



UNIVERSITÀ DEGLI STUDI DI TRENTO

Dipartimento di Ingegneria e Scienza dell'Informazione

Corso di Laurea in
Informatica

ELABORATO FINALE

**STUDIO E SVILUPPO DI UN PROTOCOLLO DI
COMUNICAZIONE PER SERVIZI NON VERBALI
TRA DISPOSITIVI “WEARABLE”**

Supervisori
Prof. Renato Lo Cigno
Dott.ssa Eleonora Mencarini

Laureanda
Laura Scoccianti

Anno accademico 2018/2019

Ringraziamenti

Desidero ringraziare il professor Renato Lo Cigno che, con le sue preziose indicazioni, mi ha guidata nella stesura di questa tesi.

Ringrazio inoltre la Fondazione Bruno Kessler, in particolare la dottoressa Eleonora Mencarini, che ha accettato di essere la mia tutor aziendale, il dottor Davide Giovanelli e il dottor Alessandro Cappelletti per il supporto tecnico fornитоми durante il tirocinio.

Infine un grazie ai miei genitori per avermi sempre sostenuto ed aver creduto in me.

Indice

Sommario	3
1 Introduzione	5
1.1 Ausili informatici esistenti	5
2 Il progetto ACROSS	7
2.1 Scenario applicativo	7
2.2 Stato dell'arte	7
2.2.1 Descrizione dell'hardware	8
2.3 Scopo della tesi	9
3 Sviluppo del software e del protocollo di comunicazione	11
3.1 Flusso di controllo	11
3.2 Prima versione	11
3.2.1 Libreria RHMESH	11
3.2.2 Problemi riscontrati	12
3.3 Seconda versione: protocollo di comunicazione ad hoc	12
3.3.1 Formato dei messaggi	12
3.3.2 Instradamento	13
3.3.3 Limiti	14
3.4 Test e risultati	15
4 Design dell'interazione	19
4.1 Interazione Uomo-Macchina nei dispositivi indossabili esistenti	19
4.2 Nuovo design dell'hardware	19
4.3 Funzionamento	21
4.4 Interfaccia utente	21
5 Conclusioni e lavori futuri	23
Bibliografia	25
A Stralci di codice	27
A.1 Dichiarazione globale delle variabili - Tabella di routing	27
A.2 Ricezione e gestione della tabella di routing	27
A.3 Ritrasmissione	27

Sommario

Sempre più persone si avvicinano alla pratica degli sport di montagna, compreso l'alpinismo. Nelle scuole di arrampicata, gli istruttori sono sempre più impegnati con un'utenza di principianti, con i quali, durante le salite in cordata, devono poter comunicare in maniera chiara ed efficace. La comunicazione tra scalatori, tradizionalmente, avviene per via vocale diretta, tramite walkie-talkie oppure con le "tirate di corda".

Tuttavia, la distanza o il rumore del vento potrebbero rendere impraticabili le prime due, mentre le "tirate di corda", oltre a richiedere una particolare concentrazione, cosa non facile per un principiante, potrebbero anche generare equivoci (ad esempio, qualche strattone dato per liberare una corda impigliata che viene scambiato per un segnale).

Da queste considerazioni, nel 2018, presso l'unità di ricerca *i3 (Intelligent Interfaces and Interaction)* della Fondazione Bruno Kessler di Trento è nato il progetto *ACROSS (Augmented Communication foR Outdoor SportS)*. Questo progetto, sovvenzionato dalla Fondazione Caritro attraverso il finanziamento Prot. SG 1939/17, propone di affiancare alle già esistenti forme di comunicazione, una tecnologia indossabile che, oltre ad avere un ingombro ridotto, consenta uno scambio di informazioni essenziale ma efficace. Inoltre, dovendo operare in zone dove non è garantita la copertura per la telefonia cellulare e internet, era necessario prevedere di poter stabilire una comunicazione a distanza in maniera autonoma.

È stato pertanto sviluppato un sistema, consistente di due prototipi basati sulla piattaforma Arduino, in grado di scambiarsi messaggi tramite rete radio.

L'interfaccia prescelta per la visualizzazione delle informazioni era una striscia di 8 LED. I messaggi trasmessi dovevano essere interpretati dall'utente in base alla quantità, al colore e alla sequenza dei LED accesi. Questa tesi illustra il lavoro svolto, successivamente a quello appena descritto, per implementare le potenzialità del sistema, al fine di renderlo rispondente alle ulteriori esigenze scaturite da una successiva e più estensiva analisi dell'argomento.

Come prima riflessione, è apparso evidente che i problemi di comunicazione trattati in precedenza non appartengono esclusivamente alle accoppiate istruttore-allievo, ma a tutti gli scalatori. Di certo quelli esperti hanno soglie di criticità più elevate rispetto agli altri, ma in situazioni estreme, che in montagna possono verificarsi anche in maniera repentina, quali forte vento o pioggia, potrebbero anche loro beneficiare di un valido supporto tecnologico. Del resto, anche in buone condizioni, la presenza di più cordate nella stessa zona potrebbe dar luogo ad un pericoloso intreccio di comandi vocali. Oppure nella stessa cordata potrebbero trovarsi persone che parlano lingue diverse con le immaginabili difficoltà di comprensione reciproca.

Si è quindi pensato che sarebbe stato utile connettere intere cordate, piuttosto che limitare a soli due dispositivi la possibilità di comunicare. Tramite un'implementazione del software, si è proceduto a creare una piccola rete costituita da vari nodi, in grado di servire un gruppo di scalatori. I vari nodi, oltre a poter comunicare ciascuno con tutti gli altri, possono anche espletare la funzione di "ripetitori", facendo quindi da tramite tra elementi che altrimenti non potrebbero colloquiare perché troppo distanti e quindi fuori portata, oppure non in visibilità ottica perché, ad esempio, separati da uno sperone roccioso.

Si è anche preso atto che l'interfaccia LED, pur se economica e sufficiente allo scopo, fosse poco pratica, poiché costringeva gli utenti ad uno sforzo mnemonico per l'interpretazione, che richiede una lucidità non sempre disponibile in condizioni critiche. La si è quindi sostituita con un piccolo display a colori. È stata modificata anche la pulsantiera deputata alla selezione ed invio dei messaggi, rendendo l'utilizzo molto più semplice ed intuitivo.

Oltre a modificare i due prototipi già esistenti, ne è stato approntato un terzo, allo scopo di effettuare tutte le necessarie verifiche di funzionamento.

Tenendo in debita considerazione che lo sviluppo del progetto è in fase embrionale, che l'hardware utilizzato, pur se gradevole e adeguato allo scopo, è piuttosto economico, con tutte le limitazioni che ciò comporta, alla luce dei test effettuati (capitolo 3.4), che hanno confermato il funzionamento della rete, ci si può ritener soddisfatti. Attualmente il sistema consta di soli tre prototipi ma, mantenendo gli stessi protocolli già predisposti, è immediatamente, e senza necessità di modifiche, estendibile ad un numero superiore.

1 Introduzione

La pratica in sicurezza degli sport di montagna richiede, oltre ad una preparazione appropriata dal punto di vista fisico e mentale, una buona coordinazione con i compagni di attività, con particolare riferimento alla comunicazione tra i vari componenti, ad esempio, di una salita in cordata. Eventuali incomprensioni tra scalatori possono essere causa di eventi drammatici, anche con esito infausto. Allo stesso modo, la difficoltà di comunicazione tra istruttore e allievo può rendere difficoltoso il processo di apprendimento, se non addirittura compromettere la sicurezza e l'incolumità delle persone.

Gli studi condotti nel tempo in merito alle criticità nella comunicazione tra scalatori, sono eloquentemente riassunti dall'alpinista Paul Petzoldt [1]:

The human voice is difficult to hear and understand on a mountain. The belayer might be out of his companion's sight, words do not carry well around rock projections, wind and rain sometimes make conversations impossible, even at short distances.

In montagna è difficile udire e capire la voce umana. L'assicuratore potrebbe essere al di fuori del campo visivo del compagno, le parole non si trasmettono bene attorno alle prominenze rocciose, il vento e la pioggia talvolta rendono la comunicazione impossibile, anche su brevi distanze.

Oltre agli ostacoli ambientali, non sono da sottovalutare i problemi legati a un'errata comunicazione: laddove due alpinisti esperti e in sintonia possono concedersi delle chiacchiere informali durante una salita [2], lo stesso non è concesso a un alpinista inesperto. Lo scambio di messaggi verbali deve essere chiaro e preciso, non deve lasciare spazio ad ambiguità, poiché anche da esso dipende la sicurezza di tutta la cordata.

Quando non c'è possibilità di comunicare verbalmente, è consuetudine utilizzare i "tiri di corda" (strattoni alla corda di sicurezza con un codice predefinito, ad esempio tre strattoni separati da 1s significano "sei assicurato, puoi arrampicare"), i quali tuttavia sono affetti da ambiguità: degli strattoni dati per liberare una corda impigliata potrebbero essere erroneamente interpretati come segnale di via libera.

