

IMprovEEsation: Intelligent Musical Evolutionary Entertainment

Davide Berardi, Matteo Martelli, Marco Melletti, Federico Montori

2 dicembre 2014

Sommario

- 1 Introduzione
- 2 Stato dell'Arte
- 3 Modello del Dominio

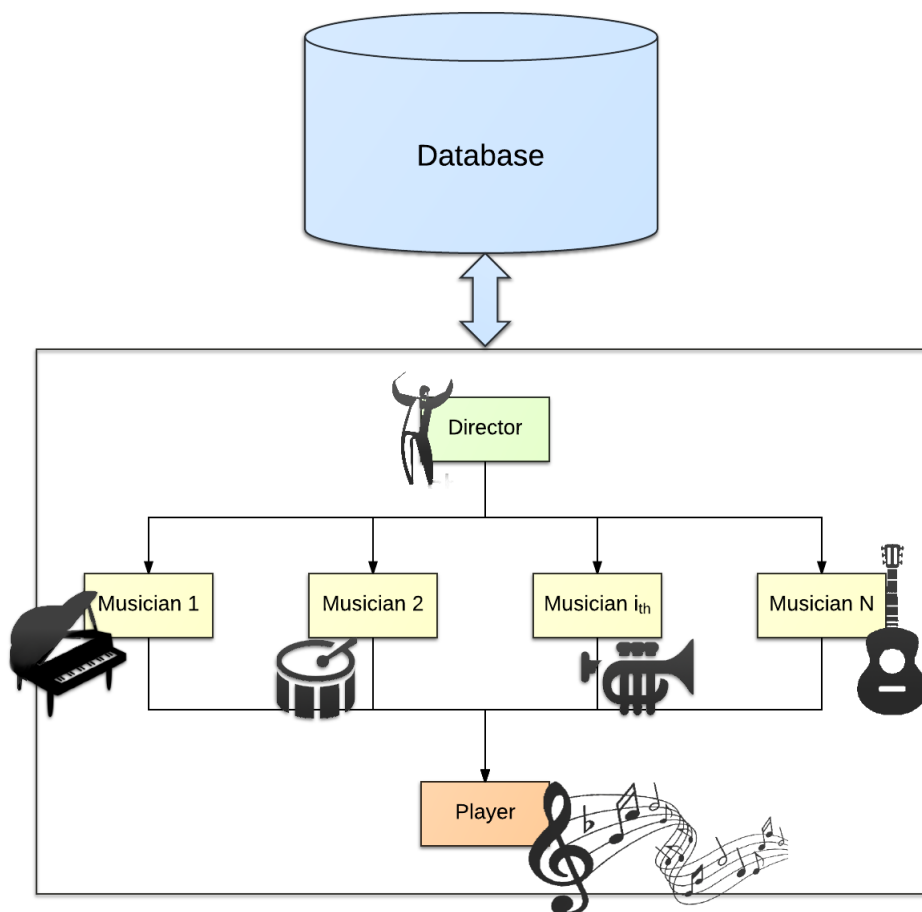


Figura 1: Schema dello scenario di progetto

<https://www.lucidchart.com/documents/edit/67c03865-48ea-4f6a-b052-e9f9c4cd8196?>

4 Overview dei Componenti

Il progetto è composto principalmente di tre parti principali:

Un **direttore**, il quale svolge il compito di centralizzare l'improvvisazione, det-

tando regole e decidendo i parametri generali per ogni punto dell'improvvisazione; potrebbe in un certo senso essere visto come una sorta di *coscienza comune*, la quale amministerebbe silenziosamente i vari musicisti, vedremo più avanti come la natura centralizzata del direttore inoltre aiuti, ad esempio, a sincronizzare i vari musicisti.

Un **player**, il quale assieme al direttore compone l'architettura centralizzata del progetto; il compito del player è di ricostruire e di assemblare le varie improvvisazioni provenienti dai vari musicisti, ha inoltre un'interfaccia modulare per salvare o riprodurre l'improvvisazione, per analogia con il direttore, il quale è il punto d'ingresso del progetto, il player è il punto dove viene formato l'output utile del progetto.

Vari **musicisti**, i quali prendono decisioni in base alla loro configurazione e all'output del direttore, processandole secondo vari meccanismi euristici (vedremo ad esempio implementazioni di meccanismi randomici o basati su algoritmi evolutivi).

