Pedagogica*mente* e didattica*mente*

25

Pedagogica*mente* e didattica*mente*

collana diretta da
Raffaella Biagioli e Marinella Muscarà

Oltre il coding

Pensiero computazionale, robotica educativa,
superamento dei divari di genere,
per una prospettiva educativa integrata e consapevole

a cura di
Beatrice Miotti, Giovanni Nulli

Edizioni ETS

www.edizioniets.com

PNR 2021-2027 Programma nazionale per la ricerca

Ministero dell'Università e della Ricerca

INDIRE ISTITUTO NAZIONALE DOCUMENTAZIONE INNOVAZIONE RICERCA EDUCATIVA

PNR 2021 II ANNUALITA' - CUP B53C22009650001

Edizioni ETS
Palazzo Roncioni - Lungarno Mediceo, 16, I-56127 Pisa
info@edizioniets.com - www.edizioniets.com

Distribuzione: Messaggerie Libri SPA - Sede legale: via G. Verdi 8 - 20090 Assago (MI)

Promozione: PDE PROMOZIONE SRL - via Zago 2/2 - 40128 Bologna

ISBN 978-884677633-4

INTRODUZIONE

Daniela Bagattini

Negli ultimi due decenni il rapporto tra scuola, tecnologie digitali e processi di apprendimento è diventato uno dei principali terreni di confronto nel dibattito educativo internazionale. La progressiva diffusione degli strumenti digitali nella vita quotidiana, nei contesti professionali e nelle dinamiche sociali ha infatti richiesto ai sistemi scolastici una riflessione sempre più urgente sulle competenze necessarie per vivere e partecipare consapevolmente alla società contemporanea. In questo scenario si collocano temi come il pensiero computazionale, il coding, la robotica educativa, la competenza digitale e, più recentemente, l'intelligenza artificiale generativa, che sono entrati stabilmente nel lessico delle politiche educative, della formazione degli insegnanti e della ricerca pedagogica (Wing, 2006; Bocconi et al., 2016; Redecker, 2017).

Tuttavia, la crescente centralità di questi concetti non ha sempre coinciso con una reale chiarezza teorica e metodologica. Al contrario, il rapido successo di termini come "pensiero computazionale" o "didattica digitale" ha spesso prodotto interpretazioni differenti, talvolta contraddittorie, che hanno contribuito a generare pratiche scolastiche molto eterogenee. In numerosi casi, coding e robotica educativa sono stati interpretati principalmente come attività tecniche o come strumenti motivazionali, concentrando l'attenzione sull'utilizzo delle tecnologie piuttosto che sui processi cognitivi, pedagogici e disciplinari che tali tecnologie possono sostenere (Lodi & Martini, 2021).

Questo volume nasce dall'esigenza di affrontare criticamente tali questioni, proponendo un percorso di approfondimento teorico e metodologico che colleghi il dibattito scientifico internazionale con le pratiche didattiche quotidiane.

Attraverso la presentazione dei moduli di una formazione online dedicata al pensiero computazionale, alla competenza digitale, al coding, alla robotica educativa e all'intelligenza artificiale saranno messi in evidenza i presupposti teorici, le implicazioni pedagogiche e le possibili applicazioni curricolari di queste metodologie innovative.

La formazione si inserisce nel quadro delle misure previste dal Fondo per la promozione e lo sviluppo delle politiche del Programma nazionale per la ricerca (PNR), lo strumento finanziario volto a sostenere la ricerca in Italia, finanziando iniziative strategiche come contratti per giovani ricercatori, progetti collaborativi tra università ed enti, ricerca interdisciplinare e propedeutica a programmi europei con l'obiettivo di rafforzare la competitività e attrarre talenti, in linea con le priorità del PNR 2021-2027 e gli obiettivi europei. In particolare, il progetto fa parte delle Azioni di Ricerca per l'Istruzione e la Formazione, che hanno lo scopo di sostenere la ricerca educativa, migliorare l'efficacia dell'insegnamento ed innovare le didattiche.

Il volume si propone quindi non soltanto di descrivere contenuti e attività formative, ma di costruire una cornice interpretativa capace di aiutare le e i docenti a orientarsi in un panorama spesso complesso e frammentato. Per questo motivo vengono analizzati i principali riferimenti teorici che hanno contribuito allo sviluppo del dibattito contemporaneo, dalle riflessioni costruzioniste di Seymour Papert fino alle definizioni più recenti di pensiero computazionale elaborate in ambito europeo e internazionale (Papert, 1980; Wing, 2008; Bocconi et al., 2022). Parallelamente, vengono approfonditi i framework relativi alla competenza digitale, con particolare attenzione al DigComp e alla sua evoluzione verso la versione 3.0, che rappresenta oggi uno dei più importanti riferimenti europei per la definizione delle competenze digitali del cittadino (Vuorikari, Kluzer & Punie, 2022).

L'assunto che attraversa il testo è che coding, robotica educativa e intelligenza artificiale non possano essere compresi esclusivamente come strumenti tecnologici, ma debbano essere letti all'interno di una riflessione più ampia sulla didattica, sulla costruzione della conoscenza e sul ruolo della scuola nella società contemporanea. In questa prospettiva, il volume assume una posizione chiara: il valore educativo delle tecnologie non risiede negli strumenti in sé, ma nelle attività, nei processi cognitivi e nelle dinamiche relazionali che essi rendono possibili.

La semplice introduzione di strumenti digitali nella scuola, infatti, non garantisce automaticamente innovazione didattica, sviluppo del problem solving o miglioramento degli apprendimenti. La ricerca internazionale ha mostrato come il trasferimento automatico di competenze da un contesto all'altro — ad esempio dall'attività di programmazione al problem solving generale — sia molto più limitato di quanto spesso sostenuto nella retorica educativa (Salomon, 1984; Scherer, Siddiq & Sánchez Viveros,

2019). Per questo motivo il volume insiste sulla necessità di contestualizzare coding e robotica educativa all'interno di metodologie didattiche attive, laboratoriali e riflessive, nelle quali il docente assuma un ruolo di progettista di ambienti di apprendimento significativi.

In tale prospettiva assumono particolare rilevanza approcci come il *problem based learning*, il *project based learning* e la didattica laboratoriale, che permettono agli studenti di costruire conoscenze attraverso attività concrete, collaborative e orientate alla soluzione di problemi autentici (Hmelo-Silver, 2004; Thomas, 2000). Coding e robotica educativa vengono quindi interpretati non come discipline isolate o semplici esercizi tecnici, ma come strumenti capaci di sostenere percorsi interdisciplinari, processi di costruzione del sapere e sviluppo dell'autoefficacia (Nulli, Miotti & Di Stasio, 2022)

Uno degli obiettivi centrali del volume è proprio quello di mettere in relazione la dimensione tecnica con quella pedagogica. Negli ultimi anni, infatti, la formazione sulle tecnologie educative si è spesso concentrata sull'apprendimento degli strumenti, trascurando il fatto che la reale efficacia didattica dipende dalla capacità di integrare tali strumenti in percorsi coerenti dal punto di vista metodologico e curricolare. Comprendere il funzionamento di un robot educativo, di un ambiente di programmazione o di un sistema di intelligenza artificiale è certamente importante, ma lo è altrettanto comprendere quali processi cognitivi e relazionali vengano attivati durante il loro utilizzo e quali obiettivi formativi si intendano perseguire.

Il volume intende inoltre sottolineare come il tema della competenza digitale non possa più essere ridotto a una semplice alfabetizzazione tecnica. Le trasformazioni sociali e culturali degli ultimi anni hanno reso evidente la necessità di formare cittadini capaci di utilizzare criticamente le tecnologie, comprendere il funzionamento degli algoritmi, valutare l'affidabilità delle informazioni, proteggere la propria identità digitale e partecipare consapevolmente agli ambienti online (European Commission, 2023). In questo senso, il framework DigComp rappresenta uno strumento fondamentale per collocare coding, pensiero computazionale e intelligenza artificiale all'interno di una più ampia prospettiva di cittadinanza digitale.

Particolare attenzione viene dedicata anche al rapporto tra tecnologie e inclusione. La diffusione del coding e della robotica educativa ha infatti aperto nuove possibilità per la partecipazione attiva degli studenti, ma ha anche evidenziato il permanere di stereotipi e disuguaglianze, in partico-

lare rispetto ai divari di genere nelle discipline STEM (UNESCO, 2017). Per questo motivo il volume affronta esplicitamente il tema del rapporto tra ragazze e tecnologia, riflettendo sul ruolo della scuola nella costruzione di ambienti inclusivi e nella promozione di opportunità formative realmente accessibili a tutti (Bagattini, Miotti, 2022).

Il testo si inserisce nel quadro delle Azioni di Ricerca per l'Istruzione e la Formazione sostenute dal Programma Nazionale per la Ricerca (PNR), finalizzate a promuovere percorsi di innovazione educativa basati su evidenze scientifiche, sperimentazione e ricerca partecipata. La formazione descritta nel volume nasce infatti dall'idea che l'innovazione scolastica non possa essere affidata esclusivamente a interventi occasionali o all'introduzione di nuovi strumenti tecnologici, ma richieda la costruzione di una cultura professionale condivisa, fondata sulla riflessione pedagogica, sulla progettazione didattica e sulla collaborazione tra docenti.

In questa prospettiva, il volume propone una visione della formazione come processo lungo e complesso, che richiede studio, sperimentazione, revisione critica delle pratiche e ristrutturazione dei propri saperi professionali. La figura del docente viene così interpretata non come semplice utilizzatore di tecnologie, ma come professionista riflessivo capace di progettare ambienti di apprendimento, interpretare criticamente le trasformazioni digitali e adattare metodologie e strumenti ai bisogni educativi concreti degli studenti (Schön, 1983).

I capitoli che seguono accompagnano il lettore in un percorso progressivo: dall'analisi critica del concetto di pensiero computazionale alla riflessione sulla competenza digitale, dall'approfondimento delle metodologie didattiche fino alla progettazione curricolare, passando attraverso il ruolo della robotica educativa e dell'intelligenza artificiale. L'obiettivo complessivo è offrire strumenti teorici, metodologici e operativi per comprendere come le tecnologie possano essere integrate nella scuola in modo pedagogicamente fondato, criticamente consapevole e orientato alla formazione di cittadini autonomi, creativi e responsabili.

In definitiva, questo volume non intende proporre ricette semplici o soluzioni definitive, ma contribuire alla costruzione di uno spazio di riflessione condivisa sul rapporto tra educazione e trasformazione digitale. Un rapporto che oggi, più che mai, richiede alla scuola non soltanto di insegnare a usare le tecnologie, ma di aiutare gli studenti a comprenderle, interpretarle e governarle criticamente all'interno della complessità del mondo contemporaneo.

Riferimenti bibliografici

Bocconi, S. et al. (2016). *Developing Computational Thinking in Compulsory Education*. European Commission.

Bocconi, S. et al. (2022). *Reviewing Computational Thinking in Compulsory Education*. European Commission.

European Commission (2023). *European Declaration on Digital Rights and Principles*.

Hmelo-Silver, C. E. (2004). Problem-Based Learning: What and How Do Students Learn? *Educational Psychology Review*, 16(3).

Lodi, M., & Martini, S. (2021). Computational Thinking, between Papert and Wing. *Science & Education*, 30(4).

Papert, S. (1980). *Mindstorms: Children, Computers, and Powerful Ideas*.

Redecker, C. (2017). *European Framework for the Digital Competence of Educators: DigCompEdu*.

Salomon, G. (1984). On the development of abilities and transfer. *New Ideas in Psychology*, 2(2).

Scherer, R., Siddiq, F., & Sánchez Viveros, B. (2019). The cognitive benefits of learning computer programming: A meta-analysis. *Journal of Educational Psychology*, 111(5).

Schön, D. (1983). *The Reflective Practitioner*.

Thomas, J. W. (2000). *A Review of Research on Project-Based Learning*.

UNESCO (2017). *Cracking the Code: Girls' and Women's Education in STEM*.

Vuorikari, R., Kluzer, S., & Punie, Y. (2022). *DigComp 2.2: The Digital Competence Framework for Citizens*. European Commission.

Wing, J. (2006). Computational Thinking. *Communications of the ACM*, 49(3).

Wing, J. (2008). Computational Thinking and Thinking about Computing. *Philosophical Transactions of the Royal Society A*.

1.
IL PENSIERO COMPUTAZIONALE: UN'ANALISI CRITICA PER LA DIDATTICA

Giovanni Nulli

Introduzione

Per il docente contemporaneo, specialmente nell'ambito della scuola del primo ciclo, il termine "pensiero computazionale" è diventato una presenza familiare, se non inevitabile. Esso ricorre in corsi di formazione, documenti ministeriali, proposte di attività di coding e robotica. Tuttavia, questa familiarità spesso non coincide con una chiarezza concettuale. Le definizioni proposte sono molteplici, a volte perfino incongruenti tra loro. Comprendere l'origine, l'evoluzione e il dibattito attorno a questo concetto non è quindi un esercizio accademico, ma un passaggio operativo essenziale. Serve al docente per orientarsi criticamente nell'offerta formativa, per interpretare le indicazioni normative e, soprattutto, per compiere scelte didattiche consapevoli su come e perché introdurre le tecnologie digitali in classe. Questo capitolo si propone di fare chiarezza, ricostruendo la storia del termine, mappando i diversi ambiti scientifici e istituzionali che lo hanno plasmato, analizzandone i limiti teorici – in particolare il problematico concetto di transfer – e arrivando a una delimitazione utile e pragmatica per l'agire didattico quotidiano.

1.1. La genesi di un dibattito globale: dalle scienze informatiche alle politiche educative

La questione del pensiero computazionale assume una rilevanza centrale nel dibattito scientifico e pedagogico a partire dal 2006, quando Jeannette Wing, allora professoressa associata di informatica alla Carnegie Mellon University, propose in modo quasi provocatorio di considerare il saper programmare una competenza di base, al pari del leggere, dello scrivere e del far di conto (Wing, 2006). Questa posizione, che Wing avrebbe precisato e difeso in articoli successivi (2008, 2010), nasceva da una

constatazione evidente: gli aspetti computazionali stavano diventando fondamentali in quasi tutti i campi della ricerca scientifica. La mappatura del genoma umano, avvenuta pochi anni prima, ne era un esempio emblematico; essa era stata resa possibile solo dall'uso massiccio del calcolo automatico. Il biologo, l'ingegnere, e sempre più figure professionali, devono possedere, almeno in ambito disciplinare, delle competenze di programmazione. Per Wing, era dunque cruciale introdurre il pensiero computazionale in maniera precoce nei sistemi di apprendimento.

Questa visione, spesso sintetizzata nello slogan "pensare come un informatico", mirava a estrarre dall'informatica un atteggiamento mentale e una cassetta degli attrezzi logici – come la scomposizione di problemi complessi in sotto-problemi gestibili, il riconoscimento di pattern, la creazione di modelli astratti – che potessero essere applicati in qualsiasi campo. Non si trattava quindi di insegnare a tutti a diventare sviluppatori software, ma di fornire una "grammatica" universale per affrontare la complessità del mondo contemporaneo, sempre più modellato da algoritmi e dati. Wing immaginava un cittadino in grado non solo di usare la tecnologia, ma di comprenderne i principi fondanti, di decostruirne il funzionamento e, quindi, di esercitare un giudizio critico sulle sue implicazioni sociali ed etiche. Questa ambizione "alfabetizzante" fu il motore che trasformò una proposta settoriale in un movimento globale.

La proposta di Wing ha innescato un dibattito serrato, coinvolgendo non solo gli informatici ma anche i pedagogisti e i decisori politici. Già nel 2012, un rapporto della Royal Society britannica dava struttura a questa discussione (The Royal Society, 2012), che di lì a poco si sarebbe tradotta nell'introduzione della programmazione in modo strutturato nel curricolo della scuola primaria del Regno Unito. Un dibattito analogo, seppur con modalità differenti spesso partendo da casi di studio nelle scuole, ha interessato intorno al 2018 i paesi scandinavi (Bocconi, Chioccariello & Earp, 2018). A livello europeo, a partire dal 2015, il Joint Research Centre (JRC) della Commissione Europea ha avviato una sistematica mappatura di come i diversi Stati membri affrontassero il concetto (Bocconi et al., 2016). In Italia, il Piano Nazionale Scuola Digitale (PNSD) del 2015 ha istituzionalizzato il termine nell'Azione 17 (MIUR, 2015), un riferimento poi ripreso nel documento ministeriale "Indicazioni nazionali e nuovi scenari" del 2017 (MIUR, 2018). Parallelamente al percorso istituzionale, sono fiorite iniziative dal basso, spesso di matrice no-profit, come la piattaforma Code.org (nata negli USA nel 2013) e l'Europe Code Week, promossa dalla Commissione Europea sempre nel 2013. In pochi anni, quindi, la spinta

proveniente dalla ricerca scientifica, dalle istituzioni e dalla società civile ha portato la programmazione e il pensiero computazionale al centro dell'agenda educativa.

Il dibattito non verteva solo sul "se" introdurlo, ma soprattutto sul "come" e sul "perché". Da un lato, una visione più "strumentale" e orientata al mercato del lavoro, che vedeva nel coding una competenza tecnica fondamentale per l'economia digitale, un modo per colmare il previsto skill gap e alimentare l'innovazione industriale. Dall'altro, una visione più "culturale" e pedagogica, che lo interpretava come un nuovo alfabetismo, una forma di espressione e di pensiero critico, uno strumento per l'empowerment personale e la partecipazione democratica. Questa tensione tra finalità economiche (preparare una forza lavoro) e finalità educative (formare persone e cittadini) non è mai stata risolta e rimane un tratto caratteristico e spesso conflittuale del panorama attuale, influenzando la scelta degli obiettivi, dei contenuti e delle metodologie. Come vedremo, questa dicotomia si riflette anche nella ricerca di una definizione operativa.

1.2. Un "termine cappello": la ricerca (vana?) di una definizione univoca

Ma cosa significa esattamente "pensiero computazionale"? La prima rilevazione sistematica a livello europeo, condotta nel 2015 da European Schoolnet, un consorzio di ministeri dell'istruzione, mostrò immediatamente una significativa divergenza nelle definizioni adottate dai diversi paesi (Balanskat & Engelhardt, 2015). Lo spettro andava dal potenziamento generico del problem solving all'introduzione precoce della computer science. Nel 2016, il primo studio del JRC tentò una sintesi, identificando sei abilità costitutive: astrazione, pensiero algoritmico, automazione, scomposizione, debugging e generalizzazione (Bocconi et al., 2016). Alcune di queste (come il debugging) sono strettamente legate all'informatica, altre (come astrazione e generalizzazione) appartengono a un concetto più ampio di metodo scientifico. Il documento italiano del 2017 introduceva ulteriori elementi, come la logica e il pensiero analitico (MIUR, 2018). Il pensiero computazionale appariva così come un "termine cappello", un contenitore elastico e ambiguo sotto il quale potevano convivere processi eterogenei.

La Commissione Europea, con le sue finalità ampie di innovazione dei sistemi formativi e del mercato del lavoro, ha spesso insistito sulla necessità di formare nuovi "programmatori". Se questa esigenza è indub-

bia, il termine "programmatore" rimane a sua volta un concetto-cappello, suscettibile di molte interpretazioni: dallo sviluppatore di software specializzato all'hacker creativo. La stessa definizione iniziale di Wing – "pensare come un informatico" – si adatta bene a questa "indecidibilità". Ciò che può essere utile sul piano politico-comunicativo (un termine unico per mobilitare risorse e attenzione), però, diventa problematico in ambito esecutivo e didattico: come può un docente tradurre in scelte progettuali concrete un concetto così sfuggente?

Uno studio successivo del JRC (2022) ha segnato una svolta realistica, abbandonando l'idea di una definizione univoca e statica. Il rapporto *Reviewing Computational Thinking in Compulsory Education* (Bocconi et al., 2022) propone invece di vedere il PCT non come un monolite, ma come un costrutto composito e dinamico, interpretato in tre aree distinte ma spesso sovrapposte nei documenti politici e nelle pratiche:

1. Un insieme di concetti della programmazione applicabili anche ad altri ambiti (es. usare la metafora del ciclo per spiegare un processo ripetitivo in storia o in scienze).
2. Un insieme di concetti informatici stretti (dove astrazione, algoritmi, dati sono studiati nella loro specificità disciplinare).
3. Un riferimento generale al problem solving (un approccio metodologico sistematico a problemi aperti).

L'analisi ha rivelato che i diversi Stati europei – e spesso al loro interno, diversi attori – tendono a vedere e a promuovere il pensiero computazionale come una combinazione variabile e spesso inconsapevole di queste tre aree. La confusione nasce proprio quando, parlando di PCT, non si specifica a quale di queste tre "anime" ci si riferisce. Ad esempio, un'attività in cui si chiede di creare una storia animata con Scratch mobilita soprattutto la prima area (applicazione creativa di concetti di programmazione), sfiora la seconda (si impara a usare blocchi di comando) e può toccare la terza (si risolve il problema di come far muovere un personaggio). Un corso di informatica curricolare, invece, si focalizza sulla seconda area. Di fronte a questa "nuvola di significati", è imperativo per la scuola e per il singolo docente operare una scelta chiara e consapevole, dichiarando esplicitamente quali obiettivi si intendono perseguire: avvicinare alla logica della programmazione? Insegnare concetti informatici? Sviluppare un metodo di problem solving? Questa chiarezza è il primo, indispensabile passo per una progettazione didattica efficace e valutabile.

BOX DI APPROFONDIMENTO A: Papert vs. Wing – Due visioni fondative a confronto.

Aspetto	Visione di Seymour Papert (Costruzionista, anni '70-'80)	Visione di Jeannette Wing (Cognitiva/Istituzionale, 2006)
Origine	Pedagogia, costruttivismo, matematica.	Informatica, scienze cognitive, ingegneria del software.
Focus	L'apprendimento significativo e la costruzione di conoscenza. Il computer come partner cognitivo per esplorare idee potenti (Papert, 1984).	Il problem solving efficiente e il pensiero strategico. Il computer come modello astratto di esecutore di procedure (Wing, 2006).
Mezzo privilegiato	La programmazione come linguaggio espressivo e dialogico (Logo). Scrivere codice è un modo di "pensare ad alta voce", di dare forma concreta e sperimentabile al proprio ragionamento.	L'algoritmo come concetto astratto e trasferibile. La programmazione è un'applicazione possibile, ma il PCT è prima di tutto un modo di pensare indipendente dalla macchina.
Obiettivo primario	Empowerment dell'apprendente. Sviluppare una relazione fluida, creativa e personale con la conoscenza (soprattutto matematica), superando ansie e stereotipi (Papert, 1996).	*Literacy* (alfabetizzazione) di base per tutti. Fornire a ogni cittadino gli strumenti mentali per comprendere e agire in un mondo governato da algoritmi e dati.
Natura del PCT	Un processo di pensiero naturale che fiorisce in ambienti ricchi di oggetti "parlanti" (tartaruga Logo). L'informatica lo ha reso evidente.	Un prodotto dell'informatica da disseminare in tutte le discipline e nella società.
Eredità nel dibattito	È la radice della didattica laboratoriale, del making e dell'attenzione alla motivazione intrinseca (Capitoli 3 e 6).	È il fondamento delle definizioni istituzionali, del legame con le competenze-chiave e il framework DigComp (Capitolo 2).

Perché questo confronto è utile al docente? Perché mostra che il dibattito non è tecnico, ma filosofico-pedagogico. Nella pratica, l'insegnante efficace spesso ibrida queste due visioni: usa il coding in modo creativo e giocoso (alla Papert) per sviluppare competenze di problem solving sistematico (alla Wing). Riconoscere queste radici aiuta a evitare derive: un approccio puramente tecnicista (solo Wing) rischia di essere alienante; un approccio solo espressivo (solo Papert) rischia di perdere di vista la struttura logica. La sintesi sta nel vedere la tecnologia come

un "materiale per pensare" (alla Papert) per sviluppare un "pensiero per risolvere" (alla Wing). Ugolini (2024) parla di due "impostazioni archetipiche" per riassumere la posizione di Wing e quella di Papert, dove per la prima si tratta di un processo di problem solving e per il secondo quello di creazione e condivisione di oggetti computazionali.

1.3. Il problema del transfer: smontare un mito pedagogico

La chiarezza sulle finalità è ancor più urgente quando si affronta uno degli equivoci più pericolosi e persistenti legati al PCT: l'idea che esso sia di per sé un vettore automatico di competenze di alto livello (come il problem solving complesso, il pensiero critico o l'astrazione) trasferibili magicamente in qualsiasi contesto disciplinare. Questa convinzione, spesso data per scontata nella retorica politica e in molti corsi di formazione, trova scarso supporto nella ricerca empirica e nella teoria dell'apprendimento.

Già negli anni '80, Gavriel Salomon metteva in guardia contro la facilità con cui si ipotizzano "far transfer", ovvero il trasferimento di abilità cognitive da un dominio molto specifico (es. la programmazione in Logo) a domini lontani e dissimili (es. il rendimento in lettura o le abilità sociali). Salomon sottolineava come il transfer non sia un processo automatico, ma richieda una mediazione didattica esplicita e sia fortemente influenzato dalla similarità superficiale e strutturale tra i compiti, nonché dalla percezione che lo studente ha di tale similarità (Salomon, 1984). In altre parole, saper scomporre un problema in Scratch non implica automaticamente la capacità di scomporre un problema di comprensione del testo, a meno che il docente non aiuti attivamente gli studenti a costruire un ponte concettuale tra i due contesti, rendendo espliciti i processi mentali comuni.

Le ricerche più recenti di Ronny Scherer e colleghi confermano questo scetticismo. Una meta-analisi del 2019, che ha esaminato decine di studi sul tema, ha riscontrato che l'apprendimento della programmazione ha effetti di transfer positivi, ma limitati e specifici (Scherer, Siddiq & Sánchez Viveros, 2019). I benefici cognitivi più consistenti si osservano in domini vicini, come le abilità matematiche e spaziali, mentre le evidenze per un miglioramento delle abilità di ragionamento verbale o del problem solving generale sono deboli e inconsistenti. Scherer aveva già segnalato nel 2016 la necessità di andare oltre l'entusiasmo aneddotico e di pretendere evidenze empiriche solide prima di propagandare il coding come una panacea per lo sviluppo cognitivo (Scherer, 2016).

Questa visione critica è ripresa dall'analisi di Lodi e Martini (2021), che rileggono la storia del PCT proprio attraverso la lente di questa tensione irrisolta. Il dibattito tra Papert e Wing, sostengono gli autori, può essere interpretato anche come un dibattito sul transfer: per Papert, il "valore" del coding non sta in un ipotetico transfer di abilità, ma nell'esperienza diretta, significativa e motivante che esso offre nel qui e ora dell'apprendimento della matematica. Per Wing, invece, il valore sta proprio nella presunta trasferibilità universale dei suoi principi logici. L'affermazione di Wing secondo cui il PCT è una competenza per tutti i campi presuppone un livello di transfer che la ricerca psicopedagogica non conferma.

Alla luce di queste evidenze, appare chiaro che promuovere il PCT con l'argomentazione che "sviluppa il problem solving" è fuorviante e scientificamente debole. Il problem solving non è una competenza monolitica e astratta; è una famiglia di pratiche che si declinano in modo profondamente diverso a seconda del contesto disciplinare, delle conoscenze pregresse e della natura del problema. Risolvere un enigma logico, una disequazione, un conflitto sociale o un guasto meccanico attivano processi cognitivi e conoscenze molto diverse. Sostenere che il coding, di per sé, insegni a risolvere problemi "in generale" non è solo inesatto, ma rischia di deresponsabilizzare la scuola dal compito, ben più complesso, di insegnare il problem solving dentro ogni disciplina, con le sue specifiche strategie e codici.

1.4. Oltre il transfer: autoefficacia, motivazione intrinseca e accesso alla disciplina

Se l'argomento del transfer è fragile, su quali solide basi pedagogiche possiamo allora fondare l'introduzione del coding e della robotica a scuola? La risposta va cercata tornando alla lezione di Papert e spostando il focus dagli ipotetici effetti collaterali a lungo termine agli effetti diretti e osservabili che una didattica ben condotta può produrre. Due concetti diventano centrali: l'autoefficacia e la motivazione intrinseca.

A nostro avviso, la scelta più fruttuosa è quella di ritornare alle origini pedagogiche del concetto, prima ancora che informatiche. Seymour Papert, matematico e pedagogista, utilizzò il termine molto prima di Wing, ma con un intento radicalmente diverso (Lodi & Martini, 2021). Papert, che si considerava un prosecutore dell'attivismo di John Dewey, non era interessato a formare dei programmatori. Il suo obiettivo era permettere agli studenti

un accesso personale e significativo alle discipline – nel suo caso, la matematica – attraverso la creazione attiva con un linguaggio di programmazione (il LOGO) e con artefatti robotici (Papert, 1984; 1996). Per Papert, l'acquisizione di competenze informatiche era "accidentale", secondaria e funzionale all'obiettivo primario: conoscere la matematica "facendola", costruendo oggetti concettuali e fisici. Questo approccio inverte la prospettiva: non si impara a programmare per diventare programmatori; si utilizza la programmazione (o la robotica) come una potente "porta d'accesso" alle discipline, che nel contempo, in modo secondario ma naturale, forgia anche una competenza digitale (Nulli, Miotti & Di Stasio, 2022).

