13 The *HymnQuest* Software: A DARMS Parser for Hymn-Tune Searching

Jim Stanley[1] and Antony Kearns[2]

[1] 2771 Pinelawn Drive
La Crescenta, CA 91214
Jim.Stanley@Jacobs.com
www.hymnquest.com

[2] Stainer & Bell Ltd.
PO Box 110, Victoria House
23 Gruneisen Road
London N3 1DZ, UK
post@stainer.co.uk
www.stainer.co.uk

Abstract

The software described is intended to provide churches and other interested parties with a practical database of hymn tunes and hymn texts currently in use in the British Isles. The structure of the database is discussed as it relates to input, output, and to the speed and quality of searches. The legal apparatus designed to protect copyright owners is described, and issues of platform independence are discussed.

COMPUTING IN MUSICOLOGY 12 (1999-2000), 207–215.

HymnQuest, an application developed for the Pratt Green Trust by the UK music publisher Stainer & Bell, is intended to provide a comprehensive database of hymn texts and tunes currently in use in the British Isles. One of the program's goals is to provide fast searching capabilities for both text and music. When it was first released, in May 2000, *HymnQuest* stored over 12,000 hymn-tune incipits and over 12,000 texts. The incipits are displayed graphically on screen and change as the user scrolls through the database of tunes.

HymnQuest is distributed on CD-ROM by license. At present, *Hymn-Quest* is available only for use in England, Scotland, Wales, Northern Ireland, and Eire. Further expansion of the data resources and legal apparatus will eventually enable representation and distribution in North America.

The entry, retrieval, display, and playback of musical incipits form only one part of the project. Some text-based capabilities are these:

- first lines (original language) of verses, choruses, and antiphons, with linked translations and variants;
- tune names, with cross-references to alternative names;
- meters of melodies, with sub-categorization by metrical foot;
- opening intervals of metrical melodies;
- surnames, forenames, and biographical dates of authors, translators, and composers; also brief biographies and thumbnail portraits of some writers;
- biblical, thematic, liturgical, and seasonal indexes for many texts (including all pieces for which full texts are included in the database);
- details of presence and location in hymn anthologies (270+ books);
- full literary texts for over 12,000 hymns, carols, and congregational songs.

Designed for practical use in churches, *HymnQuest* is the musical complement to a Web site called *VisualLiturgy* [*www.vislit.com*]. Effectively, the *HymnQuest* repertory can be approached from many directions. For example, (1) texts can be searched by their principal themes and suitability for particular seasonal and liturgical purposes; (2) author and composer lists can be searched with a given year of birth or death; and (3) the textual or musical contents of any included hymnbook can be reconstructed.

The commentary offered here deals primarily with elements of the software design and legal apparatus through which materials with a large number of owners can be used cooperatively.

13.1 Design Considerations

While recent developments in database technology have made it possible to store and retrieve non-alphanumeric data such as sound and graphics, these storage technologies still have limitations, and there are still development tools and operating environments that cannot handle such data. As a result, in many instances display systems are reduced to storing and managing massive numbers of graphics and/or sound files, which can result in administrative headaches and increased chances for error.

This application has been developed using Borland's *Delphi* development tool, which uses proprietary object-oriented extensions to the Pascal programming language. These run on 32-bit *Windows* operating platforms. Since the application is designed for a single-user workstation, the *Paradox* database package that comes with *Delphi* was chosen to store the data. *Paradox* provides a subset of Structured Query Language [SQL-92] capabilities that enable the search data-tables by several different criteria. Since the application is distributed on a CD-ROM, optimum solutions to storage questions were essential. It was also intended that the user should not require any special equipment to use the application.

The original plan was to create the musical graphic files using the *Capella* notation package, save them as *Windows* meta-files, and store the meta-files in a binary large-object [BLOB] field.[1] However, binary data fields under *Paradox* assume a minimum block size that is inefficient for storing small amounts of data; thus, there was a lot of wasted space in the files containing the BLOB fields. In addition, MIDI playback for each incipit was needed, and this would have required yet another BLOB field to store MIDI data for each incipit. With the large number of incipits, file space is at a premium, even with much of the data on CD-ROM.

[1] A meta-file stores the machine- or operating-system instructions for drawing an image rather than the image itself. The files can be quite small but are tied to a single operating platform.

