A developer's eye on smart glasses

In a Horizon 2020 project led by a major eyewear company, an advanced smart glasses product was developed that supports users in their daily activities. Consortium member Alten describes the main technical challenges and explains how they were solved.

Tamas Niks
Jacco van der Spek
Bernard Venemans

What if your glasses were to monitor your environment and support you in your behavior? This type of smart glasses has been developed in a project as part of the European Horizon 2020 program. The sensors and electronics inside the glasses allow the wearer to monitor their activity and exposure to light, leading to a healthier and happier life. The European consortium was led by one of the major companies in the eyewear industry. Alten developed the entire software chain: firmware, algorithms, mobile applications, cloud storage and data analysis.

Several quality requirements apply to such a product. First, the measurements must be precise and consistent to gain user trust. In addition, changes in the environment or behavior need to be processed quickly for minimal latency. Second, the energy consumption must be low, so that it's possible to wear the glasses for at least a full day before having to charge the batteries. Third, memory usage must be low to keep the hardware costs at a minimum. Finally, personal data must be secured and only accessible to entitled stakeholders. Based on these requirements, an IoT architecture was established for the product, consisting of the smart glasses with firmware, a smartphone with an app for the user and a cloud environment for data storage and processing.

Being the actual IoT device, the smart glasses are equipped with a broad variety of sensors. The firmware controls these sensors and numerous advanced algorithms turn their raw outputs into useful information. The sensor data is processed real-time for high reliability, minimal latency and to reduce the data transmission. Bluetooth Low Energy (BLE) was chosen to transfer the data to the mobile app in an energy-efficient way. To minimize power consumption, the glasses have no Wi-Fi capability, which means that the app is needed as a gateway to the cloud environment.

Energy spender

Two aspects were critical for the development of the firmware: battery life and RAM usage. Since the goal was a battery life of at least a day and the glasses are equipped with a relatively small battery, the challenge was to minimize energy consumption. To start the optimization, the team did some research and measurements to find the most energy-hungry components. These turned out to be the CPU, the BLE stack and a built-in notification LED.

For all three, the approach was simple: use them as little as possible. The amount of time the LED was turned on has been minimized. Together with the UX designers, a trade-off was made between visibility and energy consumption. To throttle the CPU's consumption, optimizations in the real-time operating system were made to put the processor to sleep whenever possible. Furthermore, to facilitate more sleep time for the CPU, the sampling frequencies of the

sensors were lowered as much as possible without introducing a noticeable latency.

The BLE stack was the biggest energy spender by far: in the original design, all sensor data was being streamed to the mobile application for processing. To limit stack usage, the design was altered to run (parts of) the algorithms in the firmware. In addition, an asynchronous communication model was implemented, which only transfers data when the data changes. The extra CPU load, and thus energy consumption, was negligible in comparison to the prolonged battery life due to reduced BLE usage.

The implementation of the numerous algorithms in firmware introduced a second challenge: limited RAM. The algorithms were designed to run on a desktop or mobile phone with unlimited RAM. Therefore, changes in the algorithm programming were needed, preferably without affecting the quality of the outcome. Raw data is buffered in a ring buffer in order to implement an efficient jumping window. Statistics for classification algorithms are computed using the 'running' implementation (eg running mean, running variance), which only requires a few variables to store the intermediate result. Overcoming these two challenges resulted in a doubled battery life.

Modular setup

A mobile app visualizes the information to the user and, based on data received from the glasses, gives warnings in case of dangerous situations. Through a setup wizard, the data is configured and calibrated for the user's personal conditions and physical characteristics. The mobile app is available for both Android and IOS.

In the early days of the project, cross-platform solutions offered insufficient BLE support. Therefore, the Android and IOS apps were developed in their respective native languages Java and Swift. One of the challenges for the mobile application was to keep the Bluetooth connection with the glasses alive and stable. For both Android and IOS, built-in solutions exist for setting up the connection. However, neither of these supports a continuous connection with an external device (except when using a streaming device, like a speaker or headset). The reason is that both Google and Apple want to make sure that the smartphone has an optimal battery life, without energy-consuming background applications. As a solution, a watchdog system was implemented that reboots the connection as soon as the application is forced to close.

The mobile application streams data to the cloud environment. This is managed by the Azure IoT hub, which allows devices to connect with the required high level of security. Once the connection is established, the mobile app will start streaming data for each new message. To reduce the number of transmissions and to preserve energy, the streaming frequency depends on the actual usage of the glasses.

Upon receipt by the IoT hub, the data needs to be processed and stored according to European security regulations. A distinction was made between different types of data: sensor-related information and user data. These were separately stored and are only accessible with proper authentication. Multiple APIs were implemented to handle this, making the cloud solution modular in its setup.

A PowerBI environment displays all the sensor-related information for visual inspection and processing. This data from actual usage enables further optimization of the features and algorithms in a next stage of market research and validation.

Tamas Niks and Jacco van der Spek are software engineers with Alten. Bernard Venemans is a project manager at this multinational technology consulting and engineering company.

Edited by Nieke Roos