STRATEGIES FOR I-NET PUBLISHING

Version 1.0 12/14/03

PART I: THE BIG PICTURE

The purpose of this study is to look at what software and capabilities we have at the present time and compare them with what we would like to have and where we want to be somewhere in the future. This study will focus mainly on the display and printing of musical scores and parts.

I. A vision of the future.

The entire repertory of classical music will be available free on the internet. The data can be downloaded either piece-meal or in bulk for any purpose the user desires. Possible uses include.

Application	Technology Required
1. Searches; thematic and otherwise	Display, search engines + indices
2. Reference; checking data, expanding knowledge of repertory.	Display, indices
3. Printing of scores/parts for occasional or one-time use	Display, printing on 8.5 x 11
4. Source for editing and publishing electronic and/or hardcopy editions.	Display, editing, printing all sizes.

All of the above applications presuppose the existance of one or more massive databases of music in several formats with software that is able to access the data and convert it to a format desired by the user.

II. A discussion of data formats.

I suspect that there will never come a time when there is total convergence on a single set of formats for representing musical data. Reasons:

1) Music is complicated to represent, and as a result, representations tend to be application dependent.

2) As a corrolary, representations that attempt to be universal are complex and difficult to work with. For specific applications it is often easier to develop a new representation than to try to use an existing one.

3) There will always be commercial opportunities in this field, and commercial enterprises will often have their own proprietary ways of representing musical data.

4) This is a field which is still in its early development. We can expect that new knowledge will change the way we think about things and the way we do things. This will lead to "improved" methods of music representation and new data formats.

III. A discussion of databases

Just as with formats, I suspect that there will never come a time when all available musical data will be found in one massive database. I view musical databases as organic entities, which will grow, flourish and eventually die -- to be replaced by newer forms. The data, itself, will continue on, transformed as it moves from one database to the next.

At the present time, there are probably somewhere between ten and twenty public or semi-public databases of music which are known to groups of users (I include here both commercial and non-commercial databases). In addition, there are probably hundreds, if not thousands, of private databases of music, developed by individuals or small groups and used exclusively by them (I am not including in this group individuals who have built private collections by simply downloadeding or purchasing sets of data).

IV. Where the CCARH databases fit into the present picture.

The CCARH databases are based on a SOLAR concept of data representation. The idea is that the music be represented in an archival SOURCE format and that from this format various other application specific databases can be derived. The representation of music may be divided into two types of elements: those elements which are fixed, or static, and those elements which require some form of interpretation in their represention. The idea is that those elements which are fixed, such as for example, the pitchs, durations, articulations, etc. as notated by the composer, be represented in the SOURCE format. Think of it this way. If Bach gives a manuscript to his copyist and asks for a clean score and a set of parts, the fixed elements of this data set are those elements which do not change when they are transferred from manuscript to the clean score -- elements, in other words, that are shared in common. Those elements which are variable, or which are changed as they pass from manuscript to clean score, are elements which we say are subject to interpretation. Indeed, one might say that what distinguishes a good copyist from a bad one (other than speed and accuracy) is the ability to the represent the music on the page in an easily readable form. From experience, I can say that this requires certain interpretive skills.

The MUSEDATA Stage2 format developed at CCARH represents our current best effort to represent classical music in its SOURCE form. The representation takes the form of ASCII text records and is designed to be easily readable (for purposes of correction) and largely self-explanatory. The representation includes records called **print suggestions** and **sound suggestions**. These are NOT part of the source data, but are included to assist our software in the conversion of data from the stage2 format to formats for display (printing) and for sound production (principally MIDI files).

Here is where the picture gets interesting. There are no user applications which use the MUSEDATA Stage2 format directly; the format is too general and too diffuse to be useful for software applications. Why then do we use the SOLAR model? The answer is that almost all applications require musical data that includes some degree of interpretation, and this interpretation will vary depending on the application. If we represent music in a format specific to one application (Leland Smith's SCORE format, for example), we might find it difficult, if not impossible, to convert this data to other formats for entirely different applications. If we dispense with the SOLAR data and keep only the application data, then every time we want to change, update, or make a derivitive version of the data, we will need to do this in all the various formats. And we will have no direct way of checking the various databases for consistency.