Il principale supporto tecnologico per la comunicazione negli sport di montagna è dato dall'utilizzo di *walkie talkie*, ma questi dispositivi presentano degli svantaggi quali il peso, l'ingombro, il prezzo e soprattutto la necessità di fissarli all'abbigliamento dello sportivo in una posizione abbastanza agevole per l'utilizzo e che al contempo non intralci l'utilizzo delle corde e i movimenti. Bisogna inoltre tener conto che, in ambienti estremi come la montagna, possono sempre verificarsi fenomeni improvvisi e rumorosi, come il forte vento, che potrebbero rendere impossibile una conversazione verbale anche tramite apparati elettronici. Sembra inoltre superfluo citare i problemi che potrebbero crearsi in un gruppo di scalatori che parlano lingue diverse.

1.1 Ausili informatici esistenti

Negli ultimi anni gli smartphone e gli smartwatch abbinati hanno affiancato e talvolta sostituito le radio. Purtroppo, pur avendo superato alcuni svantaggi, presentano a loro volta dei limiti per le nostre esigenze: uno smartwatch, pur avendo la comodità di essere *wearable*, presentando quindi un ingombro estremamente limitato e un'alta usabilità, è inservibile se non viene associato a uno smartphone e, come quest'ultimo, necessita che ci sia segnale telefonico e banda internet.

In generale, gli sport di montagna costituiscono un terreno quasi inesplorato dai sistemi informatici, soprattutto dal punto di vista della comunicazione. È vero che esistono svariati software più o meno commerciali per reperire informazioni sulle vie e le palestre di roccia, per registrare gli allenamenti e i progressi personali. Inoltre, si fa sempre più uso di sensori per analizzare i movimenti e

le prestazioni dello scalatore, mentre, con l'utilizzo della realtà aumentata, sono stati sviluppati dei programmi per l'analisi delle pareti rocciose per meglio studiare come affrontarle, aumentando il livello di consapevolezza e di sicurezza.

Non risultano però reperibili esempi rilevanti di tecnologia in supporto della comunicazione durante l'attività, fatta eccezione per le radio. Nonostante il recente e grandissimo interesse per le Wireless Sensor Networks e l'Internet of Things, le poche soluzioni proposte in questo ambito sono generalmente orientate al monitoraggio delle funzioni vitali e alla reperibilità in caso di incidenti, per l'eventuale successivo intervento di personale di soccorso [3].

L'obiettivo di questa tesi è la progettazione e realizzazione di un sistema di comunicazione locale, che consenta a più dispositivi di scambiarsi reciprocamente informazioni, comportandosi di fatto come nodi di una rete peer-to-peer. Per perseguire il risultato, non si vuole ricorrere a protocolli esistenti, quali Bluetooth, WiFi, 4G, bensì svilupparne uno ad hoc, basato su comunicazione radio, semplice e poco esigente a livello di hardware, tanto da poter essere supportato da dispositivi estremamente economici e di dimensioni ridotte, che ne facilitino l'indossabilità.

La tesi è organizzata in tre capitoli principali. Il capitolo 2 presenta il progetto e lo stato dell'arte e pone gli obiettivi che si vogliono conseguire alla fine dello sviluppo; il capitolo 3 offre una panoramica sulle soluzioni implementative adottate, mentre il capitolo 4 tratta il design dell'interazione come conseguenza della nuova implementazione, per poi passare a una breve analisi dei risultati raggiunti nelle conclusioni.

2 Il progetto ACROSS

Lo sviluppo del progetto ACROSS¹ (*Augmented Communication foR Outdoor SportS*) è possibile grazie al supporto economico fornito dalla Fondazione Caritro (finanziamento Prot. SG 1939/17), nonché dalla Fondazione Bruno Kessler e in particolare l'unità di ricerca *i3*² (*Intelligent interfaces and interaction*), che offre il supporto logistico.

Il progetto ACROSS nasce con l'obiettivo di sviluppare una tecnologia di supporto agli sport di montagna che consenta una comunicazione efficace tra gli atleti, anche in presenza degli ostacoli già esposti nel capitolo di introduzione. In una prima fase di studio, propedeutica alla progettazione e allo sviluppo, sono state dettagliatamente valutate le necessità di comunicazione tra scalatori e le connesse criticità, anche tramite interviste ai diretti interessati [4] [5].

Si è scelto di sviluppare una tecnologia di tipo indossabile, sull'idea dei moderni *smartwatch*, che, in virtù della sua minima invasività, non vada ad ostacolare in maniera significativa lo svolgimento dell'attività sportiva. I dispositivi realizzati permettono lo scambio di messaggi preimpostati volti a segnalare situazioni problematiche o semplicemente ad accettare le condizioni dei compagni di attività o fornire indicazioni.

2.1 Scenario applicativo

Lo scenario di utilizzo ottimo è quello delle prime uscite in cordata di un allievo accompagnato da un istruttore: è possibile immaginare numerosi casi in cui una delle due parti possa aver bisogno di comunicare velocemente con l'altra. Bisogna anzitutto considerare che tra due persone in cordata si ha una distanza che può arrivare agli 80 metri, pertanto comunicare verbalmente potrebbe essere molto difficoltoso. Inoltre, è possibile che in alcuni tratti della via di roccia ci siano degli ostacoli ad impedire il contatto visivo tra due persone, rendendo quindi necessaria una comunicazione basata su segnali fatti solamente tirando la corda.

In simili situazioni, i dispositivi proposti dal progetto ACROSS, oggetto di studio in questo lavoro di tesi, renderebbero la comunicazione di base, volta a dare informazioni sulla percorribilità della via o a verificare che non ci siano problemi, molto più immediata e, cosa estremamente rilevante, inequivocabile.

2.2 Stato dell'arte

La fase realizzativa del progetto è stata avviata durante l'estate 2018. Il sistema sviluppato si compone di due soli dispositivi, in grado di comunicare tra loro con uno scambio di pacchetti radio. Gli utenti interagiscono con i prototipi mediante tre pulsanti e possono visualizzare i messaggi (Tabella 2.1) su una striscia di otto LED.

Funzionamento Ogni prototipo può trovarsi, alternativamente, in tre stati: ricezione, invio, inattivo. L'utente può cambiare modalità utilizzando il tasto `state`. Il secondo pulsante `mode` viene usato per navigare tra i messaggi inviabili e si procede quindi all'invio di quello selezionato premendo `send`. Il dispositivo torna automaticamente nello stato di ricezione dopo l'invio di un messaggio, così da minimizzare quanto più possibile il numero di interazioni dell'utente durante l'attività sportiva.

¹Sito web del progetto: <http://i3.fbk.eu/projects/across>

²<http://i3.fbk.eu/>

Messaggio	Colore	Rappresentazione	
		Selezione	Ricezione
Received <i>Ricevuto</i>	Bianco	1: 2: 3: 4:	1: 2: 3: 4:
Go <i>Vai</i>	Verde	1: 2: 3: 4:	1: 2: 3: 4: 5: 6: 7: 8:
Stop	Rosso		 Intermittente
How are you? <i>Come stai?</i>	Blu	1: 2: 3: 4:	1: 2: 3: 4: 5: 6: 7: 8:
Problem <i>Problema</i>	Giallo		
OK	Verde		

Tabella 2.1: Messaggi inviabili e loro codifica

La ricezione di un messaggio è segnalata da una vibrazione del prototipo e dall'accensione dei LED. Per interrompere la visualizzazione del messaggio ricevuto e, di conseguenza, la vibrazione, si deve semplicemente posizionare il dispositivo nella modalità di invio, senza necessità di trasmettere alcunché. Pertanto, sarà sufficiente navigare in successione i tre stati fino a tornare a quello di ricezione.

Nonostante non sia possibile ricevere mentre si è in modalità invio, quando si è nella fase di scelta del messaggio da inviare, questo viene visualizzato su 4 LED degli 8 disponibili, onde evitare ogni possibile fraintendimento.

Il sistema si compone di due soli prototipi, pertanto i pacchetti radio inviati non contengono alcuna informazione sul mittente e sul destinatario e la comunicazione avviene sempre direttamente.

2.2.1 Descrizione dell'hardware

Il progetto è stato sviluppato su piattaforma Arduino. L'hardware presentato di seguito rappresenta l'insieme dei componenti utilizzati per realizzare i prototipi descritti finora.

Scheda Adafruit Feather 32u4 Radio con modulo RFM69HCW, Arduino compatibile
 Arduino è una piattaforma hardware open-source, flessibile e di facile utilizzo. Grazie anche all'ampia documentazione, alle numerose librerie e ai costi contenuti, risulta essere un ottimo strumento per la prototipazione in vari campi di applicazione, primi tra tutti la robotica e l'automazione. Per questo progetto è stato scelto di usare la scheda Arduino compatibile *Adafruit Feather 32u4 Radio* (Figura 2.1) per le dimensioni ridotte e la presenza del modulo radio RFM69HCW integrato, che opera sulla frequenza di 433MHz.

