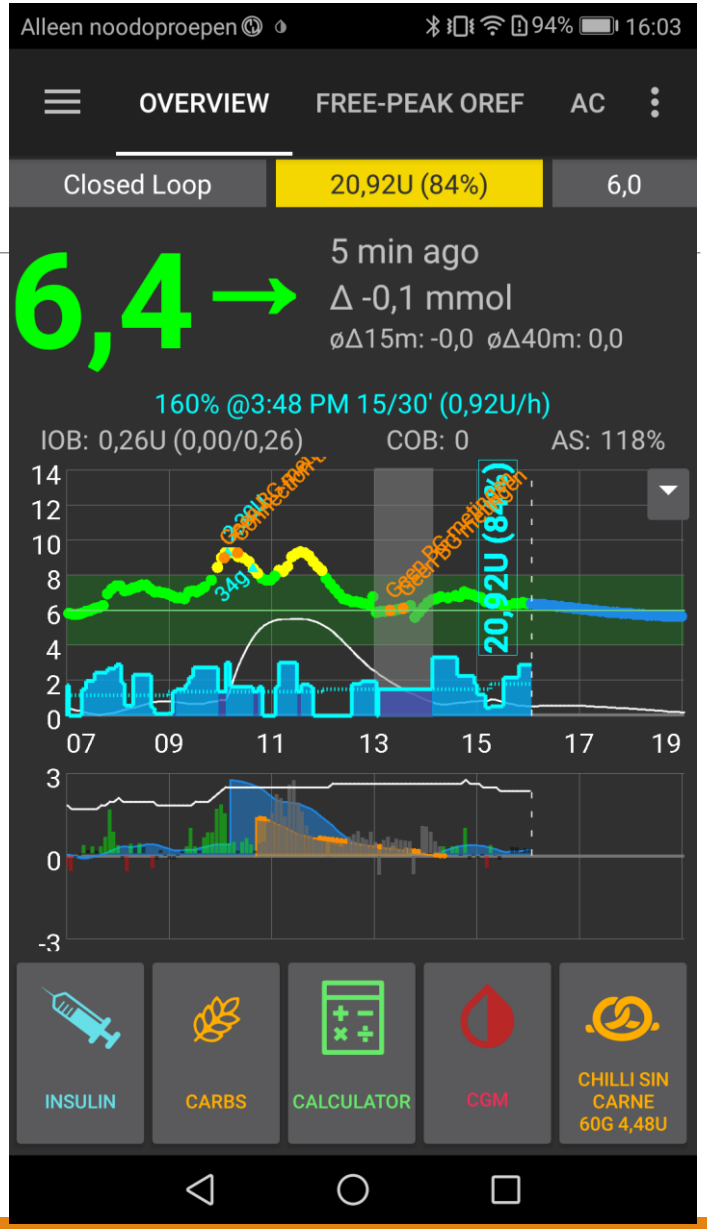
Similar situation

- Java aplication for Android
- Runs JavaScript for insulin dose calculation

Author trusts his own software to a degree, when he set it up in a closed loop mode for his own daughter (10 years old)

The code is in a similar state, as the OpenAPS one

Author himself proclaims, that he is „not a good programmer“



All systems

Wearable electronics

- Requires a communication protocol (network)

DIY systems use old, deprecated hardware

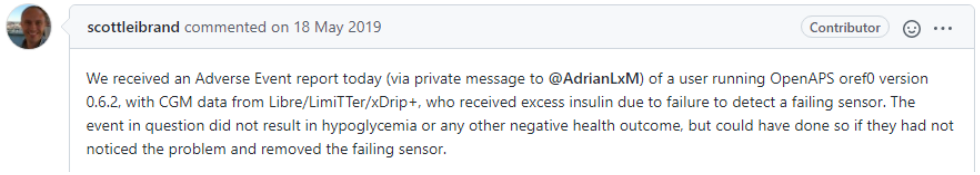
- A number of exploits in the protocol
- Buggy
- No warranty
- Who is responsible for injuries?
  - Patient sets it up on his/her own
  - A physician (diabetologist) tolerates the use, sometimes even encourages it

DIY systems fail

In fact, pretty often

- A selection of recent issues in the OpenAPS repository

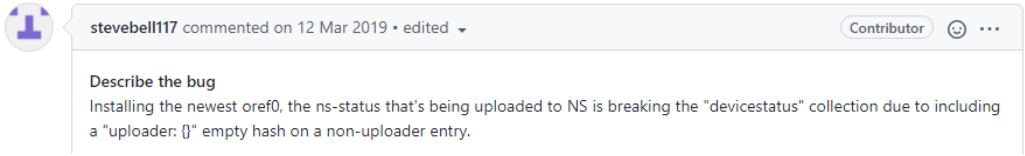
Too much insulin due to sensor failure



scottleibrand commented on 18 May 2019 — Contributor

We received an Adverse Event report today (via private message to **@AdrianLxM**) of a user running OpenAPS oref0 version 0.6.2, with CGM data from Libre/LimiTTer/xDrip+, who received excess insulin due to failure to detect a failing sensor. The event in question did not result in hypoglycemia or any other negative health outcome, but could have done so if they had not noticed the problem and removed the failing sensor.

