IoT Smart Shoe Solution for Neuromuscular Disease Monitoring

Davide La Rosa[1], Filippo Palumbo[1](✉), Alessandra Ronca[2], Francesco Sansone[3], Mario Tesconi[4], Alessandro Tonacci[3], and Raffaele Conte[5]

[1] Institute of Information Science and Technologies, National Council of Research (ISTI-CNR), Pisa, Italy
filippo.palumbo@isti.cnr.it
[2] Department of Information Engineering, University of Pisa, Pisa, Italy
[3] Institute of Clinical Physiology, National Research Council of Italy (IFC-CNR), Pisa, Italy
[4] Adatec S.r.l., Navacchio, Pisa, Italy
[5] National Research Council of Italy (CNR), Rome, Italy

Abstract. Recent advances in sensing, processing, and learning of physiological parameters, make the development of non-invasive health monitoring systems increasingly effective, especially in those situations that need particular attention to the usability of devices and software solutions due to the frailty of the target population. In this context, we developed a sensorized shoe that detects significant features in subjects' gait and monitors variations related to an intervention protocol in people affected by Neuromuscular Disorders (NMDs).

This paper outlines the challenges in the field and summarizes the approach used to overcome the technological barriers related to connectivity, deployment, and usability that are typical in a medical setting. The proposed solution adopts the new paradigm offered by Web Bluetooth based on Bluetooth WebSocket.

We show the architectural and deployment choices and how this solution can be easily adapted to different devices and scenarios.

Keywords: Web Bluetooth · Smart Shoe · IoT Health Device

1 Introduction

Neuromuscular Disorders (NMDs) include a wide range of health conditions affecting the function of muscular structures. They are related to the changes in the muscle or in the peripheral nerves sending signals to the muscles. Their diagnosis and clinical representation are often difficult even for skilled, experienced physicians. This is mainly due to the hundreds of specific NMDs present in the clinical experience, which are classified according to various principles. One of the most widely accepted ones distinguishes between: i) muscular dystrophies; ii) myopathies; iii) neuromuscular junction disorders; iv) motor/sensory neuropathies [1]. However, clinical hallmarks of such disorders are often similar, making their differential diagnosis troublesome. Also, the development of non-intrusive methods for their diagnosis and monitoring is an open challenge, making

A. Tsanas and A. Triantafyllidis (Eds.): PH 2022, LNICST 488, pp. 104–115, 2023.
https://doi.org/10.1007/978-3-031-34586-9_8

the need for easy-to-use and affordable devices and evaluation tests a key aspect of such a scenario.

To this end, we tried to merge the outcomes of two main projects, namely InGene 2.0 and Ki-Foot, both from the clinical and sensing perspective to offer a technological solution that, using a set of a few state-of-the-art short exercises wearing a sensorized shoe, can detect and monitor the evolution of one of the most widely studied conditions in people affected by NMDs, the Facioscapulohumeral muscular dystrophy (FSHD).

From a technological point of view, the shoe can detect significant features from the gait of the user primarily related to the FSHD condition. In this medical context, one of the main barriers to the adoption of technological solutions like this is the lack of connectivity and the reliability of the data collection. To this end, we developed a novel paradigm offered by the Bluetooth standard, namely Web Bluetooth[1], that allows us to connect to the smart shoe (or the data collection Bluetooth device) directly from the browser without any configuration by the medical personnel that can exploit the functionalities of the smart shoe in real-time while performing the exercises.

The chosen protocol is a specification for Bluetooth APIs to allow websites to communicate with devices in a secure and privacy-preserving way, it is still in a beta version and not completely adopted by all the browsers in their current versions, but its promising capabilities are worth the investigation in the eHealth context. For this reason, we show the chosen architectural and deployment solution in order to give a reference development guide to those interested in the implementation of this paradigm in their monitoring solutions.

1.1 The InGene 2.0 Project

Several attempts have been made to relate the clinical diagnosis of an individual, or a group of subjects, with NMD, to their genotype (i.e., their genetic background), to retrieve similarities, clustering, and differences, which might be a useful add-on to the current clinical practice. This is the main basis for the InGene 2.0 project, funded by Tuscany Region, Italy, under the Bando Salute 2018 call for grants, attempting at making use of technological (both hardware and software) tools to support the clinician in the diagnosis of NMD and the relationship retrieval between genotype and phenotype. The project, involving four clinical centers and two research institutions in the Region, aims at proposing this new paradigm of diagnosis and treatment, fruitfully supported by technology, to the regional and national decision-making, with likely positive outcomes in terms of a correct diagnosis and with a truly person-tailored treatment in such disorders.