We began to experiment with encoding the incipit data using DARMS. Stanley had done experiments in creating a Pascal-based DARMS parser.

Departing from Brinkman's approach (1990), we used a finite-state machine model so that DARMS data could not only be read from a file or database field, but also dynamically parsed as the user-entered DARMS code in an edit box.[2]

[2] A finite-state machine establishes a series of "states" with one-way connections to other states. In the case of the DARMS parser, each state "expects" one of a given set of allowable characters that makes sense in the given context; the next character then puts the machine into a different state with a different set of expected characters.

In addition to the parser, program-level objects corresponding to the notation symbols used were designed and tested. As the user moves to a new database record, the incipit is parsed character-by-character and notation objects are created and put into a linked list in memory. The display process consists of telling each notation object in the list to "draw itself." These objects rely on *Capella*'s own notation font (licensed from WHC Musiksoftware for this application) to produce the visible symbols, rather than drawing the symbols as graphics. At present, the incipits are limited for the sake of simplicity to a single voice, although the parser is capable of understanding multi-staved and multi-voiced DARMS code.

Enough information is encoded into the notation objects so that a procedure could be written to play the incipit through the computer's sound-card. The volume, tempo, and instrument can all be controlled by the program; the amount of separation between the notes is set to a program default. In this way, DARMS is used for both visual and aural data-storage, and neither graphics nor MIDI data are required. This resulted in a savings of at least 90 megabytes of storage space, and no external graphics or MIDI files need to be managed.

13.2 Musical Searches

User input for tune retrieval is entered on a graphical "keyboard." Melodies can be found by entering a tune name. Metrical melodies can be searched by their opening intervals. The facility will be provided to input the opening six notes in any key on a screen keyboard. Over 90% of the retrieved melodies will both be shown by a notational incipit and "performed" on a PC sound card. The details are given in the following section.

Figure 1. *Hymnquest 2000*'s on-screen keyboard.

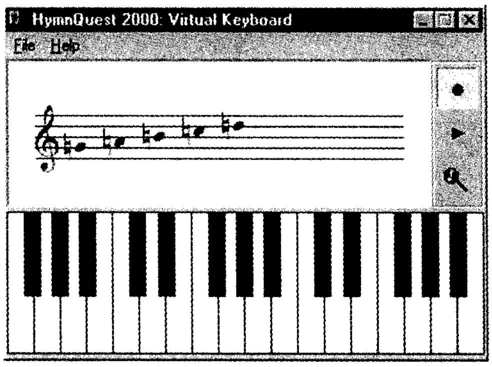
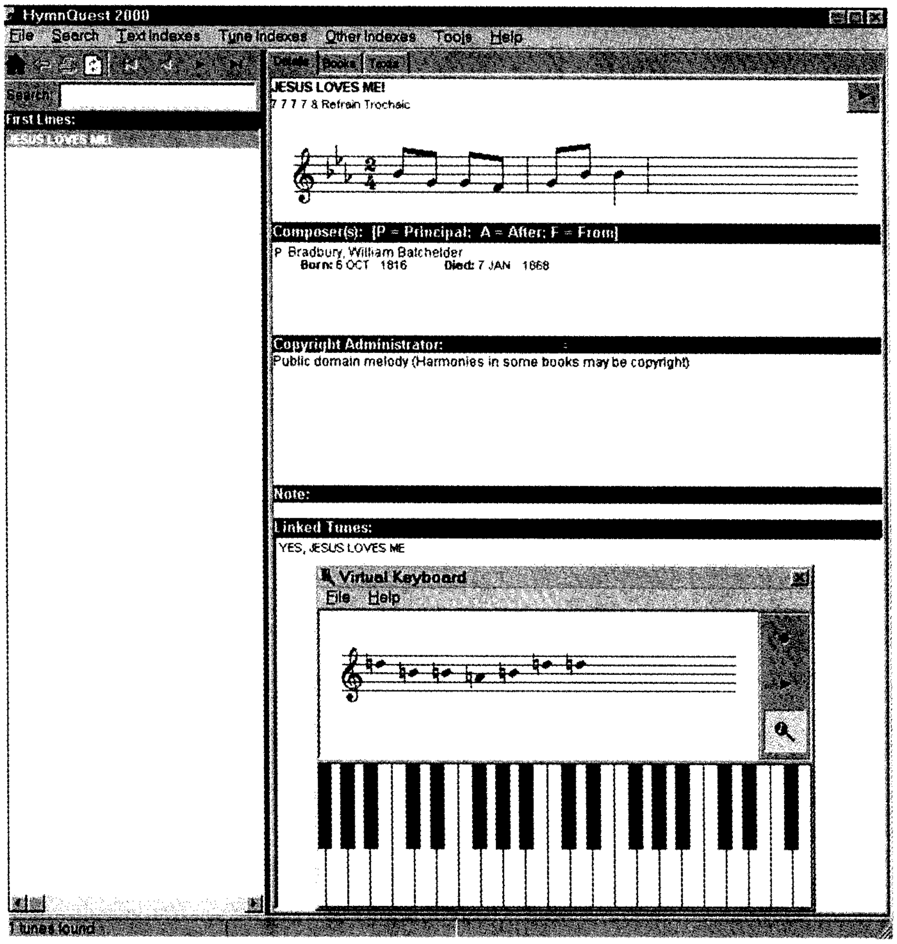