Il modulo permette di realizzare una rete wireless più estesa di una 802.15.4 a 2.4GHz, più flessibile di una rete basata su Bluetooth e senza l'alta richiesta energetica del WiFi. In questo modo è stato possibile realizzare dei dispositivi totalmente indipendenti da telefoni cellulari e collegamenti ad internet e in grado di coprire distanze di diverse decine di metri. Questa particolare caratteristica si rivela fondamentale, dal momento che questi dispositivi sono pensati per un uso in ambiente montano,

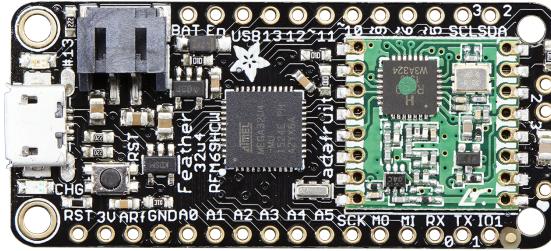


Figura 2.1: Adafruit Feather 32u4 con modulo radio RFM69HCW integrato

dove non è raro che i telefoni e gli *smartwatch* risultino inutilizzabili per la mancanza di segnale e l'utilizzo dei *walkie talkie* presenta le limitazioni già evidenziate nell'introduzione.

Alla scheda sono collegati tre pulsanti e una striscia di 8 LED, come già accennato in precedenza, e un motorino per la vibrazione con il relativo *controller* (Figura 2.2). In figura 2.4 si può osservare lo schema elettrico del prototipo allo stato dell'arte.

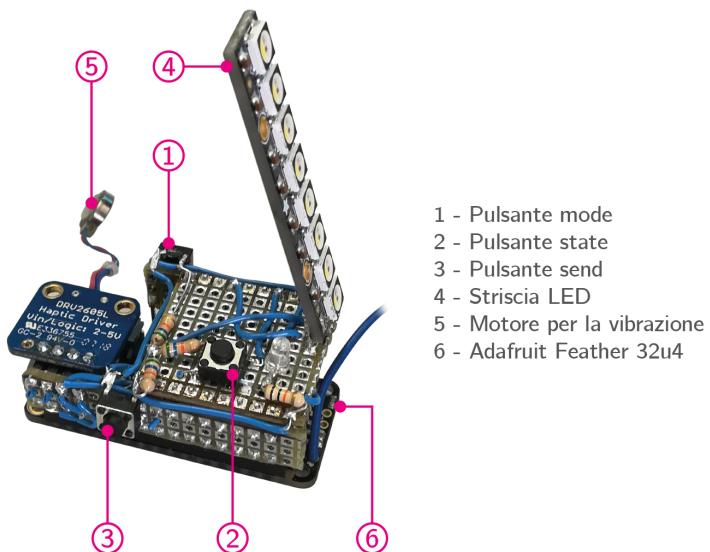


Figura 2.2: Prototipo completo.

2.3 Scopo della tesi

Il dover rappresentare i messaggi mediante l'accensione di 8 LED, seppure utilizzando una codifica basata sui colori e sulla sequenza di accensione, suscita non poche perplessità, legate alle capacità mnemoniche degli utilizzatori, soprattutto in condizioni ambientali ed operative critiche. Inoltre, prevedendo un sistema composto da soli due apparati, la comunicazione è uno-a-uno, pertanto a livello di trasmissione continua ad avere molte delle limitazioni di una coppia di *walkie talkie*.

L'obiettivo di questa tesi è estendere il sistema per creare una piccola rete di dispositivi - in questo caso 3, ma con possibilità di aggiungerne altri, entro i limiti ragionevoli di un gruppo di cordata, senza che il funzionamento ne risulti compromesso -, con un protocollo di comunicazione ad hoc che permetta meccanismi di *message passing*, andando a creare un'essenziale rete a maglia. In questo modo, oltre a poter trasmettere in broadcast o a un utente specifico, due dispositivi non nella portata l'uno dell'altro possono comunicare tra loro, grazie ai nodi intermedi, che fungono da ripetitori (Figura 2.3).

Questa nuova implementazione richiede che l'utente selezioni il destinatario della trasmissione. Per questo motivo e per rendere inequivocabile la visualizzazione dei messaggi è stato necessario un

re-design dell'interazione e dell'hardware.

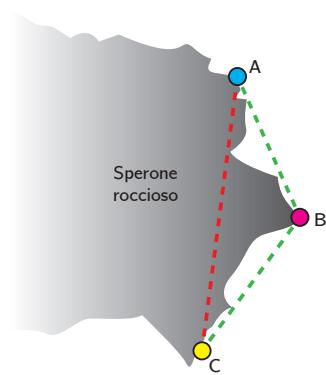
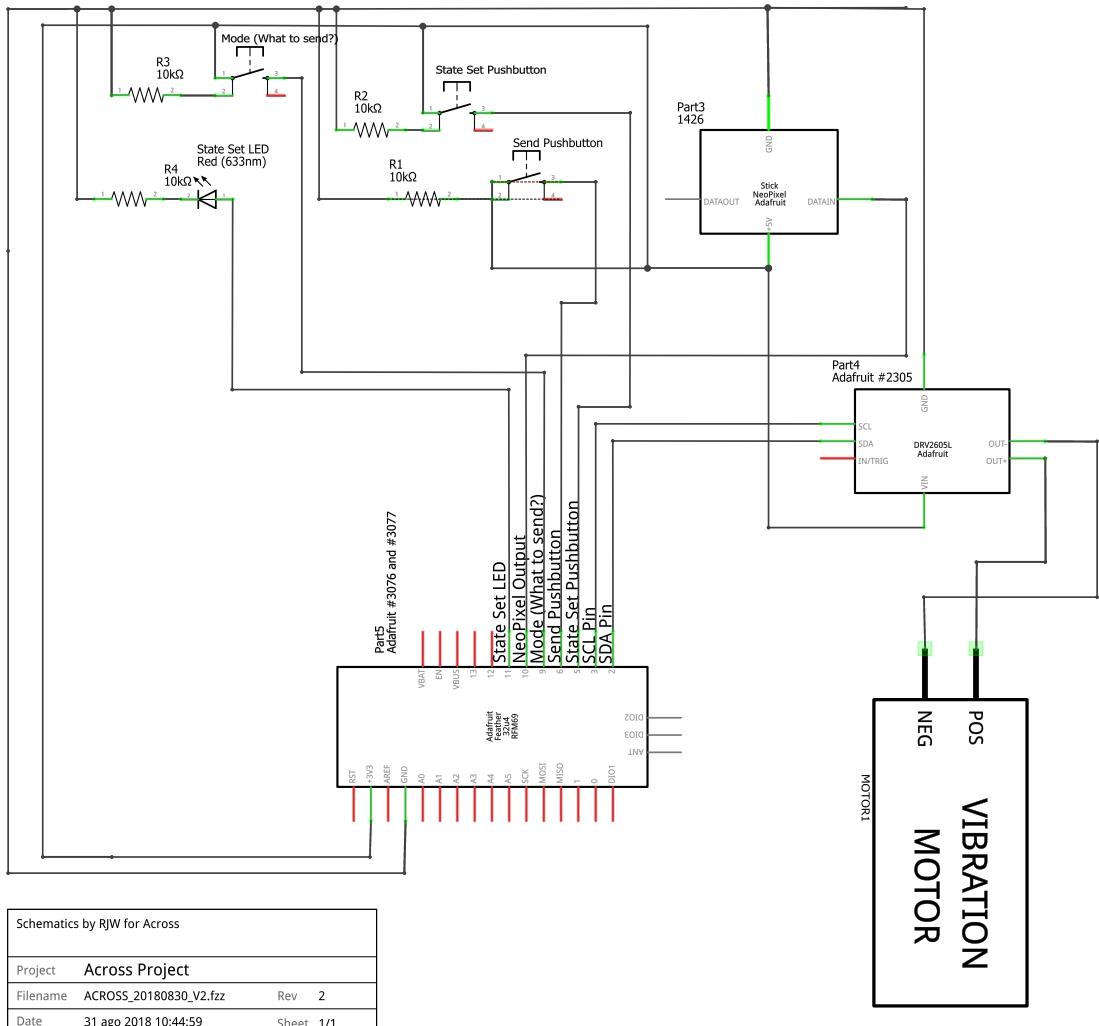


Figura 2.3: I nodi A e C non possono comunicare direttamente per via dell'ostacolo roccioso nel mezzo. Con il protocollo sviluppato, A e C possono comunicare e i messaggi seguiranno il cammino (A,B,C)



fritzing

Figura 2.4: Schema elettrico del prototipo

3 Sviluppo del software e del protocollo di comunicazione

Stabilito l’obiettivo del progetto, preso atto delle necessità, delle criticità da superare e deciso con quali dispositivi operare, è giunto il momento di procedere con la fase attuativa. Il primo step è stato la realizzazione di una rete mesh in grado di garantire un funzionamento affidabile e caratterizzata da protocolli abbastanza leggeri da poter essere supportati dall’hardware.

