The *JRing* System for Computer-Assisted Musicological Analysis

Andreas Kornstädt
Arbeitsbereich Softwaretechnik (SWT), Fachbereich Informatik, Universität Hamburg
Vogt-Kölln-Straße 30, 22527 Hamburg, Germany
++49 40 69424-200
kornstae@informatik.uni-hamburg.de

ABSTRACT

Among other factors, high complexity and mandatory expert computer knowledge make many music IR and music analysis systems unsuitable for the majority of largely computer-illiterate musicologists. The *JRing* system offers highly flexible yet intuitively usable search and comparison operations. to aid musicologists during score analysis. This paper discusses the requirement analysis that led to *JRing's* inception, its IR tools and graphical user interface plus the kind of musical material it works on and the *Humdrum*-based technical realization of IR operations.

1 USER NEEDS

JRing was conceived as set of tools to assist musicologists during that kind of score analysis which aims at:

- a score in which all musicologically relevant elements are completely marked up
- a catalogue which contains all occurrences of all of these elements in complete form plus an arbitrary set of annotations

Depending on the type of work and / or analysis, the elements could be themes, leitmotivs or sets. At the beginning of the *JRing* development process, musicologists at the University of Hamburg and Stanford University were asked to specify the kind of computer assistance they would like to have during analysis. The five results that directly or indirectly pertain to IR were:

(1) Print-like rendition of all musical materials (score, excerpts, search results) as the basis for identifying and comparing elements optically.

(2) Search capabilities for finding elements by arbitrary combinations of musical features (pitch, harmony, and rhythm in various forms).

(3) Tools that help to create catalogue entries, comprising (a) making excepts and (b) filling in information about the elements that can be automatically derived from the excerpt such as set class or position within the score.

(4) Catalogue management capabilities to sort and filter catalogue entries according to certain criteria.

(5) Customization of the structure of catalogue entries and consequently the search and comparison operations based on them.

Other – non IR-related – requirements included the ability to switch back and forth between different kinds of analyses plus a maximum degree of platform independence.

It becomes evident from the composition of the set of requirements what at least the musicologists that took part in the development of *JRing* do aim for. It is not a “big automaton” that can be fed with a score and some kind of theory description and that churns out a results that has to be interpreted by the musicologist [1, 7]. Instead, what is asked for is a set of tools that leaves the analyst permanently in charge of analysis decisions and that merely assist him in making choices faster and with less effort. The basic ways of traditional, manual analyses should not be changed.

2 SOLUTION COMPONENTS

A comprehensive solution that meets all the above-mentioned requirements can hardly be furnished single-handedly. Although there is no adequate reusable and platform independent graphical user interface, many results in the area of data storage and retrieval can be incorporated and made accessible through a new user interface.

2.1 Data

The foremost problem is a lack of data to be analyzed that is available in an appropriate format. Although MIDI and score notation data is widely available, these formats are ill-suited for analysis and musical IR.

- MIDI data focuses on pitch while durations are often not quantized. As MIDI is mainly intended for controlling electronic instruments, enharmonic spelling, articulation, ornamentation, and lyrics cannot be represented among other things. Although there are several extensions of the MIDI file format that try to overcome some of these limitations, none has captured a sizable market share and is thus a good source for widely usable analytical data [9].
- Data from notation programs such as SCORE, Finale or NP-DARMS tends to be layout-oriented and often subordinates logical musical information (C##) to purely spatial descriptions (“black circle between two horizontal lines”). The true pitch can only be determined by scanning backwards for a clef, key signatures, ottava marks, etc. Although, these formats are mostly page-oriented so that musical features that cross page boundaries are very difficult to recognize. Therefore, analytic applications that work on this kind of data often restrict themselves to dealing with works that fit on a single page [8]. Also, they need to implement complex algorithms to extract the logical musical information from the spatial information.

