Code to Learn in una Scuola Primaria: il Progetto COGITO

Alessandro Ricci[1], Angelo Croatti[1], Laura Tarsitano[1], Paolo Venturi[1]
Giuseppe Messina[2], Arminda Iavarone[2], Renata Righi[2], Maria Capizzi[2],
Bruna Borgognoni[2], Flavia Persico[2], Manuela Pollini[2], Andrea Vaccari[3]

[1] DISI - CRIAD - ISI - Università di Bologna, Campus di Cesena
{a.ricci|a.croatti}@unibo.it, l.tarsitano@criadcoding.it
[2] Direzione Didattica III Circolo Didattico Cesena – Scuola Primaria "Carducci"
[3] FabLab Romagna — posta@fablabromagna.org

Abstract. Pensiero Computazionale e Coding sono oggi al centro di importanti iniziative in atto nelle Scuole - sia a livello nazionale, sia internazionale - che hanno come obiettivo l'insegnamento della programmazione e delle basi dell'informatica come competenza di base. In questo articolo descriviamo e discutiamo il progetto COGITO, iniziativa promossa e realizzata da una gruppo di insegnanti e ricercatori universitari in una Scuola Primaria, con l'obiettivo di costruire l'apprendimento dei principi del Pensiero Computazionale e del Coding come *meta-competenza*, esplorandone concretamente la valenza in relazione alle altre competenze e attività curricolari.

1 Introduzione

Mai come oggi, *Pensiero Computazionale* (PC nel prosieguo) e *coding* sono al centro di iniziative sia a livello nazionale, sia a livello internazionale, volte ad introdurre principi e pratiche dell'informatica e della programmazione nelle Scuole di ogni livello e grado, come quarta abilità di base. Nella comunità scientifica, il concetto di PC è passato alla ribalta con un articolo di J. Wing del 2006 [17] ed è tuttora oggetto di dibattito all'interno comunità, in termini di definizioni e caratterizzazioni, sia in relazione alla scuola [1,8], sia più in generale nella riflessione relativa all'informatica come competenza fondamentale [3,4,9].

Al di là delle elucubrazione della comunità scientifica, l'impatto maggiore più concreto a livello di società e istituzionali lo stanno avendo iniziative di organizzazioni/comunità internazionali come *Code.org*, che fornisce percorsi dedicati a insegnanti e studenti e con un approccio orientato al problem solving, oppure CAS (Computing At School), che supporta attività, progetti, definizione di percorsi curricolari per portare l'informatica nelle scuole primarie e secondarie, avendo come scopo "to promote and support excellence in computer science education" [4]. In Italia, le principali iniziative volte a far crescere le competenze digitali e a diffondere il pensiero computazionale all'interno delle scuole, invece, sono state la legge della Buona Scuola e Programma il Futuro. Quest'ultima iniziativa, avviata dal MIUR in collaborazione con il Consorzio Interuniversitario Nazionale per l'Informatica (CINI), è nata proprio per incoraggiare la

[4] https://www.computingatschool.org.uk/

diffusione del coding nella scuola primaria e secondaria, avendo il materiale di Code.org come riferimento.

In queste iniziative è prevalente una visione dell'informatica come competenza, importante per vari livelli, non ultimo quello del lavoro. Tuttavia questa è solo una parte della visione più ampia in merito al pensiero computazionale e al coding introdotta da pionieri quali Seymour Papert [11], Cynthia Solomon [16], Alan Kay [7] e la Scuola Costruzionista [6], a partire dalla fine degli anni 1960. Questa visione, che continua nel lavoro fra gli altri di Mitchel Resnick e colleghi del MIT Lifelong Kindergarten, è ben riassunta dal motto *learn to code, code to Learn*, che mette al centro il ruolo del coding e del PC come strumento concettuale e operativo per costruire ambienti e processi di apprendimento e di formazione.

Su questa base, a partire da fine 2015, è stato messo in piedi il progetto COGITO [5], una iniziativa triennale in una Scuola Primaria che visto la cooperazione di una squadra multi-disciplinare composta da insegnanti, docenti e ricercatori dell'Università (Dipartimento di Informatica DISI, UniBO e parte del CRIAD, Centro di Ricerche e Studi sull'Informatica applicata alla Didattica) e FabLab. La chiave innovativa del progetto – rispetto al panorama sia nazionale, sia internazionale – è stata quello di mettere in piedi e sperimentare sin dal principio una visione sinergica fra pensiero computazionale e pensiero psico-pedagogico, in cui esplorare il ruolo del coding come una meta-competenza, oltre che quarta abilità di base, da applicare alle altre competenze.