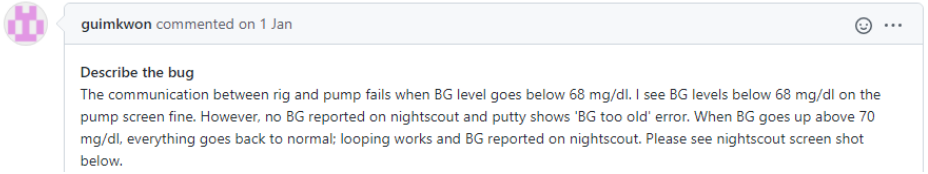
Insufficient testing (CI/CD)



stevebell117 commented on 12 Mar 2019 • edited — Contributor

Describe the bug
Installing the newest oref0, the ns-status that's being uploaded to NS is breaking the "devicestatus" collection due to including a "uploader: {}" empty hash on a non-uploader entry.

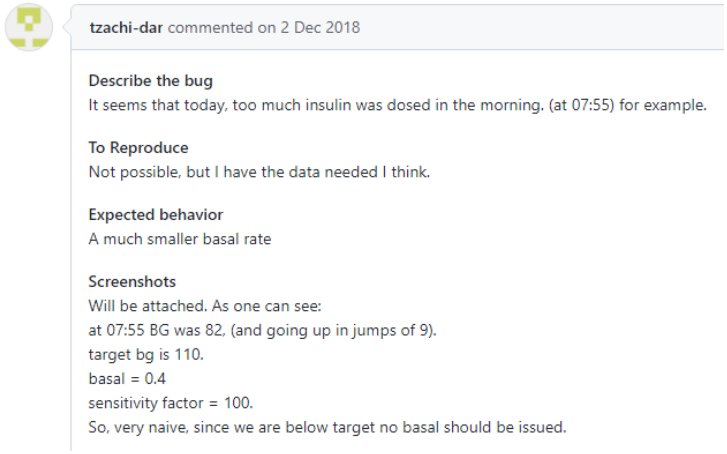
Bugs in code due to its unmaintainability



guimkwon commented on 1 Jan

Describe the bug
The communication between rig and pump fails when BG level goes below 68 mg/dl. I see BG levels below 68 mg/dl on the pump screen fine. However, no BG reported on nightscout and putty shows 'BG too old' error. When BG goes up above 70 mg/dl, everything goes back to normal; looping works and BG reported on nightscout. Please see nightscout screen shot below.

Contradictory control rules



tzachi-dar commented on 2 Dec 2018

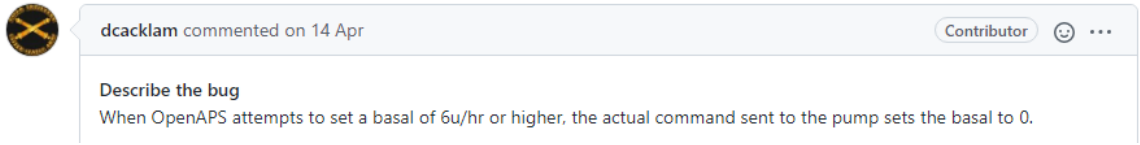
Describe the bug
It seems that today, too much insulin was dosed in the morning. (at 07:55) for example.

To Reproduce
Not possible, but I have the data needed I think.

Expected behavior
A much smaller basal rate

Screenshots
Will be attached. As one can see:
at 07:55 BG was 82, (and going up in jumps of 9).
target bg is 110.
basal = 0.4
sensitivity factor = 100.
So, very naive, since we are below target no basal should be issued.

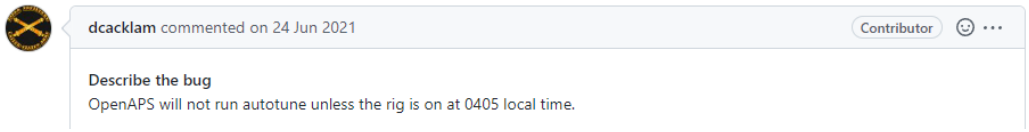
Silent failure of OpenAPS



dcacklam commented on 14 Apr — Contributor

Describe the bug
When OpenAPS attempts to set a basal of 6u/hr or higher, the actual command sent to the pump sets the basal to 0.

Weird runtime requirements



dcacklam commented on 24 Jun 2021 — Contributor

Describe the bug
OpenAPS will not run autotune unless the rig is on at 0405 local time.