In contrast to highly specialized MIDI and notation formats, analytical formats like *Humdrum* [4] and *MuseData* [3] are better suited for analytic applications. Both have been specifically designed with music analysis and IR in mind. They do not exclusively focus on one single aspect of the score (audible pitch or visual rendition) but offer flexible, extensible encoding schemes that can be used to represent arbitrary musical abstractions in different (parts of) files. Common abstractions besides pitch and duration are melodic contour, rhythmic weight, scale degree, pitch class, frequency, MIDI events or lyrics (full text, syllables and phonetic equivalents). All in all, over predefined 40 *Humdrum* formats exist. As *Humdrum* makes only minimal structural requirements, new formats can be added to encode almost any kind of new musical abstraction that musicologist can conceive.

The separation of different musical aspects within the data representation forms the basis for that kind of search and comparison features that users require according to the results of section 1. Therefore, developers of music analysis and IR programs can make use of these existing analytical formats instead of coming up with completely new encoding schemes.

Because MuseData has been less widely publicized than *Humdrum* and does not split different musical aspects into different files, *Humdrum* is becoming the main target format for conversion programs. If copyright problems can be solved, the vast majority of high quality scores can be converted into *Humdrum* using *FinalSCORE* (by Leland Smith), *scr2hmd* [6] (by this author) and *muse2kern* (by Bret Aarden and this author).

2.2 Information Retrieval

As on the data side, *Humdrum* offers an ideal platform for information retrieval programs. It comes with over 40 specialized UNIX-programs and tools, many of which can work with all file formats. As all formats have the same structure, they can be manipulated and analyzed with a few common tools that – among other things – can assemble individual specialized files into one combined file or extract certain section either by data type (e.g. pitch) or by position (e.g. measures 60 to 70). Analytic questions that pertain to a certain representation can be answered by running chains ("pipes" in UNIX-lingo) of small programs, each of which performs a very limited task. Plugged together in a useful way, these pipes can find answers to quite complex questions such as "Do melodic passages in folk songs tend to exhibit an arch shape?". Because *Humdrum* (1) runs in the UNIX environment, (2) stores its data in ASCII format, and (3) provides a wide range of reference records for encoding non-musical information, the standard UNIX tools for data management (**`find`**, **`grep`**, **`sort`**, **`sed`**, etc.) can be used.

For example, in order to mark instances of a certain pattern in a score by its semitone contour, the following UNIX pipe of commands is necessary:

```
extract –i'**kern' score.krn | semits -x |
xdelta –s = | patt –t MotivX –s = -f MotivX.dat |
extract –i'**patt' | assemble score.krn
```

The complexity of the patterns to be matched depends on the program used:

(1) When the **`patt`** command is used, search patterns are limited to quite simple sequences of tokens. To search for a melodic contour of "same, up 2, down 2", that pattern looks like this:

```
0
+2
-2
```

(2) The **`pattern`** command allows for highly complex search patterns. The following sequence of tokens matches one or more G naturals followed optionally by a single G-sharp followed by one or more records containing one or more pitches from an A major triad the last of which must end a phrase.

```
[Gg]+[^#-] +
[Gg]+#[^#-] ?
([Aa]+|([Cc]+#)|[Ee]+)[^#-] *
(}.*([Aa]+|([Cc]+#)|[Ee]+)[^#-]))|(
([Aa]+|([Cc]+#)|[Ee]+)[^#-].*})
```

(3) A different level of flexibility can be achieved by using the **`simil`** command. It does not only match precise instances of a given pattern but measures the editing distance between the given pattern and the material in the score [5]. The result is a list of numbers between 1 (perfect match) and 0 (no similarity at all) that quantifies the degree of similarity for every position in the score.

Humdrum data and tools form the ideal technical infrastructure for analytic and IR applications. They permit a wide range of analytical and IR operations while being open for custom-made extensions. Especially, *Humdrum* is suitable to realize all those search and comparison features described in the requirements section. The basic modus operandi can be described as follows:

1. Any element (theme / leitmotiv / set) as well as the score is converted into those specific *Humdrum* formats that can be used for conducting searches and comparisons. E.g. semitone intervals, pitch classes, durations and lyrics in syllables.