In questo lavoro, dopo una breve panoramica dei punti essenziali del pensiero di Papert e della Scuola Costruzionista (Sez. 2) che fa da sfondo al contributo, descriviamo i tratti essenziali del progetto (Sez. 3), entrando nel merito degli aspetti che consideriamo più innovativi e quindi una sua valutazione e discussione critica, a partire dai risultati ottenuti (Sez. 4). Concludiamo il lavoro con una riflessione sull'idea di Scuola Creativa/4.0, che segna il passo delle nostre future iniziative (Sez. 5).

2 Background: Papert e la Scuola Costruzionista

L'approccio al pensiero computazionale è prezioso per ogni studente e non solo per gli appassionati e i programmatori di mestiere. I suoi principi si possono infatti applicare non solo alle discipline scolastiche, ma anche ai comportamenti quotidiani. Il maggior promotore di questa visione fu Seymour Papert, padre del *costruzionismo*, una teoria dell'apprendimento multidisciplinare che vede la scuola come un luogo di costruzione e non solo di trasmissione della conoscenza [2]. Scuola in cui bambini devono scoprire da soli le conoscenze di cui hanno bisogno. Punto cardine del costruzionismo è il ripensare il ruolo del computer nella scuola: 'il ruolo del computer è come quello della creta con cui costruire una scultura' [10]. Il computer non deve stare nel laboratorio informatico ma sul banco, diventando così uno strumento per manipolare, conoscere, scoprire e costruire, e non più soltanto di gestione delle informazioni; tramite il computer gli studenti diventano creatori del proprio processo di crescita e apprendimento. Con Papert, il computer diventa mezzo attivo con cui fare scuola spostando il controllo dell'apprendimento sullo studente, permettendogli di esplorare nuovi modelli e costruire le proprie conoscenze [2]. Papert

[5] `http://cogito.apice.unibo.it`

si riferisce a forme di apprendimento attivo: “dovrebbe essere il bambino a programmare il computer e non il computer a programmare il bambino” [2].

Il linguaggio LOGO, creato e ideato dallo stesso Papert, permette di applicare questi concetti costruendo, manipolando, sbagliando, ricostruendo e testando. In LOGO, Papert getta le basi per la comprensione del concetto di errore (bug e debug). Secondo Papert, la distinzione giusto/sbagliato è controproducente e può addirittura bloccare il processo dell’apprendimento nel bambino, il quale davanti a un errore tende a rimuoverlo e a dimenticarlo [12]. La correzione degli errori fa parte del processo di comprensione: il bambino che programma deve essere incoraggiato a studiare il bug e non a cancellarlo in fretta dalla sua memoria. La fase di debug guida la scoperta di ciò che non è andato come previsto, e, attraverso la comprensione, di come tale comportamento si possa correggere. Sbagliare significa esplorare, studiare il malfunzionamento di un programma e individuare soluzioni alternative al problema. L’obiettivo di LOGO non è quello di formare generazioni di programmatori di computer, ma di creare un ambiente per ”imparare a imparare”; la programmazione è l’espressione di se stessi attraverso artefatti cognitivi [2].

A partire dal lavoro di Papert, la Scuola Costruzionista si è sviluppata con numerosi contributi [6]. Fra gli altri, oggi un ruolo di riferimento è dato dal MIT Lifelong Kindergarten, guidato da Mitchel Resnick. E’ il gruppo ideatore della piattaforma/universo *Scratch*, ambiente basato sulla programmazione visuale a blocchi, utilizzato per introdurre i concetti di pensiero computazionale e coding da milioni di ragazzi in tutto il mondo e riferimento per le scuole di ogni ordine e grado. Il motto *learn to code, code to learn* [13] riassume efficacemente in estrema sintesi l’importanza della programmazione come strumento utile per mettere in atto strategie di apprendimento e problem solving, per progettare progetti e a comunicare idee, abilità utili non solo per i programmatori.

Nel Media Lab si sviluppano nuove tecnologie e strategie proprio per supportare il creative learning (apprendimento creativo). L’approccio si basa su quattro principi (le *4 P*) [14]: Project (Progetti), si lavora e si apprende meglio quando si lavora su progetti piuttosto che su esercizi singoli; Peers (Collaborare tra pari), Quando si collabora e si scambiano idee, si lavora meglio; Passion (Passione), Se si lavora su qualcosa che piace e che appassiona si lavora meglio, ci si concentra di più e si e determinati nel portare a termine il lavoro; Play (Giocare), L’apprendimento può essere giocoso: sperimentare nuove cose, testando i limiti provando e riprovando.