Figure 2. *HymnQuest 2000* search, from an entry in the Virtual Keyboard to the results of the search.



JESUS LOVES ME!
7 7 7 7 & Refrain Trochaic

Composer(s): [P = Principal; A = After; F = From]
P Bradbury, William Batchelder
Born: 6 OCT 1816 Died: 7 JAN 1868

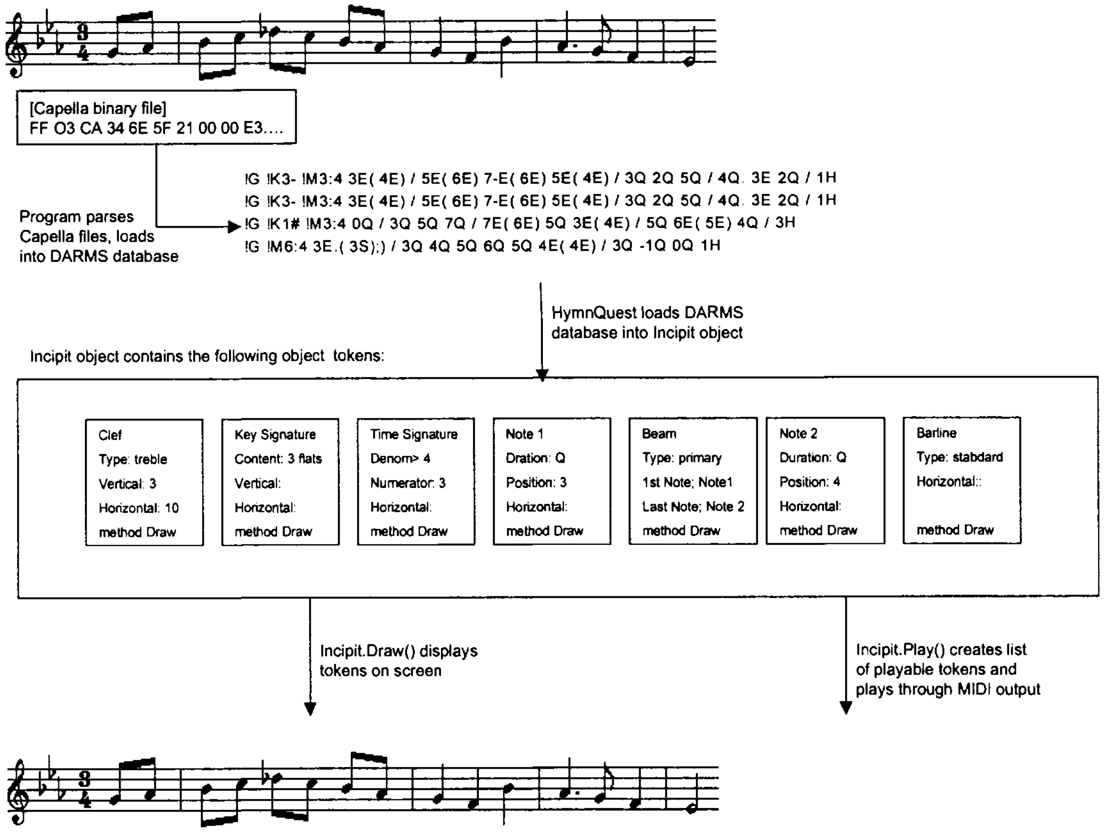
Copyright Administrator:
Public domain melody (Harmonies in some books may be copyright)