2. According to the specifications of the user, one or more of these representations are separately processed with the appropriate program (**`patt`**, **`pattern`**, **`simil`**). E.g. the semitone representation is searched for the sequence "same, up 2, down 2" while the duration representation is searched for "`Hap-py birth-day`"

3. The results are merged to form the combined result. For example, only those positions in the score are returned that feature the semitone pattern AND the lyrics.

Despite its benefits, for the vast majority of musicologists, *Humdrum* can only serve as technical infrastructure and not as IR or analytic system itself because it does not provide a graphical user interface that allows analysts to manipulate the score, its elements and catalogues in a way that they are accustomed to. Although there are sometimes GUIs that help constructing the command pipes [2, 10], working with musical materials the traditional way is still not possible.

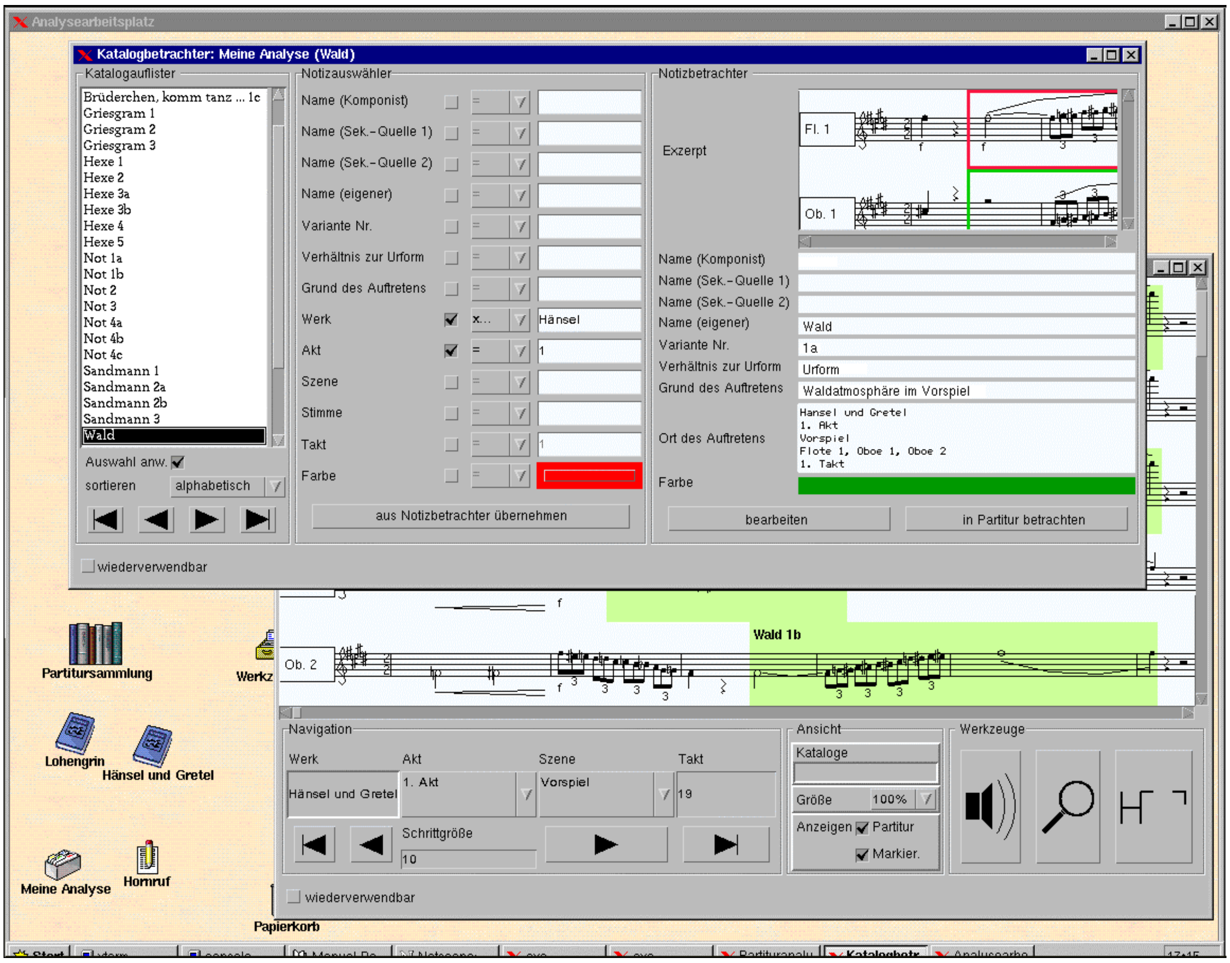
3 *JRING*

JRing aims at providing musicologists with an electronic equivalent of their physical desktop complete with fully graphical scores, thematic / leitmotivic / set notes and catalogues of these elements. Scores, notes and catalogues can be perused in the same way as their physical equivalents, i.e. always with a view of the print-like view of the material. It also offers IR-related functionality that goes beyond what can be done with physical scores and catalogues.

In order to better understand the way in which IR functionality is embedded into *JRing*, its non-IR related features are described first.

3.1 Non-IR-Related Features

JRing works on scores, notes and catalogues.



Katalogauflister

Brüderchen, komm tanz ... 1c
Griesgram 1
Griesgram 2
Griesgram 3
Hexe 1
Hexe 2
Hexe 3a
Hexe 3b
Hexe 4
Hexe 5
Not 1a
Not 1b
Not 2
Not 3
Not 4a
Not 4b
Not 4c
Sandmann 1
Sandmann 2a
Sandmann 2b
Sandmann 3
Wald

Auswahl anw. ✓
sortieren: alphabetisch

Notizauswähler

Name (Komponist)		=	
Name (Sek.-Quelle 1)		=	
Name (Sek.-Quelle 2)		=	
Name (eigener)		=	
Variante Nr.		=	
Verhältnis zur Urform		=	
Grund des Auftretens		=	
Werk	✓	x...	Hänsel
Akt	✓	=	1
Szene		=	
Stimme		=	
Takt		=	1
Farbe		=	🟥

aus Notizbetrachter übernehmen

