Simone Francia

Coding ergo sum

Riflessioni sul coding musicale a Scuola

Premessa

In occasione dell'incontro di formazione per Animatori Digitali organizzato dal Servizio Marconi TSI nel 2019, a Salsomaggiore Terme, ho avuto modo di conoscere il software Sonic Pi in quell'occasione presentato da Roberto Agostini.
Prima di allora non ero a conoscenza di sistemi informatici di codifica per comporre musica. Nonostante al primo sguardo l'interfaccia risultasse molto distante rispetto a consueti software per la produzione musicale, rimasi affascinato dalla possibilità di controllare in modo infinitamente preciso ogni parametro del suono attraverso l'utilizzo di valori numerici, inoltre gli esempi mostrati dal formatore all'ascolto risultavano di ottima qualità.

Le aspettative nei confronti del coding a scuola

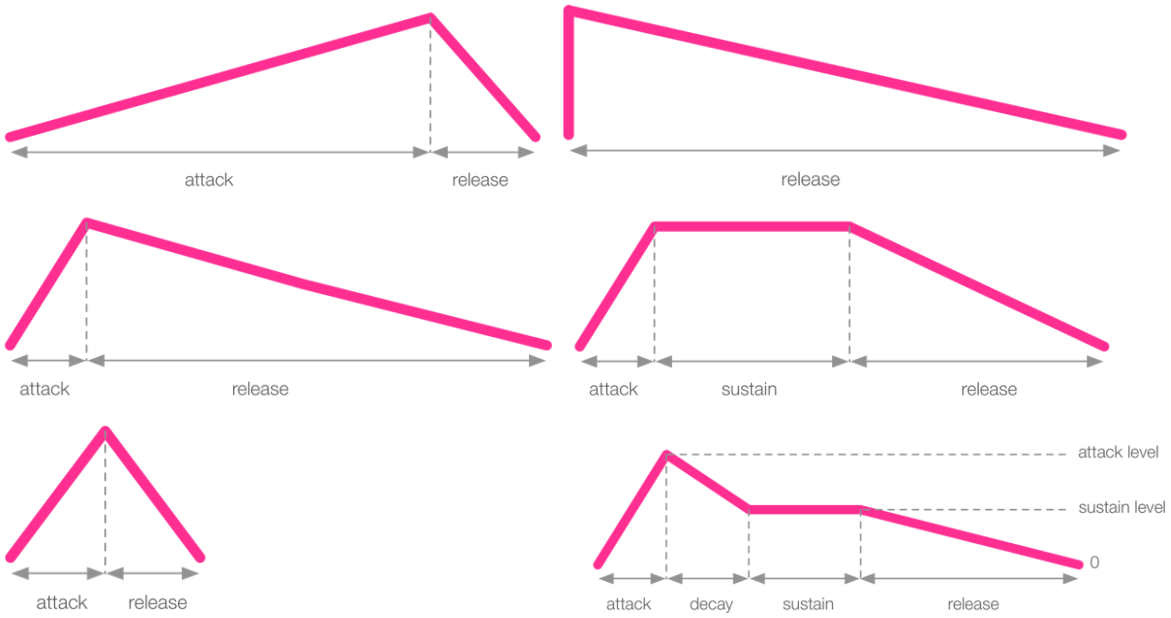
Da alcuni anni si parla della necessità di sviluppare il cosiddetto pensiero computazionale[1], mentre più recentemente invece si è sviluppato un interessante dibattito riguardo lo sviluppo del pensiero digitale creativo[2]. Per quanto riguarda la composizione musicale Roberto Agostini e Leo Izzo hanno affrontato il tema del "coding espressivo" con un contributo pubblicato in Musica Domani: nel loro articolo accennano circa la consuetudine di collocare le attività di coding nelle aree scientifiche e matematiche «dove l'ambito dei problemi che il pensiero computazionale è chiamato ad affrontare è spesso centrato sulle esigenze del mondo della produzione industriale e informatica». Leggendo i contributi che incoraggiano la scuola ad implementare lo sviluppo del pensiero computazionale si può notare che sono perlopiù pubblicati da gruppi editoriali legati al mondo industriale; inoltre, osservando l'interfaccia dei software con i quali è possibile comporre musica si può notare la somiglianza con

