POLITECNICO DI TORINO

Collegio di Ingegneria Biomedica

Corso di Laurea Magistrale

in Ingegneria Biomedica

Tesi di Laurea Magistrale

Studio e sviluppo di un plantare sensorizzato per la valutazione delle distribuzioni delle pressioni plantari

Relatori

Prof. Ing. Marco Knaflitz

Prof. Ing. Manuel Jesus Domínguez Morales

Candidato

Luca Farci

Aprile 2019

Ringraziamenti

Ringrazio il Professor Manuel Domínguez Morales dell'Escuela Técnica Superior de Ingeniería Informatica de Sevilla per avermi offerto la possibilità di sviluppare un appassionante studio e per la disponibilità e precisione dimostratemi durante tutto il periodo di stesura.

Un ringraziamento particolare al Professor Marco Knaflitz per la Sua grande disponibilità.

Un affettuoso ringraziamento ai miei genitori, Fernando e Rita, grazie ai quali, con il loro instancabile sostegno sia morale sia economico, ho potuto portare a termine i miei studi.

Inoltre, ringrazio mia sorella Silvia e mia zia Michela per il loro sostegno nei momenti di difficoltà. Nonostante la lontananza ho sentito il loro costante appoggio che mi ha incoraggiato a raggiungere gli obiettivi prefissati.

Sommario

Il piede fornisce la superficie di contatto e d'interazione con l'ambiente esterno sia in posizione statica sia durante la locomozione pertanto è importante preservare la salute del piede e diagnosticare precocemente eventuali problemi. Patologie generali o specifiche possono alterare la sua fisiologica funzione.

Sulla pianta del piede si distribuisce il peso del corpo e il rilevamento di alterate pressioni, in statica e in dinamica, può prevenire o evitare la formazione e/o il peggioramento di lesioni del piede e in generale dell'arto inferiore, mediante la realizzazione e adozione di plantari personalizzati.

La pressione plantare è un campo pressorio che agisce tra il piede e il plantare, superficie di supporto, durante la normale attività motoria.

L'obbiettivo di questo lavoro di tesi è lo sviluppo di un sistema di misura delle pressioni plantari che identifica le aree di maggiore e minore carico per permettere la creazione di plantari personalizzati.

Il dispositivo progettato è stato concepito per essere agganciato alla caviglia del soggetto consentendo la massima comodità anche nel normale cammino.

Una serie di sensori resistivi posizionati nel plantare della scarpa e collegati al dispositivo agganciato alla caviglia rilevano le variazioni di pressione che si determinano in statica o nelle varie fasi del passo.

I dati delle misurazioni vengono trasmessi via Bluetooth al software correlato che mostra in tempo reale la pressione rilevata da ciascun sensore del plantare, salvando i dati acquisiti in un file.

L'elaborato è stato strutturato descrivendo nel primo capitolo l'anatomia del piede, ponendo anche l'attenzione sulle patologie che possono modificare la sua funzionalità.

Nel secondo capitolo è stato rappresentato lo stato dell'arte.

Nel terzo capitolo sono stati descritti i componenti scelti per la realizzazione del dispositivo e esposte le diverse fasi di sviluppo.

Nel quarto capitolo è stato esposta sia la modalità di utilizzo del dispositivo da parte del soggetto sia la procedura di visualizzazione dei dati acquisiti.

Il quinto capitolo sono state descritte le prove sperimentali in riferimento ai tre possibili gruppi di appoggio del piede.

Nel sesto capitolo è stata realizzata una possibile pianificazione temporale e finanziaria per la realizzazione del dispositivo, concludendo con i possibili miglioramenti che possono essere apportati al sistema e gli eventuali e ipotizzabili campi di utilizzo dello stesso.

Abstract

During the locomotion the foot is a surface of contact and a source of interaction with the external environment. Because of this, its health is important to monitor in order to precociously diagnose its problems.

General or specific pathologies can change its physiological function.

The measurements of pressure in both dynamic and static conditions can avoid, and even prevent, a worsening of a foot or inferior limb lesion. Plantar pressure exams are then essential to create personalized soles.

Foot plantar pressure is the pressure field that acts between the foot and the support surface during everyday locomotive activities.

In this thesis work, a plantar pressure measurement system was developed to identify areas with higher or lower pressure load to allow the creation of a personalized sole.

The designed device has been planned to be attached to the patient ankle allowing maximum comfort during the normal gait.

A series of resistive sensors is attached to the shoe sole, which is then connected to a device placed around the ankle, measures the pressure variations in static condition or during the gait cycle.

The measured data are transmitted via Bluetooth to the correlated software.

The software of the measurement system shows the pressure in each small sensor element of the sole in real time and saves the data in a file.

In the first chapter the foot anatomy is presented, with a focus on its pathologies that could modify its correct behaviour.

The second chapter gives a landscape of the current technologies used in this field.

After the introduction the third chapter lists the components used and the development phases employed.

The fourth chapter focuses on the testing phase of the system, while at the end a business plan is shown, with further improvement that can be added to the system.

Indice

Introduzione

La distribuzione e l'entità delle pressioni plantari forniscono importanti informazioni nell'ambito della salute e dello sport.

Patologie e conformazioni plantari anomale alterano spesso la funzionalità del piede con conseguenze su tutto l'apparato muscolare degli arti inferiori e osteoarticolare (ginocchia, bacino, schiena) e provocando disturbi nel mantenimento della stazione eretta e nel cammino.

In ambito sportivo, in particolare in quegli sport in cui l'atleta è costantemente in movimento (atletica, calcio, basket ecc.), l'analisi delle pressioni plantari in statica ma in particolare in dinamica, può risultare un fattore determinante nella preparazione di plantari personalizzati affinchè non si creino danni o lesioni alle strutture del piede e dell'arto inferiore in generale, causati da carichi abnormi e non fisiologici.

Un alterato appoggio nella statica e nella dinamica del passo dell'atleta può ripercuotersi sia nel suo rendimento, causando maggiore fatica e minore resa durante l'esercizio, sia nella formazione di lesioni locali al piede e più in generale all'arto inferiore e alla colonna.

In ambito sportivo l'analisi delle pressioni plantari è utile nella progettazione di scarpe e plantari su misura. Scarpe e plantari personalizzati riescono a compensare le alterazioni anatomiche e di carico in modo che, nella fase d'impatto del piede al suolo, il peso del corpo venga ripartito in maniera corretta sulla pianta del piede.

Attualmente, sono due i sistemi utilizzati per la misura di pressioni plantari: i sistemi su pedana/piattaforma e i sistemi integrati nelle calzature.

I sistemi su pedana, utilizzati per misure di pressioni plantari in statica e in dinamica, forniscono informazioni quantitative sul supporto plantare tramite il calcolo di parametri quali superficie, pressione massima, pressione media e centro di pressione.

A causa della loro complessità questi sistemi sono utilizzabili in ambienti controllati come in laboratorio o in ospedale e non per uso giornaliero.

Al contrario, i sistemi integrati nelle calzature possono essere utilizzati dal soggetto in qualsiasi ambiente e durante lo svolgimento di qualsivoglia attività senza causare alcun fastidio e interferire nel naturale cammino.

Capitolo 1

Il piede

1.1 Anatomía.

Il piede è la parte distale dell'arto inferiore che sorregge il peso del corpo e permette la locomozione.

Nel mantenimento della stazione eretta il piede collabora al controllo della delicata attività muscolare richiesta per conservare l'equilibrio.

Nel corso della deambulazione invece le funzioni del piede sono due:

- una funzione passiva, che prevede una prima fase di ammortizzamento dell'impatto cui il corpo umano è sottoposto durante il cammino e la corsa e
- una funzione attiva, che prevede il trasferimento al suolo delle forze interne generate dai muscoli durante la propulsione.

Lo scheletro del piede è formato da ventisei ossa riunite in tre parti:

- il tarso, un complesso osseo costituito da sette ossa brevi distinte in due file: la fila prossimale formata dal talo (o astragalo) e dal calcagno, quella distale formata dal navicolare, dai tre cuneiformi e dal cuboide,
- il metatarso, formato da cinque ossa lunghe che vengono numerate da uno a cinque in senso mediolaterale,
- le falangi, formano lo scheletro delle dita; sono in tre dal secondo al quinto dito: prossimale, media e distale, e sono due nel primo dito (alluce): prossimale e distale.

La struttura è completata da trentatré articolazioni (comprendenti le intertarsiche, tarsometatarsiche, intermetatarsiche, metatarsofalangee e interfalangeee) e da strutture muscolari intrinseche del piede, inserzioni di muscoli aventi origine nella gamba, tendini e legamenti.

La figura 1.1.1 riporta in dettaglio lo scheletro del piede e le ossa che lo costituiscono.

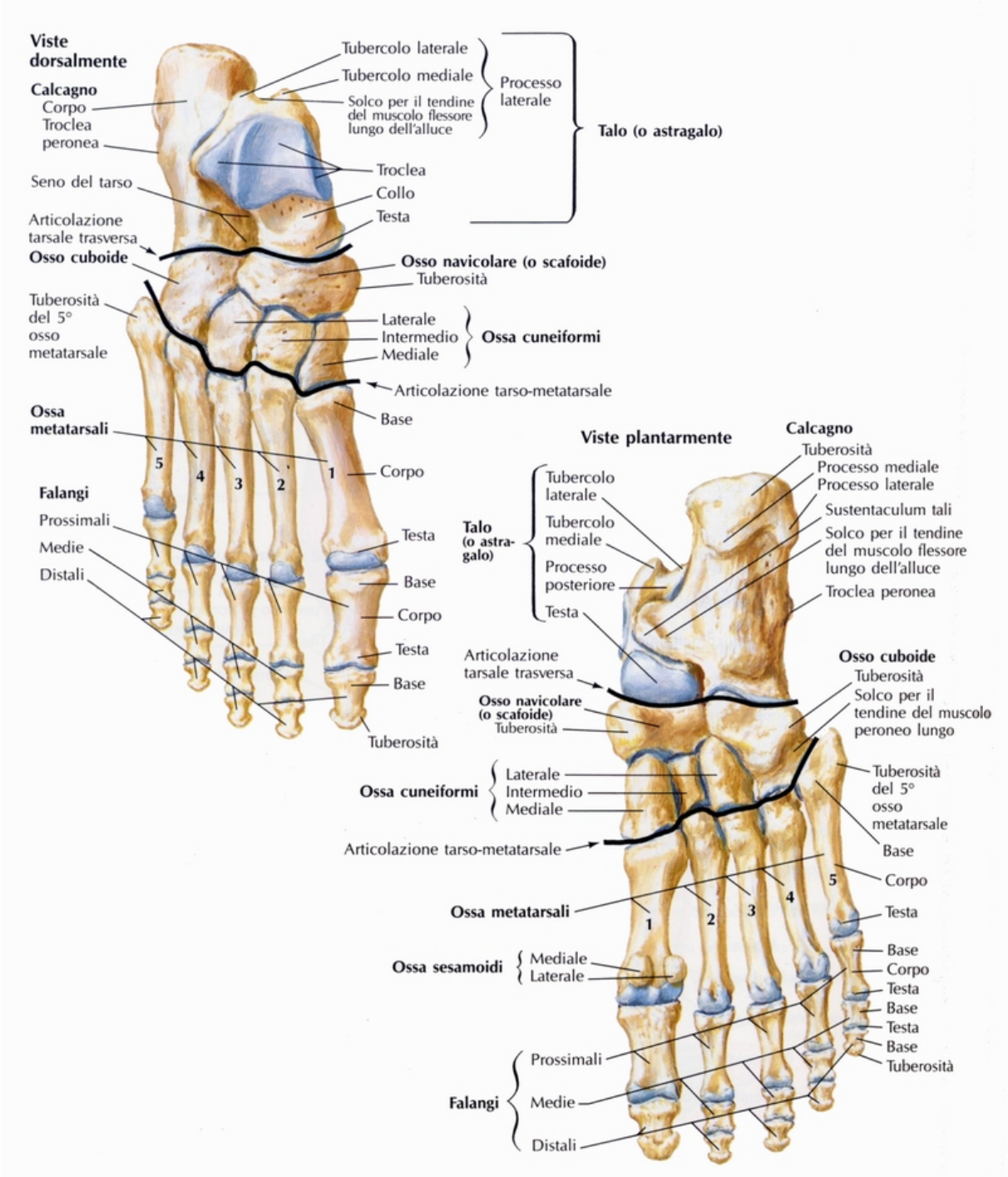


Figura 1.1.1 Scheletro del piede [1]

Dal punto di vista funzionale il piede può essere suddiviso in tre zone:

- il retropiede, composto dall' astragalo e calcagno o tallone. Le due ossa lunghe che compongono la gamba, tibia e perone, si connettono con la parte

superiore dell'astragalo attraverso l'articolazione tibio-tarsica. Questa parte realizza una funzione stabilizzatrice.

- il mesopiede, composto da cinque ossa irregolari: cuboide, naviculari e tre ossa cuneiformi, le quali costituiscono l'arco del piede, che serve come ammortizzatore. Il mesopiede è connesso con l'avampiede e il retropiede mediante muscoli e la fascia plantare.
- l'avampiede, composto dai cinque metatarsali che formano il metatarso e le falangi del piede. Ha una funzione dinamica [2].

La figura 1.1.2 riporta in dettaglio la suddivisione in dipendenza della sua funzionalità.

Figura 1.1.2 Divisione del piede in tre parti [3].

1.2 Archi e pressioni plantari

La pianta del piede non aderisce completamente al terreno ma forma una curva, più o meno accentuata, definita *volta plantare*.

Per la sua complessità e per il suo delicato equilibrio la struttura del piede è comparata in molti casi a quella di un arco architettonico (figura 1.2.1). Infatti, proprio come avviene negli archi architettonici, quelli plantari hanno la

funzione di trasformare le forze verticali in forze laterali con l'obbiettivo di ripartire in maniera armonica il peso del corpo sulla ridotta superfice del piede (pressione plantare).

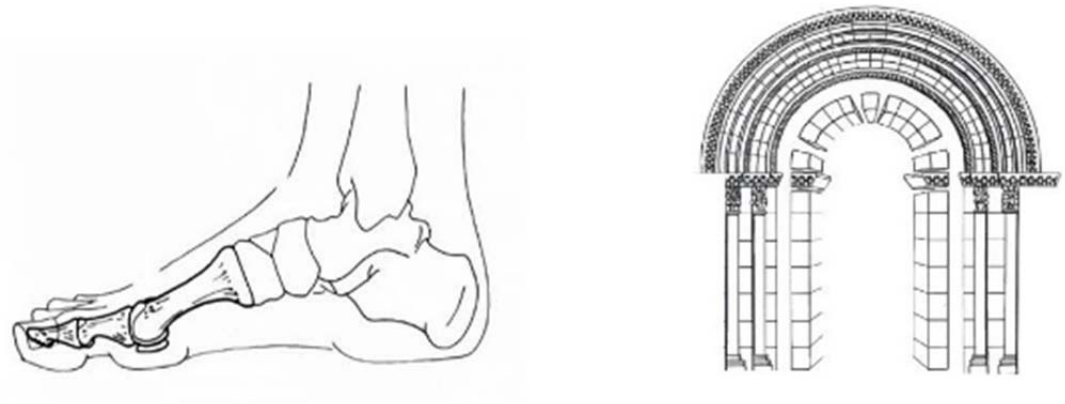


Figura 1.2.1 Comparazione tra l'arco architettonico romano e il piede

Gli archi plantari conferiscono una forma perfetta al piede, permettendogli di adattarsi a tutti i tipi di terreni e quindi di non far perdere l'equilibrio al corpo umano.

La pianta del piede presenta tre archi plantari principali: uno anteriore, uno esterno e uno interno; essi devono essere adeguatamente equilibrati per poter consentire un appoggio perfetto del piede sia durante la deambulazione sia in posizione statica eretta.

Gli archi plantari sono sottoposti a grandi forze e pressioni e il loro disequilibrio può portare all'alterazione della morfologia del piede.

Se ne possono individuare due e danno origine al:

- piede cavo: in cui l'arco di volta plantare è accentuato,
- piede piatto: in cui l'arco di volta plantare cade parzialmente o totalmente; si può formare per la debolezza dei suoi mezzi naturali di sostegno: i muscoli e legamenti; l'insufficienza di questi muscoli fa sì che il peso del corpo cada sopra l'arco plantare [4].

In conclusione, il disequilibrio delle forze e delle pressioni plantari potrà causare un'alterazione degli archi, influenzando l'anatomia del piede e di conseguenza la deambulazione.

Lo studio della distribuzione delle pressioni plantari in statica e in dinamica (in cammino e in corsa) è utile per valutare le proprietà dell'appoggio plantare e della camminata umana e consente di acquisire importanti informazioni per diagnosticare i differenti disturbi del piede.

Lo studio delle pressioni plantari può essere affrontato;

- su soggetti statici o in cammino (bassa frequenza) e quindi acquisendo informazioni sul mantenimento dell'equilibrio,
- su soggetti in corsa e quindi interessando il campo della biomeccanica.

1.3 Ciclo del passo

La deambulazione è data da una successione ciclica di movimenti ritmici alternati che viene chiamata *ciclo del passo* (gait cycle: periodo che intercorre tra due appoggi successivi dello stesso piede sul terreno).

Ogni ciclo del passo è diviso in due periodi:

1. fase d'appoggio (stance): identifica l'intero periodo durante il quale il piede è in contatto con il terreno,
2. fase d'oscillazione (swing): identifica il tempo in cui il piede si trova sollevato dal terreno per l'avanzamento dell'arto.

La figura 1.3.1. raffigura le due fasi, di appoggio e di oscillazione, durante il ciclo del passo a bassa frequenza ovvero camminando.

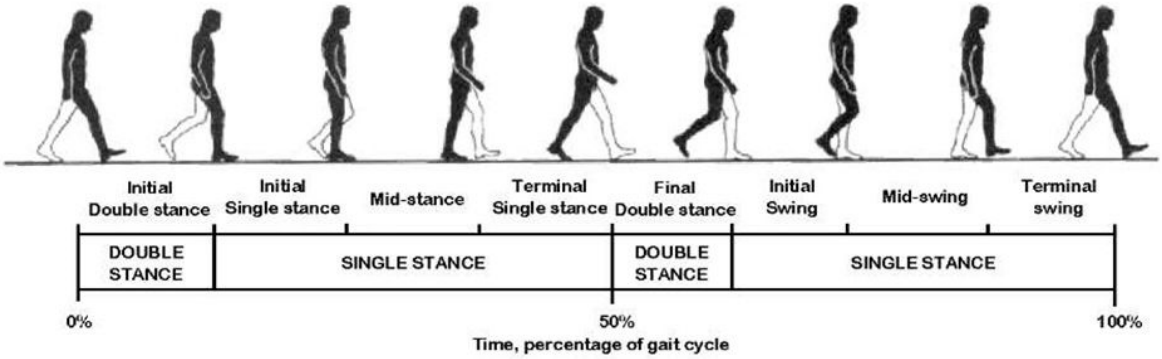


Figura 1.3.1 Ciclo del passo nel camino

Ognuna delle due fasi è poi suddivisa in altre sotto-fasi.

La fase d'appoggio a bassa frequenza può essere suddivisa in tre sotto-fasi in funzione della zona del piede interessata (figura 1.3.2):

1. **Contatto o initial contact:** è la prima sotto-fase durante la quale il piede è a contatto con il suolo. Si realizza con il tallone e sopporta la grande tensione nel momento dell'impatto.
2. **Appoggio o midstance:** è la seconda sotto-fase durante la quale il piede è completamente appoggiato al suolo e il peso della persona si riparte tra il tallone e la parte anteriore del piede mediante l'arco plantare. In questa fase si può osservare se il piede ruota verso l'interno, verso l'esterno o si mantiene in linea retta.
3. **Propulsione o push off:** è la terza e ultima sotto-fase durante la quale il piede presenta la parte anteriore a contatto con il suolo e che conferisce una spinta per iniziare il movimento [5].

Figura 1.3.2 Fase d'appoggio a bassa frequenza

Nella figura 1.3.3 sono raffigurate in sequenza le aree del piede interessate durante la fase d'appoggio: il tallone nella sotto-fase di contatto, l'intero piede nella seconda sotto-fase di appoggio, l'avampiede e le dita nella terza sotto-fase di propulsione.

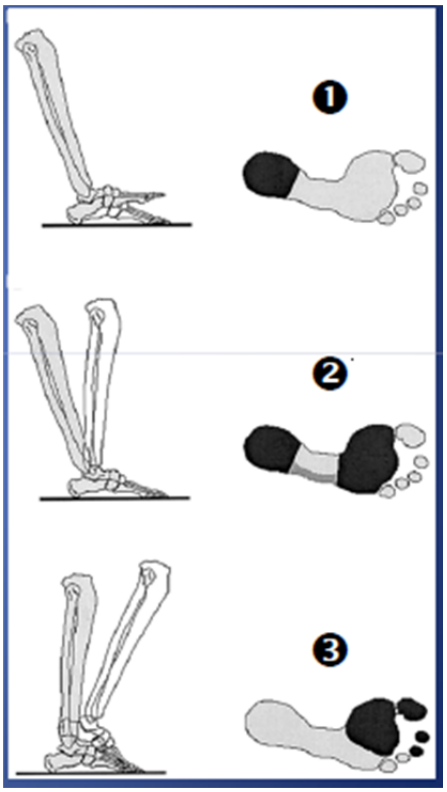


Figura 1.3.3. Aree del piede durante la fase d'appoggio

Nel ciclo di marcia durante il cammino, esistono periodi di doppio appoggio (ambo i piedi a contatto con il suolo): questo è il tratto caratteristico

che lo distingue dagli altri tipi di marcia. In questa situazione la fase di appoggio impegna più del 60% del ciclo del passo.

La fase di appoggio interessa anche cicli di marcia diversi dal cammino: la corsa e lo sprint, caratterizzati da diverse velocità rispetto al cammino e da differenti durate della fase di appoggio.

La corsa ha velocità intermedia tra cammino e sprint. Il limite tra il cammino e la corsa si avverte quando i periodi d'appoggio (i due piedi a contatto con il terreno in uno specifico istante) vengono sostituiti da periodi "fluttuanti" (nessun piede è in contatto con il terreno) al principio e alla fine del ciclo di marcia. In questa situazione la fase di appoggio non impegna più del 60% del ciclo del passo ma è la fase di oscillazione quella predominante.

La figura 1.3.4 descrive le fasi della corsa. In particolare, la descrizione riguarda la gamba e il piede destro: il ciclo di marcia incomincia e termina con il contatto iniziale (inizial contact IC) della parte mediale del piede destro. La Stance Phase è definita come il periodo di tempo tra il primo IC e il distacco dell'alluce dal suolo (toe-off, TO). La fase di oscillazione (Swing Phase) incomincia con il TO sino al successivo IC. [6]

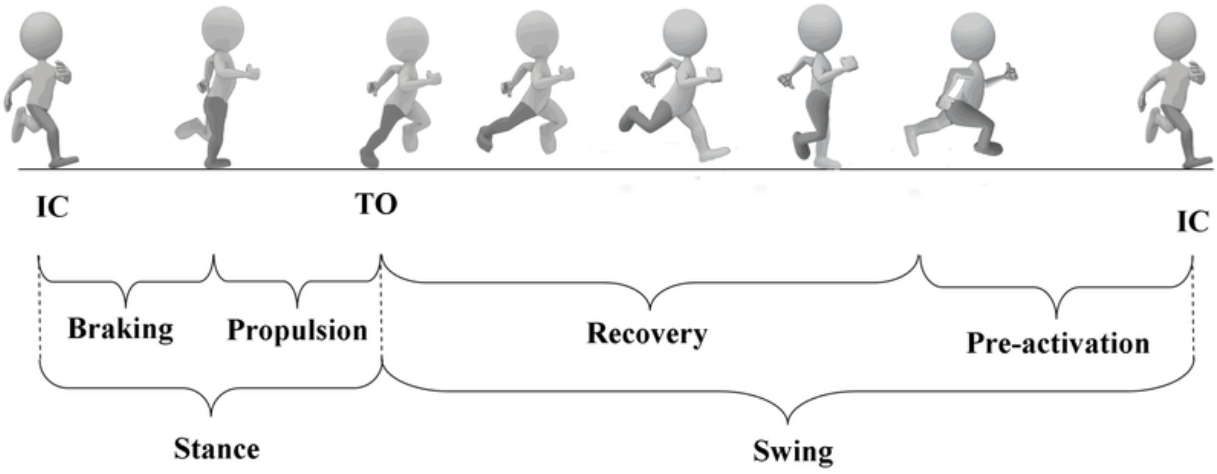


Figura 1.3.4. Fasi durante la corsa.

Lo sprint è caratterizzato dalla massima velocità.

In questo caso il contatto iniziale con il suolo si produce nella parte anteriore del piede e non nella parte mediale come avviene nella corsa.

Questo è dovuto a i differenti obiettivi dello sprint e della corsa. Lo sprint è adottato per percorrere una distanza nel minor tempo possibile, mentre nella corsa si percorrono lunghe distanze senza nessuna priorità di tempo.

La fase d'oscillazione può essere suddivisa in quattro sotto-fasi in funzione della posizione degli arti e della distanza tra il piede e il suolo:

- **Pre-oscillazione:** è la parte finale della fase d'appoggio e il suo obbiettivo è quello di preparare l'arto all'oscillazione, iniziando la flessione del ginocchio.
- **Oscillazione iniziale:** inizia con il distacco delle dita dal suolo e termina con l'oscillazione del piede sull'altra gamba che quindi diventa portante; in questa sotto-fase si interpone una adeguata distanza tra il piede e il suolo.
- **Oscillazione intermedia**: viene mantenuta la distanza piede-terreno mentre l'altra gamba è nell'ultima parte del medio appoggio.
- **Oscillazione terminale:** la gamba decelera e il piede si preposiziona correttamente per il contatto; questa sotto-fase inizia con la tibia verticale e termina quando il piede tocca il terreno. L'altra gamba è nel termine della fase di carico.

1.4 Tipi di appoggio

Sono stati individuati tre gruppi di appoggi del piede al suolo: prono, supino e neutro. Ogni essere umano ha uno specifico appoggio differente.

- **Appoggio pronatore:** la pronazione induce il piede ad inclinarsi verso la sua zona interna nel momento della fase di appoggio della deambulazione; è quindi la parte interna del piede quella che è sottoposta a maggiore tensione e sforzo, sopportando la maggior parte del peso del corpo. Quando l'inclinazione è superiore a quella che gli specialisti stimano normale si

verifica la cosiddetta *sovrapronazione*. Normalmente un soggetto con piede piatto presenta un appoggio pronatore (ma non tutte le persone con appoggio pronatore hanno un piede piatto e viceversa).

- **Appoggio supinatore:** la maggior parte del peso della persona si riparte nella zona esterna del piede, sottoponendola a maggiore tensione e sforzo e causando la sua inclinazione verso l'esterno. Una persona con appoggio supino è solito avere un piede cavo (ma non tutte le persone con un appoggio supino hanno un piede cavo e viceversa).

- **Appoggio neutro o normale o universale:** l'appoggio è nella parte centrale del piede e pertanto il peso del corpo si ripartisce uniformemente in tutta la pianta.

Una persona dotata di appoggio neutro nella fase di appoggio del ciclo di marcia disegna una virtuale linea retta.

La figura 1.4.1 riporta i tre grandi gruppi di appoggio.

