

MONASH UNIVERSITY

CSE3308 Software Engineering: Analysis and Design Practice Class, Week 7

Object-Oriented Analysis: Interaction Diagrams and State Diagrams

1 Interaction Diagram

MIDI¹ is an international standard for the communication between digital musical equipment and other digital devices, such as computers or sequencers, commonly used in modern day compositional methods.

MIDI specifies a file format which describes how to play a piece of music on a MIDI device. Most people would use a soundblaster as their MIDI sequencer, but there are other cards which can be placed in a computer which allow the music to be played on a keyboard/synthesizer.

MIDI files are organised as follows:

- There are several **tracks** in a file. These generally correspond to the different instruments, for example one track could be the piano part, another track could be the flute part, and another track could be the violin part. These tracks are all played back simultaneously.
- Each track is comprised of a list of **events**. There are several main types of events: Note-on, note-off, control-change, program-change, pressure, pitch-bend and system exclusive. Some events have one byte of data, others have two. Every event has a timing value, which states how many 'ticks' are between the preceding event from the same track and it, and this is used to control when the event should occur during playback.

Typically, the hardware which renders the music is capable of being given only one event at a time, and so playing a MIDI file requires the determination of which event is next to occur (i.e. the one with time-until-ready closest to 0), and sending that one next.

To play a MIDI file requires the following actions: ²

1. All tracks are rewound to the start of their event list, ready for playing.
2. For each track, the time until the next (first) event is determined.
3. The Hardware Device is told to begin playing. This causes it to start requesting the next MIDI event to process.
4. For each request for a MIDI event, each track is checked and that whose next event is soonest to occur (smallest time-until-ready value) will be selected.

¹Musical Instrument Digital Interface

²Note: this is a simplification. In reality there is more complicated processing involved.

5. The event will be sent to the Hardware device immediately.
6. Because the timing value of the event may be greater than one, each of the other tracks' next-event must be updated to account for the difference in timing between it and the event which was selected. This will possibly alter the time-until-ready value for the events of the other tracks.
7. Because there is typically more than one event in a track, the track from which an event was selected must prepare itself for the next event it contains. (Note, this new event may occur before the events which are waiting on the other tracks; it's timing value will not be altered at this stage as the other tracks' events were).
8. If there are no more events in a track, the track is 'finished'.
9. When there are no more tracks with events remaining to be played, the hardware is told to stop, and no further requests for MIDI events are made.

Draw either a sequence diagram or a collaboration diagram for a system which plays MIDI files.

You may wish to use the following classes: Track, MidiEvent, MidiFile, MidiHardware

2 State Diagram

In a jewellery manufacturing plant a product goes through several stages of manufacture before being completed. Some components can be sold after undergoing only the initial development stages (for final assembly else where in the world). Alternatively, they can be completed and turned into a number of different items (eg a pendant to hang on a chain and the left earring of a pair may have started off as the same item).

A number of items have gone missing due to theft. You have been asked to design a system for monitoring the manufacture of items and keeping track of them. You have decided to use an object-oriented language, and to have a "base" item which can turn into either an earring or a pendant. A base item has the following properties:

- unique ID
- Design number
- date entered into the system
- expected completion date and state (being created, in storage, sold as base, being converted to earring, being converted to pendant).

The states tell you where in the building the item can be located. (Basic construction is on the ground floor, storage is on the second floor, conversions to earrings are on the third and conversions to pendants is in a building across the road.) In extreme cases, items can be pulled apart and turned back into base items.

The base item also has a number of operations:

- `placeInStock()`
- `convertToEarring()`
- `convertToPendant()`
- `convertToBase()`
- `create()`
- `sell()`

Draw the state diagram for a base item.