Notizbetrachter

Exzerpt

Name (Komponist)	
Name (Sek.-Quelle 1)	
Name (Sek.-Quelle 2)	
Name (eigener)	Wald
Variante Nr.	1a
Verhältnis zur Urform	Urform
Grund des Auftretens	Waldatmosphäre im Vorspiel
Ort des Auftretens	Hansel und Gretel 1. Akt Vorspiel Flote 1, Oboe 1, Oboe 2 1. Takt
Farbe	🟩

bearbeiten | in Partitur betrachten

wiederverwendbar

Navigation

Werk	Akt	Szene	Takt
Hänsel und Gretel	1. Akt	Vorspiel	19

Schrittgröße: 10

Ansicht: Kataloge; Größe 100%; Anzeigen ✓ Partitur ✓ Markier.

Werkzeuge

wiederverwendbar

Figure 1. The main components of the *JRing* system: desktop, score analyzer (partly covered), and catalogue browser.

3.1.1 Scores

Scores are displayed in a high quality, print-like rendition. Voices are basically arranged the same way as in the printed score but vertical positions are fixed for every voice. Therefore, the topmost row is always reserved for e.g. the piccolo while the lowest voice always holds the display for e.g. the double base. Therefore, the vertical size of the score is always the same even if some voices are pausing. This grid-like organization of the score makes it easy (1) to browse the score without the need to search for a specific instrument and (2) to mark an occurrence of a specific element in one piece even if it covers one or more system / page breaks in the printed score.

The tool that works on scores is the score analyzer. It displays the score in the above-mentioned way. It has several features:

- The position within the score can be changed by either making use of the scroll bars or by jumping to any logical location such as measure 23 of the 2nd scene of the 3rd act.
- Already marked up elements can be shown or hidden. The ability to see intermediate results gives valuable information on where to look for new elements. In combination with zooming out to, say, a 10% magnification this feature to get an overview over large-scale patterns of elements.
- The score can be searched for arbitrary combinations of musical features (see section 3.2.2).
- Newly found elements can be graphically marked up with a marker tool (see figure 2). Although marks can consist of a single contiguous block, they can be made up of any number of blocks. Marks form the basis for notes.