Mentre Wing parte dall'informatica per esportarne i principi nel mondo, Papert parte dal bambino e dalle sue modalità di apprendimento (il 'costruzionismo') per importare nella scuola gli strumenti che meglio ne supportano l'esplorazione e la costruzione di significato. Per Wing, il pensiero computazionale è un prodotto dell'informatica da disseminare; per Papert, è un processo di pensiero che fiorisce naturalmente in un ambiente ricco di oggetti 'parlanti' (come la tartaruga del LOGO), e che l'informatica ha il merito di aver reso evidente e potenziato. In questa visione, il valore del coding non risiede in un misterioso "potenziamento cognitivo", ma nella sua capacità di rendere il pensiero visibile, manipolabile e condivisibile. Quando uno studente programma un robot per disegnare un quadrato, non sta solo imparando un comando; sta concretizzando la sua comprensione degli angoli, delle ripetizioni, della precisione. Sta agendo la matematica, non subendola.

È in questa azione significativa che si coltivano l'autoefficacia e la motivazione intrinseca. L'autoefficacia (la convinzione di essere capaci di organizzare e portare a termine un compito specifico) cresce quando lo studente sperimenta, attraverso piccoli successi incrementali (il robot si muove! La sequenza funziona!), di avere un controllo attivo sul proprio apprendimento e sugli strumenti. Il coding, con il suo ciclo rapido di prova-errore-correzione (debugging), fornisce un feedback immediato e non giudicante: l'errore non è un fallimento, ma un'istruzione che non ha funzionato come previsto, e che può essere sistemata. Questa esperienza costruisce una mentalità di crescita e una resilienza di fronte alla difficoltà. La motivazione intrinseca (fare qualcosa per il piacere e la soddisfazione intrinseca dell'attività stessa) è alimentata dalla natura creativa, esplorativa e spesso ludica del coding e della robotica. Costruire un videogioco, animare una storia, far compiere una missione a un robot sono compiti autentici che hanno un senso per lo studente, spostando il focus dal "dover imparare"

al "voler creare". Questo senso di agency e di piacere è il vero, potente motore dell'apprendimento profondo. È questa percezione di competenza e di controllo sul proprio apprendimento che, a differenza di un astratto "pensiero computazionale", può tradursi in un atteggiamento positivo verso lo studio, in un metodo di lavoro perseverante e in una maggiore disponibilità ad affrontare sfide anche in altri ambiti. Non è un transfer di abilità, ma un trasferimento di un atteggiamento positivo verso l'apprendimento (*learning disposition*), che è molto più prezioso e realistico.

BOX DI APPROFONDIMENTO B: Il quadro normativo italiano – Tra opportunità e ambiguità.

Documento (Anno)	Riferimenti espliciti e impliciti al PCT	Commento critico e implicazioni didattiche
Piano Nazionale Scuola Digitale (PNSD) (MIUR, 2015)	– Azione #17: "Portare il pensiero computazionale a tutta la scuola primaria". Formazione di docenti e diffusione di risorse. – Azione #15: "Scenari innovativi per lo sviluppo di competenze digitali applicate". Finanziamento per atelier creativi (making, coding, robotica).	Il PNSD è il volano operativo e finanziario che ha lanciato il tema. Tuttavia, l'Azione #17 è emblematica dell'ambiguità: promuove il "PCT" senza definirlo, lasciando a corsi di formazione eterogenei il compito di colmare il vuoto. L'abbinamento con gli atelier creativi (Azione #15) suggerisce però una via praticabile: interpretare il PCT in senso papertiano, come apprendimento attraverso la creazione di artefatti in ambienti laboratoriali.
Indicazioni Nazionali e Nuovi Scenari (MIUR, 2018)	– Introduzione esplicita del termine "pensiero computazionale" tra le competenze di base. – Lo definisce come capacità di "risolvere problemi applicando la logica, analizzando i dati" e di "affrontare problemi complessi scomponendoli in più parti". – Lo collega alla cittadinanza digitale.	Questo è il documento che legittima curricolarmente il PCT, elevandolo a competenza trasversale. La definizione adottata mischia abilmente la seconda e la terza area del JRC (concetti informatici e problem solving). Il collegamento con la cittadinanza digitale è cruciale: invita a non fermarsi alla tecnica, ma a discutere l'etica degli algoritmi, la privacy, il bias nei dati – temi che saranno centrali nel Capitolo 4.2 sull'IA generativa. Per il docente, è un mandato a progettare attività che abbiano una dimensione critica e riflessiva.

Documento (Anno)	Riferimenti espliciti e impliciti al PCT	Commento critico e implicazioni didattiche
Linee Guida per l'Educazione al Rispetto (MIUR, 2017)	– Promuove una didattica che valorizzi "tutte le intelligenze" e "pluralità di linguaggi". – Invita a superare stereotipi, anche di genere, nelle attitudini verso le materie scientifiche.	Questo documento, apparentemente lontano, fornisce la cornice inclusiva ed etica entro cui collocare le attività di PCT. Un approccio laboratoriale e creativo al coding può essere uno strumento potentissimo per coinvolgere tutti gli studenti, contrastando gli stereotipi di genere che allontanano le ragazze dalla tecnologia – tema che esploreremo in profondità nel Capitolo 7. Ricorda al docente che l'obiettivo non è selezionare i "più bravi in informatica", ma offrire a tutti un nuovo canale espressivo e di ragionamento.

Leggere la normativa con spirito critico: I documenti offrono un mandato forte ma generico. Sta al docente, come professionista riflessivo, interpretarli alla luce della ricerca: evitare la trappola del "transfer magico", privilegiare un approccio costruzionista (Papert) che ponga al centro l'esperienza significativa dello studente, e utilizzare le tecnologie non come fine, ma come mezzo per l'accesso alla conoscenza, lo sviluppo dell'autoefficacia e la crescita di cittadini consapevoli.

1.5. Conclusione: per una didattica chiara e fondata

In conclusione, se vogliamo promuovere competenze di problem solving, pensiero logico e creatività digitale, è più produttivo, onesto e chiaro per il docente fare riferimento a due assi distinti ma sinergici, abbandonando definitivamente la vaghezza contraddittoria del "termine-cappello":

1. La progettazione di una didattica attiva e laboratoriale (come l'*Inquiry-Based Learning* o il *Project-Based Learning*), in cui il docente costruisce contesti autentici in cui lo studente è chiamato a ricercare, agire, costruire e collaborare. In questi contesti, il coding e la robotica possono trovare una collocazione naturale come strumenti potenti di esplorazione e creazione.

2. L'utilizzo esplicito e contestualizzato del pensiero algoritmico e dei concetti informatici, intesi non come astrazioni universali, ma come strumenti definiti e definibili all'interno di specifici contesti disciplinari o progettuali. Insegniamo la scomposizione per realizzare un'animazione; insegniamo i loop per programmare una sequenza di movimenti; insegniamo le condizioni per far reagire un personaggio a un evento.

Questa distinzione ci libera dalla retorica e ci restituisce strumenti concettuali precisi per progettare interventi didattici efficaci e fondati. Ci permette di rispondere in modo convincente alla domanda "perché fare coding a scuola?": non per un vago "sviluppare il pensiero computazionale", ma per dare agli studenti un linguaggio in più per esprimersi, uno strumento in più per esplorare il mondo, e un'esperienza concreta di competenza e agency in un'epoca digitale. La tecnologia, in questa visione, non è il fine, ma un mezzo potente – tra gli altri – per l'accesso alla conoscenza, la crescita personale dello studente e la formazione di un cittadino capace di comprendere, e non solo di subire, la trasformazione algoritmica della società.

Riferimenti bibliografici

Balanskat, A., & Engelhardt, K. (2015). *Computing our Future. Computer Programming and Coding Priorities, School Curricula and Initiatives Across Europe*. European Schoolnet.

Bocconi, S., Chioccariello, A., & Earp, J. (2018). *The Nordic Approach to Introducing Computational Thinking and Programming in Compulsory Education*. Report for the Nordic@BETT2018 Steering Group.

Bocconi, S., et al. (2016). *Developing Computational Thinking in Compulsory Education: Implications for Policy and Practice*. Publications Office of the European Union.

Bocconi, S., et al. (2022). *Reviewing Computational Thinking in Compulsory Education* (A. Inamorato Dos Santos, R. Cachia, N. Giannoutsou, Y. Punie, A cura di). Publications Office of the European Union.

Lodi, M., & Martini, S. (2021). Computational Thinking, between Papert and Wing. *Science and Education*, 30(4), 883–908.

MIUR – Ministero dell'Istruzione, dell'Università e della Ricerca. (2015). *Piano Nazionale Scuola Digitale*.

MIUR – Ministero dell'Istruzione, dell'Università e della Ricerca. (2018). *Indicazioni Nazionali e Nuovi Scenari*.

Nulli, G., Miotti, B., & Di Stasio, M. (2022). *Robotica educativa e coding: strumenti per la trasformazione del curricolo*. Carocci.

Papert, S. (1984). *Mindstorms. Bambini, computer e creatività*. Emme. (Ed. orig. 1980).

Papert, S. (1996). An Exploration in the Space of Mathematics Educations. *International Journal of Computers for Mathematical Learning*, 1(1), 95–123.

Salomon, G. (1984). On ability development and far transfer: A response to Pea and Kurland. *New Ideas in Psychology*, 2(2), 169-174.

Scherer, R. (2016). Learning from the Past: The Need for Empirical Evidence on the Transfer Effects of Computer Programming. In *Proceedings of the 7th Workshop in Primary and Secondary Computing Education (WiPSCE '16)*. ACM.

Scherer, R., Siddiq, F., & Sánchez Viveros, B. (2019). The cognitive benefits of learning computer programming: A meta-analysis of transfer effects. *Journal of Educational Psychology*, 111(5), 764–792.

The Royal Society. (2012). *Shut down or restart? The way forward for computing in UK schools*.

Ugolini, Francesco Claudio (2024). Coding e sviluppo del pensiero computazionale. Mondadori Università.

Wing, J. (2006). Computational Thinking. *Communications of the ACM*, 49(3), 33-35.

Wing, J. (2008). Computational Thinking and Thinking about Computing. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 366(1881), 3717-3725.

Wing, J. (2010). Computational Thinking: What and Why? *The Link Magazine*, The Carnegie Mellon University School of Computer Science.

2.
COMPETENZA DIGITALE: TRA FRAMEWORK EUROPEO E PRATICA DIDATTICA

Giovanni Nulli

2.1. La competenza digitale: un framework dinamico per la cittadinanza e la didattica

L'introduzione delle tecnologie digitali nella didattica, e in particolare la promozione del pensiero computazionale, del coding e della robotica educativa, non avviene in un vuoto concettuale. Queste pratiche si inscrivono in un quadro di riferimento più ampio e strutturato: lo sviluppo della competenza digitale come competenza chiave per il cittadino del XXI secolo. Questo capitolo offre una panoramica di tale concetto, con particolare riferimento al DigComp, il quadro di riferimento europeo sviluppato dal Joint Research Centre (JRC) della Commissione Europea. Comprendere il DigComp non è un esercizio burocratico, ma un passaggio strategico per il docente: permette di collocare le attività di programmazione all'interno di un orizzonte educativo completo, di coglierne le intersezioni con altre dimensioni fondamentali – dalla gestione consapevole dei dati alla cittadinanza digitale responsabile – e di aderire a un linguaggio comune e scientificamente fondato, in continua evoluzione.

Perché dedicare un modulo alla competenza digitale in una formazione focalizzata su pensiero computazionale e robotica? La risposta è duplice e strategica. In primo luogo, la competenza digitale è una delle otto competenze chiave per l'apprendimento permanente definite dall'Unione Europea e recepite nelle Indicazioni Nazionali italiane fin dal 2007 (Raccomandazione del Parlamento Europeo e del Consiglio del 18 dicembre 2006, recepita in Italia nel 2007). La programmazione ne costituisce una componente specifica, ma non esaustiva. In secondo luogo, esiste una relazione di mutuo sostegno tra una solida competenza digitale di base e l'affinamento del pensiero computazionale. Ad esempio, una buona conoscenza di come le immagini siano rappresentate digitalmente (formati, risoluzione, compressione) facilita la successiva scrittura di algoritmi che

le elaborano. Più in generale, un uso critico e consapevole degli strumenti digitali crea un terreno fertile per approcciare la logica della programmazione con maggiore dimestichezza e profondità. Il docente che conosce il framework può dunque progettare attività che, partendo dalla competenza digitale in senso lato, costruiscano naturalmente il ponte verso il pensiero computazionale, rafforzando negli studenti un senso di autoefficacia che nasce dalla padronanza progressiva di un sistema integrato di abilità.

La nozione di competenza digitale emerge con forza nel solco della didattica per competenze e in risposta alla progressiva digitalizzazione della società, che investe tanto la sfera professionale quanto quella della vita quotidiana e dei servizi al cittadino. La definizione del 2006, ancora valida nelle sue linee essenziali, la descrive come il «saper utilizzare con dimestichezza e spirito critico le tecnologie della società dell'informazione per il lavoro, il tempo libero e la comunicazione» (Raccomandazione 2006/962/CE). Essa implica «abilità di base nelle tecnologie dell'informazione e comunicazione (TIC)», come l'uso del computer per «reperire, valutare, conservare, produrre, presentare e scambiare informazioni», nonché per «comunicare e partecipare a reti collaborative tramite Internet». Tuttavia, una definizione così ampia rischiava di rimanere astratta senza uno strumento operativo per declinarla in conoscenze, abilità e attitudini osservabili.

Per colmare questo gap, la Commissione Europea ha incaricato il Joint Research Centre (JRC) – il suo servizio di scienza e conoscenza – di sviluppare un framework scientificamente fondato. Il lavoro, iniziato nel 2010, è partito da una precisa domanda di ricerca: «Quali sono le componenti della competenza digitale e quale tipo di conoscenze, abilità e attitudini le persone dovrebbero avere per essere digitalmente competenti al giorno d'oggi e nel futuro?». Il processo, iterativo e basato su ampia consultazione, è stato meticoloso. Fase 1: Analisi e sintesi (2010-2012). Il JRC condusse una revisione sistematica della letteratura accademica e una analisi comparativa di oltre 15 framework preesistenti utilizzati in diversi paesi e contesti (come ECDL, ISTE NETS, UNESCO ICT-CFT). Questo lavoro di mappatura, documentato in rapporti preliminari, servì a identificare i temi e i costrutti ricorrenti (Vuorikari et al., 2016). Fase 2: Consultazione e validazione (2012-2013). Il draft iniziale del framework fu sottoposto a un processo di validazione attraverso consultazioni con esperti internazionali di digitale e istruzione, e a test sul campo con gruppi di cittadini per verificarne la comprensi-

bilità e la pertinenza. Questo passaggio fu cruciale per garantire che il DigComp non fosse un costrutto puramente teorico, ma rispecchiasse bisogni reali e fosse applicabile in contesti diversi. Fase 3: Pubblicazione e diffusione (2013). Il frutto di questo lavoro fu la pubblicazione della prima versione del DigComp 1.0 nel 2013 (Carretero, Vuorikari & Punie, 2017), che stabilì l'architettura fondamentale delle 5 aree e 21 competenze.

Questo percorso di continua revisione non si è fermato. Le versioni successive (2.0 nel 2016, 2.1 nel 2017) hanno apportato migliorie progressive. Il DigComp 2.0 ha introdotto un linguaggio più coerente e orientato all'azione per i descrittori delle competenze, rendendo il framework più utilizzabile per la progettazione didattica. Il DigComp 2.1 ha arricchito il quadro con esempi di utilizzo concreti per ciascun livello di competenza, fornendo agli educatori scenari applicativi immediati. La versione DigComp 2.2 (2022) ha rappresentato un aggiornamento sostanziale, integrando nuove competenze emerse con forza nel frattempo, come la capacità di interagire con sistemi di Intelligenza Artificiale (IA) in modo consapevole e critico, e di riconoscere e contrastare la disinformazione online. Questa evoluzione evidenzia un aspetto cruciale: la competenza digitale non è un insieme statico di conoscenze da memorizzare, ma un ambito dinamico in evoluzione, che la scuola può seguire e a cui può contribuire attivamente, ad esempio documentando e riflettendo sulle pratiche didattiche più efficaci.

Il JRC ha successivamente declinato il framework per diversi target, dando vita al DigCompEdu per i docenti (un vero e proprio strumento di autovalutazione e sviluppo professionale), al DigCompOrg per le organizzazioni educative (per valutare e pianificare la trasformazione digitale a livello di istituto) e a strumenti di autovalutazione come il DigCompSAT. Per il nostro scopo, ci concentreremo sul framework per il cittadino (DigComp), la cui struttura offre una mappa preziosa per orientare la progettazione didattica. Il DigComp organizza la competenza digitale in cinque aree distinte e interdipendenti:

1. Alfabetizzazione su informazioni e dati: identificare, localizzare, recuperare, archiviare, organizzare e analizzare informazioni digitali, valutandone rilevanza e affidabilità. È la base per ogni apprendimento in rete e prelude al pensiero critico verso il flusso informativo.
2. Comunicazione e collaborazione: comunicare in ambienti digitali, condividere risorse, collaborare attraverso strumenti digitali, ma an-

che gestire la propria identità digitale in modo consapevole. Quest'area è centrale per la didattica collaborativa e per la formazione di una cittadinanza digitale attiva e responsabile.

3. Creazione di contenuti digitali: sviluppare e integrare contenuti digitali, rielaborare conoscenze ed espressioni artistiche. È in quest'area che trova esplicita collocazione la programmazione, intesa come competenza specifica per creare artefatti digitali complessi e non solo consumarli.
4. Sicurezza: proteggere dispositivi, dati personali, salute e benessere nell'ambiente digitale. Un aspetto cruciale, che include la capacità di proteggersi da cyber-rischi e di comprendere l'impatto psicologico e sociale della vita online. È il fondamento di un rapporto sano e sostenibile con la tecnologia.
5. Risoluzione di problemi: identificare bisogni e problemi tecnici, scegliere strumenti digitali adeguati, risolvere problemi concettuali attraverso mezzi digitali, aggiornare le proprie competenze. Quest'area sintetizza e applica tutte le altre in contesti reali, ed è quella che più direttamente dialoga con il problem solving e il pensiero computazionale.

Questa architettura chiarisce come il pensiero computazionale (soprattutto nell'area 3) non sia isolato, ma si relazioni strettamente con la capacità di valutare criticamente le informazioni (area 1), di interagire in modo etico e sicuro (aree 2 e 4) e di adattare gli strumenti a problemi reali (area 5). Ad esempio, un progetto di robotica che preveda la raccolta e l'analisi di dati ambientali tocca trasversalmente tutte e cinque le aree: ricerca di informazioni sul sensore (A1), condivisione dei dati in cloud (A2), programmazione del robot (A3), attenzione alla sicurezza dei dati (A4), risoluzione degli errori di misura (A5).

Un'innovazione centrale del DigComp è l'introduzione di un sistema di livelli di padronanza (da base a altamente specializzato), articolati lungo tre dimensioni: la complessità dei compiti, il grado di autonomia dell'utente e il dominio cognitivo richiesto (dal semplice ricordo alla creazione e valutazione critica, secondo una rilettura della Tassonomia di Bloom). Questo sistema, pensato per il *lifelong learning*, riconosce che la competenza digitale si sviluppa lungo un continuum. Per la scuola, è uno strumento prezioso per progettare progressioni didattiche graduali e per valutare non solo la correttezza tecnica, ma anche l'autonomia e la riflessività degli studenti nell'uso del digitale.

In conclusione, fare riferimento al framework DigComp nell'insegnamento del pensiero computazionale significa adottare una prospettiva sistemica, pedagogicamente fondata e al passo con i temi. Significa:

1. Contestualizzare il coding e la robotica all'interno di un panorama più ampio di cittadinanza digitale, andando oltre la tecnica.
2. Collegare esplicitamente la logica della programmazione ad altre competenze essenziali, come la gestione dell'identità online, la sicurezza e il problem solving, in un'ottica di formazione integrale della persona.
3. Partecipare a un linguaggio comune europeo, che rende le competenze degli studenti riconoscibili, validabili e spendibili in un contesto più vasto, sia formativo che professionale.
4. Seguire un modello dinamico e basato su evidenza, che invita a un aggiornamento continuo e a una progettazione didattica riflessiva, come dimostra l'evoluzione verso il DigComp 3.0.

In questa luce, la competenza digitale cessa di essere un semplice corollario tecnologico e diventa il quadro di senso in cui inserire ogni intervento didattico con le tecnologie, garantendo che esso contribuisca a formare non solo utenti abili, ma cittadini consapevoli, critici, creativi e resilienti, pronti ad affrontare le sfide di un mondo in costante trasformazione.

2.2. DigComp 3.0: L'evoluzione del framework verso le sfide digitali del presente

Il 27 novembre 2025 è uscito il nuovo DigComp 3.0, pertanto abbiamo deciso di descrivere questa nuova versione del framework. In questo modo potremo approfondire la struttura delle precedenti edizioni, in quanto la versione 3.0 è sviluppata sulle linee generali delle versioni precedenti e ne aggiorna i contenuti in relazione agli sviluppi tecnologici e politici più recenti. In questo modo daremo contezza delle ragioni che stanno muovendo la Commissione Europea a investire risorse sia per lo sviluppo sia per l'aggiornamento dello strumento DigComp.

L'evoluzione del DigComp ha seguito nel tempo una precisa linea di rigore che ne mantiene la solidità. L'obiettivo è quello di mantenere in

modo critico una definizione di competenza digitale che sia stabile e centrata su competenze stabili che prescindano dalle tecnologie. Il DigComp 3.0 non rivoluziona l'impianto – le cinque aree e le ventuno competenze restano – ma lo raffina e lo consolida sulla base di anni di utilizzo e di feedback da parte di esperti e stakeholder. La vera novità risiede nell'aggiornamento dei contenuti per riflettere i rapidi sviluppi tecnologici (dall'intelligenza artificiale generativa alla cybersecurity) e nell'introduzione, per la prima volta, di un approccio basato sui *learning outcomes* – oltre cinquecento risultati di apprendimento declinati in conoscenze, abilità e attitudini.

2.2.1. Perché un nuovo aggiornamento? Il contesto delle competenze digitali in Europa

La spinta a pubblicare una nuova edizione del framework non è accidentale. I dati raccolti dalla Commissione Europea dipingono un quadro che richiede interventi urgenti e coordinati. Nel 2023, solo il 56% degli adulti europei possedeva competenze digitali di base, un dato lontano dall'obiettivo dell'80% fissato per il 2030 dal Programma Strategico per il Decennio Digitale. Ancora più allarmante è la situazione tra i giovani: nello stesso anno, il 43% degli studenti della scuola secondaria non raggiungeva un livello sufficiente di competenze digitali di base.

La penetrazione delle tecnologie nel mondo del lavoro rende questi divari ancora più critici. Tra il 2024 e il 2025, oltre nove lavoratori su dieci (92%) nell'Unione Europea hanno utilizzato tecnologie digitali nella loro attività professionale, e il 30% ha dichiarato di usare sistemi di Intelligenza Artificiale (IA) nel lavoro. Parallelamente, nel 2024, il 42% dei lavoratori europei ha segnalato un gap di competenze in materia di IA, ma solo il 15% aveva partecipato a una formazione specifica sull'argomento. Chiudere queste distanze non è solo una questione di equità sociale, ma una necessità per la competitività delle economie europee.

2.2.2. Un quadro sinergico: le iniziative europee per la competenza digitale

Il DigComp 3.0 non nasce in un vuoto politico, ma si inserisce in modo sinergico in un ecosistema di iniziative che l'Unione Europea ha introdotto per promuovere le competenze digitali a tutti i livelli. Tra le più rilevanti troviamo:

- Il Programma Strategico per il Decennio Digitale (*Digital Decade Policy Programme*), che fissa obiettivi e target per la trasformazione digitale dell'Europa verso il 2030, monitorando regolarmente i progressi.
- L'Unione delle Competenze (*Union of Skills*) (comunicazione del marzo 2025), una strategia quadro che mira a garantire che ogni cittadino europeo possa sviluppare solide basi di competenze e impegnarsi nell'aggiornamento e riqualificazione continua (*lifelong upskilling and reskilling*). Al suo interno, il *Piano d'Azione per le Competenze di Base* (*Action Plan on Basic Skills*) cita esplicitamente DigComp 3.0 come uno strumento operativo.
- Il Piano d'Azione per l'Istruzione Digitale (*Digital Education Action Plan – DEAP*) 2021-2027, sotto il quale era stata completata la versione DigComp 2.2 e che ha contribuito a costruire la strada verso il 3.0.
- La Strategia Europea per un Internet più Sicuro per i Bambini (BIK+) (*Better Internet for Kids*), che mira a proteggere, rispettare e responsabilizzare i minori negli ambienti digitali.
- L'AI Continent Action Plan e la Apply AI Strategy, che promuovono l'alfabetizzazione all'IA tra i lavoratori di tutti i settori e incoraggiano l'adozione di approcci basati su framework come DigComp.
- Un corpus articolato di leggi e regolamenti digitali che definiscono diritti e doveri dei cittadini e dei fornitori di servizi: dal GDPR (Regolamento Generale sulla Protezione dei Dati) al Digital Services Act (lotta alla disinformazione e protezione degli utenti), dall'AI Act (gestione dei rischi dei sistemi di IA) al Cyber Resilience Act (sicurezza dei prodotti hardware e software), passando per il Digital Markets Act e la European Digital Identity Regulation (identità digitale europea).

Tutte queste iniziative condividono un bisogno comune: disporre di un linguaggio condiviso e di uno strumento di riferimento per definire, valutare e certificare le competenze digitali. DigComp 3.0 risponde proprio a questa esigenza, fornendo il quadro concettuale che permette di tradurre gli obiettivi politici in pratiche educative e formative concrete.

2.2.3. I principi guida e le novità del DigComp 3.0

Il DigComp 3.0 incarna i valori della Dichiarazione Europea sui Diritti e Principi Digitali per il Decennio Digitale (2023), che promuove una visione *human-centric* della trasformazione digitale, organizzata attorno a sei temi: le persone al centro, solidarietà e inclusione, libertà di scelta, partecipazione, sicurezza e sostenibilità.

Rispetto alla versione 2.2, il nuovo framework introduce aggiornamenti sostanziali guidati da cinque priorità emerse dalla consultazione con esperti e stakeholder: (1) competenza nell'IA (incluse quelle generative), (2) competenza in materia di cybersecurity, (3) diritti, scelta e responsabilità digitali, (4) benessere negli ambienti digitali e (5) capacità di affrontare disinformazione e *misinformation*.

Tuttavia, per comprendere appieno il salto di qualità rappresentato dal DigComp 3.0, è necessario entrare nel dettaglio della sua architettura. Nelle prossime pagine esploreremo la struttura delle aree e delle competenze, i nuovi livelli di padronanza e, soprattutto, il sistema dei *learning outcomes*, uno strumento prezioso per chi, come il docente, deve tradurre il framework in obiettivi di apprendimento concreti e osservabili.

2.2.4. Le differenze tra DigComp 2.2 e DigComp 3.0: continuità e innovazione

Prima di addentrarci nelle novità più rilevanti del DigComp 3.0, può essere utile comprendere cosa cambia – e cosa rimane – rispetto alla versione precedente. L'*Annex* 1 del documento ufficiale (pagg. 77-83) offre una comparazione puntuale che ci guida nell'evoluzione del framework. L'aspetto più significativo è che la struttura portante non viene stravolta: le aree di competenza restano cinque, le competenze restano ventuno. Ciò che cambia è il lessico, la profondità dei descrittori e l'introduzione di nuovi elementi che rendono il framework più operativo e aggiornato alle sfide del presente.

La tabella seguente (adattata da *Table* A1, p. 77 del documento JRC) sintetizza i principali cambiamenti strutturali tra le due versioni.