Considering the most widely studied conditions in this field, Facioscapulohumeral muscular dystrophy (FSHD) attracts a lot of attention from clinicians for its relative incidence with respect to other NMDs, and for their clinical characteristics. In fact, FSHD is a disorder characterized by muscle weakness and wasting (atrophy). The disorder gets its name from muscles that are affected, namely those in the face (*facio*), around the shoulder blades (*scapulo*), and in the upper arms (humeral)[2]. What is particularly intriguing in FSHD is the presence of a peculiar muscular involvement, represented by

[1] https://www.w3.org/community/web-bluetooth/.

[2] https://rarediseases.org/rare-diseases/facioscapulohumeral-muscular-dystrophy/.

a relative weakness of the anterior muscles of the leg, mainly the tibialis anterior, which is of relevance according to the clinicians, also due to their somewhat relationship with the relative clinical severity of the disorder [2, 3]. Sometimes, such a muscular structure reflects minor changes related to the disease course even years before the onset of clear, related clinical signs, making it a useful target for tailored investigations. However, although efforts have been made to develop proper methods for the analysis of muscular involvement, which are usable, informative, and affordable, most studies still rely on the use of Magnetic Resonance Imaging (MRI) tools, which are expensive, somewhat obtrusive and not always well accepted by the patients [4, 5].

To this extent, new methodologies combining a fast, user-compliant, cost-affordable approach to study the tibialis anterior involvement in FSHD are desirable, making one of them the core of the investigation presented here. In particular, we chose a set of specific short exercises performed while wearing the smart shoes in order to detect and monitor such a condition: six minute walk test, ten meters walk, timed up and go, and four steps climbing.

1.2 A Smart Shoe for Gait Analysis

The core sensing device of the overall system is the smart shoes, which are sensorized footwear. Even if smart shoes, from an aesthetic point of view, they are not different from a normal pair of shoes, the upper side is made of leather while the sole belongs to the "Gommus" line, which is a rubber sole line for high performances, high quality, and high design products[3]. Inside the sole, different sensors and communication components[4] are present that allow an accurate analysis of different gait parameters. The Ki-Foot [6] shoe, based on the Motus prototype developed Carlos s.r.l., Fucecchio, Italy, has five pressure sensors integrated under the insole to monitor the mechanical interaction of the foot with the ground: three sensors under the forefoot, and the remaining two under the heel. In this way, almost complete coverage of the entire surface of the sole of the foot is ensured. The pressure sensors are custom-made piezo-resistive transducers produced by using a conductive material on a flexible substrate. These force sensors are sampled at 50 Hz. This kind of sensor has already been tested in different scenarios like sleep monitoring with smart bed slats [7]. A digital Inertial Measurement Unit (IMU) is integrated into the frontal part of the shoe. It consists of a 9-axis inertial platform with a 3D accelerometer, a 3D gyroscope, and a 3D magnetometer. Each value of the accelerometer represents the measure of acceleration of the corresponding axis, and it is measured in mg (milli-gravity). The gyroscope measures the angular speed for each corresponding axis and is expressed in dps (degrees per second). The magnetometer indicates the measurement of the earth's magnetic field for each axis and is expressed in mG (milli-Gauss). A Bluetooth Low Energy (BLE) transmission module is integrated with the rest of the electronic unit in the heel of the shoe to enable low-energy data transmission to a BLE device. The rechargeable battery type LIPO allows the complete operation of the system for 48 h. Being completely wireless controlled, the subject is not conditioned during movement and can move freely and independently for several days. Thanks to these sensors and to

[3] http://www.gommus.it.
[4] http://www.adatec.it.

these measures, there is a realistic analysis of the step and the characteristics of the two feet independently. We extract different features (Fig. 1) from the raw data collected to be used by clinicians in their gait analysis while performing the prescribed exercises. In the *Temporal* space, we extract step time, stride time, stance time, single limb stance time, double-limb stance time, swing time, swing-stance ratio, and cadence. The *Spatial* characteristics extracted are stride length, step length, and speed. The *Control* features are gait speed and stride regularity, while in the *Pressure* category falls the parameters related to Center of Pressure (CoP), mean pressure value, peak pressure value, and speed of CoP shift.