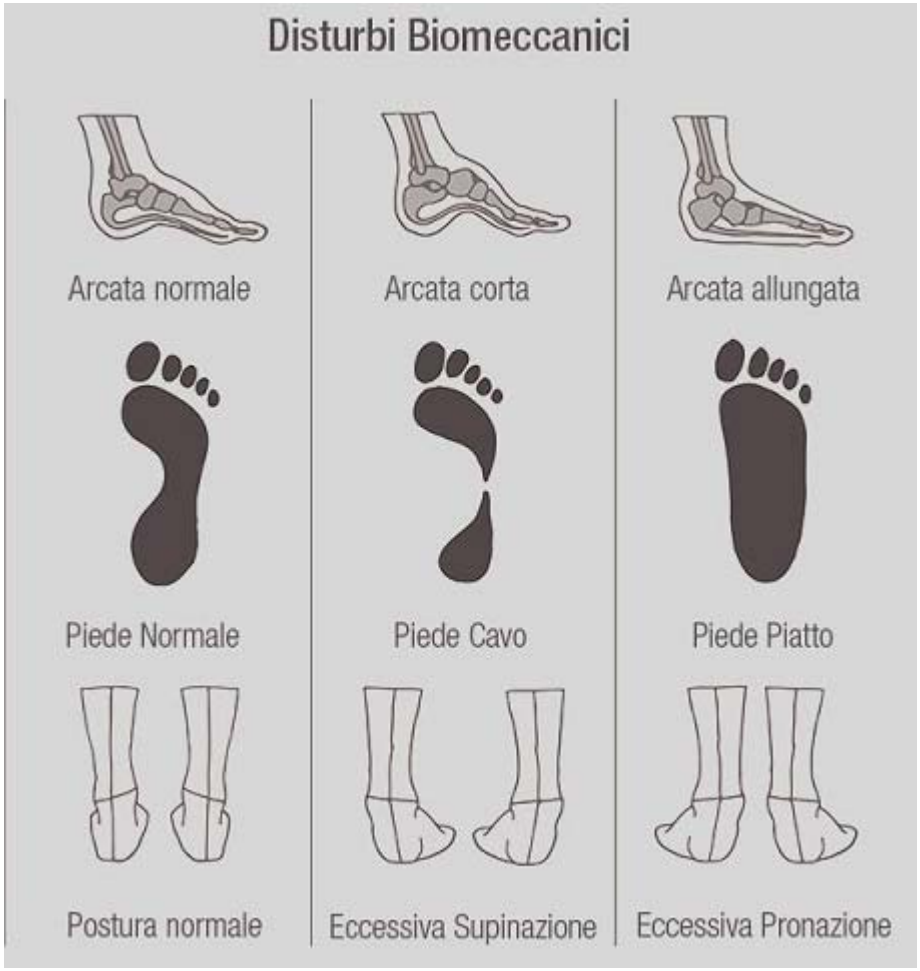


Figura 1.4.1. Differenti tipi d'appoggio. [7].

1.5 Patologie.

Patologie e conformazioni plantari anomale alterano la funzionalità del piede, provocando problemi sia nel mantenimento della posizione eretta sia nella deambulazione.

Le alterazioni dei piedi possono essere espressione di problemi sistemici come: diabete, patologie cardiocircolatorie, patologie reumatiche e osteoarticolari o manifestarsi localmente nelle strutture osteoarticolari con dolore nella deambulazione: artrite reumatoide, alluce valgo ecc.

Nei successivi paragrafi saranno descritti alcuni di questi problemi.

1.5.1 Diabete mellito.

Il diabete mellito è una patologia cronica caratterizzata da una carenza assoluta o relativa della secrezione di insulina. Il mancato compenso glicometabolico nel tempo può determinare tra le sue complicanze croniche il "piede diabetico".

Il piede diabetico è un piede con alterazioni anatomo – funzionali determinate dalla neuropatia diabetica e dall'arteriopatia obliterante periferica. Elevati livelli di glucosio nel sangue con il tempo determinano alterazioni dei nervi periferici e dei vasi sanguigni determinando una riduzione o scomparsa della sensibilità nel piede (neuropatia) e una ridotta irrorazione dei tessuti (arteriopatia). La perdita di sensibilità non permette di avvertire la formazione di lesioni quali tagli, bolle, ecc. che possono progredire in ulcere infette e che nei casi più gravi possono portare a amputazioni minori (parti del piede) o maggiori (piede, gamba e coscia), specie se associate ad arteriopatia obliterante [8].

1.5.2 Arteriopatia crónica agli arti inferiori.

L'arteriopatia cronica agli arti inferiori o PVD (peripheral vascular disease), è un'ostruzione parziale o totale delle arterie che irrorano gli arti inferiori: i vasi periferici sono più o meno ostruiti per la formazione di placche aterosclerotiche. La riduzione del flusso sanguigno determina una ridotta ossigenazione dei muscoli e dei tessuti che, per la comparsa di una sintomatologia dolorosa, determinano una riduzione via via crescente dell'autonomia di marcia sino alla comparsa di dolore a riposo.

1.5.3 Artrite reumatoide

L'artrite reumatoide è una patologia infiammatoria cronica di origine autoimmune che tra le varie sedi interessa anche il piede e la caviglia provocando dolore, deformità e difficoltà nel camminare.

Il piede reumatico si manifesta nell'89% dei pazienti e nella maggior parte dei casi interessa l'avampiede. È caratterizzato da rigidità dell'arto con riduzione dei movimenti, da forti dolori e da irritazioni cutanee.

Deve essere trattato sin dalla comparsa dei primi sintomi per evitare che degeneri rapidamente in forme avanzate di difficile trattamento.

Benché sia un disturbo degenerativo, è possibile prevenirne gli effetti utilizzando calzature adatte che consentano al piede di muoversi liberamente e di camminare mantenendo una posizione il più naturale possibile. Oltre ad idonee calzature, è importante indossare plantari confezionati su misura e utilizzare tutori e strumenti adatti a correggere la struttura alterata del piede [9].

1.5.4 Problemi ariticolari.

Un cattivo appoggio del piede o la non corretta biomeccanica del cammino può produrre dolori e lesioni a piedi, caviglie, bacino e colonna vertebrale.

- **Caviglia**: permette la flesso-estensione del piede sia in statica sia in dinamica. Portare scarpe strette o tacchi alti può causare una costante flessione plantare nel piede e la destabilizzazione della caviglia, con incremento di tensioni nei muscoli e nei legamenti durante il movimento naturale del cammino, aumentando il rischio di distorsioni

Inoltre, si produce un accorciamento di muscoli e tendini della parte posteriore del polpaccio, riducendo l'elasticità di questi muscoli.

- **Ginocchio**: permette il movimento di flesso estensione con l'aiuto del quadricipite. La posizione alterata del piede aumenta l'angolo di flesso estensione del ginocchio provocando un aumento della tensione muscolare del quadricipite, inoltre la muscolatura ischio tibiale si riduce e perde di funzionalità. La flessione eccessiva con il tempo aumenta il rischio di artrosi dell'articolazione del ginocchio con usura della rotula.
- **Pelvi:** è la parte più distale del tronco e lo connette agli arti inferiori. Qualunque disequilibrio nella parte superiore o inferiore del corpo si

ripercuote sulla pelvi, limitando il suo bilanciamento e movimento. La posizione del piede sul suolo incide sulla pelvi determinando un aumento dell'inclinazione anteriore della stessa e quindi un aumento della tensione della muscolatura superiore e inferiore del tronco, dei muscoli stabilizzatori dell'anca e della muscolatura del pavimento pelvico. I flessori dell'anca si accorciano incidendo sulla funzionalità del quadricipite e della parte inferiore della pelvi.

- **Colonna lombare:** il mantenimento della curvatura naturale di questa parte della colonna è molto importante per il movimento.

L'uso di calzature non convenzionali aumenta la curvatura della colonna lombare determinando dolore in regione lombare per compressione e/o spostamento e allineamento delle vertebre. Questo produce una ridistribuzione del peso addominale, traslandolo in avanti e aumentando il rischio di una ernia discale a livello lombare [10].

La figura 1.5.4.1. schematizza i problemi articolari legati ad un non corretto appoggio del piede.

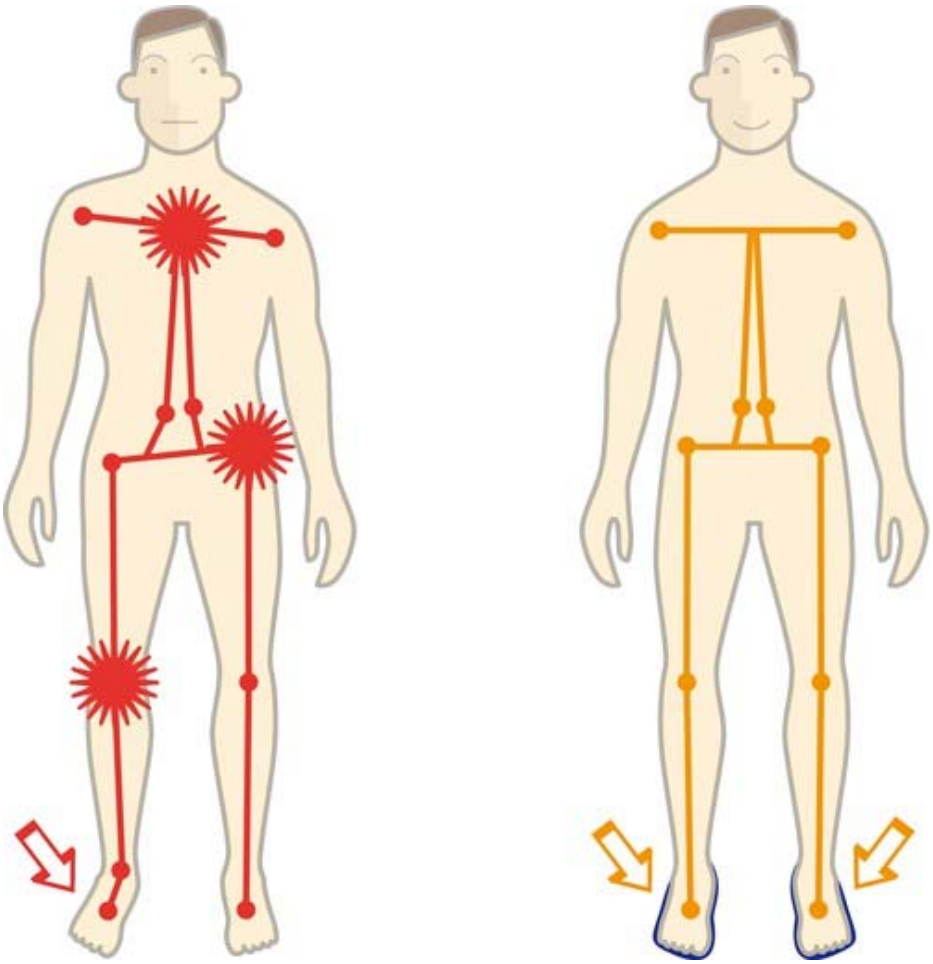


Figura 1.5.4.1 Problemi articolari legati a un cattivo appoggio del piede

Capitolo 2

Stato dell'arte

2.1 Pressioni plantari

Si possono individuare due tipi di sistemi di misura di pressioni plantari:

- sistemi su pedana e
- sistemi integrati nelle calzature.

2.2 Sistemi su pedana

I sistemi su pedana sono costruiti su una lastra in cui sono posizionati i sensori di pressione e sono integrati nel pavimento per permettere il normale cammino.

Consentono gli studi in dinamica e in statica però generalmente ristretti all'osservazione in laboratorio.

Il sistema su pedana è di facile utilizzo poiché la piattaforma di appoggio è statica e piana ma il soggetto necessita di tempo per abituarsi alla normale locomozione.

Altre limitazioni sono lo spazio, le misure solo indoor e l'abilità del paziente di creare un contatto con la piattaforma.

La figura 2.2.1 raffigura il soggetto in movimento in un sistema su pedana

Figura 2.2.1 Sistemi su pedana [11]

2.3 Sistemi integrati nelle calzature

Negli ultimi anni è in crescita l'interesse sullo sviluppo di sistemi per la detenzione di pressioni plantari in dinamica con sistemi integrati nel plantare della scarpa o nella calza mediante sistemi wireless o con cavi per permettere un esame baropodométrico preciso.

La baropodometría è una tecnica che permette la misura punto per punto di pressioni esercitate del piede nel suolo, del soggetto in posizione eretta (baropodometría statica) o durante la camminata (baropodometría dinamica).

I sistemi devono garantire la misura della pressione plantare in tempo reale e in maniera precisa.

Uno dei lavori più completi in questo settore è quello portato a termine da Bamberg *et al.* [12] dell'Istituto Tecnologico del Massachusetts (MIT). In questo sistema i sensori includono tre accelerometri ortogonali, tre giroscopi ortogonali, quattro sensori di forza, due sensori di flessione bidirezionale, due sensori di pressione dinamica e sensori d'intensità del campo elettrico.

I suddetti dispositivi, raffigurati nella figura 2.3.1, sono stati capaci di rilevare l'appoggio del tallone, stimare l'orientamento e la posizione del piede e lo staccarsi delle dita dal suolo. Il microcontrollore, un modulo RF monolitico, un'antenna e la fonte d'alimentazione sono incorporati nella scarpa.

Figura 2.3.1 Dispositivo creato da Bamberg *et al.* [12]

Nel 2009, Benocci *et al.* [13] dell'Università di Bologna hanno sviluppato un sistema senza fili per realizzare analisi posturali e in marcia. La figura 2.3.2 raffigura il sistema.

Il sistema consta di 24 idrocellule per misurare la pressione plantare, e dell'unità di misurazione inerziale (IMU: Inertial Measurement Unit) in

ciascuna plantare. L'IMU si compone di un accelerometro a tre assi e di un giroscopio digitale a tre assi. Per controllare il sistema si usa un microcontrollore della Texas Instrument (MPS430) e un modulo Bluetooth come recettore. I dati captati dai sensori permettono all'utente di determinare le differenti fasi della marcia, come l'oscillazione e l'appoggio, e la durata del singolo e doppio appoggio.

Figura 2.3.2 Dispositivo realizzato da Benocci *et al.* [13]

Shu *et al.* [14] hanno realizzato un sistema di misura e analisi di pressioni plantari basato su un array di sensori in tessuto tessile conduttore, leggero e con elevata sensibilità.

I sensori sono stati inseriti in sei posizioni differenti del plantare, come mostrato nella parte sinistra della Figura 2.3.3.

Il microcontrollore PIC18F452 e il modulo Bluetooth sono stati agganciati alla caviglia del paziente, come mostrato nella parte destra della Figura 2.3.3.

Figura 2.3.3 Sistema di misura e analisi di pressioni plantari di Shu *et al.*

Il sistema può interagire con un computer o con uno smartphone con la possibilità di calcolare la pressione media, il picco di pressione, il coefficiente di prestazione (COP: coefficient of performance) e il cambio di velocità del COP. Può essere usato in statica e in dinamica.

La figura 2.3.4 è un esempio di interfaccia grafica del sistema di Shu *et al.* [14].

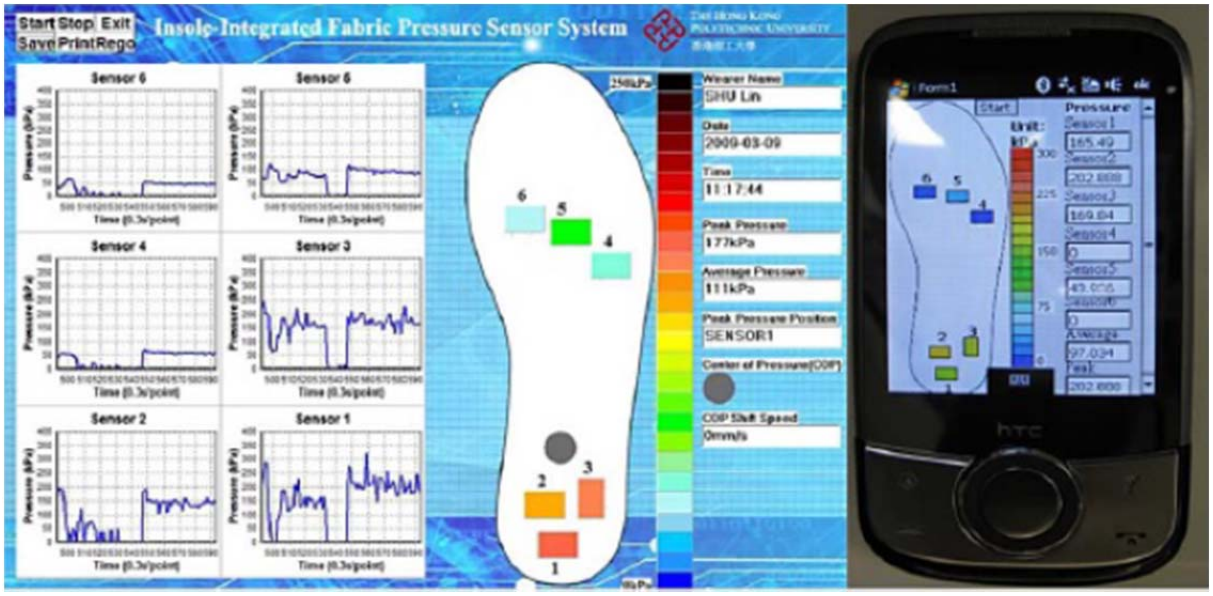


Figura 2.3.4 Esempio di interfaccia grafica con computer e smartphone del sistema di Shu *et al.* [14].

I principali vantaggi di questo sistema sono il basso costo e la facilità d'uso: l'utente può utilizzarlo nelle attività giornaliere per valutare quanta energia consuma durante un esercizio, l'abilità di coordinazione delle gambe durante la corsa o il controllo dell'equilibrio durante una specifica attività.

Da SensorMedica [15] è stato inoltre sviluppato il sistema chiamato FlexinFit (Figura 2.3.5).

Il sistema contiene più di 400 sensori resistivi integrati nelle suole (214 ciascuna) con una risoluzione a 10 bit, quindi consente di effettuare analisi accurate all'interno della scarpa del soggetto. È dotato di una batteria Li-Po con una durata di 4 ore, ricaricabile via USB, e di un modulo Bluetooth che permette una trasmissione dei dati in un raggio di 100 metri in campo aperto.

Il costo totale di questo dispositivo, comprensivo di software per l'analisi biomeccanica e della postura mediante una camera HD, è di 10.500€.

Figura 2.3.5 Dispositivo FlexinFit

Capitolo 3

Progettazione e sviluppo del sistema

3.1 Scelta dei componenti

I componenti che compongono il sistema sono:

- microcontrollore: Arduino Nano,
- elemento di captazione: in questo caso sensori di forza resistivi (in numero di sei-otto),
- circuito di condizionamento: con il tipo di sensori scelti è sufficiente un partitore resistivo,
- elemento per l'invio dei dati: modulo Bluetooth,
- alimentazione: batteria ricaricabile.
- sistema di visualizzazione: un computer e/o uno smartphone,

3.1.1 Microcontrollore

L'elemento essenziale del dispositivo è la scheda Arduino Nano della famiglia Arduino.

Le schede Arduino sono impiegate in molteplici progetti, in tutto il mondo, da hobbisti e professionisti per la loro facilità d'utilizzo.

Le schede Arduino sono programmabili in un linguaggio derivato dal C/C++, molto semplice, in un IDE (Integrated Development Environment) scaricabile gratuitamente dal sito ufficiale di Arduino. Inoltre, sono presenti numerosissime librerie online che ne possono incrementare le funzionalità e la possibilità di utilizzo in diversi contesti.

La versione Nano di Arduino ha le stesse funzioni di una board classica, ma è pensata per un uso specifico nei progetti; ciò è dovuto soprattutto alle sue dimensioni (43.18 mm × 18.54 mm), peso (7g) e costo ridotto (22$).

La scheda è basata su un microcontrollore a 8-bit AVR prodotto da Atmel: l'ATmega328.

L'Arduino Nano ha le seguenti caratteristiche principali:

- supporto mini-USB,
- digital pin: sono14 (D0-D13) e possono generare una tensione di 0V (Low) o 5V (High) con una corrente massima erogata di 40mA,
- external interrupts D2 e D3,
- comunicazione Seriale: pin 0 (RX) e 1 (TX) per, rispettivamente, ricevere e trasmettere dati,
- analog input: 8 (A0-A7) con una risoluzione a 10 bits (1024 valori differenti),
- consumo di corrente: 19mA,
- frequenza di clock: 16MHz,
- alimentazione:
    - Pin Vin: pin per alimentare Arduino Nano con una fonte di alimentazione esterna non regolata tra 6V e 12V,
    - Pin 5V: tensione di alimentazione regolata a 5V,
    - Tramite cavo USB collegato ad un PC [16].

La figura 3.1.1 raffigura scheda Arduino Nano.

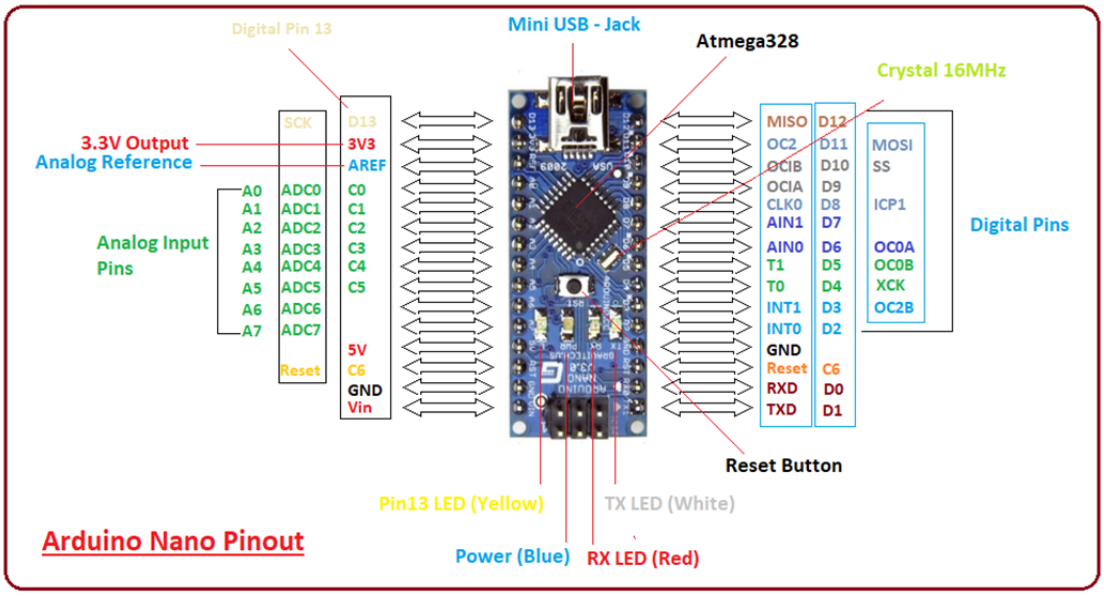


Figura 3.1.1 Piedinatura dell'Arduino Nano. Modificata da [17]

3.1.2 Sensori di forza resistivi

Nella molteplicità di sensori resistivi è presente il sensore di forza resistivo (FSR: Force-Sensing Resistor) che converte la pressione applicata in variazione di resistenza elettrica, rappresentato nella figura in 3.1.2.

Questo tipo di sensori possono recepire forze applicate in un rango di 100g - 10kg.

Figura 3.1.2 Sensore di forza resistivo [18]

Il FSR è un sensore di pellicola di polimero composto da un substrato semiconduttore, un substrato con elettrodi e un adesivo che li separa, come rappresentato in figura 3.1.3.

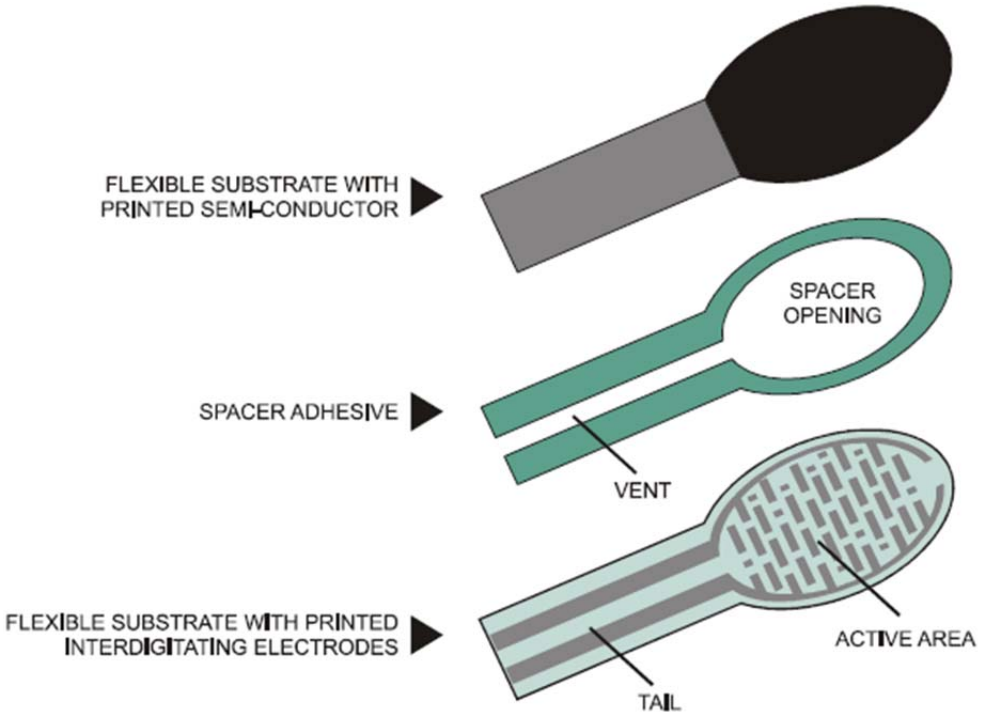


Figura 3.1.3 Struttura del FSR [19]

La sua resistenza diminuisce con l'aumentare della forza applicata sulla superfice attiva, parte che capta la forza. Quando non viene esercita alcuna forza sopra il FSR la sua resistenza sarà maggiore di 1MΩ.

Per il nostro obbiettivo è importante considerare le dimensioni del sensore visto che verrà posto su un plantare quindi su una superfice abbastanza ridotta:

- Lunghezza totale: 60,3 mm.
- Larghezza: 19,1 mm.
- Diametro sensore: 12,7 mm
- Spessore: 0,46 mm.

Il suo prezzo è di circa: 4,51$ cada uno [18].

3.1.3 Circuito di condizionamento

La maggior parte dei sensori ha bisogno di un circuito di condizionamento, più o meno complesso. Lo scopo di un circuito di condizionamento è una leggera traduzione del dato ricevuto ad una grandezza che sia facilmente assimilabile dal sistema di interpretazione.

Questo dipenderà da molti aspetti, come l'ampiezza dell'uscita del sensore (può essere di alcuni volt, di millivolt, ecc.), il tipo di uscita (può essere una tensione, in corrente, una resistività variabile, ecc.), la frequenza d'uscita, la risposta all'ingresso ecc.

Nel nostro caso una delle configurazioni elettriche base che si utilizzano con il FSR è il partitore di voltaggio, come rappresentato in figura 3.1.4. Nel circuito, la resistenza di misura RM è stata scelta per massimizzare la sensibilità della forza e del limite di corrente.

In questo caso non sarà necessario un amplificatore operazionale visto che il segnale all'uscita del circuito ha un grande range di valori e non necessità essere amplificato.

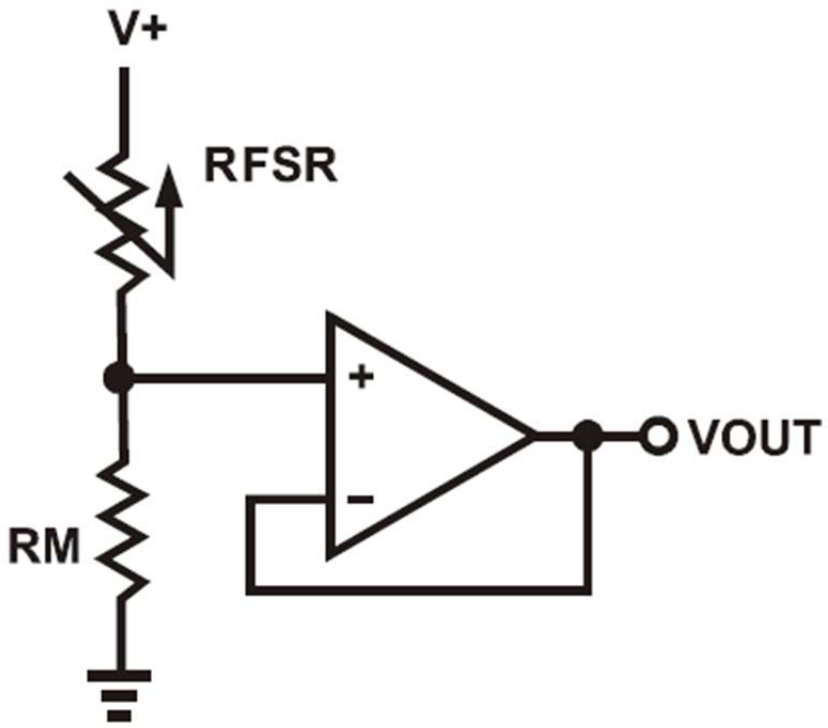


Figura 3.1.4 Circuito di condizionamento del FSR [19]

Nella figura 3.1.4 il divisore di voltaggio del circuito con FSR è il VOUT che corrisponde alla tensione d'uscita, *R*FRS è la resistenza del sensore, *R*M è la resistenza di misura e $V+$ la tensione d'ingresso. Le grandezze sono legate dalla seguente relazione:

$$V_{OUT} = \frac{V_+}{[1 + \frac{R_{FSR}}{R_M}]}$$

In cui:
Vout: tensione d'uscita [V].
V_+: tensione d'ingresso [V].
R_{FSR}: resistenza del sensore [Ω].
R_M: resistenza di misura [Ω].

3.1.4 Elemento per l'invio dei dati: Modulo bluetooth

Per ottenere un sistema senza fili abbiamo bisogno di un modulo Bluetooth.