3.1 Flusso di controllo

Innanzitutto, si è stabilito che i devices possano assumere, alternativamente, uno di questi stati:

- 0: il dispositivo è in stand-by (non trasmette e non riceve – da utilizzare solo in momenti di inattività, al fine di realizzare un risparmio energetico);
- 1: modalità di invio dei messaggi;
- 2: modalità di ricezione dei messaggi.

Tali stati sono rappresentati da una variabile (`count`). L’utente può navigare tra questi stati mediante un apposito pulsante. Tramite il ciclo di *fetch-decode* di Arduino, si va continuamente a verificare lo stato della variabile `count` e, in base a quanto riscontrato, si devia di volta in volta il flusso verso il ramo di esecuzione software appropriato. All’accensione, la variabile `count` ha valore 0 ed il sistema è in stand-by. In fase di attività, la condizione di default è `count=2`.

3.2 Prima versione

In un primo momento, si è ritenuto opportuno basare lo sviluppo del codice sull’utilizzo della classe *RHMesh* della libreria open-source *RadioHead*¹ per Arduino, la quale fornisce tutte le funzioni necessarie per la costruzione di una rete a maglia.

Nella fase iniziale di studio e sviluppo, per motivi pratici, si è deciso di controllare le schede tramite la porta seriale di un PC, simulando il funzionamento dei prototipi.

3.2.1 Libreria RHMesh

Le funzioni presenti all’interno di RHMesh consentono di inviare pacchetti indirizzati attraverso una rete, con meccanismi di *route discovery* automatici. RHMesh è particolarmente adatta a reti con una topologia dinamica, con i nodi che possono spostarsi o disattivarsi, come nel caso dei prototipi presentati in questa tesi.

Una sua importante caratteristica è la presenza di un meccanismo di *acknowledgement*, che garantisce l’avvenuta consegna al *next-hop*. Tuttavia il mittente non ha garanzia del fatto che il pacchetto abbia effettivamente raggiunto la destinazione desiderata, a meno che non si tratti del next-hop.

Costruzione e gestione della routing table

All’accensione, nessun nodo conosce la *route* verso gli altri nodi della rete. All’invio di un messaggio, viene consultata la tabella di routing per verificare l’esistenza di un percorso verso la destinazione

¹Il codice e la documentazione completa sono reperibili al seguente indirizzo: www.airspayce.com/mikem/arduino/RadioHead/

indicata e, in caso contrario, si ricorre alla *route discovery*. Di conseguenza, il nodo invia un apposito messaggio in broadcast e ogni ricevitore controlla se è una richiesta di una *route* verso sé stesso: in caso affermativo, invierà un'apposita risposta, altrimenti rimbalzerà in broadcast la richiesta, aggiungendo il proprio indirizzo alla lista dei nodi già da essa visitati.

Se, ad un certo momento, un nodo smette di essere raggiungibile da un altro, quest'ultimo invierà un apposito messaggio di segnalazione in broadcast, che comporterà l'aggiornamento delle tabelle di routing.

3.2.2 Problemi riscontrati

L'uso di *RHMesh* costituirebbe senz'altro un'ottima soluzione ai fini del progetto, poiché offre una buona gestione delle tabelle di routing e, soprattutto, un sistema di *acknowledgement* che rende la comunicazione affidabile.

D'altra parte, la necessità per l'utente di poter scegliere il destinatario dei messaggi ha comportato il dover sostituire l'interfaccia LED con un display, per il cui utilizzo è necessario ricorrere ad un'apposita libreria (tale modifica verrà trattata in maniera specifica nel capitolo successivo).

I microcontrollori montati su schede Arduino - o Arduino-compatibili, come in questo caso - dispongono di tre tipi di memoria:

- La *Flash Memory*, dove vengono caricati lo *sketch*² e le librerie necessarie;
- La SRAM, utilizzata durante l'esecuzione;
- La EEPROM, che contiene informazioni a lungo termine.

Nel caso in esame, i devices utilizzati hanno le seguenti caratteristiche:

- Flash Memory: 32KB (di cui disponibili circa 28KB);
- SRAM: 2KB;
- EEPROM: 1KB.

Purtroppo, la capacità della Flash Memory non era sufficiente ad accogliere sia lo sketch che le librerie per RHMesh e per la gestione del display. Non potendo rinunciare alla libreria per il display, si è deciso di eliminare quella di RHMesh. È stato quindi implementato lo sketch con del nuovo codice che potesse assolvere ad una gestione semplificata della rete. Ne deriva che il nuovo protocollo di comunicazione è molto più essenziale rispetto a RHMesh, pur mantenendo un elevato livello di funzionalità in relazione alla specificità dell'utilizzo.

3.3 Seconda versione: protocollo di comunicazione ad hoc

Per la realizzazione del nuovo protocollo, dopo un'accurata analisi delle necessità e tenuto conto anche delle limitazioni imposte dall'hardware, si è scelto di operare come descritto di seguito in merito al formato dei messaggi e del loro instradamento.

3.3.1 Formato dei messaggi

Per i messaggi, è stato scelto un formato essenziale composto di quattro campi, come illustrato in figura 3.1.

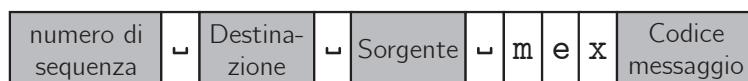


Figura 3.1: Formato generale dei messaggi

²Qualsiasi programma scritto per Arduino viene chiamato *sketch*

I campi sorgente e destinazione contengono gli ID univoci dei prototipi, assegnati durante la programmazione. Il numero di sequenza è associato al dispositivo sorgente e viene incrementato a ogni invio. Il contenuto del messaggio, quindi quello che l'utente visualizzerà sul display, è indicato dal carattere seguente la stringa `mex`. A ogni carattere dalla lettera `A` alla lettera `F` corrisponde uno dei messaggi illustrati precedentemente in tabella 2.1.

Un esempio di messaggio è il seguente:

```
12 3 1 mexA
```

Dove `12` indica il numero di sequenza del messaggio, `3` il dispositivo destinatario, `1` il mittente e `A` il messaggio. Le relative visualizzazioni sui display saranno come in figura 3.2.



Figura 3.2: A) Display mittente; B) Display destinatario.

Il messaggio inviato è una stringa di caratteri ASCII, dichiarata come `char radiopacket[15]`. Ne deriva che i limiti di trasmissione saranno i seguenti:

- Numero di sequenza: compreso tra 0 e 255, poiché rappresentato con un intero unsigned a 8 bit (`uint8_t`). Richiede al massimo tre caratteri (3 bytes);
- ID di mittente e destinatario: teoricamente compresi tra 0 e 255 (`uint8_t`), ma volutamente limitati tra 0 e 98 - 99 è l'indirizzo di broadcast - in quanto ritenuto un numero più che sufficiente per rappresentare un gruppo di scalatori. Richiedono al massimo due caratteri (2 bytes) l'uno;
- Stringa `mex` e carattere identificativo del messaggio: 4 caratteri totali (4 bytes);
- Tre spazi per separare i campi: 3 bytes;
- Il quindicesimo byte istanziato nella dichiarazione del messaggio è riservato al carattere terminatore della stringa.

È possibile variare i limiti appena descritti con semplici modifiche al codice.

La decodifica del messaggio ricevuto viene effettuata mediante la funzione `sscanf()`, come visibile in allegato A.2.

3.3.2 Instradamento

L'instradamento dei messaggi si basa sul *flooding* controllato con i numeri di sequenza. Il nodo sorgente inserisce il suo identificatore e un numero di sequenza nel pacchetto, oltre all'indirizzo di destinazione (99 per la trasmissione in broadcast). Ogni nodo mantiene una struttura dati in memoria in cui associa ogni altro nodo con il quale è venuto in contatto al numero di sequenza dell'ultimo messaggio da questo ricevuto.

Alla ricezione di un messaggio, il nodo controlla se la destinazione corrisponde al suo ID e se la coppia (`numero di sequenza, sorgente`) è valida, cioè se la differenza con il numero di sequenza salvato in memoria è maggiore o uguale a 1. Se il nodo è la destinazione e il messaggio è valido, questo verrà mostrato sul display del dispositivo e l'utente sarà avvisato con una vibrazione.

Se il messaggio è valido ma la destinazione non corrisponde al nodo ricevente, quest'ultimo aspetterà un intervallo di tempo casuale, dopo il quale ritrasmetterà il pacchetto senza modifiche. Si è ritenuto opportuno inserire tale ritardo casuale nella ritrasmissione per ridurre la remota possibilità di collisioni causate da trasmissioni contemporanee.