Componente	DigComp 2.2	DigComp 3.0	Tipo di cambiamento
Definizione di competenza digitale	Utilizzata la definizione del 2018 (Raccomandazione Consiglio UE)	Stessa definizione, invariata	Nessuna modifica
Aree di competenza (Dimension 1)	5 aree con titoli e descrittori	5 aree con titoli e descrittori aggiornati	Aggiornamento lessicale e concettuale
Competenze (Dimension 2)	21 competenze con titoli e descrittori	21 competenze con titoli e descrittori aggiornati	Aggiornamento lessicale e concettuale
Livelli di padronanza (Dimension 3)	8 livelli (da *Foundation* a *Highly Specialised*)	4 livelli (*Basic*, *Intermediate*, *Advanced*, *Highly Advanced*) + *competence statements*	Riorganizzazione e semplificazione
Esempi di conoscenze, abilità e attitudini (Dimension 4)	Presenti come esempi illustrativi	Non più presenti in questa forma; sostituiti dai *learning outcomes*	Sostituzione con nuovo approccio
Use cases (Dimension 5)	Scenari di impiego in contesti lavorativi e formativi	Non riprodotti in DigComp 3.0	Eliminati
Competence statements	Non presenti	Nuovi: descrizioni sintetiche per ogni competenza e livello	Nuova aggiunta
Learning outcomes	Non presenti	Nuovi: oltre 500 risultati di apprendimento classificati per conoscenze, abilità e attitudini	Nuova aggiunta
Glossario	Presente, limitato	Ampliato a circa 120 termini	Significativa espansione

Fonte: adattato da JRC, DigComp 3.0, Table A1, p. 77.

2.2.5. Il cambiamento lessicale delle aree di competenza

Le cinque aree del DigComp restano, ma i loro titoli sono stati rivisti per riflettere meglio l'evoluzione del dibattito europeo sulla competenza digitale. La tabella seguente mostra i cambiamenti principali.

DigComp 2.2	DigComp 3.0	Osservazione
1. *Information and data literacy*	1. *Information search, evaluation and management*	Il termine "literacy" viene sostituito da una descrizione più operativa: ricerca, valutazione e gestione delle informazioni
2. *Communication and collaboration*	2. *Communication and collaboration*	Titolo invariato, ma descrittore arricchito
3. *Digital content creation*	3. *Content creation*	Il termine "Digital" è rimosso in quanto superfluo
4. *Safety*	4. *Safety, wellbeing and responsible use*	Ampliamento significativo: si aggiungono benessere e uso responsabile
5. Problem solving	5. *Problem identification and solving*	Si aggiunge l'identificazione come fase preliminare e distinta

Questo aggiornamento lessicale non è puramente formale. Riflette una maturazione del pensiero europeo sulla competenza digitale: non si tratta più solo di "alfabetizzazione" (*literacy*) ma di capacità operative di ricerca, valutazione e gestione; non si tratta solo di "sicurezza" ma di benessere complessivo e uso responsabile; non si tratta solo di "risolvere problemi" ma di saperli prima identificare.

Perché cambiano i descrittori delle competenze?

Anche le ventuno competenze sottostanti hanno subito aggiornamenti lessicali e concettuali. La logica che guida questi cambiamenti è triplice:

- Aggiornamento tecnologico – tenere conto dell'evoluzione degli strumenti digitali, in particolare dei sistemi di IA.
- Chiarezza e coerenza – rendere i descrittori più comprensibili e uniformi tra loro.
- Enfasi su etica e responsabilità – introdurre esplicitamente il riferimento a un uso etico e responsabile delle tecnologie.

Per dare un'idea concreta di come si manifesta questo aggiornamento, si consideri l'esempio della Competenza 2.5, che in DigComp 2.2 era denominata *Netiquette* e in DigComp 3.0 diventa *Digital behaviour*. Ecco come cambia il descrittore (barrato ciò che è stato eliminato, sottolineato ciò che è stato aggiunto):

2.2.6. DigComp 2.2 → DigComp 3.0

To be aware of behavioural norms, and to know how to behave respectfully while using digital technologies and interacting in digital environments. To adapt communication to specific contexts, and to be aware of and respect cultural, generational and other diversity in digital environments.

L'esempio mostra bene la direzione del cambiamento: si passa da un approccio centrato sulla conoscenza delle norme (*know-how*) a un approccio centrato sul comportamento rispettoso; si estende l'adattamento della comunicazione dall'"audience specifica" ai più ampi "contesti specifici"; si riconosce esplicitamente che la diversità può essere di tipo culturale, generazionale o altro (abilità, background sociale, ecc.).

Un altro esempio significativo riguarda la Competenza 4.3, che da *Protecting health and well-being* diventa *Supporting wellbeing* – il verbo cambia da "proteggere" a "sostenere", segnalando un approccio meno difensivo e più orientato alla promozione attiva del benessere.

Questi aggiustamenti, che possono sembrare minuti, hanno invece un impatto profondo sulla progettazione didattica. Un docente che legge "sostenere il benessere" invece di "proteggere la salute" è orientato verso attività diverse: non solo insegnare a evitare i rischi, ma anche a usare la tecnologia per il proprio benessere psicofisico e sociale.

2.2.7. I nuovi *competence statements*

Una delle innovazioni più rilevanti del DigComp 3.0 è l'introduzione dei *competence statements*. Si tratta di descrizioni sintetiche per ciascuna delle 21 competenze, declinate per ognuno dei quattro livelli di padronanza (*Basic*, *Intermediate*, *Advanced*, *Highly Advanced*). A differenza dei *learning outcomes* (che vedremo più avanti), i *competence statements* non distinguono tra conoscenze, abilità e attitudini, ma contengono tutti gli elementi chiave dei *learning outcomes* in una forma più compatta e immediata.

Perché sono utili? Perché offrono a chi progetta percorsi formativi (un docente, un formatore, un responsabile delle risorse umane) una visione d'insieme di ciò che ci si può attendere da un individuo a un dato livello di padronanza, senza dover consultare l'intera batteria dei *learning outcomes*. Sono lo strumento ideale per una prima lettura orientativa del framework.

Nel documento ufficiale, i *competence statements* occupano la Sezione 3 (pagg. 28-51) e sono organizzati per area e competenza, con un sistema di etichettatura che segnala se la competenza ha una rilevanza esplicita o implicita rispetto all'IA ([AI-E] o [AI-I]).

2.2.8. I livelli di padronanza: da otto a quattro

Forse il cambiamento più visibile rispetto al passato riguarda i livelli di padronanza. Nelle versioni precedenti del DigComp (dalla 2.0 alla 2.2), la progressione era articolata su otto livelli, che andavano da *Foundation* (livello base) a *Highly Specialised* (altamente specializzato). Questa articolazione, sebbene utile per una valutazione fine delle competenze, risultava talvolta complessa da applicare in contesti scolastici e formativi ordinari.

Il DigComp 3.0 riduce i livelli a quattro: *Basic*, *Intermediate*, *Advanced*, *Highly Advanced*. La tabella seguente (adattata da *Table* A3, p. 83 del documento JRC) mostra la corrispondenza tra i nuovi quattro livelli e i precedenti otto, nonché un'alternativa a sei livelli utilizzata in alcuni contesti nazionali.

DigComp 3.0	Corrispondenza con gli 8 livelli 2.2	Descrizione sintetica	Scopo
Basic	Livelli 1-2	Con guida se necessaria, l'individuo ricorda e implementa compiti semplici	Sostenere obiettivi personali, di apprendimento e/o lavorativi e partecipare alla società
Intermediate	Livelli 3-4	In autonomia, identifica e implementa compiti ben definiti e risolve problemi ben definiti	Sostenere obiettivi personali, di apprendimento e/o lavorativi e partecipare autonomamente alla società
Advanced	Livelli 5-6	Valuta e applica soluzioni a una varietà di compiti complessi in autonomia, adattandosi a contesti diversi; guida altri se necessario	Sostenere obiettivi personali, di apprendimento e/o lavorativi complessi
Highly Advanced	Livelli 7-8	Valuta, risolve problemi altamente complessi o specializzati, crea nuove soluzioni o adatta quelle esistenti; guida altri	Sostenere obiettivi personali, di apprendimento e/o lavorativi altamente specializzati

Fonte: adattato da JRC, DigComp 3.0, Table A3, p. 83.

Per comprendere il senso di questa semplificazione, è utile ricordare brevemente come funzionavano i livelli nelle versioni precedenti. Il DigComp 2.1 e 2.2 articolavano la padronanza secondo tre dimensioni: la complessità dei compiti (da semplici a complessi), il grado di autonomia (da guidato a autonomo) e il dominio cognitivo (una rilettura della tassonomia di Bloom: dal ricordare al creare, passando per comprendere, applicare, analizzare, valutare). Gli otto livelli erano il prodotto incrociato di queste dimensioni. Ad esempio, il livello 2 (Foundation) corrispondeva a compiti semplici eseguiti con autonomia; il livello 4 (*Intermediate*) a compiti ben definiti eseguiti in autonomia; il livello 6 (*Advanced*) a compiti complessi eseguiti in autonomia; e così via.

Questa architettura, raffinata e coerente, aveva però un costo in termini di usabilità. Per un docente che deve progettare un'attività per una classe di studenti della secondaria di primo grado, distinguere tra un livello 3 e un livello 4 poteva risultare eccessivamente analitico. Inoltre, molti degli utilizzatori del framework (scuole, enti di formazione, servizi per l'impiego) hanno progressivamente adottato mappature semplificate, riducendo gli otto livelli a quattro o sei, per adattarli ai propri contesti.

Il DigComp 3.0 raccoglie questa lezione e incorpora nella versione ufficiale ciò che era già una pratica diffusa. La riduzione a quattro livelli non è una perdita di precisione, ma un allineamento alle esigenze reali degli utilizzatori. La tabella sopra mostra come i livelli 1-2 del vecchio sistema convergano nel nuovo *Basic*; i livelli 3-4 in *Intermediate*; i livelli 5-6 in *Advanced*; i livelli 7-8 in *Highly Advanced*.

Una precisazione importante riguarda il livello *Highly Advanced*. Nel DigComp 3.0, questo livello non coincide con le competenze dello specialista ICT (informatico professionista). Piuttosto, descrive un cittadino digitalmente competente che è in grado di affrontare problemi altamente complessi o specializzati utilizzando strumenti digitali, di creare nuove soluzioni e di guidare altri nello sviluppo delle competenze digitali. È un livello raggiungibile anche da chi non è un tecnico informatico, ma opera in contesti professionali avanzati (un insegnante che progetta percorsi innovativi, un ricercatore che analizza dati complessi, un manager che guida la trasformazione digitale del proprio team).

La tabella offre anche una colonna con una mappatura alternativa a sei livelli, utilizzata in alcuni paesi e contesti. Questa flessibilità conferma la natura non prescrittiva del framework: i livelli di padronanza vanno adattati e localizzati in base al contesto di utilizzo, all'età dei soggetti e agli obiettivi formativi.

2.2.9. Dalla Dimensione 4 ai *learning outcomes*

Nelle versioni precedenti del DigComp (2.0, 2.1, 2.2), la cosiddetta Dimensione 4 conteneva esempi di conoscenze, abilità e attitudini per ciascuna competenza e livello di padronanza. Questi esempi avevano uno scopo puramente illustrativo: aiutare il lettore a comprendere il significato di una competenza fornendo casi concreti. Tuttavia, non avevano la pretesa di essere esaustivi né di costituire un sistema coerente di risultati di apprendimento.

Il DigComp 3.0 va oltre. La Dimensione 4 non viene semplicemente aggiornata, ma sostituita da un sistema organico di *learning outcomes* (risultati di apprendimento). La differenza è sostanziale:

- Negli esempi del passato, l'approccio era *esemplificativo*: "ecco alcune cose che una persona potrebbe sapere o saper fare".
- Nei *learning outcomes*, l'approccio è *prescrittivo e classificatorio*: "questo è ciò che ci si aspetta che una persona sappia, comprenda o sia in grado di fare dopo un processo di apprendimento".

I *learning outcomes* del DigComp 3.0 sono 523 in totale, distribuiti su tutti i livelli di padronanza (29% *Basic*, 32% *Intermediate*, 23% *Advanced*, 16% *Highly Advanced*) e classificati per tipo: conoscenze (42%, 217), abilità (38%, 199) e attitudini (20%, 107). Ogni *learning outcome* ha un codice identificativo unico (es. LO1.1.01) ed è etichettato anche in base alla sua rilevanza rispetto all'IA (esplicita, implicita o assente).

Perché questa innovazione è importante? Perché i *learning outcomes* sono uno strumento consolidato nelle politiche europee di educazione e formazione. Come sottolinea il Cedefop (il Centro europeo per lo sviluppo della formazione professionale), i *learning outcomes* fungono da "collante" tra il mondo dell'istruzione e quello del lavoro: guidano lo sviluppo dei curricoli, supportano la valutazione, facilitano il riconoscimento delle competenze acquisite in contesti formali, non formali e informali, e contribuiscono alla trasparenza delle qualifiche.

Per un docente, avere a disposizione i *learning outcomes* del DigComp 3.0 significa poter tradurre direttamente il framework in obiettivi di apprendimento per le proprie classi, senza dover operare complesse mediazioni interpretative. Significa anche poter valutare in modo più trasparente e confrontabile il livello di competenza digitale degli studenti.

2.2.10. Cosa non c'è più: gli *use cases* (Dimensione 5)

Il DigComp 2.2 includeva una Dimensione 5 dedicata agli *use cases* (casi d'uso) – scenari di impiego della competenza digitale in contesti lavorativi e formativi. Questi esempi, sebbene utili per illustrare applicazioni concrete, presentavano il rischio di invecchiare rapidamente a fronte dell'evoluzione tecnologica.

Nel DigComp 3.0, gli *use cases* non sono stati riprodotti. La scelta della Commissione Europea è stata quella di concentrare le risorse sullo sviluppo dei *learning outcomes* e dei *competence statements*, lasciando agli utilizzatori finali (singoli, scuole, enti di formazione, regioni, stati membri) il compito di adattare e contestualizzare il framework ai propri specifici bisogni. Questa scelta è coerente con la natura non prescrittiva del DigComp, che non intende fornire ricette chiuse ma uno strumento flessibile da plasmare in base ai contesti.

2.2.11. Le grandi novità del DigComp 3.0: *learning outcomes* e integrazione dell'Intelligenza Artificiale

Abbiamo visto come il DigComp 3.0 mantenga la struttura portante delle precedenti edizioni, aggiornandola nel lessico e nei descrittori. Tuttavia, il salto di qualità rispetto al passato risiede in tre innovazioni fondamentali: l'introduzione sistematica dei *learning outcomes* (risultati di apprendimento), l'integrazione trasversale dell'Intelligenza Artificiale come dimensione che attraversa tutte le competenze, e l'attenzione ai prerequisiti per il livello base di competenza digitale. A queste tre novità dedichiamo i prossimi paragrafi.

2.2.12. L'integrazione dell'Intelligenza Artificiale nel DigComp 3.0

Forse la novità più rilevante, per chi legge questo libro in un'ottica di formazione al pensiero computazionale e alla robotica, è il modo in cui il DigComp 3.0 affronta il tema dell'Intelligenza Artificiale. L'IA non viene trattata come una competenza separata o aggiuntiva, ma come una dimensione trasversale che attraversa l'intero framework.

Per comprendere la scelta compiuta dal JRC, è utile citare direttamente il documento ufficiale: *"DigComp 3.0 builds on the initial work of DigComp 2.2 to systematically include the aspects of AI competence that are relevant for individuals to develop as part of their digital competence. AI competence*

is intertwined with and builds on other elements of digital competence, as AI systems are widely diffused, and increasingly embedded within existing digital technologies" (DigComp 3.0, p. 24). In altre parole, non avrebbe senso creare un'area separata per l'IA, perché i sistemi di IA non sono tecnologie a sé stanti: sono sempre più spesso incorporati negli strumenti digitali che già utilizziamo – nei motori di ricerca, nei traduttori automatici, nei software di videoscrittura, nei social network, nei robot educativi.

La definizione di IA adottata dal DigComp 3.0 è quella dell'AI Act (Articolo 3(1)), che la descrive come: *"un sistema basato su macchine, progettato per operare con diversi livelli di autonomia, che può mostrare adattabilità dopo l'implementazione e che, per obiettivi espliciti o impliciti, deduce dall'input che riceve come generare output, come previsioni, contenuti, raccomandazioni o decisioni che possono influenzare ambienti fisici o virtuali"*.

Questa definizione è volutamente ampia e include anche l'IA generativa, che viene definita come *"un sottoinsieme dell'IA che utilizza modelli specializzati di machine learning progettati per produrre una varietà ampia e generale di output, capaci di una gamma di compiti e applicazioni, come la generazione di testo, immagini o audio"* (DigComp 3.0, p. 24).

Ma come si traduce questa integrazione nella pratica? Il DigComp 3.0 introduce una distinzione operativa, illustrata nel Box 3 (pag. 25 del documento), tra competenze *AI-explicit* e *AI-implicit*:

- *AI-explicit* [AI-E] significa che i sistemi di IA sono esplicitamente rilevanti per quella competenza. Ad esempio, un *competence statement* o un *learning outcome* che parla di "sviluppare prompt per un sistema di IA generativa" o di "riconoscere che i dati di addestramento di un modello IA influenzano l'affidabilità delle sue risposte" riceve l'etichetta AI-E.
- *AI-implicit* [AI-I] si applica a competenze per le quali i sistemi di IA sono rilevanti in modo meno diretto, ma comunque significativo, per quattro possibili ragioni:

    1. La competenza coinvolge l'uso di sistemi di IA come una tra diverse tecnologie digitali disponibili.
    2. La competenza coinvolge l'uso di una tecnologia digitale che ha funzionalità di IA incorporate.
    3. La competenza richiede la comprensione di come operano i sistemi di IA.
    4. La competenza riguarda le implicazioni personali, etiche o sociali dei sistemi di IA.

Per dare un'idea della pervasività di questa scelta, il JRC ha calcolato che, sui 362 *competence statements* del DigComp 3.0, il 14% (50) sono *AI-explicit*, il 68% (246) sono *AI-implicit*, e solo il 18% (67) non presentano rilevanza esplicita o implicita rispetto all'IA. Sui 523 *learning outcomes*, il 13% (69) sono *AI-explicit*, il 63% (330) *AI-implicit*, e il 24% (124) privi di rilevanza. In pratica, oltre l'80% delle competenze digitali ha a che fare – esplicitamente o implicitamente – con l'Intelligenza Artificiale. Questa cifra restituisce la portata del cambiamento: non si può più pensare alla competenza digitale senza includere la capacità di interagire consapevolmente, criticamente ed eticamente con i sistemi di IA.

Per chi si occupa di pensiero computazionale e robotica educativa, questa scelta ha implicazioni profonde. La programmazione non è più solo "dare istruzioni a un computer", ma sempre più spesso "interagire con sistemi che apprendono dai dati". La competenza 3.4 (Computational thinking and programming) è esplicitamente etichettata in larga parte come *AI-explicit* o *AI-implicit*, e include *learning outcomes* che vanno dalla distinzione tra ciò che è e non è un sistema di IA, alla comprensione del machine learning, alla valutazione etica dello sviluppo e dell'implementazione di programmi e sistemi di IA.

2.2.13. I *learning outcomes*: una svolta operativa

La seconda grande novità del DigComp 3.0 è l'introduzione dei *learning outcomes* (risultati di apprendimento). Come anticipato, nelle versioni precedenti il framework offriva esempi di conoscenze, abilità e attitudini a scopo puramente illustrativo. Nel DigComp 3.0, questi esempi sono stati sostituiti da un sistema organico e classificatorio di *learning outcomes*.

Ma cosa sono esattamente i *learning outcomes*? Il documento ufficiale (p. 23) li definisce come *"dichiarazioni di ciò che un individuo sa, comprende o è in grado di fare al completamento di un processo di apprendimento, definite in termini di conoscenze, abilità e attitudini"*. Si tratta quindi di descrizioni operative e verificabili di ciò che ci si aspetta che una persona sappia, sappia fare o sappia essere al termine di un percorso formativo, formale o informale.

I *learning outcomes* sono uno strumento consolidato nelle politiche educative europee. Come sottolineato dal Cedefop (il Centro europeo per lo sviluppo della formazione professionale), essi fungono da "col-

lante" (*glue*) tra il mondo dell'istruzione e quello del lavoro. Nelle parole del DigComp 3.0 (p. 23): *"Influenzano la politica, l'istruzione, la formazione, la valutazione e il mercato del lavoro in una serie di modi – per guidare lo sviluppo, l'implementazione e la revisione di curricoli o corsi; come punto di riferimento per il riconoscimento e la validazione dell'apprendimento formale, non formale e informale; per definire e informare i quadri delle qualifiche e gli standard; per supportare la valutazione sommativa e formativa; per segnalare le competenze chiave per l'occupazione nella profilazione professionale; e per contribuire all'analisi dei bisogni settoriali"*.

Nel DigComp 3.0, i *learning outcomes* sono 523 in totale. La loro distribuzione per livello di padronanza è la seguente (p. 25): il 29% (151) è al livello *Basic*, il 32% (170) al livello *Intermediate*, il 23% (119) al livello *Advanced* e il 16% (83) al livello *Highly Advanced*. Per tipologia, il 42% (217) riguarda conoscenze, il 38% (199) abilità e il 20% (107) attitudini.

L'Appendice 2 del documento (pagg. 84-113) fornisce il dettaglio completo dei *learning outcomes*, organizzati per competenza, livello di padronanza e tipologia (conoscenza, abilità, attitudine). Le Tabelle A4 e A5 (p. 85) offrono esempi utili per comprendere come sono strutturati.

La Tabella A4 mostra esempi di *learning outcome* per ciascuna delle tre tipologie:

Competenza	Livello	*Learning outcome* (traduzione)
Conoscenza: 2.6 Gestire l'identità digitale	*Basic*	Identificare le forme e gli usi comuni dell'identità digitale
Abilità: 1.1 Navigare, cercare e filtrare informazioni	*Intermediate*	Selezionare strumenti di ricerca digitali appropriati in base ai bisogni informativi
Attitudine: 3.2 Integrare e rielaborare contenuti digitali	*Advanced*	Dare priorità a pratiche trasparenti ed etiche nei compiti di integrazione e rielaborazione di contenuti digitali

Fonte: adattato da JRC, DigComp 3.0, Table A4, p. 85.

La Tabella A5 (p. 85) fornisce esempi di verbi utilizzati per i *learning outcomes* dei diversi livelli, un aspetto importante per chi deve progettare valutazioni o rubriche. Ecco una sintesi:

Livello	Descrizione sintetica	Esempi di verbi utilizzati
Basic	Ricorda e implementa compiti semplici con guida se necessaria	Conoscenze: riconoscere, identificare, distinguere tra; Abilità: usare, applicare, implementare; Attitudini: riconoscere l'importanza, riconoscere i benefici
Intermediate	Identifica e implementa compiti ben definiti e risolve problemi ben definiti in autonomia	Conoscenze: riconoscere, identificare, distinguere tra, definire, descrivere; Abilità: usare, applicare, selezionare, valutare; Attitudini: riconoscere l'importanza, dare priorità, esplorare intenzionalmente
Advanced	Valuta e applica soluzioni a una varietà di compiti complessi in autonomia, adattandosi a contesti diversi; guida altri se necessario	Conoscenze: identificare, definire, descrivere; Abilità: valutare, applicare, combinare, assistere altri, supportare altri; Attitudini: riconoscere l'importanza, tenere conto di, dare priorità, esplorare continuamente
Highly Advanced	Valuta, risolve problemi altamente complessi o specializzati, crea nuove soluzioni o adatta quelle esistenti; guida altri	Conoscenze: nessuna (catturate nelle attitudini); Abilità: valutare, sviluppare e implementare, progettare, consigliare, spiegare; Attitudini: riconoscere l'importanza, rimanere aggiornati su, promuovere e supportare

Fonte: adattato da JRC, DigComp 3.0, Table A5, p. 85.

Il Box A1 (p. 85) riassume le caratteristiche principali dei *learning outcomes* del DigComp 3.0. Tra queste, è utile evidenziare:

- I *learning outcomes* sono intenzionali, non raggiunti: descrivono ciò che ci si aspetta che una persona sappia o sappia fare dopo un apprendimento, non ciò che necessariamente già sa o sa fare.
- Sono progettati per essere sufficientemente generici da applicarsi a contesti diversi, ma sufficientemente precisi da essere chiari e non ambigui.
- La scelta dei verbi è guidata dalla Tassonomia di Bloom (rivista da Anderson & Krathwohl, 2001), che classifica i processi cognitivi dal più semplice (ricordare) al più complesso (creare, valutare).
- I *learning outcomes* sono più numerosi ai livelli inferiori (*Basic* e *Intermediate*) perché, come suggerito da esperti e stakeholder, è a questi livelli che si concentra la maggior parte degli sforzi formativi.

2.2.14. Un esempio concreto: i *learning outcomes* per la Competenza 3.4 (Computational thinking and programming)

Per comprendere concretamente come funzionano i *learning outcomes* nel DigComp 3.0, analizziamo alcuni esempi tratti dalla Competenza 3.4 – Computational thinking and programming (pagg. 103-105 dell'Appendice 2). Questa competenza è particolarmente rilevante per chi legge questo libro, poiché tocca direttamente il cuore del pensiero computazionale e della programmazione.

Ecco una selezione di *learning outcomes* per diversi livelli, con la loro etichetta AI:

ID	*Learning Outcome* (originale inglese)	Traduzione	Livello	Tipo	AI label
LO3.4.01	*Acknowledge the essential role of humans in determining how computer programs and AI systems are used*	Riconoscere il ruolo essenziale degli umani nel determinare come vengono utilizzati i programmi informatici e i sistemi di IA	*Basic*	Attitudine	*AI-Explicit*
LO3.4.04	*Recognise what AI is in general terms*	Riconoscere in termini generali cosa sia l'IA	*Basic*	Conoscenza	*AI-Explicit*
LO3.4.07	*Give basic instructions to a computer to perform simple tasks*	Dare istruzioni di base a un computer per eseguire compiti semplici	*Basic*	Abilità	*AI-Implicit*
LO3.4.18	*Recognise that machine learning is a branch of AI that enables algorithms to learn from data and make predictions*	Riconoscere che il machine learning è un ramo dell'IA che consente agli algoritmi di apprendere dai dati e fare previsioni	*Intermediate*	Conoscenza	*AI-Explicit*
LO3.4.23	*Develop basic programs with control structures*	Sviluppare programmi di base con strutture di controllo	*Intermediate*	Abilità	*AI-Implicit*
LO3.4.33	*Assess ethical and practical aspects of the development and deployment of computer programs and AI systems*	Valutare gli aspetti etici e pratici dello sviluppo e dell'implementazione di programmi informatici e sistemi di IA	*Advanced*	Abilità	*AI-Explicit*
LO3.4.35	*Apply programming tools or AI systems to complex computational thinking tasks*	Applicare strumenti di programmazione o sistemi di IA a compiti complessi di pensiero computazionale	*Advanced*	Abilità	*AI-Explicit*
LO3.4.37	*Stay informed about current developments in programming techniques and related applications of AI systems, such as robotics*	Rimanere aggiornati sugli sviluppi attuali delle tecniche di programmazione e delle applicazioni correlate dei sistemi di IA, come la robotica	*Highly Advanced*	Attitudine	*AI-Explicit*

Fonte: adattato da JRC, DigComp 3.0, Annex 2, pp. 103-105.

Questi esempi mostrano chiaramente come il DigComp 3.0 integri l'IA nella competenza di programmazione a più livelli. Già al livello *Basic*, si chiede di riconoscere il ruolo umano nel determinare l'uso dei sistemi di IA (un'attitudine fondamentale per un uso consapevole) e di sapere cosa è e cosa non è un sistema di IA. Al livello *Intermediate*, si introduce la conoscenza del machine learning. Al livello *Advanced*, si richiede di valutare gli aspetti etici dello sviluppo di programmi e sistemi di IA, e di applicare strumenti di IA a compiti complessi. Al livello *Highly Advanced*, si chiede di rimanere aggiornati sugli sviluppi della robotica e delle applicazioni dell'IA.

Notiamo anche la presenza di *learning outcome AI-Implicit*, come LO3.4.07 (dare istruzioni di base a un computer) e LO3.4.23 (sviluppare programmi con strutture di controllo): sono competenze di programmazione "classica" che non menzionano esplicitamente l'IA, ma che costituiscono il fondamento su cui si innesta la competenza specifica sull'IA. Questa distinzione è preziosa per chi progetta percorsi didattici: non si tratta di abbandonare la programmazione tradizionale, ma di integrarla con una consapevolezza crescente del ruolo dell'IA.

2.2.15. La rilevanza per la didattica del pensiero computazionale

Cosa significa tutto questo per un corso di formazione sul pensiero computazionale, il coding e la robotica educativa? Significa che il DigComp 3.0 offre oggi uno strumento di progettazione e valutazione estremamente più ricco e operativo rispetto al passato. Un docente che voglia progettare un percorso di coding non deve più chiedersi vagamente "quale competenza digitale sto sviluppando?", ma può:

1. Identificare i *learning outcomes* specifici che intende raggiungere, scegliendoli dalla batteria di 523 disponibili.
2. Classificare le attività in base alle aree del DigComp (non solo "creazione di contenuti digitali" ma anche "valutazione delle informazioni", "collaborazione", "risoluzione di problemi").
3. Valutare il livello di padronanza raggiunto (*Basic*, *Intermediate*, *Advanced*, *Highly Advanced*) utilizzando i descrittori dei *competence statements* e dei *learning outcomes*.
4. Integrare consapevolmente l'IA nelle attività di programmazione, distinguendo tra competenze *AI-explicit* e *AI-implicit*.