Let's look at the case of music printing. Suppose we have a musical score formatted to fit an 8.5 x 11 page, but we want to recast this score to fit an 11 x 11 page. In this case, the system breaks and page breaks will need to be recalculated. This amounts to a "reinterpretation" of the musical data. Most commercially available music printing programs can do this because they represent the music internally in a quasi-solar form. But in most cases we do not have access to this internal representation. Nor would this internal representation necessarily be "MIDI friendly", or "humdrum friendly" (for musical analysis), or easily convertable to the internal format of some other music printing program. Finally, using a data format based on propriatary software could place restrictions on the free distribution of the data itself (for example, you need to buy the program to use the data).

So, the primary source material in the CCARH databases is stored in the MUSEDATA Stage 2 format. Since this data is not application friendly, CCARH maintains several SATELLITE versions of the data for specific applications. It is the question of which satellite versions of the data to maintain, and what corresponding software to develop and distribute that concerns us in this discussion.

V. What is the purpose of maintaining satellite data?

The goal of creating a large body of source data in music is laudible, but the impact will be minor if there are no uses for the data. Commercial outfits have an incentive to develop applications for public use, but they also have an incentive to try to "capture" customers and steer them towards application models which require the continued investment in their software and/or services. If the CCARH data is to have an impact in the real world, there must be satellite data **and** software which is useful to the public. That is not to say the CCARH is anti-commercial; quite the opposite. We recognize that with our limited resources we cannot possibly provide all of the software and services the public wants and deserves. Our goal should simply be to offer some sort of basic alternative to what commercial outfits provide, so that the public has a low cost/no cost starting point to various applications, thereby lowering the barriers to use for many people around the world.

VI. The current situation

The internal CCARH database is a hierarchical tree structure, sub-divided according to COMPOSER - SOURCE - (WORK CATAGORY) - WORK. For each WORK, there are four (possible) sublibraries:

STAGE2, OUTPUTS, DISTRIB, and EDITIONS

We have not fully worked out what the public's view of this data should be. I would suggest there are two possible ways of proceding.

Alternative 1:

Make the data available in essentially the same structure as it is kept internally, that is, organized by (in order) COMPOSER - SOURCE - (WORK CATAGORY) - WORK. Under each work there would be three relevant sub-libraries: OUTPUTS, DISTRIB (which includes Stage2) and EDITIONS.

Alternative 2:

Organize the data first by application, the theory being that most users come to the database for a specific reason, and that reason should be the highest organizing feature. Put another way, a user interested in MIDI files, is going to see more in common between BACH MIDI files and MOZART MIDI files, than between the various forms of BACH data or the various forms of MOZART data. Such an organization might look like this:

APPLICATION:
Sound - MIDI files (both types)
Analysis - HUMDRUM files
Source Data - STAGE2 files
Printing/Export - SCORE files
CCARH Display and Printing - CFT files (includes editions)

The hierarchy under this scheme would be:

APPLICATION - COMPOSER - (WORK CATAGORY) - WORK - (SOURCE)

It should be pointed out that the question of file organization is somewhat academic, since a system of pointers can be used to create the impression of any organization we choose. The public's view of the CCARH data is really determined by how the web page is organized. Nevertheless, a logical organization of the files can help us to think clearly about what we have and what we are doing with it.

VII. Software

It is useful to divide the question of software into three parts: 1) CCARH software used internally, 2) CCARH software made available for public use, and 3) Software from third parties, including commercial software.

1. CCARH internal software

Right now, we have software to accomplish the following tasks

1.	Stage 1 data entry	WBH\ONLINE\NEWMUS\INPUT.Z (with midi enabled)
2.	Stage 1 to Stage2 conversion	BEGIN.Z
3.	Stage 2 supplemental data entry	SS2ED.Z
4.	Conversion to score i-files	AUTOSET.Z MSKPAGE.Z/XMSKPAGE.Z
5.	Conversion to parts i-files	AUTOSET.Z MKPART.Z
6.	Display/printing of page i-files	DSKPAGE.Z/PSKPAGE.Z
7.	Editing of page i-files	ESKPAGE.Z
8.	Conversion to PAGE.MPG files	VSPACE.Z
9.	Compilation of .CFT files from .MPG	DMUSE environment
10.	Display/print of .MPG and .CFT files	DMUSE environment
11.	Conversion of Stage2 to SCORE files	SCWORK\PROGS\AUTOSCR.Z PAGING.Z SCORECON.Z
12.	Creation of MIDI files	MIDICON\MKMIDI.Z
13.	Creation of HUMDRUM files	???
14.	Creation of TIFF files for PDF conversion	DMUSE environment (barely)