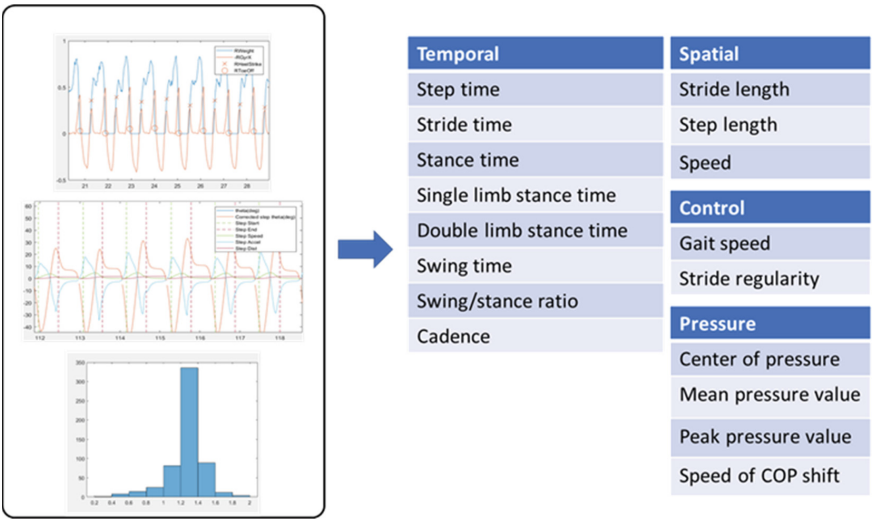


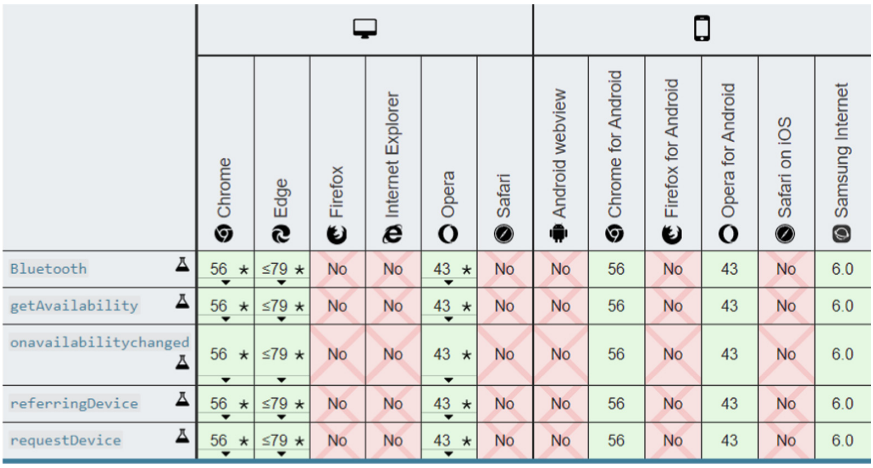
Fig. 1. From raw data to gait's temporal, spatial, control, and pressure related features

2 The Web Bluetooth Solution

Nowadays, browsers are evolving, bringing new APIs and ways to connect to other devices and allowing access to more functionality than they ever did before. One such API is the Web Bluetooth API[5]. This Web Bluetooth API is still in beta as of this writing, but once this gets released to the public, it will open a whole lot of opportunities for researchers and developers who want to use Bluetooth but don't have the possibility to create a native application for each platform.

The Web Bluetooth API is a low-level API allowing Web applications to pair with the nearby Bluetooth Low Energy-enabled peripheral devices and access their services exposed. Subsets of the Web Bluetooth API are available in some browsers as in Fig. 2. This means it is possible to request and connect to nearby Bluetooth Low Energy devices, read/write Bluetooth characteristics, receive GATT Notifications, know when a Bluetooth device gets disconnected, and even read and write to Bluetooth descriptors.

[5] https://www.chromestatus.com/feature/5264933985976320.