3.1.2 Notes

Notes describe one occurrence of a musically relevant element of a work. They consist of a graphical rendition of the marked element (see above) plus an arbitrary number of additional fields. As notes used in manual analysis, they contain fields for non- or meta-musical information such as the position within the score, the formal relation of the element to other elements, the reason for

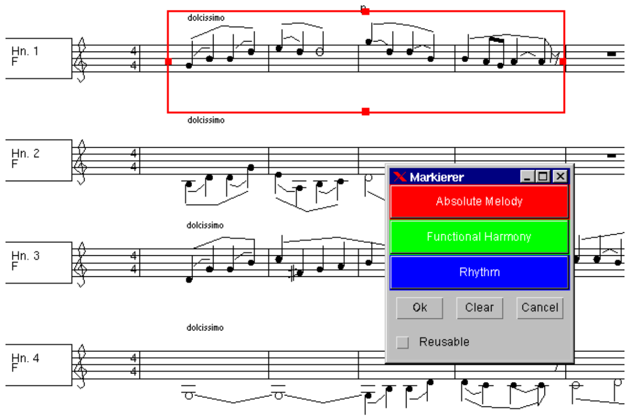


Figure 2. The marker tool.

the element's occurrence or – for leitmotivic analysis – the name of the leitmotiv. Note fields form the basis for comparisons with other elements which are penned down on other notes.

Two tools directly work on notes:

- Note editors pop up after an element has been marked up using the marker tool. Its graphical excerpt and its position within the score are automatically filled in. Information that depends on human judgement (formal relation, reason for occurrence, etc.) needs to be filled in manually.
- Note viewers (see rightmost sub tool in catalogue browser in figure 1) display notes.

A third kind of note-related tool are note selectors. They do not work on individual notes but on catalogues.

3.1.3 Catalogues

Catalogues contain all the notes an analyst furnishes. As a musicologist can find a huge number of elements in a sizable score, tool support for managing a catalogue is need. This support is provided by catalogue browsers.

Catalogue browsers consist of three sub tools: A catalogue lister, a note selector and a note viewer. Basically, the user can select a note in the note lister which is then shown by the note viewer. The note selector can be used to make the lister show only a specific subset of notes from the catalogue.

Note selectors (see central sub tool in catalogue browser in figure 1) are structurally similar to note viewers and note editors: Like these, they list every field of the note and like note editors every filed is editable. Different from these, the values entered into the fields of a note selector do not describe a specific note but a pattern that is matched against all notes contained in a catalogue. If for example "1" is entered into the field named "Act", only notes that pertain to elements from the first act are listed in the note lister.

Although valuable, matching a single criterion does not meet the user requirements stated in section 1 (cf.). Therefore, note selectors offer two additional features for describing note selections more precisely:

1. A combo box to select the matching condition. For textual fields the options are "equals", "contains", "starts with" and "ends with". For numerical fields these are "equals", "less than", "less than or equals", etc.
2. A checkbox which users can employ to indicate whether a field should be taken into account when finding matches or not. In figure 1 this feature is used to show only those notes in the lister that matches certain work names AND occur in the first act.

3.2 IR-Related Features

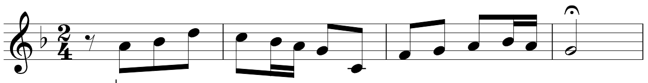
The features described so far replicate the traditional way of dealing with scores, notes and catalogues. Although they facilitate the analytic process considerably by relieving musicologists of strenuous tasks by means of automatically doing excerpts, filling out large parts of notes, and dealing with catalogues but they do not provide any IR capabilities.

JRing provides these capabilities not through completely new tools but integrates them into the tools already described in section 3.1.

3.2.1 Catalogue browser

In manual analysis, the musicologist just needs to write down the excerpt and can then make any kind of comparison based solely on that excerpt. Because he can make arbitrary abstractions of that excerpt, he can simply choose to concentrate on certain abstract features and then browse the other excerpts in the catalogue to find exact, partial or fuzzy matches. As a computer IR system cannot know in which respect two notes should be compared, this choice of the analyst has to made explicit in a computer assisted system. Therefore, in addition to the text fields discussed in section 3.1.2, each note in *JRing* carries a list of musical abstractions that are automatically derived from the excerpt. For example, the pitch information of the marked up element can be used to derive absolute pitch, pitch class, and any melodic contour in semitone steps (see table 1).