According to what is DIY „safe“?

As users of a patient-driven technology, OpenAPS users are self-reporting improved A1C, day-to-day glucose levels, and quality of life. Safety features important to individuals with diabetes are perceived to be embedded into OpenAPS technology. Twitter analysis provides insight on a patient population driving an innovative solution to improve their quality of diabetes care.

https://doi.org/10.1177/1932296818795705

hours. In this highly selective population, user self-reporting suggests OpenAPS is much safer than standard pump with CGM therapy, measured by time spent in hypo- and hyperglycaemia, with no self-reports of severe hypo- or hyperglycaemic events 34.

https://doi.org/10.1111/dme.13816

OpenAPS is designed to be, and has been, far safer than standard pump/CGM therapy, as measured by duration of hypoglycemia and hyperglycemia, with no reports of severe hypo or hyperglycemic

https://openaps.org/2016/06/11/real-world-use-of-open-source-artificial-pancreas-systems-poster-presented-at-american-diabetes-association-scientific-sessions/

https://dx.doi.org/10.1177%2F1932296816665635

Discussion and Conclusions: Closing the loop with OpenAPS in people with T1D is effective in decreasing A1c and %TIHypo, without any serious adverse event. Of note, these results were obtained with people who showed a good baseline metabolic control (A1c of 7.17%). However, we need to study OpenAPS implementation on a larger sample of people with T1D and with a

https://doi.org/10.2337/db18-993-P

DIYPS[16] and the #OpenAPS project.[17] The dangers posed to patients from the do-it-yourself artificial pancreas may not be from individuals with malice, but rather from users with an excess of enthusiasm and a shortage of knowledge and experience.

https://doi.org/10.1177/1932296815583334

Physicians love DIY

Most physicians only see the results

- Results are mostly good

Psychological aspect?

„ends justify the means?“, knowingly ignoring technical imperfections

Requirements for mHealth devices

Algorithms are formally correct

- Verification
- Thorough testing within precisely bulit scenarios
  - in-silico (pre-clinical)
  - in-vivo (clinical)

Fault-tolerance and recovery

- Fault-tolerant properties
- Verification

Security

Lifecycle

- warranty, updates, regular technical maintenance, …

Certification

FDA (USA), EMA (Europe)

Very difficult process

- Long
  - Years of work (paperwork and additional work towards formal requirements)
- Expensive
  - Even tens of millions $
- Laborious

Certified devices can guarantee certain degree of safety and correctness

Classification of medical equipment (FDA)

1. Class I
   - Minimal to no risk, do not directly affect patient‘s health
   - E.g., fitness bands, thermometers, … even bandages and similar
2. Class II
   - Moderate risks, may affect patient‘s health
   - E.g., blood pressure meter, insulin pump (open-loop), glucometer, … even scalpels and needles
3. Class III
   - High risk, affects patient‘s health, may cause serious injury or even death
   - E.g., automatic insulin pump (closed-loop), CGM sensor, pacemaker, … even cochlear implant and joint replacement

Equipment approval

1. Class I
   - Register your product by the FDA
   - Some exceptions may include additional paperwork
2. Class II
   - Performance and effectivity is evaluated; even on market, they require collecting feedbacks and monitoring (for adverse effects and similar)
   - Devices must have a unique serial number, patients must be registered
3. Class III
   - Must undergo exhaustive testing and verification process
   - Clinical studies with large number of participants
   - Intentionally is a long process
     - If there is a bug in device code or hardware, the longer time, the greater probability of failure

Verification of algorithms and devices

Systematic testing of all possible states and validating responses

Simple example:

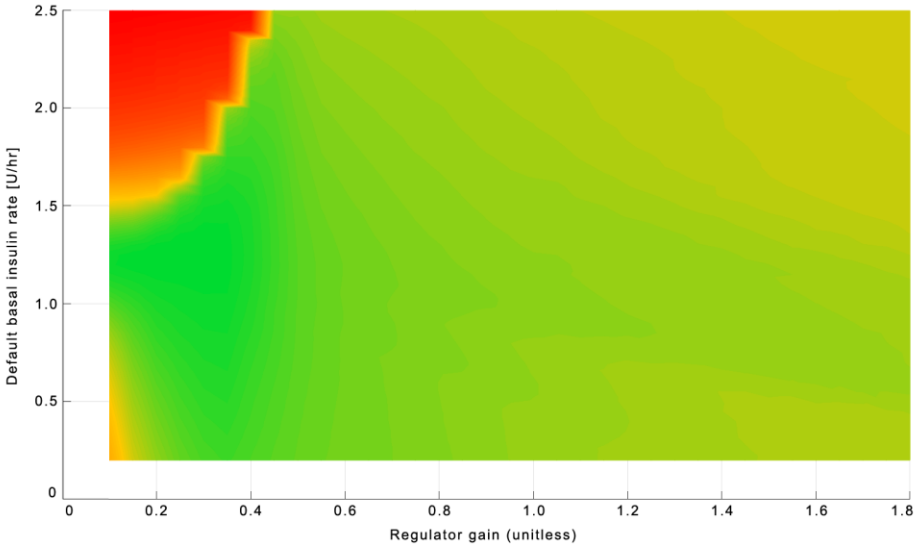
- Two-parametric controller
- Cartesian product of stepped parameter values in some (safe) boundaries
- Metric evaluation on a number of scenarios
- Attempt to identify „faulty“ combinations

