Computing Science III

GRAPHICAL MUSIC COMPILER

By Esteban Alejandro Villalba Delgadillo,
Daniel Felipe Barrera Suarez

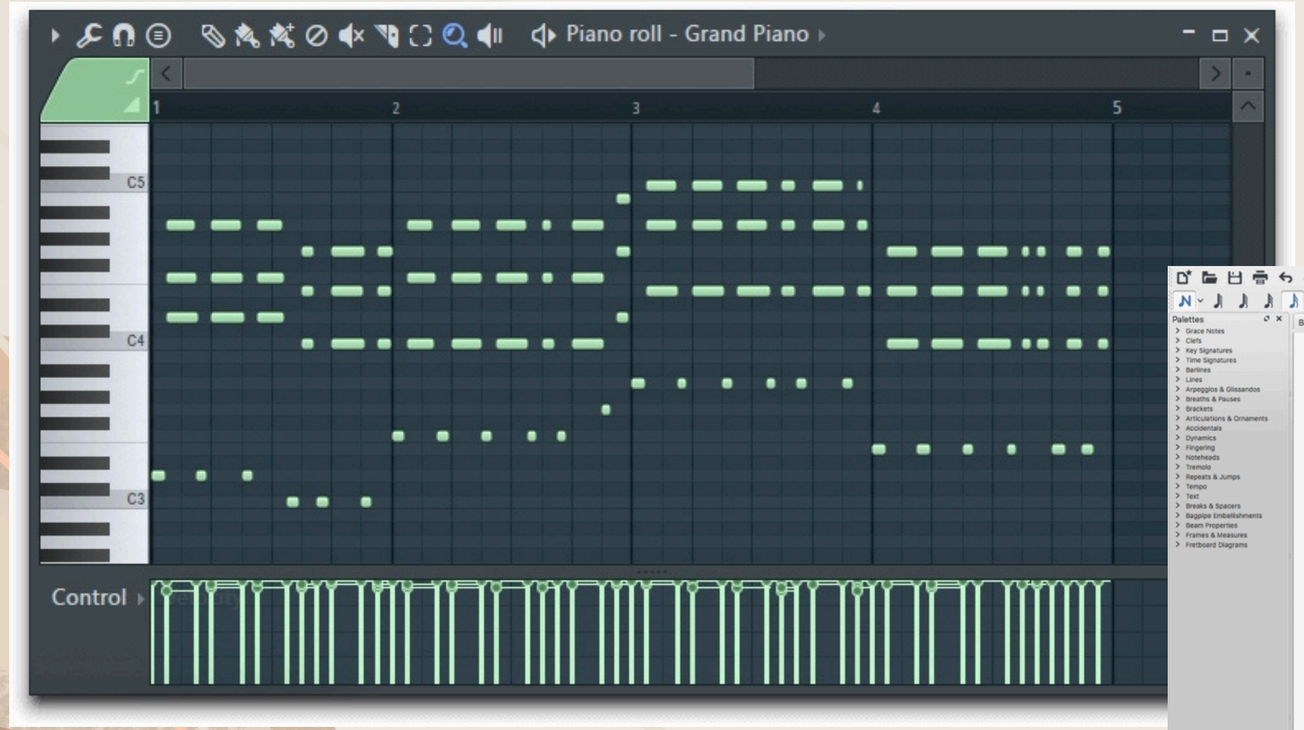
INDEX

INTRODUCTION

