Rock around Sonic Pi

Sam Aaron, il live coding e l'educazione musicale

Roberto Agostini

IC9 Bologna, Servizio Marconi TSI (USR-ER)

1. Formazione "Hands-On" con Sam Aaron

"Imagine if the only allowed use of reading and writing was to make legal documents. Would that be a nice world? Now, today's programmers are all like that"[1]. In questa immagine enigmatica e un po' provocatoria proposta da Sam Aaron nelle prime fasi di una sua recente conferenza è racchiusa un'originale visione della programmazione ricca di implicazioni, e non solo per l'educazione musicale.

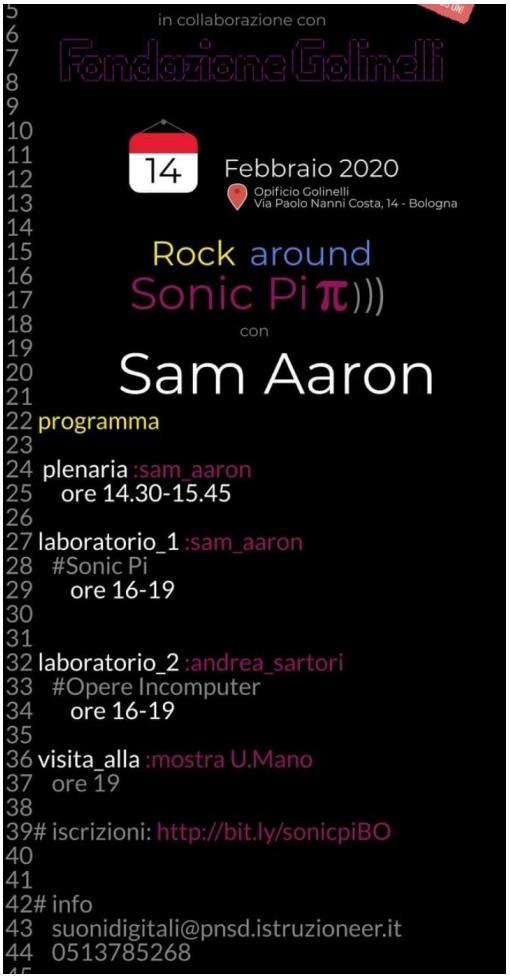
Ma andiamo con ordine: il 14 febbraio 2020, all'Opificio Golinelli di Bologna, Sam Aaron, creatore del software per *live coding* musicale Sonic Pi, è stato protagonista di un denso pomeriggio dedicato al pensiero computazionale nella musica intitolato "Rock around Sonic Pi". L'iniziativa rientrava nell'ambito della "Formazione Hands-On" promossa dal Future Lab dell'Istituto Comprensivo di Ozzano dell'Emilia in collaborazione con il Servizio Marconi TSI dell'USR-ER. Oltre alla menzionata conferenza, aperta a tutti, Sam Aaron ha tenuto un laboratorio di tre ore a numero chiuso per i docenti delle scuole[2].

- Presentazione e programma dettagliato della giornata "Rock Around Sonic Pi" (presentazione a cura di Rosa Maria Caffio).
- Ripresa integrale della conferenza di Sam Aaron (20.2.2020) (registrazione a cura dello staff dell'Opificio Golinelli).

[1] "Immaginate che l'unico modo consentito di usare la lettura e la scrittura fosse quello di produrre documenti legali. Sarebbe un bel mondo? Ora, i programmatori di oggi sono nella stessa situazione".

[2] Sam Aaron si è fermato a Bologna due giorni, durante i quali, oltre a partecipare alle attività del citato pomeriggio, ha tenuto anche un concerto ed è stato intervistato da due emittenti radiofoniche. Le interviste e qualche testimonianza fotografica della permanenza di Sam Aaron a Bologna possono essere consultate in Agostini 2020.

2. Il diritto di programmare



Seguire la conferenza di Sam è stato facile e gradevole, ma anche particolarmente impegnativo. Infatti, se l'eloquio entusiasta di Sam ha contagiato gli ascoltatori permettendo loro di sintonizzarsi agevolmente con il flusso del suo pensiero, i contenuti della conferenza non sono stati né semplici né banali.