Simple example

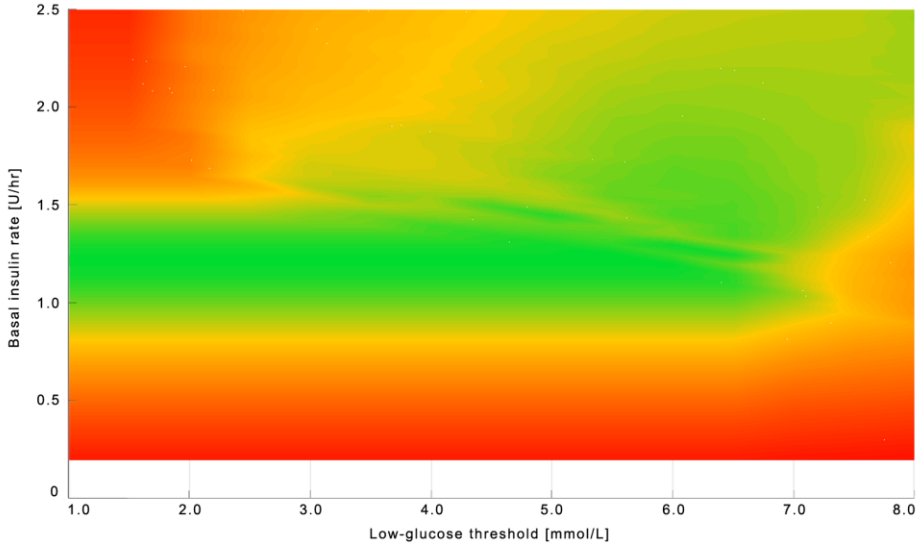
- 3 controllers, all having 2 parameters
- Only 2 of them have visible safe regions

- Legend:
  - Red – probably lethal
  - Yellow – edge case, potentially dangerous
  - Green – the best the controller can do

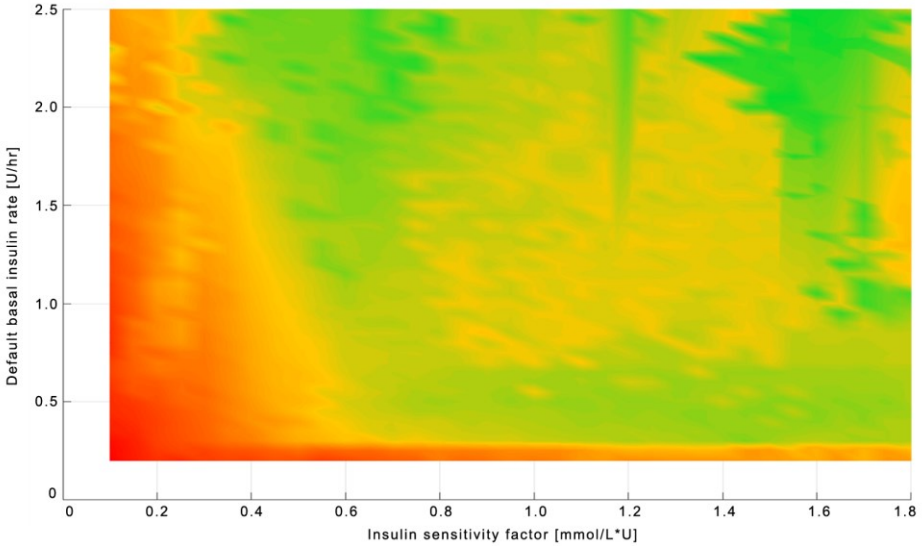
BetaPID – adaptive PID controller



LGS – treatments standard, certified



OpenAPS/oref1 – DIY



Wearable devices

Mobile phone, smart watch, fitness bands, but also CGM sensorts and more

A lot of sensors

- Lots of data
- Lots of possibilities

Accelerometer
Magnetometer
Ambient light sensor
GPS
Heartbeat sensor
Electrodermal activity sensor
Blood pressure sensor
Oxymeter
…

Wearable devices - data

Personalized medicine?

- Treatment model personalization

Telemedicine?

- Physicians always have recent data
- A parent always sees recent data of his/her child

Development of new physiological/treatment models?