	Chrome	Edge	Firefox	Internet Explorer	Opera	Safari	Android webview	Chrome for Android	Firefox for Android	Opera for Android	Safari on iOS	Samsung Internet
Bluetooth	56 ★	≤79 ★	No	No	43 ★	No	No	56	No	43	No	6.0
getAvailability	56 ★	≤79 ★	No	No	43 ★	No	No	56	No	43	No	6.0
onavailabilitychanged	56 ★	≤79 ★	No	No	43 ★	No	No	56	No	43	No	6.0
referringDevice	56 ★	≤79 ★	No	No	43 ★	No	No	56	No	43	No	6.0
requestDevice	56 ★	≤79 ★	No	No	43 ★	No	No	56	No	43	No	6.0

Fig. 2. Compatibility table of the Web Bluetooth functionalities on various browsers and platforms

The *Generic Attribute Profile* (GATT) establishes in detail how to exchange all profile and user data over a BLE connection. In contrast with *Generic Access Profile* (GAP), which defines the low-level interactions with devices, GATT deals only with actual data transfer procedures and formats. GATT also provides the reference framework for all GATT-based profiles, which cover precise use cases and ensure interoperability between devices from different vendors. All standard BLE profiles are therefore based on GATT and must comply with it to operate correctly. This makes GATT a key section of the BLE specification because every single item of data relevant to applications and users must be formatted, packed, and sent according to its rules. GATT uses the *Attribute Protocol* as its transport protocol to exchange data between devices. This data is organized hierarchically in sections called services, which group conceptually related pieces of user data called *Characteristics*.

The GATT *Profile Hierarchy* describes how a GATT *Server* contains a hierarchy of *Profiles*, *Primary Services*, *Included Services*, *Characteristics*, and *Descriptors*.

Profiles are purely logical: the specification of a *Profile* describes the expected interactions between the other GATT entities the *Profile* contains, but it's impossible to query which *Profiles* a device supports.

GATT *Clients* can discover and interact with the *Services*, *Characteristics*, and *Descriptors* on a device using a set of GATT procedures. The specification refers to *Services*, *Characteristics*, and *Descriptors* collectively as *Attributes*. All *Attributes* have a type that's identified by a UUID. Each *Attribute* also has a 16-bit *Attribute Handle* that distinguishes it from other *Attributes* of the same type on the same GATT *Server*. *Attributes* are notionally ordered within their GATT *Server* by their *Attribute Handle*, but while platform interfaces provide attributes in some order, they do not guarantee that it's consistent with the *Attribute Handle* order.

A *Service* contains a collection of *Included Services* and *Characteristics*. The *Included Services* are references to other *Services*, and a single *Service* can be included by more than one other *Service*. *Services* are known as *Primary Services* if they appear

directly under the GATT *Server*, and *Secondary Services* if they're only included by other *Services*, but *Primary Services* can also be included. A *Characteristic* contains a value, which is an array of bytes, and a collection of *Descriptors*. Depending on the properties of the *Characteristic*, a GATT *Client* can read or write its value, or register to be notified when the value changes. Finally, a *Descriptor* contains a value (again an array of bytes) that describes or configures its *Characteristic*.

As with any other protocol or profile in the Bluetooth specification, GATT starts by defining the roles that interacting devices can adopt: *Client* or *Server*. The GATT *Client* corresponds to the ATT client using *Attribute Protocol*. It sends requests to a server and receives responses (and server-initiated updates) from it. The GATT client does not know anything in advance about the server's attributes, so it must first inquire about the presence and nature of those attributes by performing service discovery. After completing service discovery, it can then start reading and writing attributes found in the server, as well as receiving server-initiated updates. The GATT *Server* corresponds to the ATT server, which uses *Attribute Protocol* for the connection. It receives requests from a client and sends responses back. It also sends server-initiated updates when configured to do so, and it is the role responsible for storing and making the user data available to the client, organized in attributes. Every BLE device sold must include at least a basic GATT server that can respond to client requests, even if only to return an error response.

The Web Bluetooth API is exposed in the most updated browsers as a Javascript API:

`navigator.bluetooth.requestDevice(serviceFilters)`

Scans for the device in range supporting the requested services. Returns a Promise.

`device.gatt.connect()`

Returns a Promise resolved with the server object providing access to the services available on the device.

`server.getPrimaryService(name)`

Returns a Promise resolved with the particular Bluetooth service on the device.

`service.getCharacteristic(name)`

Returns a Promise resolved with the GATT characteristic object.

`characteristic.readValue()`

Returns a Promise resolved with a raw value from the GATT characteristic.

`characteristic.writeValue(value)`

Writes a new value for the GATT characteristic.