3 Il Progetto COGITO

A partire dal riferimento metodologico della scuola costruzionista, dal lavoro di Papert e MIT Lifelong, il progetto COGITO è nato con quattro obiettivi principali.

Il primo è costruire un percorso di apprendimento per rendere i bimbi fluenti in merito al coding, dove *esser fluenti* in questo caso significa la capacità di saperlo concretamente usare come strumento operativo per risolvere problemi, costruire artefatti, comunicare ed esprimersi. In ottica costruzionista, per questo *learn to code* è stato privilegiato l’apprendimento incrementale mediante la realizzazione di progetti, senza esporre i bimbi all’esposizione di principi e basi teoriche, formali. Allo scopo è stata scelta la

(a) Esempio Matematica (b) Esempio Italiano

Fig. 1: Esempi Micromondi Didattici

piattaforma e linguaggio visuale a blocchi Snap! [6]. Nata come estensione del più noto e diffuso Scratch, la piattaforma Snap! è stata scelta in origine per la flessibilità che ne permetteva l'utilizzo anche su Tablet come web-app, quindi fruibile mediante browser.

Il secondo obiettivo è sviluppare l'idea di pensiero computazionale e coding come meta-competenza, per cui i progetti di cui sono sopra – chiamati *micro-mondi*, riprendendo il termine introdotto in [11] – sono stati pensati e sviluppati in stretta collaborazione fra insegnanti e ricercatori contestualmente alle attività e ai contenuti che man mano gli alunni vedevano parallelamente nelle altre materie curricolari. Concettualmente, un micromondo rappresenta un ambiente computazionale progettato per favorire l'esplorazione e apprendimento (co-costruito) di concetti e competenze interdisciplinari, ove rappresentare e risolvere problemi, progettare, costruire, eseguire, provare, simulare, che può evolvere ad essere esteso man mano—un incubatore operativo di conoscenza. Tecnicamente, nel caso di COGITO, un micro-mondo è un programma in Snap!, spesso solo parzialmente implementato, con insieme di blocchi di base scelti in relazione alle competenze e contenuti da esplorare. Un esempio è dato in Fig.1a: un micromondo matematico introdotto durante il progetto, ideato come ambiente per rappresentare e implementare la soluzione di problemi di aritmetica. Un altro esempio è fornito in Fig. 1b: in questo caso il contesto è l'italiano, un micromondo che permette di esplorare, combinare elaborare lettere, parole, elementi grammaticali. Il *code to learn* è stato quindi messo in atto implementando ed esplorando micro-mondi ad opera dei ragazzi, lavorando co-operativamente in gruppi e con gli insegnanti, sfruttando il coding e l'esecuzione dei micromondi al computer come mezzo per ragionare, capire, esplorare problemi e contenuti curricolare, nonché immaginare e creare.

Il terzo obiettivo è sfruttare questi progetti e tecnologie – ed in particolare l'effetto che hanno sui ragazzi – per suscitare interesse circa le materie stesse e promuovere il lavoro di gruppo, la scambio di idee, il confronto, nonché creare ambienti di apprendimento e situazioni che favoriscono la possibilità per ragazzi con più difficoltà di potersi esprimere e 'liberare' la proprie energie creative.

Infine, il quarto obiettivo – correlato al secondo – è l'ideazione di progetti / micromondi che promuovere la realizzazione di progetti che permettano di mettere in relazione

[6] `http://snap.berkeley.edu`

Fig. 2: Aula TEAL 3.0 “Li2Lab”, presso la Scuola Primaria “Carducci” – Cesena

mondo virtuale e mondo fisico, promuovendo quindi una visione in cui pensiero computazionale, coding e ’computer’ non siano confinati solo alla creazione di ’mondi virtuali’ ma siano utili a supporto di creazioni e attività nel mondo fisico.