Musicista e programmatore, nella figura di Sam Aaron opposizioni diffuse nel sentire comune come quelle tra arte e tecnica, espressività estetica e pensiero scientifico o immaginazione e ragione, si dissolvono per restituire a questi ambiti del sapere e della conoscenza tutta la loro complessità. Al centro di tutto, a far da ponte, i concetti di linguaggio e scrittura.

Torniamo alle parole con cui ho aperto questo articolo: cosa voleva dirci esattamente Sam con quella frase sibillina sul linguaggio e sui documenti legali? Sam ha ripreso la stessa metafora dei più avanti nella sua conferenza: "[Programming] is not just a tools for solving complicated equations. It's also a tool for as humans to express ourselves, just like writing and reading is not just a tool for lawyers and legal documents. It's also a tool for poems, for planes, for rap lyrics, for shopping lists, for lot of different things allow us to be the people that we are"[3]. In altri termini, il pensiero computazionale è troppo spesso considerato materia per una nicchia di specialisti. Di fatto, "most programmers who take these crazy computers and make them do things they mostly work for industry making rich people richer"[4] e tutto si ferma a lì.

Sam propone una visione diversa: per lui la programmazione dovrebbe essere considerata una competenza comune che deve essere diffusa nella società. In lui riecheggia l'idea di Seymour Papert che "[p]rogramming a computer means nothing more or less than communicating to it in a language that it and the human user can both 'understand'" (Papert 1993, 5, 6)[5]. Papert è colui che nel 1980 introduce l'espressione 'pensiero computazionale'. Prevedendo che un giorno i bambini sarebbero cresciuti circondati da computer, egli

[3] "[Programmare] non è solo uno strumento risolvere complicate equazioni. E' anche uno strumento per esprimere se stessi in quanto umani, così come la scrittura e la lettura non sono solo strumenti per avvocati e documenti legali, ma sono anche strumenti per le poesie, per gli aerei, per i testi rap, per le liste della spesa e per molte cose diverse che ci permettono di essere le persone che siamo."

[4] "La maggior parte dei programmatori che prendono e che fanno fare cose a questi pazzi computer lavorano perlopiù per l'industria rendendo le persone ricche più ricche".

[5] "Programmare un computer significa né più né meno che comunicare con quest'ultimo in un linguaggio che sia esso che l'utente umano possano 'capire'". Un ringraziamento ad Alessandra "Teacher Alle" Serra per avermi fatto conoscere e apprezzare il pensiero di Seymour Papert.

comincia a riflettere "how computers may affect the way people think and learn"[6] (*ibid.*, 3) e a sperimentare il digitale come nuovo mezzo per l'apprendimento. Integrando le sue intuizioni nella sua concezione pedagogica costruzionista, Papert fonda così una scuola di pensiero sul tema del pensiero computazionale ancora oggi più attuale che mai e che ritroviamo in Sam Aaron. Per dirla con le parole della Wing, che nel 2006 ha rilanciato il concetto, "computation thinking involves solving problems, designing systems, and understanding human behavior, by drawing on the concepts fundamental to computer science"[7] (Wing 2006, 33). La sua applicazione va dunque al di là dell'informatica: è un modo di ragionare che permette di affrontare problemi, di scomporli e di risolverli attraverso algoritmi. Per questo "[c]omputational thinking is a fundamental skill for everyone, not just for computer scientists. To reading, writing, and arithmetic, we should add computational thinking to every child's analytical ability"[8] (*Ibid.*).

In quest'ultima affermazione Wing non tiene però conto che non esiste solo lo scrivere, il leggere e il far di conto. Ed è qui che interviene Sam Aaron. Leggendo i passi della conferenza citati qualcuno forse è stato colto di sorpresa da alcune scelte terminologiche di Sam: davvero la programmazione è un mezzo per "esprimere se stessi"? Qui entriamo nel cuore della visione alla base del lavoro di Sam Aaron, dove emerge l'artista, il musicista: non si può vincolare l'attività della programmazione al perimetro ristretto delle esigenze della produzione industriale e dell'informatica. La programmazione è anzitutto un linguaggio e come tale ha grandi potenzialità anche nell'ambito espressivo ed artistico-estetico. Per Sam tutto è 'programmabile'. E qui ritorna Papert, come vedremo tra breve.