Allo scopo è stato scelto un modulo HC-05 (rappresentato in figura 3.5) per il suo costo 2,56$ [20] e per le sue dimensioni: 28mm x 15 mm x 2.35mm.

La caratteristica più importante per il dispositivo oggetto del progetto è che il suo consumo di corrente in sleep mode è minore di 1mA e in modalità attiva e pari a 40mA.

Questo dispositivo Bluetooth necessità una tensione d'alimentazione compresa tra 3,3V e 6V.

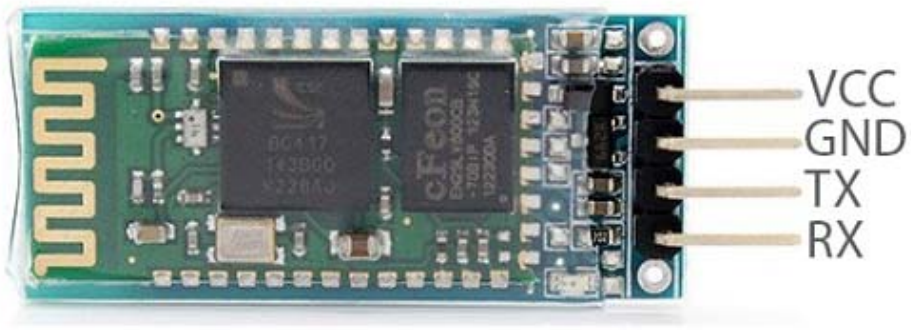


Figura 3.5. Modulo HC-05

3.1.5 Alimentazione del circuito

Il sistema sarà wireless, senza cavi, e pertanto è necessaria una fonte di alimentazione portatile come una batteria.

La scelta della batteria dipenderà dal consumo di corrente dell'intero dispositivo e dal livello di tensione necessario per ciascun componente.

I componenti che consumano energia sono: l'Arduino Nano, il modulo Bluetooth e il led.

I valori della corrente massima assorbita da questi componenti, considerando i dati indicati nei paragrafi precedenti sono:

- Arduino nano: 19mA,
- HC-05: 40mA,
- Led: 20mA.

E quindi un valore per la corrente totale (Itot) di circa 80mA.

Secondo quanto detto nel paragrafo 3.1.1 l'Arduino Nano può essere alimentato con una fonte tra 6V e 12 V attraverso il pin Vin e pertanto potremo utilizzare:

- quattro batterie alcaline ricaricabili AA di 1,5V cada una, poste in serie, con una tensione totale di 6V e una capacità di 3000mAh.

Il prezzo trovato in internet è 6,43$ per le quattro batterie [21], a cui deve essere addizionato quello dei connettori (rappresentati in figura 3.1.6) che permettono la connessione dell'Arduino Nano alle batterie e che è di circa 1,55$ (cada uno).

Il costo totale sarà di 9,53$.

La durata Δt delle batterie sarà data dalla relazione:

$$\Delta t = \frac{Q}{\text{Itot}} = \frac{3000\ mAh}{80\ mA} = 37{,}5\ ore = 1\ giorno\ e\ 13\ ore\ 30\ minuti$$

in cui:

Q: capacità delle quattro batterie [mAh],

Itot: corrente totale [mA]

Figura 3.1.6 Batterie alcaline AA con rispettivo connettore [21] [22]

oppure potremo utilizzare:

- una batteria alcalina da 9V con capacità di 1200mAh; la batteria pur avendo un voltaggio sufficiente per alimentare l'Arduino, presenta una capacità ridotta. Il suo prezzo in internet è di circa 8,76$ [23] e considerando il prezzo del connettore di circa 0,96$ [24] (Figura 3.1.7), si avrà un costo totale di circa 9,72$. La figura 3.1.7 rappresenta la soluzione con un'unica batteria.

La durata Δt della batteria sarà data dalla relazione:

$$\Delta t = \frac{Q}{Itot} = \frac{1200}{80} = 15\ ore$$

in cui:

Q: capacità totale della batteria [mAh],

Itot: corrente totale [mA]

Figura 3.1.7. Batteria alcalina da 9V con rispettivo connettore [23] [24]

Una soluzione alternativa alle precedenti è quella di utilizzare una power bank (rappresentata in figura 3.1.8) connessa direttamente alla porta mini-USB dell'Arduino Nano con capacità di 2200 mAh, DC output: 5V/1A, dimensioni di 97mm x 26mm x 23mm e peso netto de 106g.

Il prezzo di questa batteria è di circa 17,06$ [25].

La durata Δt della power bank sarà data dalla relazione:

$$\Delta t = \frac{Q}{Itot} = \frac{2200}{80} = 27.5\ ore = \ 1\ giorno, 3\ ore\ 30\ minuti$$

in cui:

Q: capacità totale della batteria [mAh],

Itot: corrente totale [mA]

Figura 3.1.8 Power Bank

Confrontando le tre configurazioni di alimentazione dell'Arduino, l'utilizzo di un'unica batteria consente di occupare meno spazio viste le dimensioni ma ha lo svantaggio di avere una capacità inferiore alla metà di ciascuna delle altre soluzioni e questo può rappresentare un problema in un dispositivo wireless. Inoltre, nelle altre soluzioni le batterie sono ricaricabili e quindi potranno essere riutilizzate più volte.

Se ora si confrontano gli altri due tipi di alimentazione rimanenti, si potrà senza dubbio scegliere quella con l'utilizzo della power bank poiché di più ridotte dimensioni, con maggiore facilità di connessione con il dispositivo (senza saldature) e la rapidità di ricarica.

Altro vantaggio è che il led integrato inizia a lampeggiare quando la batteria si sta scaricando cosi che non è necessario un controllo del livello di tensione nel codice.

3.1.6 Sistema di visualizzazione

Il sistema di visualizzazione è un elemento necessario per mostrare i dati captati dai sensori.

Sono stati scelti due sistemi: il computer e lo smartphone o tablet.

Il calcolatore elettronico o computer è un dispositivo capace di ricevere e processare dati per convertirli in informazione utile.

Permette una comunicazione con i sistemi d'ingresso e uscita attraverso una connessione USB o Bluetooth.

Un computer standard suole essere composto da un'unità di elaborazione centrale (CPU: Central Processing Unit processore centrale) e alcuni dispositivi hardware di interazione con l'utente quali tastiera, schermo e mouse.

Questo lo converte nella migliore opzione perché permette una comunicazione con l'utente in maniera facile, permettendo di creare un'interfaccia semplice e intuitiva.

La seconda opzione, o soluzione complementare, è quella di utilizzare uno smartphone o un tablet. Come il dispositivo precedente è di uso comune tra gli utenti ma consente all'utente di interfacciarsi con il sistema direttamente sullo schermo con le dita. Con una semplice connessione Bluetooth e sviluppando un'app si avrebbe a disposizione un oggetto utilizzabile ogni giorno in maniera semplice.

La figura 3.1.9 raffigura mediante un diagramma a blocchi il sistema

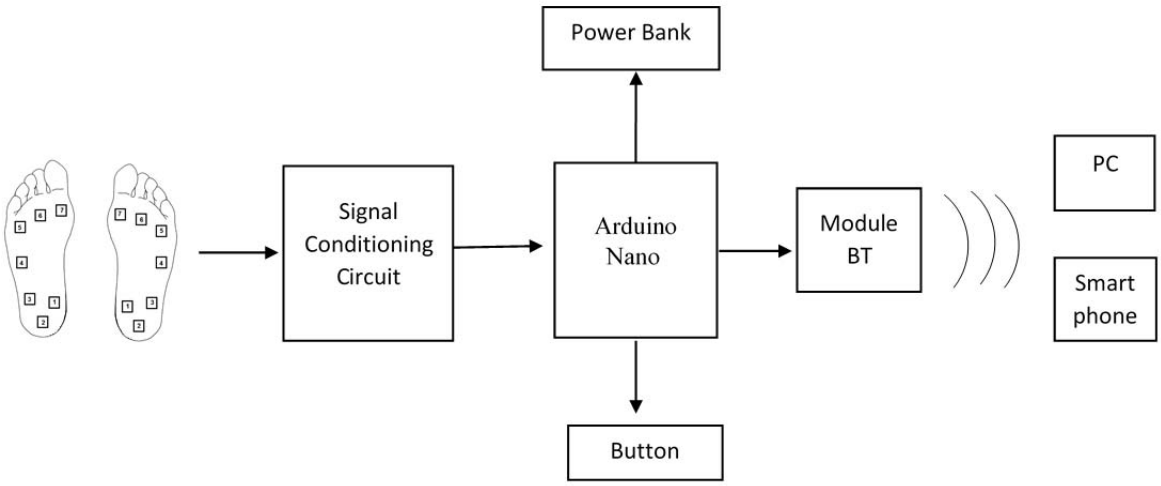


Figura 3.1.9 Diagramma blocchi del sistema

3.2 Pianificazione temporale

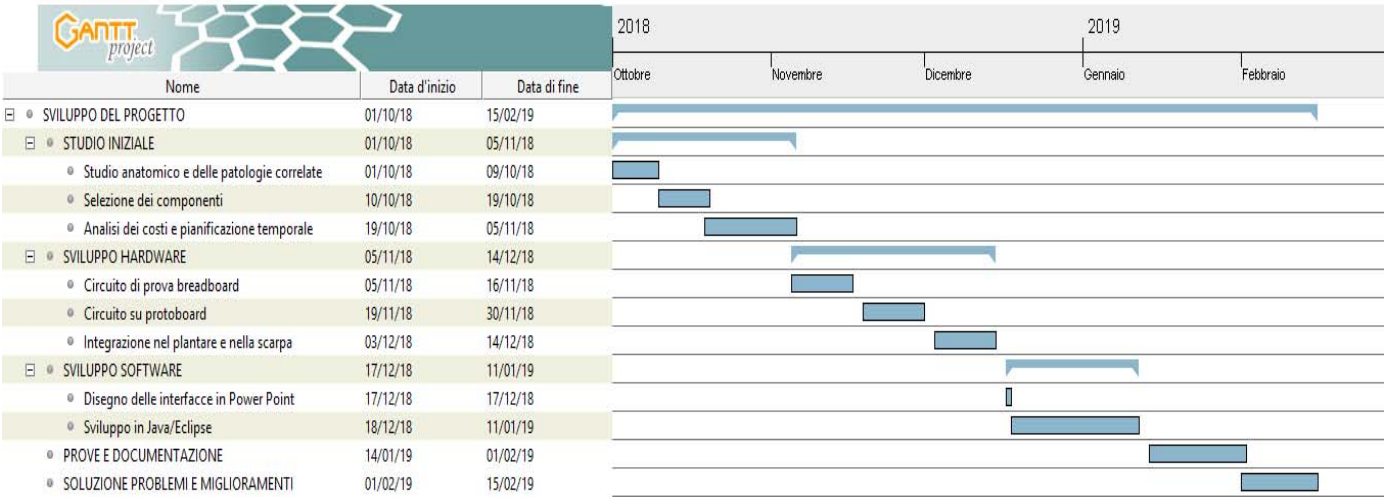
In tutti i progetti è fondamentale pianificare la durata delle diverse parti che lo compongono e la loro sequenzialità. Uno strumento efficace allo scopo è il diagramma di Gantt, costruito partendo da un asse orizzontale che rappresenta l'arco temporale totale del progetto, suddiviso in fasi incrementali (ad esempio, giorni, settimane, mesi), e da un asse verticale che rappresenta le varie attività che costituiscono il progetto stesso.

Delle barre orizzontali di lunghezza variabile rappresentano le sequenze, la durata e l'arco temporale di ogni singola attività del progetto (l'insieme di tutte le attività del progetto ne costituisce la work breakdown structure). Queste barre possono sovrapporsi durante il medesimo arco temporale ad indicare la possibilità dello svolgimento *in parallelo* di alcune delle attività. Man mano che il progetto progredisce, delle barre secondarie, delle frecce o delle barre colorate possono essere aggiunte al diagramma, per indicare le attività sottostanti completate o una porzione completata di queste. Una linea verticale è utilizzata per indicare la data di riferimento.

Ad ogni attività possono essere associate una o più risorse. Alcuni software disponibili sul mercato permettono di visualizzare il carico di lavoro di ogni risorsa e la sua saturazione, impostando una certa disponibilità per ogni risorsa. Contestualmente, può essere definito il calendario dei giorni lavorativi e festivi, e il numero di ore di lavoro giornaliere [26].

Un diagramma di Gantt permette dunque la rappresentazione grafica di un calendario di attività, utile al fine di pianificare, coordinare e tracciare specifiche attività nel progetto dando una chiara illustrazione dello stato d'avanzamento del progetto rappresentato e individua le possibili deviazioni nel progetto [27].

La Figura 6.2.1 riporta il diagramma di Gantt relativo al suddetto progetto in cui volutamente si è incrementato il tempo di realizzazione complessivo per tenere conto di alcuni imprevisti che potrebbero concretizzarsi come il ritardo nella consegna dei componenti.



Nome	Data d'inizio	Data di fine
SVILUPPO DEL PROGETTO	01/10/18	15/02/19
STUDIO INIZIALE	01/10/18	05/11/18
Studio anatomico e delle patologie correlate	01/10/18	09/10/18
Selezione dei componenti	10/10/18	19/10/18
Analisi dei costi e pianificazione temporale	19/10/18	05/11/18
SVILUPPO HARDWARE	05/11/18	14/12/18
Circuito di prova breadboard	05/11/18	16/11/18
Circuito su protoboard	19/11/18	30/11/18
Integrazione nel plantare e nella scarpa	03/12/18	14/12/18
SVILUPPO SOFTWARE	17/12/18	11/01/19
Disegno delle interfacce in Power Point	17/12/18	17/12/18
Sviluppo in Java/Eclipse	18/12/18	11/01/19
PROVE E DOCUMENTAZIONE	14/01/19	01/02/19
SOLUZIONE PROBLEMI E MIGLIORAMENTI	01/02/19	15/02/19

Figura 3.2.1 Diagramma di Gantt

3.3 Circuito di prova su Breadboard

I componenti descritti nel paragrafo precedente sono utilizzati per realizzare il prototipo del circuito su breadboard.

La figura 3.3.1 riproduce lo schema del prototipo del circuito, realizzato con Fritzing, programma (open-source) per eccellenza utilizzato per la realizzazione di schemi elettrici in progetti con Arduino.

Il circuito consta di:

- un Arduino Nano: l'alimentazione è attuata con un cavo USB (non mostrato in Figura 3.3.1) connesso direttamente alla power bank;
- un modulo Bluetooth HC-05, connesso con i pin D4 e D5 dell'Arduino Nano;
- un pulsante connesso al pin D3 dell'Arduino Nano
- un led rosso,
- un sensore che, come circuito di condizionamento, ha una resistenza uguale a 100 KΩ.

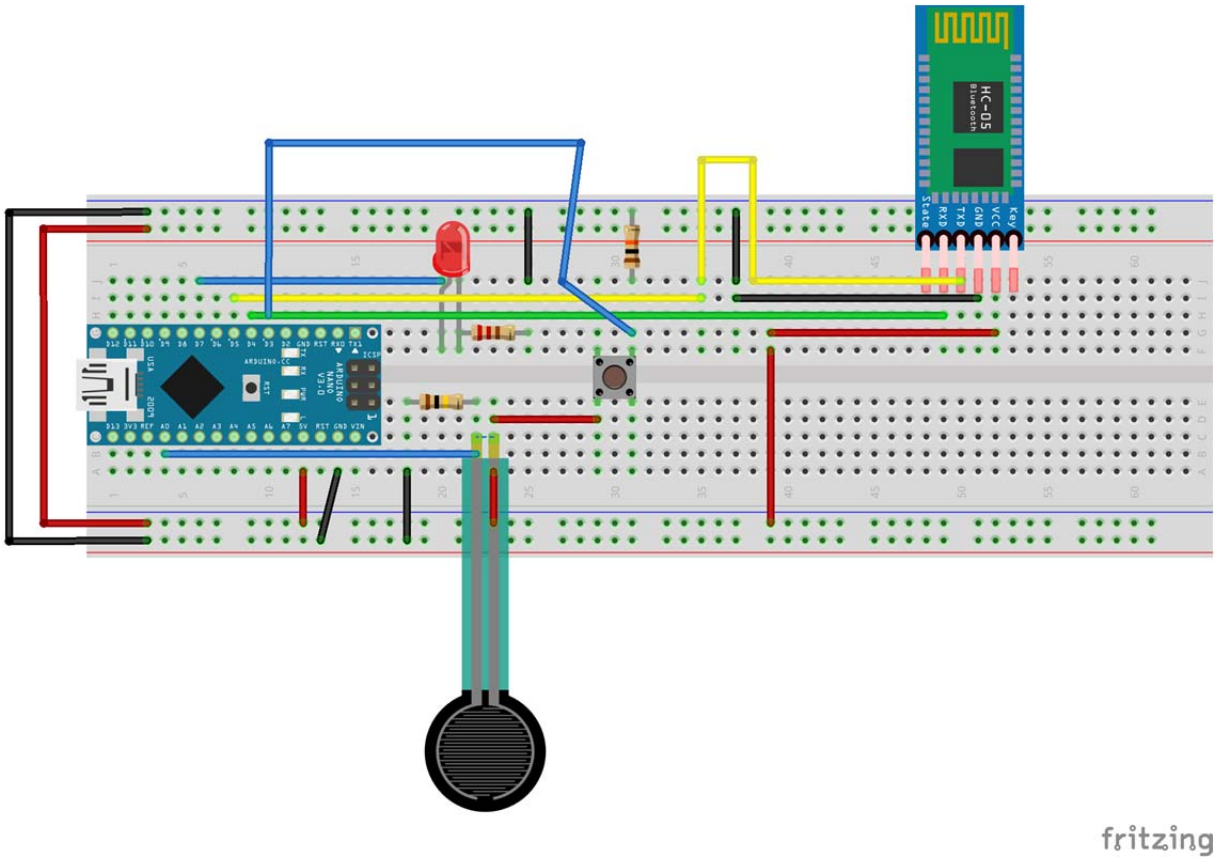


Figura 3.3.1 Schema del circuito di prova. Realizzato con Fritzing

La figura 3.3.2 rappresenta lo schema elettrico del circuito di prova.

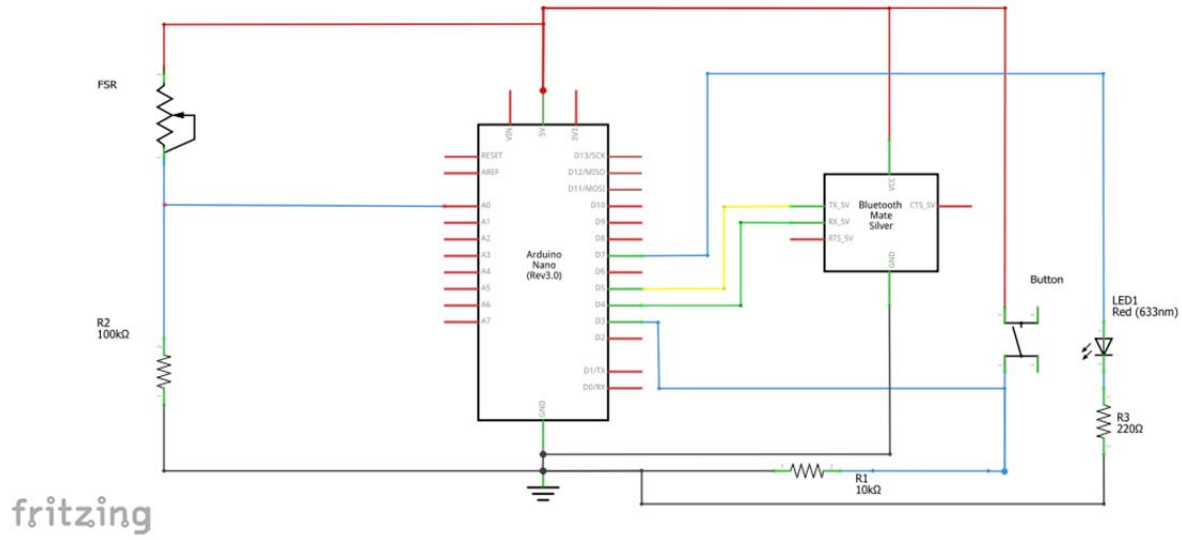


Figura 3.3.2 Schema elettrico del circuito di prova. Realizzato con Fritzing

Montato il circuito, è necessario programmare il microcontrollore e quindi scrivere il codice nel linguaggio di programmazione di Arduino. Il tutto è stato preceduto dalla realizzazione di un flow chart (rappresentato nella figura 3.3.3.) per impostare il ragionamento da seguire e aver facilitata la scrittura del codice.

Il codice consta di una prima parte d'istallazione delle librerie necessarie (Initialization libraries una relativa alla sleep mode: avr/sleep.h e l'altra per il modulo Bluetooth: SoftwareSerial.h) e una d'inizializzazione delle variabili (Initialization variables) utilizzate durante il codice per la gestione del led, del pulsante, del modulo Bluetooth ecc.

La sleep mode permette di spegnere i moduli non utilizzati nel microcontrollore e di conseguenza risparmiare energia. Nei progetti wireless, alimentati a batteria, la modalità sleep mode è dunque indispensabile per aumentare la durata di vita del dispositivo. Il microcontrollore presenta differenti tipi di sleep mode, quella in questo progetto utilizzata è la Power-down, con la quale solo gli interrupt esterni, il 2-wire Serial Interface address watch e il Watchdog continuano ad operare se abilitati.

Le variabili inizializzate presenti nel flow chart sono tre:

- *Flag*, di tipo booleano che può assumere cioè solo due valori: 0 o 1.

  È utilizzata per la gestione del pulsante di accensione-spegnimento del dispositivo: vale zero se il dispositivo è acceso, uno se il dispositivo è spento.

  In questa prima fase la variabile *Flag* viene inizializzata a zero affinché, fornendo energia al dispositivo, lo stesso non si trovi in modalità sleep mode.
- *State*, di tipo booleano, utilizzata per il salvataggio dello stato del pulsante: se premuto (high) *state*=1, se rilasciato/non premuto (low) *state*=0.

- *StateAnt,* di tipo booleano, utilizzata per conservare il valore indicante lo stato del pulsante al ciclo precedente.

Nel flow chart il blocco successivo è quello di setup, relativo ad istruzioni che vengono eseguite un'unica volta. Si configurano:

- in primo luogo, la comunicazione seriale a 9600 baud, cioè ad una velocità di trasmissione di 9600 bit per secondo (Setting Baud rate),
- in secondo luogo, il pin del led come output (Led output) e quello del pulsante come input (Button input),
- successivamente, si realizzano il settaggio e l'abilitazione della sleep mode (Enabling and setting the sleep mode).

Il blocco successivo è denominato void loop che comprende un insieme di istruzioni ripetute all'infinito.

Viene letto il valore della variabile *State* (corrispondente allo stato del pin del pulsante: high o low) e confrontato con il valore della variabile *StateAnt* relativo al ciclo precedente.

Se la variabile *State* è differente da *StateAnt* (ramo NO del flow chart) allora viene eseguito il toggle della variabile *Flag* e *StateAnt* è posta uguale a S*tate*.

A questo punto viene verificato se la variabile *Flag*=1: se non lo è sì disabilitano gli interrupt nel pin del pulsante (Detach Interrupt to the Button pin), se invece lo è (*Flag*=1) si abilitano gli interrupt nel pulsante (Attach Interrupt to the Button pin), si spegne il led (Led OFF) e si attiva la modalità sleep mode. L'abilitazione degli interrupt relativi al pin del pulsante è necessaria affinché sia consentita l'uscita dalla modalità sleep mode e quindi iniziare la lettura dei dati. Infatti, il microcontrollore rimarrà nella modalità sleep mode sino a quando il pin del pulsante non verrà posto a 1 (Button pin=1), che corrisponde al pulsante premuto.

Se invece S*tate=StateAnt* (ramo YES del flow chart*)* significa che il pulsante non è stato premuto e quindi si accende il led e possono essere letti i dati captati dal sensore.

Le operazioni di lettura e invio dei dati al relativo software sono svolte dalla funzione *ReadValue,* descritta nella figura 3.3.4 mediante un flow chart.

La funzione ReadValue permette, attraverso il pin analogico A0 dell'Arduino Nano, di leggere i valori analogici in entrata ogni 10ms (*Tsamp*, tempo di campionamento, che corrisponde a una frequenza di campionamento di 100 Hz), di farne la media su cinque misurazioni e di convertire il valore ottenuto in volt e per ultimo di inviare i dati mediante il modulo Bluetooth.

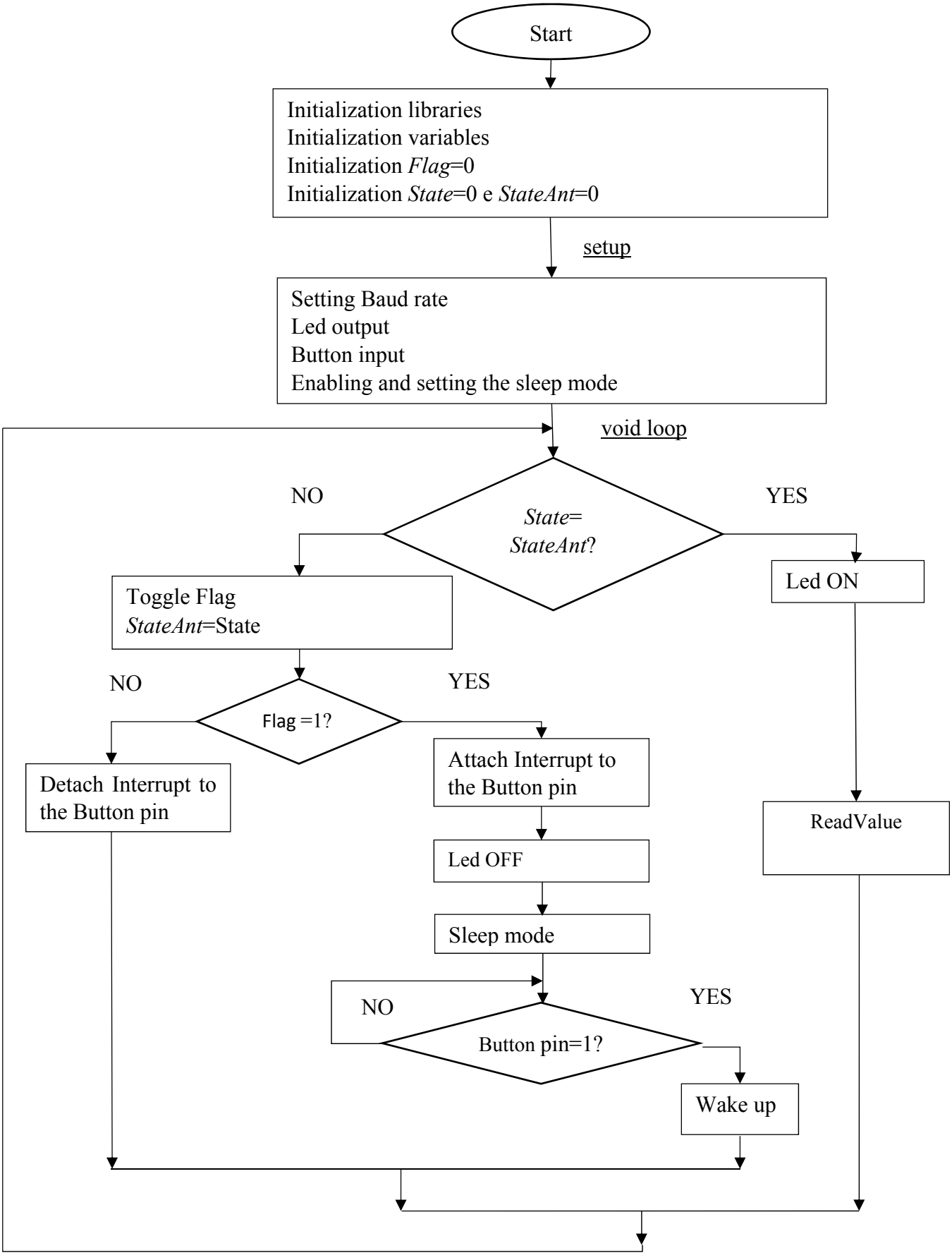


Figura 3.3.3 Flow chart del codice Arduino

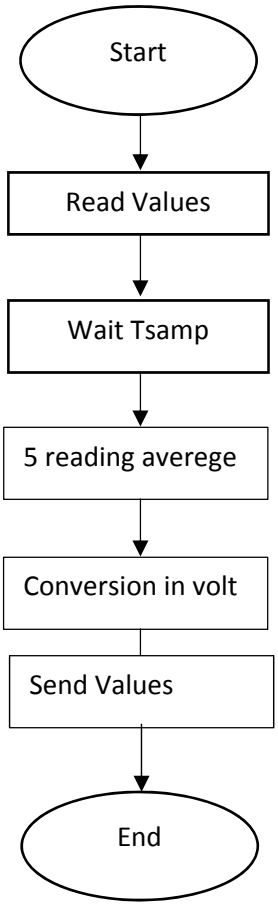


Figura 3.3.4 Flow chart della funzione ReadValue.