2.1 Web Bluetooth on the Field

Web Bluetooth is not yet a W3C standard but, besides the available implementations in Chrome platforms, roadmaps are available in Mozilla Firefox, Microsoft Edge, and WebKit (Safari)[6]. The API has gained attention in literature, and it has been implemented in various contexts. In [8], a push notification-based login method has been proposed, while in [9], authors present a method for rapid development of applications in distributed BLE IoT systems for eHealth and sports. In that work, a throughput comparison between a native and a Web Bluetooth solution has been presented and the conclusion was that,

[6] https://www.bluetooth.com/blog/the-web-bluetooth-series/.

despite the fact that at a higher transmission rate a native application outperforms the HTML5 Web Bluetooth application, the developed Web Bluetooth framework enables software and service operators to iteratively create, tune and deploy filter algorithms in distributed BLE IoT systems, without rebooting nodes or restarting programs using dynamic software updating. Also, a human centered Web-based dataset creation and annotation tool for real time motion detection has been developed [10] in which the user can effectively collect gestures from a nearby device that supports the BLE protocol, assign tags to the collected data, and store them remotely. A particular example of Web Bluetooth implementation in eHealth applications is presented in [11] that present the opportunities for Brain Computer Interaction (BCI) developers to stream data directly to a web browser.

3 The Proposed IoT Solution

Designing an IoT solution that can be used in clinical settings requires a careful evaluation of the technological aspects involved in such a context in terms of connection availability, easiness of deployment and maintenance, and usability by the medical staff or the caregivers. To analyze the possible scenarios and their implications, we considered three main entities that come into play:

- Web App: the application running within a browser on a mobile device, used by the clinicians to record the patients' data as well as handling the execution of the exercises proposed to the users
- BLE shoes: sensorized devices worn by the patients, able both to collect inertial and pressure readings and to transmit the data to a receiving device via a Bluetooth Low-Energy connection
- Logger: service in charge of recording the raw data generated by the sensorized shoes during an exercise and associating the recorded data to a specific user identifier for the subsequent processing phases

Through the Web Bluetooth API, any web application can interact with nearby Bluetooth devices in a secure and privacy-preserving way, without the need to deploy additional platform-specific apps. Depending on the environmental conditions and the features of the involved devices, we identified two reference scenarios in which the designed architecture can provide a feasible and effective solution.

3.1 Scenario 1: Web App as a Proxy

In this scenario, the Web App runs in a browser and acts as a bridge between the Bluetooth shoes and the remote logger service (Fig. 3). The shoes are directly paired with the mobile device and the Web App exploits the Web Bluetooth API provided by the browser to be able to connect to the shoes and receive the generated data in real-time. Subsequently, by employing a web socket connection to a backend device, the app sends the raw data collected from the patient exercise to the remote logger service.

The main interactions among the three involved entities are shown in the sequence diagram of Fig. 4. The Web App, as soon as the clinician has selected the patient and

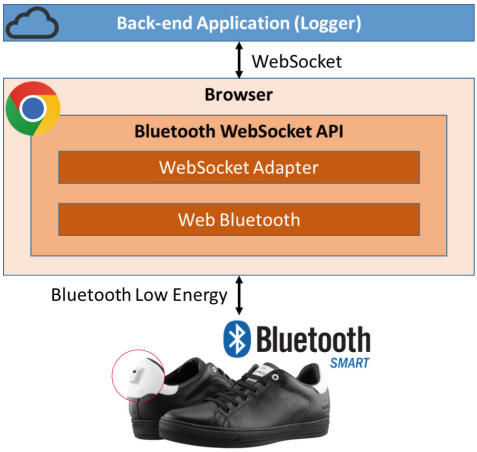


Fig. 3. The Web App collects data from BLE shoes and sends it to the remote logger

started the exercise, requests the available devices to the browser, which in turn triggers a discovery process to find the Bluetooth shoes by name. Whenever the shoes are detected, the Web App queries for both the available services and characteristics. Once the characteristic containing the sensors data is matched, the associated notification is enabled and, since then, the Web App starts to receive and locally store the data stream acquired from the shoes. This loop continues until the clinician stops the exercise from the Web App; then the connection to the shoes is closed and the locally cached recorded exercise is transferred to the remote logger service.

In this scenario the logger can be deployed on a remote backend, thus keeping the number of devices that needs to be deployed on-site at a minimum. This aspect is very important when the technical staff supporting the clinicians could not provide continuous or immediate assistance during the operational phase. On the contrary, we should note that the Bluetooth connection between the sensing devices and the smartphones or tablets can be impaired by potential technical limitations. In our case, adopting a pair of shoes transmitting at a data rate of 50 Hz each in environments where other transmitting devices are present, reduces the reliability of the communication, causing irregular data transmission and occasional connection losses. To improve the situation, we rearranged the system architecture to increase the performance of the Bluetooth data transmission, hence conceiving scenario 2 illustrated in the next section.