- Datasets for initial cross-validation

Smart clothes

Not exactly a recent trend

Development of electronics-enhanced clothes

- Health monitoring
- Work assistance, safety
- Cool effects

Hasiči testují nový chytrý oblek. Umí věci jako ze superhrdinského filmu

18. 12. 2017

V Regionálním inovačním centru elektrotechniky (RICE) při Fakultě elektrotechnické Západočeské univerzity dokončili projekt vývoje „odlehčeného“ chytrého zásahového oděvu, který dostal jméno smartPRO2. Oblek budou nyní testovat hasiči a v průběhu příštího roku by měl být uveden na trh.

Wearable devices - data

Problems?

SmartCGMS

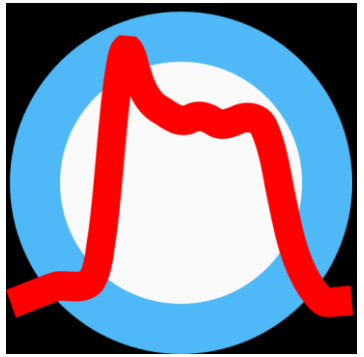
Framework designed and implemented on our department

Signal analysis framework and architecture

Built in such a way, that it may reach production qualities

- Fault-tolerance
- Verifiability
- Simplicity
- Stability
- Multi-platform
- Effectivity, low-power

Supports simulations and real-time use



SmartCGMS

Implementation split into modules of various types
- Filter
- Model
- Signal
- Solver
- Metric
- …

Every module can be verified separately
- Simplifies the verification process

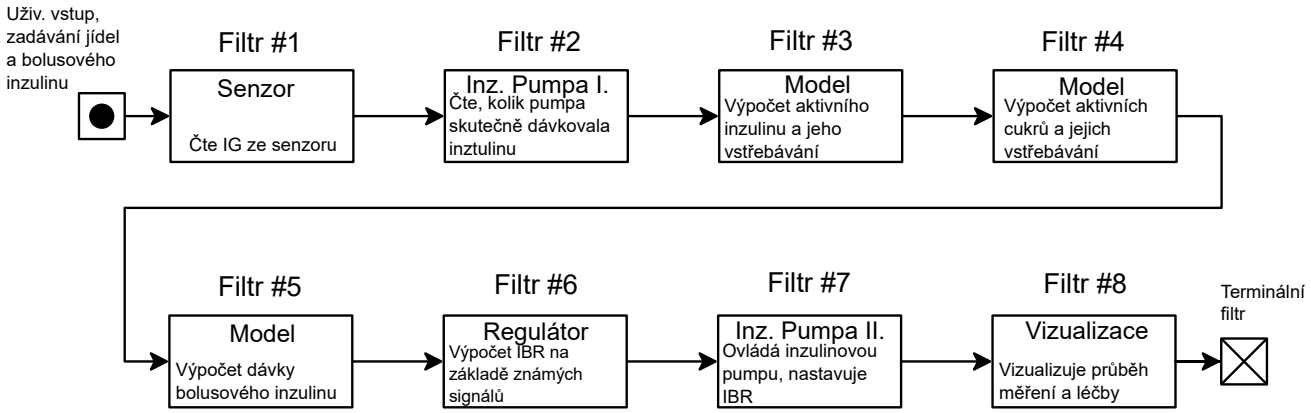
New module = verification of a single module
- It is not necessary to verify the whole system

From simulation to real-world in just a few steps
- Matter of a single module swap

SmartCGMS

Linear connection of filters

Message passing („from left to right“)




```
struct TDevice_Event {
  NDevice_Event_Code event_code;
  GUID device_id;
  GUID signal_id;
  double device_time;
  int64_t logical_time;
  uint64_t segment_id;
  union {
    double level;
    IModel_Parameter_Vector* parameters;
    wstr_container* information;
  };
};
```