Se il messaggio non è valido, viene scartato.

Se, per qualsiasi ragione, un nodo viene spento o resettato, il suo numero di sequenza si azzera. Ciò potrebbe comportare un problema, qualora avesse precedentemente trasmesso dei messaggi. Per gestire

questa eventualità, è sembrato ragionevole stabilire che, nel caso si riceva un pacchetto la cui differenza tra il numero di sequenza salvato nella memoria del ricevitore e quello del nuovo messaggio sia maggiore di 3, la trasmissione si possa considerare valida, poiché il nodo sorgente è presumibilmente stato spento o resettato. Rimane una criticità nell'eventualità che il nodo resettato avesse precedentemente trasmesso 1, 2 o 3 messaggi: in tal caso i nuovi saranno scartati fino a che non raggiungeranno il numero di sequenza memorizzato dal ricevitore aumentato di 1. Questo comporta che l'utente trasmittitore, non ottenendo risposta, dovrà ritrasmettere il messaggio fino ad un massimo di 3 volte. Tale disagio può ritenersi accettabile.

In figura 3.3 è presentato il diagramma di flusso dell'algoritmo appena descritto.

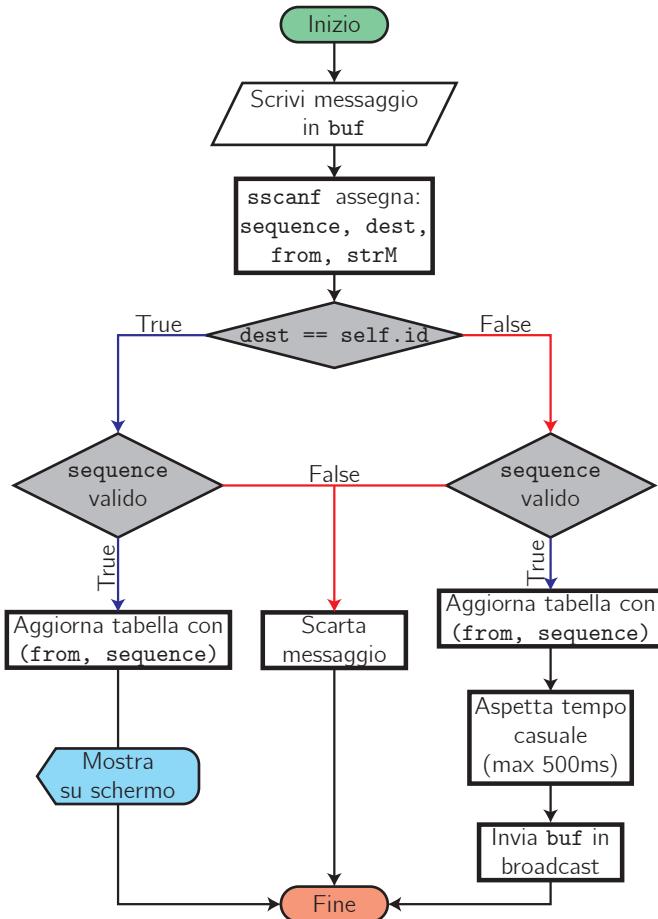


Figura 3.3: Diagramma di flusso dell'algoritmo di instradamento

3.3.3 Limiti

Il protocollo realizzato non assicura l'affidabilità della trasmissione ma del resto anche RHMESH assicura la sola consegna al next-hop. Inoltre, non gestisce l'accesso alla rete, per cui è possibile incorrere in collisioni durante le ritrasmissioni, la cui probabilità aumenta con l'aumentare del numero di nodi.

È comunque opportuno considerare quanto segue in merito alla natura della comunicazione nella pratica degli sport di montagna:

- È vero che il protocollo non assicura automaticamente che la trasmissione vada a buon fine, ma è senz'altro molto più importante che l'utente risponda manualmente nel caso riceva un messaggio;
- Nell'eventualità di una mancata risposta si possono supporre due scenari: la trasmissione potrebbe non essere andata a buon fine, oppure il destinatario è impossibilitato a rispondere. In entrambi i casi, per buon senso, sarà opportuno ritrasmettere il messaggio dopo un ragionevole lasso di tempo;

- I prototipi non sono destinati ad essere il principale strumento di comunicazione, bensì un ausilio nel caso in cui non sia possibile farlo verbalmente. Inoltre, durante l'arrampicata in cordata l'utilizzo sarà circoscritto a segnalare la via libera o eventuali difficoltà, quindi lo scambio di messaggi sarà poco frequente.
- In considerazione della consistenza dei gruppi di cordata, non si prevede che la rete si componga di un grande numero di nodi, per cui l'eventualità di collisioni sarà piuttosto improbabile.

3.4 Test e risultati

L'obiettivo del test era verificare il corretto funzionamento del protocollo sviluppato. Si rendeva pertanto necessario porre due nodi a debita distanza, affinchè fossero ciascuno fuori della portata dell'altro, mentre il terzo nodo doveva trovarsi in un punto intermedio ed essere in grado di comunicare con entrambi gli altri, quindi nella possibilità di fare da tramite.

Allo scopo, era necessario individuare un luogo che offrisse le opportune caratteristiche in termini di ampiezza, presenza di ostacoli alla trasmissione da sfruttare per simulare eventuali condizioni critiche della montagna, nonché la riservatezza e la tranquillità indispensabili per condurre le operazioni. La scelta è ricaduta sullo studentato NEST di Trento - che si ringrazia - la cui struttura, come chiaramente rilevabile dalla tabella 3.1, è così riassumibile: palazzina di quattro piani, a forma di "L", ed un giardino esterno.

Di seguito, per semplicità e sintesi, si useranno P1 e P3 per identificare i due nodi estremi, mentre con P2 si farà riferimento al nodo intermedio.

La prova è stata effettuata sia all'interno che all'esterno dello stabile. Si evidenzia che la struttura base della palazzina è metallica e densamente cablata, di conseguenza costituisce un ostacolo alla ottimale propagazione delle onde elettromagnetiche.

Prove di trasmissione sullo stesso piano Si è proceduto inizialmente a posizionare i dispositivi P1 e P3 agli estremi dei corridoi e ad interporre P2 in modo che avesse visibilità ottica con entrambi gli altri (prova numero 1, tabella 3.1). Si è rilevato, dopo cinque trasmissioni di prova, che i punti P1 e P3 non riuscivano a comunicare. È stato quindi messo in funzione P2 e si è verificato, tramite l'invio di un messaggio in broadcast, che fosse nella portata di entrambi gli altri. Dopo cinque tentativi, si è constatato che P1 e P2, distanti 17 metri tra loro, sono riusciti a comunicare nel 100% dei casi, mentre la comunicazione tra P2 e P3 non è mai avvenuta. Si è quindi proceduto ad avvicinare progressivamente P3 a P2, con intervalli di un metro, effettuando ogni volta due tentativi di comunicazione. Alla distanza di 17 metri la trasmissione è andata a buon fine. Sono state quindi effettuate dieci prove, verificando che nel 100% dei casi P1 e P3 riuscivano a comunicare per il tramite di P2. Come ulteriore verifica, P2 è stato spento e con cinque ulteriori prove è stato appurato che P1 e P3 non riuscivano a comunicare direttamente (prova numero 2).

Prove di trasmissione su due piani diversi Nella prova effettuata su due piani diversi (prova numero 3), si è dapprima appurato che P1 e P3, nei cinque tentativi effettuati, non sono mai riusciti a comunicare. Ponendo P2 sulla verticale di P3, la comunicazione tra P1 e P3, per il tramite di P2, ha avuto successo in tutte le dieci trasmissioni effettuate.

Prove di trasmissione su tre piani diversi I tre prototipi sono stati posizionati su tre piani diversi (prova numero 4). Come consuetudine, si è prima tentato di trasmettere un messaggio tra P1 e P3, posti a due piani di distanza, fallendo nel 100% delle cinque prove effettuate. Le successive dieci prove, in cui P2 faceva da tramite, hanno avuto pieno successo.

Prove di trasmissione sulle scale Analogamente alla prova numero 4, i prototipi sono stati collocati sulle scale dell'edificio (prova numero 5), ciascuno in un piano diverso. Inizialmente si è tentato di comunicare direttamente tra P1 e P3, con esito negativo. Si è invece avuto un totale successo nelle dieci prove effettuate con P2 a fare da intermediario. Come si può notare dalla figura alla riga 5

della tabella 3.1, grazie alla semiapertura offerta dal vano scale, anche effettuando degli spostamenti di qualche metro dei prototipi, la trasmissione era possibile, a differenza di quanto avveniva nelle prove numero 3 e 4, dove gli stessi erano obbligati ad essere quasi perfettamente sovrapposti.