In particolare, la distinzione tra *AI-explicit* e *AI-implicit* offre al docente una mappa per decidere quando e come introdurre l'IA nel proprio

insegnamento del pensiero computazionale. Si può iniziare con attività *AI-implicit* (programmazione tradizionale, algoritmi, strutture di controllo) per costruire solide basi, per poi passare ad attività *AI-explicit* (introduzione del machine learning, uso critico di strumenti di IA generativa, riflessione etica sull'impatto dell'IA) man mano che gli studenti acquisiscono padronanza.

2.3. Il framework in azione: un'esperienza di progettazione tra coding e competenza digitale

Per mostrare come il quadro teorico del DigComp possa tradursi in pratica didattica e guidare la progettazione di percorsi significativi, presentiamo un'esperienza condotta a partire dal 2020 presso l'IIS Malignani di Udine, scuola fondatrice del movimento Avanguardie Educative. L'obiettivo era duplice: da un lato, integrare il coding nelle discipline di base (italiano e matematica) in un'ottica di laboratorializzazione; dall'altro, utilizzare queste attività come veicolo per sviluppare e valutare specifiche dimensioni della competenza digitale degli studenti.

Il percorso, frutto di una collaborazione tra i docenti Federica Tabacco (Informatica), Silvia Liani (Matematica) e Manuela Barbierato (Italiano), è stato strutturato con la metodologia del *Problem-Based Learning* (PBL), orientata alla costruzione di compiti autentici. La scelta di focalizzarsi sulla competenza digitale è nata naturalmente dalla vicinanza di questo ambito a quello del coding e dalla volontà di sperimentare nella direzione di un apprendimento fortemente competenziale, in cui conoscenze disciplinari, abilità tecniche e attitudini personali convergessero nella realizzazione di un artefatto significativo.

Il primo passo operativo è stato quello di "localizzare" il framework DigComp 2.1 (allora la versione più recente) al contesto scolastico specifico. Questa operazione ha comportato due interventi principali:

1. Selezione delle aree di competenza: non tutte le cinque aree del DigComp erano ugualmente pertinenti al lavoro curricolare interdisciplinare immaginato. Sono state quindi selezionate tre aree focali: *Alfabetizzazione su informazioni e dati*, *Comunicazione e collaborazione* e *Creazione di contenuti digitali*.
2. Adattamento dei livelli di padronanza: i livelli del DigComp, pensati per il cittadino nell'arco di tutta la vita, includono gradazioni trop-

po elevate (fino al professionista specializzato). È stata necessaria una riscrittura formale che mantenesse l'essenza della progressione (complessità, autonomia, dominio cognitivo) ma la rendesse aderente all'età e alle esperienze degli studenti della secondaria di secondo grado. Questa "traduzione" ha permesso di mantenere la confrontabilità con il framework originale pur utilizzando un linguaggio più vicino alla realtà scolastica.

Prima di avviare l'attività curricolare vera e propria, è stato fondamentale tracciare una baseline delle competenze digitali degli studenti. Poiché questo ambito non è normalmente oggetto di valutazione sistematica nella scuola, è stato ideato un compito di realtà pre-attività. Agli studenti è stato chiesto di ricercare online l'offerta di una SIM telefonica più conveniente per partecipare a un torneo di gaming, considerando parametri come traffico dati e costi. Questo compito, svolto individualmente e poi discusso in gruppo su una piattaforma condivisa, ha permesso di osservare e valutare le capacità degli studenti nelle aree selezionate: ricerca e valutazione delle fonti (Area 1), condivisione e negoziazione in ambiente digitale (Area 2), produzione di un breve report o video esplicativo (Area 3).

Con questa base di partenza, si è passati alla progettazione dell'attività interdisciplinare. Dopo un attento lavoro di selezione degli argomenti disciplinari più adatti a essere "laboratorializzati" attraverso il coding, il gruppo ha definito la seguente consegna per gli studenti: *«Leggi il racconto breve «Funghi in città» di Italo Calvino (da «Marcovaldo»)* e *analizza il rapporto tra tempo della storia e tempo del racconto. Crea un artefatto in Scratch che mostri visivamente questa relazione».* L'attività prevedeva che gli studenti, lavorando in gruppo, estraessero dal testo dati narrativi (il "tempo della storia"), li elaborassero matematicamente per definire una funzione di rappresentazione, e infine implementassero in Scratch un'animazione che visualizzasse tale relazione. Oltre all'artefatto digitale, era richiesta la produzione di report intermedi e una presentazione finale.

L'osservazione del processo e la valutazione dei prodotti hanno rivelato esiti significativi e alcune criticità illuminanti. Da un lato, si è registrato un miglioramento tangibile nella competenza di creazione di artefatti digitali (Area 3), in particolare nella scrittura di algoritmi in Scratch, che ha generato in molti studenti una forte motivazione intrinseca. Dall'altro, è emersa una difficoltà degli studenti nel riconoscere come "competenza

digitale" le fasi di analisi dei dati testuali e di comunicazione collaborativa online. Per loro, la parte "digitale" coincideva principalmente con l'atto della programmazione, mentre la ricerca, la valutazione e la discussione in piattaforma erano percepite come attività accessorie o scontate. Questo ha richiesto un lavoro esplicito di riflessione metacognitiva da parte dei docenti, per aiutare gli studenti a ricondurre tutte le fasi del lavoro alle aree del DigComp e a comprenderne la rilevanza.

Un'ulteriore sfida è emersa a livello metodologico: gli studenti, abituati a un modello scolastico tradizionale e fortemente disciplinare, hanno inizialmente percepito come "strano" o "controcorrente" un approccio interdisciplinare basato sul PBL. Questo evidenzia quanto un cambiamento verso una didattica per competenze richieda non solo una progettazione accurata, ma anche un percorso di accompagnamento degli studenti verso nuovi modelli di apprendimento.

Da questa esperienza possono trarsi alcune indicazioni operative per i docenti che intendono lavorare con la competenza digitale:

1. Familiarizzare con la struttura del DigComp, comprendendone la logica delle aree e dei livelli di valutazione.
2. Localizzare il framework al proprio contesto, selezionando le aree più pertinenti e adattando i descrittori all'età degli studenti, senza tradirne l'essenza.
3. Valutare una baseline di partenza delle competenze digitali degli studenti, anche con compiti semplici, per progettare attività adeguate.
4. Lavorare in ottica interdisciplinare, poiché è difficile che una singola disciplina copra tutte le aree del DigComp. La collaborazione tra colleghi è un moltiplicatore di opportunità.
5. Rendere esplicito il legame tra le attività pratiche (come il coding) e le diverse dimensioni della competenza digitale, favorendo la consapevolezza metacognitiva degli studenti.
6. Prepararsi a gestire il cambiamento metodologico, sostenendo gli studenti nell'adattarsi a modalità di lavoro più aperte, collaborative e centrate sulla risoluzione di problemi.

Questa esperienza dimostra che il DigComp non è un mero adempimento, ma uno strumento di progettazione e riflessione potentissimo. Permette di andare oltre la dimensione tecnica del coding per costruire percorsi in cui la tecnologia diventa il mezzo per sviluppare, in modo integrato, competenze cognitive, digitali e di cittadinanza.

Riferimenti bibliografici

Carretero, S., Vuorikari, R., & Punie, Y. (2017). *DigComp 2.1: The Digital Competence Framework for Citizens with eight proficiency levels and examples of use*. Publications Office of the European Union.

Commissione Europea. (2006). *Raccomandazione del Parlamento Europeo e del Consiglio del 18 dicembre 2006 relativa a competenze chiave per l'apprendimento permanente*. Gazzetta ufficiale dell'Unione Europea L 394/10.

Commissione Europea, Direzione generale dell'Istruzione, gioventù, sport e cultura. (2021). *Piano d'azione per l'istruzione digitale 2021-2027*.

Cosgrove, J., & Cachia, R. (2025). *DigComp 3.0: European Digital Competence Framework – Fifth Edition*. Publications Office of the European Union. JRC144121.

Libow Martinez, S., & Stager, G. (2020). *Inventando si impara: Apprendere e sperimentare con strumenti e materiali* (L. Guasti, a cura di). Carocci. (Ed. orig. 2013).

Nulli, G., Liani, S., & Tabacco, F. (2024). Developing Digital Competence into Formal Education Involving Computational Thinking. In *EDULEARN24 Proceedings* (pp. 5647-5654). IATED.

Parlamento Europeo e Consiglio dell'Unione Europea. (2018). *Raccomandazione del Consiglio relativa alle competenze chiave per l'apprendimento permanente*. Gazzetta ufficiale dell'Unione Europea C 189/1.

Vuorikari, R., Punie, Y., Carretero, S., & Van den Brande, G. (2016). *DigComp 2.0: The Digital Competence Framework for Citizens. Update Phase 1: The Conceptual Reference Model*. JRC Science for Policy Report. Publications Office of the European Union.

Vuorikari, R., Kluzer, S., & Punie, Y. (2022). *DigComp 2.2: The Digital Competence Framework for Citizens – With new examples of knowledge, skills and attitudes*. Publications Office of the European Union. JRC128415.

Risorse online e documenti istituzionali

AGID – Agenzia per l'Italia Digitale. (s.d.). Traduzione ufficiale in italiano del DigComp 2.1. https://www.agid.gov.it/sites/default/files/repository_files/digcomp2-1_ita.pdf

AGID – Agenzia per l'Italia Digitale. (2024). Traduzione ufficiale in italiano del DigComp 2.2. https://www.agid.gov.it/sites/agid/files/2024-05/digcomp_2.2_italiano.pdf

Commissione Europea, JRC. (s.d.). *DigComp into Action – Get inspired, make it happen. Guida operativa*. https://publications.jrc.ec.europa.eu/repository/handle/JRC110624

DigCompEdu. (s.d.). Framework per i docenti (traduzione italiana a cura di CNR-ITD). https://digcompedu.cnr.it/DigCompEdu_ITA_FINAL_CNR-ITD.pdf

Sitografia istituzionale europea sulla competenza digitale

Programmi e strategie quadro

Digital Decade Policy Programme: https://commission.europa.eu/strategy-and-policy/priorities-2019-2024/europe-fit-digital-age/europes-digital-decade-digital-targets-2030_it

Union of Skills (Comunicazione marzo 2025): https://ec.europa.eu/commission/presscorner/detail/it/ip_25_864

Action Plan on Basic Skills: https://education.ec.europa.eu/sites/default/files/2025-03/Graphic%20version%20Action%20Plan%20on%20Basic%20Skills.pdf

Digital Education Action Plan (DEAP) 2021-2027: https://education.ec.europa.eu/it/focus-topics/digital-education/action-plan

Better Internet for Kids (BIK+): https://digital-strategy.ec.europa.eu/en/policies/better-internet-kids

AI Continent Action Plan: https://digital-strategy.ec.europa.eu/en/policies/ai-continent-action-plan

Apply AI Strategy: https://digital-strategy.ec.europa.eu/en/policies/apply-ai-strategy

Legislazione digitale europea

GDPR (General Data Protection Regulation): https://gdpr-info.eu/

Digital Services Act: https://commission.europa.eu/strategy-and-policy/priorities-2019-2024/europe-fit-digital-age/digital-services-act_it

AI Act: https://eur-lex.europa.eu/legal-content/IT/TXT/?uri=CELEX%3A32024R1689

Cyber Resilience Act: https://digital-strategy.ec.europa.eu/en/policies/cyber-resilience-act

Digital Markets Act: https://commission.europa.eu/strategy-and-policy/priorities-2019-2024/europe-fit-digital-age/digital-markets-act-ensuring-fair-and-open-digital-markets_it

European Digital Identity Regulation: https://digital-strategy.ec.europa.eu/en/policies/euid

Documenti del JRC e framework correlati

DigComp 3.0 (edizione integrale): https://data.europa.eu/doi/10.2760/0001149

DigComp 2.2 (edizione integrale): https://publications.jrc.ec.europa.eu/repository/handle/JRC128415

DigComp 2.1 (edizione integrale): https://publications.jrc.ec.europa.eu/repository/handle/JRC106281

Dichiarazione Europea sui Diritti e Principi Digitali per il Decennio Digitale: https://digital-strategy.ec.europa.eu/en/fact-pages/digital-rights-and-principles

3.
DALLE DEFINIZIONI ALLA PRATICA: METODOLOGIE DIDATTICHE TRA PENSIERO COMPUTAZIONALE E COMPETENZA DIGITALE

Giovanni Nulli

I capitoli precedenti hanno presentato due concetti centrali, il pensiero computazionale e la competenza digitale, ciascuno con il proprio bagaglio teorico, definizioni e framework di riferimento. Per il docente, tuttavia, la sfida non si conclude con la comprensione di questi quadri concettuali, ma inizia nel momento in cui deve tradurli in azione didattica quotidiana. Questo capitolo si propone di costruire un filo conduttore tra teoria e pratica, mostrando come i contenuti presentati possano essere efficacemente "dipanati" e ricondotti al lavoro concreto in classe attraverso il ricorso a precise scelte metodologiche e alla focalizzazione su fattori psicologici dell'apprendimento. L'obiettivo è fornire una mappa per orientarsi al di là della vaghezza di alcuni termini e per progettare interventi che vadano oltre l'effetto tecnologico superficiale, puntando a sviluppare negli studenti quelle disposizioni mentali che stanno alla base di un apprendimento duraturo e significativo, con particolare attenzione alle discipline dell'area scientifica dove spesso emergono criticità motivazionali e cognitive (Mori, 2023).

3.1. Competenza digitale: un framework operativo per la progettazione

Il DigComp offre al docente un vantaggio immediato: è un framework strutturato e articolato che fornisce un linguaggio comune e descrittori operativi. Per un utilizzo efficace nel coding e nella robotica, è utile concentrarsi su tre aree specifiche che si prestano naturalmente a essere integrate in attività didattiche attive:

- Comunicazione e collaborazione: L'esperienza della Didattica a Distanza ha mostrato come questa competenza possa essere esercitata. In attività di coding e robotica, gli studenti sono chiamati a collabo-

rare in ambienti digitali, condividere soluzioni (ad esempio, tramite repository di codice come quelli integrati in Scratch o GitHub Classroom) e negoziare significati, trasformando la tecnologia nel mezzo, non nel fine, di un'interazione sociale finalizzata. La competenza digitale qui si misura nella capacità di co-costruire conoscenza in uno spazio condiviso, rispettando netiquette e ruoli.

- Creazione di contenuti digitali: Questa area è il cuore operativo. Implica una metodologia in cui lo studente è produttore attivo di un artefatto digitale (un programma, un comportamento robotico). Tale approccio si colloca saldamente nell'ambito della didattica attiva e del costruzionismo papertiano, dove l'apprendimento passa attraverso il "fare" significativo. La creazione non è fine a sé stessa: produce un oggetto esaminabile, testabile e migliorabile, che rende visibile il pensiero dello studente.
- Problem solving (in senso tecnico): Nel DigComp, questa competenza ha un'accezione pratica: risolvere problemi tecnici con i dispositivi (es. configurare una connessione, risolvere un conflitto software, ottimizzare le prestazioni). È la competenza del "sapersela cavare", fondamentale per creare un senso di autonomia e padronanza dello strumento, prerequisito per qualsiasi progetto creativo. Senza questa capacità di base, ogni attività digitale rischia di arenarsi al primo intoppo tecnico, generando frustrazione.

Il DigComp, quindi, non è solo un elenco di cose da sapere, ma una griglia di progettazione che indica *cosa* gli studenti dovrebbero *saper fare*, guidando il docente nel *come* strutturarlo. Fornisce un ancoraggio solido per evitare che le attività digitali si riducano a mere esercitazioni tecniche, inserendole invece in un quadro più ampio di cittadinanza digitale, dove l'etica della condivisione (area 2), la creatività responsabile (area 3) e la resilienza tecnologica (area 5) sono interconnesse.

3.2. Pensiero computazionale: oltre la vaghezza, verso la metodologia

La situazione del pensiero computazionale è, come visto nel Capitolo 1, più complessa e meno coerente. Le tre macro-aree di definizione identificate dal JRC (2022) – processi informatici, competenze astratte gene-

rali, capacità di problem solving – possono però essere rilette in chiave metodologica per superarne l'ambiguità (JRC, 2022, p. 35).

- Processi mutuati dall'informatica: Concetti come il pensiero algoritmico e logico sono fortemente collegati alle discipline dell'informatica e della matematica. Il loro trasferimento non è automatico; richiede una mediazione didattica esplicita da parte del docente, che deve progettare attività ponte per aiutare gli studenti a vedere le connessioni tra ambiti spesso percepiti come separati. Senza questa progettazione intenzionale, il rischio è che il pensiero computazionale rimanga confinato all'ora di informatica. Ad esempio, il concetto di "variabile" in programmazione può essere collegato al concetto di "incognita" in algebra, ma questo collegamento va reso visibile e discusso.
- Competenze astratte generali (es. astrazione, scomposizione): L'idea che il pensiero computazionale sia un passepartout per competenze di alto livello trasferibili ovunque non è suffragata da evidenze scientifiche robuste, come discusso nel Capitolo 1 (Scherer et al., 2019; Salomon, 1984). Piuttosto che affidarsi a questa promessa generica, è più produttivo collegare l'esercizio di questi schemi cognitivi a metodologie didattiche specifiche che ne favoriscano consapevolmente l'emergere e la pratica in contesti definiti. Come vedremo, il *Problem-Based Learning* e il *Project-Based Learning* sono cornici metodologiche ideali per questo scopo (Lotti, 2022), perché strutturano l'esperienza di apprendimento proprio attorno a processi di scomposizione, astrazione e ricerca di soluzioni.
- Sviluppo di capacità di risolvere problemi: Questo rimando costante al problem solving merita un'analisi critica, poiché rappresenta il nodo centrale della traduzione in pratica. È qui che la riflessione deve spostarsi dal "*cosa*" (una competenza vaga) al "*come*" (una strategia didattica) e, infine, al *perché* (quali meccanismi psicologici si intendono attivare).

3.3. Il nodo del problem solving: tra strategia didattica e competenza psicologica

Parlare di problem solving a scuola significa distinguere con chiarezza tra due piani molto diversi, la cui confusione genera spesso frustrazione progettuale:

1. Il problem solving come strategia didattica: È il *Problem-Based Learning* (PBL), una metodologia strutturata in cui il docente progetta e propone un problema agli studenti come punto di partenza per l'indagine e l'acquisizione di nuove conoscenze (Lotti, 2022). L'efficacia di questa strategia dipende da scelte progettuali cruciali: la qualità del *problem posing* (come si formula il problema?), la sua pertinenza rispetto agli obiettivi di apprendimento e il ruolo del docente (facilitatore, risorsa, co-ricercatore). Un problema autentico, come "Come possiamo programmare un robot per distribuire il cibo agli animali del nostro biotopo scolastico in modo equo?", è aperto, sfidante e non contiene in sé la soluzione, a differenza di un semplice esercizio di calcolo ("5-2=?") o di un problema già formalizzato ("La mamma compra 5 mele e ne perde 2, quante ne ha?"). La formulazione della consegna, o *prompt*, è quindi un atto didattico importante che lavora sul linguaggio, la sua comprensione (quindi sulla logica soggiacente), e sul concetto di zona di sviluppo prossimale di Vygotskij. Vygotskij (1978) definisce la Zona di Sviluppo Prossimale (ZPD) come la distanza tra il livello di sviluppo attuale di un discente (ciò che sa fare da solo) e il livello di sviluppo potenziale (ciò che può fare con la guida di un pari più esperto o di un adulto). Un buon problema di PBL dovrebbe collocarsi proprio all'interno della ZPD degli studenti: non così facile da essere banale, non così difficile da essere paralizzante, ma sufficientemente sfidante da richiedere collaborazione, ricerca e supporto per essere risolto. Il coding, con la sua natura incrementale e "debuggabile", offre un terreno ideale per esplorare questa zona in modo concreto.
2. La competenza di risolvere problemi: Questo è un costrutto psicologico di alto livello, difficilmente insegnabile in astratto e con scarso transfer tra domini disciplinari lontani (cfr. Capitolo 1). Ciononostante, è possibile agire su schemi cognitivi e attitudinali che la sostengono, che possono essere esercitati e rafforzati attraverso la pratica metodologica. È qui che il discorso si sposta dal *cosa* insegnare al *come* favorire disposizioni mentali durature, che sono il vero valore aggiunto di un approccio didattico ben progettato.

3.4. Il cuore della questione: autoefficacia, metacognizione e motivazione intrinseca

La proposta centrale di questo capitolo è che il valore aggiunto del coding e della robotica, al di là delle specifiche competenze tecniche, risieda nella loro capacità di creare contesti privilegiati per lo sviluppo di fattori psicologici fondamentali per l'apprendimento. Questi fattori costituiscono il ponte tra la metodologia (il PBL) e lo sviluppo di una mentalità resiliente e curiosa nello studente.

- Autoefficacia: È la convinzione di essere capaci di affrontare e portare a termine un compito, definita come la "capacità di organizzare ed eseguire il corso delle azioni necessarie per produrre un dato esito" (Bandura, 1997, cit. in Mori, 2023, p. 58). Comprendere e favorire l'autoefficacia è cruciale, specialmente nelle discipline scientifiche dove spesso emergono ansie e convinzioni di inadeguatezza che possono portare a una vera e propria "impotenza appresa" (Mori, 2023, p. 58) – la sensazione che, nonostante gli sforzi, il successo sia fuori dal proprio controllo. In attività di coding e robotica, lo studente riceve un feedback immediato, tangibile e non giudicante dal suo artefatto (il programma funziona o no? Il robot si muove come previsto?). Superare ostacoli, correggere errori (debugging) e vedere realizzato il proprio progetto costruisce un solido senso di competenza e perseveranza, contrastando direttamente i meccanismi dell'impotenza appresa. L'autoefficacia è il motore che permette allo studente di affrontare problemi complessi senza arrendersi alla prima difficoltà. Mori sottolinea come lavorare su compiti sfidanti ma raggiungibili, suddivisi in sotto-obiettivi chiari (una pratica naturale nella scomposizione algoritmica), sia una strategia chiave per costruire autoefficacia, specialmente in coloro che partono da una bassa percezione delle proprie capacità in ambito scientifico.
- Metacognizione: Il processo di programmazione obbliga a un continuo monitoraggio del proprio pensiero ("Perché non funziona? Quale passaggio ho sbagliato? Cosa devo cambiare?"). Questo esercizio di riflessione sul processo è un potente allenamento delle componenti metacognitive, in particolare dei "processi cognitivi di controllo", ossia "le operazioni che un individuo mette in atto nell'esecuzione dei propri processi cognitivi", come la compren-

sione del compito, la valutazione della difficoltà e l'esame delle strategie possibili (Cornoldi & De Beni, 2020, cit. in Mori, 2023, p. 61). È un'abilità trasferibile che porta lo studente a diventare consapevole delle proprie strategie di pensiero. Come nota Mori, "argomentare in classe quali strategie funzionano per gli studenti e perché funzionano permette loro di acquisire una maggiore consapevolezza sul proprio modo di procedere" (Mori, 2023, p. 62), un passaggio fondamentale sia per il problem solving matematico che scientifico. Il debugging non è solo una correzione tecnica, ma un atto metacognitivo per eccellenza: richiede di formulare ipotesi sull'errore, testarle e rivedere il proprio modello mentale del programma.

– Motivazione Intrinseca: Il lavoro su un progetto concreto, specialmente se scelto o sentito come proprio (evoluzione dal PBL al *Project-Based Learning* - PjBL), può attivare curiosità e coinvolgimento autentico. Come sottolineano Martinez e Stager (2013), nell'approccio maker e costruzionista, la passione per ciò che si crea diventa il principale motore dell'apprendimento. L'artefatto digitale o robotico cessa di essere un compito e diventa un oggetto di desiderio cognitivo, spingendo lo studente a imparare per realizzarlo. La motivazione intrinseca, a differenza di quella estrinseca (voto, premio), è associata a un apprendimento più profondo, creativo e persistente. Il coding e la robotica, per la loro natura creativa e "potente" (permettono di dare vita alle idee), sono catalizzatori naturali di questo tipo di motivazione.

BOX DI APPROFONDIMENTO: Il circolo virtuoso della didattica con il coding.

Strategia didattica (IL COME)	Esempio con Coding/Robotica	Fattore psicologico attivato (IL PERCHÉ)	Esito apprenditivo a lungo termine
Problem-Based Learning (PBL)	"Il sensore di pioggia del nostro orto scolastico non funziona. Progetta un sistema alternativo con un microcontrollore (es. Arduino) che avvisi quando innaffiare."	Autoefficacia: Affrontare un problema reale, scomporlo, testare soluzioni e vedere un sistema funzionante costruisce la convinzione di "potercela fare". Metacognizione: Riflettere sul perché una soluzione non ha funzionato, confrontare strategie di gruppo.	Sviluppo di una mentalità di crescita e resilienza di fronte a problemi complessi, anche in altri ambiti.

Strategia didattica (IL COME)	Esempio con Coding/Robotica	Fattore psicologico attivato (IL PERCHÉ)	Esito apprenditivo a lungo termine
Project-Based Learning (PjBL)	"Creiamo un videogioco in Scratch che spieghi il ciclo dell'acqua ai bambini della primaria."	Motivazione Intrinseca: Il progetto nasce da un interesse o da una sfida creativa sentita come propria. L'apprendimento è finalizzato alla creazione di un artefatto significativo.	Coinvolgimento profondo e sviluppo della creatività applicata. L'apprendimento è percepito come utile e personale.
Debugging Sistematico	Analisi di un programma Scratch che non produce l'animazione prevista, ipotesi sulle cause, test incrementali.	Metacognizione & Autoefficacia: Monitoraggio continuo del proprio pensiero e delle proprie azioni. Successo nel risolvere l'errore rafforza il senso di controllo e competenza.	Acquisizione di un metodo di indagine sistematico e della capacità di gestire la frustrazione trasformandola in un problema da risolvere.
Collaborazione e Peer Review	Lavoro in gruppo su un progetto robotico; sessioni di condivisione del codice e suggerimenti tra pari.	Autoefficacia (sociale): Vedere che altri affrontano problemi simili e imparare da loro. Metacognizione: Spiegare il proprio ragionamento ad altri lo chiarisce a se stessi.	Sviluppo di competenze sociali e comunicative e di una comunità di apprendimento che normalizza l'errore come parte del processo.

La sintesi: Coding e robotica non sono "magici", ma sono strumenti particolarmente abilitanti. Creano naturalmente ambienti di didattica attiva e costruzionista (Papert, 1984, 1996) dove il docente, con una progettazione attenta, può coltivare con più facilità questi "terreni fertili" dell'apprendimento: l'autoefficacia, la riflessione sul proprio operato e la motivazione che nasce dal fare. Questo è l'eredità più profonda dell'approccio di Seymour Papert: vedere nel computer e nel robot non degli insegnanti, ma dei materiali per pensare (*objects-to-think-with*), che mettono in moto processi di crescita personale e intellettuale (Lodi & Martini, 2021).

3.5. Conclusioni: una mappa per il docente-progettista

In conclusione, per orientarsi nella pratica e progettare attività di coding e robotica che vadano oltre il superficiale e affrontino anche le criticità emotive e cognitive legate alle discipline scientifiche:

– Il pensiero computazionale, come termine onnicomprensivo, rischia di essere poco utile come guida operativa. È più fruttuoso decostruir-

lo, agganciandone gli elementi (logica, algoritmi) a discipline specifiche o, meglio ancora, alle metodologie (PBL, PjBL) che ne permettono l'esercizio in contesti significativi.
- Il framework della competenza digitale (DigComp) è uno strumento di progettazione più solido. Indirizza verso attività di creazione, collaborazione e risoluzione di problemi tecnici, allineandosi naturalmente con la didattica attiva e fornendo un lessico condiviso per la valutazione.
- Il vero obiettivo trasformativo, reso possibile da una progettazione didattica competente che utilizzi coding e robotica, non è tanto insegnare a "pensare come un informatico" in astratto, ma creare le condizioni per sviluppare autoefficacia, metacognizione e motivazione intrinseca negli studenti. Come evidenzia Mori, lavorare su questi aspetti significa contrastare le "credenze e misconcezioni da parte degli studenti, capaci di avere un impatto negativo sulla motivazione ad apprendere" (Mori, 2023, p. 63), specialmente nell'area scientifica. Sono queste le "competenze di alto livello" che, una volta radicate, possono davvero essere spese in contesti diversi. Non si trasferisce il problem solving, si trasferisce un atteggiamento di fiducia nelle proprie capacità di apprendere e di affrontare sfide.

