



The Design of MCM

**Cayci Suitt, Sal Ledezma, Jimar Garcia, Gene Wie
ICS 125 – Ebert**

15 May 2001

Project Context

- ✍ **MCM = Motion Capture Music**
- ✍ **Translation of Motion to Music**
- ✍ **3D Data to MIDI**
- ✍ **Design phase**
 - Mapping
 - Translation

Project Plan

- ✍ **Revised since requirements**
- ✍ **Anticipated early start on design and implementation not realized**
- ✍ **On schedule in regard to required deadlines**
- ✍ **New subdivision of tasks**

The Design

 **Architectural Overview**

 **Key Modules**

 **Modification to Requirements**

 **Integration Test Plan**

Architectural Overview

MCM (top level)

MCM::Map

- **Map::GUI**
- **Map::IO**

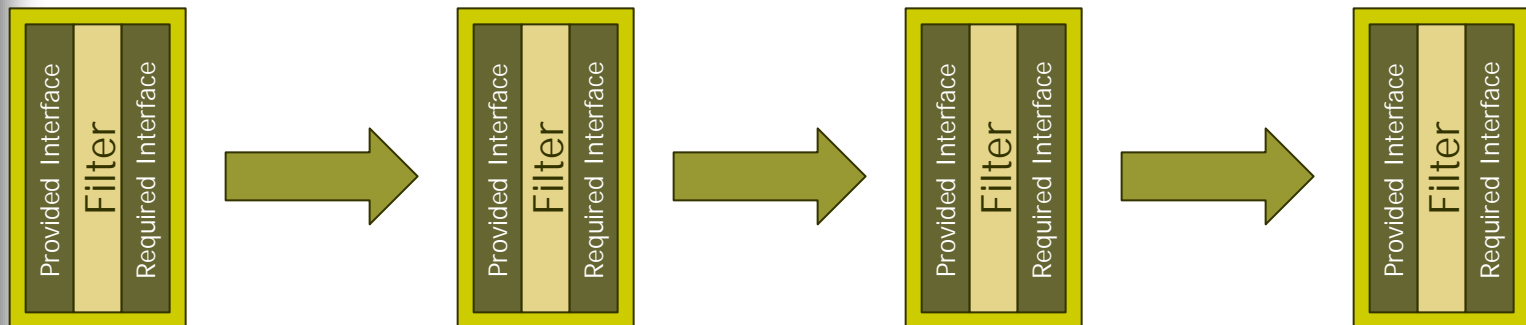
MCM::Translate

- **Translate::StreamReader**
- **Translate::Translator**
- **Translate::MIDI**
- **Translate::IO**

Architectural Overview Cont'd.

Overall

- No specific style for entire system
- MCM Map = rough hierarchy
- MCM Translate = Pipe and Filter



Key Modules

 **Map::GUI**

 **Map::IO**

 **Translate::StreamReader**

 **Translate::Translator**

 **Translate::MIDI**

 **Translate::IO**

Map::GUI




- ✍ **Provides user interface**
- ✍ **Data entry areas**
- ✍ **Functionality to launch translation executable**



Map::IO

 **Provides Save/Load functionality**

Translate::StreamReader

-  **Vicon-supplied code**
-  **Reads in motion data at TCP/IP port 800 on the RT machine**
-  **Interface to motion data in ascii format for easier translation**

Translate::Translator

-  **Correlates motion command to MIDI command based on user-specified mapping file**
-  **Room for different heuristic sets?**

Translate::MIDI

- ✍ **Freeware, open source library, written in C++**
- ✍ **Allows individual MIDI commands to be sent to a MIDI-compliant device**



Integration Test Plan

 **We're working on it...**