Al fianco di questi componenti fondamentali è presente un ambiente di supporto per facilitare l'operazione, quali ad esempio il **database** dell'applicazione, responsabile dell'immagazzinamento della conoscenza dei vari componenti, ad esempio la rappresentazione dei vari generi e dei loro pattern collegati.

5 Componenti del Sistema

Come indicato in precedenza, i principali componenti in stretto contatto tra loro sono delle seguenti tre categorie, è stata scelta un'implementazione non a camere stagne tra di essi, in modo da rendere il sistema il più modulare possibile, potendo separare (anche fisicamente, distribuendoli su varie macchine) i vari componenti, utilizzando dei processi singoli per ogni istanza del singolo.

Si procede quindi con la descrizione dettagliata del comportamento di ogni elemento del sistema.

5.1 Direttore

5.2 Musicista

Come il direttore, il musicista nel nostro software è essenzialmente un processo. Il suo scopo principale è quello di creare in tempo reale della musica. Della buona musica? Ci prova; infatti il processo musicista trascorre la sua esistenza suonando delle note che possano "andar bene" assieme alle note suonate dagli altri musicisti. Questi ultimi non vengono lasciati soli nelle decisioni prese durante un'improvvisazione ma il direttore li aiuta a prendere delle scelte che possano aver senso fra di loro e li aiuta a coordinarsi. Il direttore quindi, tramite un certo protocollo di comunicazione, invia determinati parametri globali a tutti i musicisti che a loro volta scandiscono il database per cercare delle note che possano avere senso nel loro attuale contesto. Ad ogni insieme di note che i musicisti ottengono ad ogni passo dell'esecuzione è correlato un set di probabilità, il quale viene utilizzato per filtrare le note scelte da utilizzare e ad introdurre il comportamento di improvvisazione.

5.3 Player

6 Interazione e Comunicazione

La comunicazione è uno dei punti cruciali del progetto, sia per la quantità di informazione scambiata, che per la sincronia dei messaggi.

