Sensor Throne: Improvement and Monitoring of Sitting Posture through a Mobile Application

Bence Both*, István Lukács*, Beáta Brassai†, Zsolt Szabó†, and Csaba Sulyok*
* Faculty of Mathematics and Computer Science, Babeş-Bolyai University
RO-400084 Cluj-Napoca, Romania
† Softech SRL
RO-400458 Cluj-Napoca, Romania
bence.boths@gmail.com; lukacsisti25@gmail.com; brassai.beata@codespring.ro;
szabo.zsolt@codespring.ro; csaba.sulyok@ubbcluj.ro

***Abstract*—Within the framework of the Sensor Throne project, a smart chair is developed with the aim of monitoring and improving correct sitting posture. The system consists of two components: the hardware component which consists of a microcontroller mounted to an office chair, along with its peripherals (battery, sensors, Bluetooth module), while the software component consists of a mobile application.**

The microcontroller communicates with the application via Bluetooth using its own text-based protocol. The application provides an interactive interface for the user, where they can view their current posture, statistics related to sitting on the chair, and customize the application settings.

The application is currently available in three languages, automatically selecting based on the language set on the phone. The user receives notifications in case of incorrect or prolonged sitting as well as low battery voltage of the microcontroller.

I. Introduction

Numerous recent surveys show a significant increase in the number of sedentary jobs. In Europe, 39% of the population already engages in sedentary work [1], while in the American continent the number of jobs performed in a seated position has increased by 83% compared to 1950 [2]. Consequently, Americans now spend more time sitting throughout their lives than at any other point in history. Due to this shift in lifestyle people are more prone to adopt unhealthy sitting postures, leading to a rise in various health issues among office workers, including slowed metabolism, spinal problems, musculoskeletal disorders, and digestive complaints [3].

The aim of the Sensor Throne project is to create a chair that assists the user in monitoring their sitting habits and improving their posture. In the project the user is considered to be sitting correctly if all sensors built in the chair are in contact with the user's body. If only some of the sensors are active, it indicates an incorrect posture. If none of the sensors are active, it means that the chair is not in use.

Due to the increase in sedentary jobs, movements are beginning to form to introduce more possibilities for movement at work. Studies suggest the seat-cycle method [4] present noteworthy improvements in blood pressure, sleepiness rating and in lower back pain. However, changing working posture to standing does not fully resolve the musculoskeletal issues [5].

Several studies have been published examining the association between sedentary work and musculoskeletal disorders (MSD). Zeverdegani et al. [6] examine how patterns of MSDs relate to using ergonomic chairs in offices. They conclude that the ergonomic chair design is strongly associated with MSD patterns. Bontrup et al. [7] aim to investigate the relationship between back pain and occupational sitting habits; they show that individuals with chronic low back pain manifest a trend towards more static sitting behavior.

The literature therefore warns of the importance of sitting habits. The current paper addresses this issue through a mobile application aimed at ensuring proper posture using a microcontroller mounted on an office chair, and accompanying sensors. The communication between the microcontroller and the application occurs via Bluetooth. The application provides real-time visual feedback to give the user an overview of their sitting habits.

The smart chair possesses numerous functionalities. These include successful communication between the hardware device installed on the chair and the mobile application, based on a custom text-based communication protocol. The main view of the application allows real-time monitoring. In the following view, statistics based on seating data can be tracked through bar and column charts. For the purpose of simulating the hardware component, the application can also be launched in demo mode. Furthermore, the application is accessible in three languages.

The remainder of the paper is structured as follows. Section II provides an overview of the hardware components employed in the project, whereas Section III focuses on the components related to the application. Section IV highlights the utilized technologies. Section V provides a detailed analysis of potential avenues for further improvement.