Table 1. Some melodic abstractions of a Beethoven theme



pitch	a1	b1-	d2	c2	b1-	a1	g1	c1	f1	g1	a1	b1-	a1	g1
pitch class	9	A	2	0	A	9	7	0	5	7	9	A	9	7
semitone interval	*	+1	+4	-2	-2	-1	-2	-7	+5	+2	+2	+1	-1	-2
refined contour	*	^	/	v	v	v	v	\	/	^	^	^	v	v
gross contour	*	/	/	\	\	\	\	\	/	/	/	/	\	\

In the note selector, these fields with musical abstractions can be used in the same way as text fields in order to determine the notes that are shown in the note lister: The checkbox next to them indicates whether or not a certain field is to be included in the comparison, and the combo box can be used to determine the matching condition (from a 100% perfect match down to 5% similarity).

To make filling in the fields of the note selector easier, the contents of the note displayed in the note viewer can be copied into the note selector and then modified.

3.2.2 Score Searcher

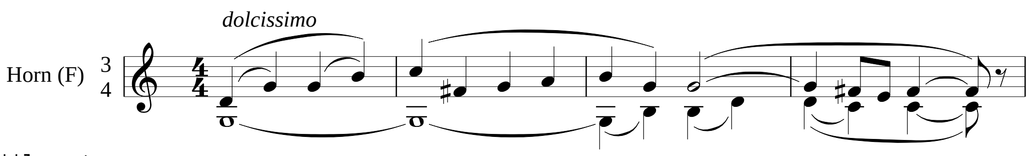
The score searcher is a sub tool of the score analyzer described in section 3.1.1. It basically has the same layout as the note selector, but has only fields for musical abstractions and no text fields except for specifying a search range within the score. When the user chooses to search the score for a certain combination of features, the results are temporarily marked up in the score and a result list browser pops up. Clicking on a result entry in the lister takes the user to the score position of the match. He can decide to discard the result list or can invoke the score marker to make individual results the basis for new notes in the catalogue.

JRing's tools are adequate to fulfill user requirements (1) to (4) stated in section 1. The remaining fifth requirement - customization of the structure of catalogue entries – is the topic of section 4.2.

4 TECHNICAL REALIZATION

4.1 IR-Related Features

As motivated in section 2, *JRing* uses *Humdrum* as its technical infrastructure and does not implement music IR functionality itself. While presenting *Humdrum* data in a way that is tailored to



```
**kern       **layout
*Icor        *SCORE
*Itrd4c7     *
=1           =1
*clefG2      *
*M4/4        *
4G           1 14 13.3602 2 10 0 1|16 14 13.6785 17 1 1 0 0 0 0 0 13 13.2094 dolcissimo|5 14 14.4385 10.5 ...
4c           1 14 19.5083 5 10 0 1
4c           1 14 25.654 5 10 0 1|5 14 27.0518 8 10 32.0118 1.348 -1
4e           1 14 31.7996 7 10 0 1|14 14 37.9542 1
=2           =2
...          ...
```

the needs of the majority of musicologists, *JRing* can be seen as an – albeit very extensive – GUI front-end for *Humdrum*. It transforms user input into *Humdrum* commands and parses the results in such a way that it can be displayed in a form that is understandable for musicologists that are used to carrying out analyses in a traditional form.