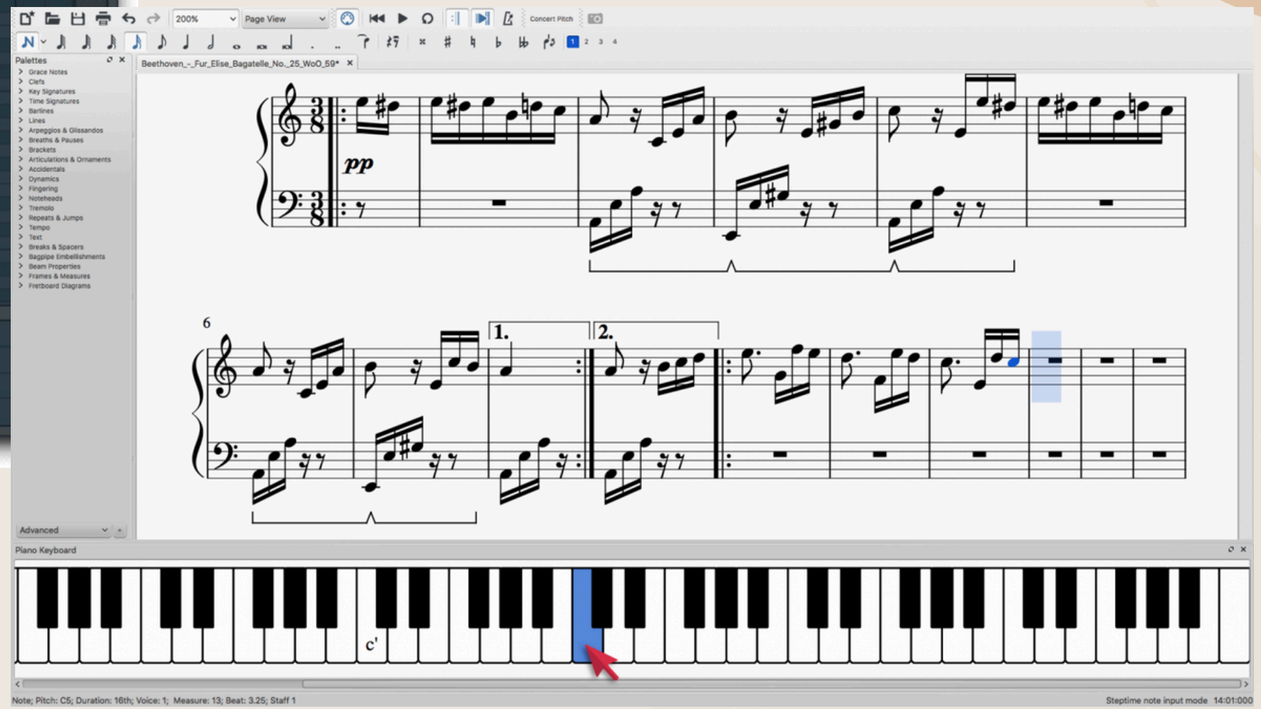
The intersection of computer science and music composition has enabled the creation of tools that automate or assist in generating musical structures

