AMADEUS!

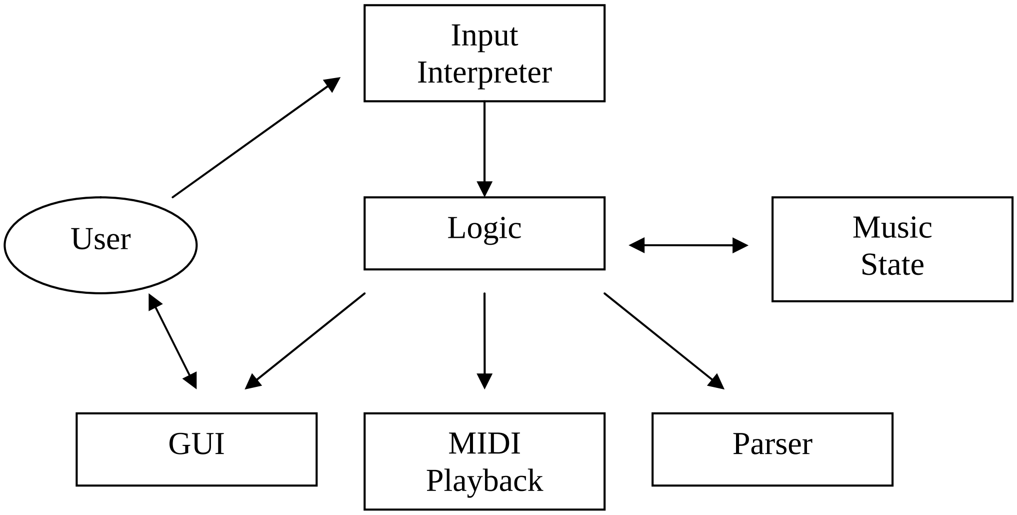
Specifications Document

By Dan Shue (dshue)
February 7, 2003

1 Description

Amadeus! is essentially a music composition program for Linux. Instead of writing music by hand, this program allows users to quickly and neatly compose music on the computer, giving them more flexibility and control over their music. There are 3 general features in Amadeus!: a) creating/editing music composition files, b) playing back the current composition in MIDI, and c) loading/saving music composition files. Target users include people who wish to compose and/or arrange music (e.g. for a cappella), and also people who wish to learn songs by listening to MIDI playback.

2 System Model Diagram



3 Annotations of Components

Input Interpreter: The Input Interpreter gathers information on what the user is performing (via keyboard & mouse) and translates it into useful data for the logic.

<u>Logic</u>: The Logic is the main control center of the program. It receives input information from the Input Interpreter, and accordingly modifies its current Music State. After modification, the Logic outputs the change via the GUI for the user to see. If the user wishes to save the change(s), the Logic will write out a file representation of the current Music State via the Parser. Conversely, if the user wishes to load a previous composition, the Parser will read in the file representation of that composition and relay the information to the Logic.

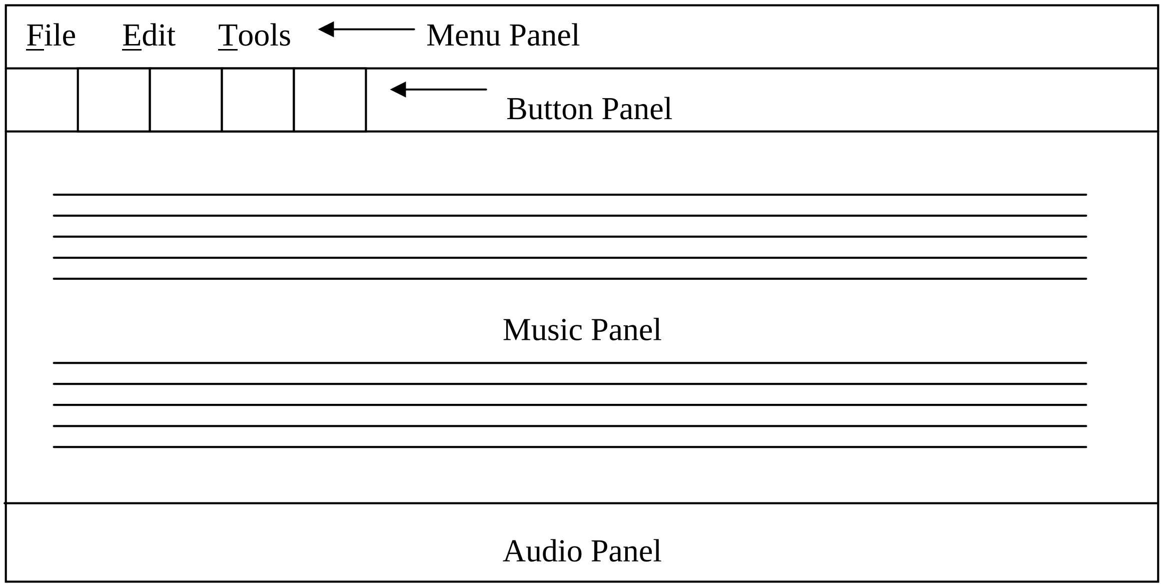
<u>Music State</u>: The Music State represents the current state of the music composition. It stores all the information about the song, and provides the platform for the GUI's display. The Parser also reads in the Music State when saving a file, and writes to it when loading a file.

