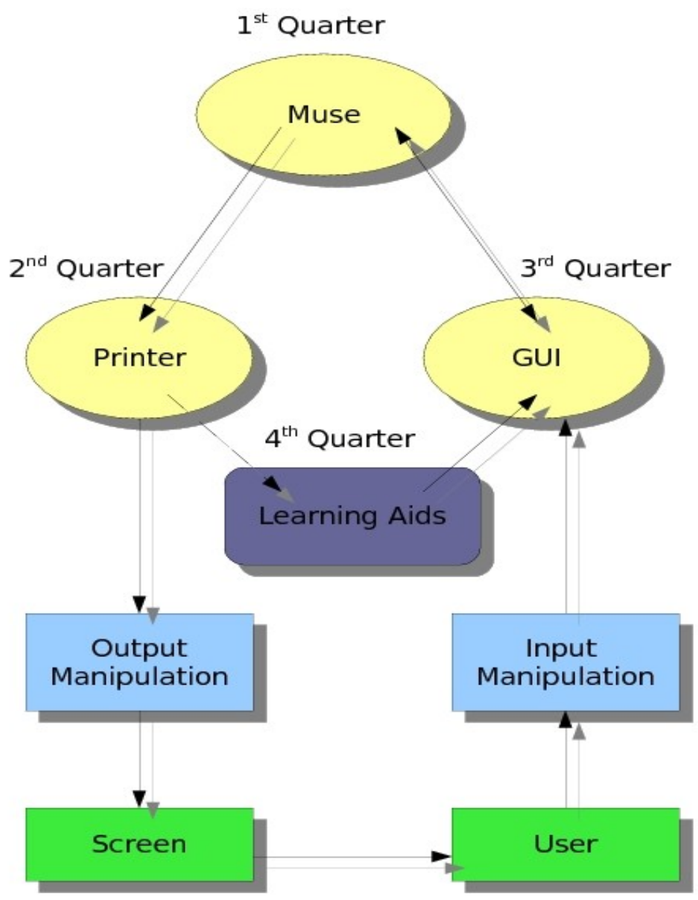
Compositional Software Development
Patrick Mutchler
2006-2007

Abstract
A new, free software designed for amateur composers and music students requires a less powerful editing system and can incorporate learning tools to aid teachers or students learning composition without the aid of a teacher.

The goal of this project is to create a music editing software that can be used as a teaching tool for music theory students

Architecture



Version System
The development system for this program has four major release versions, one for each quarter.

0.x – File input, ASCII output
1.x – File input, Graphics output
2.x – GUI input, Graphics output
3.x – GUI input, Graphics output, compositional tools, teaching aids

Expected Outcomes
A usable, free, intuitive program that can create and print simple to mildly complex scores in a variety of ways.

The program will be tested by other students in the classroom to make sure that it meets expectations in terms of usability and simplicity.

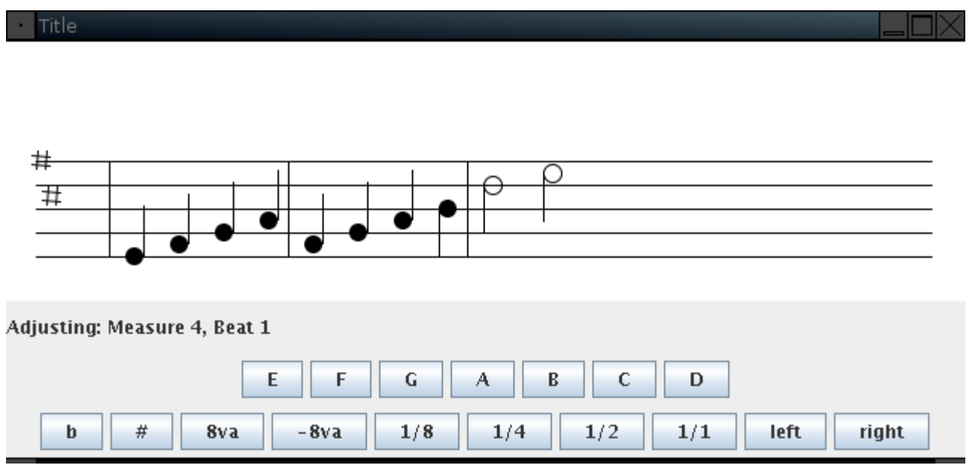
A new approach to composition software, with teaching aids included.

Background
Existing editing software falls into two major categories: compositional software and copying software with a focus on visual appeal.

Finale and Sibelius are two examples of compositional software. These programs are aids to composers to make writing music easier and nothing more. I have used some of the Sibelius interface as a model, but without so many complexities.

Lilypond is a program that focuses on music copying and making computer printed scores as appealing as hand written scores. Lilypond, however, cannot be used to create or edit scores quickly or easily due to a complicated file input system. Lilypond does not relate to my software at all, but is an example of software being developed in this field for other purposes.

Sample 2.2 GUI and Output



Class and Method Outline
Muse
•Creates the GUI and Printer objects
GUI
•Creates the GUI buttons
•Calls addNote in Printer when a button is pressed
Printer
•adds notes to the score matrix
•prints the score based on the score matrix.

Sample path of input to output
•User presses a button, GUI recognizes it
•Converted to string, passed to addNote in Printer.
•Adds the string to the score matrix
•The score is printed by measure
•Each measure is printed by note
•The paint component update