INSPIRATIONS

FLStudio



MuseScore

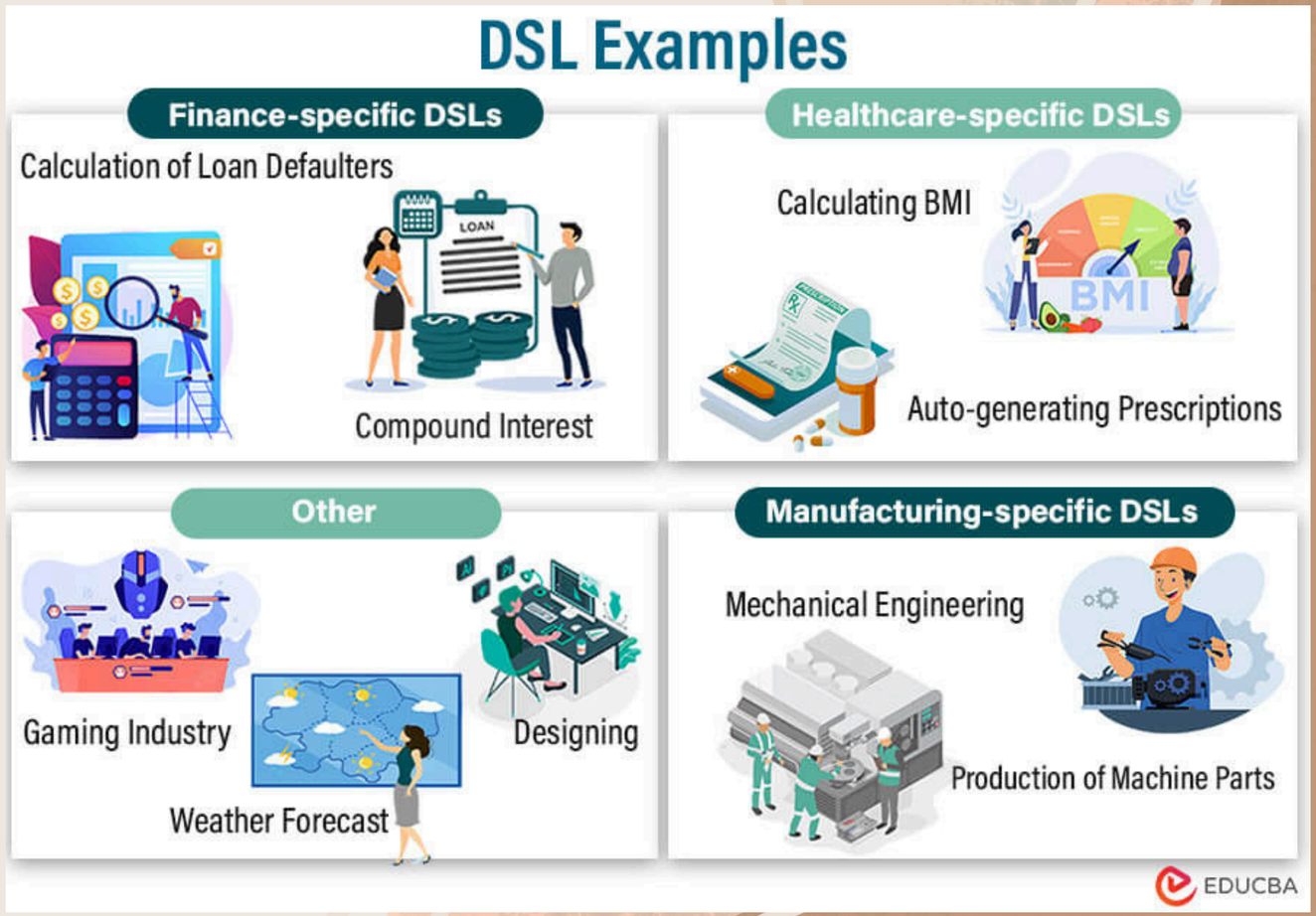


BACKGROUND

Domain-specific languages enable intuitive mappings between human intent and computational behavior. In music, DSLs like Sonic Pior TidalCycles offer powerful tools, but often require steep learning curves.