II. Hardware components

A. Choosing the right hardware components

In order to equip the chair with the appropriate microcontroller, several factors need to be considered during its selection. One aspect is the requirement for the chair to be portable, which necessitates the installation of a battery. To minimize the frequency of battery recharging, it is necessary to select a low-power microcontroller. Achieving lower energy consumption also involves optimizing the communication between the application and the electronic components. Among the options of Bluetooth and Wi-Fi devices, Bluetooth has significantly lower energy requirements compared to Wi-Fi. Another consideration is the cost of the microcontroller. This aspect is important as the chair itself already incures expenses. It would not be cost-effective to install an expensive microcontroller if a cheaper alternative can meet the requirements. This is particularly significant for mass production purposes.

At the beginning of the project, several microcontrollers were compared based on the aforementioned criteria. Specifically, the following options were evaluated: Arduino UNO[1], Particle Photon[2], ESP32[3] and Raspberry Pi Pico W[4]. Ultimately, Raspberry Pi Pico was chosen.

During the development the chair was provided by the Antares company[5].

B. Assembly process

The unit installed on the chair consists of multiple hardware modules. Since the built-in Bluetooth module of the Raspberry Pi Pico was not supported during the assembly of the project, an external Bluetooth module needed to be acquired, which could be connected to the Pico.

The selected Bluetooth module is the HC-05[6]. The communication between the module and the microcontroller operates based on the UART protocol[7]. This is one of the most widely used communication protocols. Data transmission occurs bit by bit over a cable. Since the communication between the microcontroller and the Bluetooth module is bidirectional, both devices function as transmitters and receivers. This means that the transmitter (Transmitter-TX) signal needs to be connected to the receiver (Receiver-RX) signal.

Once the necessary components for communication are in place, the next step selecting the power supply. A Pico UPS A module is connected to the Pico, providing the necessary power supply through a battery. The communication between the module and the microcontroller takes place via the I2C protocol. The devices involved in the communication are assigned master and slave functions. Communication is carried out using two threads. One thread is the SDA (Serial Data) line, where message transmission and reception occur. The other thread is the SCL (Serial Clock) line, responsible for synchronizing data exchange. This protocol is used for monitoring the battery level.

As shown in Figure 1, in addition to the Bluetooth module and the UPS module, six sensors are connected to the microcontroller to six different pins, which are responsible for monitoring the user's seating position.

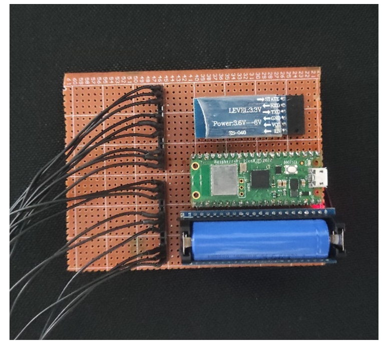

Fig. 1: Hardware component

C. Defining and testing the capacity of the battery

The microcontroller's power serves an UPS module which connecting to a battery with a capacity of 800 mAh and 3.7V. This battery has only been used in a testing environment, but there are plans to introduce a higher capacity battery in the future. This is necessary to allow the chair to be used freely for longer periods without the need of the constantly connected power source.

In order to estimate the battery's operating time several tests were performed. The battery was tested by monitoring its charge level while the microcontroller was in operation. The estimated operating time was 36.8 hours. In the second test case, a Bluetooth module was connected to the microcontroller, and the battery's operating time was 29 hours. During the measurements of the last test case, the entire system was operational and the hardware was communicating intensively with the connected mobile application. The maximum operating duration measured under these conditions was 11 hours.

Based on the conclusions drawn from the measurements, the 11-hour battery life is very short so there is need for optimization. This could be achieved through the implementation of a "deep sleep" functionality, which would significantly increase the operating time. In addition, using a higher capacity battery would also help to extend the operating time.