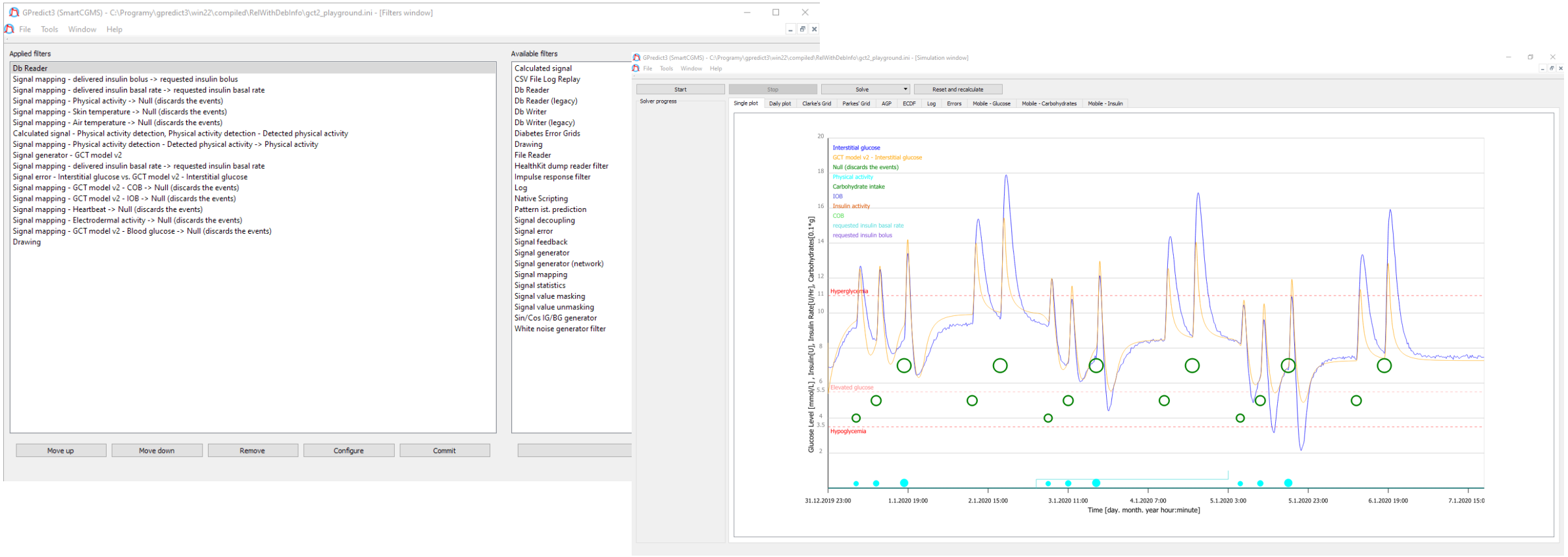
SmartCGMS

Fulfills a role of the back-end – framework, set of components and SDK

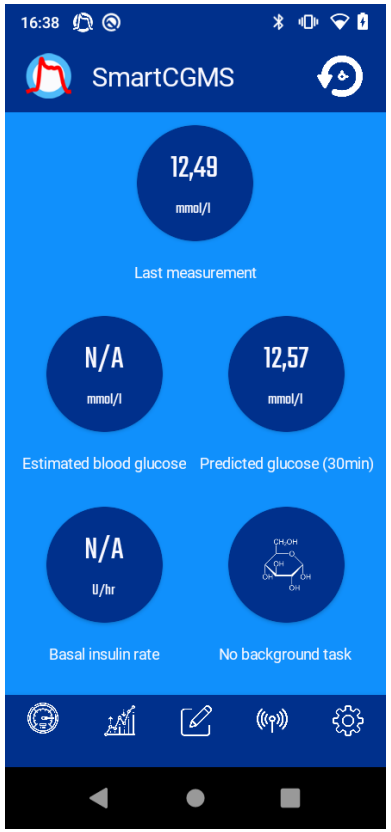
Front-ends

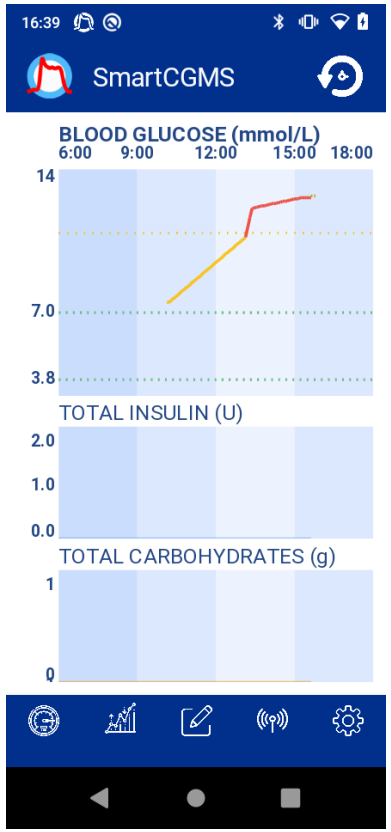
- gpredict3 – science and development
- SmartCGMS Mobile – patient monitoring
- Icarus has Diabetes - game
- Pump-Trainer – education of newly diagnosed patients