Prove di trasmissione all'aperto Nelle prove effettuate all'esterno, con visibilità diretta o con ostacoli poco rilevanti (siepe, ringhiera metallica) tra P1 e P3, la trasmissione ha avuto pieno successo in tutti i molteplici tentativi effettuati, a varie distanze, fino a un massimo di 50 metri, per limitazioni fisiche dello spazio a disposizione. Non è stato quindi possibile valutare la reale portata massima degli strumenti, che è molto probabilmente da ritenersi superiore (prove numero 6 e 7).

Posizionando i prototipi P1 e P3 ai lati del vano scale, la cui struttura è realizzata in acciaio e vetro e pertanto produce un'eccessiva attenuazione del segnale radio, la comunicazione diretta tra loro è fallita in tutti i dieci tentativi effettuati (prova numero 8). Posizionando il prototipo P2 a fare da tramite in un punto intermedio, con visibilità ottica verso gli altri due, la trasmissione ha avuto esito positivo nel 100% dei casi.

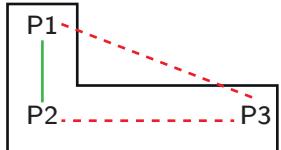
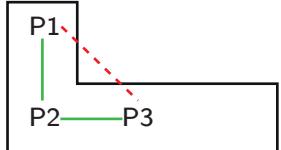
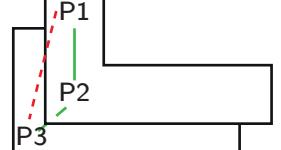
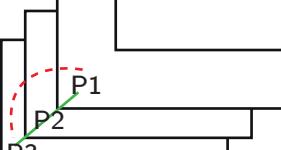
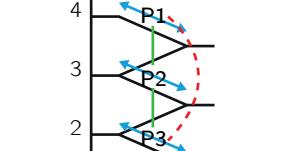
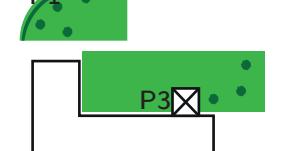
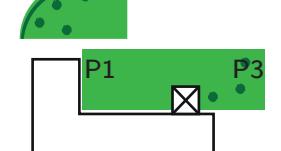
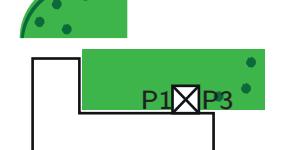
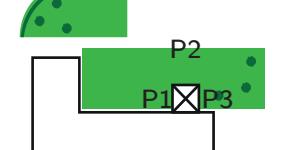
Nr.	Posizioni	Distanze	Comunicazione P1 - P3
1		P1 ←———— 17m —————→ P2 P2 ←———— 35m —————→ P3	Fallita P2 - P3 troppo distanti
2		P1 ←———— 17m —————→ P2 P2 ←———— 17m —————→ P3	Successo Tramite P2
3		P1 ←———— 17m —————→ P2 P2 ←———— 3.50m —————→ P3	Successo Tramite P2
4		P1 ←———— 3.50m —————→ P2 P2 ←———— 3.50m —————→ P3	Successo Tramite P2
5		P1 ←———— 3.50m —————→ P2 P2 ←———— 3.50m —————→ P3	Successo Tramite P2
6		P1 ←———— 50m —————→ P3	Successo Comunicazione diretta
7		P1 ←———— 50m —————→ P3	Successo Comunicazione diretta
8		P1 ←———— 10m —————→ P3	Fallita
9		P1 ←———— 20m —————→ P2 P2 ←———— 20m —————→ P3	Successo Tramite P2

Tabella 3.1: Risultati dei test

4 Design dell'interazione

Al paragrafo 3.2.2 è stato fatto cenno al motivo per cui si è deciso di sostituire i LED con un display. Non è stato però spiegato perché sia stato scelto quel particolare display e cosa abbia ispirato il design dell'interazione.

Inizialmente, tramite una approfondita ricerca, sono stati osservati dei prodotti già esistenti sul mercato che presentassero modalità di interazione uomo-macchina potenzialmente applicabili al progetto. L'attenzione si è subito focalizzata sui dispositivi indossabili.

4.1 Interazione Uomo-Macchina nei dispositivi indossabili esistenti

I dispositivi *wearable* di maggior successo sono stati progettati per fornire la massima immediatezza nell'utilizzo. È possibile distinguerli macroscopicamente in tre gruppi:

- Gli smartwatch, dotati di touchscreen e oramai praticamente privi di pulsanti, se si escludono quelli di accensione e spegnimento, che talvolta possono anche assolvere una ulteriore funzione programmabile dall'utente.
- Gli *sport tracker*, non sempre provvisti di touchscreen e, mentre i più semplici sono utilizzabili con uno o due pulsanti o con sensori tattili, i più complessi possono avere anche 5 o più tasti per permettere la navigazione in tutti i menù.
- Le *action cam*, dotate di pulsanti, in numero variabile da due a quattro in base ai modelli e alle funzionalità offerte.

Considerate le necessità del progetto, nonché le caratteristiche dell'hardware utilizzato, il modello più interessante da prendere come riferimento è risultato senza ombra di dubbio l'*action cam*. La motivazione è costituita dalla possibilità di gestire menù anche piuttosto complessi con un numero limitato di pulsanti e senza dover necessariamente ricorrere a tecnologie più avanzate e costose, come il touch screen.

4.2 Nuovo design dell'hardware

Dopo aver valutato vari display, si è scelto di acquistare la scheda **Adafruit Mini Color TFT with Joystick FeatherWing** (Figura 4.1) che, oltre ad essere dotata di uno schermo a colori da 0.96", dispone di un joystick a cinque direzioni, due pulsanti ed è perfettamente coincidente, in termini di dimensioni (51mm x 23mm) e di disposizione dei pin, con la scheda Adafruit 32u4 Feather sulla quale è montata e con la quale realizza un prototipo indossabile. È dotata inoltre di un *framebuffer*, che consente un fondamentale risparmio della memoria del microcontrollore durante l'utilizzo.

I tre tasti utilizzati nella versione nativa del progetto sono stati rimossi, mentre è stato mantenuto il motore per la vibrazione.

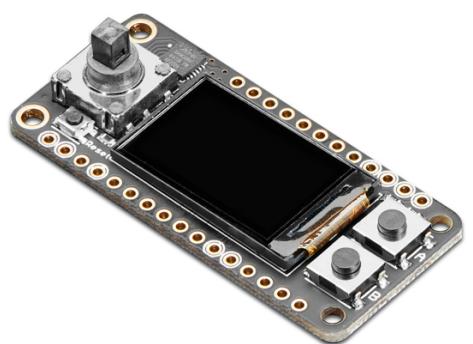


Figura 4.1: Adafruit Mini Color TFT with Joystick FeatherWing

In figura 4.2 si possono osservare i singoli componenti del sistema, nonché il prototipo assemblato allo stato attuale.

È in previsione la realizzazione di un case tramite stampante 3D, che probabilmente sarà affidata a personale tecnico della Fondazione Bruno Kessler, titolare del progetto. Nella figura 4.3 si propone un render che mostra un'ipotetico aspetto che potrebbe avere il prototipo indossabile.

Chiaramente, il lavoro finora svolto è da considerare un semplice punto di partenza, il primo approccio alla realizzazione di un'idea. Un eventuale sviluppo del progetto con scopi commerciali, necessiterebbe di un lavoro di ingegnerizzazione che coinvolgerebbe molteplici professionisti e richiederebbe sforzi economici ben più importanti rispetto a quanto sia stato sinora possibile mettere in campo. Sicuramente si otterrebbero dispositivi più funzionali, più performanti, accattivanti nell'estetica, più robusti e resistenti ad urti e intemperie.