D. Emerging issues during the project

Several problems arose during the assembly and operation of the hardware components. The first problem arose with the sensors. The initially installed sensors exhibited unstable behavior. While they worked correctly in their default state, they provided inaccurate values when were active. Therefore, it was necessary to replace these sensors, and TTP223 touch sensors were chosen as replacements. During testing, authors discovered that the selected sensor type recalibrates itself every 6-8 seconds. This means that during calibration, the sensor adjusts its offset value. Hence, after calibration, the sensor needs to be touched again to provide the correct

[1]https://docs.arduino.cc/hardware/uno-rev3
[2]https://docs.particle.io/photon/
[3]https://bit.ly/42glf3J
[4]https://bit.ly/3N4AZlY
[5]https://scaune.ro/
[6]https://bit.ly/452LYmp
[7]https://www.circuitbasics.com/basics-uart-communication/

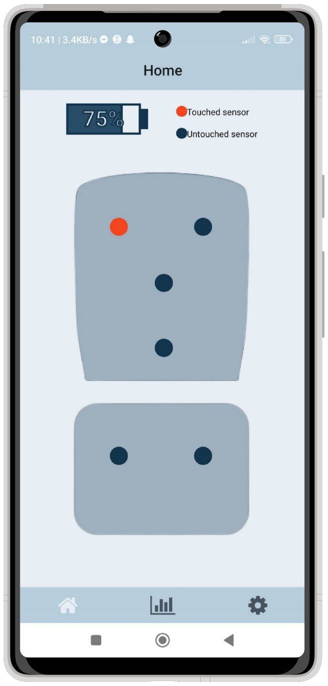

Fig. 2: Real-time monitoring

Fig. 3: Daily and yearly statistics

signal. Therefore, these sensors also seemed incompetent for the project. Eventually, the creators integrated into the chair mechanical microswitches. With the aid of microswitches, it can be measured whether the user is touching the appropriate areas on the chair. However, these newly selected sensors occupy more space compared to the TTP223 capacitive sensors, which posed installation challenges within the chair.

The support for the firmware of the microcontroller's built-in Bluetooth module also belongs to the problems that arise. At the beginning of the development, no firmware was available for the Raspberry Pi Pico W microcontroller that would enable the use of the onboard Bluetooth module. As a result, an external Bluetooth module has to be connected to the hardware component, and its operation has to be handled through software. Support for the aforementioned internal module was released in February 2023 [8], along with sample programs. Incorporating the built-in module into the system is part of the long-term plans.

The microcontroller has two cores, allowing for the development of a multi-threaded program. To ensure that all functionalities is necessary to execute in a timely manner, the software structure and operation of the microcontroller require careful planning.

III. Mobile Application

A. Onboarding

The first requirement for using the application is establishing communication between the phone and the chair. This connection is establishing via Bluetooth, so the user needs to pair their phone with the chair. To ensure that the application connects to the desired chair and to facilitate the user experience the developers introduce an onboarding process.

The onboarding process consists of 4 steps and is automatically displayed to the user when they attempt to connect their phone to the smart chair for the first time. The first view contains a welcome message. The next view displays a list of nearby available smart chairs. The user can select the desired chair from this list. Each chair will have a label or a QR code placed on it, which provides access to the chair's data. The user can obtain its name, which they can then select from the displayed list. The selection is followed by a confirmation process, which assures the user that they have connected to the correct hardware device and that the built-in sensors in the chair are functioning properly. This process involves touching the sensors on the chair in the order of their appearance on the phone screen. Once this is done, confirms a successful connection between the two devices, and after pressing the "OK" button, the application is ready to use.

B. Real-time monitoring

The main view of the application depicts a chair with the physical positions of the installed sensors. This view is responsible for real-time monitoring. When the user sits on the chair, the phone screen shows which sensor is currently active as can be seen in Figure 2. The sensors that are being touched or activated are highlighted with an orange fill on the screen. This allows the user to quickly notice if their sitting position is shifting in the wrong direction, as the orange fill disappears. Thanks to this functionality, the user can see how well they are sitting on the chair and identify any adjustments they need to make.

C. Statistics

The statistics functionality is one of the essential parts of the application. One advantage is that the user does not have to continuously pay attention to real-time data on his phone to observe their sitting habits. He can safely close the application and later revisit it to see the extent to which he have been sitting correctly. Another advantage is that it allows the user to track the changes in their sitting habits over the time.

The application provides various types of statistics, presented in the form of pie and bar charts. These views can be accessed by tapping on the graph icon at the bottom of the screen. Here, the user can examine their sitting habits in different time resolutions, including historical data.