Note:

Linked Tunes:
YES, JESUS LOVES ME

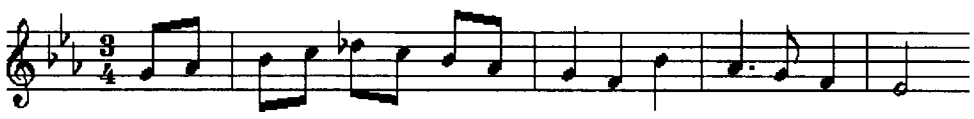
1 tunes found

Figure 3. Procedure for converting Capella binary files (input) to DARMS and then outputting graphical and sound data.

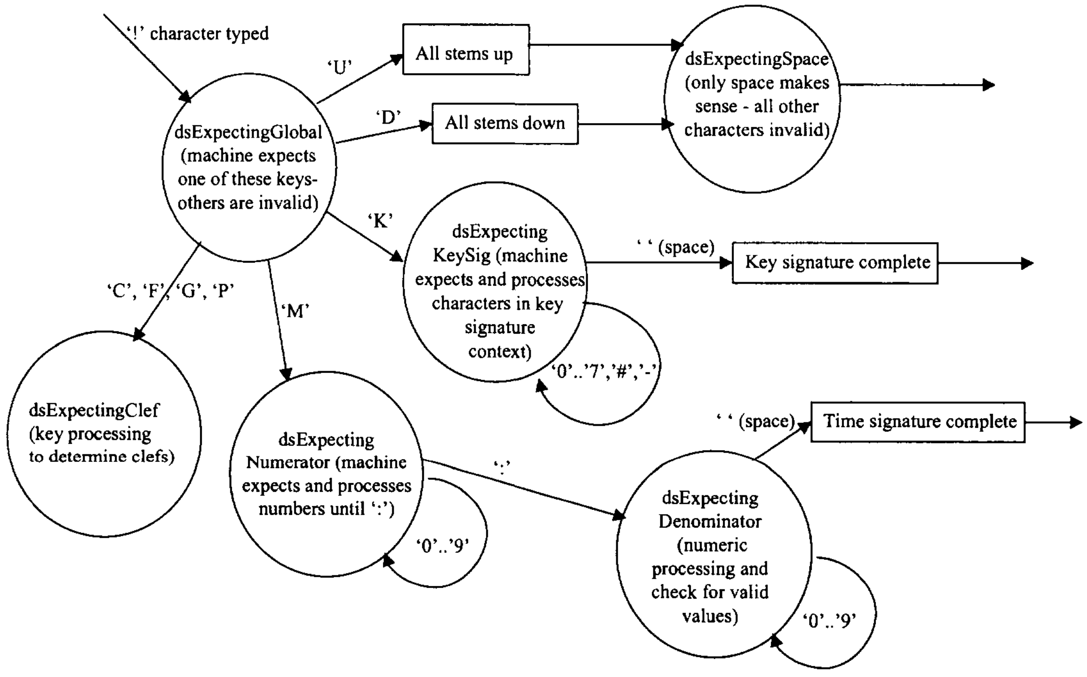


```
!G !K3- !M3:4 3E( 4E) / 5E( 6E) 7-E( 6E) 5E( 4E) / 3Q 2Q 5Q / 4Q. 3E 2Q / 1H
!G !K3- !M3:4 3E( 4E) / 5E( 6E) 7-E( 6E) 5E( 4E) / 3Q 2Q 5Q / 4Q. 3E 2Q / 1H
!G !K1# !M3:4 0Q / 3Q 5Q 7Q / 7E( 6E) 5Q 3E( 4E) / 5Q 6E( 5E) 4Q / 3H
!G !M6:4 3E.( 3S);) / 3Q 4Q 5Q 6Q 5Q 4E( 4E) / 3Q -1Q 0Q 1H
```

Figure 4. Section of the DARMS finite-state-machine parser (simplified).