[1] Cfr. Agenda Digitale, *Italiani popolo di analfabeti funzionali: serve un "maestro Manzi 4.0" e tanto coding,* a cura di Carlo Gelosa.
[2] Cfr. Il Sole 24 Ore, *Educare le giovani generazioni al digitale creativo. Perché bellezza, innovazione, arte e poesia arrivano anche dalle tecnologie,* a cura di Barbara Forresi.

interfacce dedicate al mondo della programmazione industriale, per esempio quelle per il controllo delle macchine a controllo numerico computerizzato (C.N.C.).
Tuttavia, come ho accennato poco fa, guardando l'interfaccia di Sonic Pi è altamente probabile rimanere colpiti dalla sintassi, e ancor più sorprendente è ascoltare come le variazioni numeriche modifichino nel dettaglio il suono. La precisione informatica mi appare in qualche misura in interessante antitesi rispetto all'epoca in cui viviamo, in cui la superficialità dei linguaggi e delle relazioni standardizza anche la fruizione culturale in generale e più nello specifico la fruizione musicale. Per non parlare dell'esplosione dei Social che ha ulteriormente complicato la matassa dalla quale districarsi per fruire contenuti musicali di qualità. Ora, osservando i ragazzi e dialogando con loro circa le frequentazioni musicali, negli anni ho notato un progressivo appiattimento della capacità di giudizio nei confronti degli artefatti che quotidianamente ascoltano e in particolare una consueta superficialità estetica. Sembra che le differenze, se ci devono essere, debbano essere enormi, mentre le piccole differenze sembrano non interessare i giovani.
A questo punto mi sorgono due domande: è possibile sfruttare la sintassi di software come Sonic Pi per portare i ragazzi ad una maggiore attenzione all'ascolto?
Grazie al coding è possibile sviluppare una nuova estetica del suono lontana dal frastuono bulimico dei modelli imperanti?

Alla scoperta del coding

Pieno di curiosità per il software (gratuito) sviluppato da Sam Aaron mi sono avvicinato all'esplorazione del mondo del coding musicale: inizialmente utilizzando i classici esempi da cui partono i formatori «play 60; sleep 1», i primi loop, semplici melodie, fino all'inviluppo. L'inviluppo mi ha portato a riflettere su dettagli che come musicista ho imparato a controllare senza predeterminazione: come voglio che inizi il suono? Come voglio che prosegua? Come voglio che termini? Sono tutte domande alle quali un musicista dà risposte utilizzando il proprio strumento e sono ciò che fa la differenza nelle interpretazioni, ma dovendole programmare occorre avere un'idea di suono prima che esso nasca.
Per aiutarmi nella programmazione mi è stato utile osservare le immagini trovate in un tutorial pubblicato sul sito nuovamusicaperleducazione.it.



Durante le mie esplorazioni domestiche immaginavo che portando in classe simili istanze ci sarebbe stata la rincorsa al dibattito e ai tentativi di esplorare tutte le possibili sfumature da applicare al suono, tuttavia così non è stato. La maggioranza dei ragazzi desiderava giungere a un prodotto finito in tempi rapidi e possibilmente affine ai generi ascoltati e quindi, purtroppo, il mio tentativo di decentramento dai repertori massificati grazie al coding, al momento, non ha avuto un riscontro del tutto positivo.

Nel procedere alla scoperta del linguaggio di programmazione sono rimasto affascinato dal controllo numerico applicato ai parametri del suono, un controllo talmente minimale da far sembrare un software capace di suonare come un musicista con il suo strumento. E probabilmente è proprio questa l'intenzione dello sviluppatore, creare un software in grado di suonare come uno strumento musicale, e in effetti è ciò che avviene con il live coding, pratica nella quale programmatori espertissimi sono in grado di creare piccole differenze e procedere molto rapidamente realizzando strutture.

Giovanni Mori in un suo articolo pubblicato sul sito musicaelettronica.it cita uno dei massimi esperti di live coding musicale, Thor Magnusson il quale afferma:

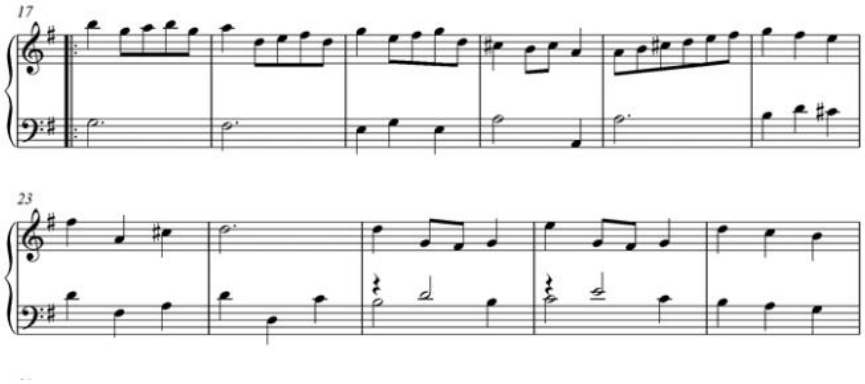
> «[...] il live coding musicale è un'attività di composizione dal vivo. Pertanto, l'impiego di questa tecnica richiede lo sviluppo di abilità sia nel campo della composizione musicale sia in quello della performance live. Il live coder deve allora acquisire la capacità di astrarre il pensiero musicale, cercando di tradurre i suoni che ha in testa in linguaggio di programmazione, in modo

> tale che l'esecutore, ovvero il software che è in esecuzione sul computer, possa interpretare le istruzioni impartite e svolgere il compito. Il codice, seguendo quanto detto sopra, viene a configurarsi come una sorta di partitura musicale, che viene scritta dal vivo e ha carattere modulare e rizomatico (per prendere in prestito un termine dai filosofi Gilles Deleuze e Felix Guattari), ovvero ha una struttura non lineare ma a blocchi non gerarchici che vengono continuamente modificati e trasformati dal performer.»

Un cambio di paradigma

Procedendo nel mio tentativo di imparare nuovi comandi che mi permettessero di progredire con nuove composizioni mi sono detto che probabilmente, come avviene in composizione, mi sarebbe stato utile trascrivere brani preesistenti. Quindi ho trascritto alcune opere più complesse rispetto le precedenti con le quali mi ero misurato: *Piano Piece in F K 33 B* di Mozart, *Für Elise* di Beethoven, qualche *Circle Song* di McFerrin, fino a capire che in realtà la difficoltà con la quale mi stavo misurando non era solo la sintassi, ma il cambio di paradigma tra una scrittura pensata in orizzontale, dove la melodia è orizzontale e l'armonia verticale, e una scrittura informatica verticale, dove la macchina esegue delle linee di calcolo dall'alto al basso. Con Sonic Pi, generalmente, si scrivono in orizzontale tutti i dettagli che il suono deve possedere (ampiezza, attacco, rilascio ecc.) e in successione verticale gli ulteriori comandi riguardanti l'articolazione temporale del suono (pause, effetti ecc.). Certo si possono scrivere delle liste orizzontali a fianco delle quali si attribuiscono i valori di durata, purtuttavia resta comunque una scrittura basata su blocchi verticali di comandi che la macchina esegue in successione.

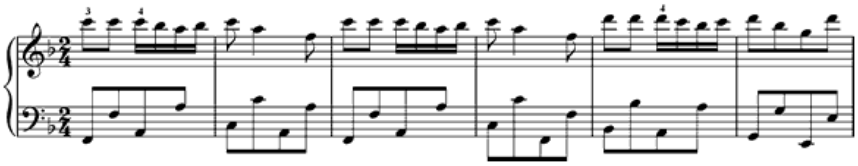
J. S. Bach, *Minuet in G N° 4* (estratto della notazione tradizionale)
from "A little notebook for Anna Magdalena Bach"



```
play_pattern_timed [83],[0.5]
play_pattern_timed [79,81,83,79],[0.25]
play_pattern_timed [81],[0.5]
play_pattern_timed [74,76,78,74],[0.25]
play_pattern_timed [79],[0.5]
play_pattern_timed [76,78,79,74],[0.25]
play_pattern_timed [73,71,73,69],[0.5,0.25,0.25,0.5]
play_pattern_timed [69,71,73,74,76,78],[0.25]
play_pattern_timed [79,78,76],[0.5]
play_pattern_timed [78,69,73],[0.5]
play 74
```

Posso affermare di aver necessitato di molti mesi di adattamento per questo tipo di organizzazione del materiale, difficoltà che invece sembra non aver riguardato i miei studenti quando abbiamo programmato in classe.