Occorre precisare che in questa prima fase di prove, la visualizzazione dei dati avviene con l'applicazione Serial Bluetooth dallo smartphone poiché non è stato ancora realizzato il software di visualizzazione.

La figura 3.3.5 mostra il corretto funzionamento del dispositivo e del relativo codice, si possono notare due modalità differenti:

- una Awake che permette la lettura dei valori captati dal sensore poiché l'ADC (Analog to Digital Converter) è attivo, e il loro invio allo smartphone grazie al modulo Bluetooth.

- l'altra Sleeping (Power-Down-Mode), di risparmio energetico. Quando è attiva non si ha lettura dei dati in quanto tutte le principali funzioni vengono poste in stand-by.

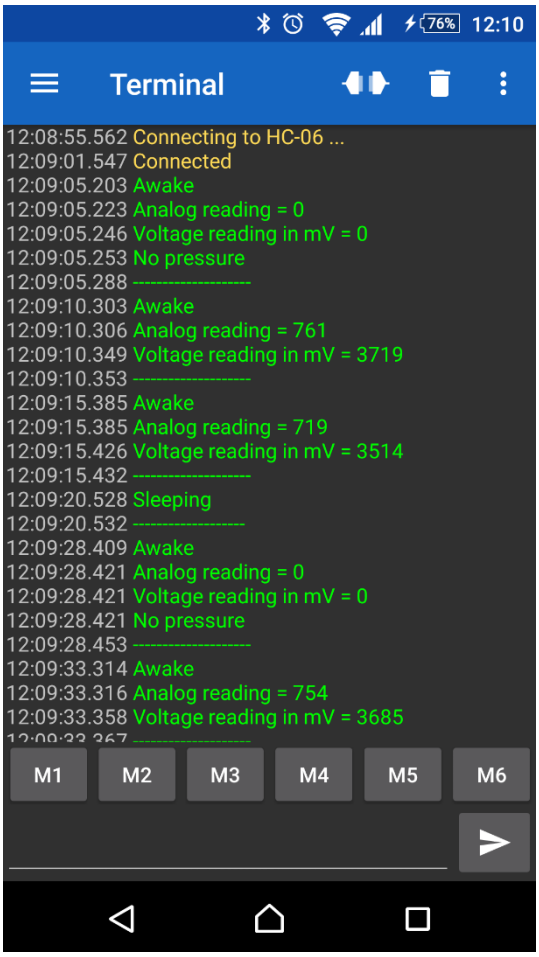


Figura 3.3.5 Visualizzazione delle misure nello smartphone con l'applicazione Serial Bluetooth

3.4 Montaggio su circuito stampato mille fori

Terminate le prove su breadboard e testato il corretto funzionamento, il passo successivo è il montaggio finale sul circuito stampato mille fori.

3.4.1 Scelta della posizione dei sensori

Prima del questo montaggio deve essere definito il numero di sensori che sarà utilizzato e la loro collocazione.

Secondo lo studio di Lin Shu *et al.* [14] nel plantare possono essere individuate 15 aree in corrispondenza delle differenti parti del piede, su cui grava la maggior parte del peso corporeo e che regolano l'equilibrio. Nella figura 3.4.1 sono rappresentate le suddette aree: le aree da 1-2 e 3 in corrispondenza del tallone, le aree 4 e 5 in corrispondenza dell'arco, le aree da 6

a 10 in corrispondenza del metatarso e le aree da 11 a 15 in corrispondenza delle dita.

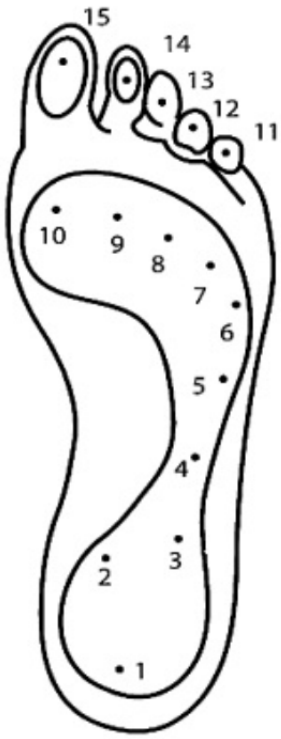


Figura 3.1.1 Aree di maggiore pressione nella pianta del piede [14]

La misura delle forze agenti in queste posizioni può essere utile per ottenere informazioni sulla struttura e la funzione dell'arto inferiore in particolare, e di tutto il corpo in generale.

Con riferimento ai paragrafi 1.3 e 1.4 possiamo dire che:

- nella fase d'impatto la parte che è in contatto con il suolo è quella del tallone. In questa zona si possono collocare due/tre sensori in modo tale che evidenzino i tre possibili gruppi di appoggio del piede:
  - appoggio neutro: tutti i sensori rilevano valori simili,
  - appoggio pronatore: il sensore interno rilava un valore più alto,
  - appoggio supinatore: il sensore esterno rileva un valore più alto;

- nella fase d'appoggio il piede, con il tallone e la parte anteriore, è in totale contatto con il suolo e il peso del corpo si distribuisce nelle due zone attraverso l'arco del piede.

  poiché sono già stati posizionati i sensori nella parte posteriore, ora dovranno essere collocati nella parte anteriore. È opportuno inserirne almeno tre considerando l'area d'appoggio è abbastanza larga.

  Per avere una migliore valutazione della pressione plantare possono essere inseriti uno/due sensori nell'arco del piede in modo da individuare in maniera rapida e facile un tipo camminata supina;

- nella fase d'impulso solo la parte anteriore del piede è in contatto con il suolo. I sensori già posizionati sono sufficienti per conoscere il tipo di appoggio e non è necessario inserirne di nuovi nelle zone delle dita del piede.

In conclusione e in prima istanza si scelgono sei zone in cui posizionare i sensori: tre nel tallone (numerate con 1, 2 e 3 in figura 3.4.2) e tre nel metatarso (numerate con 5,6 e 7 in Figura 3.4.2) poiché queste aree sono sottoposte ad una maggiore pressione durante le normali attività del soggetto.

Per uno studio più accurato si possono aggiungere un sensore (numerato con 4 in figura 3.4.2) nell'arco del piede.

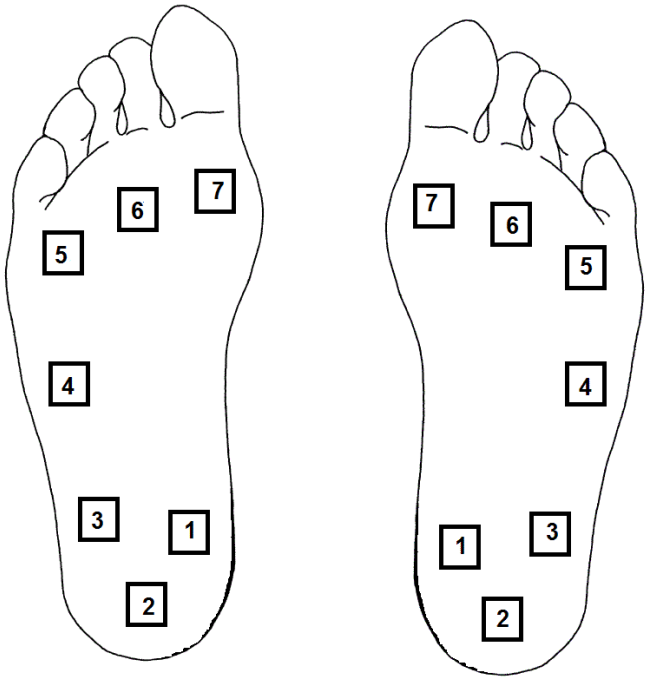


Figura 3.4.2 Posizione scelta per i sensori. Modificata da [29]

3.4.2 Montaggio

Si procede quindi alla saldatura dei componenti nel circuito stampato mille fori dopo avere posizionato nel lato destro, il bluetooth, il bottone e il led e nel lato sinistro, l'Arduino Nano e i connettori per i cavi collegati ai sensori, con le resistenze relative di 100KΩ.

Il peso del dispositivo finale e le sue dimensioni sono sufficientemente ridotte (dimensioni del dispositivo: 20mm x 55mm x 70mm; dimensioni della batteria: 97mm x 26mm x 23mm) consentendo così una facile integrazione nella scarpa senza interferire con il normale ciclo di marcia.

La figura 3.4.3 mostra il risultato finale.

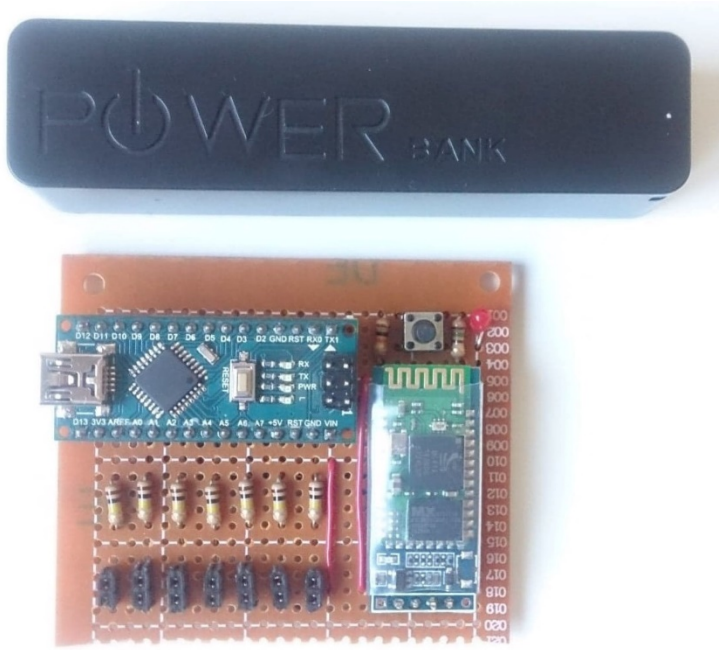


Figura 3.4.3 Montaggio su circuito stampato mille fori

Al dispositivo così ottenuto andranno connessi, mediante cavi, i sette sensori che saranno inseriti nel plantare e poi nella scarpa. La figura 3.4.4 mostra la connessione con sette sensori.

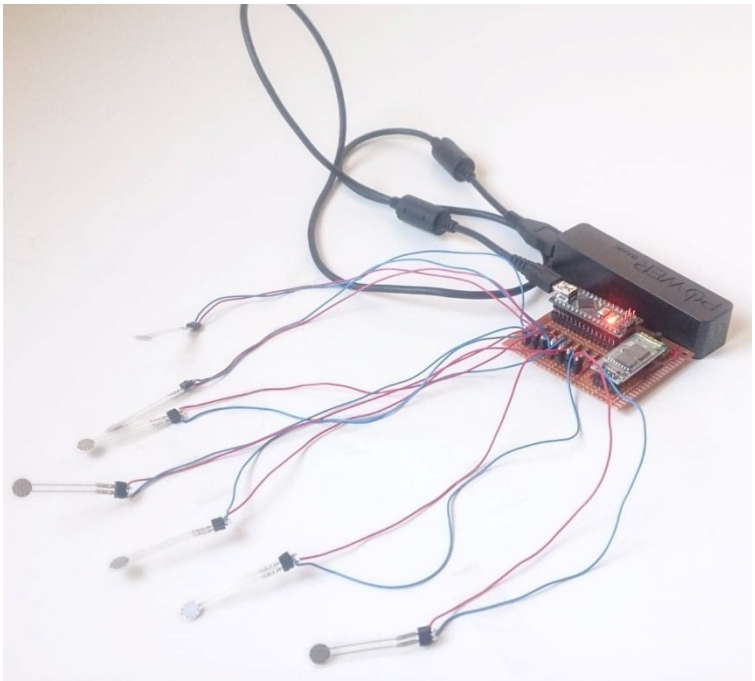


Figura 3.4.4 Connessione con i sensori

A questo punto la funzione ReadValue del paragrafo 3.3 scritta per il circuito di prova non è più utilizzabile poiché valida per un solo sensore, pertanto è stato necessario aggiornarla considerando tutti e sette i sensori.

La figura 3.4.5 mostra il flow chart aggiornato.

La funzione ReadValue così aggiornata permette, attraverso i primi sette pin analogici dell'Arduino Nano (A0-A6), di leggere i valori analogici in entrata ogni 10ms (*Tsamp*, tempo di campionamento, che corrisponde a una frequenza di campionamento di 100 Hz), di farne la media su cinque misurazioni e di convertire il valore ottenuto in volt, in modo del tutto analogo a quanto visto per un singolo sensore.

A questo punto si verifica se il passo del soggetto si è concluso, cioè se il piede è nella fase di oscillazione per l'appoggio successivo e quindi se i sensori sono sottoposti a pressione nulla con conseguente lettura di valori di tensioni in ingresso pressoché nulli.

Verificato che il passo è stato concluso, si controlla se almeno due dei tre sensori posizionati posteriormente e due dei tre sensori posizionati anteriormente hanno superato il valore di soglia (*cost*=113.64V); qualora si abbia un riscontro positivo, i dati saranno inviati attraverso porta seriale Bluetooth.

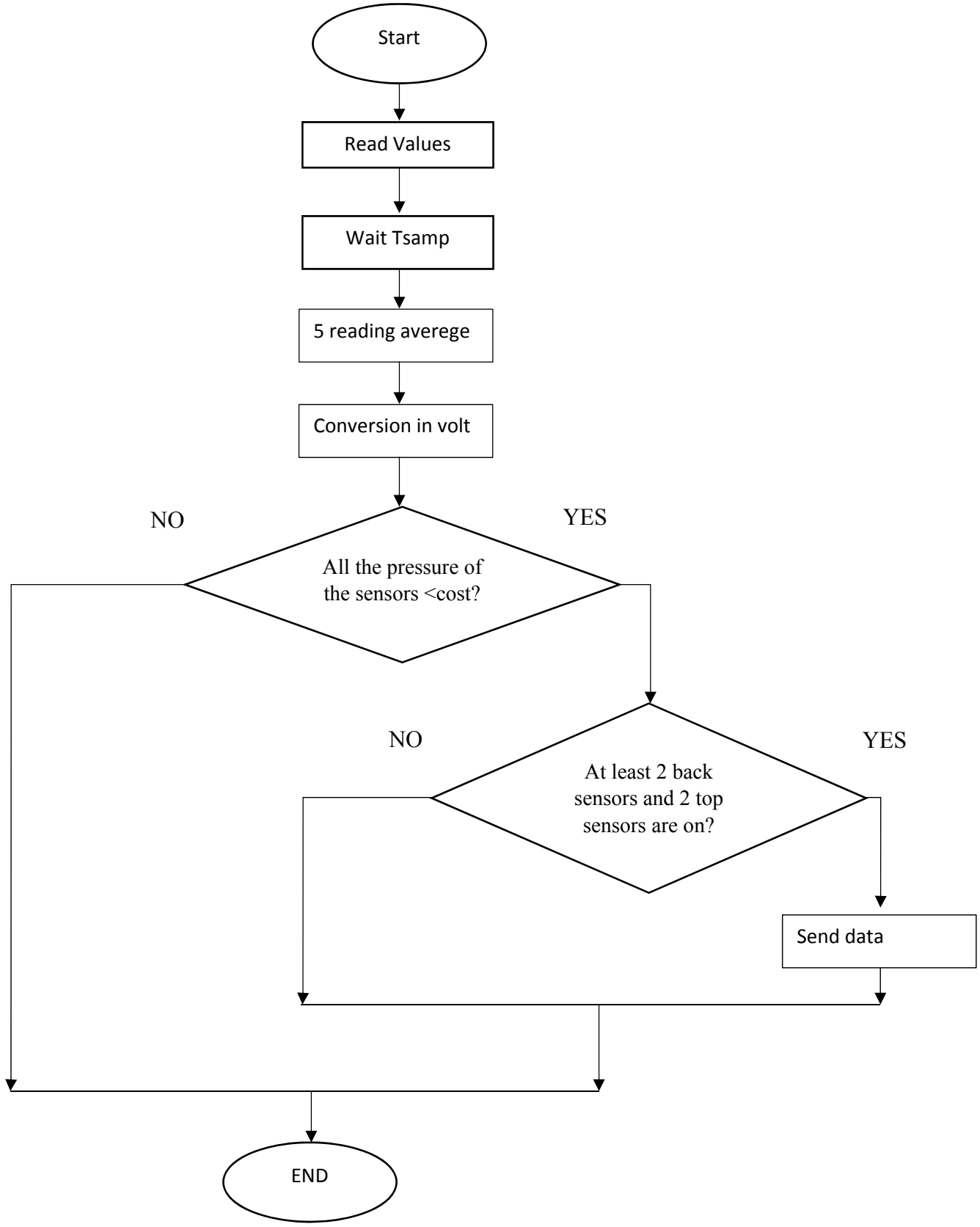


Figura 3.4.5 Flow chart della funzione ReadValue aggiornata.

Capitolo 4

Realizzazioe del sistema

4.1 Hardware

La fase successiva al montaggio è l'integrazione dei sensori nella scarpa. Nei prototipi l'integrazione sarà fatta in un'unica calzatura.

I sensori verranno fissati con nastro adesivo in specifici punti del plantare per impedire il movimento degli stessi. La figura 4.1.1 mostra in quali punti verranno fissati i sensori.

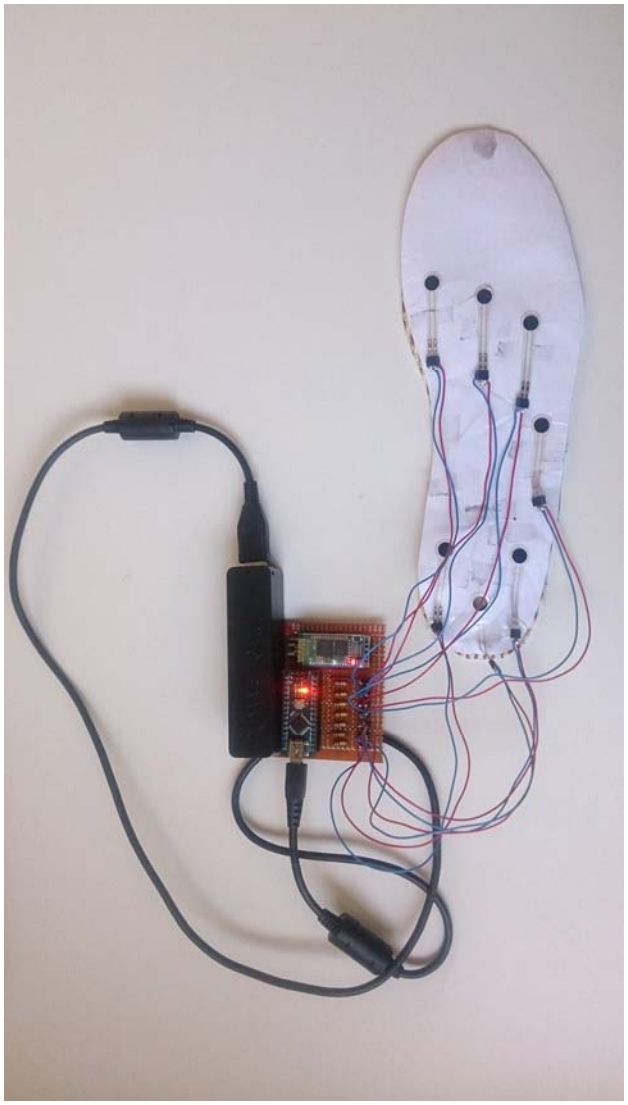


Figura 4.1.1 Posizione dei sensori nella suola

Successivamente il plantare sarà posizionato nella scarpa. I cavi verranno fatti fuoriuscire dalla parte posteriore della calzatura con l'obbiettivo di permettere la connessione con il dispositivo accoppiato alla caviglia del paziente, come mostrato in figura 4.1.2.

Figura 4.1.2 Soluzione finale: hardware

4.2 Software

Il linguaggio di programmazione utilizzato è Java sulla piattaforma Eclipse WindowBuilder.

WindowBuilder è un editor di visualizzazione utilizzato per creare interfacce grafiche per l'utente; l'interfaccia può essere modificata in due modi: scrivendo il codice o utilizzando un editor grafico attraverso una semplice operazione di drag&drop dei componenti nello schermo.

Come indicato nel paragrafo 3.1.6, il sistema di visualizzazione dei risultati acquisiti dai sensori è un computer perché rappresenta un'interfaccia semplice e accessibile a ogni tipo di utente, indipendentemente dal livello di qualifica.

Il dispositivo deve essere connesso al computer tramite Bluetooth utilizzando la password: 1234.

Per ogni paziente dovrà essere creata una cartella personale e queste saranno raggruppate dentro il database CartellaPazienti. Il software consente l'apertura di 4 finestre:

- Finestra Identificazione Paziente
- Finestra Nuovo Paziente
- Finestra Riassuntiva
- Finestra Paziente Registrato

4.2.1 Finestra Identificazione Paziente

È la prima finestra che viene visualizzata non appena il software viene aperto e consente sia la ricerca dello specifico paziente all'interno della CartellaPazienti sia l'inserimento di un nuovo paziente.

Come raffigurato in figura 4.2.1.1. la finestra presenta tre campi di testo (TextField) e due pulsanti.

Qualora si voglia cercare la cartella di un paziente devono essere compilati i tre campi di testo, inserendo nell'ordine: nome, cognome e codice fiscale. Schiacciando il pulsante "Cercare" (Buscar) il software procede alla ricerca.

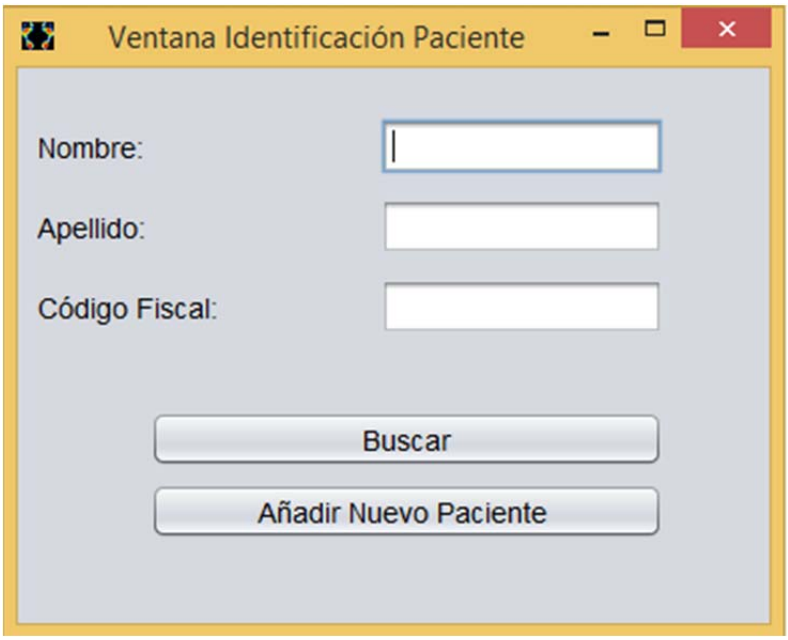


Figura 4.2.1.1 Finestra Identificazione Paziente

Se nel database non è presente alcuna cartella con i dati digitati viene visualizzato il messaggio indicato in Figura 4.2.1.2. Sino a quando non si schiaccia su "Ok" la Finestra Identificazione Paziente non risulta più accessibile.

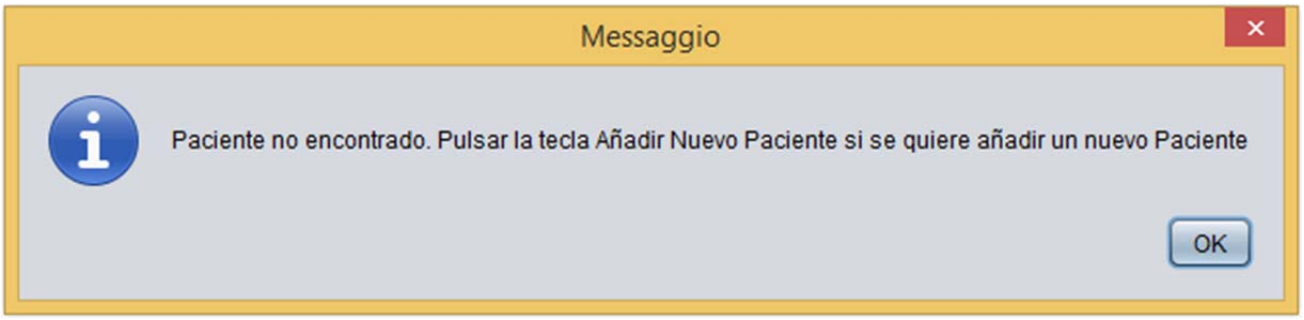


Figura 4.2.1.2 Messaggio di avviso per paziente non trovato nel database

Qualora si voglia inserire nel database un nuovo paziente allora nella Finestra Identificazione Paziente deve essere schiacciato il pulsante "Aggiungi Nuovo Paziente" necessario se il soggetto non è stato preventivamente registrato. In questo modo si apre la seconda finestra denominata Finestra Nuovo Paziente.

.

4.2.2 Finestra Nuovo Paziente

È la finestra che consente di registrare nel database un nuovo paziente pertanto presenta diversi campi di testo che devono essere compilati per identificare in modo univoco l'individuo. L'inserimento è effettivo solo dopo avere schiacciato il pulsante "Registrare". La figura 4.2.2.1 riporta i dati richiesti.

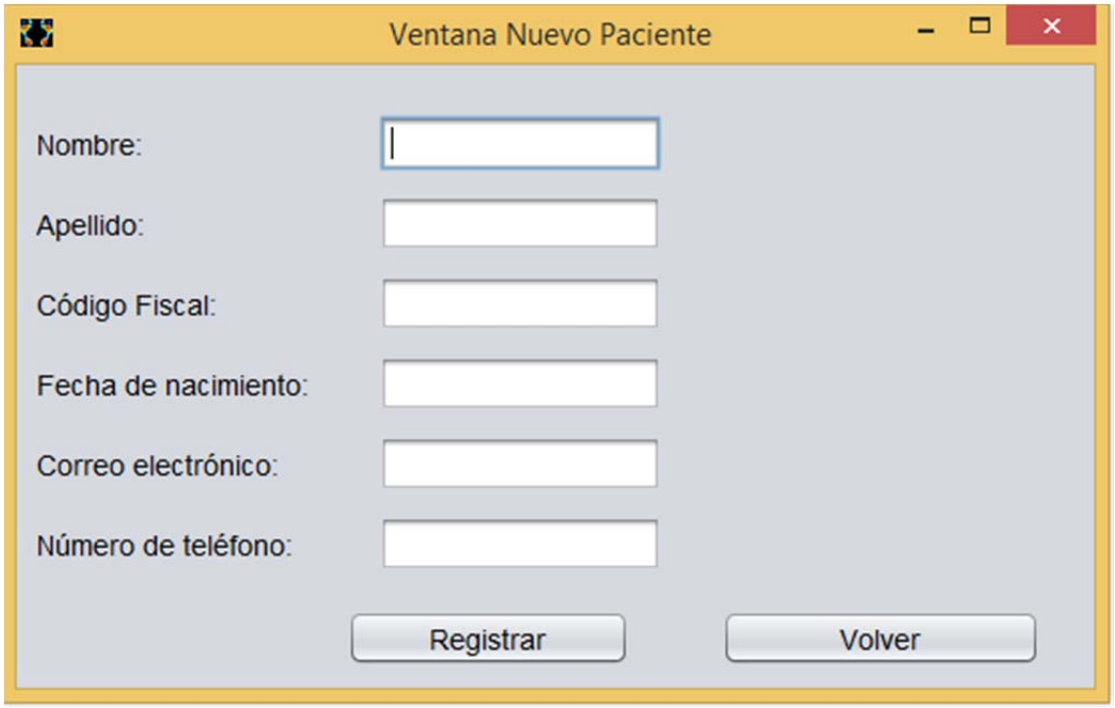


Figura 4.2.2.1 Finestra Nuovo Paziente

Qualora anche solo un campo di testo non venga compilato, schiacciando il pulsante "Registrare" appare la finestra raffigurata in figura 4.2.2.2. in cui viene visualizzato un messaggio di errore.