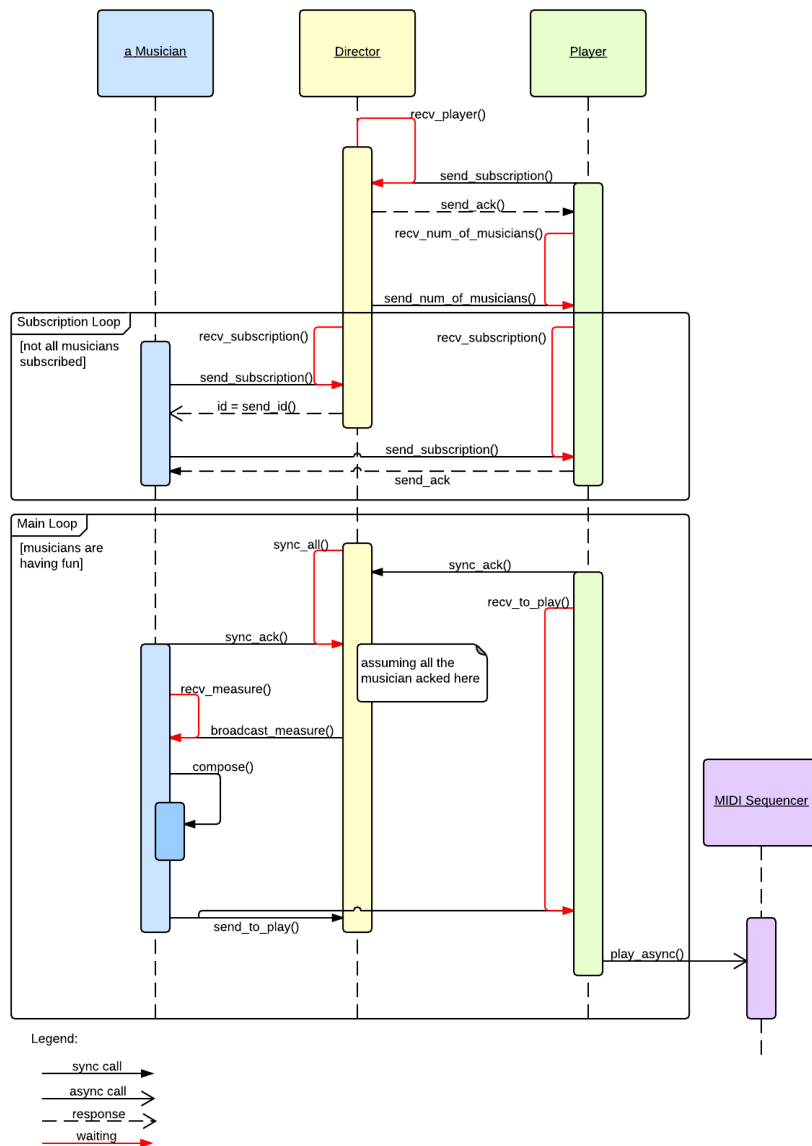


Figura 2: Diagramma del protocollo di comunicazione

Il protocollo è strutturato principalmente in tre fasi distinte.

6.1 Inizializzazione

Inizialmente il direttore riceve una sottoscrizione da parte del player. Alla sottoscrizione (opportunamente segnalata tramite *ack*), susseguirà un pacchetto contenente il numero di musicisti ¹. A questo punto l'inizializzazione tra i componenti principali (direttore e player) è completa, e la loro comunicazione, come vedremo si limiterà a messaggi di sincronizzazione.

6.2 Ciclo di Sottoscrizione

A questo punto, ogni musicista si preoccuperà di inviare la sua sottoscrizione all'improvvisazione sia al direttore che al player, in questo modo le due componenti principali avranno una chiara visione dell'improvvisazione, pur rimanendo completamente indipendenti.

6.3 Ciclo principale

Finita l'inizializzazione inizia il ciclo vero e proprio d'improvvisazione; una volta sincronizzati tutti i musicisti (e il player) a barriera, vengono inviate da parte del direttore le informazioni di improvvisazione, come vedremo più avanti questi pacchetti contengono tutte le informazioni sullo stato dell'improvvisazione, in modo da mantenere una certa coerenza tra tutti i musicisti.

Una volta composto, i musicisti inviano quindi la loro creazione ², al player il quale si occuperà di suonare la composizione nell'ordine corretto; Mentre il direttore si occuperà della sincronizzazione, proseguendo con un nuovo passo del ciclo.

6.4 Libreria di comunicazione

L'implementazione della libreria di comunicazione è presente nel file *communication.cpp* e *struct.h*, se ne descrivono di seguito i dettagli.

Principalmente i pacchetti scambiati tra le varie istanze sono di 3 tipi: *Sottoscrizioni*, *Misure* e *Play Measures*.

6.4.1 Sottoscrizioni

Sono i pacchetti scambiati per la registrazione presso il Direttore o il Player

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
coupling																															
instrument_class																flags															

Essi contengono:

¹In modo da specificare solamente al direttore il numero di istanze di musicisti.

²Si rimanda alle opportune sezioni per come queste decisioni vengano prese.

Coupling

Un indicazione (per il player) sul fatto che il musicista sia in realtà una composizione di più musicisti ³

Instrument Class

Il tipo di strumento, utilizzato per la ricerca nel database e per l'assegnamento del corretto strumento *MIDI* in uscita.

Flags

I flag disponibili per la sottoscrizione sono:

- 0x0: Nessun flag rilevante.
- 0x1: Il musicista è un solista.
- 0x2: Il musicista utilizza pratiche di machine learning genetiche.

Da qui in poi sono riservati per utilizzi futuri.

La risposta a questi pacchetti non si limita al semplice ack, ma bensì ad un pacchetto contenente l'id (univoco per la sessione) del musicista.

É necessario indicare che la registrazione del il player presso il direttore avviene inviando questo pacchetto con un coupling pari ad un valore costante (-1).

6.4.2 Misure

Sono i pacchetti contenenti l'informazione sullo stato dell'improvvisazione e il prossimo passo d'improvvisazione, sono inviati in broadcast dal direttore a tutti i musicisti.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31		
bpm																																	
soloist_id																																	
tempo.upper								tempo.lower																									
prioargs																																} 9	
note																scale																} tempo.upper	
chord																mode																} tempo.upper	
tags length																																	
tags (variable)																																	

BPM

Un indicazione sui bpm dell'improvvisazione corrente.

³Immaginiamo ad esempio la mano sinistra e la mano destra dello stesso pianista, le quali devono essere assegnate allo stesso canale di output.

Soloist ID

L'ID del musicista che improvvisa correntemente in modalità solista.