When the user navigates to the statistics view a pie chart representing the daily sitting statistics is shown (see Figure 3). Three colors are visible: orange indicates incorrect posture, dark blue represents the time spent in correct sitting posture, and light blue represents the time when the chair is not in use. The explanation of these colors can be found below the diagram. Additionally, changing interesting facts about sitting are displayed continuously below the legends.

D. Notifications

Sitting for long periods of time, even with good posture, can be bad for one's health. Therefore, it is necessary to move at regular intervals. To ensure that users remember to do so without having to set reminders on their phones, the application sends notifications. There are three different types of notifications, two related to sitting and one related to the chair battery.

The first type of notification appears when the user is sitting in an incorrect posture. After a period of time, the phone notifies the user that the sitting position needs to be adjusted. The second type of notification appears when the chair has been in use for too long period, which can be harmful to health. In the application settings the user can personalize the length of this period. The last type is related to the battery level, it is sent when the battery charge drops to a low level (less than 15%).

E. Settings

The users can customize the application through various default settings. These settings can be accessed by tapping the settings icon located at the bottom of the screen.

In this section of the application, it is possible to see which chair is connected to the phone, as well the unique identifier of the chair. By tapping on the name of the chair, the user can rename it.

Users have the ability to customize their notification preferences by selecting which notifications they wish to receive and specifying the desired intervals for receiving them. More details about this can be found in Section III-D. The version numbers of the chair and the application are displayed at the bottom of the screen as shown in Figure 4.

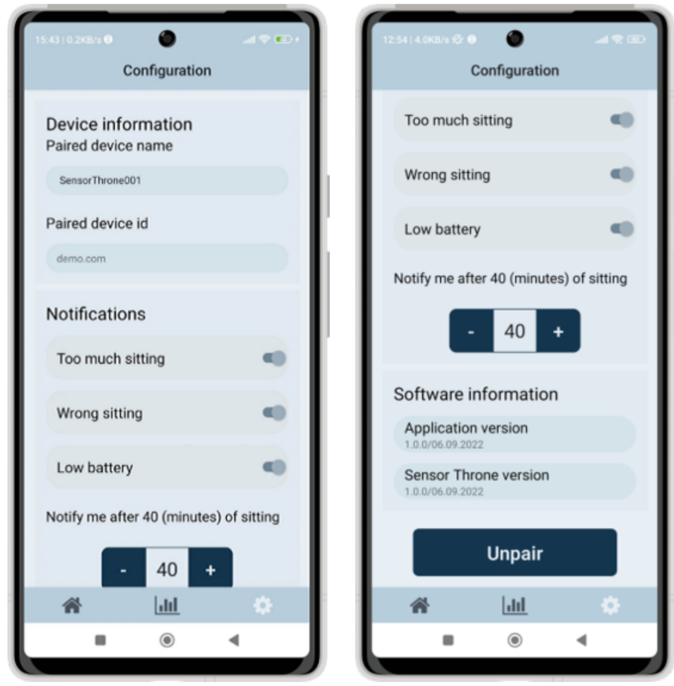

Fig. 4: Settings screen

F. Internationalization

Currently the application is available in three languages: Hungarian, Romanian and English. This is selected automatically based on the phone's language. In case the language set on the phone is not one of the three specified languages, the application will automatically load the English version.

G. Difficulties with permissions

To use the Bluetooth module of the mobile phone the application requires different permissions. In Android 11 and below is sufficient to request a single permission which is called `ACCESS_FINE_LOCATION`. This permission provides access to location information.

However, the introduction of the Android 12 causes a problem when establishing a Bluetooth connection because a single permission is no longer enough. With Android 12, in addition to location permissions two other Bluetooth specific permissions are required. The first one is `BLUETOOTH_SCAN` which allows the discovery of the available devices. The communication between two Bluetooth devices needs the `BLUETOOTH_CONNECT` permission [9]. Additionally enabling location services is crucial for the functionality. With permission of the user the location services turn on automatically.