3.2 Scenario 2: Single-Board PC as a BLE Device

To overcome the technical limitations of the specific devices we used in our experimental settings, we decided to use two receiving Bluetooth antennas, one for each shoe, to collect data. We used two antennas on a small single-board PC, like for instance the Raspberry Pi, able to host the logger service as well. In this scenario, shown in Fig. 5, the shoes are directly paired with the logger device and the Web App acts as a controller to start and stop the data recording task. The logger device exposes itself as a Bluetooth device to be detected by the browser via the Web Bluetooth API.

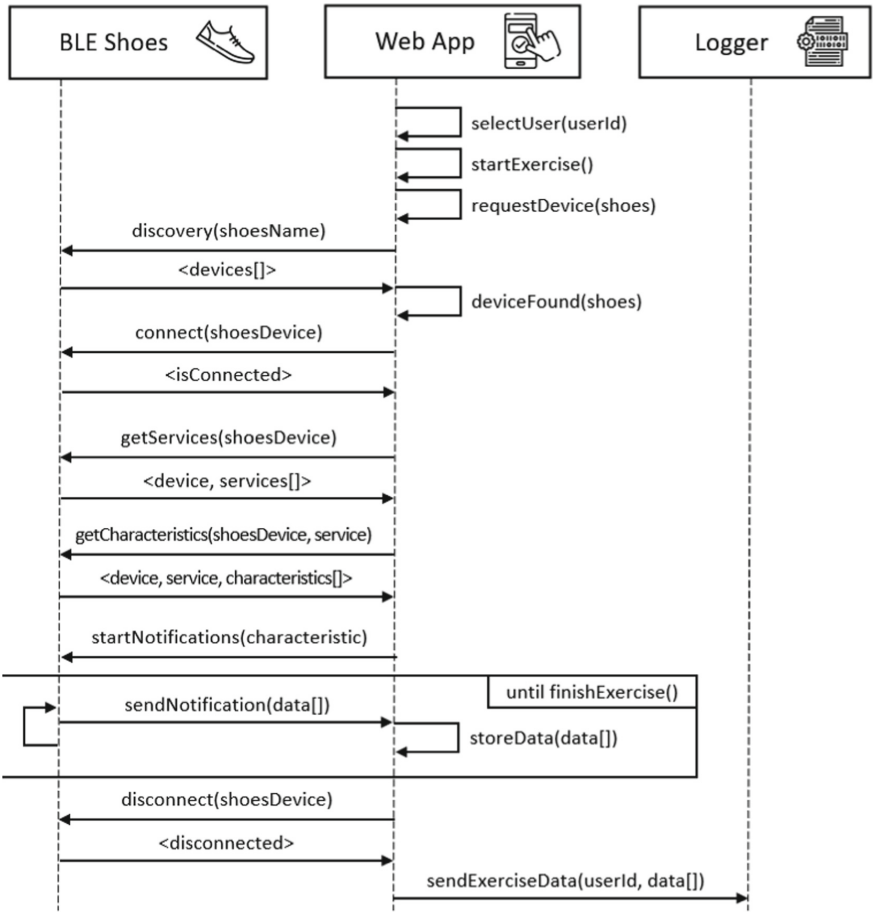


Fig. 4. Sequence diagram of the interaction among the devices in scenario 1

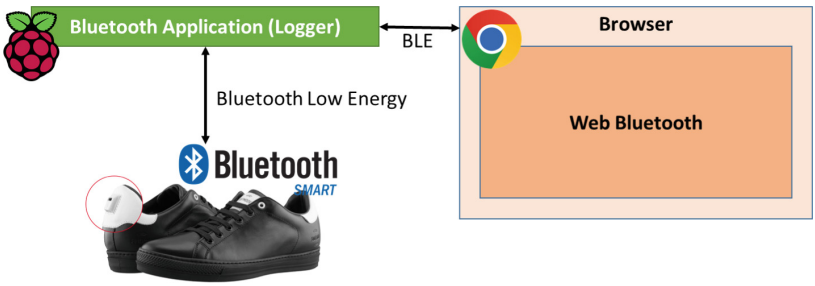


Fig. 5. The Web App connects to the logger service which, in turn, collects the data directly from the BLE shoes