Tempo (upper e lower)

Indica la *Time Signature* dell'improvvisazione corrente.

Prioargs

Sono 9 campi costanti il quale compito é specificare una scala di priorità con la quale effettuare le scelte di improvvisazione.⁴

Note e scale

Sono *tempo.upper* campi contenti la successione dei centri tonali della misura.

Chord e mode

Sono *tempo.upper* campi contenti la successione di accordi della misura.

Tags

É un campo testuale utilizzato per indicare attributi del pezzo da improvvisare (ad esempio il genere).

6.4.3 Play Measures

Sono i pacchetti contenenti l'informazione dettagliata sulle note suonate è la composizione che, in output dal musicista, viene passata al player pronta per essere allineata con le altre battute ed essere suonata.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	
id																																
size																																
musician id																																
unchanged fst																																
note.tempo								note.id								note.triplets								note.velocity								} Notes (size)
note.chord_size																																
note.notes																																
bpm																																

Id

Un numero progressivo (per musicista) indicante la posizione della battuta da suonare.

Size

Il numero di note presenti nella misura corrente.

⁴Per una spiegazione più dettagliata si rimanda ai capitoli sul come operano il Direttore e il Musicista.

Musician id

L'identificativo univoco del musicista che ha generato la misura in questione.

Unchanged fst

Un booleano indicante il fatto che la prima nota sia cambiata o meno.⁵

Note

Sono quindi presenti *size* strutture contenenti le seguenti informazioni

Tempo

La durata della nota (in sedicesimi).

ID

Un numero progressivo (per misura) indicante la posizione nella misura.

Triplets

Un indicazione sul fatto che la nota debba essere scandita con tempo terzinato.

Velocity

La velocity *MIDI*, è un indicazione sul volume della nota in output.

Chord.size

Il numero di note presenti (nel caso sia un accordo, altrimenti la nota sarà singola)

Notes

Il vettore di note (in notazione *MIDI*).

BPM

Un indicazione sui bpm da utilizzare nella riproduzione dell'improvvisazione.

6.5 Meccanismi di sincronizzazione

L'ultimo compito della libreria di comunicazione è provvedere ad alcuni meccanismi per la sincronizzazione dei vari componenti.

Questo obiettivo è raggiunto utilizzando le chiamate di sistema di linux basate sugli eventi quali *epoll*, vengono radunati su di una barriera (in qualsiasi ordine) i musicisti, prima che venga loro notificato tramite *ack* la possibilità di continuare.

7 Rappresentazione della Conoscenza

Da qui per le prossime 3 sezioni usiamo i titoli che piacciono tanto agli intelligentisti. Pagina 5 del libro di IA. Manca interpretazione del linguaggio naturale perchè non sapevo cosa metterci dentro.

⁵Questa feature é necessaria nel caso si debba effettuare dei legati o dei continui dalla misura precedente.

7.1 Regole e Pattern

7.2 Database Relazionale

8 Ragionamento Automatico

8.1 Mente del Direttore

Come ragiona il direttore quando decide i pattern?

8.2 Mente del Musicista

Come ragiona il musicista quando decide le note?

9 Apprendimento

TODO:blabla generico su come potrebbe apprendere un musicista basandosi sull'algoritmo genetico: si fornisce un insieme di samples a cui il musicista cerca di arrivare. Alla fine dovrebbe salvare tutto salvare nel database per in modo da non buttare quello appreso. Per adesso c'è solo l'algoritmo genetico ma spieghiamo comunque come salveremmo la nuova conoscenza nel DB.

9.1 Algoritmo Evoluzionistico

supercazzola genetica e tante stampe.

10 Risultati Sperimentali

Dopo tanto sbatto funziona tutto random!

11 Conclusioni

Ci vuole un DB supermegagigante!!!

12 Sviluppi Futuri

12.1 Sviluppo della Conoscenza

Spieghiamo qui o in sviluppi futuri? Comunque potremmo proporre nel futuro di salvare nel db il risultato del genetico "facendo un match" dei quarter che sono usciti dal genetico con quelli che già ci sono nel db "aggiustando" le probabilità che già ci sono. Quelle che non ci sono possiamo aggiungerle.

Riferimenti bibliografici

- [1] Homer J. Simpson. *Mmmmm...donuts*. Evergreen Terrace Printing Co., Springfield, SomewhereUSA, 1998