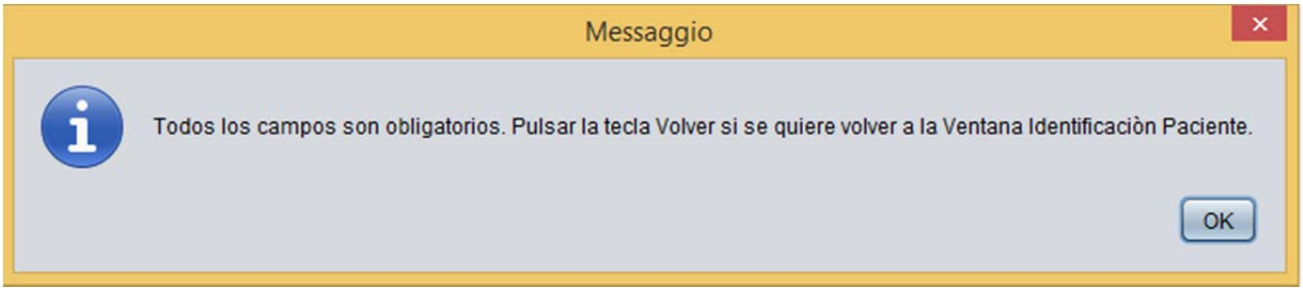


Figura 4.2.2.2 Messaggio di avviso per non avere compilato tutti i campi nella Finestra Nuovo Paziente

Una volta riempiti tutti campi di testo e premuto il tasto “Registrare”, nel database CartellaPazienti è creata in modo automatico una cartella con nome e cognome del paziente (Cartella_Nome_Cognome) che contiene un file di testo chiamato *NomeCognome* contenente le informazioni che si sono appena inserite.

Nella Cartella_Nome_Cognome dello specifico paziente saranno collocati successivamente i file corrispondenti a ciascun esame.

Sempre nella Finestra Nuovo Paziente compare in basso nel lato destro il pulsante “Indietro”, utilizzabile se si vuole ritornare alla finestra precedente.

Il programma è stato realizzato il ritorno nella finestra precedente sia preceduta da una richiesta di conferma, come indicato in figura Figura 4.2.2.3.

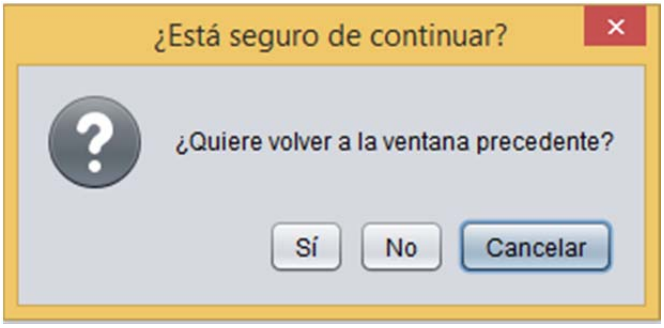


Figura 4.2.2.3 Richiesta di conferma

I bottoni “No”, “Cancel” e “X” permettono di chiudere la finestra e di rimanere in quella corrente; il tasto “Yes” consente di tornare alla pagina Finestra Identificazione Paziente.

4.2.3 Finestra Riassuntiva

È la finestra che compare quando:

- il paziente è stato trovato nel database (premendo il pulsante “Cercare” nella Finestra Identificazione Paziente) o

- il paziente non è stato trovato nel database ed è stato successivamente inserito (premendo il pulsante “Registrare” nella Finestra Nuovo Paziente dopo avere inserito tutti i campi di testo).

La figura 4.2.3.1 riporta le informazioni del paziente presenti nella Finestra riassuntiva.

Figura 4.2.3.1 Finestra Riassuntiva

Nella Finestra Riassuntiva sono altresì presenti i tasti:

- “Indietro” per ritornare alla pagina precedente,
- “Avanti” per arrivare sino alla finestra più importante che consentirà lo svolgimento dell’esame vero e proprio.

4.2.4 Finestra Paziente Registrato

Questa finestra (Figura 4.2.4.1) è la più importante e complessa.

In linea con quanto indicato nel paragrafo 4.2 si è pensato di utilizzare un’interfaccia in cui appare riflessa nello schermo la disposizione dei sensori

nel disegno della pianta del piede e, in funzione del valore della pressione, diversi colori. Nella parte destra dell'immagine sono presenti le 44 possibili colorazioni: blu scuro indica assenza di pressione, rosso pressione massima. Di default compare il blu scuro.

La parte superiore della schermata presenta un menu a barra (menuBar) con tre differenti sottomenù al suo interno: File, Info Paziente e Archivio Esami.

Al di sotto del menuBar compaiono nome e cognome del paziente con la data del giorno.

La parte destra della schermata/finestra presenta tre differenti sezioni:

- Una sezione dove vengono visualizzate le ultime letture relative agli ultimi 10 passi del paziente. La stessa immagine, mostrata in grande sulla sinistra della schermata, in questa sezione è rappresentata in versione ridotta e ripetuta per 10 volte, una per cada passo. La visualizzazione dei risultati è del tutto analoga, dove la pressione esercitata su cada sensore è rappresentata con una scala di colori dal blu scuro al rosso.
- Un'area di testo (TextArea) messa a disposizione per inserire eventuali commenti da parte dell'usuario/utente.
- Una sezione dedicata ai pulsanti.

Quest'ultima presenta 4 differenti pulsanti e un menu a tendina:

- "Indietro": permette di ritornare indietro alla Finestra Identificazione Paziente dopo aver visualizzato la finestra d'avviso della figura 4.2.2.3 nel paragrafo 4.2.2,
- "Incominciare Acquisizione": permette di iniziare l'acquisizione dei dati subito dopo avere selezionato la porta seriale di connessione Bluetooth nel rispettivo menu a tendina (comboBox) indicato dall'etichetta "Selezionare Porta Bluetooth" situata appena sopra.
- "Terminare Acquisizione": permette di terminare l'acquisizione dei dati disconnettendo il modulo Bluetooth.

- “Salvare Esame”: permette di salvare i dati acquisiti in un textFile nella cartella del paziente, nominato con la data del giorno dell’esame.

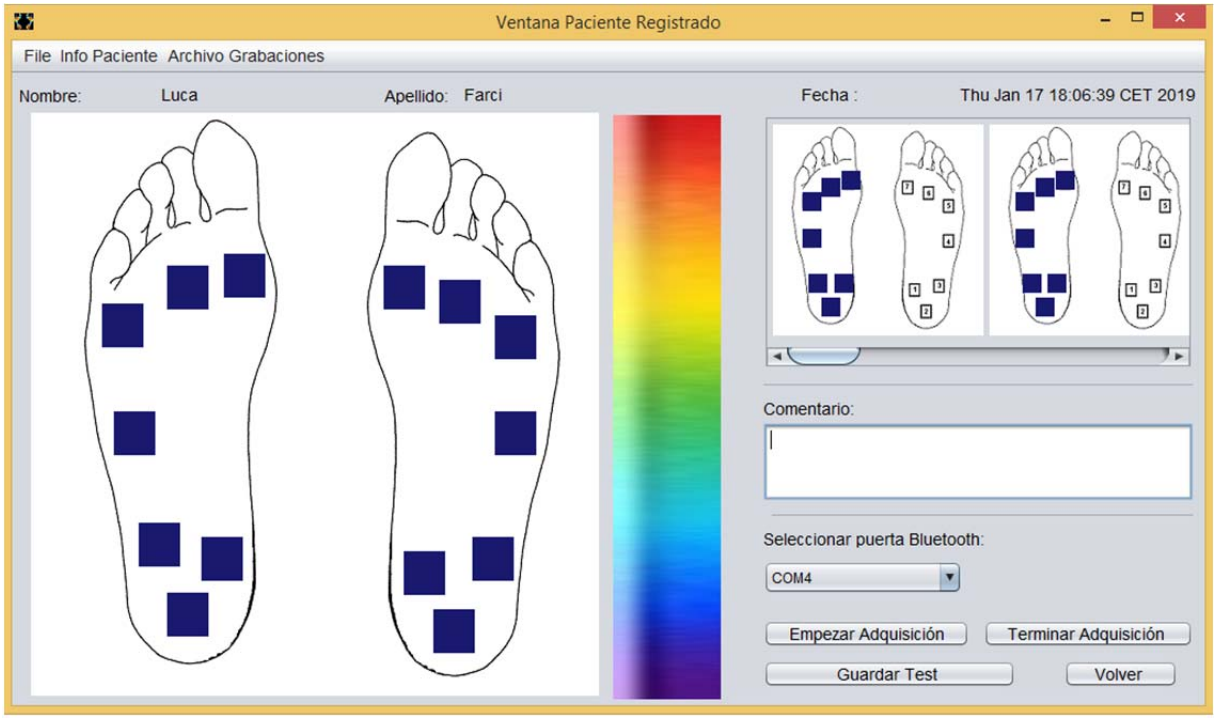


Figura 4.2.4.1 Finestra Paziente Registrato.

Capitolo 5

Sperimentazione

5.1 Prove sperimentali

Un esame baropodometrico si svolge sostanzialmente in due fasi:

- in statica, il paziente è posizionato su una pedana per rivelare i carichi pressori dei piedi in posizione statica bipodalica;
- in dinamica, il paziente svolge una serie di passi valutando i carichi pressori e come i piedi svolgono le fasi del passo.

Il solo esame statico su pedana non è esaustivo poiché alcune anomalie possono essere rilevate solo durante il cammino e quindi nella fase dinamica.

Nei convenzionali esami baropodometrici in fase dinamica si chiede al paziente di camminare su una pedana mantenendo un solo piede poggiato sulla pedana e l'altro al di fuori di essa, costringendo quindi il soggetto a svolgere un camino prestabilito.

Si ritiene pertanto che la presenza della piattaforma influenzi non poco i pazienti, i quali prestano particolare attenzione all'esecuzione del passo, rallentando spesso la velocità del cammino e introducendo un movimento dell'arto inferiore non naturale in prossimità della pedana.

Il dispositivo creato in questo progetto e descritto nei capitoli 3 e 4 ha il vantaggio di consentire contemporaneamente sia l'esame barometrico in fase statica sia in fase dinamica poiché è inserito nella calzatura del paziente, ed inoltre permette al soggetto di camminare nella maniera più fluente possibile e di riprodurre in maniera più fedele il suo passo standard.

Per valutarne il funzionamento il dispositivo è stato esaminato in riferimento ai tre possibili gruppi di appoggio: neutro, prono e supino descritti nel paragrafo 1.4.

Le prove sono state svolte sul piede sinistro di taglia 41.

Appoggio neutro:

l'appoggio è nella parte centrale del piede e pertanto il peso del corpo si ripartisce uniformemente su tutta la pianta.

Il soggetto nel ciclo di marcia disegna una virtuale linea retta e le misure dei sensori anteriori e posteriori posizionati nell'avampiede sono pressochè uniformi.

La figura 5.1.1 mostra i risultati ottenuti dopo cada passo quindi dopo il susseguirsi delle tre sottofasi di cui al paragrafo 1.3 (Contatto o initial contact, Appoggio o midstance e Propulsione o push off) della fase di appoggio.

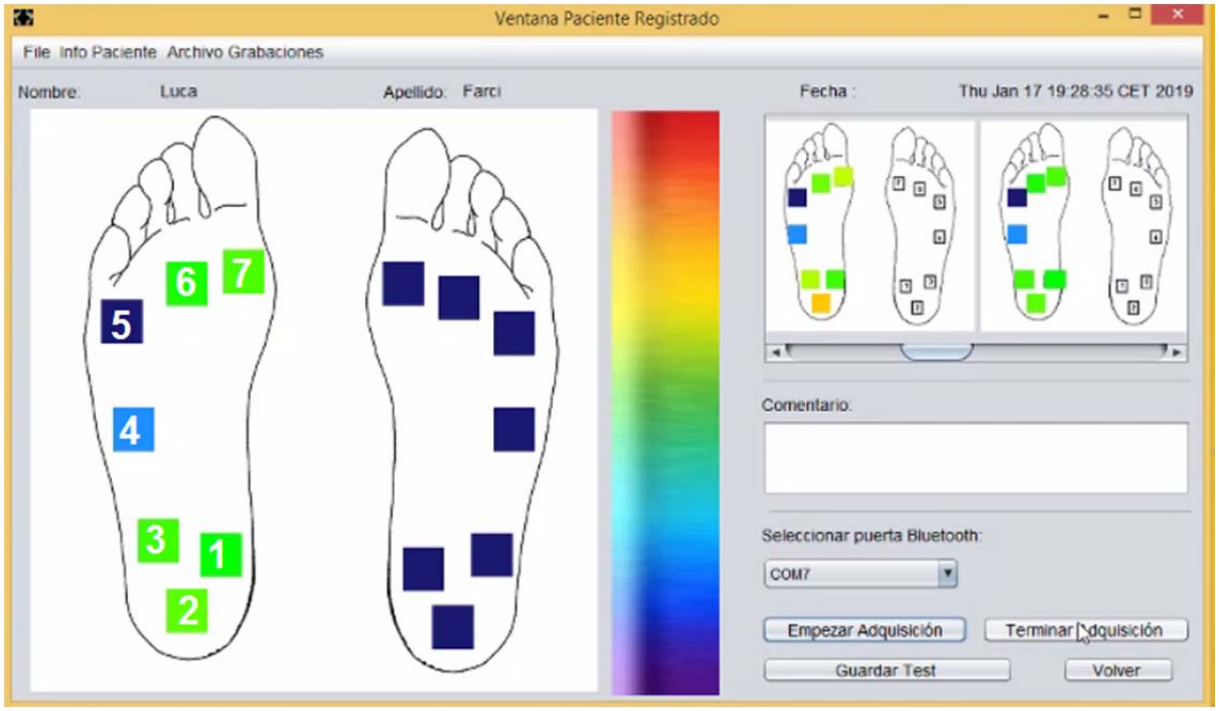


Figura 5.1.1 Risultati nell'appoggio neutro

La parte centrale della figura riporta la scala delle colorazioni associate alle differenti misure di pressione come indicato nel paragrafo 4.2.4 (il blu scuro indica l'assenza di pressione e il rosso, valori massimi di pressione) e in alto a destra sono visualizzate le ultime letture relative agli ultimi passi del paziente. I risultati ottenuti indicano che nella sottofase *Contatto o initial contact*, quando il soggetto inizia la camminata poggiando il tallone al suolo,

l'appoggio è neutro (sensori n. 1, 2 e 3) e anche nella sottofase *propulsione/impulso* predomina l'appoggio neutro (sensori n. 6 e 7).

Appoggio supino:

la maggior parte del peso del corpo si distribuisce nella zona esterna del piede, sottoponendola a maggiore tensione e sforzo e causando la sua inclinazione verso l'esterno.

Pertanto, i sensori esterni (sensori n. 2, 3, 4, e 5) rilevano una pressione maggiore rispetto a quelli posizionati nella zona interna (sensori n. 1 e 7).

La figura 5.1.2 mostra i risultati ottenuti in cui si può vedere dalla figura possiamo notare uno spiccato comportamento di supinazione sia nella fase sottofase *Contatto o initial contact* (i sensori n. 2 e 3 rilevano una pressione maggiore del sensore n. 1) sia nella sottofase *propulsione/impulso* (sensori n. 5 e 6), inoltre si può notare un valore di pressione importante rilevato dal sensore n. 4 posizionato sull'arco.

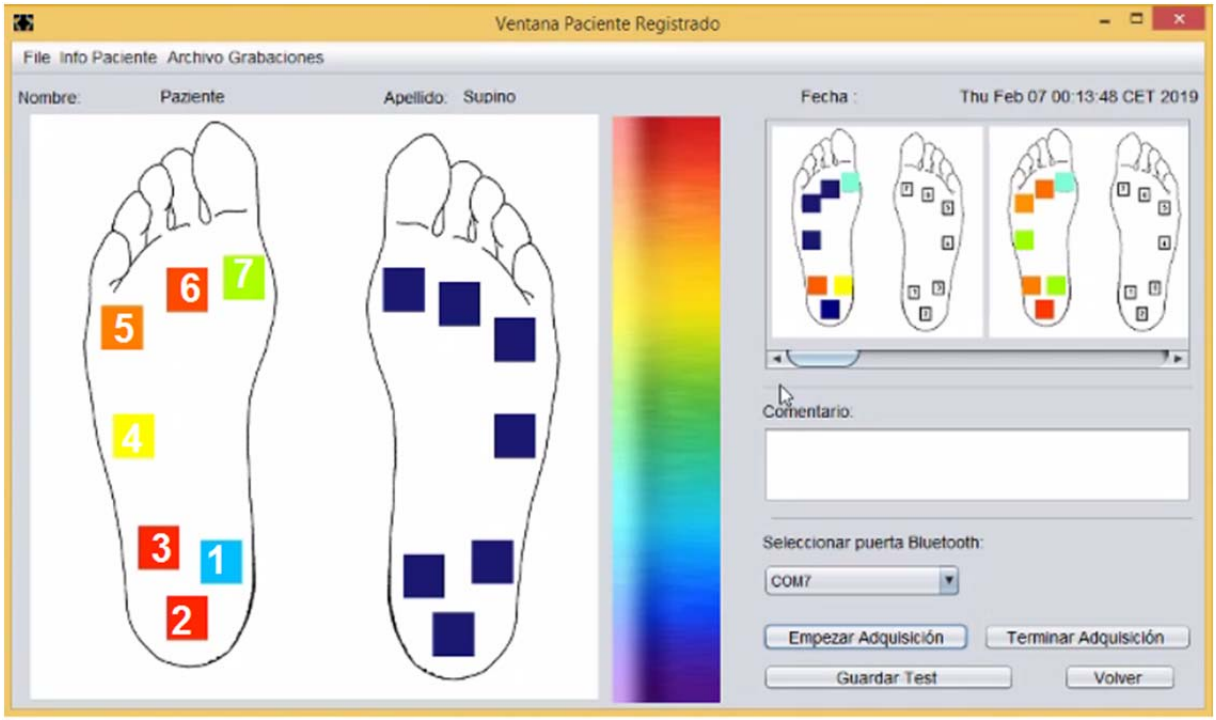


Figura 5.1.2 Risultati dell'appoggio supino

Appoggio prono:

il peso del corpo si distribuisce nella parte interna del piede pertanto i sensori posizionati in questa parte rilevano valori di pressione superiori rispetto a quelli presenti nella parte esterna sia anteriormente sia posteriomente.

La figura 5.1.3 mostra i risultati ottenuti in cui si può notare una pronazione predominante sia nella sottofase sottofase *Contatto o initial contact* sia in quella di *propulsione/impulso* nella parte anteriore del piede: il sensore posisizionato nella parte sinistra (sensore n. 5) riceve con estrema difficoltà i dati, così come quello posto nell'arco del piede (sensore n. 4).

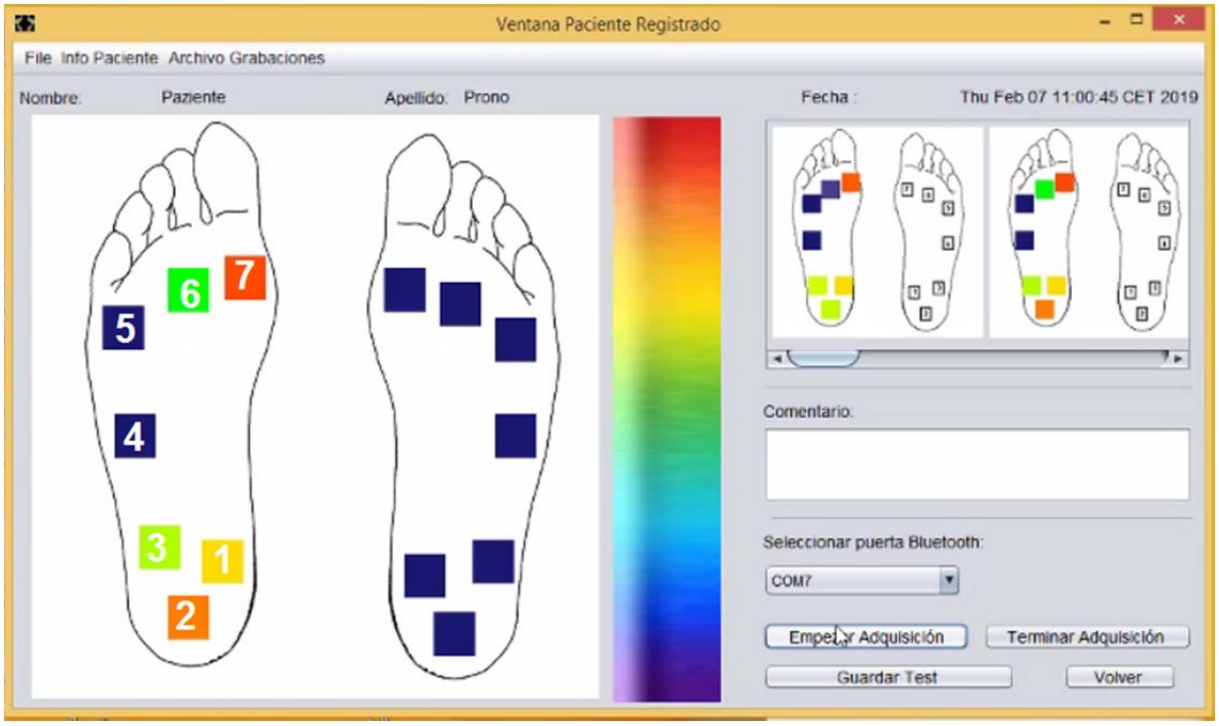


Figura 5.1.3 Risultati dell'appoggio prono

Capitolo 6

Pianificazione finanziaria e temporale

6.1 Analisi dei costi

L'analisi dei costi e il tempo di sviluppo del dispositivo rappresentano una condizione imprescindibile nel progetto ingegneristico.

Con *costi di prodotto* indichiamo il costo delle risorse associabili, in modo diretto o indiretto, alla realizzazione del dispositivo.

Si suddividono in:

- *Costi di lavoro diretto*: sono i costi relativi al personale che si occupa dell'intero progetto (per le operazioni di trasformazione fisica e di assemblaggio).

  In questo caso il lavoro è stato svolto da un solo ingegnere (técnico Superior B) che ha dedicato circa 5 mesi al progetto. Considerando che il suo salario medio mensile è di circa 2.098,13$ [28] e sommando anche i costi della previdenza sociale (in Spagna è il 34% del salario medio mensile), il costo del lavoro diretto finale pari a 14.059,76$ (10.492,94$ + 3.566,82$).

- *Costi diretti di materiali*: sono i costi relativi agli acquisti esterni di materie prime, semilavorati e componenti associabili direttamente alla realizzazione di un singolo prodotto.

Nel paragrafo 3.1 sono stati indicati i diversi pezzi che compongono il dispositivo e con essi il loro costo, riepilogato nella seguente Tabella 6.1.

Tabella 6.1 Costi diretti dei materiali

PEZZO	QUANTITÀ	COSTO UNITARIO (usd)	COSTO TOTALE (usd)
Arduino Nano	1	22,00	22,00
Sensori	7	4,51	31,57
Power Bank	1	17,06	17,06
Resistenze	8	0,10	0,80
Modulo Bluetooth	1	2,56	2,56
Spese varie (cavi, stagno per saldature)	_	5,00	5,00
Led	1	0,75	0,75
Pulsante	1	0,20	0,20
Plantare	1	1,14	1,14
TOTALE			**81,08**

- *Costi indiretti di produzione:* sono i costi associabili direttamente all'attività produttiva nel suo complesso, ma non alla realizzazione della singola unità di prodotto; vengono convenzionalmente divisi in costi indiretti variabili, dipendenti dal volume produttivo (per esempio i costi di manutenzione, di controllo qualità ecc.) e in costi indiretti fissi, indipendenti dal volume produttivo (per esempio i costi delle utenze quali energia elettrica, acqua, telefono, internet, riscaldamento, gli affitti, l'assicurazione, gli ammortamenti [30].

  I costi indiretti di produzione sono stimabili in circa il 6% (4,86$) del costo di materiali diretti, ottenendo così un costo totale per cada unità di 85,94$.

Nella tabella 6.2 sono riportati i valori dei differenti tipi di costi e quindi il costo complessivo per lo sviluppo di ogni singola sezione.

Tabella 6.2 Costo totale di sviluppo del dispositivo

TIPOLOGIA COSTO	VALORE COSTO (usd)
Costo del lavoro diretto	14.059,76$
Costi diretti dei materiali	81,08**$**
Costi indiretti di produzione	4,86$
COSTO TOTALE	**14.145,70$**

Il costo indicato è approssimativo e suscettibile di variazioni in riferimento al costo effettivo dei materiali e alla voce "spese varie". Potrebbe inoltre diminuire se venissero realizzati più dispositivi.

Per avere una stima completa del costo del dispositivo, devono poi essere addizionati anche i cosiddetti *costi di periodo* (definiti anche spese discrezionali) che comprendono attività non direttamente associabili alla realizzazione di prodotto quali i costi di ricerca e sviluppo, le spese amministrative, generali e di vendita.

Il dispositivo realizzato dovrà successivamente essere distribuito nel mercato, al prodotto verrà applicato un mark up per recuperare i costi sostenuti in fase di sviluppo.

Con mark up si definisce la differenza tra il prezzo di vendita di un bene o servizio e il suo costo di produzione, solitamente espressa in percentuale del costo stesso.

Ricordando quanto detto nella tabella 6.2 in cui il costo totale per la realizzazione di un solo dispositivo è pari a 14.145,70$, di cui 85,94$ per i costi dei materiali e di produzione per un unico dispositivo e 14.059,76$ per il costo di un'unica unità di personale, nella tabella 6.3 sono riportati: il numero minimo di dispositivi e il loro prezzo, maggiorato del mark up, che l'azienda dovrebbe commercializzare per raggiungere il break even point (punto di pareggio) ovvero la quantità di prodotto venduto necessaria a coprire i costi precedentemente sostenuti, al fine di chiudere senza profitti o perdite.

Tabella 6.3 Prospetto dei prezzi di vendita

NUMERO D'UNITA	PREZZO UNITARIO (usd)	MARK UP (%)
150	94,53$	10
143	98,83$	15
132	107,43$	25

Capitolo 7

Conclusioni

7.1 Conclusioni e sviluppi futuri

In questo lavoro si è realizzato un sistema indossabile per la misura di pressioni plantari utilizzabile sia in statica che in dinamica. Tali misurazioni sono di interesse per determinare eventuali anomalie del piede e individuare soluzioni utili a correggerle come la progettazione di plantari personalizzati.

Il sistema consiste in una serie di sette sensori integrati nel plantare della scarpa e del dispositivo vero e proprio agganciato alla caviglia del paziente connesso tramite modulo Bluetooth al software di visualizzazione dei dati installato sul computer.

Il dispositivo ad oggi presenta delle limitazioni che possono essere superate apportando una serie di migliorie nell'hardware e nel software.

Si indicano in primo luogo gli interventi apportabili nell'hardware:

- aggiungere più sensori per una realizzazione di uno studio più accurato.
  si potrebbe posizionare un sensore in ciascuna delle quindici zone suggerite da Lin Shu et al. [12]. Per fare ciò però sarebbe necessario multiplexare le entrate tenendo poichè l'Arduino Nano presenta solo otto porte analogiche;
- sostituire l'Arduino Nano con il solo microcontrollore Atmega328. In questo modo i costi sarebbero più contenuti considerando che il prezzo di un Atmega328 è di soli 1,88$ [31].
  Per contro, con questa soluzione, si dovrebbe utilizzare un'altra fonte d'alimentazione non connessa alla porta USB (le batterie da 1,5V per esempio) e scrivere il codice in linguaggio C o Assembler, sicuramente più complesso;
- aggiungere altri tipi di sensori come accelerometri;
- creare differenti plantari sensorizzati per differenti dimensioni del piede: nei piedi di misura diversa, sarà diversa anche la posizione dei sensori.

Per questo motivo per uno studio ortopedico è utile, per esempio, avere a disposizione due dispositivi, uno per ciascuno dei piedi, e cambiare i differenti plantari secondo il numero del paziente. Ciò presuppone che la connessione del dispositivo con il plantare sia facile e rapida.

Di seguito i possibili interventi sul software:

- aggiungere differenti grafici dove si rappresentano la pressione media e la forza in funzione del tempo trascorso in modo che si possa osservare l'evoluzione di uno o più cicli di marcia;
- visualizzare i risultati con una mappa di pressioni come mostrato in Figura 7.1.1: si uniscono i punti sottoposti alla stessa pressione dandogli lo stesso colore, non limitandosi ad analizzare i dati relativi alla posizione dei sensori.