3.1 Organizzazione e svolgimento

Il progetto ha coinvolto nell’arco di 3 anni quattro classi - rispettivamente, due seconde e due terze al primo anno del progetto - complessivamente una ottantina di alunni. Teatro del Progetto è stata l’aula 3.0 denominata *Li2Lab*, un aula TEAL appositamente inaugurata per ospitare le attività del progetto (Fig. 2). Dotato di lavagna interattiva, tavoli modulabili organizzabili ad isole e connessione WiFi ad Internet sempre presente, il Li2Lab ha consentito la realizzazione di due macro categorie di attività: (1) attraverso l’ausilio di Tablet e del linguaggio/piattaforma Snap!, attività di coding per la costruzione di micromondi interattivi e dinamici ed orientati a tematiche didattiche d’interesse affiancate (2) ad attività unplugged per il consolidamento dei contenuti e la valorizzazione di ulteriori competenze quali, ad esempio, la capacità di cooperazione/interazione e valore dell’errore. In Li2Lab, dal primo anno, ciascuna classe ha sempre visto la duplice supervisione dei “code wizards”, gli esperti di coding, e delle rispettive insegnanti, la cui presenza e ruolo è risultato fondamentale. A partire dal secondo anno, le attività in Li2Lab sono state affiancate anche da attività direttamente in aula, durante (alcune) materie curricolari — matematica e italiano in particolare.

Nel prosieguo riportiamo gli aspetti caratterizzanti l’approccio usato. La descrizione dettagliata delle attività svolte e dei micromondi sviluppati nel corso dei tre anni è disponibile sul portale del progetto [7].

3.2 Aspetti Caratterizzanti

Ciclo Virtuoso *Learn to Code, Code To Learn* – Apporto metodologico L’implementazione del ciclo virtuoso learn-to-code – code-to-learn conduce applicare metodi e pratiche promosse da pensiero computazionale e approccio costruzionista trasversalmente a supporto e vantaggio dell’apprendimento delle materie curriculari stesse. In questo, il computer in quanto tecnologia di esecuzione svolge un ruolo strategico, fondamentale—come già evidenziato in [11].

[7] `http://cogito.apice.unibo.it`

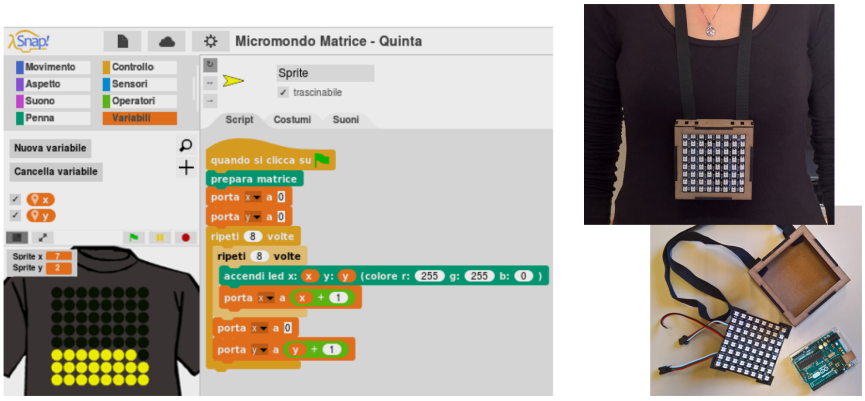
Da un lato, la progettazione, sviluppo e analisi di micromondi comporta l'applicazione di metodi quali decomposizione/astrazione/generalizzazione in una chiave che non rimane confinata al codice e ad elementi informatici, ma si proietta immediatamente sul dominio di cui è oggetto il micromondo, quindi il problema / competenza curricolare. Dall'altro, la costruzione di micromondi promuove un approccio *incrementale* e iterativo nella risoluzione dei problemi o nella creazione più in generale di artefatti, e quindi nell'affrontare i problemi e sfide oggetto dei micromondi stessi.

Questo diviene uno strumento concreto per *gestire l'indeterminatezza* [2] ed evitare lo spaesamento di fronte a problemi la cui complessità è tale non poter inquadrare immediatamente la risoluzione. La possibilità di provare, eseguire porzioni non complete, di studiare il comportamento ottenuto diventa uno strumento concreto a supporto sia del problem solving, sia di design. In questo processo, l'errore diventa un elemento fondamentale positivo, poiché permette agli alunni di riflettere sul motivo per cui il comportamento ottenuto è diverso da quello atteso. La possibilità quindi di ri-eseguire più volte un programma e di fare debugging diventano strumenti non solo lato coding, ma relativo al dominio oggetto del programma, alle relative conoscenze e competenze.