In sintesi, rinchiudere l'attività del programmatore in una nicchia di specialisti che lavorano nei vari settori economici per rendere più efficienti le procedure di produzione è una grave svista: la programmazione è un linguaggio comune e serve essenzialmente per esprimersi e dunque anche per fare arte. Di fatto, esso è già entrato in dialogo con i mezzi di espressione artistico-estetica tradizionali. Quindi, come la scuola si dedica ad insegnare i linguaggi verbale, aritmetico, visivo, musicale e via dicendo in tutte le loro potenzialità espressive, essa dovrebbe parimenti dedicarsi anche ad insegnare il pensiero computazionale in tutte le sue potenzialità espressive. Insomma, per Sam il pensiero computazionale deve entrare nelle scuole non solo e non tanto per le indubbie ricadute educative o per creare futuri ingegneri, ma anzitutto perché avere competenze in questo ambito è un diritto di tutti i cittadini. E chi vedesse in tutto questo un discorso politico ha perfettamente colto nel segno.

[6] "Come i computer possono influenzare il modo in cui le persone pensano e imparano".
[7] "Il pensiero computazionale implica la soluzione di problemi, la progettazione di sistemi e la comprensione del comportamento umano attingendo ai concetti fondamentali dell'informatica".
[8] "Il pensiero computazionale è un'abilità fondamentale per tutti, non solo per gli informatici. Tra le abilità analitiche di ciascun bambino dobbiamo aggiungere alla lettura, alla scrittura e all'aritmetica il pensiero computazionale".

Sam Aaron (Opificio Golinelli, 14.2.2020, fotografia di Luigi Parisi)

3. Competenza comune

Ascoltando Sam Aaron non ho potuto fare a meno di ripensare agli insegnamenti di Gino Stefani e al suo concetto di competenza comune, alla sua lotta per il diritto del cittadino all'educazione musicale e al suo impegno sociale (vedi, ad esempio, Stefani 1976, 1977 e 1982). Nel caso di Stefani, naturalmente, l'oggetto è la competenza musicale, ma la critica alla separazione tra talentuosi esperti e persone comuni, e il duro giudizio sull'autoreferenzialità del mondo degli esperti, sono elementi comuni tra i due studiosi. In fondo, se nell'idea di Stefani che la musica debba essere considerata un linguaggio comune sostituiamo la parola 'musica' con 'programmazione', ritroviamo pari pari la visione di Sam Aaron e, in fin dei conti, anche quella di Papert: in entrambi i casi la questione riguarda le competenze di base.

In questo senso è curioso notare che i libri di Stefani da me citati sono stati scritti proprio mentre Papert lavorava al suo *Mindstorms*, pubblicato nel 1980, dove viene introdotto il concetto di pensiero computazionale. Di fatto Stefani e Papert, musicologo il primo e matematico il secondo, hanno in comune molto: la generazione, l'interesse per la pedagogia, l'impegno politico-sociale e una concezione di apprendimento esperienziale centrata sul fare, individualizzata, e basata sull'idea di competenza comune. Ancora: come la teoria della competenza comune di Stefani mostra le connessioni tra le attività cognitive, affettive e

motorie, così Papert non pensa il linguaggio di programmazione pertinente solo alla matematica e alle attività cognitive. Anzi, l'intero suo libro più famoso "is about how computers can be carriers of powerful ideas and of the seeds of cultural change, how they can help people form new relationships with knowledge that cut across the traditional lines separating humanities from sciences and knowledge of the self from both of these" (Papert 1993, 4)[9].

E chi se non Sam Aaron, programmatore e musicista, poteva mettere insieme Gino Stefani e Seymour Papert?