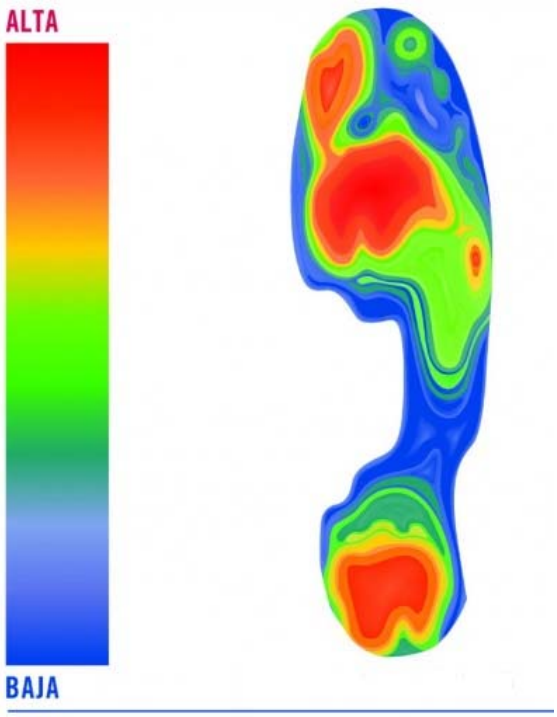


Figura 7.1.1. Esempio di pressione della pianta del piede. Modificata da [36].

Come opzione complementare, si potrebbe affiancare al software di visualizzazione istallato nel computer, un sistema/app basato su iOS o in Android, aumentando così la portabilità del sistema anche in sistemi di visualizzazione (Smartphone o tablet) più immediati.

Bibliografia:

1. https://www.flickr.com/photos/71742660@N04/6788678407
2. https://es.wikipedia.org/wiki/Pie
3. http://www.scielo.org.ve/scielo.php?script=sci_arttext&pid=S0798-40652013000400015
4. https://www.fisioterapia-online.com/articulos/el-pie-su-estructura-sus-arcos-y-los-tipos-de-pies-segun-estos-arcos
5. https://www.runners.es/entrenamiento/articulo/aprende-elegir-tu-calzado
6. Roisin Howard, November 2017, The application of data analysis methods for surface electromyography in shot putting and sprinting.
7. https://www.webconsultas.com/ejercicio-y-deporte/medicina-deportiva/tipos-de-pisada
8. https://medlineplus.gov/spanish/diabeticfoot.html
9. http://www.noene-italia.com/piede-reumatico/
10. https://www.quiropracticaagote.com/2018/07/pisada-y-dolor/
11. http://www.zebris.de
12. Bamberg, S.; Benbasat, A.Y.; Scarborough, D.M.; Krebs, D.E.; Paradiso, J.A. Gait analysis using a shoe-integrated wireless sensor system. *IEEE Trans. Inf. Technol. Biomed.* **2008**, *12*, 413–423.
13. Benocci, M.; Rocchi, L.; Farella, E.; Chiari, L.; Benini, L. A Wireless System for Gait and Posture Analysis based on Pressure Insoles and Inertial Measurement Units. In *Proceeding of the 3rd International Conference on Pervasive Computing Technologies for Healthcare, 2009. Pervasive Health 2009*, London, UK, 1–3 April 2009; pp. 1–6.
14. Lin Shu, Tao Hua, Yangyong Wang, Qiao Li, David Dagan Feng, and Xiaoming Tao, In-shoe plantar pressure measurement and analysis system based on fabric pressure sensing array, Information Technology in Biomedicine, IEEE Transactions on **14** (2010), no. 3, 767–775.
15. www.sensormedica.com
16. https://store.arduino.cc/usa/arduino-nano
17. https://www.theengineeringprojects.com/2018/06/introduction-to-arduino-nano.html
18. https://es.aliexpress.com/store/product/Free-Shipping-FSR402-Force-Sensitive-Resistor-0-5-inch-FSR-For-arduino-compatible-Force-Sensing-Resistor
19. *SparkFun Electronics.*"FSR Force Sensing Resistor Integration Guide and Evaluation parts Catalog".

20. https://es.aliexpress.com/item/1-unids-lote-HC-05-Bluetooth-Serial-Pass-m-dulo-inal-mbrico-de-comunicaci-n-en/32851854441.html?albbt=Google_7_shopping&isdl=y&slnk=&albslr=221466033&src=google&acnt=494-037-6276&aff_platform=google&crea=es32851854441&netw=u&plac=&albcp=1633820309&mtctp=&aff_short_key=UneMJZVf&gclid=CjwKCAiA767jBRBqEiwAGdAOr8KkTo3zANfpCoJAJGNCWdjnmwZRpsHc1ByzQ0AoOY2aFqsqYdw_RxoCsWcQAvD_BwE&albag=63890294393&albch=shopping&albagn=888888&trgt=296904914040&device=c&gclsrc=aw.ds

21. https://it.aliexpress.com/item/4-pcs-lot-New-Brand-AA-rechargeable-battery-3000mah-1-5V-New-Alkaline-Rechargeable-batery-for/32425789857.html

22. https://www.amazon.it/SODIAL-Scomparto-batterie-plastica-colore/dp/B00FFY2UWC/ref=sr_1_2?s=electronics&ie=UTF8&qid=1539678293&sr=1-2&keywords=scomparto+per+batterie

23. https://it.aliexpress.com/item/1-pz-lotto-800-mAh-Li-Ion-9-V-Batterie-Ricaricabili-Per-rilevatori-di-Fumo-Microfoni/32840725183.html?spm=a2g0y.search0104.3.298.6fe636fcSfNGKs&transAbTest=ae803_5&ws_ab_test=searchweb0_0%2Csearchweb201602_5_10065_10068_318_319_10696_450_10084_10083_10618_452_535_534_10304_533_10307_10820_532_10821_10302_204_10843_10059_10884_10887_100031_320_10103_448_449%2Csearchweb201603_60%2CppcSwitch_0&algo_pvid=e38acf56-93fd-4810-a2c9-04d5f6904f18&algo_expid=e38acf56-93fd-4810-a2c9-04d5f6904f18-42

24. https://it.aliexpress.com/item/5-x-nero-custodia-in-plastica-14-99-cm-2-connettore-del-cavo-9-v-batteria/32844579365.html?spm=a2g0y.search0104.3.96.6fe636fcSfNGKs&transAbTest=ae803_5&ws_ab_test=searchweb0_0%2Csearchweb201602_5_10065_10068_318_319_10696_450_10084_10083_10618_452_535_534_10304_533_10307_10820_532_10821_10302_204_10843_10059_10884_10887_100031_320_10103_448_449%2Csearchweb201603_60%2CppcSwitch_0&algo_pvid=e38acf56-93fd-4810-a2c9-04d5f6904f18&algo_expid=e38acf56-93fd-4810-a2c9-04d5f6904f18-13

25. https://www.amazon.fr/Clip-Sonic-Technology-TEA109S-Batterie/dp/B00NMKYCBI
26. https://it.wikipedia.org/wiki/Diagramma_di_Gantt

27. *https://www.lexington.es/blog/metodo-diagrama-gantt-gestionar-proyectos-empresa/*

28. Boletín official del estado, Núm. 8 Martes 9 de enero de 2018 Sec. III. Pág. 3859

29. Reflessologia Zu Morfologia, Autore A. E. Baldassarre Pagina: 148

30. Occhiocupo F. *Certificazione di un prodotto e analisi dei costi di un dispositivo medico*, 20/07/2011.

31. https://www.microchip.com/wwwproducts/en/ATmega328p

Appendice A, Codici Arduino.

A.1 Codice di prova.

```
#include <avr/sleep.h>// AVR library sleep modes
#include <SoftwareSerial.h> //to add one more serial port called software serial.
#define RX 4  // D4      and that serial port can be created at any two pins of Arduino
#define TX 5  // D5

SoftwareSerial SoftSerial(RX,TX);
const int wakeUpPin=3; //Pin interrupts D3: Int1
int pinLed=7;
bool flag = false; // false No wake up true wake up
bool estado, estadoAnt = false;
int a0;
                ///Sensor 0\\\
int fsrPin=0 // the FSR and 100K pulldown are connected to a0
int fsrReading; // the analog reading from the FSR resistor divider
int fsrVoltage; // the analog reading converted to voltage

void setup() {
  // put your setup code here, to rA0un once:
  Serial.begin(9600); //9600 baud Serial Comuniacation
  SoftSerial.begin(9600); // 9600 baud Software Comuniacation
  digitalWrite(LED_BUILTIN,OUTPUT);
  pinMode(pinLed,OUTPUT);
  pinMode(wakeUpPin,INPUT_PULLUP);
  sleep_enable();//Enabling sleep mode
```

```
 set_sleep_mode(SLEEP_MODE_PWR_DOWN);//Setting the sleep mode
 digitalWrite(pinLed,HIGH);
 digitalWrite(LED_BUILTIN,HIGH);
 //attachInterrupt(1, Wake_Function, RISING); // when the button is press go to Wake_Function
    /*              |  |     \ Define the mode to trigger the interrupt  LOW to trigger the interrupt whenever the pin is low,
                  |  |                                    CHANGE to trigger the interrupt whenever the pin changes value
                  |  |                                  RISING to trigger when the pin goes from low to high,
                  |  |                                 FALLING for when the pin goes from high to low.
                  |   Function called when the interrupt is called
                   \
                     interrupt pin 1,D3  */
}


void loop() {

 // put your main code here, to run repeatedly:
 estado = digitalRead(wakeUpPin);
  if (estado != estadoAnt){
     flag = !flag;
     estadoAnt = estado;

  // Allow wake up pin to trigger interrupt on low.
  if (flag) {  // if flag = HIGH go to sleep
     Sleep_Function();
  }
  else {
```

```
        // Disable external pin interrupt on wake up pin.
        detachInterrupt(1);
    }
    // do what you must do here
    }
    Serial.println("Awake");
    SoftSerial.println("Awake");
    digitalWrite(pinLed,HIGH);
    digitalWrite(LED_BUILTIN,HIGH);
    Read_Value();
    delay(5000); // <---press de button for 5 sec to put to sleep arduino

}



void Sleep_Function(){
  attachInterrupt(1, Wake_Function, RISING);
        // Wake up when wake up pin is low.

         Serial.println("Sleeping");
         Serial.println("--------------------");
         SoftSerial.println("Sleeping");
         SoftSerial.println("--------------------");
         digitalWrite(pinLed,LOW);
         digitalWrite(LED_BUILTIN,LOW);
         delay(1000);
         sleep_cpu();
 }
// here the interrupt is handled after wakeup
void Wake_Function(){
```

```
  // execute code here after wake-up before returning to the loop() function
  // timers and code using timers (serial.print and more...) will not work here.
  // we don't really need to execute any special functions here, since we
  // just want the thing to wake up
   }

void Read_Value (){
  fsrReading = analogRead(0);
  Serial.print("Analog reading = ");
  Serial.println(fsrReading);
  SoftSerial.print("Analog reading = ");
  SoftSerial.println(fsrReading);
  // analog voltage reading ranges from about 0 to 1023 which maps to 0V to 5V (= 5000mV)
  fsrVoltage = map(fsrReading, 0, 1023, 0, 5000);
  Serial.print("Voltage reading in mV = ");
  Serial.println(fsrVoltage);
  SoftSerial.print("Voltage reading in mV = ");
  SoftSerial.println(fsrVoltage);
  if (fsrVoltage == 0) {
  Serial.println("No pressure");
  SoftSerial.println("No pressure");
  }
  Serial.println("--------------------");
  SoftSerial.println("--------------------");
}
```

A.2 Codice Finale.

```
#include <avr/sleep.h>// AVR library sleep modes
#include <SoftwareSerial.h> //to add one more serial port called software serial.
#define RX 4  // D4      and that serial port can be created at any two pins of Arduino
#define TX 5  // D5
SoftwareSerial SoftSerial(RX,TX);
const int wakeUpPin=3; //Pin interrupts D3: Int1
int pinLed=7;
bool flag = false; // false No wake up true wake up
bool estado, estadoAnt = false;
int a0,a1,a2,a3,a4,a5,a6;
                ///Sensor 0\\\
int fsrPin0  0 // the FSR and 10K pulldown are connected to a0
int fsrReading0; // the analog reading from the FSR resistor divider
int fsrVoltage0; // the analog reading converted to voltage
                 ///Sensor 1\\\
int fsrPin1 = 1; // the FSR and 10K pulldown are connected to a1
int fsrReading1; // the analog reading from the FSR resistor divider
int fsrVoltage1; // the analog reading converted to voltage
               ///Sensor 2\\\
int fsrPin2  2 // the FSR and 10K pulldown are connected to a0
int fsrReading2; // the analog reading from the FSR resistor divider
int fsrVoltage2; // the analog reading converted to voltage
                 ///Sensor 3\\\
int fsrPin3= 3; // the FSR and 10K pulldown are connected to a1
int fsrReading3; // the analog reading from the FSR resistor divider
int fsrVoltage3; // the analog reading converted to voltage
               ///Sensor 4\\\
int fsrPin4 4 // the FSR and 10K pulldown are connected to a0
int fsrReading4; // the analog reading from the FSR resistor divider
```

```
int fsrVoltage4; // the analog reading converted to voltage
                ///Sensor 5\\\
int fsrPin5= 5; // the FSR and 10K pulldown are connected to a1
int fsrReading5; // the analog reading from the FSR resistor divider
int fsrVoltage5; // the analog reading converted to voltage
                ///Sensor 6\\\
int fsrPin6= 6; // the FSR and 10K pulldown are connected to a1
int fsrReading6; // the analog reading from the FSR resistor divider
int fsrVoltage6; // the analog reading converted to voltage


void setup() {
  // put your setup code here, to rA0un once:
  Serial.begin(9600); //9600 baud Serial Comuniacation
  SoftSerial.begin(9600); // 9600 baud Software Comuniacation
  digitalWrite(LED_BUILTIN,OUTPUT); // builtin led off, save battery
   pinMode(pinLed,OUTPUT);
  pinMode(wakeUpPin,INPUT_PULLUP);
  sleep_enable();//Enabling sleep mode
  set_sleep_mode(SLEEP_MODE_PWR_DOWN);//Setting the sleep mode
  digitalWrite(pinLed,HIGH);
  digitalWrite(LED_BUILTIN,HIGH);
  //attachInterrupt(1, Wake_Function, RISING); // when the button is press go to
Wake_Function
     /*            | |     \ Define the mode to trigger the interrupt  LOW to trigger the interrupt
whenever the pin is low,
                  | |                                    CHANGE to trigger the interrupt whenever
the pin changes value
                  | |                                   RISING to trigger when the pin goes from
low to high,
                  | |                                    FALLING for when the pin goes from high
to low.
                  |    Function called when the interrupt is called
```

```
                    \
                      interrupt pin 1,D3  */
}

void loop() {
 // put your main code here, to run repeatedly:
  estado = digitalRead(wakeUpPin);
    if (estado != estadoAnt){
        flag = !flag;
    estadoAnt = estado;
    // Allow wake up pin to trigger interrupt on low.
    if (flag) {  // if flag = HIGH go to sleep
        Sleep_Function();
    }
    else {
        // Disable external pin interrupt on wake up pin.
        detachInterrupt(1);
    }
    //  do what you must do here
    }
    //Serial.println("Wake");
    digitalWrite(pinLed,HIGH);
    digitalWrite(LED_BUILTIN,HIGH);
    Read_Value();
    delay(1000); // <---press de button for 1 sec to put to sleep arduino
  }
void Sleep_Function(){
   attachInterrupt(1, Wake_Function, RISING);
         // Wake up when wake up pin is low.
          /*Serial.println("Sleeping");
```

```
        Serial.println("--------------------");
        SoftSerial.println("Sleeping");
        SoftSerial.println("--------------------");
        */
        digitalWrite(pinLed,LOW);
        digitalWrite(LED_BUILTIN,LOW);
        delay(1000);
        sleep_cpu();
 }
// here the interrupt is handled after wakeup
void Wake_Function(){
// execute code here after wake-up before returning to the loop() function
  // timers and code using timers (serial.print and more...) will not work here.
  // we don't really need to execute any special functions here, since we
  // just want the thing to wake up
   }

void Read_Value (){
    // Average 5 analogRead
    for (int i=0;i<5;i++){
    a0 += analogRead(A0);
    a1 +=  analogRead(A1);
    a2 += analogRead(A2);
    a3 +=  analogRead(A3);
    a4 += analogRead(A4);
    a5 +=  analogRead(A5);
    a6 +=  analogRead(A6);
         }
  fsrReading0=a0/5;
  fsrReading1 =a1/5;
```

```
  fsrReading2=a2/5;
  fsrReading3 =a3/5;
  fsrReading4=a4/5;
  fsrReading5 =a5/5;
  fsrReading6=a6/5;
  a0=0;
  a1=0;
  a2=0;
  a3=0;
  a4=0;
  a5=0;
  a6=0;
  fsrVoltage0 = map(fsrReading0, 0, 1023, 0, 5000);
  fsrVoltage1 = map(fsrReading1, 0, 1023, 0, 5000);
  fsrVoltage2 = map(fsrReading2, 0, 1023, 0, 5000);
  fsrVoltage3 = map(fsrReading3, 0, 1023, 0, 5000);
  fsrVoltage4 = map(fsrReading4, 0, 1023, 0, 5000);
  fsrVoltage5= map(fsrReading5, 0, 1023, 0, 5000);
  fsrVoltage6 = map(fsrReading6, 0, 1023, 0, 5000);
// send to the serial monitor
 Serial.print("9999");
 Serial.print("-");
 Serial.print(fsrVoltage0);
 Serial.print("-");
 Serial.print(fsrVoltage1);
 Serial.print("-");
 Serial.print(fsrVoltage2);
 Serial.print("-");
 Serial.print(fsrVoltage3);
 Serial.print("-");
```

```
 Serial.print(fsrVoltage4);
 Serial.print("-");
 Serial.print(fsrVoltage5);
 Serial.print("-");
 Serial.print(fsrVoltage6);
 Serial.print("-");
 Serial.print("9999");
 Serial.print("-");
//send to the bluetooth module
 SoftSerial.print("0000");
 SoftSerial.print("-");
 SoftSerial.print(fsrVoltage0);
 SoftSerial.print("-");
 SoftSerial.print(fsrVoltage1);
 SoftSerial.print("-");
SoftSerial.print(fsrVoltage2);
 SoftSerial.print("-");
 SoftSerial.print(fsrVoltage3);
 SoftSerial.print("-");
 SoftSerial.print(fsrVoltage4);
 SoftSerial.print("-");
 SoftSerial.print(fsrVoltage5);
 SoftSerial.print("-");
 SoftSerial.print(fsrVoltage6);
 SoftSerial.print("-");
 SoftSerial.print("9999");
 SoftSerial.print("-");
}
```

Appendice B, Codici Java.

B.1 Codice Finestra Nuovo Paziente.

```
// Create the directory: CarpetaPacientes


/**
 * @author Luca Farci
 * @version 1
 */
package v_ident_paciente.views;

import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.border.EmptyBorder;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JTextField;
import javax.swing.JButton;
import java.awt.event.ActionListener;
import java.io.*;
import java.awt.event.ActionEvent;
import java.awt.Toolkit;
import java.awt.Window;

public class Ventana_Nuevo_Paciente extends JFrame {

	private JPanel contentPane;
	private JTextField txtFechaN;
	private JTextField txtCodigoF;
	private JTextField txtApellido;
	private JTextField txtCorreoE;
	private JTextField txtMovil;
	private JTextField txtNombre;
	private JButton btnRegistra;
	private JButton btnVolver;
	protected Window frame;

	/**
	 * Launch the application.
	 */

	/**
	 * Create the frame.
	 */
	public Ventana_Nuevo_Paciente() {
		setIconImage(Toolkit.getDefaultToolkit()
```

```
        .getImage(Ventana_Nuevo_Paciente.class.getResource("/v_ident_paciente/common/images.jpg")));
                initComponents();
                createEvents();
        }

        private void initComponents() {

                setTitle("Ventana Nuevo Paciente ");
                setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                setBounds(100, 100, 842, 428);
                contentPane = new JPanel();
                contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
                setContentPane(contentPane);
                contentPane.setLayout(null);

                // 40 pixel distance, all the same size
                JLabel lblNombre = new JLabel("Nombre: ");
                lblNombre.setBounds(10, 25, 141, 28);
                lblNombre.setFont(new Font("", Font.PLAIN, 14));
                contentPane.add(lblNombre);

                JLabel lblApellido = new JLabel("Apellido:");
                lblApellido.setBounds(10, 65, 141, 28);
                lblApellido.setFont(new Font("", Font.PLAIN, 14));
                contentPane.add(lblApellido);

                JLabel lblCodigoFiscal = new JLabel("C\u00F3digo Fiscal: ");
                lblCodigoFiscal.setBounds(10, 105, 141, 28);
                lblCodigoFiscal.setFont(new Font("", Font.PLAIN, 14));
                contentPane.add(lblCodigoFiscal);

                JLabel lblFechaDeNascimiento = new JLabel("Fecha de nacimiento: ");
                lblFechaDeNascimiento.setBounds(10, 145, 159, 28);
                lblFechaDeNascimiento.setFont(new Font("", Font.PLAIN, 14));
                contentPane.add(lblFechaDeNascimiento);

                JLabel lblCorreoElectronico = new JLabel("Correo electr\u00F3nico:");
                lblCorreoElectronico.setBounds(10, 185, 141, 28);
                lblCorreoElectronico.setFont(new Font("", Font.PLAIN, 14));
                contentPane.add(lblCorreoElectronico);

                JLabel lblMovil = new JLabel("N\u00FAmero de tel\u00E9fono:");
                lblMovil.setBounds(10, 225, 141, 28);
                lblMovil.setFont(new Font("", Font.PLAIN, 14));
                contentPane.add(lblMovil);

                txtNombre = new JTextField();
                txtNombre.setBounds(181, 25, 141, 28);
```

```
        txtNombre.setFont(new Font("", Font.PLAIN, 14));
        contentPane.add(txtNombre);
        txtNombre.setColumns(10);

        txtApellido = new JTextField();
        txtApellido.setBounds(181, 65, 141, 28);
        txtApellido.setFont(new Font("", Font.PLAIN, 14));
        contentPane.add(txtApellido);
        txtApellido.setColumns(10);

        txtCodigoF = new JTextField();
        txtCodigoF.setBounds(181, 105, 141, 28);
        txtCodigoF.setFont(new Font("", Font.PLAIN, 14));
        contentPane.add(txtCodigoF);
        txtCodigoF.setColumns(10);

        txtFechaN = new JTextField();
        txtFechaN.setBounds(181, 145, 141, 28);
        txtFechaN.setFont(new Font("", Font.PLAIN, 14));
        contentPane.add(txtFechaN);
        txtFechaN.setColumns(10);

        txtCorreoE = new JTextField();
        txtCorreoE.setBounds(181, 185, 141, 28);
        txtCorreoE.setFont(new Font("", Font.PLAIN, 14));
        contentPane.add(txtCorreoE);
        txtCorreoE.setColumns(10);

        txtMovil = new JTextField();
        txtMovil.setBounds(181, 225, 141, 28);
        txtMovil.setFont(new Font("", Font.PLAIN, 14));
        contentPane.add(txtMovil);
        txtMovil.setColumns(10);

        btnRegistra = new JButton("Registrar");
        btnRegistra.setBounds(165, 272, 141, 28);
        btnRegistra.setFont(new Font("", Font.PLAIN, 14));
        contentPane.add(btnRegistra);

        btnVolver = new JButton("Volver");
        btnVolver.setBounds(352, 272, 159, 28);
        btnVolver.setFont(new Font("", Font.PLAIN, 14));
        contentPane.add(btnVolver);

}

private void createEvents() {

        // btn Volver go to Confirma and then to the other window
        // Ventana_Identificacion_Paciente
```

```
		btnVolver.addActionListener(new ActionListener() {
			public void actionPerformed(ActionEvent arg0) {
				Object[] options = { "Sí", "No", "Cancelar" };
				int n = JOptionPane.showOptionDialog(frame, "¿Quiere volver a la ventana precedente?",
						"¿Está seguro de continuar?", JOptionPane.YES_NO_CANCEL_OPTION, JOptionPane.QUESTION_MESSAGE,
						null, options, options[2]);
				if (n == JOptionPane.YES_OPTION) {
					System.out.println("Clicked Yes");
					Ventana_Identificacion_Paciente vip = new Ventana_Identificacion_Paciente();
					vip.setVisible(true);
					dispose();
				}
			}
		});

		// Btn Registra--> Save Data in a file
		btnRegistra.addActionListener(new ActionListener() {
			public void actionPerformed(ActionEvent arg0) {
				String Npaciente; // Nombre
				String Apaciente; // Apellido
				String CFpaciente; // codigo fiscal
				String FNpaciente; // fecha Nacimiento
				String CEpaciente;// correo electronico
				String Mpaciente; // Movil
				Npaciente = txtNombre.getText();
				Apaciente = txtApellido.getText();
				CFpaciente = txtCodigoF.getText();
				FNpaciente = txtFechaN.getText();
				CEpaciente = txtCorreoE.getText();
				Mpaciente = txtMovil.getText();
				BufferedReader br = null;
				String curline = null;
				// if one of the txtbox it 's not write
				if (Npaciente.equals("") | Apaciente.equals("") | CFpaciente.equals("") | FNpaciente.equals("")
						| CEpaciente.equals("") | Mpaciente.equals(""))
					JOptionPane.showMessageDialog(null,
							"Todos los campos son obligatorios. Pulsar la tecla Volver si se quiere volver a la Ventana Identificaciòn Paciente. ");

				// if everything it's complete save paciente--> write-Save file
				else {
					SaveFile(Npaciente, Apaciente, CFpaciente, FNpaciente, CEpaciente, Mpaciente, br, curline);
				}
			}
```

```
                });// end Registra
        }

        // Function Write-Save File\\
        public void SaveFile(String Npaciente, String Apaciente, String CFpaciente, String
FNpaciente, String CEpaciente,
                        String Mpaciente, BufferedReader br, String curline) {
                createEvents();
                FileWriter fichero = null;
                try {
                        String Path = "c:\\\\Users\\\\luca93\\\\Desktop\\\\CarpetaPacientes\\\\";
                        String directoryName = Path + "Carpeta_" + Npaciente + "_" +
Apaciente;
                        File file = (new File(directoryName + "/" + Npaciente + Apaciente +
".txt"));
                        file.getParentFile().mkdirs(); // create automatically a directory
                        fichero = new FileWriter(file);
                        fichero.write(String.format(""));
                        fichero.write(System.lineSeparator()); // new line
                        fichero.write(String.format("Nombre Paciente: %s", Npaciente));
                        fichero.write(System.lineSeparator()); // new line
                        fichero.write(String.format("Apellido Paciente: %s", Apaciente));
                        fichero.write(System.lineSeparator()); // new line
                        fichero.write(String.format("Codigo Fiscal Paciente: %s",
CFpaciente));
                        fichero.write(System.lineSeparator()); // new line
                        fichero.write(String.format("Fecha Nacimiento Paciente: %s",
FNpaciente));
                        fichero.write(System.lineSeparator()); // new line
                        fichero.write(String.format("Correo Electronico Paciente: %s",
CEpaciente));
                        fichero.write(System.lineSeparator()); // new line
                        fichero.write(String.format("Movil Paciente: %s", Mpaciente));
                        fichero.write(System.lineSeparator()); // new line
                        fichero.close();

                        new Ventana_Resumen(Npaciente, Apaciente, CFpaciente, FNpaciente,
CEpaciente, Mpaciente).setVisible(true);
                        dispose();
                } catch (Exception e) {
                        e.printStackTrace();
                }
        };

} // end
```