Chiavi Narrative e Principi del Pensiero Computazionale Le metafore e chiavi narrative sono fondamentali per permettere agli alunni a digiuno di qualsiasi concetto di programmazione di avere un quadro di riferimento generale per interpretare man mano i vari concetti che vengono introdotti e concretamente utilizzati nella costruzione dei micromondi. La metafora e chiave narrativa più generale utilizzata è stata la teatralizzazione, ovvero la visione di un programma (su Snap!) come spettacolo teatrale da mettere in scena. Analogamente ad uno spettacolo teatrale, un programma in Snap! (e in Scratch) è costruito da un insieme di Sprite (attori), di cui il programmatore (regista) deve stabilire e specificare il copione (Script), utilizzando un linguaggio per descrivere il copione comprensibile per l'attore stesso (l'insieme dei blocchi). La descrizione di un copione include sequenze di istruzioni, che tipicamente l'attore deve eseguire a fronte di determinati eventi che avvengono durante l'esecuzione dello spettacolo. Le istruzioni possono includere anche azioni con cui gli attori interagiscono con l'ambiente dello spettacoli, altri attori (mediante scambio di messaggi asincrono) e con il pubblico stesso (come forme di I/O). Oltre agli attori, uno spettacolo è caratterizzato da una scenografia (Stage) che fa da sfondo all'azione degli attori, di cui il regista può definire comportamento e immagine - modificabile durante lo spettacolo dagli attori stessi.

Questa metafora permette agli alunni di diventare immediatamente operativi – per analogia – nella formulazione di programmi di una certa complessità, che vedano più attori concorrenti che computano e comunicano, dal comportamento reattivo. In questo ritroviamo gli elementi principali di paradigmi di programmazione di alto livello, quale il paradigma ad oggetti, ad attori, e ad agenti - al di là quindi della programmazione procedurale.

Micromondi come Livello di Astrazione e Linguaggio Un aspetto importante dei micro-mondi è data dalla scelta dei blocchi iniziali, con i quali comporre gli script e definire nuovi blocchi. Questo - nelle mani di un insegnante - permette di definire il livello di astrazione con cui si vuole sviluppare l'attività di apprendimento, ovvero l'insieme di concetti di prima classe che devono essere utilizzati e composti nel ragionamento e nella progettazione e quindi, dualmente, gli aspetti da cui astrarre, poiché non significativi per



(a) Micromondo MatriceLed (b) BoxMatrice

Fig. 3: Matrice di Led – Smart T-Shirt

gli obiettivi dell'attività. Conseguentemente, definire un livello di astrazione e l'insieme dei blocchi significa definire anche linguaggio che gli alunni useranno per rappresentare il problema e ragionare sulla soluzione o sul progetto da sviluppare. Tale linguaggio - quindi l'insieme dei blocchi - possono essere quindi progettati flessibilmente in relazione all'età, alle competenze, e più in generale alle specificità degli alunni.

Snap! In questo caso fornisce una funzionalità essenziale - non fornita da Scratch: ovvero la possibilità di nascondere blocchi, oltre che - come Scratch - di crearne di nuovi.

Micromondi Ibridi - Il tool snap2ino L'idea di micromondo così caratterizzata permette di avere uno strumento flessibile per progettare esperienze che spazino da ambienti totalmente virtuali ad ambienti ibridi, che includano anche elementi e things del mondo fisico, sfruttando tecnologie embedded/ indossabili e Internet delle Cose. A supporto di questa visione è stato sviluppato un tool chiamato snap2ino[8], un transpiler che permette di tradurre un micromondo eseguibile su Snap! In un programma Wiring direttamente eseguibile su piattaforme embedded come Arduino o ESP8266. A differenza di strumenti già disponibili – es: Snap4Arduino – snap2ino permette di considerare micromondi aperti ed eventualmente complessi, a più sprite. snap2ino non si limita a introdurre blocchi specifici per il controllo di sensori/attuatori, ma permette di definire flessibilmente nuovi blocchi, eventualmente mappandoli – una volta tradotti – su procedure/oggetti in C/C++ per usare librerie presenti su Wiring. Un esempio di micromondo ibrido introdotto al terzo anno del Progetto COGITO è dato dal micromondo Smart T-Shirt (Fig. 3a). Il micromondo è costituito da una maglietta con una matrice di led RGB 8x8, virtuale. I blocchi caratterizzanti (oltre a quelli di base algoritmici classici) permettono di accendere/spegnere i led, ovvero - nella versione virtuale - di colorare i pallini del colore specificato. Mediante questo micromondo è possibile creare animazioni, implementando opportuni algoritmi (in figura quello per far pulsare un quadrato rosso). In Fig. 3b è mostrato invece la versione fisica, un kit realizzato in collaborazione con il FabLab – che supporta il progetto COGITO – costituito da un box di tagliato con laser cutter,

[8] Disponibile qui: `https://github.com/criadcoding/snap2ino`

contenente un Arduino e una matrice led WS2812B. Mediante snap2ino il micromondo che gira su Snap! Viene tradotto e mandato in esecuzione in modo standalone sul kit.