4. Sonic Pi

L'idea di creare un software per produrre musica attraverso il linguaggio di programmazione nasce nel 2013, dalle ricerche di Sam Aaron presso l'Università di Cambridge in collaborazione con la Raspberry Pi Foundation[10]. Sonic Pi nasce dunque in primo luogo come strumento educativo parte della suite didattica di Raspberry Pi. Solo in un secondo momento diventa uno strumento musicale dedicato alla performance professionale, mantenendo comunque sempre il suo approccio intuitivo e immediato.

Oggi Sonic Pi è sviluppato in modo del tutto autonomo da Sam Aaron, determinato a mantenere il software open source e gratuito nonostante l'assenza di supporti istituzionali. Si tratta di un progetto ambizioso dallo slogan assai significativo: "The Live Coding Music Synth for Everyone". Per scaricarlo basta collegarsi al sito https://sonic-pi.net/. Chi volesse sostenere il progetto può offrire il suo contributo su Patreon.

Sonic Pi è un software di live coding. In "Cos'è il live coding? Una breve introduzione" Giovanni Mori afferma che "Il live coding è una tecnica performativa improvvisativa in cui il performer utilizza un linguaggio di programmazione per impartire le istruzioni da svolgere al calcolatore in forma di algoritmi" (Mori 2018). Le performance di live coding possono essere musicali e grafiche, e ci sono software che permettono di controllare entrambi.

Ci sono varie importanti differenze tra questa attività e la programmazione tradizionale. In quest'ultima il codice viene progettato a tavolino in successive riscritture che possono anche prendere lunghi periodi di lavoro. E, prima di essere distribuito, il codice viene testato e verificato. Il *live coder*, invece, programma in diretta di fronte al pubblico mentre il codice stesso sta girando. Il live coding è dunque un'attività performativa che prevede estemporaneità, improvvisazione, estro, rischio. Per rendere evidente questa dimensione *on the fly*, i *live coders* proiettano in uno schermo il codice sorgente in forma di algoritmi che viene costantemente modificato. Un po' come ad un concerto l'ascoltatore non comprende a pieno le tecniche utilizzate dallo strumentista, ma ammira la performance e il risultato sonoro, così anche il *live coders* rende visibile le sue tecniche che il pubblico può seguire a vari livelli a seconda delle sue competenze.

[9] "Tratta di come i computer possano essere portatori di potenti idee e dei semi del cambiamento culturale, su come possano aiutare le persone a sviluppare nuove relazioni con la conoscenza che attraversino le linee tradizionali che separano le discipline umanistiche dalle scienze e la conoscenza del sé da queste".
[10] Ringrazio Stefano Rini e Giovanni Mori per i loro i preziosi commenti su questa seconda parte dell'articolo.

5. Ma è musica questa?

La domanda sorge spontanea: “Ma è musica questa?”. L’assenza della notazione musicale tradizionale e delle tecniche esecutive strumentali è infatti un ostacolo alla piena accettazione delle pratiche di live coding nel mondo della musica e dei musicisti. A ben vedere, però, la domanda sorge spontanea perlopiù in quella generazione che ha conosciuto la musica in un’epoca dove la computer music non era diffusa. Oggi, in un’epoca in cui la computer music può contare su una illustre tradizione, certi ostracismi suonano certo anacronistici.