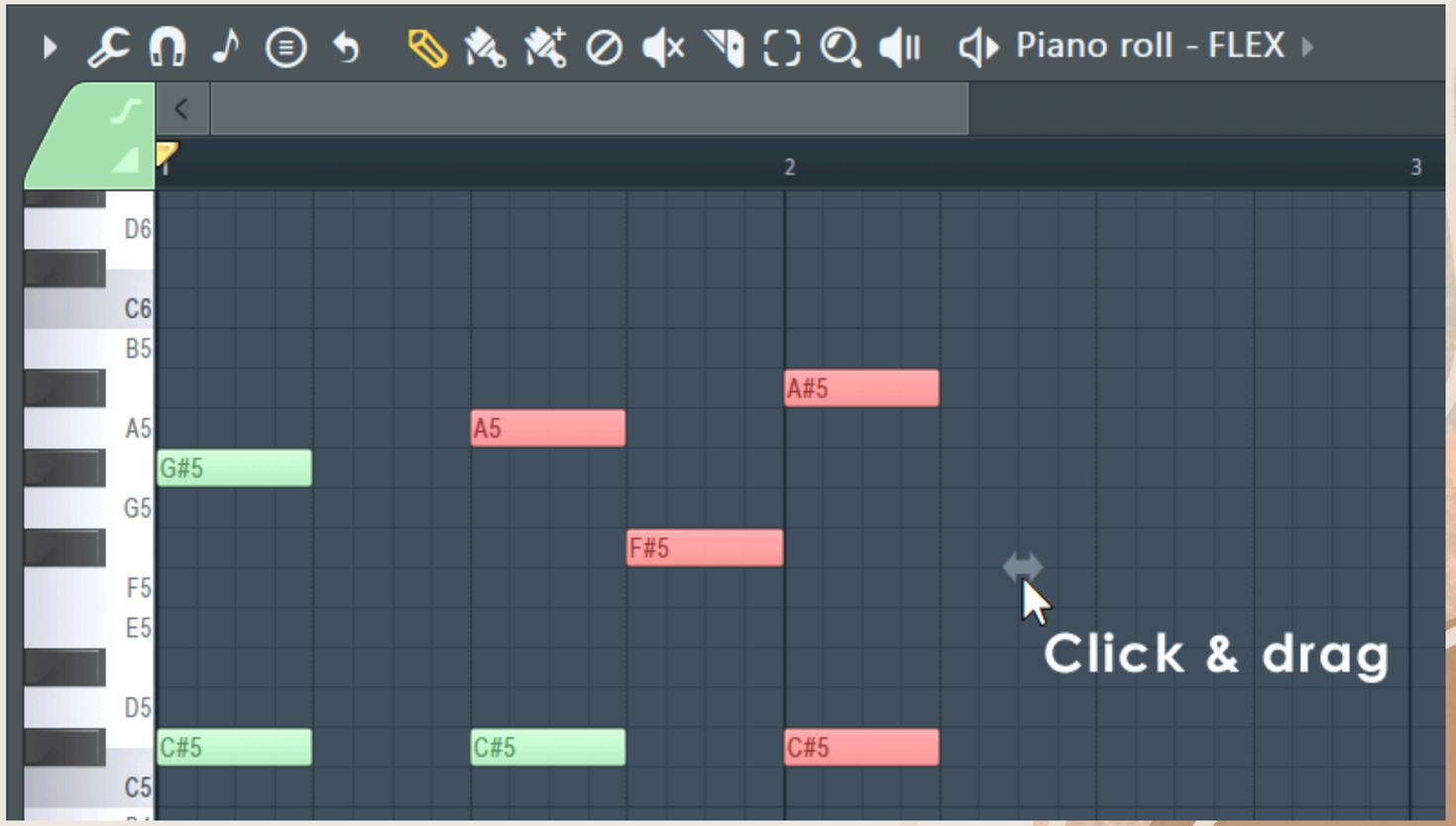
DOMAIN-SPECIFIC LANGUAGES (DSL)

Domain-specific languages enable intuitive mappings between human intent and computational behavior. In music, DSLs like Sonic Pior TidalCycles offer powerful tools, but often require steep learning curves.



Our approach simplifies this by defining a minimal grammar:

play C4 for 1.0 at 0.0
play E4 for 0.5 at 1.0



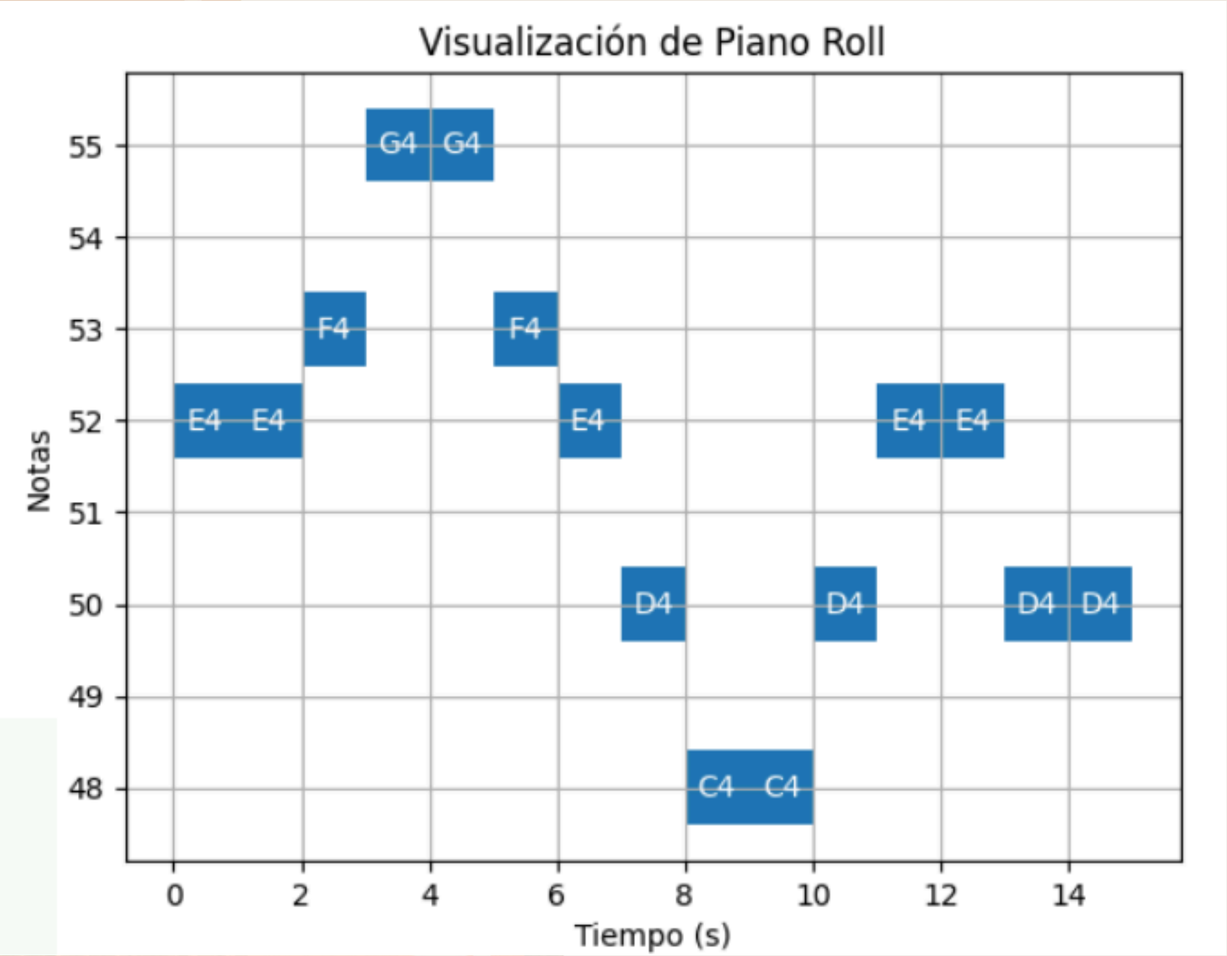
Objectives

DESIGN A SYMBOLIC INTERPRETER FOR MUSICAL PHRASES

CREATE A TEXTUAL INTERFACE TO INTERPRET THESE PHRASES.

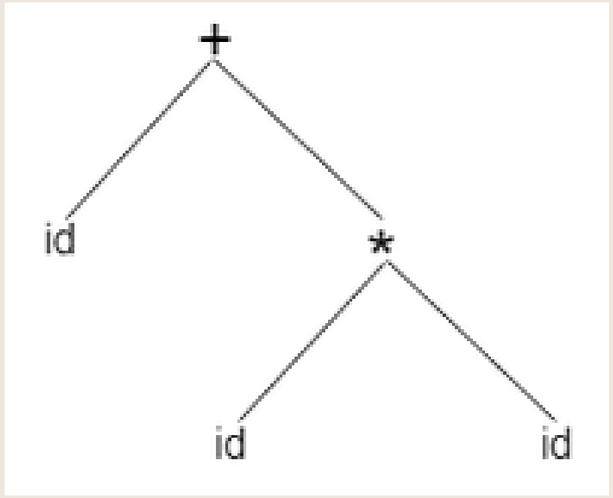
IMPLEMENT A SOUND EXECUTION PROTOTYPE.

ESTABLISH THE FOUNDATION FOR REAL-TIME VISUALIZATION.



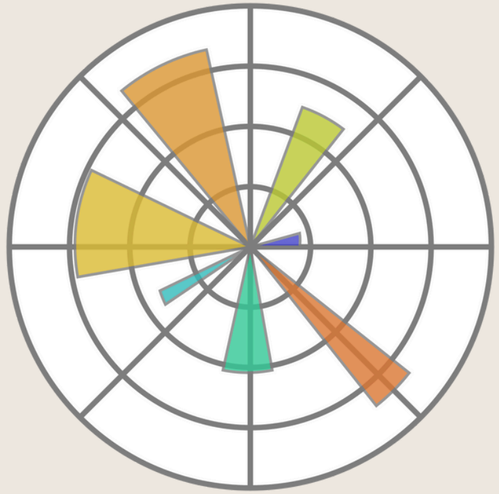
METHODOLOGY

The methodology begins with the definition of a minimal symbolic language that allows users to describe notes and their durations.