Questo kit ha permesso di realizzare attività in cui i ragazzi a gruppi di 2/3 hanno inventato, progettato e implementato uno spettacolo coinvolgendo più T-shirt, con animazioni prima progettate e sviluppate nel virtuale poi eseguite sui kit fisici.

4 Discussione

Il progetto COGITO ha fin dall'inizio intricato e affascinato gli insegnanti, nonostante non ci fosse la piena consapevolezza del percorso che si stava man mano *inventando*. Sicuramente ha richiesto al gruppo degli adulti professionisti coinvolti di trovare armonia, che voleva dire *accordarsi* nelle nostre diversità per far scaturire ogni individualità. Imprescindibile è stato il confronto costante tra gli adulti stessi, per garantire la ricerca di azioni efficaci e trasversali. Riuscire a far convivere e contaminare attività e percorsi fortemente innovativi che utilizzino strumentazioni tecnologiche con proposte di studio *tradizionali* è stata la scommessa per una scuola in cui la curiosità, la creatività, l'impegno e l'innovazione possano trovare equilibrio.

L'ideazione e la programmazione umana che si accordano a macchine digitali ha trasmesso agli insegnanti e ai bambini il suo fascino. Ha entusiasmato la possibilità di riconciliare le discipline per raccontarci e raccontare percorsi che non aprono porte mentre altre si chiudono, ma che tengono socchiuse le possibilità di varcare soglie, avanzare, ritornare, ripetere, in un gioco di apprendimento che dà il senso di esserci e di valere. COGITO è stato caratterizzato anche dalla serenità di affrontare ogni percorso, talvolta faticoso, ma stimolante e in cui ci si ritrova ogni volta. Il progetto ha evidenziato l'integrazione a vari livelli del mondo digitale/computazionale con attività pratiche e materiali del mondo fisico, in modo da promuovere un quadro dove si persegua un bilanciamento virtuoso e armonioso fra questi due mondi. Questo mediante l'ideazione di attività e progetti che permettano di usare pensiero computazionale e coding come strumenti concettuali e operativi per realizzare oggetti fisici, artigianali con un'*anima digitale*, ovvero elementi digitali che ne arricchiscono in modo creativo le funzionalità o l'estetica (esempio: magliette con matrici di led programmabili) Armonia quindi ed equilibrio di tempi e di spazi: dove, che sia bambino o adulto, docente o discente, non mi sento stretto e dove non mi annoio, dove posso pensare, domandare, provare. Dove anche l'errore trova il suo equilibrio perché non è giudizio ma consapevolezza e possibilità di *risistemare*; equilibrio non come assenza di forze e contrasti ma come accordo tra esse.

L'attività condotta in classe, si è rivelata certamente utile per osservare gli allievi mentre si cimentavano a sperimentare e conoscere nuovi ambienti di sviluppo del pensiero. Gli insegnanti hanno potuto così confermare o modificare la conoscenza che avevamo di ognuno di loro e delle loro capacità. L'osservazione ha fornito loro nuovi importanti elementi di riflessione, alcuni allievi in forte difficoltà rispetto alle normali prestazioni scolastiche, posti in contesti differenti, hanno potuto far emergere capacità e abilità diverse riuscendo a realizzare l'obiettivo richiesto con una notevole ricaduta positiva sull'autostima. La responsabilità e l'orgoglio di essere dentro una *novità* scolastica li ha resi realmente protagonisti e co-costruttori di uno spazio di pensiero altro ,in cui riconoscersi e mettersi in gioco. Le competenze in COGITO vengono costruite

mediante attività basate su e a partire da problemi, contesti e progetti la cui valenza sia chiaramente riconoscibile dagli insegnanti e dagli alunni nelle altre discipline e attività curricolari. La nozione di micromondo viene in questo caso utilizzata per definire programmi (e relativi linguaggi, ambienti) costruiti incrementalmente dagli alunni, in cooperazione a vari livelli con insegnanti e esperti, per studiare, esplorare, esercitare competenze e obiettivi nelle varie discipline curricolari (sia scientifiche, sia umanistiche). I micromondi sono stati anche oggetto di progettazione (analisi/valutazione) per gli insegnanti.Infatti l'insegnante può costruire dei micromondi come ambienti utili per l'esplorazione e apprendimento costruttivo/computazionale di competenze e contenuti relativi alle materie. L'insieme dei micromondi sviluppati nel corso delle attività viene a far parte della "micromondoteca", come luogo in rete dove gli alunni possono tener traccia e mostrare quanto hanno fatto, condividere, copiare, riusare.