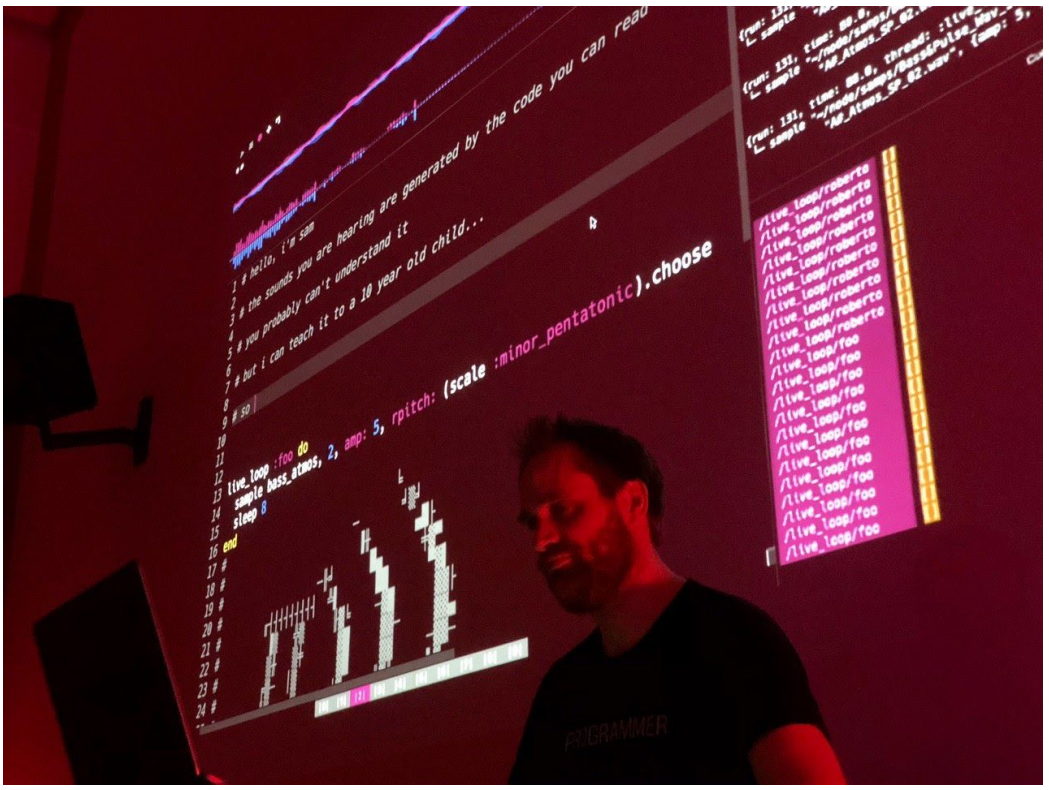
Di fatto, il live coding è un fenomeno emergente, ma ormai ben consolidato. Nel 2004 è nata TOPLAP, una comunità internazionale dedicata al live coding la cui filosofia si basa sull’idea che il “Live coding is inclusive and accessible to all”. Da allora sono nate molte altre iniziative mentre i software si sono moltiplicati. Bisogna sottolineare che il live coding non è un genere, ma un mezzo per fare musica, e la musica dei *live coders* va dalla scena Algorave alle ricerche musicali vicine alla musica e all’arte contemporanea. Il punto è che la scena musicale dei *live coders* è ormai una realtà e gli artisti intraprendono carriere professionali[11].

Alcuni video di performance di live coding:

- Sam Aaron live coding with Sonic Pi, *Practice Stream*, https://youtu.be/kxX6ZEhFlm4
- Sam Aaron e Ben Smith, *Live Jam*, https://youtu.be/zdc94RAX9UY
- Juan Romero e Patrick Borgeat, *Live-Coding* (TEDxKIT), https://youtu.be/Ix2b_qFYfAA. Il software usato è SuperCollider (vedi “Sitografia”).
- Francisco Bernardo, Chris Kiefer e Thor Magnusson, *Three Pidgins, Live Coding Performance*, https://youtu.be/DRxBCabqsqA. Il sistema usato è Sema, una tecnologia collaborativa basata sul web (vedi “Sitografia”).

E comunque, per tutti gli scettici, Sam ha ben dimostrato che il live coding è musica a tutti gli effetti. L’ha fatto a parole, spiegando come, in fondo, il linguaggio di programmazione non sia altro che un codice esattamente come la notazione tradizionale[12]. E l’ha dimostrato anche nei fatti, giovedì 13 febbraio, il giorno prima del laboratorio, quando si è esibito in una scatenata session a Granata, a Bologna, in un’esibizione che nulla ha avuto da invidiare a un qualsiasi altro concerto. Sam Aaron, con un portatile Mac collegato a una scheda audio e a un sintetizzatore Moog Sirin, ha trascinato con i suoi ritmi frenetici un pubblico ipnotizzato dal fiume di algoritmi che scorrevano sullo schermo.