The core of the transformation process are the structurally identical files the contain the analytical score information and the graphical layout information. Because they are synchronized, user input that pertains to the graphic representation on the screen can be mapped to a precise portion of analytical information and search results can be mapped back to the graphical score rendition (see figure 3). A search in the score is thus carried out the following way:

1. For every field that the user marked by ticking the checkbox next to it, its contents are interpreted and converted into the appropriate *Humdrum* representation.
2. If the user has not included one of the selected representations, the required abstraction file is generated with *Humdrum*. E.g. `**semits` for semitone intervals.
3. For every representation, a separate *Humdrum* pattern matching tool is invoked. If perfect matches are desired, **`patt`** is used, **`simil`** otherwise.
4. The individual results are merged. Only those hits become part of the comprehensive result that occur in *every* individual result.
5. *JRing* compiles the result list from the comprehensive result. By going from analytical spines to synchronized layout spines, the its precise position within the graphically rendered score can be determined.

Because melodic matching often requires finding roving or hidden themes [9], *JRing* generates an extra file that contains the highest consonant pitches of the whole score. This "melody" file is automatically included into any melodic search.

The technical realization of note comparisons using the note selector is similar to using the score searcher. The only difference is that notes are matched against each other and that results are listed in the note lister instead and not in a separate result lister.

4.2 Customization

Although the list of musical abstractions discussed so far does cover some of the more frequently used features, it is not exhaustive by far. Taking into account that *Humdrum* comes with over 40 data formats and that new formats can easily be added, no tool that comes with a fixed and thus limited set of abstractions / note elements can be very useful. If all possible abstractions are automatically generated, notes become huge and unwieldy. On the other hand, if the "right" abstractions are not offered, the tool is in risk of becoming useless.

Similar cases can be made for the capability to display scores in different formats (mensural, CMN, chromatic) or to work with different *Humdrum* implementations or without any *Humdrum*-subsystem.

To deal with the demand for flexibility, *JRing* can be customized in three ways:

(1) **The structure of notes.** Depending on the type of analysis, notes can be made up of different fields. In a leitmotivic analysis, there might be several name fields for leitmotiv names according to different sources, and musical abstractions from the melodic, harmonic and rhythmic domain. For an analysis based on Forte's set theory, the name of the element can be automatically derived from the excerpt (by determining the prime form and its set class) while there is no need for melodic, harmonic and rhythmic abstractions.

(2) **The type of score rendition.** To represent a score on screen, there needs to be a component that takes a graphical score presentation and displays it. Depending on the type of score notation (mensural, CMN, chromatic, or something completely new), different display components can be selected.

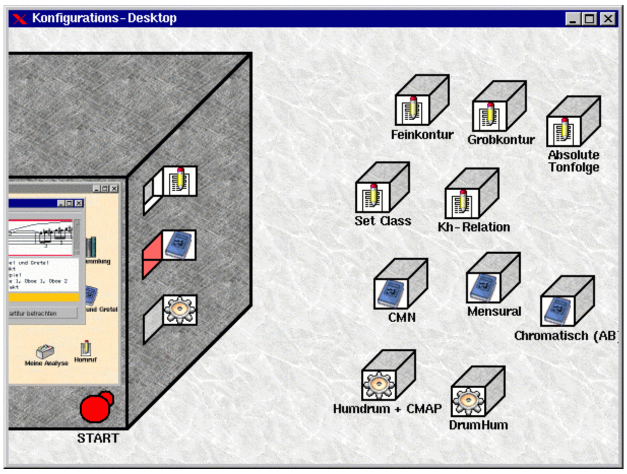
(3) **Type of musical subsystem.** As *Humdrum* consists of UNIX commands, it requires a UNIX shell for operation. This might not be available on non-UNIX systems because the UNIX shell emulators available on these platforms cannot be used free of charge. Therefore, the component that connects to the musical subsystem can be totally absent in some cases or a substitute might be available. If totally absent, *JRing* functions normally except that searches and comparisons based on musical features are disabled.

The source of *JRing's* flexibility is the slide-in approach. Components implementing (1) a single musical abstraction, (2) score renderer, or (3) musical subsystem can be put into matching "holes" of the *JRing* core system at startup time. In contrast to plug-ins or other kinds of dynamic libraries, slid-ins have the following advantages:

1. Slide-ins implement exactly one flexible feature of the system whose functionality actually means something to its users.
2. Slide-ins only fit into the matching slide-in frame ("hole") of the core system, thus making misconfigurations impossible.