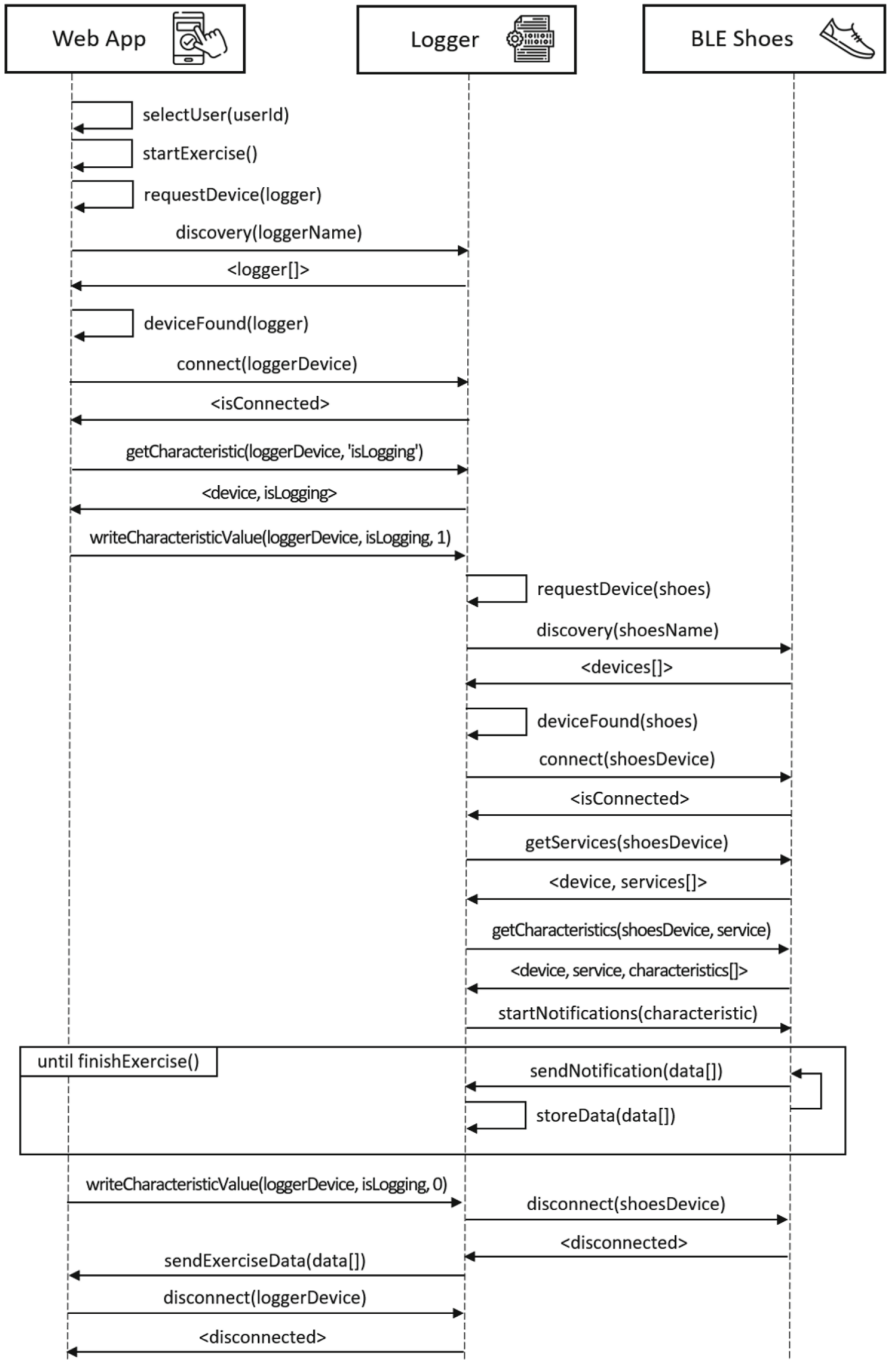


Fig. 6. Sequence diagram of the interaction among the devices in the scenario 2

The main components interactions taking place in this scenario are shown in Fig. 6. Once the clinician selects the user and starts the exercise, the Web App requests the discovery of the Bluetooth devices looking for a logger with a predefined name. If the logger is found, the Web App connects to it and searches for a specific characteristic used as a Boolean value to communicate whenever the logger must start or to stop the recording from the shoes. Writing the value '1' to this characteristic, triggers the logger into discovering the shoe devices, connecting to them, searching for the services and characteristics, and starting the notification to receive the sensors data. During the exercise, the acquired data is locally stored on the logger device. As soon as the clinician stops the exercise from the web interface, the Web App writes the value '0' to the logger characteristic causing its disconnection from the shoes and the transmission of the cached exercise data to the Web App. Eventually, when the data transfer in completed, the Web App disconnects from the logger device.