[11] Per approfondire gli scenari del live coding, vedi Mori 2020.
[12] Vedi Thor Magnusson in “Sitografia”.



Sam Aaron in concerto a Granata, a Bologna (13.2.2020, fotografia di Luigi Parisi).

Attrezzatura di Sam Aaron al concerto a Granata (Bologna). Si riconoscono il MacBook Pro, il Moog Sirin e le due parti di una tastiera separabile retroilluminata. A sinistra del computer si intravede la scheda audio (13.2.2020, fotografia di Luigi Parisi).

6. Educazione musicale e pensiero computazionale

Come abbiamo detto, Sonic Pi nasce pensando all'educazione musicale. In questo senso Sam Aaron ha anche proposto alcune riflessioni sul tema (vedi Aaron e Blackwell 2013; Aaron, Blackwell e Burnard 2016; Aaron 2016a, 2016b, 2017, 2020). Vale la pena di segnalare, però, che il coding musicale a scuola non è una novità[13]. Software per la programmazione a blocchi come ad esempio Scratch o i kit di SamLabs e Arduino (e derivati)[14] propongono infatti anche percorsi didattici musicali, spesso sotto l'etichetta di 'attività STEAM'[15]. Ma Sonic Pi ha una differenza fondamentale rispetto agli strumenti sopra menzionati. Questi ultimi sono nati con finalità molto ampie, e quella dei suoni è solo una delle possibili strade di sviluppo, una strada che neppure figura tra quelle più battute. Sonic Pi, invece, è un software specifico per la musica, progettato e organizzato pensando unicamente al mondo dei suoni. Anzi, è uno dei software di live coding oggi più popolari e diffusi tra i musicisti prima che tra gli insegnanti. Per questo Sonic Pi ha un punto di forza fondamentale per un insegnante di musica: permette di creare musica in modo versatile e con suoni di grande qualità[16]. Insomma, con Sonic Pi le creazioni dei ragazzi suonano come 'autentiche' composizioni, non come il surrogato di musica che spesso gli insegnanti si trovano fronte quando affrontano la musica elettronica.

Non si sta dicendo che altri strumenti debbano essere accantonati per lasciare spazio a Sonic Pi. Anzi, la programmazione a blocchi, più facile e intuitiva rispetto a Sonic Pi, è certo fondamentale per introdurre i principi del coding seguendo la moderna didattica sul pensiero computazionale. Anche i microcontrollori di Arduino sono utili per sviluppare a scuola percorsi sull'elettronica musicale, anche se in effetti il loro uso richiede competenze avanzate certo non scontate per un insegnante di musica e implica dei costi[17]. Quello che sostengo è che Sonic Pi apre frontiere musicali infinitamente più ampie di questi strumenti pur rimanendo in linea con gli stessi principi pedagogici e didattici.

Siamo dunque di fronte a un nuovo mezzo per sviluppare progetti di educazione musicale? Sicuramente sì: è possibile utilizzare Sonic Pi per esemplificare concetti di teoria musicale e per studiare l'acustica del suono, oppure per comporre melodie e ritmi. Sonic Pi permette di entrare nel vivo dei principi della musica elettronica, ed è inoltre un ottimo mezzo per lavorare sulla creatività perché permette agli studenti di progettare e realizzare idee musicali personali in modo relativamente immediato. Infine, la pratica del live coding è una pratica

[13] Lo stesso Papert, vista la sua visione olistica, non trascura certo la musica. Anzi, fin dai suoi primi esperimenti con LOGO prevedeva che alcuni bambini si potessero esprimessero in forma 'matematica' oppure in forma 'verbale' oppure anche artisticamente, sia nell'ambito 'visivo' sia in quello 'musicale' (Papert 1993, 13). Per un'interessante riflessione sul rapporto tra pensiero computazionale e la musica, vedi Chong 2018.

[14] Per informazioni sulle risorse citate, vedi "Sitografia".