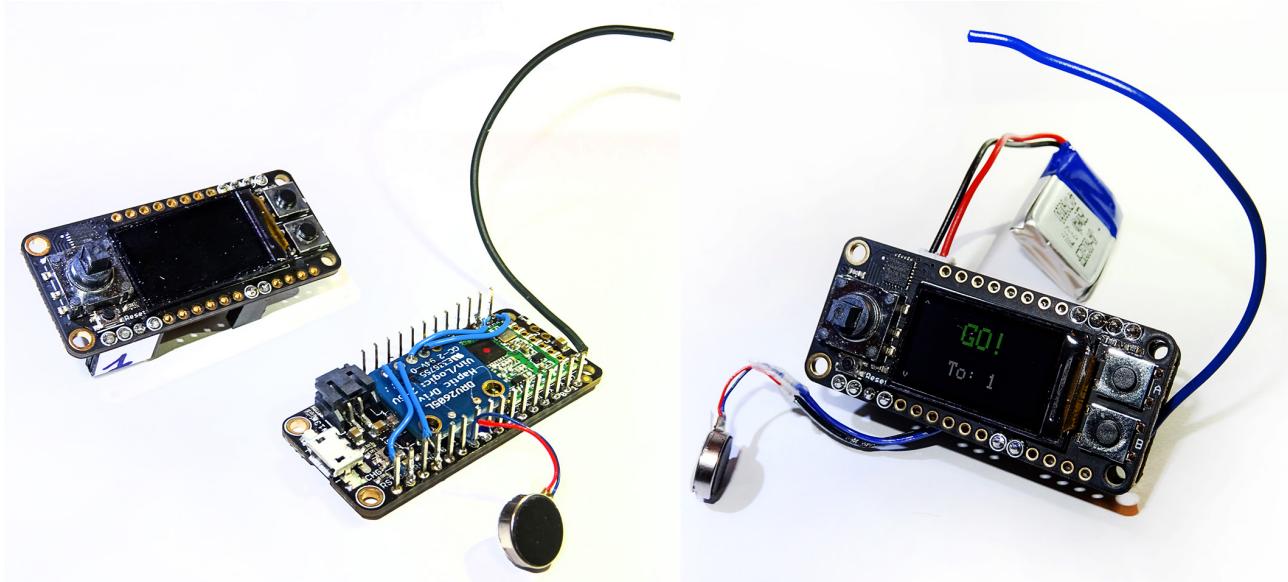


Figura 4.2: Sinistra: componenti del sistema. Destra: prototipo assemblato.

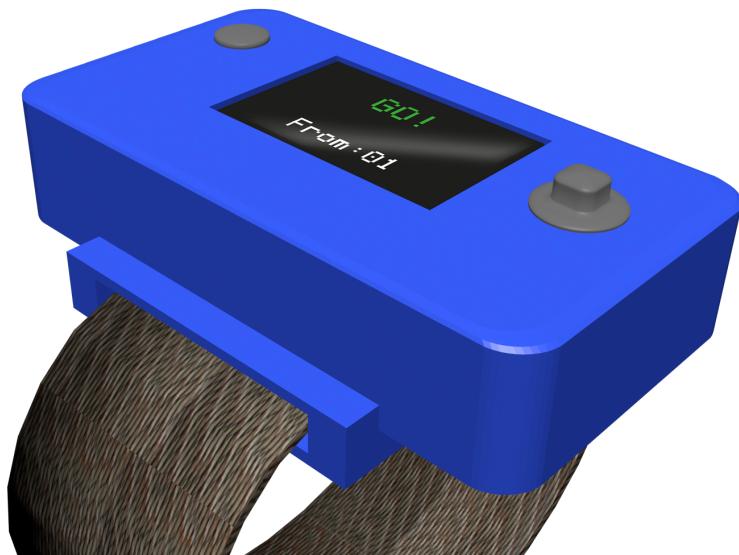


Figura 4.3: Render del prototipo indossabile.

4.3 Funzionamento

Di seguito, con riferimento alla figura 4.4, si illustrano le nuove modalità di utilizzo.

La navigazione tra gli stati avviene tramite il pulsante B, che va quindi a sostituire il pulsante *state*, già citato nella sezione 2.2. I tasti *mode* e *send*, tramite i quali si provvedeva alla selezione e all'invio dei messaggi, sono invece stati rimpiazzati dal joystick.

Dopo aver impostato il prototipo nello stato di invio, tramite il joystick sarà possibile selezionare il messaggio e, di seguito, anche il destinatario dello stesso. Infine, effettuando una pressione verticale, sempre sul joystick, il messaggio verrà trasmesso. È importante notare come l'intera operazione sia espletabile tramite l'utilizzo di un unico dispositivo di puntamento e selezione.

In qualsiasi momento l'invio può essere annullato premendo B e tornando in ricezione.

Modificando una minima quantità di righe di codice, si può ribaltare la disposizione delle informazioni e dei comandi sul display per adattarli sia ad utenti destrorsi che mancini, evitando così che la mano si interponga tra gli occhi e lo schermo, come si evidenzia nella rappresentazione in figura 4.5.

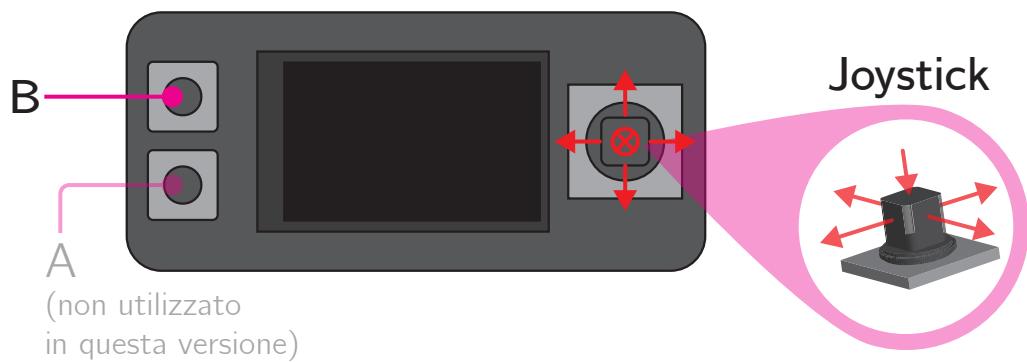


Figura 4.4: Schema dei pulsanti sul prototipo, nel caso di utilizzo sul polso sinistro.



Figura 4.5: A) Disposizione corretta; B) Disposizione errata.

4.4 Interfaccia utente

L'obiettivo principale per l'interfaccia grafica (GUI) è di garantire una leggibilità ottimale. Per questo, si è scelto di utilizzare lo sfondo nero e mostrare la minor quantità possibile di elementi grafici.

Per rendere l'interpretazione dei messaggi più immediata, si è scelto di associarli agli stessi colori mostrati nella tabella 2.1. Nella tabella 4.1 è rilevabile la differenza tra l'utilizzo dei LED e del display. Nel primo caso, l'utente deve concentrarsi sia sul colore che sulla sequenza di accensione dei singoli LED. Nel secondo caso la rappresentazione è esplicita e l'interpretazione inequivocabile, con in più la possibilità di riconoscere alcuni messaggi direttamente dal colore, senza neanche doverli leggere (ad esempio, "Stop!").

Messaggio	Rappresentazione		
	Colore	LED (Ricezione)	Display
Received <i>Ricevuto</i>	Bianco	1: [●○○○○○○○] 2: [●○○○○○●●] 3: [●●○○○●●●] 4: [●●○○○○●●] <i>In sequenza</i>	RECEIVED
Go <i>Vai</i>	Verde	1: [●○○○○○○○] 2: [●○○○○○○○] 3: [●●●○○○○○] 4: [●●●●○○○○] 5: [●●●●●○○○] 6: [●●●●●●○○] 7: [●●●●●●●○] 8: [●●●●●●●●] <i>In sequenza</i>	GO!
Stop	Rosso	[●●●●●●●●] <i>Intermittente</i>	STOP!
How are you? <i>Come stai?</i>	Blu	1: [●○○○○○○○] 2: [●●○○○○○○] 3: [●●●○○○○○] 4: [●●●●○○○○] 5: [●●●●●○○○] 6: [●●●●●●○○] 7: [●●●●●●●○] 8: [●●●●●●●●] <i>In sequenza</i>	HOW ARE YOU?
Problem <i>Problema</i>	Giallo	[●●●●●●●●]	PROBLEM
OK	Verde	[●●●●●●●●]	OK!

Tabella 4.1: Differenze nella rappresentazione dei messaggi tra lo stato dell'arte e la situazione attuale

Ogni stato in cui il prototipo può trovarsi è riconoscibile da una scritta sul display. Si evidenzia che, a differenza degli altri stati, nella modalità invio due frecce verticali ed un messaggio suggeriscono di utilizzare il joystick per selezionare un messaggio e un destinatario, scorrendo le rispettive liste.

Segue una rappresentazione dello schermo in modalità invio (Figura 4.6) e in modalità ricezione (Figura 4.7).



Figura 4.6: Modalità invio - A: schermata di default; B: l'utente sta selezionando il messaggio; C: l'utente sta selezionando il destinatario.



Figura 4.7: Modalità ricezione - A: schermata di default; B: l'utente ha ricevuto un messaggio.

5 Conclusioni e lavori futuri

Come già esposto, questa tesi è la prosecuzione del lavoro svolto, a partire dal 2018, nell'ambito del progetto ACROSS.

Allo stato dell'arte, erano stati realizzati due prototipi, basati su piattaforma Arduino, in grado di stabilire una comunicazione a distanza tra istruttore di alpinismo ed allievo, tramite lo scambio di semplici messaggi visualizzabili su una striscia di LED.

Analizzando in maniera più approfondita le problematiche riscontrabili dagli scalatori in merito alla comunicazione, si è giunti alla conclusione che, mantenere solo nell'ambito istruttore - allievo il campo di applicazione di questi dispositivi fosse troppo riduttivo.