Il docente, quindi, non è un semplice trasmettitore di concetti computazionali o digitali, ma un progettista di contesti di apprendimento. I moduli successivi, dedicati alla progettazione didattica (Capitolo 6), forniranno gli strumenti operativi per tradurre questa mappa concettuale in percorsi concreti, in cui la tecnologia sia al servizio della crescita cognitiva ed emotiva dello studente, andando ben oltre il fugace "effetto wow" e costruendo le basi per un atteggiamento positivo e proattivo verso l'apprendimento e la risoluzione di problemi, a scuola e nella vita.

Riferimenti bibliografici

Bandura, A. (1997). *Self-efficacy: The exercise of control.* W.H. Freeman. (Fonte originale del costrutto di autoefficacia, citato in Mori, 2023).

Cornoldi, C., & De Beni, R. (2020). *Imparare a studiare: strategie, stili cognitivi, metacognizione e atteggiamenti nello studio.* Erickson. (Fonte per il costrutto di metacognizione, citato in Mori, 2023).

JRC - Joint Research Centre. (2022). *Reviewing Computational Thinking in Compulsory Education.* (A. Inamorato Dos Santos, R. Cachia, N. Giannoutsou, Y. Punie, A cura di). Publications Office of the European Union. (Fondamentale per la tassonomia delle tre aree del PCT e per l'analisi critica delle politiche).

Lodi, M., & Martini, S. (2021). *Computational Thinking, between Papert and Wing.* Science and Education, 30(4), 883–908. (Analisi filosofica essenziale per comprendere le due anime del PCT e la visione papertiana degli "oggetti per pensare").

Lotti, A. (2022). *Problem Based Learning.* FrancoAngeli. (Testo di riferimento per la metodologia PBL, con esempi e indicazioni operative per la progettazione).

Martinez, S. L., & Stager, G. (2013). *Invent to Learn: Making, Tinkering, and Engineering in the Classroom.* Constructing Modern Knowledge Press. (Trad. it.: Libow Martinez, S., & Stager, G. (2020). *Inventando si impara: Apprendere e sperimentare con strumenti e materiali* (L. Guasti, A cura di). Carocci). (Manifesto del movimento maker e costruzionista, centrale per comprendere il legame tra creazione, motivazione intrinseca e apprendimento).

Mori, S. (2023). *Sviluppare le competenze nell'area scientifica: il ruolo delle credenze, dei feedback e delle strategie metacognitive nel processo di apprendimento.* Rivista Lasalliana, 90(1), 53–63. Testo chiave del capitolo. Fornisce una sintesi aggiornata e basata su evidenze del ruolo cruciale di autoefficacia, metacognizione e motivazione nell'apprendimento scientifico, offrendo un ponte solido tra psicologia dell'educazione e didattica delle discipline STEM.

Papert, S. (1984). *Mindstorms. Bambini, computer e creatività* (ed. or. 1980). Emme.

Papert, S. (1996). An Exploration in the Space of Mathematics Educations. *International Journal of Computers for Mathematical Learning*, 1(1), 95–123.

Vygotskij, L. S. (1978). *Mind in society: The development of higher psychological processes.* Harvard University Press. (Fonte originale del concetto di Zona di Sviluppo Prossimale, fondamentale per la progettazione di attività sfidanti ma accessibili).

4.
EVOLUZIONE DI UN'IDEA: DALLA ROBOTICA PEDAGOGICA ALL'INTELLIGENZA ARTIFICIALE

4.1. Le radici pedagogiche e l'evoluzione aperta della robotica educativa

Giovanni Nulli

Ripercorrere la storia del coding e della robotica educativa non è un esercizio di archeologia tecnologica, ma un viaggio nella storia delle idee pedagogiche. È una storia "squisitamente pedagogica", che ci permette di comprendere come il senso del nostro agire didattico con queste tecnologie sia il risultato di un lungo processo costruito nel tempo, non di un determinismo tecnologico lineare (Nulli, 2019). Guardando a ritroso, il percorso appare inevitabile, ma in realtà è stato plasmato da scelte consapevoli, visioni politiche e, soprattutto, da un dialogo costante con le teorie dell'apprendimento. Ricostruire questa cronologia ci aiuta a inserire in un quadro più ampio e significativo i concetti di pensiero computazionale e competenza digitale, mostrando come le riflessioni metodologiche siano spesso anticipatrici degli sviluppi tecnologici e come una visione pedagogica "senza tempo" possa guidare un uso critico e consapevole degli strumenti più avanzati.

Seymour Papert e le origini costruzioniste

Agli albori di questa storia c'è la figura di Seymour Papert. Matematico di formazione, dopo il dottorato si trasferì a Ginevra per studiare con Jean Piaget. Partendo dalle tesi costruttiviste di Piaget – secondo cui la conoscenza è attivamente costruita dal discente in un contesto sociale – Papert elaborò la sua teoria del costruzionismo. Per Papert, la conoscenza si costruisce in modo ancor più efficace attraverso la creazione di artefatti cognitivi, oggetti tangibili o digitali complessi. "Si apprende ciò che si pratica", affermava, evidenziando l'importanza dell'azione. Osservando come l'apprendimento del linguaggio fosse "naturale" in un contesto che lo parla,

mentre quello della matematica apparisse astratto e difficile, Papert intuì che il computer poteva diventare un "materiale per pensare" (object-to-think-with), creando un ambiente in cui "parlare matematica" e apprendere attraverso la passione e la motivazione intrinseca (Papert, 1984; Teaching Children to be Mathematicians vs. Teaching About Mathematics).

Costruttivismo (Piaget): Teoria secondo cui la conoscenza non è trasmessa passivamente, ma attivamente costruita nella mente del discente attraverso l'interazione con l'ambiente e la risoluzione di conflitti cognitivi. L'accento è sul processo individuale e interno di costruzione di schemi mentali.	Costruzionismo (Papert): Estensione del costruttivismo. Sostiene che l'apprendimento avviene in modo ancor più efficace quando la costruzione di conoscenza si realizza attraverso la creazione di un artefatto concreto o digitale (un programma, un robot, una storia) che può essere condiviso, discusso e rifinito. L'accento è sull'azione esterna e sociale del "fare per apprendere".

Trasferitosi al MIT di Boston, Papert co-fondò il Laboratorio di Intelligenza Artificiale e diede vita al linguaggio di programmazione LOGO, il primo e forse unico linguaggio creato esplicitamente con l'obiettivo di far apprendere. Data la scarsità di personal computer con interfacce grafiche negli anni '70, fu più semplice realizzare il suo ideale di apprendimento attivo attraverso un automa fisico: nacque così la "tartaruga" robotica, un oggetto concreto che i bambini potevano programmare per muoversi nello spazio. La pubblicazione del libro Mindstorms nei primi anni '80 segnò la formalizzazione della teoria costruzionista e coincise con l'avento dei personal computer, rendendo il LOGO e il suo approccio accessibili a un pubblico più vasto (Breve storia del personal computer). Per Papert, il computer non era solo una macchina d'ufficio, ma lo strumento qualitativamente diverso che era mancato a pedagogisti come John Dewey per realizzare appieno la didattica attiva: il motore a reazione che avrebbe permesso all'"aereo" delle loro idee pedagogiche di decollare.

L'intersezione cruciale: etica, licenze e l'open source

Parallelamente allo sviluppo pedagogico, un'altra linea di evoluzione, di natura giuridica ed etica, si stava rivelando fondamentale. Negli anni

'80, il modello tradizionale di *copyright* sul software iniziò a mostrare i suoi limiti di fronte alla crescita di comunità di sviluppatori che vedevano nella condivisione del codice un moltiplicatore di innovazione. Nacque così il movimento del software libero e open source, con licenze come la GNU GPL.

Il *copyleft* è un metodo giuridico per rendere un'opera (software, documento, creatività) libera, imponendo che tutte le sue versioni modificate e derivate rimangano altrettanto libere. A differenza del semplice "permesso d'uso", il *copyleft* (letteralmente "sinistri di copia") usa le norme sul *copyright* per garantire la libertà, non per limitarla. La licenza GNU GPL (General Public License) è l'esempio più famoso: chi distribuisce software modificato sotto GPL deve rendere disponibile anche il suo codice sorgente.

Queste licenze garantivano la libertà di usare, studiare, modificare e ridistribuire il software, a patto che le modifiche fossero rilasciate sotto la stessa licenza, preservando così i benefici per la comunità (Nulli, 2019). Questa filosofia, nata per il codice, trovò un terreno di espansione ideale con la nascita del World Wide Web (1991) e si estese successivamente ai contenuti digitali con le licenze Creative Commons (all'inizio degli anni 2000).

Le Creative Commons (CC) sono licenze di diritto d'autore standardizzate e gratuite che permettono agli autori di concedere al pubblico alcuni diritti d'uso sulle proprie opere (testi, immagini, video, musica), mantenendo altri diritti riservati. Non sostituiscono il *copyright*, ma lo rendono più flessibile. Le licenze sono combinate da quattro condizioni (Attribuzione, NonCommercial, NoDerivatives, ShareAlike), permettendo scelte come "uso libero purché non commerciale e citando l'autore".

Questa cultura della condivisione e della collaborazione decentralizzata rappresentò una "singolarità" – un punto di convergenza di linee tecnologiche, legali e sociali – che avrebbe permesso lo sviluppo successivo. Si preparava il terreno per una transizione epocale: dal software all'hardware.

Dai laboratori al mercato e alle comunità: la democratizzazione

La visione di Papert cominciò a uscire dai laboratori accademici e a permeare il mondo commerciale e comunitario. Alla fine degli anni '80, la collaborazione tra Papert e la LEGO portò alla nascita di LEGO Mindstorms, il primo kit robotico programmabile commerciale, che portava il nome del libro manifesto di Papert. Negli anni '90, nacquero competizioni come la RoboCup (in ambito universitario) e la FIRST LEGO League, che univano aspetti ludici, di ricerca e forti valori etici (i Core Values di FIRST: scoperta, innovazione, inclusione, lavoro di squadra). Queste realtà, pur diverse, condividevano una base non commerciale e una forte spinta motivazionale intrinseca, radicata nel piacere della costruzione e della risoluzione di problemi (Nulli, 2019).

La vera democratizzazione avvenne però a metà degli anni 2000 con l'avvento dell'open hardware. Progetti come Arduino (2005) e RepRap (2005) applicarono i principi open source alla progettazione di schede elettroniche e stampanti 3D.

Arduino è una piattaforma di prototipazione elettronica open-source basata su hardware e software facili da usare. Al suo centro c'è una scheda che integra un microcontrollore – un piccolo computer programmabile su un singolo chip, capace di leggere input (da sensori, pulsanti), elaborarli e controllare output (motori, luci, display). La sua semplicità, basso costo e la vasta comunità hanno reso Arduino uno strumento fondamentale per la didattica della robotica, del making e dell'Internet of Things (IoT).	Una stampante 3D è una macchina che realizza oggetti solidi tridimensionali a partire da un modello digitale, aggiungendo materiale strato su strato (tecnologia *additiva*). A differenza delle macchine a sottrazione (che scolpiscono da un blocco), costruisce progressivamente. In ambito educativo, stampanti 3D open source come la RepRap (autoriproducibile) permettono di realizzare prototipi, componenti meccanici e modelli didattici, avvicinando gli studenti ai concetti di progettazione digitale e manifattura personale.

Rilasciando pubblicamente gli schemi costruttivi, permisero a chiunque, in qualsiasi parte del mondo, di autocostruirsi o modificare questi dispositivi a costi accessibili. Questo diede un impulso decisivo al movimento maker, che trova nelle Maker Faire (la prima nel 2006) i suoi mo-

menti di incontro e condivisione. Come sottolinea Mitch Resnick, erede di Papert al MIT e creatore di Scratch, il cuore di questo movimento non è il business, ma l'apprendimento attraverso il fare creativo e motivato (Resnick, 2018; Lifelong Kindergarten).

La sintesi in un linguaggio per tutti: Scratch e l'istituzionalizzazione

La convergenza di queste linee – la pedagogia costruzionista, la cultura open source, la democratizzazione dell'hardware e la spinta comunitaria – trovò una sintesi potentissima nel 2007 con il lancio di Scratch da parte del gruppo di Resnick. Scratch era (ed è) più di un linguaggio di programmazione a blocchi: era un progetto pedagogico open source e gratuito, sostenuto da una vivace comunità online per la condivisione di progetti. Incorporava in un unico strumento accessibile l'idea papertiana dell'imparare facendo, la logica del costruzionismo, l'etica della condivisione e la semplicità d'uso necessaria per raggiungere un vasto pubblico.

È in questo contesto maturo, dove nicchie di appassionati, educatori e maker avevano ormai costruito un ecosistema vitale, che il dibattito istituzionale iniziò a recepire e formalizzare queste pratiche. La proposta di Jeannette Wing sul "pensiero computazionale" (2006), l'introduzione dell'informatica nel curricolo inglese (2012), la definizione del framework DigComp (2013) e, in Italia, il Piano Nazionale Scuola Digitale (2015) e le Indicazioni Nazionali (MIUR, 2015, 2018), sono tutti eventi che arrivano dopo decenni di sperimentazione pedagogica e di evoluzione tecnologica "dal basso". Anche iniziative no-profit come la Settimana del Codice Europea (nata nel 2013) e l'Ora del Codice (promossa da Code.org) si inseriscono in questo solco, promuovendo valori di accessibilità, creatività e problem solving (CodeWeek.eu; Hour of Code).

Conclusione: riappropriarsi della complessità pedagogica

Questa ricostruzione storica rivela una verità fondamentale: il coding e la robotica educativa non sono nati da un'ispirazione meramente tecnica o economica, ma da una profonda riflessione pedagogica. Il termine "pensiero computazionale", utile per focalizzare l'attenzione politica e istituzionale su queste discipline, rischia oggi di appiattire e nascondere la ricchezza e la complessità di questo percorso (Bocconi et al., 2016, 2022). A distanza di oltre dieci anni dalle prime introduzioni nei curricoli, è tempo che i docenti, in quanto professionisti dell'apprendimento, si ri-

approprino di questa storia. Comprenderne le radici pedagogiche (Papert, il costruzionismo), le dinamiche sociali (l'etica open source, il movimento maker) e le convergenze storiche è essenziale per passare da un uso strumentale e talvolta superficiale della tecnologia a una progettazione didattica consapevole, critica e realmente trasformativa, che ponga al centro non la macchina, ma l'alunno e il suo processo di crescita. È necessario, come suggerito in origine, "sviscerare il discorso" e andare a riprendere quei temi sul "fare scuola" che gli ideatori di queste tecnologie perseguivano.

La seguente tabella fornisce una mappa degli eventi significativi in funzione di pedagogia e ricerca, sviluppo tecnologico, sviluppo giuridico e sviluppo istituzionale e normativo. Ha, cioè due funzioni: quella di evidenziare degli avvenimenti significativi rispetto alla cronologia e di fornire delle categorie attraverso cui identificarli, mostrando come l'interazione di queste categorie è fondamentale per lo sviluppo scientifico e sociale.

Tabella 4.1: Cronologia convergente: pedagogia, tecnologia open e quadro normativo.

Anno	Evento	Pedagogia e Ricerca	Tecnologia	Giuridico	Istituzionale/ Normativo
1970	Papert sviluppa l'idea del computer come strumento di apprendimento.	Prime idee costruzioniste.	Primi ambienti di programmazione per bambini.		
1972	Papert & Solomon pubblicano *Twenty Things to Do with a Computer.*	Proposta concreta di attività educative con il calcolatore.			
1980	Pubblicazione di "Mindstorms" di Seymour Papert.	Formalizzazione del costruzionismo. Teoria degli *objects-to-think-with.*	Presentazione del linguaggio LOGO e della "tartaruga" robotica.		
1984	Nasce la Free Software Foundation (FSF).			Nascita del movimento del software libero (Richard Stallman). Licenza GNU GPL (v1, 1989) che introduce il principio del *copyleft.*	
1991	Nascita del World Wide Web (Tim Berners-Lee).		Strumento per la condivisione globale di informazioni.	Rilasciato nel dominio pubblico, non brevettato.	

Anno	Evento	Pedagogia e Ricerca	Tecnologia	Giuridico	Istituzionale/ Normativo
1995	Pubblicazione di *I bambini e il computer* (ed. it. di *The Children's Machine*) di Papert.	Riflessione sull'impatto del computer sui processi di apprendimento e di pensiero.			
1998	LEGO Mindstorms commerciale (RCX).	La robotica educativa esce dai laboratori accademici.	Primo kit robotico programmabile commerciale di successo.		
1998	Nascita della FIRST LEGO League.	Introduzione di *Core Values* etici (scoperta, inclusione, cooperazione).	Competizioni con kit LEGO Mindstorms.		
2001	Nascita di Wikipedia.	Modello di costruzione collaborativa e aperta della conoscenza.	Piattaforma wiki.	Contenuti rilasciati con licenza libera (GNU FDL).	
2002	Creazione delle licenze Creative Commons.			Alternative flessibili al *copyright* tradizionale per contenuti creativi.	
2003	Sviluppo di Arduino.		Microcontroller open hardware, accessibile e programmabile.	Schemi elettrici e software rilasciati open source.	
2005	Nascita del progetto RepRap (stampante 3D autoriproducibile).		Open hardware per la fabbricazione digitale personale.		
2006	Jeannette Wing pubblica l'articolo *Computational Thinking*.	Lancio del concetto di pensiero computazionale come competenza fondamentale per tutti.			
2006	Raccomandazione UE sulle Competenze Chiave.				La competenza digitale diventa una delle 8 competenze chiave per l'apprendimento permanente.
2007	DM 139/2007 (Italia).				Recepisce le raccomandazioni UE del 2006, definendo le 8 competenze chiave nel sistema italiano.

Anno	Evento	Pedagogia e Ricerca	Tecnologia	Giuridico	Istituzionale/ Normativo
2007	Pubblicazione di Scratch (MIT Media Lab).	Sintesi del costruzionismo: metodo delle 4P (Progetti, Passione, Pari, Play).	Software di programmazione a blocchi gratuito.	Progetto open source.	
2008	Lancio del robot umanoide NAO (Aldebaran Robotics).	Diventa una piattaforma di ricerca e didattica per robotica e IA.	Robot programmabile, utilizzato in ricerca e scuole.		
2012	Indicazioni Nazionali per il curricolo (MIUR).				Introduzione delle competenze di base, tra cui quella digitale.
2012	Introduzione dell'informatica nel curricolo obbligatorio nel Regno Unito.				Primo grande paese europeo a introdurre il *computing* come materia.
2013	Prima versione del framework DigComp (1.0) del JRC.				Fornisce un modello di riferimento comune europeo per la competenza digitale.
2013	Prima Europe Code Week.	Promozione del coding come alfabetizzazione.			Iniziativa della Commissione Europea.
2015	Piano Nazionale Scuola Digitale (PNSD) in Italia.	Azione #17: "Portare il pensiero computazionale a tutta la scuola primaria".	Finanziamento per atelier creativi (making, coding).		Normativa ministeriale italiana di sistema.
2016	DigComp 2.0 e 2.1 (aggiornamento con esempi).				Affinamento del framework europeo.
2018	Indicazioni Nazionali e Nuovi Scenari (MIUR).	Introduzione esplicita del pensiero computazionale tra le competenze di base.			Aggiornamento del curricolo italiano.
2022	Lancio di ChatGPT (OpenAI).	Accesso di massa all'IA conversazionale; nuove sfide per la didattica e la valutazione.	Modello proprietario di Large Language Model (LLM) che democratizza l'accesso all'IA generativa testuale.		

Anno	Evento	Pedagogia e Ricerca	Tecnologia	Giuridico	Istituzionale/ Normativo
2022	Pubblicazione di "Reviewing Computational Thinking" (JRC, Bocconi et al.).	Analisi critica dell'evoluzione del PCT e proposta delle 3 aree (informatica, concetti, problem solving).			Studio di riferimento per le politiche educative europee.
2022	DigComp 2.2 (inclusione di IA e contrasto alla disinformazione).				Integrazione delle competenze per l'era dell'IA generativa.
2023	Lancio di DeepSeek (Depth Seek).	Nuovo attore nel panorama dell'IA open-source e gratuita per la ricerca e l'educazione.	Modello di IA open-source e gratuito che promuove l'accessibilità e la trasparenza nella ricerca.		
2025	Pubblicazione del DigComp 3.0.				Aggiornamento con maggiore enfasi su etica, inclusione e strumenti di validazione.

Riferimenti bibliografici

Bocconi, S. et al. (2016). *Developing Computational Thinking in Compulsory Education: Implications for Policy and Practice* (P. Kampylis, Y. Punie, A cura di). Publications Office of the European Union.

Bocconi, S. et al. (2022). *Reviewing Computational Thinking in Compulsory Education* (A. Inamorato Dos Santos, R. Cachia, N. Giannoutsou, Y. Punie, A cura di). Publications Office of the European Union.

MIUR – Ministero dell'Istruzione, dell'Università e della Ricerca. (2015). *Piano Nazionale Scuola Digitale*.

MIUR – Ministero dell'Istruzione, dell'Università e della Ricerca. (2018). *Indicazioni Nazionali e Nuovi Scenari*.

Nulli, G. (2019). Costruire autonomia e consapevolezza attraverso coding robotica e making. In D. E. Bettini & E. Tondini (a cura di), *La prevenzione via per un nuovo sviluppo. Atti del Forum del Gran Sasso* (Vol. 2, Parte 3, Area 9, pp. 131-156). Teramo.

Papert, S. (1984). *Mindstorms. Bambini, computer e creatività* (ed. or. 1980). Emme.

Resnick, M. (2018). *Come i bambini. Immagina, crea, gioca e condividi. Coltivare la creatività con il Lifelong Kindergarten del MIT*. Erickson.

Risorse online

CodeWeek.eu. (s.d.). I nostri valori. https://codeweek.eu/

Hour of Code. (s.d.). https://hourofcode.com/it

4.2. La generazione di immagini con l'intelligenza artificiale: tecnologie, opportunità e responsabilità educative

Lorenzo Guasti

4.2.1. Introduzione

L'intelligenza artificiale generativa rappresenta uno degli sviluppi più rilevanti dell'innovazione digitale contemporanea e sta progressivamente incidendo anche sui contesti educativi. Tra le sue applicazioni più diffuse e accessibili rientrano i sistemi di generazione di immagini a partire da descrizioni testuali (*text-to-image*), che consentono di produrre rappresentazioni visive attraverso sofisticati modelli di apprendimento automatico.

Nel contesto scolastico, tali tecnologie offrono nuove opportunità per la didattica, in particolare per la visualizzazione di concetti astratti, il supporto alla creatività e l'inclusione. Allo stesso tempo, il loro utilizzo richiede una comprensione di base dei meccanismi di funzionamento, una riflessione sui limiti intrinseci dei modelli e un'attenzione costante alle implicazioni etiche e pedagogiche.

Il presente capitolo intende fornire un inquadramento introduttivo ma rigoroso delle principali tecnologie di generazione di immagini, analizzandone le potenzialità educative, i rischi e le condizioni per un uso consapevole nella scuola.

4.2.2. Inquadramento tecnologico

I moderni sistemi di generazione di immagini non operano secondo modalità assimilabili al disegno umano. Le tecnologie oggi più diffuse si basano prevalentemente sui cosiddetti modelli di diffusione, una classe di modelli probabilistici formalizzata scientificamente nel 2020 da Ho, Jain e Abbeel nel loro articolo seminale pubblicato negli Advances in Neural Information Processing Systems (NeurIPS).

Il principio fondamentale di questi modelli è il denoising: durante la fase di addestramento, il modello impara progressivamente a rimuovere il rumore da immagini reali. Durante la fase di generazione, il processo viene invertito: si parte da una distribuzione casuale di pixel (puro rumore) e, attraverso una sequenza di passaggi iterativi, si ottiene un'immagine coerente con la descrizione testuale fornita.

Un'evoluzione significativa di questo approccio è rappresentata dai Latent Diffusion Models, descritti da Rombach, Blattmann, Lorenz, Esser e Ommer nel 2022 e presentati alla conferenza IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). In questi modelli, il processo di diffusione non avviene direttamente sull'immagine a piena risoluzione, ma su una rappresentazione compressa (lo "spazio latente") ottenuta tramite un autoencoder. Questo approccio riduce drasticamente i costi computazionali — accelerando l'inferenza di almeno 2,7 volte rispetto alla diffusione diretta — rendendo la tecnologia più efficiente e accessibile.

Nel contesto educativo, la conoscenza di questi principi non ha finalità tecniche o ingegneristiche, ma contribuisce allo sviluppo di una alfabetizzazione di base sull'intelligenza artificiale, utile per comprendere le potenzialità e i limiti degli strumenti utilizzati.

4.2.3. La formulazione dei prompt come competenza trasversale

La qualità delle immagini generate dipende in larga misura dalla capacità dell'utente di formulare istruzioni testuali chiare, coerenti e contestualizzate, comunemente denominate prompt. La costruzione del prompt può essere interpretata come una forma di mediazione comunicativa tra l'intenzione umana e il comportamento del sistema.

Liu e Chilton, in uno studio pubblicato negli atti della conferenza CHI 2022 sull'interazione uomo-computer (ACM Conference on Human Factors in Computing Systems), hanno analizzato sistematicamente 5.493 generazioni di immagini per identificare le pratiche più efficaci nella formulazione dei prompt. Le loro conclusioni, validate empiricamente, indicano che i prompt più efficaci condividono alcune caratteristiche ricorrenti: specificità del soggetto, chiarezza del contesto, assenza di ambiguità e coerenza interna.

In ambito scolastico, un prompt efficace dovrebbe includere: il soggetto principale dell'immagine, il contesto o l'ambientazione, lo stile desiderato (fotografico, illustrativo, pittorico) e, quando rilevante, indicazioni

sulla destinazione d'uso. Chiedere contemporaneamente elementi contraddittori — ad esempio un paesaggio che sia al tempo stesso marino e montano — genera confusione nel modello e risultati incoerenti.

L'uso guidato dei generatori di immagini permette inoltre di sperimentare processi iterativi di revisione: l'analisi dei risultati ottenuti e la modifica progressiva delle istruzioni favoriscono una riflessione metacognitiva sul rapporto tra linguaggio, rappresentazione e interpretazione. Questa attività può essere valorizzata come esercizio di scrittura funzionale, di sintesi e di precisione linguistica.

4.2.4. Strumenti di generazione e criteri di scelta

Il panorama degli strumenti di generazione di immagini è in rapida evoluzione e comprende sia piattaforme commerciali sia soluzioni open source. In ambito scolastico, la scelta degli strumenti deve essere orientata prioritariamente da criteri di sicurezza, affidabilità e moderazione dei contenuti.

DALL·E, sviluppato da OpenAI e integrato nei servizi Microsoft, rappresenta una delle opzioni più affidabili per l'ambiente scolastico grazie ai suoi robusti filtri di contenuto. Accessibile attraverso Bing Image Creator in modo gratuito, offre un elevato livello di moderazione che minimizza il rischio di generazione di contenuti inappropriati.

S*table* Diffusion, sviluppato inizialmente da Stability AI e rilasciato come modello open source, rappresenta un'alternativa potente ma che richiede maggiore cautela in ambito educativo. La sua natura aperta significa che esistono numerose implementazioni online con livelli di moderazione variabili. Alcuni portali potrebbero includere personalizzazioni — tecnicamente denominate LoRA (Low-Rank Adaptation) — che orientano il modello verso stili o contenuti non appropriati per il contesto scolastico.

È responsabilità dell'istituzione scolastica e del docente selezionare piattaforme adeguate all'età degli studenti, verificare preventivamente i risultati generati e definire regole chiare di utilizzo.

4.2.5. Bias, rappresentazioni e responsabilità educativa

Un aspetto critico dei sistemi di generazione di immagini riguarda la presenza di bias, ovvero distorsioni sistematiche nelle rappresentazioni prodotte. Tali bias derivano principalmente dai dati di addestramento,

che riflettono le asimmetrie culturali, sociali e geografiche presenti nelle fonti disponibili in rete.

La ricerca scientifica ha documentato ampiamente questo fenomeno. Uno studio dell'Università di Washington presentato alla conferenza EMNLP 2023 (Conference on Empirical Methods in Natural Language Processing) ha dimostrato che S*table* Diffusion, quando riceve la richiesta di generare l'immagine di "una persona", produce sistematicamente rappresentazioni di uomini con carnagione chiara, sottorappresentando le identità non binarie e i popoli indigeni (Ghosh, Caliskan et al., 2023).

Queste dinamiche rendono evidente che le immagini generate non sono riproduzioni oggettive della realtà, ma costruzioni statistiche basate su distribuzioni di dati. In ambito educativo, tali limiti possono essere trasformati in oggetto di riflessione critica, offrendo l'opportunità di discutere con gli studenti il rapporto tra tecnologia e società, il concetto di neutralità algoritmica e il ruolo delle scelte umane nella progettazione dei sistemi digitali.