[15] L'acronimo STEAM significa *Science*, *Technology*, *Engineering*, *Arts* e *Mathematics*. Deriva dall'idea di integrare l'espressività artistica (la A di Arts) a quel tipo di percorsi didattici di carattere tecnico-matematico raccolti sotto l'etichetta STEM. Per dirla con Georgette Yakman, che ha proposto questo acronimo una decina d'anni fa, STEAM è "Science and Technology, interpreted through Engineering and the Arts, all based in a language of Mathematics" (Yakman 2010, 18).

[16] Il sintetizzatore usato da Sonic Pi è SuperCollider, uno degli strumenti più diffusi anche a livello professionale.

[17] Da qualche anno si è imposto all'attenzione degli insegnanti di musica LitteBeats (vedi "Sitografia"), uno strumento musicale creato con un insieme di moduli elettronici. Più spesso i musicisti che padroneggiano Arduino preferiscono dilettarsi nella auto-costruzione di strumenti musicali.

ormai affermata e, come tale, credo vada inserita negli argomenti da trattare in educazione musicale per motivi culturali.

Ma il live coding è qualcosa di più che un semplice nuovo mezzo per l'educazione musicale. Il live coding, infatti, apre le porte a nuovi territori musicali e quindi anche a nuovi territori per l'educazione musicale, con tutte le conseguenze del caso. Se da una parte va certo tenuto conto che l'utilizzo del digitale nell'ora di musica – fare musica attraverso interfacce grafiche assistite o algoritmi – genera una riduzione del tempo dedicato all'apprendimento delle competenze strumentali e teoriche tradizionali, dall'altra parte dobbiamo anche tenere conto di ciò che invece le tecnologie musicali *permettono di fare, di potenziare, di scoprire*. E, nel caso del pensiero computazionale, le novità sono sostanziali. In particolare,

> [l]ive coding implicitly challenges several classical conventions in technical music education. Most significantly, it challenges the conventional distinction between performance and composition, by notating music at performance time. Loss of this distinction between performance and composition introduces ambiguities with regard to the nature of notation, improvisation and interpretation[18] (Aaron, Blackwell, Burnard 2016, 76)
>
> Furthermore, controlling digital technologies through code rather than a pre-programmed GUI promises user empowerment, with more expressive languages that are able to reconfigure a device instead of simply operating it within predetermined constraints[19] (*ibid.*, 77).

Non dobbiamo infine dimenticare le conseguenze nell'ambito della progettualità interdisciplinare che sono implicite nel live coding. Con Sonic Pi è infatti possibile insegnare tutti i principi fondamentali della programmazione. Così, se spesso le attività di programmazione sono associate a discipline quali Matematica e Tecnologia, e se già con lo sviluppo delle STEAM si è visto che è possibile pensare a attività di coding che prendano in considerazione anche il mondo delle arti, con Sonic Pi anche la musica può entrare *da protagonista* in progetti interdisciplinari orientati alla programmazione.

Riferimenti

Aaron, Sam (2016a), "Live coding education", *MagPi Educator's Edition*, Special Edition issue 2, pp. 44-47. http://sonic-pi.net/files/articles/Live-Coding-Education.pdf (20.3.2020).

Aaron, Sam (2016b), "Sonic Pi – performance in education, technology and art", *International Journal of Performance Arts and Digital Media*, 12, 2, pp. 171–178.

Aaron, Sam (2017), intervista su CAS TV, https://youtu.be/7sEMKXrRaAs.

Aaron, Sam (2020), conferenza a Bologna del 14 febbraio 2020, https://youtu.be/c_Y8UBdA-_I (20.3.2020).

[18] "Il live coding sfida implicitamente diverse convenzioni tradizionali nell'educazione musicale di carattere tecnico. La cosa più significativa è che il live coding, basandosi sulla scrittura della musica durante la performance, sfida la classica distinzione tra esecuzione e composizione. La perdita di questa distinzione tra esecuzione e composizione introduce ambiguità riguardo alla natura della notazione, dell'improvvisazione e dell'interpretazione".

[19] "Inoltre, il controllo delle tecnologie digitali attraverso il codice piuttosto che attraverso una interfaccia grafica pre-programmata assicura un potenziamento delle capacità dell'utente attraverso linguaggi più espressivi in grado di riconfigurare un dispositivo invece che semplicemente di farlo funzionare entro vincoli prestabiliti".