Proporre percorsi di apprendimento in cui gli allievi siano messi nelle condizioni di creare, mobilitando le competenze acquisite e superando le eventuali difficoltà, attiva un circolo virtuoso: sentirsi consapevolmente competenti genera una forte motivazione e sostiene il pensiero creativo e divergente, che è alla base del pensiero computazionale al quale il coding naturalmente tende. Anche l'errore diventa una potente occasione di crescita, fornendo nuove possibilità di analisi e conseguente revisione della strategia utilizzata, al pari degli informatici impegnati nel debugging: tollerare la frustrazione e trasformarla in autocontrollo e riflessione. Inoltre, la collaborazione tra pari finalizzata alla realizzazione di un prodotto comune contribuisce naturalmente a rinnovare lo spirito di appartenenza e di coesione. Nelle attività proposte c'è sempre stata l'attenzione al lavoro in gruppo e piccolo gruppo e la parola, per noi preziosa, ha trovato il giusto spazio per tradurre le idee, condividere le ipotesi, le riformulazioni, prendere le decisioni.

Rispetto agli apprendimenti relativi alle discipline curricolari, gli insegnanti hanno potuto osservare che l'attività trasversale attuata tramite il coding, potenzia e amplia abilità che rendono possibile un'acquisizione più consapevole dei contenuti disciplinari specifici: leggere, comprendere, scrivere, misurare, calcolare. Inoltre durante le sessioni di lavoro la mobilitazione delle capacità logiche è sempre stata condizione imprescindibile: nel formulare e applicare costrutti logici, nel consolidare la progettualità e il pensiero procedurale, nello sviluppare soluzioni ai percorsi. Abbiamo notato che le procedure sperimentate in COGITO, hanno *contaminato* positivamente gli stili di apprendimento dei bambini, le strategie messe in atto nell'organizzazione e nello studio individuale e anche in ambiti informali come il gioco libero.

A fronte degli aspetti positivi, è importante sottolineare come il progetto non possa essere considerato esaustivo nella sua sperimentazione didattica. Infatti non ha neppure i requisiti per una verifica scientifica con i criteri che le sono propri. Però è stata validata l'efficienza degli artefatti nella loro fruibilità e correttezza funzionale. Sono state verificate le capacità di programmazione divenute competenze replicabili e la giusta ricostruzione/riordino dei processi, degli algoritmi. COGITO non si è avvalso di un gruppo di controllo per cui talvolta gli insegnanti si sono interrogati in merito ad un problema importante, ovvero se le competenze raggiunte sarebbero comunque diventate patrimonio individuale dei bambini. Sicuramente il percorso può essere 'narrato', modalità a noi più consona e la trama del racconto è intessuta di parole chiave come interesse, passione, protagonista, collaborazione, innovazione.

5 Conclusioni – verso una Scuola 4.0, una Scuola Creativa

Il progetto COGITO ha permesso di maturare la chiara convinzione sull'importanza in campo informatico/tecnologico, per gli allievi, di acquisire competenze che vadano oltre la semplice gestione del mezzo tecnico. In futuro dovranno utilizzare massivamente, sia a livello lavorativo che personale, le tecnologie, risulta dunque fondamentale che la Scuola fornisca loro le competenze per governarle e controllarle in modo critico. I dispositivi tecnologici con cui si sono approcciati restano comunque degli strumenti ma sono i percorsi con essi attuati che hanno permesso loro di *costruire conoscenza*.

L'esperienza è risultata valore aggiunto nel percorso scolastico; piacevole, stimolante, coinvolgente e da ultimo, ma assai importante in una scuola dove il benessere è una priorità, divertente per tutti gli attori coinvolti, adulti e bambini. E' stata una grande opportunità educativa-didattica per gli adulti e per quel gruppo di bambini che ha messo in gioco, con tante variabili, in un intreccio intricato e intrigante in grado di ampliare capacità di apprendimento, creatività e costruzione creativa, garantendo il raggiungimento di competenze e conoscenze specifiche di altre discipline incluse nel percorso didattico/curricolare.