H. Demo mode

The developers introduce a demo mode in addition to other functionalities. Thanks to this feature the development is easier and smoother. With the demo mode, all functionalities of the application can be tested without the use of hardware

```
APP goldfish_x86 0 REQ ONBOARDING step_s0~
PICO goldfish_x86 0 RESP ONBOARDING 0~
```

Fig. 5: Protocol messages

device. Can be viewed randomly generated statistics, real-time monitoring and the user can personalize the application settings.

I. Communication between the application and the chair

The communication between the application and the microcontroller installed on the chair is done via Bluetooth connection. In order to track and organize the communication between the hardware and software components, it was necessary the introduction of a communication protocol. The use of the protocol makes the system more stable and secure, and it also allows for error handling.

This code snippet in Figure 5 represents a message exchange between the application and the chair. The protocol consists of seven parts, each separated by a space, which terminates with the ∼ character.

1) The first component of the protocol message is formed from the identifier of the generating unit, which can take the next two values: `APP` and `PICO`. This indicates that the message comes from the application or from the microcontroller.
2) The second component of the protocol is the unique identifier of the sending unit, which allows to precisely identify which phone takes part in the communication.
3) The third component is the `Session ID`, which contains a numerical value. This value increments by one for each message pair. It allows to track the delivery of every message and helps in error handling. If one device sends a request and does not receive a response, it resends the message, thus eliminating data loss and increasing stability.
4) The fourth component is the command type, which can take the following values: `REQ`, `RESP`, `ACK`, `INFO`, and `ERROR`. According to the protocol definition, a `REQ` (request) message should always be followed by a `RESP` (response) message. An information message is indicated by the keyword `INFO`. If a sequence of messages occurs within a single command, the receiver acknowledges the message arrival to the sender using the keyword `ACK`. In case of an error, the `ERROR` command is included in the message.
5) The fifth component of the message is the command, which specifies the exact operation to be performed between the microcontroller and the application. It can take the following values: `ONBOARDING`, `GET_STATISTICS`, `SEND_SENSOR_STATUS`, `STOP_SENSOR_STATUS`, `SEND_PHONE_ID`, and `GET_BATTERY`. In the case of `ONBOARDING`, the process described in Section III-A is executed. During `GET_STATISTICS`, synchronization takes place between the chair and the application. When transitioning to the main screen, real-time monitoring is activated using the `SEND_SENSOR_STATUS` command. If another view is in focus, the keyword `STOP_SENSOR_STATUS` should be used. When the user wants to connect their phone to the chair, the application automatically sends the mobile identifier using the `SEND_PHONE_ID` command. The battery level can be obtained by using the `GET_BATTERY` code.
6) The penultimate part of the message is the actual information that the sender wishes to transmit. It is a plain character string that concisely contains the information.
7) The last part of the protocol message is the closing character, which, according to the protocol, is the ∼ character. This signifies the end of the message by the sending unit.

IV. Technologies

In this project, the firmware is written in *MicroPython* [10], which is a lightweight and efficient implementation of the Python 3 programming language. It includes a subset of the Python standard library and it is optimized to run on microcontrollers and in constrained environments. MicroPython is excellently suited for projects that prioritize low power consumption, low memory usage, and low computational performance.

The application is being developed using *React Native*[8], which is an open-source framework developed by Facebook that enables the development of mobile applications by combining the web-based React framework with native development. It builds applications using general JavaScript and React APIs and allows for development across multiple platforms (e.g., iOS and Android) simultaneously, significantly reducing development time and costs. The reduction in development time is further facilitated by the fact that when something is changed in the code, there is no need to rebuild the entire application, just reloading is sufficient.

Currently, the Sensor Throne application is only available on Android devices, as Android is the dominant operating system [11] on most users' devices. Support for the iOS platform is planned in the long term.

React Native pairs well with *TypeScript*, which is a strongly typed programming language that builds upon JavaScript and provides better tooling for code organization in projects of any size. It is advantageous to use TypeScript in the project because it adds several syntax enhancements to JavaScript, enabling early error detection in the editor. It helps to identify potential issues before they become runtime errors, making code maintenance and refactoring safer and more efficient.