B.2 Codice Finestra Identificazione Paziente.

```
/**
 * @author Luca Farci
 * @version 1
 * Date 18/12/2018
 */
package v_ident_paciente.views;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.border.EmptyBorder;
import javax.swing.JButton;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JTextField;
import java.io.BufferedReader;
import java.io.File;
import java.io.FileReader;
import java.io.IOException;
import java.util.Scanner;
import javax.swing.UIManager;
import java.awt.Toolkit;

public class Ventana_Identificacion_Paciente extends JFrame {

	private JPanel contentPane;
	private JButton btnBusca;
	private JButton btnAñadirNuevoPaciente;
	private JTextField txtCodigoF;
	private JTextField txtApellido;
	private JTextField txtNombre;
	String[] array = new String[1000];
	String[] arrOfStr = new String[2];

	/**
	 * Launch the application.
	 */

	public static void main(String[] args) {
		try {

	UIManager.setLookAndFeel("javax.swing.plaf.nimbus.NimbusLookAndFeel");
		} catch (Throwable e) {
			e.printStackTrace();
		}
		EventQueue.invokeLater(new Runnable() {
			public void run() {
```

```
                        try {
                                Ventana_Identificacion_Paciente frame = new
Ventana_Identificacion_Paciente();
                                frame.setVisible(true);
                        } catch (Exception e) {
                                e.printStackTrace();
                        }
                }
        });

    }

    /**
     * Create the frame.
     */

    public Ventana_Identificacion_Paciente() {
        setIconImage(Toolkit.getDefaultToolkit()

    .getImage(Ventana_Identificacion_Paciente.class.getResource("/v_ident_paciente/co
mmon/images.jpg")));
        initComponents();
        createEvents();
    }

    // this method contains all the code for creating end initializing components
    private void initComponents() {

        setTitle("Ventana Identificación Paciente");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setBounds(100, 100, 896, 316);
        contentPane = new JPanel();
        contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
        setContentPane(contentPane);
        contentPane.setLayout(null);

        JLabel lblNombre = new JLabel("Nombre: ");
        lblNombre.setBounds(10, 25, 141, 28);
        lblNombre.setFont(new Font("", Font.PLAIN, 14));
        contentPane.add(lblNombre);

        JLabel lblApellido = new JLabel("Apellido:");
        lblApellido.setBounds(10, 65, 141, 28);
        lblApellido.setFont(new Font("", Font.PLAIN, 14));
        contentPane.add(lblApellido);

        JLabel lblCodigoFiscal = new JLabel("C\u00F3digo Fiscal: ");
        lblCodigoFiscal.setBounds(10, 105, 141, 28);
        lblCodigoFiscal.setFont(new Font("", Font.PLAIN, 14));
        contentPane.add(lblCodigoFiscal);
```

```
		txtNombre = new JTextField();
		txtNombre.setBounds(181, 25, 141, 28);
		contentPane.add(txtNombre);
		txtNombre.setFont(new Font("", Font.PLAIN, 14));
		txtNombre.setColumns(10);

		txtApellido = new JTextField();
		txtApellido.setBounds(181, 65, 141, 28);
		txtApellido.setFont(new Font("", Font.PLAIN, 14));
		contentPane.add(txtApellido);
		txtApellido.setColumns(10);

		txtCodigoF = new JTextField();
		txtCodigoF.setBounds(181, 105, 141, 28);
		txtCodigoF.setFont(new Font("", Font.PLAIN, 14));
		contentPane.add(txtCodigoF);
		txtCodigoF.setColumns(10);

		btnBusca = new JButton("Buscar");
		btnBusca.setBounds(66, 171, 256, 28);
		btnBusca.setFont(new Font("", Font.PLAIN, 14));
		contentPane.add(btnBusca);

		btnAñadirNuevoPaciente = new JButton("A\u00F1adir Nuevo Paciente ");
		btnAñadirNuevoPaciente.setBounds(66, 207, 256, 28);
		btnAñadirNuevoPaciente.setFont(new Font("", Font.PLAIN, 14));
		contentPane.add(btnAñadirNuevoPaciente);
	}

	// this method contains all the code for creating events
	private void createEvents() {

		// btn Añadir nuevo paciente-> ventana Nuevo paciente
		btnAñadirNuevoPaciente.addActionListener(new ActionListener() {
			public void actionPerformed(ActionEvent e) {
				Ventana_Nuevo_Paciente vnp = new
Ventana_Nuevo_Paciente();
				vnp.setVisible(true);
				dispose();
			}
		});

		// find Paciente
		btnBusca.addActionListener(new ActionListener() {
			public void actionPerformed(ActionEvent e) {
				String Npaciente; // Nombre
				String Apaciente; // Apellido
				//String CFpaciente; // codigo fiscal
				Npaciente = txtNombre.getText();
```

```
                    Apaciente = txtApellido.getText();
                    //CFpaciente = txtCodigoF.getText();
                    BufferedReader br;
                    String curline;
                    String parentDirectory =
("c:\\Users\\luca93\\Desktop\\CarpetaPacientes\\");
                    String fileExtension = ".txt";
                    String directoryName = parentDirectory + "Carpeta_" +
Npaciente + "_" + Apaciente;
                    File parentDir = new File(directoryName);
                    String[] listOfTextFiles = parentDir.list();
                    // Put the names of all files ending with .txt in a String array

                    int i = 0;
                    if (listOfTextFiles != null) {
                        for (String f : listOfTextFiles) {
                            String fichero = Npaciente +
Apaciente.concat(fileExtension);
                            if (fichero.equals(f)) {
                                i = 1 + i;
                                System.out.println(f);
                            }
                        }
                        if (i == 0) { // paciente no found
                            JOptionPane.showMessageDialog(null,
                                    "Paciente no encontrado. Pulsar la
tecla Añadir Nuevo Paciente si se quiere añadir un nuevo Paciente. ");
                        } else { // found
                            try {
                                String filepath = (directoryName + "/" +
Npaciente + Apaciente + ".txt");
                                br = new BufferedReader(new
FileReader(filepath));
                                int j = 0;

                                while ((curline = br.readLine()) != null) {
                                    Scanner in = new
Scanner(curline);
                                    arrOfStr = curline.split(": ");
                                    for (String a : arrOfStr) {
                                        array[j] = a;
                                        j++;
                                    }
                                    in.close();
                                }
                                String nombre = array[2];
                                String apellido = array[4];
                                String codigofiscal = array[6];
                                String fechanascimiento = array[8];
                                String correoelectronico = array[10];
```

```
                                        String movil = array[12];
                                        new Ventana_Resumen(nombre, apellido,
codigofiscal, fechanascimiento, correoelectronico,
                                                movil).setVisible(true);
                                        dispose();
                                        br.close();
                                    } catch (IOException e1) {
                                        e1.printStackTrace();
                                    }
                                }
                            } else {
                                JOptionPane.showMessageDialog(null,
                                        "Paciente no encontrado. Pulsar la tecla
Añadir Nuevo Paciente si se quiere añadir un nuevo Paciente ");
                            }
                        }
                    }); // end bnt Busca
            }// end create events
} // Ventana_Identificacion_Paciente
```

B.3 Codice Finestra Riassuntiva.

```
/**
 * @author Luca Farci
 * @version 1
 * Date 18/12/2018
 */

package v_ident_paciente.views;

import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.border.EmptyBorder;
import javax.swing.JLabel;
import javax.swing.JButton;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import java.awt.Toolkit;

public class Ventana_Resumen extends JFrame {

        private JPanel contentPane;
        private JButton btnSeguir;
        private JButton btnVolver;
```

```
	private JLabel lblNombre;
	private JLabel lblApellido;
	private JLabel lblCodigoFiscal;
	private JLabel lblFechaDeNascimiento;
	private JLabel lblCorreoElectronico;
	private JLabel lblMovil;

	/**
	 * Launch the application.
	 */
	/**
	 * Create the frame.
	 */
	public Ventana_Resumen(String npaciente, String apaciente, String codigofiscal,
String fechanascimiento,
			String correoelectronico, String movil) {
		setIconImage(Toolkit.getDefaultToolkit()

	.getImage(Ventana_Resumen.class.getResource("/v_ident_paciente/common/images.j
pg")));
		setTitle("Resumen");
		initComponents();
		createEvents();
		// set the fields with the parameters founded in the file (read in Ventana
		// Indentificacion Paciente)
		lblNombre.setText(npaciente);
		lblApellido.setText(apaciente);
		lblCodigoFiscal.setText(codigofiscal);
		lblFechaDeNascimiento.setText(fechanascimiento);
		lblCorreoElectronico.setText(correoelectronico);
		lblMovil.setText(movil);
	}

	// this method contains all the code for creating end initializing components
	private void initComponents() {

		setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
		setBounds(100, 100, 573, 398);
		contentPane = new JPanel();
		contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
		setContentPane(contentPane);
		contentPane.setLayout(null);

		btnSeguir = new JButton("Seguir");
		btnSeguir.setFont(new Font("", Font.PLAIN, 14));
		btnSeguir.setBounds(225, 298, 136, 28);
		contentPane.add(btnSeguir);

		btnVolver = new JButton("Volver");
		btnVolver.setFont(new Font("", Font.PLAIN, 14));
```

```
btnVolver.setBounds(407, 298, 136, 28);
contentPane.add(btnVolver);
JLabel label = new JLabel("Nombre: ");
label.setFont(new Font("Dialog", Font.PLAIN, 14));
label.setBounds(6, 25, 141, 28);
contentPane.add(label);

JLabel label_1 = new JLabel("Apellido:");
label_1.setFont(new Font("Dialog", Font.PLAIN, 14));
label_1.setBounds(6, 65, 141, 28);
contentPane.add(label_1);

JLabel lblCdigoFiscal = new JLabel("C\u00F3digo Fiscal: ");
lblCdigoFiscal.setFont(new Font("Dialog", Font.PLAIN, 14));
lblCdigoFiscal.setBounds(6, 112, 141, 28);
contentPane.add(lblCdigoFiscal);

JLabel lblFechaDeNacimiento = new JLabel("Fecha de nacimiento: ");
lblFechaDeNacimiento.setFont(new Font("Dialog", Font.PLAIN, 14));
lblFechaDeNacimiento.setBounds(6, 152, 159, 28);
contentPane.add(lblFechaDeNacimiento);

JLabel lblCorreoElectrnico = new JLabel("Correo electr\u00F3nico:");
lblCorreoElectrnico.setFont(new Font("Dialog", Font.PLAIN, 14));
lblCorreoElectrnico.setBounds(6, 192, 141, 28);
contentPane.add(lblCorreoElectrnico);

JLabel lblNmeroDeTelfono = new JLabel("N\u00FAmero de tel\u00E9fono:");
lblNmeroDeTelfono.setFont(new Font("Dialog", Font.PLAIN, 14));
lblNmeroDeTelfono.setBounds(6, 232, 141, 28);
contentPane.add(lblNmeroDeTelfono);

lblNombre = new JLabel("Nombre: ");
lblNombre.setBounds(208, 25, 343, 28);
lblNombre.setFont(new Font("", Font.PLAIN, 14));
contentPane.add(lblNombre);

lblApellido = new JLabel("Apellido:");
lblApellido.setBounds(208, 65, 343, 28);
lblApellido.setFont(new Font("", Font.PLAIN, 14));
contentPane.add(lblApellido);

lblCodigoFiscal = new JLabel("Codigo Fiscal: ");
lblCodigoFiscal.setBounds(208, 105, 343, 28);
lblCodigoFiscal.setFont(new Font("", Font.PLAIN, 14));
contentPane.add(lblCodigoFiscal);

lblFechaDeNascimiento = new JLabel("Fecha de nascimiento: ");
lblFechaDeNascimiento.setBounds(208, 145, 343, 28);
lblFechaDeNascimiento.setFont(new Font("", Font.PLAIN, 14));
```

```
                contentPane.add(lblFechaDeNascimiento);

                lblCorreoElectronico = new JLabel("Correo electronico:");
                lblCorreoElectronico.setBounds(208, 185, 343, 28);
                lblCorreoElectronico.setFont(new Font("", Font.PLAIN, 14));
                contentPane.add(lblCorreoElectronico);

                lblMovil = new JLabel("Movil:");
                lblMovil.setBounds(208, 225, 343, 28);
                lblMovil.setFont(new Font("", Font.PLAIN, 14));
                contentPane.add(lblMovil);
        }

        // this method contains all the code for creating events
        private void createEvents() {

                // go to Ventana Paciente Registrado
                btnSeguir.addActionListener(new ActionListener() {
                        public void actionPerformed(ActionEvent e) {
                                String Npaciente = lblNombre.getText();
                                String Apaciente = lblApellido.getText();
                                // String CFpaciente = lblCodigoFiscal.getText();
                                Ventana_Paciente_Registrado vpr = new
Ventana_Paciente_Registrado(Npaciente, Apaciente);
                                vpr.setVisible(true);
                                dispose();
                        }
                }); // end btnSeguir

                btnVolver.addActionListener(new ActionListener() {
                        public void actionPerformed(ActionEvent e) {
                                Ventana_Identificacion_Paciente vip = new
Ventana_Identificacion_Paciente();
                                vip.setVisible(true);
                                dispose();
                        }
                });

        } // end btnVolver
}// end
```