4.2.6. Supervisione e contesto d'uso

L'impiego dei generatori di immagini nella didattica richiede una supervisione attiva e consapevole. Anche in presenza di sistemi dotati di filtri di moderazione, possono verificarsi risultati inattesi o non coerenti con il contesto educativo.

Un esempio concreto illustra questo rischio: la richiesta di rappresentare un gruppo familiare composto da un adulto di quarant'anni, una ragazza di quindici anni e un ragazzo di undici può, su alcune piattaforme, essere reinterpretata dal sistema in modi inappropriati, enfatizzando indebitamente alcuni aspetti della descrizione a scapito di altri. Questo tipo di comportamento anomalo, spesso dovuto a personalizzazioni del modello orientate verso determinati stili, sottolinea l'importanza di testare preventivamente gli strumenti e mantenere una supervisione attiva durante le attività didattiche.

È pertanto raccomandabile che i docenti sperimentino preventivamente gli strumenti, definiscano obiettivi didattici chiari e accompagnino gli studenti nella valutazione critica dei contenuti prodotti. L'uso della tecnologia deve essere sempre subordinato al progetto educativo.

4.2.7. Applicazioni didattiche e prospettive inclusive

I generatori di immagini possono essere integrati in numerose discipline e attività didattiche. Nelle scienze, consentono di visualizzare processi non osservabili direttamente — come le transizioni evolutive per le quali non esistono testimonianze fotografiche. Nella storia e nella geografia, permettono di esplorare rappresentazioni di contesti lontani nel tempo e nello spazio. Nelle discipline linguistiche e artistiche, supportano attività di narrazione e storytelling.

La ricerca sulle tecnologie assistive basate sull'intelligenza artificiale sta documentando risultati promettenti: una meta-analisi condotta da Zhang e colleghi nel 2024 — la prima focalizzata specificamente sugli studenti con disabilità in contesti pre-K–12 — ha rilevato un effetto complessivo di entità media (Hedges' g = 0,588) su 29 studi (quasi-) sperimentali condotti a livello globale, con interventi basati su robot, software educativi e sistemi di realtà virtuale intelligente (Zhang et al., 2024).

In una prospettiva coerente con i principi dell'Universal Design for Learning, la generazione di immagini può contribuire alla personalizzazione degli apprendimenti e all'ampliamento dei canali comunicativi disponibili, permettendo a tutti gli studenti di esprimere la propria creatività attraverso il medium che padroneggiano meglio.

4.2.8. Conclusioni

L'integrazione dei sistemi di generazione di immagini nella scuola rappresenta un'opportunità significativa, ma richiede un approccio critico, informato e responsabile. La scuola è chiamata non solo a utilizzare queste tecnologie, ma anche a promuovere una comprensione consapevole dei loro meccanismi, dei loro limiti e delle loro implicazioni etiche.

Gli studenti devono essere accompagnati nello sviluppo di una *literacy* dell'intelligenza artificiale che li renda fruitori consapevoli e critici. Comprendere come funzionano i modelli di diffusione, riconoscere i bias nelle rappresentazioni generate, formulare prompt efficaci e valutare criticamente i risultati ottenuti sono competenze che trascendono l'uso specifico dei generatori di immagini e preparano a un'interazione matura con l'ecosistema sempre più pervasivo dell'intelligenza artificiale generativa.

Accompagnare studenti e docenti nello sviluppo di competenze di lettura critica dell'intelligenza artificiale significa contribuire alla formazione di cittadini capaci di interagire in modo responsabile con un ecosistema digitale sempre più complesso.

Riferimenti bibliografici

Ghosh, S., Caliskan, A., et al. (2023). S*table* Diffusion perpetuates racial and gendered stereotypes. *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics.

Ho, J., Jain, A., & Abbeel, P. (2020). Denoising Diffusion Probabilistic Models. *Advances in Neural Information Processing Systems*, 33, 6840-6851.

Liu, V., & Chilton, L. B. (2022). Design Guidelines for Prompt Engineering Text-to-Image Generative Models. *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems* (Article 384, pp. 1–23). ACM.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., & Ommer, B. (2022). High-Resolution Image Synthesis with Latent Diffusion Models. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 10684-10695.

Zhang, L., Carter, R. A. Jr., Liu, Y., & Peng, P. (2024). Let's CHAT about Artificial Intelligence for students with disabilities: A systematic literature review and meta-analysis. *Review of Educational Research*, 96(1), 215–257. https://doi.org/10.3102/00346543241293424

5.
ESERCIZI DI PROGRAMMAZIONE: COME STRUTTURARE UN CURRICULUM DI ROBOTICA

David Scaradozzi, Martina Morano, Flavia Gioiello e Benedetta Castagna

5.1. Introduzione

Negli ultimi anni la robotica educativa si è progressivamente affermata come uno strumento efficace per innovare la didattica e rendere più attivi i processi di apprendimento (Mangina et al., 2024). Eppure, nonostante la crescente disponibilità di strumenti dedicati e la moltiplicazione delle sperimentazioni nelle scuole, essa continua a occupare, in molti contesti, una posizione periferica: compare come progetto extracurricolare, come iniziativa temporanea, come esperienza aggiuntiva rispetto al percorso ordinario. Ciò non è secondario, perché per essere educativa e non informativa il modo in cui la robotica viene inserita nella progettazione didattica ordinaria determina in larga misura ciò che gli studenti effettivamente imparano.

A questa criticità si aggiunge un secondo elemento, altrettanto rilevante: la formazione dei docenti. Come evidenziato da Tondeur et al. (2017), le convinzioni pedagogiche dei docenti giocano un ruolo decisivo nelle scelte relative all'uso delle tecnologie in aula. Non basta che gli strumenti siano disponibili: occorre che chi insegna sappia come interpretarli, valorizzarli e collegarli agli obiettivi educativi. In assenza di questa visione, i contenuti legati alla robotica rischiano di essere introdotti in modo frammentario, perdendo gran parte della loro efficacia.

Vale poi la pena distinguere due definizioni spesso sovrapposte. La prima, talvolta definita *robotics in education*, usa il robot come supporto per insegnare altre discipline: uno strumento che rende più accattivante una lezione, ma che resta sullo sfondo. La seconda, la *robotica educativa* propriamente detta, è qualcosa di diverso: un ambiente in cui lo studente progetta, costruisce e programma artefatti, confrontandosi con problemi autentici, vincoli reali e processi iterativi di miglioramento. La robotica educativa si configura come un contesto privilegiato per sviluppare

competenze trasversali quali il problem solving, il lavoro collaborativo, il pensiero progettuale e la capacità di modellizzare fenomeni complessi (Scaradozzi, Screpanti, et al., 2019), competenze oggi centrali non solo sul piano tecnologico, ma anche su quelli sociale, economico e ambientale (Sabri, 2025).

Il problema, dunque, non è l'introduzione di ulteriore tecnologia nelle scuole. È costruire un percorso didattico strutturato e coerente, in cui la presenza del robot non sia fine a sé stessa, ma parte di un impianto pedagogico che aiuti gli alunni a passare dall'idea alla sua realizzazione concreta. Questa impostazione è coerente con le prospettive educative riconducibili al paradigma maker e all'apprendimento laboratoriale, che attribuiscono un ruolo centrale alla costruzione attiva di artefatti e alla riflessione sul processo di progettazione (Papert, 1980). In tali contesti, la robotica educativa consente di coniugare conoscenze teoriche e pratiche, rendendo visibile il rapporto tra idea, rappresentazione, azione e verifica (Di Stasio & Miotti, 2021).

5.2. La rilevanza della robotica educativa nella scuola primaria: la robotica come componente curricolare

Numerosi studi hanno evidenziato come l'introduzione delle discipline STEM fin dai primi anni di istruzione possa sostenere i processi di apprendimento e rafforzare la motivazione degli studenti. Nella scuola primaria, tuttavia, i curricula tradizionali continuano spesso a riservare maggiore spazio a discipline consolidate, mentre gli ambiti legati alla tecnologia rimangono meno sviluppati. La robotica può colmare parte di questo spazio, avvicinando i bambini (e i ragazzi) ai concetti fondamentali della programmazione e della tecnologia attraverso attività *hands-on*, e favorendo al contempo competenze relazionali e organizzative che emergono naturalmente nel lavoro collaborativo, nella costruzione di soluzioni, nel confronto con l'errore (Scaradozzi, Screpanti, et al., 2019).

Le sperimentazioni condotte nella scuola primaria mostrano che questi benefici sono reali, ma non automatici: emergono in modo stabile solo quando la robotica è parte di una progettazione didattica coerente, non un'attività occasionale del percorso (Scaradozzi et al., 2015). Ciò richiede tuttavia una progettazione curricolare attenta, in cui le attività siano distribuite lungo i diversi anni scolastici, coerenti con l'età degli studenti e integrate nel quadro del syllabus nazionale.

Nella scuola primaria, un esempio concreto di come questo percorso possa essere strutturato è offerto dal curricolo quinquennale sviluppato e sperimentato nel progetto OpenFISH.Science (Scaradozzi, Cesaretti, et al., 2019). Il curricolo introduce la robotica come materia ordinaria, non extracurricolare, distribuita in tutti e cinque gli anni della scuola primaria, con una progressione intenzionale che parte dai fondamenti e si estende, negli anni conclusivi, verso concetti di Internet of Things, di controllo e di automazione.

Nel primo anno l'obiettivo non è ancora costruire un robot funzionante, ma preparare il terreno concettuale che lo renda possibile. Gli studenti imparano a riconoscere e classificare gli elementi meccanici, a descrivere la posizione di oggetti nello spazio, a comprendere il concetto di verifica e a familiarizzare con i principi della RoboEtica come primo approccio alla relazione tra macchine e responsabilità umana. L'attività pratica principale consiste nella progettazione di un semplice artefatto statico con materiali strutturati o di recupero: un oggetto costruito con intenzione, che richiede di scegliere i materiali, interpretare le istruzioni e verificare il risultato.

Nel secondo anno il concetto di robot viene esplicitamente introdotto: una macchina che esegue un compito specifico in modo autonomo. Gli studenti costruiscono il loro primo robot semplice e incontrano per la prima volta la distinzione tra sensori e attuatori anche attraverso un'analogia con il corpo umano.

Nel terzo anno il percorso compie un salto qualitativo con l'introduzione dei dispositivi programmabili: gli studenti iniziano a governare il comportamento del robot attraverso sequenze di istruzioni, con i primi accenni a sistemi in cui più componenti comunicano tra loro.

Nel quarto anno, gli studenti costruiscono robot autonomi in grado di comunicare e di reagire all'ambiente. Il codice non è più una sequenza lineare, ma incorpora selezioni e ripetizioni — i costrutti fondamentali della programmazione procedurale. La verifica e la validazione diventano parte integrante del processo: il robot non si considera finito finché non si dimostra che fa ciò che dovrebbe fare.

Nel quinto anno il curricolo raggiunge la sua massima complessità: si introducono il codice riutilizzabile, le reti di sensori, l'attuazione distribuita e i sistemi di controllo centralizzati e distribuiti. Per chi segue il percorso con attenzione all'ambiente, questo è anche l'anno in cui gli studenti assemblano un robot ispirato alla morfologia di un animale acquatico, integrando sensori, attuatori e strategie di navigazione in un sistema progettato per esplorare e monitorare l'ambiente marino.

Al di là della progressione dei contenuti, ciò che caratterizza il curricolo è la struttura ritmica delle attività, che si ripete ogni anno con variazioni di complessità crescente. Ogni unità didattica si apre con un momento di osservazione e raccolta: gli studenti esplorano il contesto tecnologico, discutono i ruoli dei robot nella società e affrontano le implicazioni etiche della progettazione. Segue la fase di costruzione dell'artefatto, in cui i materiali vengono selezionati, assemblati e testati. Si passa poi alla programmazione, condotta secondo un approccio di *peer tutoring* in cui gli studenti sono incoraggiati a spiegarsi il codice a vicenda prima ancora che intervenga l'insegnante. Ogni ciclo si chiude con una fase di verifica e revisione, sia individuale sia di gruppo. Questo ritmo riflette il modello *Think–Make–Improve* (Martinez & Stager, 2019): gli studenti osservano un problema, formulano un'ipotesi, progettano una soluzione, la testano e la migliorano. In tutto il percorso, le attività sono organizzate in gruppi di lavoro con ruoli definiti, rendendo la dimensione collaborativa parte strutturale dell'esperienza, non un accessorio.

La robotica educativa assume pieno significato, allora, quando viene pensata come componente curricolare: non come esperienza accessoria, ma come contesto in cui teoria e pratica si incontrano, in cui la rappresentazione diventa azione e la dimensione cognitiva si intreccia con quella laboratoriale. Perché questo sia possibile, tuttavia, non è sufficiente stabilire cosa insegnare, con quale progressione e secondo quale metodologia. Occorre anche domandarsi quali strumenti siano più adatti a sostenere, nei diversi momenti del percorso, gli obiettivi educativi che si intendono perseguire.

5.3. Strumenti a supporto della didattica della robotica

Se la robotica deve essere integrata in un curriculum progressivo e coerente, anche la scelta degli strumenti non può essere affidata esclusivamente alla loro disponibilità commerciale o alla loro diffusione nelle scuole. Gli strumenti tecnologici, infatti, orientano le forme di interazione, rendono possibili alcune attività e ne limitano altre, rendono più o meno trasparente il funzionamento dei sistemi e influenzano direttamente il tipo di esperienza di apprendimento offerta agli studenti. Per questo motivo, la selezione delle tecnologie dovrebbe essere guidata da criteri pedagogici prima ancora che tecnici.

Un primo criterio riguarda la coerenza tra lo strumento, gli obiettivi didattici e il livello di sviluppo cognitivo degli studenti. Le fasi iniziali del percorso richiedono ambienti semplici e accessibili, capaci di rendere immediatamente comprensibile la relazione tra istruzione e azione. Le fasi successive possono invece prevedere strumenti che permettano una maggiore profondità nell'esplorazione dei sistemi tecnologici. La scelta degli strumenti è parte integrante della progettazione didattica stessa.

Una prima categoria rilevante è quella degli ambienti di programmazione. È possibile distinguere, in senso generale, tra ambienti visuali e ambienti testuali. I primi rappresentano il codice attraverso blocchi grafici combinabili, riducendo il carico sintattico e spostando l'attenzione sulla logica delle istruzioni e sulle relazioni tra le diverse parti del programma. Per questa ragione risultano particolarmente adatti nelle prime fasi del percorso, quando l'obiettivo principale è aiutare gli studenti a costruire una comprensione strutturata dei concetti fondamentali del coding. Ambienti come ScratchJr e Scratch consentono di introdurre gradualmente questi principi adattando le attività alle diverse età: il loro valore non risiede solo nella semplicità d'uso, ma nella capacità di rendere visibili le strutture logiche del programma e di sostenere forme di apprendimento attive e sperimentali.

La programmazione visuale non rappresenta tuttavia un punto di arrivo, ma una fase di un percorso più ampio. L'introduzione progressiva alla programmazione testuale, con linguaggi come Python, utilizzato attraverso editor o *notebook* interattivi, permette agli studenti di acquisire una comprensione più articolata della struttura dei programmi, mantenendo una continuità concettuale con le esperienze precedenti.

Accanto agli ambienti di programmazione, un secondo gruppo di strumenti comprende i dispositivi fisici: kit robotici, microcontrollori, sensori e attuatori. Questi strumenti svolgono un ruolo cruciale perché collegano il codice al comportamento di un sistema reale, rendendo tangibile il rapporto tra software, hardware e ambiente. Nelle prime fasi, questa relazione può essere introdotta anche attraverso attività unplugged che aiutino gli studenti a padroneggiare concetti quali sequenza, orientamento e risposta del sistema, prima ancora di interagire con un dispositivo fisico.

Con il percorso è possibile introdurre sistemi più articolati. In questo contesto risultano particolarmente significative le piattaforme modulari, composte da elementi combinabili in base agli obiettivi didattici. La modularità non è solo una caratteristica tecnica: ha un valore educativo preciso, perché consente di mostrare agli studenti che un sistema robo-

tico non è una scatola chiusa, ma un insieme di componenti che interagiscono tra loro, ciascuno con una funzione specifica. Sistemi di questo tipo, detti *white box*, permettono agli studenti di osservare come il comportamento emerga dall'interazione tra le parti e di intervenire su di esse in modo consapevole. Un ulteriore elemento di interesse è la possibilità di programmare questi dispositivi sia tramite ambienti visuali sia tramite linguaggi testuali, il che li rende adatti ad accompagnare gli studenti lungo una progressione che spazia dalle interazioni semplici alle forme di progettazione più strutturate.

Nelle fasi più avanzate del curriculum possono poi essere introdotte piattaforme che consentono di affrontare in modo più esplicito concetti legati all'elettronica, al funzionamento dei sistemi digitali e alla gestione di sensori e attuatori. Anche in questo caso, l'interesse didattico non risiede nel valore tecnologico dello strumento in sé, bensì nel tipo di esperienza cognitiva che lo rende possibile.

Questo porta a considerare un criterio di scelta che nella robotica educativa assume un peso particolare: il grado di trasparenza dello strumento. Un rischio ricorrente nell'uso scolastico delle tecnologie è trasformare la robotica in un'esperienza di tipo *black box*, in cui gli studenti osservano un risultato senza comprendere quali componenti siano in gioco e quali processi rendano possibile il comportamento del sistema. Alcuni studi hanno evidenziato la tensione tra il potenziale educativo degli approcci maker e l'uso esclusivo di kit completamente prefabbricati, che talvolta riducono la possibilità di esplorare il funzionamento interno dei dispositivi (Alimisis et al., 2019). Ciò non significa escludere i kit pronti all'uso: in molti contesti rappresentano una soluzione efficace, soprattutto nelle fasi iniziali. La questione non è opporre rigidamente strumenti prefabbricati e modulari, ma costruire un equilibrio didattico tra semplicità d'uso, accessibilità e profondità dell'esperienza formativa.

5.4. Un esempio di percorso formativo

La struttura curricolare descritta nelle sezioni precedenti non riguarda solo gli studenti. Per poter progettare e condurre attività di robotica educativa in modo consapevole, anche i docenti hanno bisogno di un percorso formativo che ne rispecchi la stessa logica progressiva: partire dai fondamenti, costruire familiarità con gli strumenti, sperimentare prima di insegnare.

Il percorso qui descritto non è un curricolo verticale completo, ma un esempio condensato che attraversa i principali nodi concettuali e pratici dagli strumenti per la scuola dell'infanzia fino alla programmazione avanzata con l'obiettivo di offrire ai docenti una visione d'insieme prima ancora che una sequenza da replicare anno per anno. Il percorso si articola in sei moduli con una progressione che è intenzionale: ogni modulo prepara il terreno per quello successivo e la durata complessiva è volutamente contenuta, nell'idea che la formazione debba essere densa ma accessibile.

La Tabella 5.1 fornisce una descrizione dei contenuti affrontati nei sottomoduli e i principali strumenti adottati. Va sottolineato come gli strumenti qui descritti non siano gli unici disponibili: essi sono stati selezionati in quanto allineati con gli obiettivi degli argomenti trattati nei sottomoduli. Fintanto che gli strumenti sono coerenti con le scelte pedagogiche effettuate, possono essere sostituiti con alternative equivalenti.

Tabella 5.1: Sottomoduli relativi al modulo di programmazione e alla robotica. Vengono fornite informazioni relativamente ai contenuti principali di ciascun sottomodulo con i principali strumenti adottati.

#	Titolo	Durata	Contenuti principali	Strumenti
1	Principi di informatica	1 h	Pensiero computazionale, algoritmi, diagrammi di flusso, variabili, strutture dati, istruzioni sequenziali/condizionali/iterative, funzioni, classi, librerie	–
2	Scratch e i linguaggi di programmazione	2 h	Introduzione ai linguaggi di programmazione, IDE, programmazione a blocchi (Scratch: sprite, stage, script), programmazione testuale (Python: Hello World, somma di numeri, gioco con numero casuale), UIFlow	Scratch, Python (Google Colab), UIFlow
3	Robotica e coding per infanzia e primaria (cl. 1–2)	1 h	Introduzione alla robotica, coding unplugged (percorso del robot), ScratchJr, programmazione di Guizzo con UIFlow (sequenze di colori)	ScratchJr, UIFlow + Guizzo (M5StickC Plus)
4	Robotica e coding per primaria e secondaria I grado	2 h	Componenti del robot (sensori, microcontrollore, attuatori), programmazione visuale e testuale con UIFlow: accensione LED, lettura sensore di temperatura (ENV IV unit), equivalenti Python	UIFlow (Blockly + Python), M5StickC Plus, ENV IV unit

#	Titolo	Durata	Contenuti principali	Strumenti
5	Programmazione avanzata	2 h	Elettronica ed elettrotecnica (corrente, tensione, legge di Ohm, componenti passivi e attivi, porte logiche, circuiti combinatori e sequenziali), telecomunicazioni, microcontrollori e microprocessori, Arduino IDE: accensione LED, semaforo a tre colori	Arduino IDE, Arduino Uno, breadboard, LED, resistenze
6	Esempi	1 h	Esempi di robotica e di coding creativo	Eddie, SonicPI, Scratch

5.5. Valutazione degli apprendimenti nelle attività di robotica educativa

In un contesto educativo, non può essere trascurata la dimensione della valutazione dell'apprendimento. Nelle attività di robotica educativa essa assume un ruolo particolarmente delicato, poiché non si tratta soltanto di accertare l'acquisizione di conoscenze specifiche, ma anche di comprendere come gli studenti affrontano i problemi, collaborano, rivedono le proprie strategie e costruiscono progressivamente il significato delle azioni che svolgono.

La valutazione deve dunque tenere insieme la comprensione dei contenuti, la percezione di sé rispetto alla tecnologia e la qualità del percorso di apprendimento sviluppato durante l'attività. Questa esigenza deriva dalla natura stessa delle attività di robotica, spesso caratterizzate da compiti *open-ended* in cui non esiste un'unica soluzione corretta né un solo percorso valido per raggiungere l'obiettivo. Una valutazione che si limiti a verificare se il robot "funziona" o meno risulterebbe riduttiva: il prodotto finale è certamente un elemento osservabile, ma non esaurisce il valore formativo dell'esperienza. È necessario considerare anche il processo di progettazione, le strategie di problem solving adottate, la capacità di revisione e di debugging, la gestione dell'errore e il livello di autonomia raggiunto.

In questa prospettiva è possibile distinguere due principali categorie di strumenti (Scaradozzi et al., 2020). La prima comprende modalità di assessment più tradizionali: questionari, rubriche di valutazione, griglie di osservazione strutturate, test di conoscenza, utili a raccogliere informazioni sulle conoscenze, sugli atteggiamenti e sulle percezioni degli

studenti. Questi strumenti vengono generalmente utilizzati all'inizio e alla fine del percorso per osservare eventuali cambiamenti, ma possono essere impiegati anche durante le attività attraverso, ad esempio, osservazioni sistematiche. Il loro valore risiede nella capacità di rendere leggibili aspetti dell'apprendimento che non emergono dall'osservazione del prodotto tecnico: il modo in cui gli studenti collaborano, motivano le proprie scelte, affrontano le difficoltà e rielaborano gli errori.

La seconda categoria comprende strumenti basati sulla raccolta automatica dei dati generati durante l'interazione degli studenti con ambienti di programmazione e dispositivi robotici. Le sequenze di comandi, le modifiche al codice, i tentativi di correzione, i tempi di esecuzione e le iterazioni del programma possono essere registrati e analizzati, offrendo al docente la possibilità di osservare con maggiore dettaglio il percorso effettivamente seguito dagli studenti e di mettere in luce i pattern di lavoro, le difficoltà persistenti e le modalità di evoluzione dell'apprendimento nel tempo.

Sviluppi recenti in questo ambito hanno esplorato l'integrazione di agenti conversazionali basati sull'intelligenza artificiale all'interno delle attività di robotica educativa, con l'obiettivo di fornire feedback immediato e contestualizzato durante lo svolgimento dei compiti, riducendo uno dei principali colli di bottiglia del lavoro didattico: la difficoltà di osservare simultaneamente molti percorsi di apprendimento che si sviluppano in parallelo (Morano et al., 2026). I risultati preliminari di questa sperimentazione mostrano che gli studenti hanno percepito il sistema come utile e che l'uso di agenti conversazionali pedagogicamente allineati, progettati per guidare il ragionamento anziché fornire risposte dirette, può migliorare sia la qualità del feedback percepito sia il coinvolgimento degli studenti nelle attività collaborative (Morano et al., 2026).

I due approcci non si escludono, ma si completano. Questionari, rubriche e osservazioni aiutano a interpretare gli atteggiamenti, le percezioni e i processi. I sistemi di raccolta automatica consentono di ricostruire con maggiore precisione il comportamento operativo degli studenti durante l'attività. La loro integrazione consente di ottenere una visione più ricca dell'apprendimento e di orientare l'intervento didattico in modo più tempestivo ed efficace. Per il docente, questo significa adottare una prospettiva valutativa in cui l'attenzione non si concentra sull'efficienza tecnica dell'artefatto finale, ma sulla qualità del percorso che ha portato alla sua costruzione: riconoscere che l'apprendimento si manifesta anche nei tentativi non riusciti, nelle revisioni del codice, nelle discussioni tra pari e nella progressiva capacità di comprendere il funzionamento di un sistema.

5.6. Riflessioni per l'estensione dei progetti a livelli successivi di scolarità: nella secondaria e in contesti extrascolastici

Il percorso curricolare descritto nelle sezioni precedenti, pur centrato sulla scuola primaria, apre naturalmente verso una sua estensione negli ordini di scuola successivi e verso contesti extrascolastici. La secondaria di primo grado rappresenta un momento di transizione significativo: gli studenti hanno già acquisito i fondamenti della programmazione e dell'interazione con sistemi fisici, e sono pronti ad affrontare una maggiore complessità sia concettuale sia progettuale. In questa fase, la progressione degli strumenti può essere ulteriormente approfondita, consolidando, ad esempio, il passaggio dalla programmazione visuale a blocchi alla programmazione testuale in Python.

Nella secondaria di primo grado e in quella di secondo grado, il curriculum può estendersi verso l'utilizzo di piattaforme robotiche più evolute, come il robot umanoide NAO. NAO offre un contesto particolarmente ricco per l'apprendimento: la sua struttura, dotata di sensori, attuatori, sistemi di riconoscimento vocale e della vista, consente di affrontare in modo concreto temi avanzati quali la percezione ambientale, la pianificazione del movimento e l'interazione uomo-macchina. Dal punto di vista della programmazione, NAO mantiene la stessa dualità già incontrata nelle fasi precedenti del percorso: è possibile programmarlo tramite un ambiente visuale a blocchi (Choregraphe), adatto a chi si avvicina per la prima volta al coding per robotica, oppure tramite Python, per chi intende approfondire la struttura logica dei comportamenti del sistema. Questa continuità non è casuale: riflette una scelta pedagogica precisa, volta a rendere visibile la coerenza tra i diversi strumenti incontrati nel percorso e a consolidare la transizione dal coding come attività espressiva al pensiero sistemico come competenza trasversale.

Nel secondo ciclo, la progressione può spingersi verso concetti di automazione e controllo, tradizionalmente riservati agli istituti tecnici o alle università. Portare questi temi anche in licei e istituti non tecnici richiede strumenti e metodologie capaci di trasmettere concetti fondamentali, come quello di feedback, senza rinunciare alla comprensione, privilegiando l'esperienza concreta rispetto alla formalizzazione matematica. È in questa direzione che si muove il progetto Erasmus+ AutoSTEM, che sperimenta percorsi di questo tipo in istituti non tecnici, strutturando

attività che impiegano simulazione virtuale e interazione con piattaforme robotiche reali.

Questo approccio assume un valore particolarmente rilevante anche in relazione al coinvolgimento delle studentesse nelle discipline STEM. La letteratura evidenzia infatti come il divario di genere in questi ambiti non sia riconducibile a differenze nelle capacità cognitive, bensì a fattori socioculturali, tra cui stereotipi che influenzano la percezione di sé e le aspirazioni professionali delle ragazze, anche a fronte di risultati scolastici analoghi a quelli dei coetanei maschi (Wang & Degol, 2017). In questo contesto, attività didattiche basate su problemi concreti si sono dimostrate efficaci nel contrastare tali dinamiche, contribuendo a ridurre le barriere che ancora limitano la partecipazione femminile ai percorsi scientifici e tecnologici (Atmatzidou & Demetriadis, 2016).

Un ulteriore piano di riflessione riguarda l'estensione della robotica educativa al di là dei contesti scolastici. Le iniziative svolte dalla Fondazione Ospedale Salesi di Ancona offrono in questo senso due esempi significativi di come le tecnologie sviluppate e sperimentate in ambito educativo possano essere reimmaginate per contesti di cura. Il primo riguarda l'impiego di NAO in ambito ospedaliero: il robot viene utilizzato per accompagnare i bambini durante il percorso di cura, interagendo con loro e riducendo l'ansia in un ambiente che può risultare disorientante. La presenza di NAO, capace di comunicare, muoversi e interagire, assume una funzione relazionale, e richiede una progettazione attenta delle modalità di interazione, dei contenuti proposti e dei tempi di coinvolgimento.

