UNIVERSITA' POLITECNICA DELLE MARCHE

FACOLTA' DI INGEGNERIA

Corso di Laurea triennale in Ingegneria Informatica e dell'Automazione

"Strumenti di Robotica Educativa per l'Identificazione e la Modellazione dell'Apprendimento"

"Educational Robotic Tools for the Identification and Modelling of Learning"

Relatore: Chiar.mo
Prof. **David Scaradozzi**

Tesi Laurea di:
Rumeysa Nur Gulesin

Matricola: 1087508

Correlatori:

Laura Screpanti, Lorenzo Cesaretti

A.A. **2020/2021**

Ringraziamenti

Prima di procedere con la trattazione, vorrei ringraziare sentitamente tutti coloro che hanno fatto la differenza durante questo percorso di crescita personale e professionale.

Un ringraziamento speciale al mio relatore David Scaradozzi e ai miei correlatori Laura Screpanti e Lorenzo Cesaretti per aver creduto nelle mie idee e per avermi fatto riscoprire la curiosità che mi animava da bambina e che credevo ormai perduta.

Un grazie alla mia mamma, che vorrebbe aiutarmi ad affrontare il mondo tenendomi sempre per mano.

Al mio papà, che se potesse mi donerebbe un paio d'ali per aiutarmi a spiccare il volo.

Un dolce grazie alla mia cara nonna, che ha sognato ad occhi aperti la mia laurea più di quanto l'abbia fatto io.

Alla mia straordinaria sorella, che veglia su di me da sempre, che non ha mai smesso di credere nelle mie capacità nemmeno per un secondo e che adora di me ogni difetto, più dei pregi.

Alla mia insostituibile Maria che, seppur lontana, è l'orecchio che mi ascolta, la voce che mi rassicura, la spalla su cui piango e l'amica di cui non posso fare a meno.

Al mio prezioso Simone, che mi presta ogni giorno i suoi occhi colmi d'amore e sincera stima per potermi guardare senza il velo delle mie insicurezze.

Infine un particolare ringraziamento agli amici che non mi aspettavo di trovare, ma con cui ho avuto la fortuna di condividere tutto, dai fallimenti più amari alle vittorie più dolci. Li ringrazio dal profondo del mio cuore per avermi regalato tre anni indimenticabili. Per citare Barney Stinson:"Tutto quello che farete nella vita non sarà così leggendario, se non avrete con voi degli amici." (How I Met Your Mother).

Indice

Introduzione

Lo scopo di questo studio è lo sviluppo di un sistema esperto di raccolta dati da esperienze educative. L'implementazione di tale sistema deriva dalla necessità di ottenere il modello digitale di una classe e dei suoi studenti attraverso l'acquisizione di informazioni relative all'approccio adottato per la risoluzione di *task* assegnati dall'insegnante. La riproduzione di un modello digitale accurato richiede l'impiego di strumenti educativi capaci di comunicare con il sistema di raccolta dati esperto e questa tesi ne descrive nel dettaglio l'implementazione. Il sistema esperto permette di sfruttare tecniche di *learning analitics* e *educational data mining* atte al *tuning* iterativo del modello digitale.

Il sistema sviluppato si fonda sulla progettazione e l'impiego di "blocchi parlanti", utilizzabili sull'IDE UIFlow [1] (UIFlow - M5Stack, s.d.), in grado di compiere delle operazioni e allo stesso tempo di comunicarle su un server Discord ([2] Discord, s.d.). E' stato creato un kit di strumenti costituito da i *Talking Blocks*, l'eseguibile python "switchID.py" e gli M5StickC ([4] M5StickC ESP32-PICO Mini IoT Development Kit, s.d.) con il *DIY KIT Rover_StickC.* Tale kit verrà fornito ad una classe in cui l'insegnante dividerà in gruppi gli studenti e si occuperà di personalizzare i *Talking Blocks* per ogni gruppo utilizzando "switchID.py". I *Talking Blocks* personalizzati verrano messi a disposizione degli studenti per la risoluzione dei compiti assegnati. Tali blocchi si metteranno in comunicazione con dei canali testuali Discord [2], opportunamente creati per ogni gruppo di lavoro, scrivendo dei messaggi che riportano l'operazione compiuta dagli studenti, data e ora dell'esecuzione.

L'elaborato si compone di quattro capitoli, così articolati: lo stato dell'arte è riportato nel primo capitolo, all'interno del quale si illustrano le necessità da soddisfare, introducendo le soluzioni adottate. Il secondo capitolo descrive i materiali impiegati quali la scheda programmabile M5StickC [4], l'IDE UIFlow [1] e l'applicazione di Discord [2]; infine una breve digressione sui BOT Discord [2]. Nel terzo capitolo viene spiegato nel dettaglio l'implementazione del sistema di raccolta dati da esperienze educative, illustrando passo dopo passo la progettazione dell'eseguibile "switchID.py", dei *Talking Blocks* e del protocollo per lo scambio dei messaggi che li caratterizza; in ultimo la definizione del ruolo dell'insegnante e dello studente per la realizzazione del collettore esperto di informazioni per la gestione della classe. Il quarto capitolo conclude

l’elaborato presentando un caso studio per l’applicazione degli strumenti prodotti e gli sviluppi futuri previsti per questi ultimi.

1 Stato dell'arte

La robotica è una scienza che racchiude diverse discipline e si occupa della progettazione, programmazione e sviluppo dei robot. Essa trova applicazione nell'ambito educativo, fornendo potenti strumenti di apprendimento per gli studenti; tali strumenti potrebbero riverlarsi anche efficaci mezzi di valutazione per i docenti purché si comprenda come misurare i risultati ottenuti dagli alunni e certificarli opportunamente. La presente tesi ha come obbiettivo quello di proporre una soluzione alla problematica sopracitata e affrontata all'interno dell'abstract "Identification and Assessment of Educational Experiences: Utilizing Data Mining With Robotics" ([3] Scaradozzi, Cesaretti, Screpanti, & Mangina). E' necessaria l'implementazione di un sistema che collezioni e analizzi i dati raccolti fornendo dei risultati comprensibili al personale docente.

E' stato quindi proposto un sistema di raccolta dati da esperienze educative basato sullo sviluppo dei *Talking Blocks*, ovvero di "blocchi parlanti" in grado di svolgere operazioni e di comunicare con l'applicazione di Discord [2] attraverso dei messaggi. Tali messaggi contengono le informazioni fondamentali per la comprensione dell'approccio adottato dagli studenti durante la risoluzione del compito assegnato in classe.