Mozart, *K 33* (estratto della notazione tradizionale)

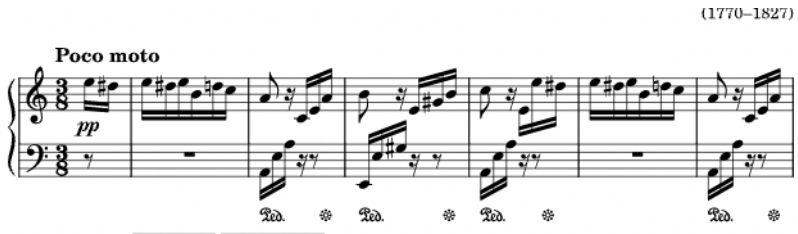


Mozart, *K 33* (estratto della notazione in coding)

```
in_thread do
  play :C5, release: 0.75
  sleep 1
  play :C5, release: 0.75
  sleep 1
  play :C5
  sleep 0.5
  play :As4
  sleep 0.5
  play :A4
  sleep 0.5
  play :As4
  sleep 0.5
  play :C5, release: 0.75
  sleep 1
  play :A4, release: 2
  sleep 2
  play :F4, release: 0.75
  sleep 1
  play :C5, release: 0.75
  sleep 1
  play :C5, release: 0.75
  sleep 1
  play :C5
  sleep 0.5
  play :As4
  sleep 0.5
  play :A4
```

```
in_thread do
  play :F2
  sleep 1
  play :F3
  sleep 1
  play :A3
  sleep 1
  play :F3
  sleep 1
  play :C3
  sleep 1
  play :F3
  sleep 1
  play :A3
  sleep 1
  play :F3
  sleep 1
  play :F2
  sleep 1
  play :F3
  sleep 1
  play :A3
  sleep 1
  play :F3
  sleep 1
  play :C3
```

Beethoven, *Für Elise* (estratto della notazione tradizionale)



Beethoven, *Für Elise* (estratto della notazione in coding)

```
7     use_synth [:hollow, :tri].choose
8
9     play :E5, amp: 1.5
10    sleep 0.5
11    play :Ds5, amp: 1.5
12    sleep 0.5
13    play :E5, amp: 1.5
14    sleep 0.5
15    play :Ds5, amp: 1.5
16    sleep 0.5
17    play :E5
18    sleep 0.5
19    play :B
20    sleep 0.5
21    play :D5
22    sleep 0.5
23    play :C5
24    sleep 0.5
25    play chord( :A3, :minor), release: 3, amp: 1.25
26    play :A4, amp: 1.5
27    sleep 1.5
28    play :C4, amp: 1.5
29    sleep 0.5
30    play :E4, amp: 1.5
31    sleep 0.5
32    play :A4, amp: 1.5
33    sleep 0.5
34    play chord( :E3, :major), release: 3, amp: 1.25
35    play :B4
36    sleep 1.5
37    play :E4, amp: 1.5
38    sleep 0.5
39    play :Gs4, amp: 1.5
40    sleep 0.5
```

È stato dunque interessante osservare come un cervello formato negli anni a leggere spartiti in orizzontale abbia faticato di più rispetto a cervelli meno formati in tal senso: infatti la mia fatica di immaginare le melodie in verticale (anche se con frammenti organizzati in liste) non abbia interessato i miei studenti che da questo punto di vista non hanno manifestato difficoltà (qui sarebbe interessante aprire una riflessione su programmi scolastici e formazione in uscita nei rispettivi cicli: cosa devo sapere o saper fare? Come lo devo apprendere? Come devo insegnare?).

Applicazione in ambito didattico

Dopo aver trascritto molto materiale e aver tentato di apprendere il più possibile da corsi e tutorial, con un minimo di sicurezza sulla gestione di eventuali problemi, ho deciso di proporre un laboratorio pomeridiano agli studenti interessati a misurarsi con il coding in ambito musicale.

Le lezioni, articolate in 10 incontri della durata di 1 ora ciascuna, sono state condotte utilizzando il metodo *trasmissivo addestrativo* (l'insegnante fornisce l'esempio e lo studente lo riproduce) e il metodo definito *attivismo spontaneo* (l'insegnante predispone un setting educativo all'interno del quale il ruolo assunto dall'educando è di tipo attivo nel gruppo dei pari e si struttura a partire da alcuni *input* offerti dal setting predisposto).

Le prime lezioni sono state caratterizzate dallo stupore dei ragazzi nell'ascoltare come quei codici si traducevano in suono. In alcuni casi lo stupore si è manifestato in modo assai esplicito con espressioni facciali buffe e commenti felici; ciononostante con il procedere delle lezioni i ragazzi si sono misurati con la difficoltà maggiore: la sintassi. Nel programmare in coding non sono ammessi errori nella gestione della sintassi, e anche il più piccolo sbaglio può bloccare l'esecuzione del brano.