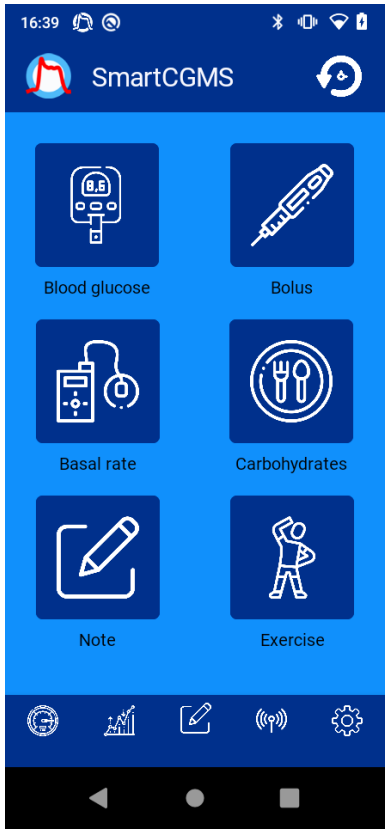
SmartCGMS – gpredict3

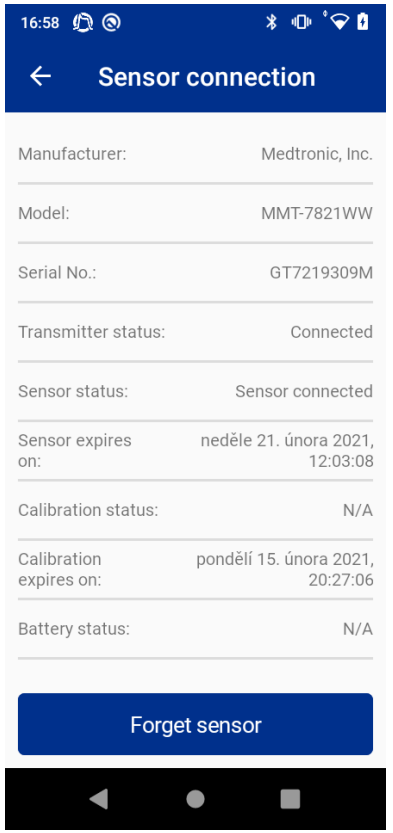


SmartCGMS – Mobile

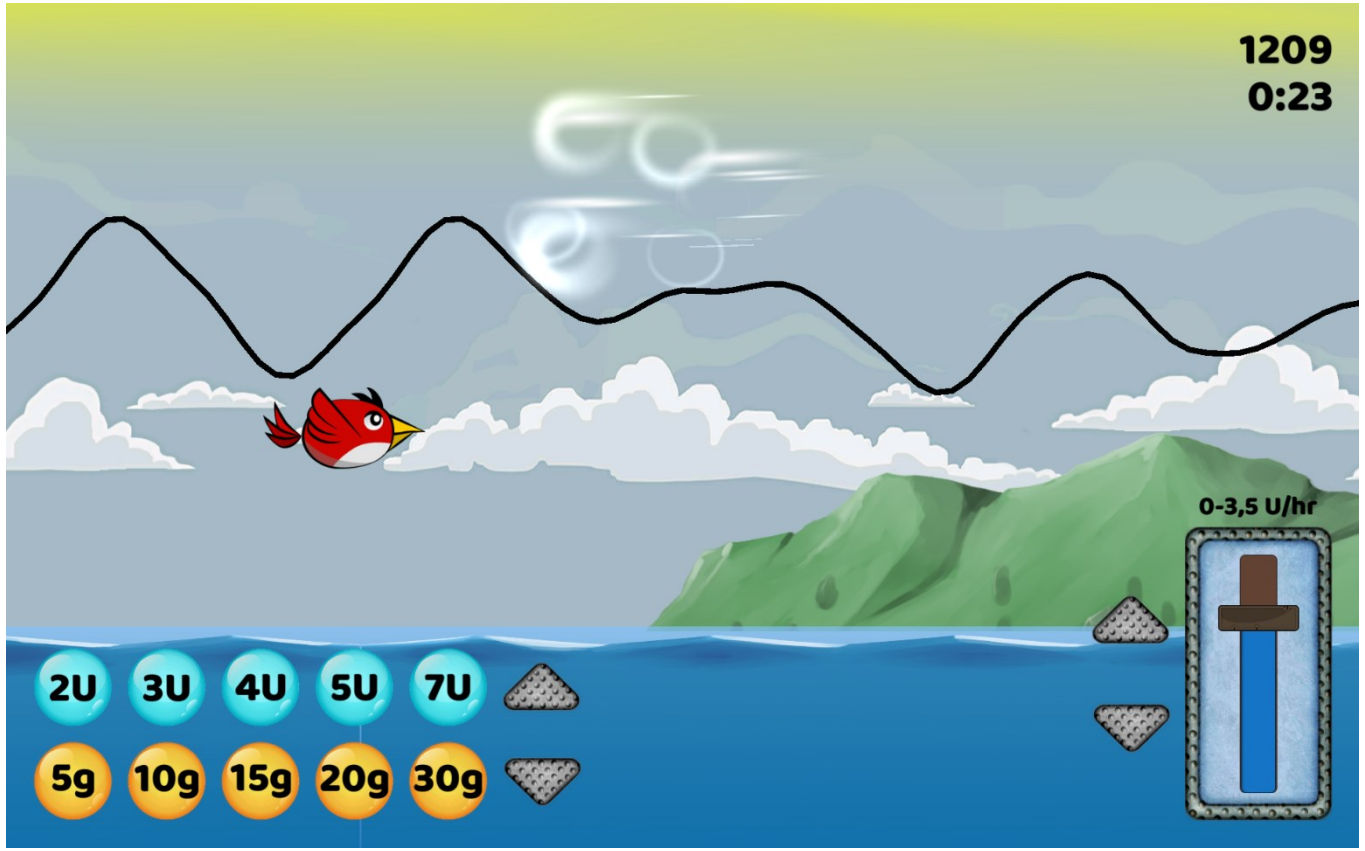