Il secondo esempio è il progetto Edutainment4Care, rivolto ai bambini oncologici. Il progetto impiega i componenti della piattaforma M5 per realizzare kit educativi con attività ludico-didattiche STEAM, musicoterapia e giocoterapia, combinando educazione e intrattenimento per offrire ai bambini un'esperienza coinvolgente anche durante il percorso di cura. Questo impiego evidenzia una delle qualità più interessanti delle piattaforme modulari: la loro adattabilità a contesti molto diversi da quelli scolastici per cui sono state originariamente sperimentate, mantenendo al contempo una logica di costruzione e personalizzazione che coinvolge attivamente chi le utilizza.

Queste esperienze invitano a riflettere su una dimensione spesso trascurata nella progettazione curricolare: la robotica educativa non si esaurisce nelle aule scolastiche. Le competenze che si sviluppano attraverso la progettazione e la programmazione di sistemi robotici trovano

applicazione in ambienti e contesti molto più ampi. Portare la robotica in ospedale e farlo con una progettazione pedagogicamente fondata, significa anche mostrare agli studenti, e ai futuri progettisti e tecnici, che la tecnologia può essere uno strumento di cura, di accompagnamento e di umanizzazione degli spazi di vita. Il filo che attraversa tutte queste estensioni, dalla secondaria ai contesti di cura, è lo stesso che ha guidato l'intero capitolo: strumenti, obiettivi e contesto devono restare allineati.

5.7. Conclusioni

La robotica educativa rappresenta oggi un'opportunità significativa per ripensare la didattica in chiave più attiva, laboratoriale e orientata allo sviluppo delle competenze. Il suo valore, tuttavia, non risiede semplicemente nell'introduzione di nuovi strumenti tecnologici in classe, ma nella possibilità di creare contesti di apprendimento in cui gli studenti possano osservare, progettare, sperimentare, verificare e rivedere le proprie soluzioni. In questa prospettiva, la robotica non costituisce un'aggiunta accessoria al percorso scolastico, ma può diventare una componente didattica capace di collegare conoscenze teoriche e pratiche, pensiero e azione, progettazione e riflessione.

Nel corso del capitolo si è evidenziato come, in particolare nella scuola primaria, la robotica educativa possa offrire un contributo rilevante allo sviluppo del pensiero computazionale, del problem solving, della collaborazione e della capacità di affrontare in modo costruttivo l'errore. Affinché questo potenziale possa tradursi in pratiche realmente efficaci, è però necessario superare una visione episodica o extracurricolare della robotica, collocandola invece all'interno di un curriculum progressivo, coerente con l'età degli studenti e con i loro livelli di sviluppo cognitivo. La questione centrale non è dunque introdurre più tecnologia, ma costruire percorsi intenzionali, nei quali attività, strumenti e obiettivi educativi risultino chiaramente allineati.

Questo percorso non si limita alla scuola primaria. La secondaria di primo grado rappresenta un momento di consolidamento e approfondimento, in cui la progressione verso la programmazione testuale e l'uso di piattaforme più evolute, come NAO, consente di affrontare temi più complessi mantenendo la continuità con le esperienze precedenti. Nel se-

condo ciclo, la robotica può aprirsi verso concetti di automazione e controllo, tradizionalmente distanti dai curricula non tecnici, contribuendo al contempo a ridurre il divario di genere nelle STEM attraverso esperienze ancorate alla realtà e significative. Le stesse tecnologie trovano infine applicazione in contesti extrascolastici, come dimostrano le esperienze in ambito ospedaliero, dove la robotica diventa strumento di cura, accompagnamento e benessere.

In questa direzione, anche la scelta degli strumenti assume un significato profondamente pedagogico. Ambienti di programmazione visuale, kit robotici, microcontrollori, sensori e piattaforme modulari non sono semplicemente risorse operative, ma elementi che influenzano il modo in cui gli studenti comprendono il rapporto tra istruzione, comportamento del sistema e interazione con l'ambiente. Per questo motivo, gli strumenti dovrebbero essere selezionati non solo in base alla loro accessibilità o diffusione, ma soprattutto in funzione della loro coerenza con il curriculum e della loro capacità di rendere l'esperienza di apprendimento progressivamente più ricca, trasparente e significativa.

Un'attenzione analoga deve essere riservata alla valutazione. Nei contesti di robotica educativa, infatti, non è sufficiente osservare il funzionamento finale del robot: è necessario considerare anche il processo che ha portato a quel risultato, le strategie adottate, la qualità della progettazione, la capacità di debugging, le revisioni effettuate e il percorso di apprendimento sviluppato dagli studenti. In questo senso, l'integrazione tra strumenti tradizionali e digitali, inclusi i sistemi di *data logging*, apre prospettive particolarmente interessanti, poiché consente di raccogliere evidenze più articolate e di sostenere in modo più tempestivo il lavoro del docente.

Per i docenti, ciò implica non solo acquisire familiarità con strumenti e ambienti di programmazione, ma soprattutto sviluppare una visione progettuale capace di integrare la robotica all'interno di percorsi didattici coerenti, inclusivi e sostenibili.

In conclusione, portare la robotica a scuola non consiste tanto nell'insegnare a usare un robot, quanto nel creare le condizioni affinché gli studenti imparino a comprendere, costruire e trasformare sistemi, idee e soluzioni in modo consapevole, collaborativo e riflessivo. È in questa capacità di connettere tecnologia, apprendimento e progettazione che risiede il suo significato più profondo per la didattica contemporanea.

Riferimenti bibliografici

Alimisis, D., Alimisi, R., Loukatos, D., & Zoulias, E. (2019). Introducing maker movement in educational robotics: Beyond prefabricated robots and "Black Boxes". In *Smart Learning with Educational Robotics: Using Robots to Scaffold Learning Outcomes* (pp. 93–115). Springer.

Atmatzidou, S., & Demetriadis, S. (2016). Advancing students' computational thinking skills through educational robotics: A study on age and gender relevant differences. *Robotics and Autonomous Systems*, 75, 661–670. https://doi.org/10.1016/j.robot.2015.10.008

Di Stasio, M., & Miotti, B. (2021). Perspectives for School: Maker Approach, Educational Technologies and Laboratory Approach, New Learning Spaces. In *Lecture Notes in Networks and Systems*. Springer. https://doi.org/10.1007/978-3-030-77040-2_1

Mangina, E., Psyrra, G., Screpanti, L., & Scaradozzi, D. (2024). Robotics in the Context of Primary and Preschool Education: A Scoping Review. *IEEE Transactions on Learning Technologies*, 17, 342–363. https://doi.org/10.1109/TLT.2023.3266631

Martinez, S. L., & Stager, G. (2019). *Invent to Learn: Making, Tinkering, and Engineering in the Classroom*. Constructing Modern Knowledge Press.

Morano, M., Ifenthaler, D., Cesaretti, L., Di Nardo, F., Screpanti, L., & Scaradozzi, D. (2026). Design and Evaluation of an AI-Based Conversational Agent for Adaptive Feedback in Educational Robotics. *IEEE Transactions on Learning Technologies*, 19, 302-315. https://doi.org/10.1109/TLT.2026.3676621

Papert, S. (1980). *Mindstorms: Children, Computers, and Powerful Ideas*. Basic Books.

Sabri, O. K. (2025). Rethinking Sustainability in Engineering Education: A Call for Systemic Change. *Frontiers in Education*, 10, 1587430.

Scaradozzi, D., Cesaretti, L., Screpanti, L., Costa, D., Zingaretti, S., & Valzano, M. (2019). Innovative Tools for Teaching Marine Robotics, IoT and Control Strategies Since the Primary School. In L. Daniela (A c. di), *Smart Learning with Educational Robotics: Using Robots to Scaffold Learning Outcomes* (pp. 199–227). Springer. https://doi.org/10.1007/978-3-030-19913-5_8

Scaradozzi, D., Cesaretti, L., Screpanti, L., & Mangina, E. (2020). Identification of the Students Learning Process During Education Robotics Activities. *Frontiers in Robotics and AI*. https://doi.org/10.3389/frobt.2020.00021

Scaradozzi, D., Screpanti, L., & Cesaretti, L. (2019). Towards a Definition of Educational Robotics: A Classification of Tools, Experiences and Assessments. In L. Daniela (A c. di), *Smart Learning with Educational Robotics: Using Robots to Scaffold Learning Outcomes* (pp. 63–92). Springer. https://doi.org/10.1007/978-3-030-19913-5_3

Scaradozzi, D., Sorbi, L., Pedale, A., Valzano, M., & Vergine, C. (2015). Teaching Robotics at the Primary School: An Innovative Approach. *Procedia - Social and Behavioral Sciences*, 174, 3838–3846. https://doi.org/10.1016/j.sbspro.2015.01.1122

Tondeur, J., van Braak, J., Ertmer, P. A., & Ottenbreit-Leftwich, A. (2017). Understanding the relationship between teachers' pedagogical beliefs and technology use in education: A systematic review of qualitative evidence. *Educational Technology Research and Development*, 65(3), 555–575. https://doi.org/10.1007/s11423-016-9481-2

Wang, M. T., & Degol, J. L. (2017). Gender gap in science, technology, engineering, and mathematics (STEM): Current knowledge, implications for practice, policy, and future directions. *Educational Psychology Review*, 29(1), 119–140. https://doi.org/10.1007/s10648-015-9355-x

6.
PROGETTARE L'APPRENDIMENTO ATTIVO: UN MODELLO STRUTTURATO PER CODING E ROBOTICA

Giovanni Nulli

6.1. Introduzione: dalla teoria alla struttura operativa

I moduli precedenti hanno fornito le coordinate teoriche e storiche per inquadrare il coding e la robotica come pratiche pedagogiche consapevoli. Questo capitolo compie il passo decisivo verso la prassi didattica, fornendo strumenti concreti per la progettazione, la sperimentazione e la revisione di attività interdisciplinari che coinvolgano questi strumenti. L'obiettivo è supportare il docente – singolarmente o in team collaborativi – in un percorso di progettazione riflessiva, che trasformi l'intuizione pedagogica in un intervento strutturato, documentato e valutabile, in linea con le competenze chiave e il quadro DigComp analizzati nei capitoli precedenti.

Il modello presentato si articola in strumenti di progettazione intenzionalmente ridondanti e circolari. La ridondanza non è un difetto, ma una scelta formativa: in fase di studio e di prima approssimazione, è necessario rallentare il processo decisionale, entrare nel dettaglio e documentare le scelte per acquisire una piena consapevolezza delle dinamiche in gioco (Nulli, Miotti & Di Stasio, 2022). La circolarità, ispirata ai modelli di apprendimento esperienziale e ripresa dalla filosofia del *Lifelong Kindergarten* di Mitch Resnick, riflette la natura iterativa e migliorativa dell'insegnare e dell'apprendere attraverso il fare.

L'approfondimento sui fondamenti metodologici: Il modello si fonda su una solida triade di riferimenti. Il ciclo di apprendimento esperienziale di David Kolb (1984) fornisce la struttura epistemologica di base. Kolb postula che l'apprendimento efficace derivi dall'integrazione di quattro momenti: esperienza concreta (fare), osservazione riflessiva (rivedere ciò che è stato fatto), concettualizzazione astratta (trarre conclusioni e modelli) e sperimentazione attiva (applicare le nuove idee). Il nostro modello traduce questa spirale nelle fasi di *Make* (esperienza), *Improve/Osservazione* (riflessione), *Riprogettazione* (concettualizzazione) e nuova *Progettazione/Think* (sperimentazione). Il lavoro di Pfeiffer e Ballew (1988) sugli *structured ex-*

periences nell'apprendimento degli adulti conferma l'importanza di inserire l'esperienza pratica in una cornice strutturata che ne guidi la riflessione e ne massimizzi il potenziale formativo, evitando che si riduca a un semplice "fare". Infine, la filosofia del creative learning di Mitch Resnick (2018) – sintetizzata nel motto delle 4P: Projects, Passion, Peers, Play – fornisce l'anima pedagogica. Resnick sostiene che l'apprendimento più profondo e creativo avvenga quando gli studenti lavorano su progetti che suscitano la loro passione, in collaborazione con i pari, in un'atmosfera di gioco esplorativo. Il modello qui presentato è un tentativo di strutturare (Kolb, Pfeiffer & Ballew) questo approccio creativo e libero (Resnick) per renderlo praticabile e valutabile nel contesto scolastico quotidiano, senza soffocarne lo spirito innovativo.

La proposta si compone di due percorsi paralleli, declinati per ordine scolastico: uno per studenti più giovani (scuola dell'infanzia e primaria) e uno per studenti più grandi (secondaria di primo e secondo grado). Ciascun percorso è suddiviso in tre moduli consecutivi che corrispondono a tre fasi temporali distinte:

1. Modulo 1 - Progettazione: La riflessione e la pianificazione a tavolino.
2. Modulo 2 - Osservazione e Riflessione: La documentazione e l'analisi di quanto avvenuto in classe dopo la sperimentazione.
3. Modulo 3 - Riprogettazione o Sviluppo: La revisione del progetto originario o l'ideazione di un nuovo sviluppo, basati sulle evidenze emerse.

Questa struttura tripartita obbliga a una pausa riflessiva tra la pianificazione e la revisione, inserendo un indispensabile "bagno di realtà". È un percorso che richiede un impegno iniziale in termini di tempo e di pensiero, ma che si ripaga nel medio periodo, costruendo nel docente una prassi di auto-riflessione sistematica che consente di affinare gli interventi e di risparmiare energie in progettazioni future. I moduli operativi, nella loro versione integrale e pronta per l'uso, sono riportati in Appendice al presente volume.

6.2. Il quadro metodologico: *Problem-Based Learning* e il ciclo *Think-Make-Improve*

Il modello poggia su due pilastri metodologici solidi e complementari. La cornice generale è quella del *Problem-Based Learning* (PBL), preferita, in questa fase introduttiva, al *Project-Based Learning* per la sua maggiore

focalizzazione e gestibilità. Il PBL fornisce l'architettura di base: un problema autentico, sfidante e aperto, posto dal docente come motore per l'indagine e l'acquisizione attiva di conoscenze.

All'interno di questa cornice, si inserisce il ciclo operativo *Think-Make-Improve* (Pensa, Realizza, Migliora), mutuato dalla filosofia maker esposta nel testo *Invent to Learn* (Martinez & Stager, 2013; trad. it. 2020). Questo ciclo fornisce una struttura metodologica chiara sia per il docente che per gli studenti, guidando il processo di risoluzione del problema oltre la tentazione del mero "tentativo ed errore" casuale.

Il ciclo *Think-Make-Improve* (Resnick) è il motore operativo all'interno della cornice PBL. Ecco come si declina concretamente in due esempi per diversi ordini scolastici:

BOX 6.1: Il ciclo *Think-Make-Improve* in azione.

Fase	Scuola primaria (es.: programmare un percorso)	Scuola secondaria (es.: progettare un sistema di monitoraggio ambientale)
THINK (Pensa)	Analisi del problema: "Il nostro robot-apina deve raccogliere polline da tre fiori disposti in angoli diversi del prato (tappeto)." Gli studenti discutono in gruppo: Dove sono i fiori? Quali comandi conosciamo (avanti, indietro, girare)? Disegnano una possibile traiettoria su carta.	Analisi e progettazione: "Dobbiamo monitorare la temperatura e l'umidità dell'orto scolastico e visualizzare i dati." Gli studenti ricercano sensori compatibili con Arduino (DHT11), discutono dove posizionarli, progettano su software (Tinkercad) il circuito e abbozzano lo pseudocodice per la lettura e la trasmissione dati.
MAKE (Realizza)	Costruzione e programmazione: Con il robot (es. Bee-Bot) o in Scratch, traducono il disegno in sequenze di comandi. Testano una parte alla volta ("Va dal fiore 1 al 2?").	Prototipazione: Montano il circuito sulla breadboard, collegano i sensori ad Arduino, scrivono e caricano il codice. Verificano che i dati vengano letti correttamente sul monitor seriale.
IMPROVE (Migliora)	Test e debug: Provano il percorso completo. Se il robot sbaglia angolo, tornano a THINK ("Di quanti gradi deve girare?") e modificano il programma. Confrontano le soluzioni dei diversi gruppi.	Ottimizzazione e test: Il codice funziona ma i dati sono instabili? Tornano a THINK (serve un filtro software? un condensatore nel circuito?). Testano il sistema in condizioni diverse (sole, ombra). Valutano l'efficienza del codice.
Valore aggiunto	Allena il pensiero spaziale e sequenziale, la collaborazione e la resilienza di fronte all'errore.	Sviluppa competenze sistemiche (elettronica, programmazione, analisi dati), problem solving tecnico e progettazione ingegneristica.

Questo ciclo non è lineare ma iterativo. Da "Improve" si può, e spesso si deve, tornare a "Think" per una riprogettazione, in un processo di affinamento continuo che educa alla perseveranza, al pensiero critico e alla metacognizione. È la stessa logica che Resnick ha promosso nella comunità di Scratch, dove la creazione condivisa è un processo permanente di rielaborazione (Resnick, 2018).

6.3. La struttura dei moduli: una guida ragionata

Il cuore operativo del modello risiede nella compilazione dei tre moduli, progettati come strumenti di lavoro e, soprattutto, di studio della propria pratica didattica. La loro compilazione accurata non è burocrazia, ma un esercizio di metacognizione professionale che, se condiviso con un collega, può moltiplicarne il valore formativo.

Il Modulo 1: la progettazione a tavolino

Questo modulo fissa le intenzioni didattiche prima di entrare in classe. La sua compilazione richiede di definire con precisione diversi elementi, la cui articolazione varia leggermente tra il percorso per studenti più giovani e quello per studenti più grandi.

- Percorso per studenti più giovani (es. scuola dell'infanzia e primaria):
  - *Contesto e organizzazione:* Come si organizzerà il lavoro (gruppi, individuale)? Quali sono gli obiettivi disciplinari e di competenza?
  - *Sfondo integratore (facoltativo ma consigliato):* Specialmente per le prime esperienze o per alunni piccoli, è utile calare l'attività in una cornice narrativa (es.: "Il robot-cerotto deve aiutare l'orsetto a raggiungere il miele"). Questo stratagemma aggancia la motivazione e dà senso alle azioni astratte dell'algoritmo.
  - *Consegna (prompt):* È il fulcro della progettazione. Deve essere breve, con parole semplici ma ponderate, esporre un problema autentico ed essere "immune da valutazione" iniziale, nel senso che non deve suggerire o privilegiare una sola soluzione. Deve essere ambigua quanto basta per ammettere soluzioni multiple e creative.
  - *Pianificazione delle fasi:* Si descrive come si prevede che si svolgano concretamente in classe le tre fasi *Think, Make, Improve*.

– Percorso per studenti più grandi (es. secondaria):
    - *Analisi del contesto:* una riflessione più strutturata sul contesto-classe (interessi degli studenti, livello di partenza, risorse della scuola) può fornire spunti preziosi per agganciare la motivazione e tarare la sfida.
    - *Propedeutica tecnica:* con strumenti più complessi (Arduino, LEGO Spike Prime, linguaggi testuali) è spesso necessario prevedere e pianificare moduli di formazione tecnica iniziali su montaggio, interfacce di programmazione o sintassi.
    - *Giustificazione pedagogica:* un esercizio cruciale: scrivere nero su bianco *perché* si sceglie di usare coding/robotica per quel dato argomento disciplinare. Questo step costringe a chiarire il valore aggiunto dello strumento ed evita un uso fine a sé stesso.
    - *Presentazione della metodologia (facoltativa):* Se la classe non è avvezza al PBL o al ciclo *Think-Make-Improve*, può essere utile prevedere un momento di presentazione e patto metodologico.
    - *Consegna (prompt) e pianificazione:* Come per i più piccoli, ma con un'attenzione ancora maggiore alla precisione linguistica. La consegna può essere anche scritta, ma bisogna essere certi che venga letta e compresa per intero – un rischio concreto, come insegnano esperienze sul campo dove gli studenti si fermano alla prima riga. La pianificazione delle tre fasi dovrà essere quanto più dettagliata e realistica possibile.

Il Modulo 2: l'osservazione e la riflessione dopo la sperimentazione

Questo modulo si compila solo dopo aver svolto l'attività in classe. È il momento della verità e dell'apprendimento più profondo per il docente. Segue la stessa struttura del Modulo 1, ma affianca a ogni sezione progettuale due nuove voci:

1. Documentazione (narrazione libera): Cosa è realmente successo in classe per ciascuna fase? Si annotano eventi, dialoghi significativi, difficoltà inattese, reazioni degli studenti.
2. Riflessione (punti di forza/debolezza): In base a quanto osservato, si analizza criticamente la propria progettazione: cosa ha funzionato? Cosa no? Perché? Questa analisi è la base per ogni miglioramento futuro.

La parte più importante del Modulo 2 è spesso la riflessione sulla consegna. Fu efficace? Compresa? Ha generato l'effetto voluto? Dedicare tempo a questa analisi è investire sulla qualità delle future progettazioni. La riflessione finale del modulo deve essere sintetica, ma "incisiva e onesta", e getta le basi per il modulo successivo.

Il Modulo 3: la riprogettazione o lo sviluppo

Questo modulo chiude il cerchio, applicando concretamente il principio "Improve" alla propria didattica. Si riprende la struttura del Modulo 1, ma la si riempie con una progettazione rinnovata, guidata dalle evidenze emerse nel Modulo 2.

- Se l'attività ha incontrato difficoltà, il Modulo 3 diventa uno strumento di riprogettazione mirata: si riscrivono le parti che non hanno funzionato (la consegna, una fase tecnica, l'organizzazione dei gruppi).
- Se l'attività è andata bene, il Modulo 3 diventa uno strumento di sviluppo e approfondimento: si progetta una nuova attività che estende le competenze acquisite, o si immagina come lo stesso progetto possa evolversi in un percorso più ampio (ad esempio, verso il *Project-Based Learning*).

6.4. Affrontare le sfide pratiche: tempo, competenze disparate e valutazione

Un modello teorico solido deve fare i conti con la realtà delle aule. La sua efficacia dipende dalla capacità del docente di anticipare e gestire alcune criticità ricorrenti:

- Gestione del tempo: Le attività di coding e robotica sono per natura time-consuming. La fase di *Make* e soprattutto di *Improve* (debugging) può dilatarsi oltre il previsto. È fondamentale segmentare il progetto in sotto-obiettivi chiari e temporalizzati, e essere pronti, come docente, a "congelare" uno stato di avanzamento per riprenderlo nella lezione successiva, documentando il punto raggiunto.
- Differenze di competenza e interesse: In una stessa classe si possono trovare il "nativo digitale" appassionato e lo studente che non ha

mai toccato un robot. Questo divario rischia di creare frustrazione o, all'opposto, di far lievitare i tempi. Strategie utili includono: formare gruppi eterogenei per competenza (dove il più esperto fa da tutor, ma non risolve tutto), predisporre schede di aiuto differenziate (dalle basi alle sfide avanzate), e prevedere ruoli all'interno del gruppo (programmatore, documentarista, tester).

- Valutazione degli artefatti digitali: Come valutare un programma o un comportamento robotico? La tentazione di valutare solo il prodotto finale (funziona/non funziona) è forte, ma riduttiva. È essenziale valutare anche il processo (documentato nel Modulo 2 di osservazione) e le competenze trasversali. Una rubrica di valutazione potrebbe includere: correttezza algoritmica, efficienza/elegance del codice, capacità di debugging, collaborazione nel gruppo, qualità della documentazione e della presentazione. La valutazione tra pari durante la fase di *Improve* può essere uno strumento formativo potente.

6.5. Conclusioni: un modello per diventare professionisti riflessivi

Il percorso tripartito qui descritto – Progettazione, Osservazione, Riprogettazione – non è un mero schema burocratico. È un dispositivo formativo che guida il docente in un processo di professionalità riflessiva. La sua efficacia è direttamente proporzionale alla precisione e all'onestà con cui si compilano i moduli, soprattutto nella fase di osservazione.

Come evidenziato dalle analisi condotte su sperimentazioni analoghe che hanno coinvolto decine di classi, l'uso sistematico di strumenti progettuali strutturati non garantisce solo una migliore riuscita delle attività, ma accresce la consapevolezza, l'autonomia e la competenza progettuale del docente (Nulli, Miotti & Di Stasio, 2022).

Questo modello operativo rappresenta la traduzione concreta dei principi psicologici discussi nel Capitolo 3. Il ciclo *Think-Make-Improve*, con il suo feedback immediato e il superamento degli ostacoli, è un potente generatore di autoefficacia. La compilazione riflessiva dei moduli, specialmente del Modulo 2, allena la metacognizione professionale del docente, che a sua volta guida quella degli studenti durante il debugging. Infine, l'ancoraggio a problemi autentici e la possibilità di creare artefatti significativi (nelle migliori progettazioni) sono il carburante per la motivazione intrinseca.

Questo capitolo, per ragioni di spazio, ha deliberatamente focalizzato l'attenzione sul *Problem-Based Learning* come porta d'accesso più gestibile, lasciando al *Project-Based Learning* – dove l'idea progettuale nasce o viene co-costruita con gli studenti – il ruolo di evoluzione naturale e più ambiziosa. Allo stesso modo, sono stati solo accennati temi cruciali come le dinamiche del lavoro di gruppo o le sfide dell'interdisciplinarità profonda, che meritano un approfondimento dedicato.

In conclusione, il modello presentato offre una mappa per navigare nella complessità della progettazione con il coding e la robotica. Non sostituisce la creatività e la sensibilità del docente, ma le incanala in una struttura che trasforma l'esperienza, anche quando fallimentare, in apprendimento professionale, ponendo le basi per una didattica sempre più consapevole, efficace e trasformativa.

Riferimenti bibliografici

Kolb, D. A. (1984). *Experiential Learning: experience as the source of learning and development*. Prentice Hall. (Fondamentale per il ciclo dell'apprendimento esperienziale che ispira la struttura circolare del modello).

Libow Martinez, S., & Stager, G. (2020). *Inventando si impara: Apprendere e sperimentare con strumenti e materiali* (L. Guasti, A cura di). Carocci. (Ed. orig. 2013). (Testo manifesto del movimento maker, fonte diretta del ciclo *Think-Make-Improve* e della filosofia del learning-by-making).

Nulli, G., Miotti, B., & Di Stasio, M. (2022). *Robotica educativa e coding: strumenti per la trasformazione del curricolo*. Carocci. (Fonte primaria del modello a tre moduli qui presentato e delle evidenze sulla sua efficacia nella formazione docenti).

Pfeiffer, J. W., & Ballew, A. (1988). *Using Structured Experiences in Human Resource Development* (Vol. 1). University Associates. (Fonte per l'importanza di inquadrare l'esperienza pratica in una struttura che ne guidi la riflessione e l'apprendimento).

Resnick, M. (2018). *Come i bambini. Immagina, crea, gioca e condividi. Coltivare la creatività con il Lifelong Kindergarten del MIT*. Erickson. (Esposizione della filosofia delle 4P e del creative learning, base pedagogica per un approccio costruttivista e giocoso alla tecnologia)

7.
CODING, ROBOTICA EDUCATIVA E DIVARI DI GENERE

Daniela Bagattini

7.1. Definire il problema: la persistenza dei divari di genere nel sistema educativo

L'integrazione del coding e della robotica nel curricolo scolastico non avviene in un vuoto sociale. Essa si inserisce in un contesto educativo caratterizzato da profonde e persistenti asimmetrie di genere, che influenzano percorsi formativi, risultati di apprendimento e, infine, scelte professionali. Affrontare il tema dei divari di genere non è quindi un'appendice opzionale alla formazione su queste tecnologie, ma una premessa necessaria per una loro applicazione didattica consapevole, equa e realmente inclusiva. Questo capitolo si propone di analizzare il problema articolandolo in tre momenti fondamentali: una definizione del fenomeno e della sua rilevanza per la scuola; un'analisi delle cause profonde, radicate nei processi di socializzazione; e, infine, una riflessione su come le metodologie didattiche innovative, come il coding e la robotica, possano diventare strumenti attivi di contrasto a queste disuguaglianze.

La scuola come luogo di (presunta) parità e il risveglio dell'attenzione istituzionale

Fino a non molto tempo fa, la questione del genere nell'istruzione era percepita come un tema di nicchia. Come nota Irene Biemmi, autrice di studi sugli stereotipi nei libri di testo, per decenni il dibattito sul contributo della scuola alla promozione delle pari opportunità è rimasto circoscritto ad ambienti specialistici, forse perché nel sentire comune la scuola stessa è spesso vista come un'istituzione intrinsecamente neutra e garante di parità (Biemmi, 2010). Negli ultimi quindici anni, tuttavia, questo panorama è radicalmente mutato. Tre fattori convergenti hanno portato la questione di genere al centro dell'agenda educativa.