3. Individual slide-in frames have specific capacities. While the slide-in frame for musical abstractions can hold any number of slide-ins, the slide-in frame for musical subsystems can have at most one slide-in (either *Humdrum*, a substitute or nothing at all), and the slide-in frame for score renders must have exactly one slide-in.

Because slide-ins are easy to visualize, a configuration desktop is provided that can even be used by those musicologists that would not normally be able to configure technical systems.



As does *Humdrum*, *JRing* just offers a core system that can be easily extended in compliance with its interfaces. In the case of *Humdrum*, these interfaces are the *Humdrum* file format structure and the POSIX standard. In the case of *JRing*, these interfaces are the three slide-in frames (for (1) new abstractions, (2) score renderers, and (2) musical subsystems) plus the Java programming language. As with *Humdrum*, users of *JRing* can profit from its high reuse potential, i.e. when starting a new project, chances are that the required slide-ins already have been written by some one else. If not, just the missing slide-ins need to be implemented and can later be passed on into the pool of available slide-ins so that they can be used in future projects by others.

5 DISCUSSION

JRing meets all requirements listed in section 1 by providing a graphical user interface that can easily be used by the vast majority of musicologists. While this GUI is merely a *Humdrum* GUI from a technical point of view, it adds the user-oriented features of notes and catalogues that are not present in the *Humdrum* world. The system allows for complex searches and comparisons by combining arbitrary musical abstractions in precise or fuzzy searches using combinations of *Humdrum*'s **`patt`** and **`simil`** commands. Due to its compliance with the slide-in approach, it can easily be extended in a fashion similar to *Humdrum*.

The limitation of *JRing* stem from the way in which searches and comparisons can be formulated. While *Humdrum* allows extremely complex pattern definitions (see section 2.2), *JRing* pares down searches to fixed length patterns that can merely be combined with Boolean conjunctions (ANDs). Although patterns can be matched using similarity, not even Boolean subjunctions (ORs) are possible.

Still, this limitation appears to be acceptable as most musicological queries done by traditional musicologists do not require the maximum degree of flexibility offered by *Humdrum*. To make this limitation acceptable to more demanding musicologists, *JRing* maintains all notes as separate *Humdrum* files (with text information as reference records). These files can be used independently from *JRing* in *Humdrum* pipes to make full use of *Humdrum*'s powerful yet difficult pattern matching syntax.

6 REFERENCES

[1] Bo Alphonce, The Invariance Matrix, Dissertation, New Haven, CT, Yale University, 1974.

[2] Peter Castine, Set Theory Objects: Abstractions for Computer-Aided Analysis and Composition of Serial and Atonal Music, Frankfurt: Peter Lang, 1994.

[3] Walter Hewlett, „*MuseData*: Multipurpose Representation“, Eleanor Selfridge-Field (ed.), Beyond MIDI: The Handbook of Musical Codes, Cambridge, MA: The MIT Press, 1997, pp. 402-447.

[4] David Huron, The Humdrum Toolkit Reference Manual, Menlo Park, CA, CCARH, 1994.

[5] David Huron, Keith Orpen, „Measurement of Similarity in Music: A Quantitative Approach for Non-parametric Re presentations“, Computers in Music Research, Vol. 4, 1992, pp. 1-44.

[6] Andreas Kornstädt, „SCORE-to-Humdrum: A Graphical Environment for Musicological Analysis“, Computing in Musicology, Vol. 10, 1996, pp. 105-122.

[7] Guerino Mazzola, Thomas Noll, and Oliver Zahorka, „The RUBATO Platform“, Computing in Musicology, Vol. 10, 1996, pp. 143-149.

[8] Nigel Nettheim, „Melodic Pattern-Detection Using MuSearch in Schubert's Die schöne Müllerin“, Computing in Musicology, Vol. 11, 1998, pp. 159-168.

[9] Eleanor Selfridge-Field, „Introduction“, Beyond MIDI: The Handbook of Musical Codes, Cambridge, MA, The MIT Press, 1997, pp. 3-38

[10] Michael Taylor, Humdrum Graphical User Interface, MA Thesis, Belfast, Queen's University, 1996.