Important internal software we do not yet have

15. Automatic creation of TIFF files for convertion to PDF
16. Direct postscript output from page i-files
    We can do this indirectly by first converting to SCORE
17. Conversion from SCORE to Stage2
18. Robust conversion of Stage2 to Humdrum
    We can do this indirectly through Andreas' software

2. CCARH public software

John Brenneise is working on this. His program is called VMUSE and includes in its "backend" the modules from DMUSE that display and print .MPG and .CFT files. VMUSE is designed to run on Windows 2000 and newer MS operating systems.

There is still much design work to be done here. VMUSE needs to be coordinated more closely with the CCARH web page.

3. Third party software (incomplete) description

Software can be divided into two catagories: conversion and application.

a) Conversions

Michael Good	Music XML exchange format
Andreas Kornstaedt	SCORE to Humdrum

b) Applications

David Huron, et. al.	HUMDRUM tool kit
Leland Smith	SCORE music printing
Andreas Kornstaedt	Analysis

PART II: SOFTWARE AT CCARH

I. Creating CFT files for display and print output

I first divide CFT files into three classes: **low-quality** for simple viewing and study purposes, **medium-quality** for viewing and possible performance, and **high-quality** for performing editions.

A. Creating low-quality CFT files.

Low-quality CFT files are distinguished by what they do not have.

a) Movements are typeset separately. No attempt is made to combine movements into an integrated set of pages.

b) The last line in a movement might not be right justifed.

c) Attempts to correct vertical and horizontal placement of musical elements to prevent "clashes" (overstriking) are minimal.

Low-quality CFT score and short-score files are created with the following sequence of actions. For each movement separately,

1) run AUTOSET on the **score** or **short** file groups with notesize = 14. Create page independent i-files in the OUTPUTS/SCORE and OUTPUTS/SHORT directories

2) run MSKPAGE. Create a FORMAT file, and page specific i-files in the OUTPUTS/SCORE and OUTPUTS/SHORT directories

3) run DSKPAGE. Verify that the operation was successful

4) run DMUSE module to create a CFT file for desired movements

Low-quality CFT parts files are created with the following sequence of actions. For each movement separately,

1) run AUTOSET on the **parts** file group with notesize = 21. Create page independent i-files in the OUTPUTS/PARTS21 directory.

2) run MKPART. Create page specific i-files in the OUTPUTS/PARTS21/PAGES/<part name> directory.

3) run DSKPAGE. Verify that the operation was successful

4) run DMUSE module to create a CFT file for desired part set.

For parts such as CONTINUO which may combine a second part with the bass part, the procedure for making a score must be followed. Output directories must be allocated according.

B. Creating medium-quality CFT files.

Medium-quality CFT files differ from low-quality files in that:

a) Movements are typeset sequentially, with the intention that they be combined into an integrated set of pages. After each movement is typeset, the amount of remaining space must be evaluated to determine whether it would be suitable to start a new movement on that page or to begin on a new page. Normally movements will be typeset so that they end at the bottom of a page.

b) The last line in a movement is almost always right justified.

c) Attempts are made to correct the vertical and horizontal placement of musical elements in order to avoid "clashes" (overstriking). Some attempt may be made to fine tune horizontal placement of notes on a system.

d) Lines and systems are vertically adjusted to prevent inter-line clashes. Lines are spaced vertically such that the page is bottom justified. Page numbers are included.

Medium-quality CFT score and short-score files are created with the following sequence of actions. For each movement separately,

1) run AUTOSET on the **score** or **short** file groups with notesize = 14. Create page independent i-files in the OUTPUTS/SCORE and OUTPUTS/SHORT directories.

2) run MSKPAGE. Create a FORMAT file, and page specific i-files in the OUTPUTS/SCORE and OUTPUTS/SHORT directories

3) run DSKPAGE. Verify that the operation was successful.

4) using DSKPAGE or PSKPAGE, determine where horizontal and vertical clashes occur that are solely line or part dependent. Run SS2ED on the relevant stage2 files to correct these problems (by adding print suggestions).

5) run AUTOSET and MSKPAGE again. Run DSKPAGE to verify that operation 4) was successful.