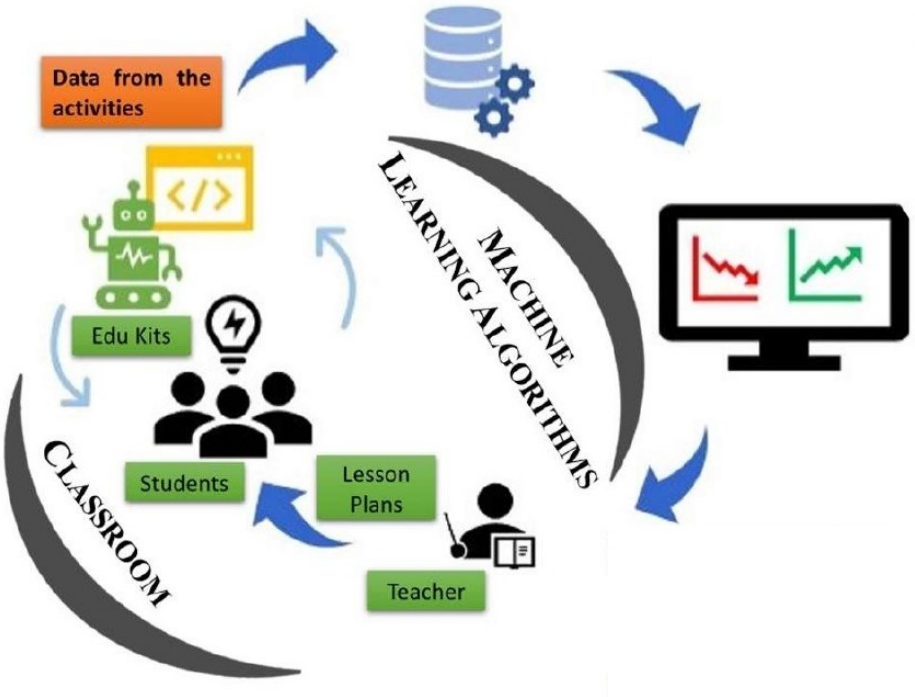


Figura 1: (ER) Infrastruttura delle attività di lavoro

In Figura 1 è rappresentato il modo in cui gli studenti vengono introdotti alla robotica attraverso dei *lesson plans* tenuti dall'insegnante. Gli studenti prendono confidenza con il robot, con il quale interagiscono programmandolo e osservando il risultato delle scelte prese in fase di scrittura del codice. Il robot infine fornisce i dati raccolti durante la lezione per un'analisi approfondita tramite opportune tecniche di *learning analitics*. Le informazioni raccolte, opportunamente elaborate, arricchiscono la valutazione del docente, rendendo maggiormente efficiente il suo metodo di insegnamento.

2 Materiali e Metodi

In questo capitolo sono descritti gli strumenti impiegati e i metodi adottati per l'implementazione del sistema di raccolta dati esperto. Tra gli strumenti forniti allo studente vi sono la scheda di sviluppo programmabile **M5StickC [4]** della **M5Stack** ([5] M5Stack, s.d.) e la piattaforma di programmazione UIFlow [1], all'interno della quale si possono espletare i compiti assegnati dall'insegnante. Quest'ultimo/a tiene traccia dell'attività svolta dallo studente su un opportuno canale testuale nell'applicazione di **Discord [2]**, in cui può prendere visione dei blocchi utilizzati dallo studente all'interno dell'IDE **UIFlow [1]** all'avvio di ogni esecuzione di programma. Si dedicherà l'ultima parte del capitolo ai **BOT** di Discord [2] e al loro potenziale.

2.1 M5StickC

L'M5StickC [4] è una scheda di sviluppo programmabile in blockly attraverso l'apposito IDE UIFlow [1], in C con l'IDE Arduino ([6] Arduino IDE, s.d.) e in MicroPython; esso presenta:

- dimensioni 5x2.5 cm
- display da 0,96 pollici (risoluzione 160x80)
- microcontrollore ESP32
- microfono
- buzzer
- trasmettitore IR
- WiFi
- Bluetooth
- accelerometro
- giroscopio a 6 gradi di libertà
- LED di segnalazione integrato

- pulsanti A e B programmabili
- batteria da 80 mAh
- memoria flash da 4MB
- modalità di programmazione via USB e WiFi

Grazie all'interfaccia GROVE e GPIO, è possibile collegarvi numerosi dispositivi esterni disponibili sullo store ufficiale di M5Stack [5].

Per accendere il dispositivo bisogna esercitare una pressione di 2 secondi sul *power button*, che si trova a sinistra; alla comparsa del logo di M5StickC [4], è possibile accedere al menù di *setting* premendo il pulsante centrale (M5), ci si può spostare all'interno di quest'ultimo servendosi del pulsante in alto a destra. Per impostare una modalità di programmazione, si accede al menù *Setup* in cui si ha la possibilità di selezionare l'*USB Mode* o la *WiFi mode*.

L'M5StickC [4] si presenta dunque come uno strumento particolarmente versatile, intuitivo e con buone prestazioni a un prezzo accessibile.

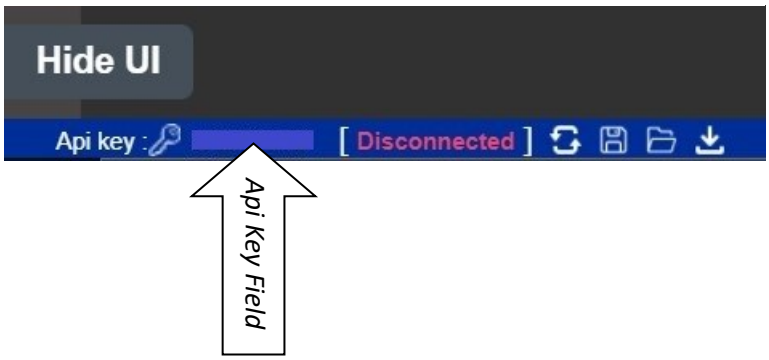
Figura 2 : M5StickC [4]

2.2 UIFlow

UIFlow [1] è la piattaforma di programmazione progettata per i dispositivi M5Stack [5], può essere utilizzato on-line o localmente scaricandolo dal sito ufficiale di M5Stack [5] e presenta una prima interfaccia dedicata alla programmazione in blockly e una seconda che permette di visualizzare il codice MicroPython corrispondente.