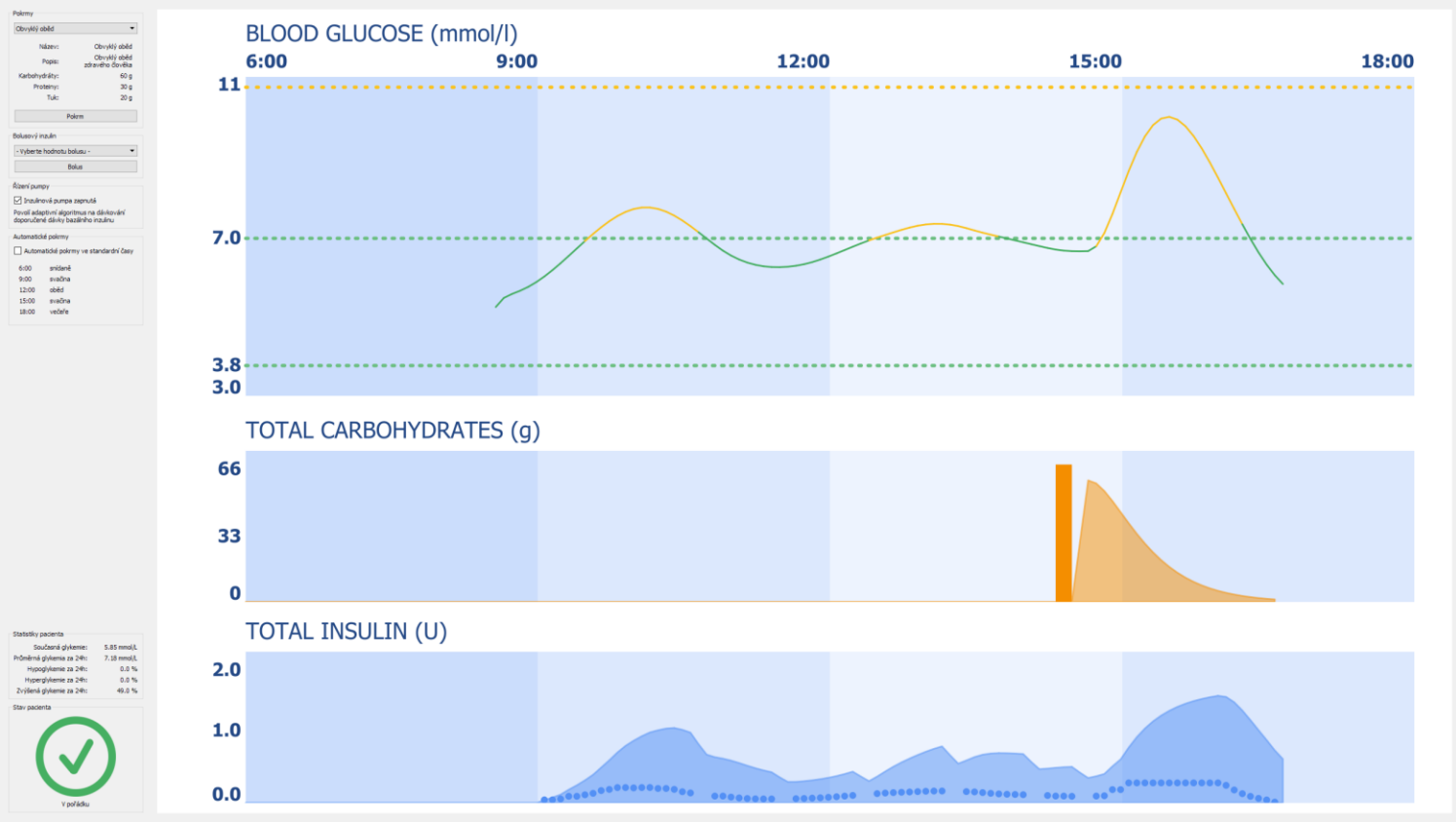




SmartCGMS – Icarus has Diabetes



SmartCGMS – Pump-Trainer



Thank you for your attention

Questions, discussion…