6) now evaluate where the movement ends and the number of pages it occupies. Is the presentation readable? If combined with the previous movement on the same page, is the placement of the start correct? Can the music be compressed onto fewer pages, either by putting more systems on a page or by compressing the music typesetting horizontally? Can the movement be made to end at the bottom of the page?

7) Now circle back to 5) and run AUTOSET again, this time with a different compression factor. Run MSKPAGE again, with possibly different line spacings. Continue this process

until a satisfactory result is achieved in 6).

8) Create an edform file and run the VSPACE program to assemble the page-specific i-files into a set of .MPG files which are bottom justified and include page numbers.

9) run DMUSE module to create a CFT file for the complete set of movements.

Medium-quality CFT parts files are created with the following sequence of actions. For each movement separately,

1) run AUTOSET on the **parts** file group with notesize = 21. Create page independent i-files in the OUTPUTS/PARTS21 directory.

2) run MKPART. Create page specific i-files in the OUTPUTS/PARTS21/PAGES/<part name> directory.

3) run DSKPAGE. Verify that the operation was successful

4) using DSKPAGE or PSKPAGE, determine where horizontal and vertical clashes occur that are solely line or part dependent. Run SS2ED on the relevant stage2 files to correct these problems. (This may not be necessary for most files since it was already done for the score)

5) run AUTOSET and MSKPAGE again. Run DSKPAGE to verify that operation 4) was successful.

6) now evaluate where the movement ends and the number of pages it occupies. Is the presentation readable? If combined with the previous movement on the same page, is the placement of the start correct? Can the music be compressed onto fewer pages, either by putting more systems on a page or by compressing the music typesetting horizontally? Can the movement be made to end at the bottom of the page?

7) Now circle back to 5) and run AUTOSET again, this time with a different compression factor. Run MSKPAGE again, with possibly different line spacings. Continue this process until a satisfactory result is achieved in 6).

8) Create an edform file and run the VSPACE program to assemble the page-specific i-files into a set of .MPG files which are bottom justified and include page numbers.

9) run DMUSE module to create a CFT file for the complete set of movements.

For parts such as CONTINUO which may combine a second part with the bass part, the procedure for making a score must be followed. Output directories must be allocated according.

C. Creating high-quality CFT files.

High-quality CFT files differ from medium-quality files only in that greater care is taken to fine-tune the presentation. Once a page has been created, i.e. the system breaks have been determined, ESKPAGE can be used to move the elements around to achieve whatever result is desired. It should be pointed out that once ESKPAGE has been applied to a page file, that file is in a sense "cut off" from the derivative generation process. Each step in the derivative generation process is documented either by changes in the source files (with print suggestions) or by auxiliary format files (the FORMATS files, the EDFORM files). Up to a point, a musical score can be recreated automatically simply by running AUTOSET, MSKPAGE, and VSPACE. But once ESKPAGE is used to fine-tune the placement of musical elements, the resulting page file contains the only documentation of what was done. For this reason, high-quality CFT files should be built only for purposes of specific editions whose presentation is intended to be of music publishing quality.

There is a slight exception to the above statement. If one needs to make a substantive change to a page file, such as adding a note or a slur or a musical directive, and if this change does not effect the system breaks, it is possible to make the change in the relevant stage2 file, run AUTOSET, MSKPAGE and VSPACE and compare the resulting page.MPG file with the former file of the same name (now saved under a different name). Using the screen editor, it is possible to merge the two files by hand into the desired update. Such an editing process can only be done by persons with expert knowledge of the i-file representation.

NOTES:

At this point, the following tasks need to be done

1) Add the ability to make notesize-specific print suggestions in stage 2 file. Fix AUTOSET to selectively recognize these suggestions. **DON**

2) Add the ability to run AUTOSET with various compression factors. This feature is intended to be used only during the music layout process (parts 5, 6, and 7 of the medium-quality process). Once a compression factor has been determined, this should be included as a print suggestion in all relevant stage2 files. **DON**

3) Make sure that AUTOSET and AUTOSCR are "converged" **DON**

4) Merge MSKPAGE and XMSKPAGE into one source. Make sure that these are "converged" with PAGING **DON**

5) Try adding the feature to XMSKPAGE that records all changes to group positions. Experiment with adding group position data (larr(.,.) data) to the FORMATS file. This feature would allow certain group-wise (horizontal) fine-tuning to be memorialized (preserved) in the derivative generation process. What other edit commands might be included? How far can we take this? What are the downsides?