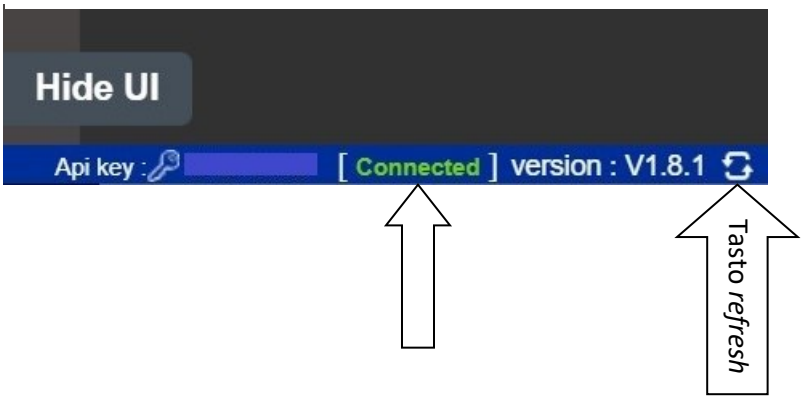
Per poter programmare all'interno dell'IDE ed eseguire le sequenze di blocchi prodotte, è necessario collegare l'M5StickC [4] in modalità USB scrivendo nell'apposito campo l'*Api key* del dispositivo (Figura 3),

Figura 3: Api Key Field



oppure in modalità WiFi cliccando sul tasto *refresh* e assicurandosi che il dispositivo sia connesso (Figura 4).

Figura 4: Verifica della connessione WiFi del dispositivo



Una volta connesso l'M5StickC [4], si possono produrre sequenze di blocchi trascinandoli all'interno dell'area di programmazione, collegandoli al blocco *Setup*.

UIFlow [1] offre inoltre la possibilità di creare nuovi pacchetti di blocchi accedendo alla sezione *Custom*, cliccando su *Create *.m5b file* e accedendo a **UIFlow Block Maker** ([7] UIFlow Block Maker, s.d.) (Figura 5); in quest'area di sviluppo è possibile progettare blocchi di tipo *Execute* e di tipo *Value*, passando dei parametri di tipo *Label*, *String*, *Number* o *Variable* e scrivendo il codice all'interno dello spazio *Block Code* (Figura 6). Questa *feature* si è rivelata di fondamentale importanza per la progettazione di blocchi personalizzati in grado di imitare il comportamento di blocchi esistenti, con l'aggiunta di un protocollo per lo scambio di messaggi in grado di soddisfare la necessità di tenere traccia del tipo di blocchi utilizzati ad ogni esecuzione dallo studente. L'implementazione dei *Talking Blocks* è trattata nello specifico all'interno del Capitolo 3 di questa tesi.

Figura 5: Sezione Custom

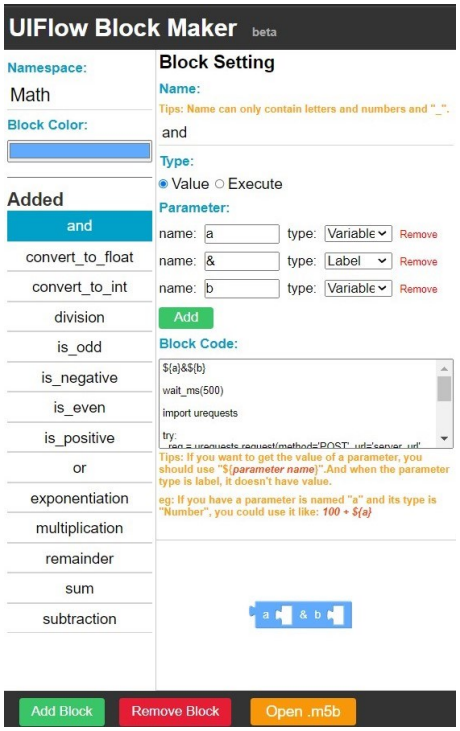


Figura 6: UIFlow [1] Block Maker

2.3 Discord

Discord [2] è una piattaforma di VoIP, messaggistica istantanea e distribuzione digitale che nasce con l'obbiettivo di favorire la comunicazione tra comunità di videogiocatori. Gli utenti possono comunicare con chiamate vocali, videochiamate e messaggi di testo in chat private oppure come membri di un server all'interno dei canali vocali e/o dei canali testuali. L'applicazione funziona su *Windows*, *macOS*, *Android*, *iOS*, *iPadOS*, *Linux* e nei browser web. L'interfaccia intuitiva e le numerose *features* lo rendono adatto all'obbiettivo che questa tesi si propone di soddisfare ed è stato per cui adottato come strumento di implementazione di una raccolta dati esperta, sfruttando i suoi server e i canali testuali degli stessi per acquisire le informazioni relative allo studente sottoforma di messaggi.

Figura 7: Discord [2] Logo

Nel Capitolo 3 è contenuta una trattazione approfondita dell'uso dell'applicazione ai fini della tesi.

2.4 BOT Discord

I BOT Discord sono programmi autonomi in grado di svolgere numerose funzionalità. Discord propone diversi BOT per la gestione dei server e per l'intrattenimento degli utenti, offrendo anche la possibilità a questi di personalizzarli o di crearne di nuovi.

I BOT possono essere aggiunti ad un server accedendo ad un vasto catalogo fornito da siti come ***discordbotlist.com*** ([8] Discord Bot List, s.d.).

I BOT sono stati considerati inizialmente come strumento intermedio tra l'M5StickC [4] e Discord [2] per il reperimento dei messaggi in forma anonima. La soluzione infine adottata viene descritta nel dettaglio nel Capitolo 3.

Si propone di creare un BOT in grado di trasferire i messaggi dal canale testuale ad un file di testo, automatizzando l'ultima fase della raccolta dati esperta. Nella Figura 8 è riportato un esempio di BOT gratuito che offre la possibilità di eseguire dei backup sia manuali che automatici dei server.

Figura 8: Icona del Discord BOT Xenon ([9] Merlin#1337, s.d.)

3 Implementazione del sistema di raccolta dati da esperienze educative

Il seguente capitolo descrive nel dettaglio l'implementazione del sistema di raccolta dati tratti da esperienze educative. Tale implementazione mira ad ottenere un file di log contenente le informazioni relative ai blocchi utilizzati dallo studente all'interno di UIFlow [1] ad ogni esecuzione effettuata. Sono stati sviluppati di conseguenza dei *Talking Blocks*, ovvero dei blocchi in grado di comunicare l'operazione da essi compiuta, inviando un messaggio ad un canale testuale di Discord [2].

3.1 Implementazione dei Talking Blocks

I *Talking Blocks* sono stati implementati su UIFlow Block Maker [7], con l'obiettivo di imitare il comportamento di blocchi esistenti su UIFlow [1] con l'aggiunta di una porzione di codice dedicata ad un protocollo per lo scambio di messaggi con i canali testuali di Discord [2].