Section of DARMS Finite State Machine Parser (simplified)


'!' character typed
dsExpectingGlobal (machine expects one of these keys- others are invalid)
'U'
All stems up
'D'
All stems down
dsExpectingSpace (only space makes sense - all other characters invalid)
'K'
dsExpecting KeySig (machine expects and processes characters in key signature context)
' ' (space)
Key signature complete
'0'..'7','#','-'
'C', 'F', 'G', 'P'
'M'
dsExpectingClef (key processing to determine clefs)
dsExpecting Numerator (machine expects and processes numbers until ':')
'0'..'9'
':'
' ' (space)
Time signature complete
dsExpecting Denominator (numeric processing and check for valid values)
'0'..'9'

13.3 Data Issues

For proofing and indexing purposes, the incipits were entered and saved in *Capella*. A program was designed to open the *Capella* files, extract the relevant notation information, construct an equivalent DARMS string, and save the string in a *Paradox* table. After several iterations, few corrections to the DARMS code were needed. The one exception repeatedly encountered was related to beaming issues in incipits with pick-up bars.

In addition to the DARMS data, the first five intervals of each incipit are stored numerically in the table. A second incipit display attached to a virtual keyboard allows the user to enter a series of notes to search the database. The program constructs a dynamic SQL statement to retrieve the records matching the given interval vector. The number of intervals stored was determined heuristically. Kearns has found that searching on all five intervals generally narrows the search down to a set of usually less than 15 records.[3]

[3] See also John Howard's account (1998: 121–123) of Joachim Schlichte's work in this area with the RISM Zentralredaktion musical-incipit data.

Since the table has a secondary index on the intervals, the search is quite rapid and does not require a full table scan. A fuzzy search algorithm based on the work of Smith et al. (1998) has been proposed. At present, we are trying to construct a method of fuzzy searching that would not require a full table scan.

For specific operating platforms that support notation fonts, it is actually quite easy to generate simple, one-line incipits. The process is more difficult if the operating platform is not known. Since Web pages and the Java language are designed to be platform-independent, Java in particular does not support the concept of a notation font, and it would be necessary to draw the notes as graphics to preserve platform-independence in the user's (unknowable) environment. However, it should be possible to use current database technologies to dynamically create graphics files from encoded music and thus enable the display of music over the Web with much smaller and more maintainable resources.

13.4 Legal Apparatus

A routine that requires the user to respect copyright, where it pertains, is an essential part of the *HymnQuest* software. One text field either (a) reports that the work retrieved is in the public domain, or (b) gives the name of the copyright holder and his/her acknowledgement that the material may be used in the manner supported by *HymnQuest*, or (c) states that permission of the copyright holder is pending.

The following authorizations are granted:

> For public domain texts (and others where the copyright holder has so requested): the ability to read from screen, print, copy to a word-processing program, or make available to *Visual Liturgy*.
>
> For almost all of the remaining copyright texts: the facility to read the full text on the screen (encryption ensures users cannot make copies digitally and warns against making illegal copies by laborious copying without consent) and the ability to make sophisticated searches for words/phrases in all verses of the texts.
>
> The ability to read from screen, print, copy to a word-processing program, and make available to *Visual Liturgy* both public-domain and copyrighted texts by participating copyright administrators and individuals.

The copyright administrators in the United Kingdom are named, where appropriate, under the musical incipits.

General enquiries may be made to The Pratt Green Trust, Rev. Canon Alan Luff (Chairman), 12 Heol Tyn Y Cae, Cardiff CF4 6DJ, Wales, UK; fax: 01222 616023; e-mail: *luff@globalnet.co.uk*

References

Brinkman, Alexander (1990). *Pascal Programming for Music Research.* Chicago: University of Chicago Press.

Howard, John (1998). "Strategies for Sorting Musical Incipits," *Melodic Similarity* (*Computing in Musicology* 11), 119–128.

Selfridge-Field, Eleanor (ed.) (1997). *Beyond MIDI: The Handbook of Musical Codes.* Cambridge, MA: MIT Press.

Smith, Lloyd A., Rodger J. McNab, and Ian H. Witten (1998). "Sequence-Based Melodic Comparison: A Dynamic-Programming Approach," *Melodic Similarity* (*Computing in Musicology* 11), 101–118.