V. Conclusion and Further Development

During the Sensor Throne project, a hardware component connected to a chair and its accompanying application were

[8]https://reactnative.dev/

successfully developed, which monitor the user's sitting posture and attempt to improve it through various diagrams and notifications.

The chair is equipped with a microcontroller responsible for monitoring the sensors and transmitting the data to the phone. Communication takes place via Bluetooth between the microcontroller and the phone.

Through the mobile application, the user can see their sitting posture in real-time and review statistics based on different diagrams. Additionally, the user has access to settings, allowing them to customize notifications. The application also supports multiple languages.

Optimizing energy consumption could significantly extend the battery life. This optimization could be achieved by introducing "deep sleep" functionality on the microcontroller side. When the chair is not in use, the microcontroller stops monitoring the sensors and enters a mode where it consumes minimal energy, resulting in serious energy savings.

Another optimization would be the utilization of the microcontroller's built-in Bluetooth module, which received software support in February 2023. By introducing this feature, the battery life could be extended, and the communication speed could be increased.

In the interest of better observation and thorough study, the incorporation of pressure sensors is also part of the long-term plans.

The Over-the-Air (OTA) update functionality would be also useful to enable remote software updates on the Raspberry Pi Pico. This would be beneficial as it eliminates the need for physically connecting to each electronic unit, and instead, the software update could be deployed to the microcontroller with a simple button press.

Collecting user data would be a valuable feature for medical science. It would enable the extraction of information regarding the user's sitting habits, which could be utilized in the realm of medical science for generating diverse reports and analyses.

References

[1] Eurostat, "Sit at work? You are one of 39%," 2019. [Online]. Available: https://bit.ly/45GobcQ

[2] N. F. Roberts, "Americans sit more than anytime in history and it's literally killing us," 2019. [Online]. Available: https://bit.ly/43DGE7U

[3] J. A. Wells, J. F. Zipp, P. T. Schuette, and J. McEleney, "Musculoskeletal disorders among letter carriers. a comparison of weight carrying, walking & sedentary occupations," *J Occup Med*, vol. 25, no. 11, pp. 814–820, Nov. 1983.

[4] M. Chia, B. Chen, and H. T. Suppiah, "Office sitting made less sedentary: A future-forward approach to reducing physical inactivity at work," 2015. [Online]. Available: https://repository.nie.edu.sg/handle/10497/17585

[5] J. P. Callaghan, D. D. Carvalho, K. Gallagher, T. Karakolis, and E. Nelson-Wong, "Is standing the solution to sedentary office work?" *Ergonomics in Design*, vol. 23, no. 3, pp. 20–24, 2015. [Online]. Available: https://doi.org/10.1177/1064804615585412

[6] S. K. Zeverdegani, M. Yazdi, and A. H. MollaAghaBabaee, "Latent class-derived patterns of musculoskeletal disorders in sedentary workers and chair ergonomic design," *Int J Occup Saf Ergon*, vol. 28, no. 3, pp. 1636–1641, May 2021.

[7] C. Bontrup, W. R. Taylor, M. Fliesser, R. Visscher, T. Green, P.-M. Wippert, and R. Zemp, "Low back pain and its relationship with sitting behaviour among sedentary office workers," *Applied Ergonomics*, vol. 81, p. 102894, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0003687019301279

[8] X. Zeng, "How to Try Bluetooth on the Raspberry Pi Pico W," 2023. [Online]. Available: https://bit.ly/3MN4aIU

[9] "Android developer documentation: Bluetooth permissions," 2023. [Online]. Available: https://bit.ly/45Hyygp

[10] G. Halfacree and B. Everard, *Get Started with MicroPython on Raspberry Pi Pico*. Raspberry Pi Press, 2021.

[11] GlobalStats, "Mobile Operating System Market Share Worldwide - July 2023," 2023. [Online]. Available: https://gs.statcounter.com/os-market-share/mobile/worldwide/#monthly-200901-202308