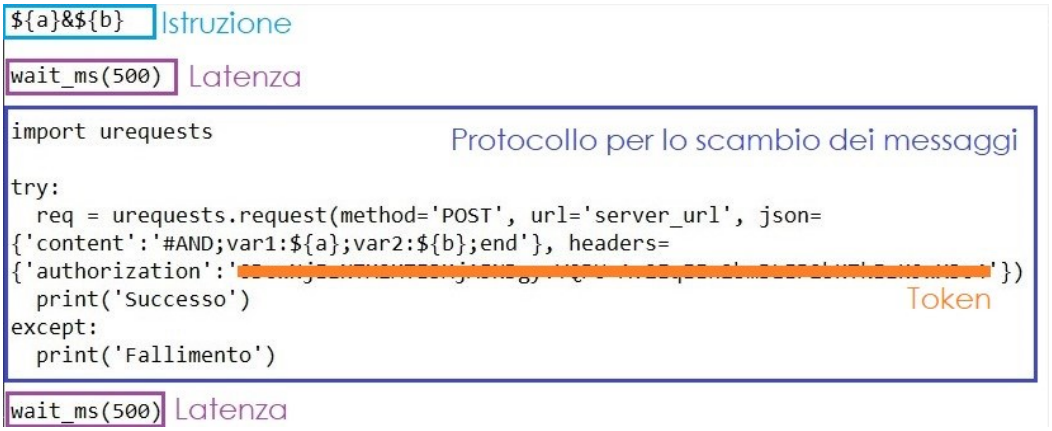


Figura 9: Struttura del codice in MicroPython di un Talking Block

In Figura 9 è rappresentata la struttura classica del codice in MicroPython di un *Talking Block* ed è caratterizzata da:

- Una parte dedicata all'operazione specifica del blocco. Nel caso riportato si ha l'operazione logico-matematica "AND";
- Una parte dedicata al protocollo per lo scambio di messaggi che verrà descritto nel dettaglio nel paragrafo successivo (3.2);
- Due latenze poste agli estremi del protocollo per lo scambio dei messaggi.

E’ stato osservato che in caso di esecuzione di lunghe sequenze di blocchi, l’M5StickC [4] tende ad ignorare il protocollo per lo scambio di messaggi; di conseguenza sono state inserite tali latenze le quali prevengono nella maggior parte dei casi il fenomeno.

Si propone di indagare le cause scatenanti del fenomeno poiché l’inserimento di latenze rallenta l’esecuzione delle sequenze di blocchi.

Di seguito sono riportati i risultati di tre tipi di test svolti sfruttando i blocchi personalizzati *LED ON* e *LED OFF.* Ogni test è stato effettuato sfruttando su UIFlow [1] una sequenza composta da un totale di 10 blocchi, di cui 5 *LED ON* e 5 *LED OFF* fra di loro alternati.

Figura 10: Sequenza di blocchi LED ON/LED OFF

In Figura 11 è riportato il primo test. E’ stato effettuato riproducendo su UIFlow [1] la sequenza riportata in Figura 10, con l’aggiunta di una latenza alla fine del codice del blocco. Sono state avviate 10 esecuzioni consecutive con lo scopo di rilevare quante volte i messaggi vengono inviati correttamente.

Il test presenta una percentuale di successo del 40%, ovvero le esecuzioni in cui i messaggi sono stati ricevuti correttamente sono solo 4 su 10.

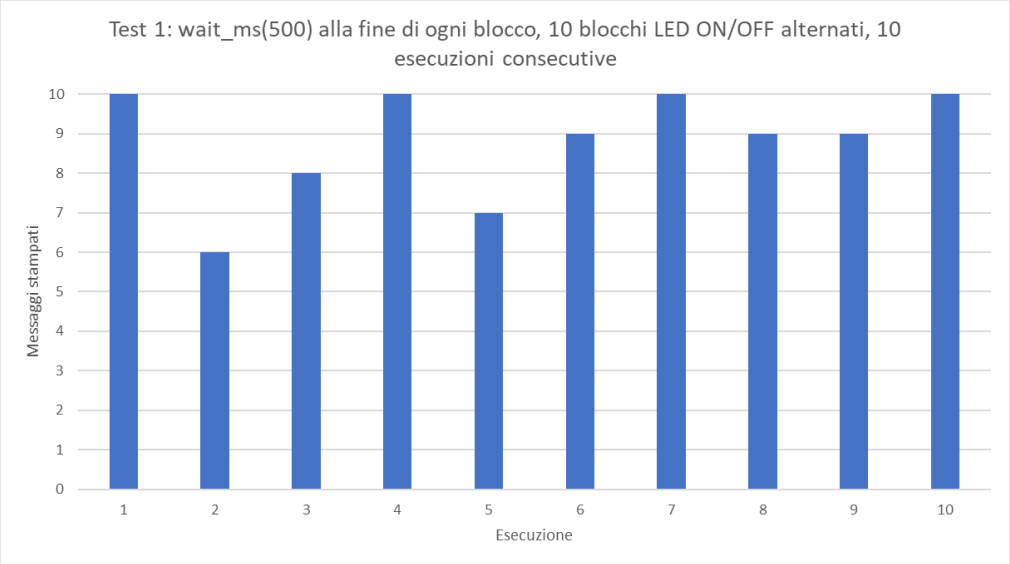


Figura 11: Test 1

Supponendo che l'errore fosse causato da UIFlow [1], è stato condotto un secondo test le cui esecuzioni sono precedute dal riavvio dell'IDE.

In Figura 12 è riportato il secondo test. E' stato effettuato utilizzando la sequenza in Figura 10, con l'aggiunta di una latenza alla fine del codice del blocco. Le esecuzioni, come anticipato, sono state precedute dal riavvio dell'IDE UIFlow [1].