This configuration allows us to obtain the maximum performance in terms of connection reliability and transfer rate even in environments where several Bluetooth devices are present. Abstracting from the specific use case, this architecture is valuable whenever due to technical limitations or particular Bluetooth requirements, the connection between the Bluetooth equipment and the mobile device doesn't perform well and can indeed be improved by employing external or multiple antennas. On the downside, with respect to the scenario 1, the on-site installation of an additional device increases the deployment complexity and might require protracted technical support during the system operation. Adopting a local Bluetooth connection between the mobile device on which the Web App is running and the logger device, avoids the need to set up a communication channel based on the Wi-Fi connection. Since in clinical environments it might be difficult to obtain access to existing Wi-Fi networks for security reasons, this would require to either deploy a local access point or a hotspot. Instead, using the Bluetooth connectivity, the user experience of the healthcare staff operating the Web App is not burdened with troublesome configuration operations.

4 Conclusions

In this paper, we presented an IoT solution to help in the clinical diagnosis of subjects with NMD by using gait information from a BLE sensorized shoe. Although we focused on collecting the data from the sensorized shoes to analyze the human gait characteristics, the IoT technological solutions we envisaged can be applied to any kind of BLE device that exposes an accessible data interface. This is possible thanks to the capabilities offered by the novel Web Bluetooth API that is a candidate to become the standard de facto for connectivity between smart devices and web applications. The presented architectural choices can be easily modified and adopted by any developer in need for a seamless integrated solution for their reference domain.

References

1. Engel, W.K.: Classification of neuromuscular disorders. Birth Defects Orig. Artic. Ser. **7**(2), 18–37 (1971). PMID: 4950913

2. Olsen, D.B., Gideon, P., Jeppesen, T.D., et al.: Leg muscle involvement in facioscapulohumeral muscular dystrophy assessed by MRI. J. Neurol. **253**, 1437–1441 (2006)
3. Gijsbertse, K., Goselink, R., Lassche, S., et al.: Ultrasound imaging of muscle contraction of the tibialis anterior in patients with facioscapulohumeral dystrophy. Ultras. Med. Biol. **43**(11), 2537–2545 (2017)
4. Dorobek, M., Szmidt-Sałkowska, E., Rowińska-Marcińska, K., Gaweł, M., Hausmanowa-Petrusewicz, I.: Relationships between clinical data and quantitative EMG findings in facioscapulohumeral muscular dystrophy. Neurol. Neurochir. Pol. **47**(1), 8–17 (2013)
5. Veltsista, D., Chroni, E.: Ultrasound pattern of anterolateral leg muscles in facioscapulohumeral muscular dystrophy. Acta Neurol. Scand. **144**(2), 216–220 (2021)
6. Barsocchi, P., et al.: Detecting user's behavior shift with sensorized shoes and stigmergic perceptrons. In: 2019 IEEE 23rd International Symposium on Consumer Technologies (ISCT), pp. 265–268 (2019)
7. Barsocchi, P., Bianchini, M., Crivello, A., La Rosa, D., Palumbo, F., Scarselli, F.: An unobtrusive sleep monitoring system for the human sleep behaviour understanding. IEEE International Conference on Cognitive Infocommunications (CogInfoCom), pp. 91–96 (2016)
8. Varshney, G., Misra, M.: Push notification based login using BLE devices. In: 2017 2nd International conferences on Information Technology, Information Systems and Electrical Engineering (ICITISEE), pp. 479–484 (2017)
9. Wåhslén, J., Lindh, T.: A javascript web framework for rapid development of applications in IoT systems for eHealth. In: 2018 IEEE 20th International Conference on e-Health Networking, Applications and Services (Healthcom), pp. 1–6 (2018)
10. Bardoutsos, A., Markantonatos, D., Nikoletseas, S., Spirakis, P.G., Tzamalis, P.: A human-centered Web-based tool for the effective real-time motion data collection and annotation from BLE IoT devices. In: 2021 17th International Conference on Distributed Computing in Sensor Systems (DCOSS), pp. 380–389 (2021)
11. Stegman, P., Crawford, C.S., Andujar, M., Nijholt, A., Gilbert, J.E.: Brain–computer interface software: a review and discussion. IEEE Trans. Hum.-Mach. Syst. **50**(2), 101–115 (2020)