In primo luogo, un rinnovato interesse della ricerca accademica, che ha iniziato a focalizzarsi specificamente sul nesso tra pedagogia e genere, focalizzando l'attenzione non solo sul legame tra genere e sfera istituzionale e familiare, ma anche tra questo e il mondo educativo.

Il secondo fattore, di natura socio-politica, è la spinta dei movimenti femministi che, a partire dagli anni Novanta, hanno portato la violenza di genere dalla sfera del "privato" a quella del "problema pubblico". Questo percorso, culminato a livello internazionale con la Convenzione di Istanbul del 2011, ha introdotto un approccio sistemico al contrasto della violenza. La Convenzione riconosce esplicitamente il legame tra violenza, disuguaglianze sociali e stereotipi di genere, e assegna al mondo dell'educazione un ruolo strategico e proattivo nella loro prevenzione. Da questo quadro sono derivate, anche in Italia, linee guida e indicazioni ministeriali (si veda, ad esempio, il documento "Linee guida per l'educazione al rispetto") che impegnano la scuola a integrare questi temi nella didattica ordinaria, sebbene spesso in forma di principi generali che attendono ancora piena traduzione operativa.

Il terzo fattore, di natura economico-strategica, è la crescente preoccupazione per gli effetti della segregazione orizzontale. Questo termine descrive il fenomeno per cui uomini e donne si concentrano in percorsi formativi e professionali diversi. La sotto-rappresentazione femminile nei settori scientifico-tecnologici (STEM e ICT) è vista non solo come una questione di equità, ma come uno spreco di capitale umano e un freno all'innovazione e alla competitività dei sistemi-paese, motivo per cui è diventata una priorità anche per istituzioni come l'Unione Europea.

I dati della segregazione: performance, scelte e opportunità mancate

La dimensione del problema è misurabile attraverso dati inequivocabili, che mostrano come le disuguaglianze agiscano già nei risultati di apprendimento per poi strutturarsi in scelte formative divergenti. Le rilevazioni INVALSI, come evidenziato anche nelle "Linee guida per le discipline STEM", continuano a registrare un marcato *gender gap* in matematica a svantaggio delle studentesse. I dati più recenti (2023) mostrano un preoccupante peggioramento di questo divario. La situazione italiana è particolarmente critica in un contesto internazionale: il rapporto OCSE PISA 2022 (pubblicato a dicembre 2023) posiziona l'Italia come il paese con il più ampio divario di genere in matematica tra tutti quelli esaminati, con i ragazzi che ottengono mediamente oltre 20 punti in più delle ragazze. È

cruciale notare, tuttavia, che questo gap non è una costante universale: la sua entità varia tra i paesi occidentali e, significativamente, in alcuni paesi in via di sviluppo il vantaggio è a favore delle ragazze. Questo dato suggerisce fortemente che le differenze non sono di origine biologica, ma profondamente radicate in fattori socio-culturali e pedagogici.

Queste differenze nelle performance si traducono in una segregazione delle scelte formative. I dati ministeriali mostrano una rappresentazione ancora drammaticamente squilibrata: le studentesse che frequentano gli istituti tecnici a indirizzo tecnologico sono una percentuale esigua, mentre settori come la sezione coreutica del Liceo musicale e coreutico sono pressoché a dominanza femminile. Questo schema si riproduce fedelmente a livello universitario, dove i corsi di studio in ingegneria e informatica rimangono a fortissima prevalenza maschile.

Le conseguenze di questa segregazione vanno oltre il mero squilibrio numerico. Esse rappresentano uno spreco di potenziale innovativo. La storia della tecnologia è infatti costellata di contributi fondamentali di donne, spesso nati proprio dall'esperienza vissuta e da una diversa prospettiva sul mondo. L'invenzione della lavastoviglie (Josephine Cochrane), i contributi alla teoria delle frequenze radio alla base del WiFi (Hedy Lamarr e altri), o gli algoritmi che hanno permesso lo sviluppo del GPS (Gladys West) ne sono esempi emblematici. Escludere le donne dai processi di progettazione tecnologica significa privarsi di un fondamentale punto di vista, limitando la capacità della tecnologia stessa di rispondere ai bisogni dell'intera società.

Il cuore pedagogico della questione: la costruzione dell'identità e delle possibilità

Al di là delle considerazioni economiche e sociali, esiste una ragione pedagogica profonda per intervenire sui divari di genere. La scuola ha il compito di sostenere bambini, bambine, ragazzi e ragazze nella costruzione della propria identità e del proprio progetto di vita. L'orientamento, in questa prospettiva, è un processo continuo di scoperta di sé e delle proprie potenzialità. La persistenza di percorsi così rigidamente settorializzati lungo linee di genere, mostra come questo processo sia come limitato da barriere spesso invisibili, che impediscono di portare a pieno sviluppo le potenzialità di tutte le studentesse e gli studenti, condizionando le loro aspirazioni. Comprendere le radici di questo fenomeno – il "perché succede" – è il passo necessario per progettare un'azione didatti-

ca che non si limiti a registrare le differenze, ma che si ponga l'obiettivo ambizioso di rimuovere gli ostacoli che impediscono quel pieno sviluppo della personalità già esplicitato nel dettato costituzionale.

7.2. Alle radici del fenomeno: socializzazione di genere, stereotipi e il ruolo della scuola

Se la prima parte ha fotografato la persistenza dei divari di genere nel sistema educativo, è ora necessario spostare l'analisi alle sue cause profonde. Perché, nonostante l'evidenza dei dati e una crescente consapevolezza istituzionale, ragazzi e ragazze continuano a percepire e a scegliere percorsi formativi così divergenti? La risposta non risiede in una presunta diversa "natura" o predisposizione biologica, ma in un complesso processo culturale e psicologico che plasma le identità e le aspirazioni fin dalla prima infanzia: la socializzazione di genere.

Smontare il mito della "predisposizione naturale"

Un'idea particolarmente resistente, anche tra gli educatori, è che le differenze nelle scelte e nei rendimenti siano riconducibili a una diversa inclinazione "naturale" tra maschi e femmine. Un'indagine condotta recentemente tra docenti coinvolti in progetti di coding e robotica ha rivelato che quasi il 40% degli intervistati ritiene che la "diversa predisposizione tra maschi e femmine" influenzi in modo significativo le scelte formative. Questa percezione, sebbene diffusa, è smentita da più fronti. Le neuroscienze non evidenziano differenze cognitive innate che giustifichino un diverso potenziale nell'apprendimento della matematica o delle discipline tecnico-scientifiche. I dati, come quelli ISTAT elaborati per Save the Children, mostrano anzi che le ragazze possiedono spesso competenze digitali molto solide. L'argomento più decisivo, tuttavia, viene dalla comparazione internazionale e storica: il fatto che il *gender gap* in matematica vari enormemente da paese a paese (con casi in cui è addirittura rovesciato a favore delle ragazze), e che professioni oggi percepite come "femminili" (come l'insegnamento) o "maschili" (come la programmazione informatica) abbiano avuto in passato una composizione di genere opposta, dimostra che non siamo di fronte a determinismi biologici, ma a costruzioni sociali mutevoli. Questo paradosso storico-sociale – per cui

quando un settore professionale acquisisce prestigio e redditività tende a diventare dominio maschile – indica che le dinamiche in gioco sono di potere, status e rappresentazione culturale, non di attitudine.

Il meccanismo chiave: socializzazione, auto-efficacia e "gabbie di genere"

Se non è la biologia, cosa spiega allora i diversi rendimenti e, soprattutto, le diverse autopercezioni? La letteratura psicopedagogica individua nel concetto di auto-efficacia – la convinzione che un individuo ha riguardo alle proprie capacità di organizzare ed eseguire le azioni necessarie per gestire situazioni specifiche. Questa convinzione non nasce dal nulla, ma è fortemente condizionata dal processo di socializzazione, attraverso il quale interiorizziamo aspetti della nostra cultura fino a percepirli come naturali e immutabili. L'esempio del cibo è illuminante: l'uso delle bacchette per mangiare il riso piuttosto che di una forchetta per arrotolare gli spaghetti, sono abilità apprese che, una volta interiorizzate, sembrano spontanee. Allo stesso modo, la preferenza per una colazione dolce o salata è un prodotto culturale. Ciò che per noi è "normale" è spesso il risultato di un apprendimento invisibile e pervasivo.

Questo avviene anche per quanto riguarda i ruoli di genere. Fin dalla primissima infanzia, attraverso le principali agenzie di socializzazione (famiglia, gruppo dei pari, scuola, media, linguaggio), apprendiamo in modo implicito ed esplicito cosa è "adeguato" al maschile e al femminile. Questo avviene attraverso una costante interazione di imitazione, insegnamento e rinforzo sociale: frasi come "non fare la femminuccia", "piangere è da femmine", "sei un maschiaccio" o, all'opposto, "che bella principessa", "sei proprio una brava ragazzina", sono micro-segnali che delimitano i confini dei comportamenti accettabili. La bambina che gioca prevalentemente con bambole e set per la cura interiorizzerà quella come l'area "normale" del suo interesse; il bambino dissuaso da quegli stessi giochi apprenderà che non sono per lui. Connell definisce questo sistema l'"ordine sociale di genere", un principio organizzatore così radicato da diventare una delle lenti principali attraverso cui diamo senso al mondo. Questo condizionamento influenza profondamente l'apprendimento: una ragazza può arrivare a pensare che "è normale non essere brava in matematica" o "non capire la tecnologia", non per una mancanza di capacità, ma perché la sua socializzazione non ha mai validato o incoraggiato quelle competenze come pertinenti alla sua identità femminile. Irene Biemmi parla, a proposito, di "gabbie di genere": strutture culturali invisibili che limitano

lo sviluppo pieno della personalità, ingabbiando bambini e bambine in aspettative predefinite. Il problema non è che una ragazza ami le professioni di cura, ma che possa sentirsi libera di *scegliere*, avendo avuto accesso a un immaginario più ampio delle proprie possibilità.

La scuola tra riproduzione e possibilità di cambiamento

Consapevoli di questo meccanismo, è doveroso chiedersi quale ruolo giochi la scuola. Purtroppo, spesso la scuola stessa, in modo non intenzionale, agisce come agenzia di riproduzione degli stereotipi. Questo avviene attraverso molteplici canali:

1. Gli oggetti culturali: L'analisi pionieristica di Irene Biemmi sui libri di testo della scuola primaria ha rivelato una rappresentazione fortemente stereotipata: le bambine erano ritratte prevalentemente in ambienti chiusi, con aggettivi legati all'estetica e alla cura; i bambini, al contrario, erano rappresentati come attivi, coraggiosi e in spazi aperti.
2. Il curricolo nascosto: I valori, i modelli educativi, le aspettative differenziate (anche inconsce) degli insegnanti, il linguaggio utilizzato, le modalità comunicative, costituiscono un potente messaggio implicito.
3. L'organizzazione degli spazi: Nella scuola dell'infanzia e primaria, la suddivisione degli angoli-gioco (angolo della casa vs. angolo delle costruzioni) può rinforzare la separazione di genere.
4. I simboli e i colori: L'associazione quasi automatica del rosa al femminile e dell'azzurro al maschile è un costrutto culturale recente (in passato, l'azzurro era spesso associato alla Vergine Maria) che la scuola, attraverso decorazioni, materiali o semplici convenzioni, può tacitamente avallare.

Riconoscere che la scuola può essere parte del problema non è un atto di accusa, ma il presupposto necessario per un cambiamento consapevole. Se gli stereotipi si apprendono, significa che si possono anche disimparare e decostruire. La domanda cruciale diventa allora: come può la scuola, e in particolare l'utilizzo di metodologie didattiche innovative come il coding e la robotica, trasformarsi da potenziale agente di riproduzione in uno strumento attivo per allargare gli orizzonti del possibile e promuovere un'autoefficacia equamente distribuita? È a questa sfida progettuale che risponderà la parte conclusiva di questo capitolo.

7.3. Strategie didattiche: come coding e robotica possono promuovere l'equità di genere

Dopo aver analizzato le cause culturali e strutturali dei divari di genere, è giunto il momento di esplorare la dimensione operativa: cosa può fare concretamente la scuola, e in che modo metodologie come il coding e la robotica possono trasformarsi da potenziali ambiti di segregazione in strumenti attivi di empowerment e di allargamento degli orizzonti del possibile per tutte le studentesse e tutti gli studenti. Le indicazioni internazionali (UNESCO, 2017; OECD, 2017) e nazionali (MIUR, 2022 - Linee guida STEM; PNRR) convergono nel chiedere alla scuola un'azione decisa. La sfida non è semplicemente attrarre più ragazze verso le STEM, ma decostruire gli stereotipi che ne limitano la scelta, lavorando sull'autoefficacia e sulla percezione di sé.

Il potere della didattica laboratoriale: dalla paura alla scoperta di competenze

La domanda di partenza è se una didattica laboratoriale, basata sul fare, sul problem solving e sulla valorizzazione dell'errore come tappa del processo, possa aiutare a superare l'ansia e le "gabbie di genere" (Biemmi, 2017). Un'esperienza di ricerca sperimentale condotta dal nostro gruppo su progetti di coding e robotica offre una risposta incoraggiante. Inizialmente, il progetto non aveva tra gli obiettivi espliciti quello di lavorare sulle tematiche di genere. Solo a conclusione del progetto è stato chiesto ai docenti coinvolti di riflettere sulle dinamiche osservate. Le risposte sono state illuminanti: accanto a chi non aveva notato differenze o le aveva attribuite a "predisposizioni naturali", un gruppo significativo di insegnanti ha riportato osservazioni profondamente in linea con la letteratura sull'autoefficacia. Una di queste citazioni è particolarmente significativa:

«Non sono state trovate differenze di approccio tra ragazzi e ragazze. Tutti hanno partecipato con entusiasmo, anche se le ragazze più di una volta si sono stupite per la facilità con la quale apprendevano certi argomenti nuovi o trovavano velocemente le soluzioni alle sfide. Alcune ragazze hanno evidenziato delle competenze che neanche loro pensavano di avere all'inizio del percorso» (Docente, progetto sperimentale).

Questo è il cuore della questione: "competenze che avevano e pensavano di non avere". L'esperienza pratica, concretizzata in un artefatto (un programma, un robot), fornisce un riscontro oggettivo e immediato delle proprie capacità, aggirando la barriera della sfiducia interiorizzata. Un parallelo potente viene dall'esperienza collettiva della didattica a distanza durante la pandemia: molte madri, che precedentemente si dichiaravano impaurite dalla tecnologia, si sono trovate a doverla padroneggiare per supportare i figli e le figlie, scoprendo inaspettate risorse. Significativamente, un'indagine condotta nel 2020 su operatrici di centri antiviolenza – professioniste abituate a lavorare sull'empowerment di altre donne – ha rivelato che molte di loro hanno vissuto lo stesso percorso: dall'ansia iniziale verso gli strumenti digitali alla scoperta di poterli utilizzare efficacemente. Questo dimostra come il fenomeno della bassa autoefficacia tecnologica colpisca trasversalmente, perfino chi si occupa professionalmente di disinnescare stereotipi.

Elementi chiave per una progettazione didattica sensibile al genere

Introdurre il coding e la robotica non può essere visto come una bacchetta magica contro gli stereotipi. Perché queste metodologie siano efficaci anche sotto questi aspetti, è necessario un approccio progettuale consapevole. Dalla nostra ricerca e dalla letteratura internazionale emergono alcuni principi operativi fondamentali:

1. Consapevolezza e "sguardo di genere": Il primo passo è che il docente sviluppi una consapevolezza critica dei meccanismi di socializzazione e delle proprie aspettative implicite. Porre a sé stessi domande come: "Le difficoltà che osservo nelle ragazze sono innate o il risultato di una minore esposizione a certi giochi e linguaggi?" è essenziale. Questo "sguardo di genere" guida l'osservazione in itinere e l'adattamento della proposta didattica.
2. Formazione dei gruppi con sensibilità: Non esiste una ricetta universale (gruppi separati per genere o misti). La scelta deve essere guidata dall'obiettivo e dalle dinamiche relazionali. È cruciale evitare di accoppiare una ragazza insicura rispetto alla tecnologia con un ragazzo già molto esperto e competitivo, situazione che rischia di farla "retrarre". L'obiettivo è creare setting in cui tutte le voci possano emergere.

3. Modelli di riferimento accessibili e plurali: Presentare role model femminili è importante, ma con attenzione. Proporre solo modelli di eccellenza irraggiungibili può involontariamente rinforzare l'idea che "si può fare solo se si è straordinarie". È utile affiancare a questi esempi modelli più prossimi e diversificati: ricercatrici, ingegnere, programmatrici del territorio, che mostrino percorsi possibili e sfaccettati, senza necessariamente essere "eccezionali".
4. Promozione di temi neutri e prevenzione dell'antropomorfizzazione stereotipata: Per attrarre le ragazze, a volte si ricorre a espedienti come "mettere la gonnellina al robot", come ci ha dichiarato un docente. Questa strategia, però, non scalfisce lo stereotipo, anzi, può rischiare di riprodurlo. È più efficace scegliere temi neutri (ad esempio legati al mondo animale) che interessino trasversalmente la classe, presentando la tecnologia come uno strumento per rispondere a sfide significative e condivise.
5. Valorizzazione dell'errore e del processo: Il coding e la robotica sono per loro natura "discipline del debugging", dove l'errore è un'occasione di apprendimento sistematico. Creare un clima in cui sbagliare, provare, rivedere è la norma – e non un fallimento – è particolarmente liberatorio per chi ha interiorizzato la paura di non essere all'altezza. Il rinforzo positivo sul processo (l'impegno, la strategia) più che sul solo risultato, sostiene l'autoefficacia.
6. Integrazione interdisciplinare e curricolo verticale: Per evitare di presentare il coding come una "cosa da tecnici", è fondamentale integrarlo in progetti interdisciplinari (ad esempio, creare una storia animata per italiano, simulare un fenomeno storico o scientifico). Questo ne smorza la connotazione puramente "tecnico-maschile". Inoltre, un approccio verticale, che proponga attività progressive e coerenti lungo tutto il ciclo di studi, permette a ragazzi e ragazze di costruire familiarità e competenza in modo graduale, normalizzando questi linguaggi come parte del bagaglio culturale di tutti (Nulli, Miotti & Bagattini, 2024).

Conclusione: verso una scuola che libera i talenti

La scuola ha quindi un potere e una responsabilità enormi nel contrastare i divari di genere. Il coding e la robotica, se progettati con una lente critica e inclusiva, non sono fini a sé stessi, ma diventano privi-

legiati "terreni di gioco" per mettere in pratica questo cambiamento. Consentono di lavorare concretamente su quelle che Bandura definisce le fonti dell'autoefficacia: l'esperienza diretta di successo ("ce l'ho fatta!"), l'osservazione di modelli vicini, la persuasione verbale incoraggiante del docente e il controllo degli stati emotivi (superare l'ansia attraverso il fare).

L'obiettivo finale non è produrre per forza più programmatrici, ma restituire a ogni studente e studentessa la piena libertà di esplorare le proprie attitudini, senza che queste siano predefinite da stereotipi culturali. Significa passare da una logica di *adattamento* a una cultura delle *possibilità*, in cui la tecnologia è uno strumento di espressione, creatività e cittadinanza attiva, equamente a disposizione di tutti e di tutte. In questo senso, un uso pedagogico consapevole del coding e della robotica rappresenta una risposta concreta all'appello della Convenzione di Istanbul e delle Linee guida nazionali, contribuendo a costruire, fin dai banchi di scuola, una società più equa, innovativa e libera dai pregiudizi.

Riferimenti bibliografici

Bagattini, D., & Miotti, B. (2022). *Lavorare sul genere con coding e robotica educativa*. Carocci.

Biemmi, I. (2010). *Educazione sessista. Stereotipi di genere nei libri delle elementari*. Rosenberg & Sellier.

Biemmi, I., & Leonelli, S. (2017). *Gabbie di genere. Retaggi sessisti e scelte formative*. Rosenberg & Sellier.

Colella, P. (2014). Libere e liberi di scegliere? Prospettive di genere nella didattica della matematica e della fisica. In M. S. Sapegno (a cura di), *La differenza insegna. La didattica delle discipline in una prospettiva di genere* (pp. 139-150). Carocci.

Consiglio d'Europa (2011). *Convenzione del Consiglio d'Europa sulla prevenzione e la lotta alla violenza contro le donne e la violenza domestica* (Convenzione di Istanbul).

De Marchi, V. (a cura di) (2020). *Con gli occhi delle bambine. Atlante dell'infanzia a rischio 2020*. Save the Children Italia Onlus.

MIUR – Ministero dell'Istruzione, dell'Università e della Ricerca. (2017). *Linee guida per l'educazione al rispetto*.

MIUR – Ministero dell'Istruzione, dell'Università e della Ricerca. (2022). *Linee guida per l'orientamento alle discipline STEM*.

Nulli, G., Miotti, B., & Bagattini, D. (2024). Designing a vertical curriculum for coding and robotics: the tree model. *QTIMES, Anno XVI, Numero 3*.

OECD (2017). *The Pursuit of Gender Equality: An Uphill Battle*. OECD Publishing.

OECD (2022). *PISA 2022 Results (Volume I): The State of Learning and Equity in Education*. OECD Publishing.

UNESCO (2017). *Cracking the code: Girls' and women's education in science, technology, engineering and mathematics (STEM)*.

8.
CONCLUSIONI: VERSO UNA DIDATTICA CONSAPEVOLE DEL PENSIERO COMPUTAZIONALE

Giovanni Nulli, Daniela Bagattini

8.1. Dal "che cosa" al "perché": riprendendo il filo del discorso

Quando, nell'introduzione di questo volume, ci siamo chiesti perché il pensiero computazionale, il coding e la robotica educativa faticassero a trovare un'integrazione stabile e diffusa nella didattica curricolare, abbiamo avanzato un'ipotesi che ora, al termine di questo percorso, possiamo riprendere con maggiore consapevolezza. L'ipotesi era che la formazione, tanto dei docenti quanto dei formatori, viene prevalentemente interpretata come un'acquisizione di competenze tecniche necessarie per l'utilizzo delle tecnologie. Questa prospettiva, a nostro avviso, ha generato un equivoco di fondo: gli insegnanti hanno spesso inteso l'apprendimento degli aspetti tecnici come un "male necessario" per accedere ad attività ad alto valore motivazionale; i formatori, dal canto loro, hanno talvolta cercato di semplificare eccessivamente la complessità, minimizzando le difficoltà tecniche per non scoraggiare i corsisti.

Le conseguenze di questo equivoco sono state, e in parte continuano a essere, rilevanti. In primo luogo, si è diffusa la convinzione che il valore motivazionale aggiunto risieda nell'oggetto stesso – il robot, l'app, il simulatore – e non nell'attività che con quell'oggetto si compie. Ma chiunque abbia osservato una classe dopo le prime ore di entusiasmo sa che la novità tecnologica ha una durata limitata: il robot che oggi affascina, tra un mese sarà dato per scontato. Ciò che mantiene vivo l'interesse non è l'oggetto, ma il il processo, il lavoro che si svolge che con esso si crea e si sviluppa, la sfida cognitiva che si propone e, come motore pedagogico, il senso che l'attività assume nel quadro più ampio degli apprendimenti disciplinari e trasversali.

In secondo luogo, la semplificazione degli aspetti tecnici ha rischiato di occultare proprio ciò che rende il coding e la robotica educativamente potenti: la conoscenza del funzionamento dello strumento. Un robot non

è una scatola nera magica, né un semplice esecutore di comandi: è un sistema complesso in cui convergono aspetti meccanici, elettronici, informatici e logico-algoritmici. Comprenderne il funzionamento significa entrare in contatto con concetti fondamentali – il sensore che trasforma una grandezza fisica in un dato digitale, l'attuatore che traduce un comando in un movimento, l'algoritmo che governa il comportamento – che sono a loro volta porte d'accesso a saperi disciplinari (fisica, matematica, informatica) e a competenze trasversali (problem solving, pensiero critico, progettualità).

Questo volume ha quindi inteso percorrere una strada diversa: non quella della semplificazione, ma quella della restituzione di complessità. Abbiamo scelto di presentare ai docenti non solo le pratiche, ma i quadri di senso che le fondano; non solo le procedure, ma i perché pedagogici, scientifici e giuridici che le sostengono. E lo abbiamo fatto nella convinzione che solo un docente consapevole – capace di orientarsi tra definizioni, di distinguere i diversi piani di analisi, di collegare la tecnologia ai saperi – possa progettare percorsi significativi.

8.2. Suggerimenti pratici per i docenti

Primo suggerimento: la specificità del pensiero computazionale

Nonostante le numerose definizioni, riduzioni, etc. è utile che il docente abbia chiaro come si stia muovendo ogni volta che usa il termine "pensiero computazionale" e ogni volta che il termine viene utilizzato da un collega o da un formatore. Affinché sviluppi una professionalità coerente è necessario comprendere l'ambito in cui ci si sta muovendo, evitando incongruenze e forzature

Secondo suggerimento: l'integrazione curricolare

Coding e robotica sono ancora spesso confinati in momenti extracurricolari – laboratori pomeridiani, progetti speciali, "giornate del coding". L'obiettivo dovrebbe essere la loro integrazione stabile nel curricolo, come parte ordinaria della didattica disciplinare. Ciò richiede non solo formazione dei docenti, ma, in un'ottica di autonomia scolastica, anche una revisione dei curricoli, una progettazione collegiale, un'organizzazione flessibile degli spazi e dei tempi.

Terzo suggerimento: la valutazione

Come si valuta il pensiero computazionale? Come si valutano le competenze digitali? Non disponiamo ancora di strumenti consolidati e condivisi. Il DigComp 3.0, con i suoi ***learning outcomes***, offre una base interessante, che però richiede un lavoro di riflessione da parte del docente affinché possa arrivare ad una traduzione in pratiche educative (rubriche, osservazioni, compiti di realtà, portfolio) che sia pertinente e adatta al proprio modo di insegnare.

Quarto suggerimento: l'Intelligenza Artificiale a scuola

L'IA generativa viene comunemente utilizzata nella comunicazione quotidiana e sta diventando un asset strategico nel lavoro, quindi non è più possibile ignorarla. La domanda non è "se" utilizzarla, ma "come" farlo in modo critico, etico e pedagogicamente fondato. Il pensiero computazionale e la robotica educativa offrono un quadro di riferimento utile, perché abituano a pensare la macchina come un sistema che opera su istruzioni e dati, con potenzialità e limiti. E l'integrazione di questi temi nel curricolo può diventare un veicolo per l'introduzione consapevole dell'IA,

Quinto suggerimento: la ricerca sulla formazione dei docenti

Abbiamo progettato una formazione con determinate caratteristiche – modulare, esperienziale, riflessiva, basata sulla produzione di elaborati e sulla revisione tra pari. Quale può essere il rapporto tra questo modello ed una formazione più squisitamente tecnica? A nostro parere un approccio, come già dichiarato, che abbraccia la complessità aiuta a contestualizzare la pluralità delle tecnologie e dei linguaggi di programmazione, e di conseguenza a dare senso alla molteplicità delle formazioni tecniche che è possibile trovare.

8.3. Un invito, non un punto di arrivo

Questo volume non intende proporsi come un punto di arrivo, ma come un invito a prendere sul serio il pensiero computazionale e la robotica educativa non come mode passeggere, ma come componenti es-

senziali di una didattica capace di rispondere alle sfide del XXI secolo. È un invito rivolto ai docenti, ai formatori, ai dirigenti scolastici, ai decisori politici.

Per i docenti la sfida è quella di non accontentarsi di ricette preconfezionate: la complessità della pratica docente deve essere affrontata con gli strumenti della riflessione, dello studio, della ricerca, per poter acquisire la consapevolezza necessaria a andare oltre il kit o l'esempio pratico.

Occorre guardare alla formazione senza cedere alla tentazione della semplificazione: un suggerimento ai formatori, che sappiano affrontare la sfida della complessità.

Anche dirigenti scolastici e decisori politici sono al centro di questa sfida: è importante che sia chiaro come la formazione dei docenti sia un investimento in "competenze tecniche", ma in capacità professionale. Un docente che sa progettare, sperimentare, riflettere, riprogettare è un docente che produce valore per l'intero sistema.

La ricerca che accompagna questo volume continuerà. I docenti che vorranno contribuire saranno i nostri interlocutori privilegiati. Perché la scuola non si cambia dall'alto, ma dal basso – con il lavoro quotidiano, paziente, riflessivo di chi ogni giorno sta in classe.

E con questo, lasciamo la parola all'esperienza di chi vorrà mettersi in gioco.

INDICE

Edizioni ETS
Palazzo Roncioni - Lungarno Mediceo, 16, I-56127 Pisa
info@edizionets.com - www.edizionets.com
Finito di stampare nel mese di giugno 2026