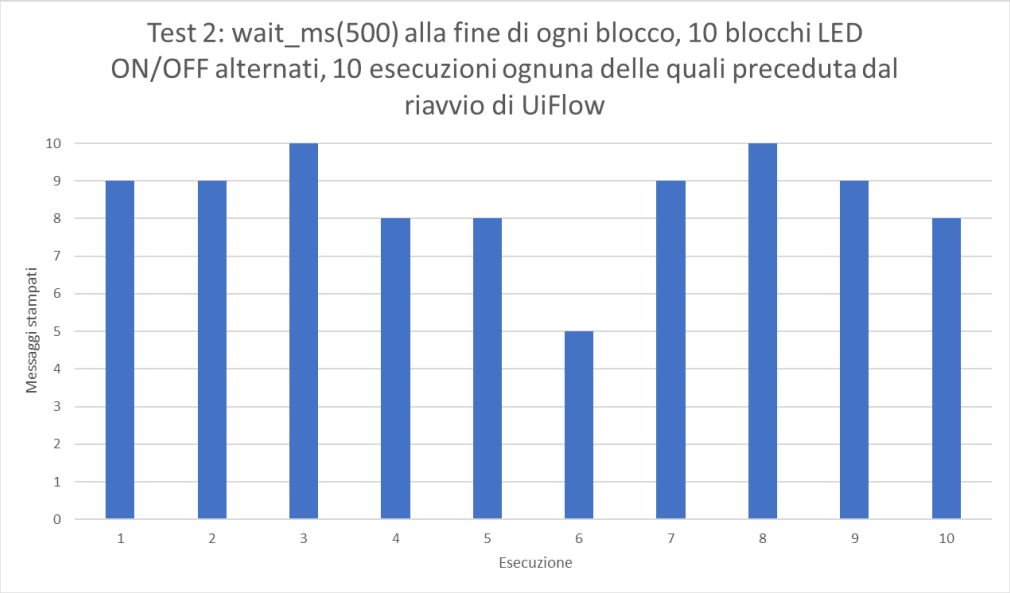


Figura 12: Test 2

Il secondo test presenta una percentuale di successo del 20%, ovvero le esecuzioni in cui i messaggi sono stati ricevuti correttamente sono soltanto 2 su 10. Il terzo test è stato di conseguenza effettuato senza riavviare UIFlow [1] ad ogni esecuzione.

In Figura 13 è riportato il terzo test. E' stata aggiunta una latenza all'inizio e alla fine del codice del blocco e le esecuzioni sono state avviate consecutivamente.

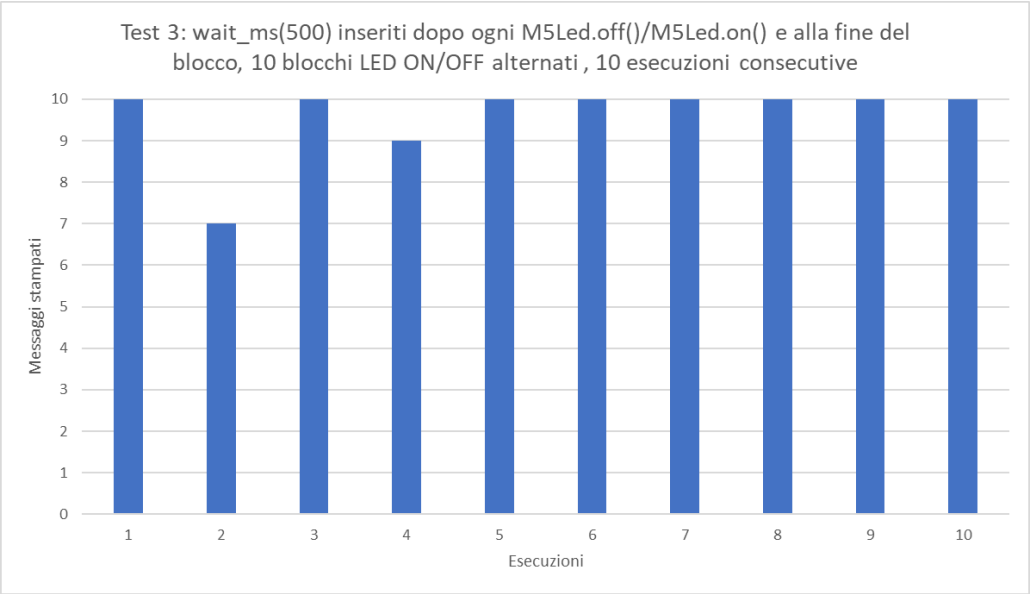


Figura 13: Test 3

Il terzo test presenta una percentuale di successo dell'80%, ovvero le esecuzioni in cui i messaggi sono stati ricevuti correttamente sono 8 su 10.

La percentuale di successo del terzo test ha portato alla conclusione che l'aggiunta di latenze agli estremi del protocollo per lo scambio dei messaggi, riduce in maniera significativa il mancato reperimento dei messaggi. Il risultato ottenuto è stato considerato soddisfacente ai fini di questa tesi; tuttavia occorre precisare la scarsa validità ai fini statistici dei suddetti test, in quanto si fa riferimento ad un campione esiguo.

3.2 Implementazione del protocollo per lo scambio dei messaggi

Il protocollo per lo scambio dei messaggi imita il comportamento dei blocchi *Http* disponibili su UIFlow [1] nella sezione *Advanced.* Il protocollo esegue una *POST* all'URL specificato nell'apposito campo, tuttavia nel caso riportato in Figura 14 non è presente alcun indirizzo URL, in quanto questo viene fornito al momento della personalizzazione

dei blocchi attraverso il programma python “switchID.py” che verrà analizzato nel paragrafo 3.3. Nel campo *content* è contenuto il messaggio stesso mentre nel campo *authorization* è inserito un *token*.

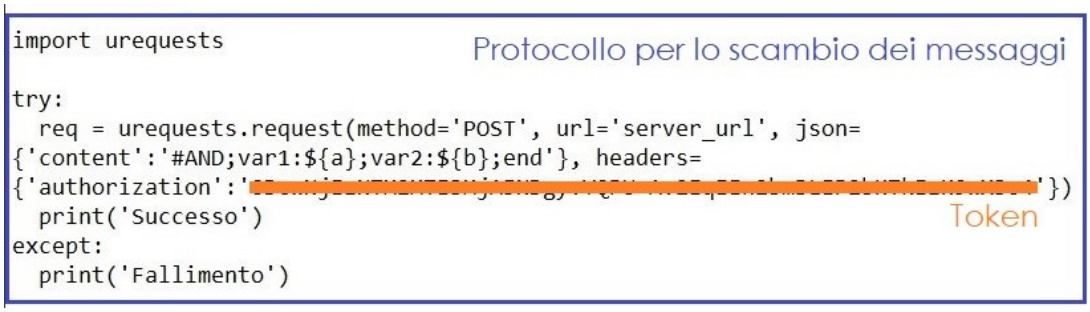


Figura 14: Protocollo per lo scambio dei messaggi

E’ stato creato un apposito account Discord [2] per l’M5StickC [4], per poi utilizzare il *token* ad esso associato. Il *token* è un codice alfanumerico univoco associato ad ogni account Discord [2] ed è l’informazione che permette all’M5StickC [4] di inviare messaggi come un utente umano con il nome “UnivPM.TalkingBlocks.M5”. Il *token* si ottiene nella seguente maniera:

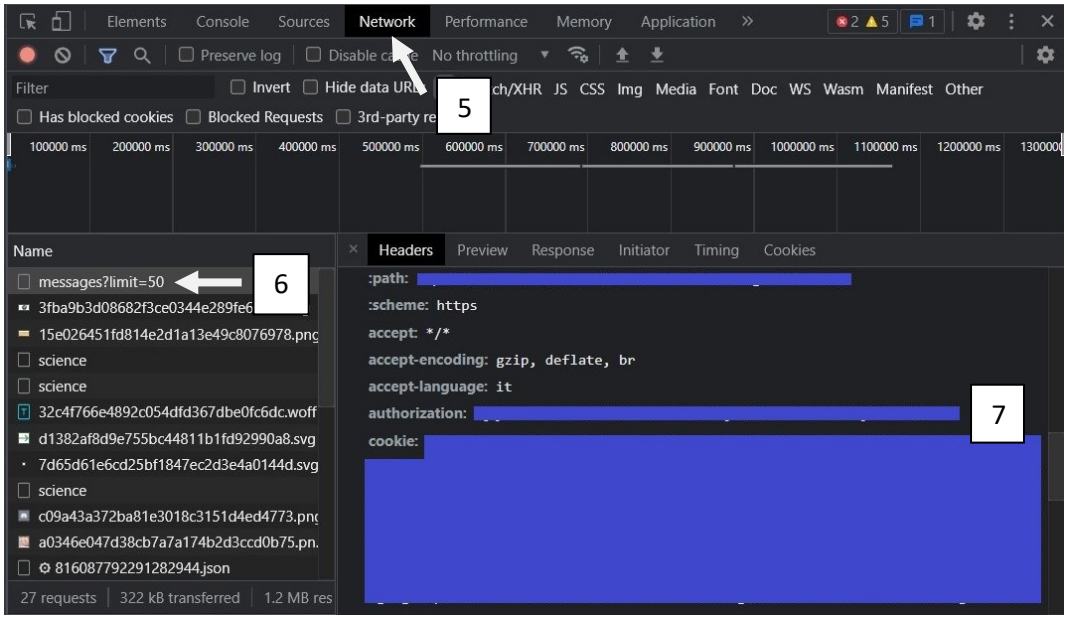


Figura 15: Chrome DevTools ([10] Chrome DevTools [10], s.d.)

1) Accedere all’account Discord [2] creato per l’M5StickC [4] su un browser web;
2) Cliccare sul tasto “impostazioni” del browser web;
3) Selezionare “Altri strumenti” e subito dopo “Strumenti per sviluppatori”;

4) Una volta aperto il *DevTools*, scrivere un messaggio di testo in un canale testuale;

5) Accedere alla sezione “Network” di *DevTools* (numero 5 in Figura 15);

6) Cliccare sulla riga “messages” (numero 6 in Figura 15);

7) All’interno della sottosezione “Headers” è possibile ricercare il campo “authorization”, all’interno del quale è specificato il *token* (numero 7 in Figura 15).

La creazione di un apposito account per l’M5StickC [4] ha permesso di preservare la privacy degli studenti e di proteggere le loro informazioni sensibili.

3.3 switchID.py

Il programma “switchID.py” è stato implementato per personalizzare pacchetti di blocchi inserendo nel loro protocollo per lo scambio di messaggi l’URL di un canale testuale Discord [2].

switchID.py

Phyton > switchID.py

```
2
3   text_channel_id=input("Scrivere l'ID del canale testuale che si intende associare ai blocchi di Blockly: ")
4
5   texttofind = 'server_url'
6   texttoreplace = "https://discord.com/api/v9/channels/" + text_channel_id + "/messages"
7   sourcepath = os.listdir('Blocchetti_M5/')
8
9   for file in sourcepath:
10      inputfile = 'Blocchetti_M5/'+ file
11      print('Conversion is ongoing for:' +inputfile)
12      with open(inputfile, 'r') as inputfile:
13          filedata = inputfile.read()
14          freq = 0
15          freq = filedata.count(texttofind)
16          destinationpath = 'OutputFile/' + file #modificare il nome della cartella di destinazione all'occorrenza
17          filedata = filedata.replace(texttofind,texttoreplace)
18          with open(destinationpath,'w') as file:
19              file.write(filedata)
20              print('Total %d Record Replace' %freq)
21              print(" \n")
22              print("L'ID inserito e': " + text_channel_id + "\n" + "I blocchi personalizzati si trovano nella cartella OutputFile")
23
```

Figura 16: switchID.py da Visual Studio Code ([11] Visual Studio Code, s.d.)

Quando “switchID.py” viene eseguito, procede in questo modo:

1) Richiede come input l’ID di un canale testuale;

2) Forma un URL servendosi dell’ID precedentemente ottenuto;

3) Legge il codice dei blocchi all’interno della cartella “Blocchetti_M5/” (Figura 16) e scrive l’URL ottenuto al posto della parola “server_url”;

4) Salva i blocchetti modificati nella cartella “OutputFile/” (Figura 16).

Al termine dei 4 passi, si ottengono dei *Talking Blocks* in grado di comunicare con il canale testuale scelto. Il codice è stato scritto considerando che le cartelle "Blocchetti_M5/" e "OutputFile/" si trovino nella stessa directory dell'eseguibile, di conseguenza si consiglia di fornire il kit di personalizzazione dei blocchi con l'eseguibile nella stessa directory delle due cartelle di lavoro.

3.4 Implementazione del collettore esperto di informazioni per la gestione della classe

I ruoli che permettono di definire l'implementazione del collettore esperto di informazioni per la gestione della classe sono quello dell'insegnante e dello studente.

L'insegnante

L'insegnante deve:

1) Creare un nuovo server per la classe, cliccando sul tasto verde "+" a sinistra (Figura 17);
2) Creare un canale testuale per ogni studente, cliccando sul tasto grigio "+" (Figura 17);
3) Eseguire il programma "switchID.py" per ottenere un pacchetto di blocchi personalizzato per ogni studente;
4) Importare i pacchetti sui computer utilizzati dagli studenti;
5) Collegare gli M5StickC [4] degli studenti ad una rete WiFi;
6) Caricare su UIFlow [1] i *Talking Blocks* nei pacchetti.

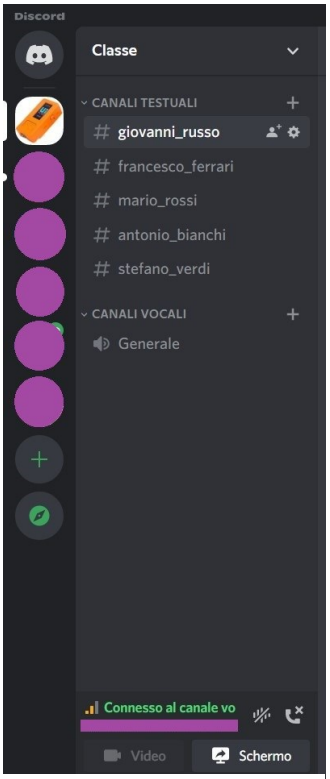


Figura 17: Server e canali testuali

Lo studente

Lo studente deve:

1) Seguire le indicazioni dell'insegnante e svolgere gli esercizi che gli verranno assegnati;

2) Assicurarsi di aggiungere il blocco *END* alla fine di ogni sequenza di blocchi creata su UIFlow [1].