LANGUAGE DESIGN

Tokens such as NOTE, DURATION, and REPEAT are identified in the lexical phase. Parsing produces an abstract syntax tree (AST), which the interpreter uses to evaluate the structure into a mathematical representation



MATHLIB INTEGRATION

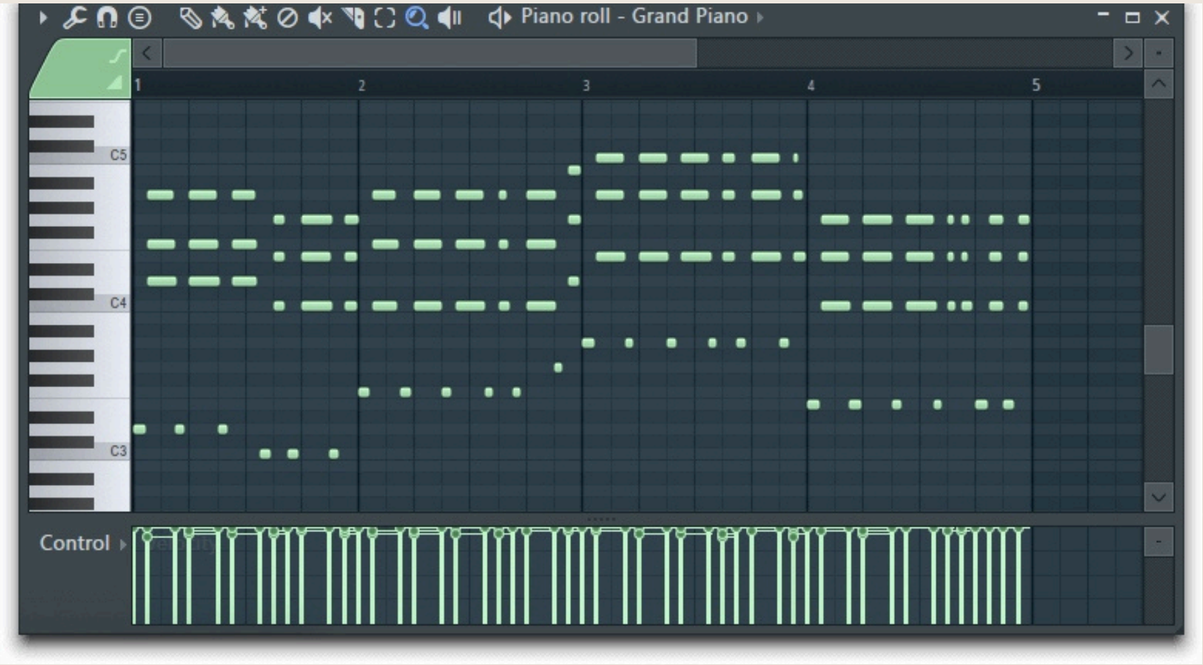
The core of the music generation is based on ‘mathlib‘, which handles mathematical transformations, including scaling and sequencing of note values. Each parsed rule translates into a function or transformation within ‘mathlib‘

GRAPHICAL REPRESENTATION

To enhance interactivity, we plan to integrate a graphical visualization layer using pygame. This component is envisioned to represent notes as visual objects on a staff or piano roll, displaying pitch and duration over time

SCOPE

This project focuses on rendering simple monophonic musical sequences using textual instructions. It supports basic rhythmic values such as quarter and eighth notes and operates at a fixed tempo, currently set at 120 beats per minute.



LIMITATIONS

The system has several limitations due to its prototype nature. It does not support polyphonic textures or real-time interaction, and playback is restricted to .wav audio files. Additionally, the project does not generate traditional sheet music notation but instead relies solely on a piano roll-style visual representation

Conclution

Using a simple language, it interprets and visualizes basic musical instructions, promoting the learning of both computational and musical concepts.

THANKS