B.4 Codice Finestra Paziente Registrata.

```
package v_ident_paciente.views;


import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JPanel;
```

```
import javax.swing.border.EmptyBorder;
import arduino.*;
import javax.swing.JLabel;
import javax.swing.ImageIcon;
import java.awt.Window;
import javax.swing.JButton;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import java.util.Date;
import java.io.BufferedReader;
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.text.SimpleDateFormat;
import java.awt.Color;
import java.awt.Dimension;
import javax.swing.JScrollPane;
import javax.swing.JMenuBar;
import javax.swing.JMenu;
import javax.swing.JMenuItem;
import javax.swing.JOptionPane;
import java.awt.Toolkit;
import javax.swing.event.MenuListener;
import javax.swing.event.MenuEvent;
import javax.swing.JSeparator;
import javax.swing.JTextArea;

/**
 * @author luca93
 *
 */
public class Ventana_Paciente_Registrado extends JFrame {

	private static int nmeasure = 0;
	private JButton btnVolver;
	protected Window frame;
	private JLabel lblDefaultNombre;
	private JLabel lblDefaultAppellido;
	private  JButton btnEmpezarAcquisicion;
	private static int baud_rate=9600;
	public static  Arduino arduino;
	private JButton btnTerminarAcquisicion;
	public static int inizio=0;
	private static JPanel panel_1L;
	private static JPanel panel_2L;
	private static JPanel panel_3L;
	private static JPanel panel_4L;
	private static JPanel panel_5L;
	private static JPanel panel_6L;
	private static JPanel panel_7L;
	private static JPanel panel_1R;
```

```
    private static JPanel panel_2R;
    private static JPanel panel_3R;
    private static JPanel panel_4R;
    private static JPanel panel_5R;
    private static JPanel panel_6R;
    private static JPanel panel_7R;
    private JButton btnGuardarTest;
    private JLabel lblNombre;
    private JLabel lblApellido;
    protected int var;
    static Color Colores[]=new Color[44];
    Thread t;
    String date = new Date().toString();
    private static String a;
    private static String [] parts=new String[20];
    private static double x1;
    private static int c;
    private static int j=0;
    private static String[] b= new String [10000];
    private static int i=0;
    private JMenuItem mntmInprimirExamen;
    private static PortDropdownMenu portList1;
    private JMenuItem mntmAnagrafica;
    private JMenuItem mntmGrabaciones;
    String[] array = new String[1000];
    String[] arrOfStr = new String[2];
    private JMenu mnArchivoGrabaciones;


    /**
     * Launch the application.
     */
    /**
     * Create the frame.
     */
    public Ventana_Paciente_Registrado(String nombre, String
apellido) {


    setIconImage(Toolkit.getDefaultToolkit().getImage(Ventana_
Paciente_Registrado.class.getResource("/v_ident_paciente/commo
n/images.jpg")));
          setVisible(true);
          setTitle("Ventana Paciente Registrado");
          initComponents();
          createEvents();
          lblDefaultNombre.setText(nombre);
         lblDefaultAppellido.setText(apellido);
         Runnable r =new ReadValue();
         t=new Thread(r);
         t.start();
```

```
	}

	class ReadValue implements Runnable{

		public ReadValue(){ // constructor
		}

		public void run() { // codigo corrispondente a la
tarea que queremos sea simultanea
			while(!Thread.interrupted()) {  //
				DisplayPanel();
			}
		}
	}


	//  this method contains all the code for creating end
initializing components
	private void initComponents() {

		setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
		setBounds(100, 100, 1024, 600);

		JMenuBar menuBar_1 = new JMenuBar();
		setJMenuBar(menuBar_1);

		JMenu mnFile = new JMenu("File");
		menuBar_1.add(mnFile);
		mnFile.setFont(new Font("", Font.PLAIN, 14));
	// dentro FILE
		mntmAadirComentario = new JMenuItem("A\u00F1adir
Comentario");
		mnFile.add(mntmAadirComentario);
		mntmAadirComentario.setFont(new Font("", Font.PLAIN,
14));

		mntmInprimirExamen = new JMenuItem("Inprimir
Examen");
		mnFile.add(mntmInprimirExamen);
		mntmInprimirExamen.setFont(new Font("", Font.PLAIN,
14));

		JMenuItem mntmGuardar = new JMenuItem("Guardar ");
		mnFile.add(mntmGuardar);
		mntmGuardar.setFont(new Font("", Font.PLAIN, 14));

		JMenuItem mntmSalir = new JMenuItem("Salir ");
		mnFile.add(mntmSalir);
		mntmSalir.setFont(new Font("", Font.PLAIN, 14));
```

```
// DENTRO Info Paciente
          JMenu mnInfoPaciente = new JMenu("Info Paciente");
          menuBar_1.add(mnInfoPaciente);
          mnInfoPaciente.setFont(new Font("", Font.PLAIN, 14));

          mntmAnagrafica = new JMenuItem("Anagr\u00E1fica ");
          mnInfoPaciente.add(mntmAnagrafica);
          mntmAnagrafica.setFont(new Font("", Font.PLAIN, 14));

          JMenuItem mntmHistoriaClinica = new
JMenuItem("Historia Cl\u00EDnica ");
          mnInfoPaciente.add(mntmHistoriaClinica);
          mntmHistoriaClinica.setFont(new Font("", Font.PLAIN,
14));

// DENTRO Archivo Grabaciones
          mnArchivoGrabaciones = new JMenu("Archivo Grabaciones
");
          menuBar_1.add(mnArchivoGrabaciones);
          mnArchivoGrabaciones.setFont(new Font("", Font.PLAIN,
14));
          JPanel contentPane = new JPanel();
          contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
          setContentPane(contentPane);
//

          lblDefaultNombre= new JLabel("Default Nombre");
          lblDefaultNombre.setBounds(126, 11, 175, 14);
          lblDefaultNombre.setFont(new Font("", Font.PLAIN,
14));

          lblDefaultAppellido = new JLabel("Default
Appellido");
          lblDefaultAppellido.setBounds(382, 11, 153, 14);
          lblDefaultAppellido.setFont(new Font("", Font.PLAIN,
14));


          JLabel lblFecha = new JLabel("Fecha :");
          lblFecha.setBounds(667, 11, 143, 14);
          lblFecha.setFont(new Font("", Font.PLAIN, 14));


          JLabel lblDefaultFecha = new JLabel(date);
          lblDefaultFecha.setBounds(801, 11, 400, 14);
          lblDefaultFecha.setFont(new Font("", Font.PLAIN,
14));

          lblNombre = new JLabel("Nombre:");
          lblNombre.setBounds(6, 11, 55, 16);
          lblNombre.setFont(new Font("", Font.PLAIN, 14));
```

```
            lblApellido = new JLabel("Apellido:");
            lblApellido.setBounds(315, 11, 55, 16);
            lblApellido.setFont(new Font("", Font.PLAIN, 14));

            JPanel panelbutton = new JPanel();
            panelbutton.setBounds(635, 451, 363, 87);

            btnEmpezarAcquisicion = new JButton("Empezar
Adquisici\u00F3n");
            btnEmpezarAcquisicion.setBounds(0, 11, 174, 23);
            btnEmpezarAcquisicion.setFont(new Font("",
Font.PLAIN, 14));

            btnTerminarAcquisicion = new JButton("Terminar
Adquisici\u00F3n ");
            btnTerminarAcquisicion.setBounds(186, 11, 177, 23);
            btnTerminarAcquisicion.setFont(new Font("",
Font.PLAIN, 14));

            btnGuardarTest = new JButton("Guardar Test ");
            btnGuardarTest.setBounds(0, 45, 213, 23);
            btnGuardarTest.setFont(new Font("", Font.PLAIN, 14));

            btnVolver = new JButton("Volver ");
            btnVolver.setBounds(254, 45, 96, 23);
            btnVolver.setFont(new Font("", Font.PLAIN, 14));

// ScrollPane 10 Images
            DeclaracionColeres();

            // Panel Image
            JPanel panel = new JPanel();
            panel.setBounds(10, 24, 492, 519);

            // RIGHT
            // sensor 1 R
            panel_1R = new JPanel();
            panel_1R.setBounds(321, 380, 35, 37);
            panel_1R.setBackground(Colores[0]);
            // sensor 2 R
            panel_2R = new JPanel();
            panel_2R.setBounds(346, 429, 35, 37);
            panel_2R.setBackground(Colores[0]);
            // sensor 3 R
            panel_3R = new JPanel();
            panel_3R.setBounds(379, 368, 35, 37);
            panel_3R.setBackground(Colores[0]);
            // sensor 4R
            panel_4R = new JPanel();
            panel_4R.setBounds(398, 262, 35, 37);
```

```
            panel_4R.setBackground(Colores[0]);
            // sensor 5R
            panel_5R = new JPanel();
            panel_5R.setBounds(398, 181, 35, 37);
            panel_5R.setBackground(Colores[0]);
            // sensor 6R
            panel_6R = new JPanel();
            panel_6R.setBounds(351, 151, 35, 37);
            panel_6R.setBackground(Colores[0]);
            // sensor 7R
            panel_7R = new JPanel();
            panel_7R.setBounds(304, 139, 35, 37);
            panel_7R.setBackground(Colores[0]);
            // LEFT
            // sensor 1 L
            panel_1L = new JPanel();
            panel_1L.setBounds(150, 368, 35, 37);
            panel_1L.setBackground(Colores[0]);
            // sensor 2 L
            panel_2L = new JPanel();
            panel_2L.setBounds(121, 415, 35, 37);
            panel_2L.setBackground(Colores[0]);
            // sensor 3 L
            panel_3L = new JPanel();
            panel_3L.setBounds(97, 356, 35, 37);
            panel_3L.setBackground(Colores[0]);
            // sensor 4L
            panel_4L = new JPanel();
            panel_4L.setBounds(76, 262, 35, 37);
            panel_4L.setBackground(Colores[0]);
            // sensor 5L
            panel_5L = new JPanel();
            panel_5L.setBounds(66, 171, 35, 37);
            panel_5L.setBackground(Colores[0]);
            // sensor 6L
            panel_6L = new JPanel();
            panel_6L.setBounds(121, 139, 35, 37);
            panel_6L.setBackground(Colores[0]);
            // sensor 7L
            panel_7L = new JPanel();
            panel_7L.setBounds(169, 129, 35, 37);
            panel_7L.setBackground(Colores[0]);

          portList1 = new PortDropdownMenu();
          portList1.setBounds(635, 412, 168, 28);
            portList1.setAutoscrolls(true);
            portList1.refreshMenu();
            portList1.setEnabled(true);

            JLabel lblSelectionarPuertaBluetooth = new
JLabel("Seleccionar puerta Bluetooth:");
```

```
        lblSelectionarPuertaBluetooth.setBounds(635, 385,
358, 16);
        lblSelectionarPuertaBluetooth.setFont(new Font("",
Font.PLAIN, 14));

                    JLabel lblNewLabel_2 = new JLabel("");
                    lblNewLabel_2.setBounds(508, 36, 91, 493);
                    lblNewLabel_2.setIcon(new
ImageIcon(Ventana_Paciente_Registrado.class.getResource("/v_id
ent_paciente/resources/Colors.jpg")));
        contentPane.setLayout(null);
        contentPane.add(lblNombre);
        contentPane.add(lblDefaultNombre);
        contentPane.add(lblApellido);
        contentPane.add(lblDefaultApellido);
        contentPane.add(lblFecha);
        contentPane.add(lblDefaultFecha);
        JPanel panelImages = new JPanel();
        panelImages.setBounds(609, 38, 1850, 186);
        panelImages.setPreferredSize(new Dimension(1850,
186));
        panelImages.setLayout(null);
        JScrollPane scrollImages=new
JScrollPane(panelImages);
        scrollImages.setBounds(635, 36, 363, 214);
        contentPane.add(scrollImages);

        // Images 1
             panel1_7 = new JPanel();
             panel1_7.setBounds(63, 44, 16, 16);
             panel1_7.setBackground(Colores[0]);
             panelImages.add(panel1_7);

             panel1_6 = new JPanel();
             panel1_6.setBounds(46, 50, 16, 16);
             panel1_6.setBackground(Colores[0]);
             panelImages.add(panel1_6);

             panel1_5 = new JPanel();
             panel1_5.setBounds(30, 62, 16, 16);
             panel1_5.setBackground(Colores[0]);
             panelImages.add(panel1_5);

             panel1_4 = new JPanel();
             panel1_4.setBounds(30, 93, 16, 16);
             panel1_4.setBackground(Colores[0]);
             panelImages.add(panel1_4);

             panel1_3 = new JPanel();
             panel1_3.setBounds(35, 131, 16, 16);
             panel1_3.setBackground(Colores[0]);
```

```
                panelImages.add(panel1_3);

                panel1_2 = new JPanel();
                panel1_2.setBounds(46, 151, 16, 16);
                panel1_2.setBackground(Colores[0]);
                panelImages.add(panel1_2);

                panel1_1 = new JPanel();
                panel1_1.setBounds(57, 131, 16, 16);
                panel1_1.setBackground(Colores[0]);
                panelImages.add(panel1_1);


                JLabel lblimage1 = new JLabel("");
                lblimage1.setIcon(new
ImageIcon(Ventana_Paciente_Registrado.class.getResource("/v_id
ent_paciente/resources/little.png")));
                lblimage1.setBounds(5, 5, 178, 178);
                panelImages.add(lblimage1);
                //   Image 2
            panel2_7 = new JPanel();
            panel2_7.setBackground((Color) null);
            panel2_7.setBounds(244, 44, 16, 16);
            panel2_7.setBackground(Colores[0]);
            panelImages.add(panel2_7);

            panel2_6 = new JPanel();
            panel2_6.setBackground((Color) null);
            panel2_6.setBounds(227, 50, 16, 16);
            panel2_6.setBackground(Colores[0]);
            panelImages.add(panel2_6);

            panel2_5 = new JPanel();
            panel2_5.setBackground((Color) null);
            panel2_5.setBounds(210, 62, 16, 16);
            panel2_5.setBackground(Colores[0]);
            panelImages.add(panel2_5);

            panel2_4 = new JPanel();
            panel2_4.setBackground((Color) null);
            panel2_4.setBounds(210, 93, 16, 16);
            panel2_4.setBackground(Colores[0]);
            panelImages.add(panel2_4);

            panel2_3 = new JPanel();
            panel2_3.setBackground((Color) null);
            panel2_3.setBounds(216, 131, 16, 16);
            panel2_3.setBackground(Colores[0]);
            panelImages.add(panel2_3);

            panel2_2 = new JPanel();
```

```
            panel2_2.setBackground((Color) null);
            panel2_2.setBounds(227, 151, 16, 16);
            panel2_2.setBackground(Colores[0]);
            panelImages.add(panel2_2);

            panel2_1 = new JPanel();
            panel2_1.setBackground((Color) null);
            panel2_1.setBounds(237, 131, 16, 16);
            panel2_1.setBackground(Colores[0]);
            panelImages.add(panel2_1);


            JLabel lbllblimage2 = new JLabel("");
            lbllblimage2.setIcon(new
ImageIcon(Ventana_Paciente_Registrado.class.getResource("/v_id
ent_paciente/resources/little.png")));
            lbllblimage2.setBounds(188, 5, 178, 178);
            panelImages.add(lbllblimage2);
            // image 3
            panel3_7 = new JPanel();
            panel3_7.setBackground(Colores[0]);
            panel3_7.setBounds(428, 44, 16, 16);
            panelImages.add(panel3_7);

            panel3_6 = new JPanel();
            panel3_6.setBackground(Colores[0]);
            panel3_6.setBounds(408, 50, 16, 16);
            panelImages.add(panel3_6);

            panel3_5 = new JPanel();
            panel3_5.setBackground(Colores[0]);
            panel3_5.setBounds(391, 62, 16, 16);
            panelImages.add(panel3_5);

            panel3_4 = new JPanel();
            panel3_4.setBackground(Colores[0]);
            panel3_4.setBounds(391, 93, 16, 16);
            panelImages.add(panel3_4);

            panel3_3 = new JPanel();
            panel3_3.setBackground(Colores[0]);
            panel3_3.setBounds(402, 131, 16, 16);
            panelImages.add(panel3_3);

            panel3_2 = new JPanel();
            panel3_2.setBackground(Colores[0]);
            panel3_2.setBounds(408, 151, 16, 16);
            panelImages.add(panel3_2);

            panel3_1 = new JPanel();
            panel3_1.setBackground(Colores[0]);
```

```
            panel3_1.setBounds(421, 131, 16, 16);
            panelImages.add(panel3_1);

            JLabel lblblimage3 = new JLabel("");
            lblblimage3.setIcon(new
ImageIcon(Ventana_Paciente_Registrado.class.getResource("/v_id
ent_paciente/resources/little.png")));
            lblblimage3.setBounds(371, 5, 178, 178);
            panelImages.add(lblblimage3);
            // image 4
            panel4_1 = new JPanel();
            panel4_1.setBackground(Colores[0]);
            panel4_1.setBounds(606, 131, 16, 16);
            panelImages.add(panel4_1);

            panel4_2 = new JPanel();
            panel4_2.setBackground(Colores[0]);
            panel4_2.setBounds(594, 151, 16, 16);
            panelImages.add(panel4_2);

            panel4_3 = new JPanel();
            panel4_3.setBackground(Colores[0]);
            panel4_3.setBounds(585, 131, 16, 16);
            panelImages.add(panel4_3);

            panel4_4 = new JPanel();
            panel4_4.setBackground(Colores[0]);
            panel4_4.setBounds(574, 93, 16, 16);
            panelImages.add(panel4_4);

            panel4_5 = new JPanel();
            panel4_5.setBackground(Colores[0]);
            panel4_5.setBounds(574, 62, 16, 16);
            panelImages.add(panel4_5);

            panel4_6 = new JPanel();
            panel4_6.setBackground(Colores[0]);
            panel4_6.setBounds(594, 50, 16, 16);
            panelImages.add(panel4_6);

            panel4_7 = new JPanel();
            panel4_7.setBackground(Colores[0]);
            panel4_7.setBounds(613, 44, 16, 16);
            panelImages.add(panel4_7);

            JLabel lbllblimage4 = new JLabel("");
            lbllblimage4.setIcon(new
ImageIcon(Ventana_Paciente_Registrado.class.getResource("/v_id
ent_paciente/resources/little.png")));
            lbllblimage4.setBounds(554, 5, 178, 178);
            panelImages.add(lbllblimage4);
```

```
            // image 5
            panel5_7 = new JPanel();
            panel5_7.setBackground(Colores[0]);
            panel5_7.setBounds(793, 44, 16, 16);
            panelImages.add(panel5_7);

            panel5_6 = new JPanel();
            panel5_6.setBackground(Colores[0]);
            panel5_6.setBounds(776, 50, 16, 16);
            panelImages.add(panel5_6);

            panel5_5 = new JPanel();
            panel5_5.setBackground(Colores[0]);
            panel5_5.setBounds(760, 62, 16, 16);
            panelImages.add(panel5_5);

            panel5_4 = new JPanel();
            panel5_4.setBackground(Colores[0]);
            panel5_4.setBounds(760, 93, 16, 16);
            panelImages.add(panel5_4);

            panel5_3 = new JPanel();
            panel5_3.setBackground(Colores[0]);
            panel5_3.setBounds(766, 131, 16, 16);
            panelImages.add(panel5_3);

            panel5_2 = new JPanel();
            panel5_2.setBackground(Colores[0]);
            panel5_2.setBounds(776, 151, 16, 16);
            panelImages.add(panel5_2);

            panel5_1 = new JPanel();
            panel5_1.setBackground(Colores[0]);
            panel5_1.setBounds(791, 131, 18, 16);
            panelImages.add(panel5_1);

            JLabel lbllblimage5 = new JLabel("");
            lbllblimage5.setIcon(new
ImageIcon(Ventana_Paciente_Registrado.class.getResource("/v_id
ent_paciente/resources/little.png")));
            lbllblimage5.setBounds(737, 5, 178, 178);
            panelImages.add(lbllblimage5);
            // image 6
            panel6_7 = new JPanel();
            panel6_7.setBackground(Colores[0]);
            panel6_7.setBounds(978, 44, 16, 16);
            panelImages.add(panel6_7);

            panel6_6 = new JPanel();
            panel6_6.setBackground(Colores[0]);
            panel6_6.setBounds(959, 50, 16, 16);
```

```
            panelImages.add(panel6_6);

            panel6_5 = new JPanel();
            panel6_5.setBackground(Colores[0]);
            panel6_5.setBounds(941, 62, 16, 16);
            panelImages.add(panel6_5);

            panel6_4 = new JPanel();
            panel6_4.setBackground(Colores[0]);
            panel6_4.setBounds(941, 93, 16, 16);
            panelImages.add(panel6_4);

            panel6_3 = new JPanel();
            panel6_3.setBackground(Colores[0]);
            panel6_3.setBounds(949, 131, 16, 16);
            panelImages.add(panel6_3);

            panel6_2 = new JPanel();
            panel6_2.setBackground(Colores[0]);
            panel6_2.setBounds(970, 131, 16, 16);
            panelImages.add(panel6_2);

            panel6_1 = new JPanel();
            panel6_1.setBackground(Colores[0]);
            panel6_1.setBounds(959, 151, 16, 16);
            panelImages.add(panel6_1);

            JLabel lbllblimage6 = new JLabel("");
            lbllblimage6.setIcon(new
ImageIcon(Ventana_Paciente_Registrado.class.getResource("/v_id
ent_paciente/resources/little.png")));
            lbllblimage6.setBounds(920, 5, 178, 178);
            panelImages.add(lbllblimage6);
            // image 7
            panel7_7 = new JPanel();
            panel7_7.setBackground(Colores[0]);
            panel7_7.setBounds(1159, 44, 16, 16);
            panelImages.add(panel7_7);

            panel7_6 = new JPanel();
            panel7_6.setBackground(Colores[0]);
            panel7_6.setBounds(1143, 50, 16, 16);
            panelImages.add(panel7_6);

            panel7_5 = new JPanel();
            panel7_5.setBackground(Colores[0]);
            panel7_5.setBounds(1126, 62, 16, 16);
            panelImages.add(panel7_5);

            panel7_4 = new JPanel();
            panel7_4.setBackground(Colores[0]);
```

```
            panel7_4.setBounds(1126, 93, 16, 16);
            panelImages.add(panel7_4);

            panel7_3 = new JPanel();
            panel7_3.setBackground(Colores[0]);
            panel7_3.setBounds(1130, 131, 16, 16);
            panelImages.add(panel7_3);

            panel7_2 = new JPanel();
            panel7_2.setBackground(Colores[0]);
            panel7_2.setBounds(1157, 131, 18, 16);
            panelImages.add(panel7_2);

            panel7_1 = new JPanel();
            panel7_1.setBackground(Colores[0]);
            panel7_1.setBounds(1143, 151, 16, 16);
            panelImages.add(panel7_1);

            JLabel lbllblimage7 = new JLabel("");
            lbllblimage7.setIcon(new
ImageIcon(Ventana_Paciente_Registrado.class.getResource("/v_id
ent_paciente/resources/little.png")));
            lbllblimage7.setBounds(1103, 5, 178, 178);
            panelImages.add(lbllblimage7);

            panel8_2 = new JPanel();
            panel8_2.setBackground(Colores[0]);
            panel8_2.setBounds(1331, 151, 16, 16);
            panelImages.add(panel8_2);

            panel8_1 = new JPanel();
            panel8_1.setBackground(Colores[0]);
            panel8_1.setBounds(1342, 131, 16, 16);
            panelImages.add(panel8_1);

            panel8_3 = new JPanel();
            panel8_3.setBackground(Colores[0]);
            panel8_3.setBounds(1316, 131, 16, 16);
            panelImages.add(panel8_3);

            panel8_4 = new JPanel();
            panel8_4.setBackground(Colores[0]);
            panel8_4.setBounds(1308, 93, 18, 16);
            panelImages.add(panel8_4);

            panel8_5 = new JPanel();
            panel8_5.setBackground(Colores[0]);
            panel8_5.setBounds(1308, 62, 16, 16);
            panelImages.add(panel8_5);

            panel8_6 = new JPanel();
```

```
            panel8_6.setBackground(Colores[0]);
            panel8_6.setBounds(1331, 50, 16, 16);
            panelImages.add(panel8_6);

            panel8_7 = new JPanel();
            panel8_7.setBackground(Colores[0]);
            panel8_7.setBounds(1346, 44, 16, 16);
            panelImages.add(panel8_7);

            JLabel lbllblimage8 = new JLabel("");
            lbllblimage8.setIcon(new
ImageIcon(Ventana_Paciente_Registrado.class.getResource("/v_id
ent_paciente/resources/little.png")));
            lbllblimage8.setBounds(1289, 5, 178, 178);
            panelImages.add(lbllblimage8);
            // image 9
            panel9_1 = new JPanel();
            panel9_1.setBackground(Colores[0]);
            panel9_1.setBounds(1519, 131, 16, 16);
            panelImages.add(panel9_1);

            panel9_4 = new JPanel();
            panel9_4.setBackground(Colores[0]);
            panel9_4.setBounds(1493, 93, 16, 16);
            panelImages.add(panel9_4);

            panel9_6 = new JPanel();
            panel9_6.setBackground(Colores[0]);
            panel9_6.setBounds(1507, 50, 16, 16);
            panelImages.add(panel9_6);

            panel9_7 = new JPanel();
            panel9_7.setBackground(Colores[0]);
            panel9_7.setBounds(1525, 44, 16, 16);
            panelImages.add(panel9_7);

            panel9_5 = new JPanel();
            panel9_5.setBackground(Colores[0]);
            panel9_5.setBounds(1493, 62, 16, 16);
            panelImages.add(panel9_5);

            panel9_3 = new JPanel();
            panel9_3.setBackground(Colores[0]);
            panel9_3.setBounds(1501, 131, 16, 16);
            panelImages.add(panel9_3);

            panel9_2 = new JPanel();
            panel9_2.setBackground(Colores[0]);
            panel9_2.setBounds(1507, 151, 16, 16);
            panelImages.add(panel9_2);
```

```
        JLabel lbllblimage9 = new JLabel("");
        lbllblimage9.setIcon(new
ImageIcon(Ventana_Paciente_Registrado.class.getResource("/v_id
ent_paciente/resources/little.png")));
        lbllblimage9.setBounds(1469, 5, 178, 178);
        panelImages.add(lbllblimage9);

        // image 10
        panel10_7 = new JPanel();
        panel10_7.setBackground(Colores[0]);
        panel10_7.setBounds(1709, 44, 16, 16);
        panelImages.add(panel10_7);

        panel10_6 = new JPanel();
        panel10_6.setBackground(Colores[0]);
        panel10_6.setBounds(1691, 50, 16, 16);
        panelImages.add(panel10_6);

        panel10_5 = new JPanel();
        panel10_5.setBackground(Colores[0]);
        panel10_5.setBounds(1675, 62, 16, 16);
        panelImages.add(panel10_5);

        panel10_4 = new JPanel();
        panel10_4.setBackground(Colores[0]);
        panel10_4.setBounds(1675, 93, 16, 16);
        panelImages.add(panel10_4);

        panel10_3 = new JPanel();
        panel10_3.setBackground(Colores[0]);
        panel10_3.setBounds(1683, 131, 16, 16);
        panelImages.add(panel10_3);

        panel10_2 = new JPanel();
        panel10_2.setBackground(Colores[0]);
        panel10_2.setBounds(1691, 151, 16, 16);
        panelImages.add(panel10_2);

        panel10_1 = new JPanel();
        panel10_1.setBackground(Colores[0]);
        panel10_1.setBounds(1700, 131, 16, 16);
        panelImages.add(panel10_1);

        JLabel lbllblimage10 = new JLabel("");
        lbllblimage10.setIcon(new
ImageIcon(Ventana_Paciente_Registrado.class.getResource("/v_id
ent_paciente/resources/little.png")));
        lbllblimage10.setBounds(1652, 5, 178, 178);
        panelImages.add(lbllblimage10);

        contentPane.add(panel);
```

```
        panel.setLayout(null);
        panel.add(panel_5L);
        panel.add(panel_7L);
        panel.add(panel_7R);
        panel.add(panel_5R);
        panel.add(panel_4L);
        panel.add(panel_4R);
        panel.add(panel_3L);
        panel.add(panel_1L);
        panel.add(panel_2L);
        panel.add(panel_1R);
        panel.add(panel_6L);
        panel.add(panel_3R);
        panel.add(panel_2R);
        panel.add(panel_6R);

        // label that contain the image
        JLabel lblNewLabel = new JLabel("");
        lblNewLabel.setBounds(6, 6, 480, 501);
        lblNewLabel.setIcon(new
ImageIcon(Ventana_Paciente_Registrado.class.getResource("/v_id
ent_paciente/resources/dada.png")));
        panel.add(lblNewLabel);
        contentPane.add(lblNewLabel_2);
        contentPane.add(lblSelectionarPuertaBluetooth);
        contentPane.add(portList1);
        contentPane.add(panelbutton);
        panelbutton.setLayout(null);
        panelbutton.add(btnEmpezarAcquisicion);
        panelbutton.add(btnTerminarAcquisicion);
        panelbutton.add(btnGuardarTest);
        panelbutton.add(btnVolver);

        JSeparator separator = new JSeparator();
        separator.setBounds(642, 372, 356, 2);
        contentPane.add(separator);

        JLabel lblComentario = new JLabel("Comentario:");
        lblComentario.setBounds(635, 275, 363, 16);
        lblComentario.setFont(new Font("", Font.PLAIN, 14));
        contentPane.add(lblComentario);

        JTextArea txtrSadga = new JTextArea();
        txtrSadga.setBounds(635, 296, 363, 65);
        contentPane.add(txtrSadga);

        JSeparator separator_1 = new JSeparator();
        separator_1.setBounds(635, 262, 363, 2);
        contentPane.add(separator_1);

        }
```

```
int e = 0;
private static int c1;
private static int c7;
private static int c6;
private static int c5;
private static int c4;
private static int c3;
private static int c2;
private static JPanel panel1_1;
private static JPanel panel1_3;
private static JPanel panel1_4;
private static JPanel panel1_5;
private static JPanel panel1_6;
private static JPanel panel1_7;
private static JPanel panel1_2;
private static JPanel panel2_7;
private static JPanel panel2_6;
private static JPanel panel2_5;
private static JPanel panel2_4;
private static JPanel panel2_3;
private static JPanel panel2_2;
private static JPanel panel2_1;
private static JPanel panel3_7;
private static JPanel panel3_6;
private static JPanel panel3_3;
private static JPanel panel3_4;
private static JPanel panel3_5;
private static JPanel panel3_1;
private static JPanel panel3_2;
private static JPanel panel4_1;
private static JPanel panel4_2;
private static JPanel panel4_3;
private static JPanel panel4_4;
private static JPanel panel4_5;
private static JPanel panel4_6;
private static JPanel panel4_7;
private static JPanel panel5_7;
private static JPanel panel5_6;
private static JPanel panel5_5;
private static JPanel panel5_4;
private static JPanel panel5_3;
private static JPanel panel5_2;
private static JPanel panel5_1;
private static JPanel panel6_7;
private static JPanel panel6_6;
private static JPanel panel6_5;
private static JPanel panel6_4;
private static JPanel panel6_3;
private static JPanel panel6_2;
```

```
    private static JPanel panel6_1;
    private static JPanel panel7_7;
    private static JPanel panel7_6;
    private static JPanel panel7_5;
    private static JPanel panel7_4;
    private static JPanel panel7_3;
    private static JPanel panel7_2;
    private static JPanel panel7_1;
    private static JPanel panel8_2;
    private static JPanel panel8_1;
    private static JPanel panel8_3;
    private static JPanel panel8_4;
    private static JPanel panel8_5;
    private static JPanel panel8_6;
    private static JPanel panel8_7;
    private static JPanel panel9_7;
    private static JPanel panel9_6;
    private static JPanel panel9_5;
    private static JPanel panel9_4;
    private static JPanel panel9_3;
    private static JPanel panel9_1;
    private static JPanel panel9_2;
    private static JPanel panel10_7;
    private static JPanel panel10_6;
    private static JPanel panel10_5;
    private static JPanel panel10_4;
    private static JPanel panel10_3;
    private static JPanel panel10_2;
    private static JPanel panel10_1;
    private JMenuItem mntmAadirComentario;
    //  this method contains all the code for creating events

    private void createEvents() {


        mnArchivoGrabaciones.addMenuListener(new
MenuListener() {
            public void menuCanceled(MenuEvent arg0) {
            }
            public void menuDeselected(MenuEvent arg0) {

            }
            public void menuSelected(MenuEvent arg0) {

                if (e==0) {
                String Npaciente;  // Nombre
                String Apaciente;  // Apellido
                Npaciente =lblDefaultNombre.getText();
                Apaciente=lblDefaultAppellido.getText();
```

```
                    String parentDirectory =
("c:\\Users\\luca93\\Desktop\\CarpetaPacientes\\" );
                    String fileExtension = ".txt";
                    String directoryName =
parentDirectory+"Carpeta_"+Npaciente+"_"+Apaciente;

                    File parentDir = new File(directoryName);
                    String[] listOfTextFiles =
parentDir.list();
                    // Put the names of all files ending with
.txt in a String array
                    if (listOfTextFiles!= null) {
                         for (String f : listOfTextFiles) {
                              String fichero =
Npaciente+Apaciente.concat(fileExtension) ;

                              if (fichero.equals(f) ) {
                              }else {
                                   System.out.println(f);
                                   mntmGrabaciones = new
JMenuItem(f);

     mnArchivoGrabaciones.add(mntmGrabaciones);
                              }
                         }
                    }
                    e=1;
               }
               }
          });

          // bnt volver ---> Confirma to go to
Ventana_Identificacion_Paciente
          btnVolver.addActionListener(new ActionListener() {
               public void actionPerformed(ActionEvent arg0) {

                    Object[] options = {"Sí","No","Cancelar"};
                    int n = JOptionPane.showOptionDialog(frame,
                              "¿Quiere volver a la ventana
precedente?",
                              "¿Está seguro de continuar?",
                              JOptionPane.YES_NO_CANCEL_OPTION,
                              JOptionPane.QUESTION_MESSAGE,
                              null,
                              options,
                              options[2]);
                    if (n == JOptionPane.YES_OPTION) {
                         System.out.println("Clicked Yes");
                         Ventana_Identificacion_Paciente vip=
new Ventana_Identificacion_Paciente();
                    vip.setVisible(true);
```

```
                    dispose();
                    }
                }
            });

          // btnEmpezar Acquisicion -> connection to Bluetooth
Arduino--> display data
                btnEmpezarAcquisicion.addActionListener(new
ActionListener() {
                    public void actionPerformed(ActionEvent e)
                    {
                         arduino = new
Arduino(portList1.getSelectedItem().toString(),baud_rate);
                        inizio=1;

                             }
                             });//end btnEmpezar Acquisicion

                btnTerminarAcquisicion.addActionListener(new
ActionListener() {
                    public void actionPerformed(ActionEvent e)
{
                         inizio=0;

                        arduino.closeConnection(); // ????????????
                         }
                });

                //btnGuardarTest save the data  from Sensors
                btnGuardarTest.addActionListener(new
ActionListener() {
                    public void actionPerformed(ActionEvent e)
{
                         String Npaciente;  // Nombre
                         String Apaciente;  // Apellido
                         //String CFpaciente; // codigo fiscal
                         Npaciente =lblDefaultNombre.getText();

     Apaciente=lblDefaultAppellido.getText();
                         //CFpaciente=txtCodigoF.getText();

                         BufferedReader br;
                    FileWriter fichero=null;
                         try {
                              Date date = new Date() ;
                              SimpleDateFormat dateFormat = new
SimpleDateFormat("yyyy-MM-dd HH-mm-ss") ;
                              File file = new
File("c:\\\\Users\\\\luca93\\\\Desktop\\\\CarpetaPacientes\\\\
"+"Carpeta_"+Npaciente+"_"+Apaciente+"\\"
+dateFormat.format(date) + ".txt") ;
```

```
                    fichero=new FileWriter(file);


     fichero.write(System.lineSeparator()); //new line
                         fichero.write("DATOS RECIBIDOS \n
"+dateFormat.format(date)+":\r\n"); // Start write data
                         String [] value=DisplayPanel();
                         for (j=0;j<value.length;j++) {
                              if (value[j]!=null) {
                                   String v=value[j];
                                   System.out.println(v);
                                   fichero.write(v); //
write data
                              }
                         }
                         fichero.close();
                         //br.close();
                    } catch (IOException e1) {
                         e1.printStackTrace();
                    }

               }

          });// end btnGuardarTest
     } //end createEvents

     public static  String [] DisplayPanel() {
          while (inizio==1) {
               if (arduino.openConnection()) {
                    if (inizio==1) {
                     a=arduino.serialRead(0);  // receive data
char from Arduino and converted in String
                    if (a!=null) {
                    b[i]=a;
                    i++;
                    }
                    if (a.isEmpty()==false) {
                         parts=a.split("-");
                         nmeasure++;
                         System.out.println(nmeasure);
                         for (j =1; j<parts.length;j++) {

                              if (parts[j] != null &&
parts[j].trim().length()>0 ){

     x1=Double.parseDouble(parts[j]);
                              if (x1==9999) {

     //System.out.println("New String");
                              } else {
                                   ChangeColors();
```

```
                                        }
                                    }
                                }
                            }
                        }
                        }
        }
        return b;
}


public static void ChangeColors()
{
        double cost=113.64; // 5000V/43colores
        double b=x1/cost;
        c= (int) b;
        if (j==1) { // sensor 1
             panel_1L.setBackground(Colores[c]); c1=c;
        }
        if (j==2) { // sensor 0
             panel_2L.setBackground(Colores[c]); c2=c;
        }
        if (j==3) { // sensor 1
             panel_3L.setBackground(Colores[c]); c3=c;
        }
        if (j==4) { // sensor 0
             panel_4L.setBackground(Colores[c]); c4=c;
        }
        if (j==5) { // sensor 1
             panel_5L.setBackground(Colores[c]); c5=c;
        }
        if (j==6) { // sensor 0
             panel_6L.setBackground(Colores[c]); c6=c;
        }
        if (j==7) { // sensor 1
             panel_7L.setBackground(Colores[c]); c7=c;
        }
        if (nmeasure <=10) {
             switch (nmeasure) {
             case 1:
                  panel1_1.setBackground(Colores[c1]);
                  panel1_2.setBackground(Colores[c2]);
                  panel1_3.setBackground(Colores[c3]);
                  panel1_4.setBackground(Colores[c4]);
                  panel1_5.setBackground(Colores[c5]);
                  panel1_6.setBackground(Colores[c6]);
                  panel1_7.setBackground(Colores[c7]);
                  break;
             case 2:
                  panel2_1.setBackground(Colores[c1]);
                  panel2_2.setBackground(Colores[c2]);
```

```
            panel2_3.setBackground(Colores[c3]);
            panel2_4.setBackground(Colores[c4]);
            panel2_5.setBackground(Colores[c5]);
            panel2_6.setBackground(Colores[c6]);
            panel2_7.setBackground(Colores[c7]);
            break;
        case 3:
            panel3_1.setBackground(Colores[c1]);
            panel3_2.setBackground(Colores[c2]);
            panel3_3.setBackground(Colores[c3]);
            panel3_4.setBackground(Colores[c4]);
            panel3_5.setBackground(Colores[c5]);
            panel3_6.setBackground(Colores[c6]);
            panel3_7.setBackground(Colores[c7]);
            break;
        case 4:
            panel4_1.setBackground(Colores[c1]);
            panel4_2.setBackground(Colores[c2]);
            panel4_3.setBackground(Colores[c3]);
            panel4_4.setBackground(Colores[c4]);
            panel4_5.setBackground(Colores[c5]);
            panel4_6.setBackground(Colores[c6]);
            panel4_7.setBackground(Colores[c7]);
            break;
        case 5:
            panel5_1.setBackground(Colores[c1]);
            panel5_2.setBackground(Colores[c2]);
            panel5_3.setBackground(Colores[c3]);
            panel5_4.setBackground(Colores[c4]);
            panel5_5.setBackground(Colores[c5]);
            panel5_6.setBackground(Colores[c6]);
            panel5_7.setBackground(Colores[c7]);
            break;
        case 6:
            panel6_1.setBackground(Colores[c1]);
            panel6_2.setBackground(Colores[c2]);
            panel6_3.setBackground(Colores[c3]);
            panel6_4.setBackground(Colores[c4]);
            panel6_5.setBackground(Colores[c5]);
            panel6_6.setBackground(Colores[c6]);
            panel6_7.setBackground(Colores[c7]);
            break;
        case 7:
            panel7_1.setBackground(Colores[c1]);
            panel7_2.setBackground(Colores[c2]);
            panel7_3.setBackground(Colores[c3]);
            panel7_4.setBackground(Colores[c4]);
            panel7_5.setBackground(Colores[c5]);
            panel7_6.setBackground(Colores[c6]);
            panel7_7.setBackground(Colores[c7]);
            break;
```

```
                case 8:
                    panel8_1.setBackground(Colores[c1]);
                    panel8_2.setBackground(Colores[c2]);
                    panel8_3.setBackground(Colores[c3]);
                    panel8_4.setBackground(Colores[c4]);
                    panel8_5.setBackground(Colores[c5]);
                    panel8_6.setBackground(Colores[c6]);
                    panel8_7.setBackground(Colores[c7]);
                    break;
                case 9:
                    panel9_1.setBackground(Colores[c1]);
                    panel9_2.setBackground(Colores[c2]);
                    panel9_3.setBackground(Colores[c3]);
                    panel9_4.setBackground(Colores[c4]);
                    panel9_5.setBackground(Colores[c5]);
                    panel9_6.setBackground(Colores[c6]);
                    panel9_7.setBackground(Colores[c7]);
                    break;
                case 10:
                    panel10_1.setBackground(Colores[c1]);
                    panel10_2.setBackground(Colores[c2]);
                    panel10_3.setBackground(Colores[c3]);
                    panel10_4.setBackground(Colores[c4]);
                    panel10_5.setBackground(Colores[c5]);
                    panel10_6.setBackground(Colores[c6]);
                    panel10_7.setBackground(Colores[c7]);
                    break;
                }
            }else {nmeasure=0;}
        }

        void DeclaracionColeres() {//Declaración de los colores
            Colores[0]=new java.awt.Color(0x19,0x19,0x70); //azul
            Colores[1]=new java.awt.Color(0x00,0x00,0x80);
            Colores[2]=new java.awt.Color(0x48,0x3D,0x8B);
            Colores[3]=new java.awt.Color(0x7B,0x68,0xEE);
            Colores[4]=new java.awt.Color(0x41,0x69,0xE1);
            Colores[5]=new java.awt.Color(0x1E,0x90,0xFF);
            Colores[6]=new java.awt.Color(0x00,0xBF,0xFF);
            Colores[7]=new java.awt.Color(0x48,0xD1,0xCC);
            Colores[8]=new java.awt.Color(0x00,0xFF,0xFF);
            Colores[9]=new java.awt.Color(0x7F,0xFF,0xD4);
            Colores[10]=new java.awt.Color(0x7C,0xFC,0x00);
            Colores[11]=new java.awt.Color(0x00,0xFF,0x00);
            Colores[12]=new java.awt.Color(0x00,0xFF,0x00);
//verde
            Colores[13]=new java.awt.Color(0x1A,0xFF,0x00);
            Colores[14]=new java.awt.Color(0x2B,0xFF,0x00);
            Colores[15]=new java.awt.Color(0x3C,0xFF,0x00);
            Colores[16]=new java.awt.Color(0x4D,0xFF,0x00);
            Colores[17]=new java.awt.Color(0x5E,0xFF,0x00);
```

```
            Colores[18]=new java.awt.Color(0x6F,0xFF,0x00);
            Colores[19]=new java.awt.Color(0x7C,0xFF,0x00);
            Colores[20]=new java.awt.Color(0x8D,0xFF,0x00);
            Colores[21]=new java.awt.Color(0x9E,0xFF,0x00);
            Colores[22]=new java.awt.Color(0xAE,0xFF,0x00);
            Colores[23]=new java.awt.Color(0xBF,0xFF,0x00);
            Colores[24]=new java.awt.Color(0xCF,0xFF,0x00);
            Colores[25]=new java.awt.Color(0xDF,0xFF,0x00);
            Colores[26]=new java.awt.Color(0xEF,0xFF,0x00);
            Colores[27]=new java.awt.Color(0xFF,0xFF,0x00);
//amarillo
            Colores[28]=new java.awt.Color(0xFF,0xFE,0x00);
            Colores[29]=new java.awt.Color(0xFF,0xED,0x00);
            Colores[30]=new java.awt.Color(0xFF,0xDB,0x00);
            Colores[31]=new java.awt.Color(0xFF,0xC8,0x00);
            Colores[32]=new java.awt.Color(0xFF,0xB6,0x00);
            Colores[33]=new java.awt.Color(0xFF,0xA4,0x00);
            Colores[34]=new java.awt.Color(0xFF,0x92,0x00);
            Colores[35]=new java.awt.Color(0xFF,0x80,0x00);
            Colores[36]=new java.awt.Color(0xFF,0x7F,0x00);
            Colores[37]=new java.awt.Color(0xFF,0x6E,0x00);
            Colores[38]=new java.awt.Color(0xFF,0x5D,0x00);
            Colores[39]=new java.awt.Color(0xFF,0x4B,0x00);
            Colores[40]=new java.awt.Color(0xFF,0x38,0x00);
            Colores[41]=new java.awt.Color(0xFF,0x26,0x00);
            Colores[42]=new java.awt.Color(0xFF,0x14,0x00);
            Colores[43]=new java.awt.Color(0xFF,0x00,0x00);
//rojo}
        }
}
```