Ogni volta che lo studente esegue un programma su UIFlow [1], i *Talking Blocks* comunicano di essere stati utilizzati con un messaggio scritto in un'opportuna notazione. Si può individuare la fine di una sequenza di *Talking Blocks* attraverso il messaggio che riporta due cancelletti (blocco *END*).

Figura 18: Talking Blocks

Durante lo svolgimento delle esperienze educative in classe, verranno raccolte le informazioni riguardanti i blocchi utilizzati dagli studenti all'interno dei canali testuali Discord [2] di ognuno. I dati raccolti sono pronti per essere importati in un file *.txt per analizzarli. Si propone di automatizzare l'importazione dei messaggi su file *.txt.

4 Verifica e validazione: rappresentazione di un caso studio per l'applicazione degli strumenti

Per procedere alla verifica e alla validazione del modello del sistema di raccolta dati esperta proposto, si riporta un caso studio per l'applicazione degli strumenti.

L'insegnante pianifica le esperienze educative tenendo 4 lezioni servendosi di slide *.pptx :

- **Corso pregresso di geometria base**: spiegazione di nozioni fondamentali per la conoscenza e la comprensione dei concetti di cerchio, circonferenza e raggio.
- **Corso pregresso di STrEM**: spiegazione di nozioni fondamentali per la conoscenza e la comprensione del funzionamento dei robot. Tra i concetti chiave vi sono i sensori, gli attuatori e l'intelligenza artificiale descritti come i mezzi attraverso i quali il robot percepisce l'ambiente circostante, lo manipola o vi si muove attraverso e lo analizza così come rispettivamente i cinque sensi, i muscoli e il cervello permettono di fare altrettanto ad un essere umano.
- **Lezione 1**: costruzione del robot con l'assistenza e la supervisione dell'insegnante. La costruzione del robot consiste nel montare il *DIY KIT Rover_StickC*.
- **Lezione 2**: esercizi di programmazione con il robot.
    - **Esercizio A**: Il robot deve percorrere un metro. L'esercizio è correttamente completato se il robot si arresta a meno di 2,5 cm dalla distanza data.
    - **Esercizio B**: Il robot deve fermarsi ad una data distanza da un ostacolo. Gli studenti si servono del sensore ad ultrasuoni del robot per espletare il compito.
- **Lezione di verifica**: competizione a squadre. Si propone agli studenti, divisi in gruppi eterogenei, di montare il robot e risolvere due esercizi simili a quelli elencati nella lezione 2 in maniera autonoma entro un certo tempo; sono concessi 5 tentativi.

Il caso studio replica il protocollo di *Educational Data Mining (EDM) and Learning Analytics (LA)* descritto nell'abstract "Utilizing data mining with Robotics for identification and assessment of educational experiences" ([3] Scaradozzi, Cesaretti, Screpanti, & Mangina).

Figura 19: M5StickC [4] con DIY KIT BalaC ([12] DIY KIT BalaC, s.d.)

Per la realizzazione delle lezioni sarà necessario il seguente materiale:

1) Corso per i docenti
    a. Per la formazione del modello digitale della classe su Discord [2] e quindi la creazione di un server della classe e di canali testuali per ogni gruppo di studenti;
    b. Per il corretto utilizzo del materiale didattico affinché essi siano in grado di personalizzare i *Talking Blocks* con "switchID.py", importarli su i terminali degli studenti, caricarli su UIFlow [1], montare il *DIY KIT Rover_StickC,* attivare la modalità WiFi dell'M5StickC [4];
    c. Per l'archiviazione delle evidenze digitali raccolte durante le lezioni e registrate su Discord [2].
2) Schede ad ausilio dei docenti come materiale di supporto per le lezioni:
    a. Slide per la costruzione guidata del robot;
    b. Slide per la programmazione guidata del robot;
    c. Slide di spiegazione degli esercizi A e B precedentemente descritti.

3) Kit robotico costituito dall'M5StickC [4] e il *DIY KIT Rover_StickC.* Ogni gruppo di studenti avrà a disposizione un kit;

4) Librerie *custom* di UIFlow [1], ad ausilio degli/lle insegnanti, contenenti i *Talking Blocks*. Tali librerie sono opportunamente personalizzate dagli/lle insegnanti attraverso "switchID.py";

5) Programmi di esempio per la risoluzione degli esercizi A e B, ad ausilio degli/lle insegnanti;

6) Schede di valutazione ad ausilio degli/lle insegnanti per la verbalizzazione dell'andamento.

4.1 Sviluppi Futuri

Il corso per i docenti sarà erogato sottoforma di brevi video-tutorial che spiegano le modalità di utilizzo di Discord [2] e del materiale didattico. L'archiviazione delle evidenze digitali, al momento, può essere esplicata importando manualmente i messaggi testuali all'interno di un file *.txt. Si prevede di fornire ai docenti delle apposite schede di valutazione che permettano di ottenere un riscontro dettagliato dell'approccio adottato dagli studenti nella risoluzione degli esercizi. Le schede di valutazione permetteranno di arricchire e migliorare l'analisi dei dati raccolti.

Si riporta, alla fine del capitolo corrente, l'insieme dei *Talking Blocks* finora progettati in collaborazione con Marco Cervigni, che affronta nel dettaglio gli aspetti tecnici e implementativi del sistema di raccolta dati esperto nel suo elaborato ([13] Cervigni, A.A 2020/2021).

Lorenzo Cichella ([14] Cichella, A.A 2020/2021) si è occupato di estendere la libreria *custom* UIFlow [1] attraverso l'aggiunta di blocchi per la risoluzione degli esercizi A e B precedentemente citati, scrivendo di ognuno il codice risolutivo di esempio ad ausilio degli/le insegnanti. I risultati da lui prodotti hanno permesso la creazione del kit robotico e delle schede per le lezioni ad ausilio del personale docente [14].