Di seguito riporto i commenti dei ragazzi al termine del laboratorio.

Intervista agli studenti

Cosa ti è piaciuto del laboratorio Coding Time?

M.B. *mi è piaciuto imparare cose nuove e avere un'attività in più al pomeriggio che mi ha permesso di socializzare. Il laboratorio mi ha dato l'opportunità di aprire uno sguardo più ampio sul mondo della musica digitale.*

M. Z. *Mi è piaciuto il momento in cui abbiamo composto un brano in totale autonomia.*

Y. K. *Mi è piaciuto scoprire un nuovo software che attraverso i codici può comporre brani*

Cosa ti ha colpito di Sonic Pi?

A. M. Z. *Mi piace che si possa utilizzare anche offline. Mi è servito anche per combattere la noia a casa.*

M. B. *Mi ha colpito la precisione della scrittura, anche se grazie all'aiuto del prof. siamo riusciti a scrivere; i suoni che posso comporre con Sonic Pi mi sembrano moderni.*

Y. K. *Mi ha colpito il fatto che il software segnala gli eventuali errori di scrittura. Ho apprezzato anche i suggerimenti contenuti nel software.*

Cosa ti è apparso difficile?

A. M. Z. *Ho faticato a memorizzare i codici.*

M. B. *Ho trovato difficile ricordare la sintassi corretta per far funzionare il software.*

Y. K. *Ho faticato a ricordare come scrivere in modo corretto i codici.*

Lo inseriresti nelle lezioni del mattino?

M. B. *Mi è piaciuto poter scegliere liberamente di partecipare, ma credo che anche nelle lezioni del mattino sarebbe apprezzato.*

A. M. Z. *Secondo me si può fare in classe con tutti, però non tutti potrebbero capire e probabilmente farebbero confusione.*

Y. K. *Secondo me è meglio lasciare la libertà di scegliere perché probabilmente molti non riuscirebbero a farlo.*

All'indirizzo sottostante è possibile ascoltare un elaborato prodotto durante il laboratorio.

https://soundcloud.com/simotrump/fx-in-school?utm_source=clipboard&utm_medium=text&utm_campaign=social_sharing

Considerazioni finali

Al momento non mi è chiaro che tipo di impatto abbia sulla formazione del pensiero un utilizzo diffuso delle tecniche di programmazione informatica. Per quanto riguarda

la Musica, di certo si tratta di un modello di organizzazione del materiale sonoro assai distante dai modelli tradizionali che lo hanno preceduto. Probabilmente un pensiero basato su logiche di programmazione informatica segue traiettorie assimilabili al concetto di apprendimento programmato (Skinner), e questo non credo sia in antitesi con un apprendimento più empirico, di scoperta dei materiali, timbri, parametri del suono. Ciononostante il modello di apprendimento basato sulla codifica di algoritmi non mi sembra offrire la possibilità di creare link e reti di sapere, ma piuttosto sembra un sapere chiuso in sé stesso e autoreferenziale, adatto a una grande riproducibilità e poco idoneo per sviluppare saperi ramificati.

È fuor di dubbio che Sonic Pi sia una grande risorsa per gli studi più avanzati (Licei musicali e Conservatori) e che il futuro della Musica elettronica debba misurarsi con i sistemi di live coding, come del resto già avviene, ma in qualità di insegnante che opera in classe ogni giorno mi domando se la grande richiesta di introdurre pratiche didattiche basate sul coding a scuola non risponda ad esigenze distanti dalla formazione dei saperi di base, e per quanto riguarda il primo ciclo forse debba rimanere in subordine alle buone pratiche artigianali di esplorazione e scoperta dei saperi.

Bibliografia

Roberto Agostini, *Rock around Sonic Pi,* Musicheria, Aprile 2020.

Roberto Agostini e Leo Izzo, *Coding the beat. Coding espressivo e algoritmi sonori,* Musica Domani, EDT, n° 185, 2021.

Sitografia

Carlo Gelosa, in Agendadigitale.eu, *Italiani popolo di analfabeti funzionali: serve un "maestro Manzi 4.0" e tanto coding*.

Barbara Forresi, in Il Sole 24 Ore, *Educare le giovani generazioni al digitale creativo. Perché bellezza, innovazione, arte e poesia arrivano anche dalle tecnologie*.

Gruppo NUME, nuovamusicaperleducazione.it.

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).