<u>GUI</u>: The GUI is the visual representation of the current Music State (via the Logic). It reflects the changes done by the user, and also provides functions for the user to perform further modifications to the music composition. More details can be found in the User Interface sections (4 & 5).

<u>MIDI Playback</u>: The MIDI Playback handles the audio segment of the program. When the user wants to listen to his/her composition, the Logic gives the current Music State to the MIDI Playback, which will translate the composition into audible music.

<u>Parser</u>: The Parser handles the file I/O segments of the program. If the user wishes to save a file, the Logic will write out the current Music State to the Parser; if the user wishes to load a file, the Parser will read in the file representation of that composition and relay the information to the Logic.

4 User Interface Diagrams



5 User Interface Description

Menu Panel: The Menu Panel contains features such as creating new files, loading/saving files, printing files, and any other "non-musical" actions.

Button Panel: The Button Panel contains features such as specifying what type of note to write in, setting the dynamics, modifying the lyrics, and any other "musical" actions.

Music Panel: The Music Panel contains the music composition itself, and is where all the editing is done.

Audio Panel: The Audio Panel contains the MIDI player, which plays back the current music composition.

6 Non-Functional Requirements

Performance: Amadeus! is expected to perform its actions within a matter of seconds, especially the MIDI playback component.

Testing: As a general rule, whatever the user inputs must be properly reflected in the program's output. This can be easily tested by checking whether or not the GUI's Music Panel is displaying the correct modifications by the user. A slightly trickier component is the MIDI playback component, which must be consistent with what's shown in the Music Panel.

Reliability: Amadeus! must be as "crash-free" as possible, but if it does crash, the program should minimize information loss.

Ease of use: Amadeus! should be easy enough so that the user composing by hand; otherwise, there wouldn't be an advantage to using the program.

Portability: Since this program is specifically made for Linux (there are already several existing PC music composition programs), Amadeus! is not meant to be portable to other operating systems.

Documentation: As a music composition program, Amadeus! must have some documentation that explains 1) general music theory, and 2) the basic features of the program.

Dependencies on other systems: The music notation in Amadeus! will probably use graphics from external sources. Same thing goes for the MIDI player (unless we really want to code it ourselves).

7 Updated Requirements

Since I did a different project for my requirements assignment, here's the new set of requirements for Amadeus!:

Basic Features*:

- Creating new music compositions
    - Prompts user for song info
        - Song title
        - Author
        - Copyright notices
        - Additional comments
    - Prompts user for song specs
        - Key signature
        - Meter
        - Beat/Tempo
        - Number of Staffs (must specify treble or bass clef)

- Editing music compositions
    - Notes and rests
        - whole, half, quarter, eight, sixteenth beats
    - Sharps and flats
    - Note decorations

        accents, staccatos, slurs, ties, etc.
    - Key signature
        - A, B, C, …G(sharp/flat) major/minor
    - Meter
        - 2/4, 3/4, 4/4, 6/8, etc.
    - Beat/Tempo
        - 100 quarter beats per minute, for example
    - Dynamics
        - *fff, ff, f, mf, mp, pp, ppp*
    - Dynamic variances
        - *crescendo, decrescendo*
    - Tempo
        - *presto, allegro, andante, adagio, largo*, etc.
    - Tempo variances
        - *accelerando, ritard*
    - Measure Numbers
    - Repeats
    - Special endings (priority 3)
    - Lyrics (priority 3)

- Undoing/redoing actions

- MIDI playback of music composition
    - For multiple staffs: can select which staffs to play and mute (priority 3)
    - Can choose from a list of different MIDI sounds
    - Can transpose song and play in different key

- Loading/saving music compositions: file I/O
    - Show history: list 5 most recently opened files

- Printing music compositions (priority 4)
    - Show a print preview of composition

* Note: All basic features have priorities of 5 (on a scale from 1-5), unless otherwise noted.

Optional Features:

- Uploading/downloading music compositions via a centralized database server and its clients (priority 3)
- Musical dictionary: a glossary of music terms (priority 1)
- Internal mp3 player like Winamp; useful for arranging music (priority 1)

8 Risky Parts

- Amadeus! is a very ambitious project for CS190. It has a lot of potential, but it may be too hard for the scope of this class.

- There are already several existing music composition programs in the market, so it'll be hard to convince people to use Amadeus! instead of other software. (However, Amadeus! is specifically for Linux)