Infatti, anche gruppi di scalatori esperti possono facilmente incorrere nelle varie problematiche più volte evidenziate nei capitoli precedenti. Andava quindi superato il limite di due dispositivi al momento gestibili dal sistema. Si è proceduto alla realizzazione di un protocollo di rete in grado di gestire, in maniera sicura ed efficace, un numero di nodi sufficiente a soddisfare un gruppo di cordata. Tale protocollo include dei meccanismi di message passing che consentono a ciascun nodo di comportarsi anche da ripetitore, fungendo da ponte tra nodi che non possono dialogare tra loro, o perché fuori portata o senza visibilità diretta a causa di ostacoli.

Aumentando i dispositivi collegati, la già poco pratica interfaccia LED risultava sicuramente inadeguata, per cui si è deciso di sostiturla con un piccolo display. Anche la vecchia pulsanteria è stata sostituita da un joystick e un tasto. La selezione dei messaggi e dei loro destinatari ora avviene tramite scorrimento di liste con il joystick, mentre il pulsante permette la navigazione tra i diversi stati dei prototipi.

Oltre a modificare i due prototipi già esistenti, adeguandoli alle nuove caratteristiche, si è provveduto ad improntarne un terzo, grazie al quale si è potuto procedere allo svolgimento dei test necessari per la verifica della funzionalità e validità del protocollo.

Gli ID dei dispositivi sono *hardcoded*, quindi assegnati direttamente nel codice, per garantire che non esistano duplicati.

È prevista la realizzazione, tramite stampante 3D, di un case per il contenimento e l'indossabilità dei prototipi.

L'hardware finora utilizzato è appena sufficiente ad assolvere le funzionalità descritte. In fase di ulteriore sviluppo del progetto, si suggerisce di potenziarlo. Di conseguenza, si potrebbero apportare interessanti miglioramenti quali, ad esempio, prevedere l'assegnazione di un *nickname* da parte dell'utente, che costituirebbe un notevole aiuto mnemonico.

Si potrebbe anche utilizzare il secondo pulsante, attualmente non abilitato, per l'invio di un messaggio automatico (“ricevuto”) di risposta. Per garantire che tale messaggio non venga trasmesso per errore, sarebbe utile prevedere una pressione prolungata del tasto.

Sarebbe interessante aggiungere un accelerometro, in grado di rilevare eventuali cadute. In tali casi, il dispositivo provvederebbe in automatico ad inviare un apposito messaggio di avvertimento in broadcast.

Infine, si potrebbe aggiungere una modalità di registrazione iniziale del gruppo, onde evitare il mescolarsi di messaggi con altre cordate dotate di apparecchiature simili. In fase di registrazione, da effettuare in luogo appartato, tutti i dispositivi, uno alla volta, dovranno inviare un apposito messaggio in broadcast. Gli indirizzi saranno registrati in un'apposita lista, che verrà utilizzata come filtro in modalità ricezione e come elenco di destinazioni in modalità invio.

Al momento è disponibile un set di sei messaggi, selezionati valutando le principali esigenze di comunicazione tra alpinisti. È comunque possibile sia ampliarlo, inserendo nuovi vocaboli, che sostituirlo. Questo rende il progetto adattabile anche ad altre attività ludico-ricreative come, ad esempio,

il softair: i giocatori, laddove impegnati in arene non coperte da connessione internet, potrebbero giovare di un sistema di messaggistica appositamente studiato per comunicare quando è indispensabile osservare il massimo silenzio. Un valido lavoro di design sia dell'interfaccia che dell'estetica dei dispositivi potrebbe rendere particolarmente invitante il prodotto per un simile mercato.

Anche l'attività venatoria, con particolare riferimento agli operatori di selezione che operano per conto di enti pubblici, potrebbe beneficiare di questo modo silenzioso di comunicare durante le lunghe sedute di appostamento.

Come qualsiasi altro progetto di ricerca, ACROSS vuole proporre un'idea nuova. Qualora suscitasse interessi commerciali, richiamando buoni investimenti, potrebbe costituire il punto di partenza per la progettazione e produzione di dispositivi dedicati, con elevato livello di performance e possibilità di successo di mercato.

Bibliografia

- [1] P. Petzoldt, *The Wilderness Handbook*. W.W. Norton and Company, 1974.
- [2] B. Moretti, “Parlare con i piedi per aria,” *Italiano e oltre*, no. 4, pp. 187–192, 1991.
- [3] S.-H. Lee, C.-Y. Hsu, and C.-S. Yang, “An intelligent emergency rescue assistance system for mountaineers,” *Int. J. Internet Technology and Secured Transactions*, vol. 8, no. 1, pp. 1–14, 2018.
- [4] E. Mencarini, C. Leonardi, A. De Angeli, and M. Zancanaro, “Design opportunities for wearable devices in learning to climb,” in *Proceedings of the 9th Nordic Conference on Human-Computer Interaction*, NordiCHI ’16, (New York, NY, USA), pp. 48:1–48:10, ACM, 2016.
- [5] E. Mencarini, A. De Angeli, and M. Zancanaro, “Emotions in climbing: A design opportunity for haptic communication,” in *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct*, UbiComp ’16, (New York, NY, USA), pp. 867–871, ACM, 2016.

Allegato A Stralci di codice

A.1 Dichiarazione globale delle variabili - Tabella di routing

In sede di dichiarazione globale delle variabili, di cui segue stralcio di codice, viene definito il tipo `node` (righe 1-4). Di seguito (riga 6) si dichiara l'array `network` (di tipo `node`), che costituirà la tabella di routing. `N_NODES` è la variabile che indica il numero massimo di nodi indirizzabili dalla rete.

Alle righe 8 e 9 viene dichiarata la variabile `self` (di tipo `node`) il cui campo `id` è inizializzato con l'identificativo assegnato al dispositivo.

```
1  typedef struct node {
2      uint8_t id = 0;
3      int lastpkt = 0;
4  };
5
6  node network[N_NODES];
7
8  node self;
9  self.id = 1;
```

A.2 Ricezione e gestione della tabella di routing

Quando il sistema riceve un messaggio (riga 11, codice A.1), procede, tramite la funzione `sscanf()` della libreria C++ `cstdio`, all'estrazione dei dati, inserendoli nelle variabili appositamente dichiarate (riga 16).

Successivamente si effettua la verifica, all'interno dell'array `network`, della presenza di precedenti messaggi dallo stesso mittente. Se l'esito è positivo, viene copiato il valore dell'indice nella variabile `k`. Contrariamente, si procede con l'inserimento dell'identificativo del nodo nella tabella e, anche in questo caso, il valore dell'indice viene salvato nella variabile `k`.

A.3 Ritrasmissione

Si prosegue verificando se il prototipo ricevente sia la destinazione del messaggio, quindi se ne controlla la validità, come già descritto al capitolo 3.3.2 (righe 34 e 42). Se entrambe le condizioni sono soddisfatte, tramite la funzione `showReceived()` alla riga 38 viene mostrato il messaggio sul display, con il formato indicato nel paragrafo 4.4. Qualora il messaggio sia valido ma il destinatario non coincida con il ricevente, si procede alla ritrasmissione. Il codice compreso tra le righe 43 e 47 provvede a generare il ritardo casuale, già descritto al paragrafo 3.3.2, nonché alla ritrasmissione del messaggio.

Listing A.1: Ricezione, gestione della tabella di routing e ritrasmissione

```
1  int sequence;
2  uint8_t dest;
3  uint8_t from;
4  char strM;
5
6
7  uint8_t buf[15];
8  uint8_t len = sizeof(buf);
9
10 //Ricezione messaggio
11 if (!rf69.recv(buf, &len)) {
12     return;
13 }
14
15 //Estrazione dei dati
16 sscanf(buf, "%d %d %d mex%c", &sequence, &dest, &from, &strM);
17
18 uint8_t k = 0;
19
20 // Ricerca e inserimento
21 for (uint8_t i = 0; i < N_NODES; i++) {
22     if (network[i].id == from) {
23         k = i;
24         break;
25     } else if (network[i].id == 0 && self.id != from) {
26         k = i;
27         network[k].id = from;
28         break;
29     }
30 }
31
32 //Ritrasmissione (se necessario)
33 if (dest == self.id || dest == 99) {
34     if (sequence - network[k].lastpkt >= 1 || sequence - network[k].lastpkt < -3) {
35         network[k].lastpkt = sequence;
36         tft.fillRect(ST77XX_BLACK);
37         showing = true;
38         showReceived(strM, from);
39     }
40 } else {
41     if (from != self.id) {
42         if (sequence - network[k].lastpkt >= 1 || sequence - network[k].lastpkt < -3) {
43             network[k].lastpkt = sequence;
44             rndNum = random(500);
45             delay((int)rndNum);
46             rf69.send((uint8_t *)buf, strlen(buf));
47             rf69.waitPacketSent();
48         }
49     }
50 }
```