In futuro, tre sono le linee principali che caratterizzano il prosieguo del progetto. La prima concerne la sperimentazione del progetto anche con una Scuola Secondaria di Primo grado, cercando di costruire e definire un ponte in merito a queste attività relativamente a scuole di grado diverso. In secondo luogo, mettere in piedi delle attività di formazione per gli insegnanti, al fine di condividere, perfezionare quanto svolto nel progetto, nonché avere un punto di partenza perché possa essere riproposto – in forme diverse – in modo autonomo dagli insegnanti stessi. Terzo punto, studiare e sperimentare forme di valutazione in armonia con il tipo di attività svolte, basate – ad esempio – sulla costruzione incrementale di un portfolio di progetti/micromondi, che diventano bagaglio che accompagni gli alunni di anno in anno, di grado in grado.

Per concludere, crediamo che in questo momento storico la scuola debba interrogarsi e interrogare la società. Una scuola realmente agenzia educativa non può restare ferma e immobile solo su parametri tradizionali. Aristotele diceva che la vita è nel movimento e certamente questo progetto ci ha fatto avanzare, nella ricerca e nell'azione. La visione di scuola sui cui è stato è stato sviluppato il progetto COGITO è una Scuola Creativa come evocata e definita da più parti [15], una Scuola 4.0 che coniuga innovazione con inclusione. La Scuola Creativa ha nel nostro paese una lunga tradizione pedagogica e matetica a più voci, che va dalla Montessori ai Cento Linguaggi di Malaguzzi e Reggio Children [5], alla Pedagogia della Lumaca di Zavalloni [18]. È una Scuola che mette al centro i "diritti naturali dei bambini" [15,18,5] e costruisce un percorso in cui innovazione e inclusione vanno a pari-passo, ben al di là di un puro discorso di formazione orientato al lavoro. Una tradizione che è stata ed è tuttora di fatto il riferimento stesso sia per il Pensiero di Papert e per le 4P di Resnick [14].

Oggi viviamo un momento straordinario, in cui c'è l'opportunità di fatto di sperimentare progetti – come COGITO – in cui unire pensiero computazionale e gli attori/idee/tecnologie più innovative a riguardo, con un pensiero psico-pedagogico e *matetico* che ha una tradizione importante, e questo costruendolo sul campo, mediante la collaborazione fra persone di competenze eterogenee — insegnanti, informatici, psico-pedagogisti, ingegneri, ricercatori.

Riferimenti

1. Armoni, M.: Computing in schools computer science, computational thinking, programming, coding: the anomalies of transitivity in k-12 computer science education. ACM Inroads 7(4), 24–27 (2016)
2. Capponi, M.: Un giocattolo per la mente. L'informazione cognitiva di Seymour Papert. Morlacchi Editore (2009)
3. Denning, P.J.: Computational thinking in science. American Scientist 105(1), 13–17 (2017)
4. Denning, P.J.: Remaining trouble spots with computational thinking. Communications of the ACM 60(6), 33–39 (2017)
5. Edwards, C., Gandini, L., Forman, G.: I Cento Linguaggi dei Bambini: l'approccio di Reggio Emilia all'educazione dell'infanzia. Junior (1995)
6. Kafai, Y.B., Resnick, M.: Constructionism in practice: Designing, thinking, and learning in a digital world. Routledge (1996)
7. Kay, A.C.: A dynamic medium for creative thought. Xerox Palo Alto Research Center (1970)
8. Lodi, M., Martini, S., Nardelli, E.: Abbiamo davvero bisogno del pensiero computazionale? Mondo Digitale (72), 1–15 (2017)
9. Nardelli, E.: Informatica nella scuola: disciplina fondamentale e trasversale, ovvero "di cosa parliamo quando parliamo di pensiero computazionale". Scienze e Ricerche (Aprile 2017)
10. Papert, S.: Bambini e adulti a scuola con il computer (1997)
11. Papert, S.: Mindstorms: Children, computers, and powerful ideas. Basic Books, Inc. (1980)
12. Papert, S.: The Children's Machine. Rethinking School in the Age of the Computer. Basic Books, Inc. (1993)
13. Resnick, M.: Learn to code, code to learn. EdSurge, May (2013)
14. Resnick, M.: Lifelong Kindergarten. Cultivating Creativity through Projects, Passion, Peers, and Play. The MIT Press (2017)
15. Robinson, K., Aronica, L.: Creative Schools: The grassroots revolution that's transforming education. Penguin Books (2016)
16. Solomon, C.: Computer environments for children: A reflection on theories of learning and education. MIT press (1988)
17. Wing, J.M.: Computational thinking. Communications of the ACM 49(3), 33–35
18. Zavalloni, G., Farinelli, F.: La pedagogia della lumaca: per una scuola lenta e nonviolenta. Emi (2012)