Operazione blocco UIFLOW	Modificabile?	Motivazione (se non lo è)	Messaggi Discord

Led ON	Si	/	#led_on;end
Led OFF	Si	/	#led_off;end
Somma Sottrazione Prodotto Divisione	Si	/	#sum;var1: ;var2: ;end #subtraction;var1: ;var2: ;end #multiplication;var1: ;var2: ;end #division;var1: ;var2: ;end
AND OR	Si	/	#AND;var1: ;var2: ;end #OR;var1: ;var2: ;end
Resto della divisione	Si	/	#remainder;var1: ;var2: ;end
È pari? È dispari?	Si	/	#is_even;var1: ;end #is_odd;var1: ;end
È positivo? È negativo?	Si	/	#is_positive;var1: ;end #is_negative;var1: ;end
È divisibile per?	No	Non si può importare la libreria “math” in un blocco di tipo “value”: la variabile deve essere inizializzata prima di poterle assegnare un valore.	/
Random fraction	No	Non si può importare la libreria “math” in un blocco di tipo “value”: la variabile deve essere inizializzata prima di poterle assegnare un valore.	/
Radice quadrata Valore assoluto Logaritmo naturale	No	Non si può importare la libreria “math” in un blocco di tipo “value”: la	/

		variabile deve essere inizializzata prima di poterle assegnare un valore.	
Seno Coseno Tangente	No	Non si può importare la libreria “math” in un blocco di tipo “value”: la variabile deve essere inizializzata prima di poterle assegnare un valore.	/
Converti in int	Si	/	#convert_to_int;var1: ;end
Converti in float	Si	/	#convert_to_float;var1: ;end
Adc init Adc read	Si	/	#init_adc0;var1: ;end / #init_adc1;var1: ;end #adc0_read_value;end / #adc1_read_value;end
Dac init Dac write	Si	/	#init_dac0;var1: ;end / #init_dac1;var1: ;end #write_value_dac0;var1: ;end / #write_value_dac1;var1: ;end
Repeat ‘x’ times, repeat while	No	Un’operazione di tipo “execute” non può essere inserita come parametro di un blocco.	/
If do, else if do	No	Un’operazione di tipo “execute” non può essere inserita come parametro di un blocco.	/

Try except	No	Un’operazione di tipo “execute” non può essere inserita come parametro di un blocco.	/
Switch case	No	Un’operazione di tipo “execute” non può essere inserita come parametro di un blocco.	/
Wait secondi	Si	/	#wait_seconds;var1: ;end
Wait millisecondi	Si	/	#wait_milliseconds;var1: ;end

Conclusione

Questa tesi si è occupata di sviluppare un sistema esperto di raccolta dati da esperienze educative. Prima di giungere alla soluzione descritta, è stata considerata la possibilità di impiegare all'interno dei *Talking Blocks* una libreria per la creazione di un file di log sotto forma di file *.txt, con l'obbiettivo di importarlo successivamente su un cloud per la fase di archiviazione e analisi dei dati. Tuttavia questa prima configurazione presentava le seguenti problematiche:

- Necessità di importare librerie esterne sull'M5StickC [4], modificando la memoria flash;
- Impossibilità di salvare il file *.txt nella memoria dell'M5StickC [4].

Per sopperire al bisogno di scrivere un file di log e contemporaneamente archiviarlo sottoforma di testo su un server, si è pensato di far sì che fosse l'M5StickC [4] stesso ad inviare i dati raccolti seguendo uno dei protocolli di messaggistica disponibili su UIFlow [1]. Il protocollo MQTT è stato dapprima considerato per poi essere escluso, dato che la sua principale caratteristica è lo sfruttamento di un modello *client/server* tramite l'utilizzo di *message broker* di terze parti. Il protocollo *Http* si è invece rivelato una soluzione istantanea e intuitiva poiché sfrutta il *method POST* applicandolo ad un URL scelto. E' bastato poi inserire il protocollo *Http* nel codice di ogni "blocco parlante", facendo sì che ciasun *Talking Block* comunicasse l'operazione da esso compiuta. Dal bisogno di poter indicare data e ora dei dati raccolti si è deciso di utilizzare la piattaforma di messaggistica istantanea di Discord [2]. Esso è caratterizzato da un'interfaccia *user-friendly* che permetta ai docenti delle scuole primarie di utilizzarlo senza particolari conoscenze pregresse.

L'elaborato si conclude proponendo dei possibili *upgrade* del modello di collettore esperto implementato. Per un'accurata verifica funzionale, si suggerisce una prima applicazione del caso studio riportato nel capitolo 4, per poi creare un set significativo di casi studio per la validazione dello strumento. La varietà dei casi studio può essere ampliata attraverso lo sviluppo di nuovi *Talking Blocks* che possano popolare le librerie attualmente progettate. A seguito di una validazione soddisfacente, si può procedere mettendo in atto le tecniche di *learning analitics* e di *educational data mining* per ottenere il *tuning* iterativo del modello digitale.

Sitografia e Bibliografia

[1] UIFlow - M5Stack. (s.d.). Tratto da M5Flow: https://flow.m5stack.com/ - visita aggiornata al 22/10/2021

[2] Discord. (s.d.). Tratto da Discord: https://discord.com/ - visita aggiornata al 22/10/2021

[3] Scaradozzi, D., Cesaretti, L., Screpanti, L., & Mangina, E. (s.d.). Identification and Assessment of Educational Experienxes: Utilizing Data Mining With Robotics. *IEEE Robotics & Automation Magazine*.

[4] M5StickC ESP32-PICO Mini IoT Development Kit. (s.d.). Tratto da M5Stack Store: https://shop.m5stack.com/products/stick-c?variant=17203451265114 - visita aggiornata al 22/10/2021

[5] M5Stack. (s.d.). Tratto da M5Stack: https://m5stack.com/ - visita aggiornata al 22/10/2021

[6] Arduino IDE. (s.d.). Tratto da Software | Arduino: https://www.arduino.cc/en/software - visita aggiornata al 22/10/2021

[7] UIFlow Block Maker. (s.d.). Tratto da M5BlockMaker: http://block-maker.m5stack.com/ - visita aggiornata al 22/10/2021

[8] Discord Bot List. (s.d.). Tratto da Discord Bot | Discord Bot List: https://discordbotlist.com/ - visita aggiornata al 22/10/2021

[9] Merlin#1337. (s.d.). *Xenon*. Tratto da Xenon - Discord Bots: https://discord.bots.gg/bots/416358583220043796 - visita aggiornata al 22/10/2021

[10] Chrome DevTools. (s.d.). Tratto da Chrome DevTools - Chrome Developers: https://developer.chrome.com/docs/devtools/ - visita aggiornata al 22/10/2021

[11] Visual Studio Code. (s.d.). Tratto da Visual Studio Code - Code Editing Redefined: https://code.visualstudio.com/ - visita aggiornata al 22/10/2021

[12] DIY KIT BalaC. (s.d.). Tratto da BALA-C ESP32 Development Mini Self-Balancing Car: https://shop.m5stack.com/products/bala-c-esp32-development-mini-self-balancing-car?variant=32128285409370 - visita aggiornata al 22/10/2021

[13] Cervigni, M. (A.A 2020/2021). *Studio e Implementazione di Strumenti di Robotica Educativa per l'Identificazione e la Modellazione dell'Apprendimento.* Università Politecnica delle Marche.

[14] Cichella, L. (A.A 2020/2021). *Studio e Progettazione di Kit Educativi Parlanti.* Università Politecnica delle Marche.

Elenco delle figure