Aaron, Sam - Blackwell, Allan F. (2013), "From Sonic Pi to Overtone", *Proceedings of the First ACM SIGPLAN Workshop on Functional Art, Music, Modeling & Design - FARM '13*, New York, Association for Computing Machinery, pp. 35-46, https://dl.acm.org/doi/10.1145/2505341.2505346 (3.4.2020).y

Aaron, Sam - Blackwell, Alan F. - Burnard, Pamela (2016), "The development of Sonic Pi and its use in educational partnerships: Co-creating pedagogies for learning computer programming", *Journal of Music, Technology and Education*, 9/1, pp. 75-94.

Agostini, Roberto (2020), "Due giorni con Sam Aaron e il suo Sonic Pi a Bologna", Servizio Marconi TSI, http://serviziomarconi.istruzioneer.gov.it/2020/04/08/due-giorni-con-sam-aaron-e-il-suo-sonic-pi-a-bologna/ (8.4.2020),

Chong, Eddy K.M. (2018), "Teaching and Learning Music through the Lens of Computational Thinking", Kuswarsantyo et al. (a cura di), *Second International Conference on Art and Arts Education (ICAAE 2018): Interdisciplinary Approach on Arts and Arts Education*, Atlantis Press, 2019, https://download.atlantis-press.com/article/125910432.pdf (20.3.2020).

Mori, Giovanni (2018), "Cos'è il live coding? Una breve introduzione", *Musica Elettronica.it*, 27/4/2018, https://www.musicaelettronica.it/cose-il-live-coding-breve-introduzione/ (2.4.2020).

Mori, Giovanni (2020), *Live Coding? What does it mean? An Ethnographical Survey on an Innovative Improvisational Approach*, Roma, Aracne, in corso di pubblicazione.

Papert, Seymour (1993), *Mindstorms. Children, Computers, and Powerful Ideas*, 2nd ed. (prima ed. 1980), New York, Basic Books.

Stefani, Gino (1976), *Introduzione alla semiotica della musica*, Palermo, Sellerio.

Stefani, Gino (1977), *Insegnare la musica*, Firenze, Guaraldi.

Stefani, Gino (1982), *La competenza musicale*, Bologna, Clueb.

Wikipedia contributors (2020), "Algorave", in *Wikipedia, The Free Encyclopedia*, https://en.wikipedia.org/w/index.php?title=Algorave&oldid=946451630 (20.03.2020).

Wing, Jeannette M. (2006), "Computational Thinking", *Communications of the ACM*, 49/3, pp. 33-35, https://www.cs.cmu.edu/~15110-s13/Wing06-ct.pdf (3.4.2020).

Yakman, Georgette (2010), "ST Σ @M Education. An overview of creating a model of integrative education", proprietà intellettuale di G. Yakman, 8/7/2010, https://www.academia.edu/8113795/STEAM_Education_an_overview_of_creating_a_model_of_integrative_education (3.4.2020).

Sitografia

Arduino, https://www.arduino.cc/.

Fondazione Golinelli, Bologna, https://www.fondazionegolinelli.it/.

Granata Workspace, Bologna, https://www.granatacoworking.it/.

LittleBeats, https://www.littlebits.cc/curriculum/music

Patreon Sonic Pi, https://www.patreon.com/samaaron.

Raspberry Pi Foundation, https://www.raspberrypi.org/.

Samlabs, https://samlabs.com/.

Scratch, https://scratch.mit.edu/.

Sema live coding environment, https://github.com/mimic-sussex/sema.

Servizio Marconi TSI, Tecnologie della Società dell'Informazione, Ufficio Scolastico Regionale Emilia-Romagna, http://serviziomarconi.istruzioneer.gov.it/.

Sonic Pi, http://sonic-pi.net/.

SuperCollider, https://supercollider.github.io/.

Thor Magnusson, http://www.ixi-audio.net/thor/.

TOPLAP, https://toplap.org/.