CEPIS UPGRADE is the European Journal for the Informatics Professional, published bimonthly at <http://cepis.org/upgrade>

Publisher
CEPIS UPGRADE is published by CEPIS (Council of European Professional Informatics Societies, <http://www.cepis.org/>), in cooperation with the Spanish CEPIS society ATI (*Asociación de Técnicos de Informática,* <http://www.ati.es/>) and its journal Novática

CEPIS UPGRADE monographs are published jointly with Novática, that publishes them in Spanish (full version printed; summary, abstracts and some articles online)

CEPIS UPGRADE was created in October 2000 by CEPIS and was first published by Novática and INFORMATIK/INFORMATIQUE, bimonthly journal of SVI/FSI (Swiss Federation of Professional Informatics Societies)

CEPIS UPGRADE is the anchor point for UPENET (UPGRADE European NETwork), the network of CEPIS member societies' publications, that currently includes the following ones:

- inforewiew, magazine from the Serbian CEPIS society JISA
- Informatica, journal from the Slovenian CEPIS society SDI
- Informatik-Spektrum, journal published by Springer Verlag on behalf of the CEPIS societies GI, Germany, and SI, Switzerland
- ITNOW, magazine published by Oxford University Press on behalf of the British CEPIS society BCS
- Mondo Digitale, digital journal from the Italian CEPIS society AICA
- Novática, journal from the Spanish CEPIS society ATI
- OCG Journal, journal from the Austrian CEPIS society OCG
- Pliroforiki, journal from the Cyprus CEPIS society CCS
- Tölvumál, journal from the Icelandic CEPIS society ISIP

Cover page designed by Concha Arias-Pérez
"Liberty with Risk" / © ATI 2011
Layout Design: François Louis Nicolet
Composition: Jorge Llácer-Gil de Ramales

Editorial correspondence: Llorenç Pagés-Casas <pages@ati.es>
Advertising correspondence: <info@cepis.org>

Subscriptions
If you wish to subscribe to CEPIS UPGRADE please send an email to info@cepis.org with 'Subscribe to UPGRADE' as the subject of the email or follow the link 'Subscribe to UPGRADE' at <http://www.cepis.org/upgrade>

The opinions expressed by the authors are their exclusive responsibility

ISSN 1684-5285

CEPIS **UP**GRADE

The European Journal for the Informatics Professional
http://cepis.org/upgrade

Vol. XII, issue No. 5, December 2011

Farewell Edition

Monograph
Risk Management
(published jointly with Novática*)
Guest Editor: *Darren Dalcher*

./..

* This monograph will be also published in Spanish (full version printed; summary, abstracts, and some articles online) by **Novática**, journal of the Spanish CEPIS society ATI (*Asociación de Técnicos de Informática*) at <http://www.ati.es/novatica/>.

CEPIS UPGRADE

The European Journal for the Informatics Professional
http://cepis.org/upgrade

Vol. XII, issue No. 5, December 2011

Farewell Edition

Cont.

UPENET (UPGRADE European NETwork)

CEPIS NEWS

IT for Music

Katmus: Specific Application to support Assisted Music Transcription

Orlando García-Feal, Silvana Gómez-Meire, and David Olivieri

This paper will be published, in Spanish, by ***Novática***. ***Novática*** <http://www.ati.es/novatica>, a founding member of **UP**ENET, is a bimonthly journal published by the Spanish CEPIS society ATI (*Asociación de Técnicos de Informática* – Association of Computer Professionals).

In recent years, computers have become an essential part of music production. Thus, versatile music composition software which is well mapped to the underlying process of producing music is essential to the professional and novice practitioner alike. The demand for computer music software covers the full spectrum of music production tasks, including software for synthesizers, notation editors, digital audio sequencers, automatic transcription, accompaniment, and educational use. Since different music composition tasks are quite diverse, there is no single application that is well suited to all application domains and so each application has a particular focus. In this paper, we describe a novel software package, called Katmus, whose design philosophy accurately captures the specific manual process of transcribing complex musical passages from audio to musical scores. A novel concept, introduced within Katmus, is the synchronization between the audio waveform and the notation editor, intimately linking the time segments of the music recording to be transcribed to the measures of the sheet music score. Together with playback, frequency domain analysis of the input signal and a complete project management system for handling multiple scores per audio file, this system greatly aids the manual transcription process and represents a unique contribution to the present music software toolset.

Keywords: Education, Katmus, Music Transcriptions, Open Source Software, Teaching Resources.

1 Introduction and Motivation

Just as books and texts capture thoughts and human speech, musical notation provides a written representation of a complex musical performance. While only approximately perfect, this notational system, in its modern incarnation, provides not only a representation of the full set of notes and their durations, but also key signatures, rhythm, dynamic range (volume), and a set of ornamental symbols for expressing suggested performance queues and articulations, using a complex system of symbols [1].

Within the scope of this paper, musical transcription can be defined as the act of listening to a melody (or polyphony arrangement) and translating it into its corresponding musical notation, consisting of the set of notes with their duration with respect to the in-

Authors

Orlando García-Feal was born in Spain. He obtained an engineering degree in 2008 from the ESEI, *Universidad de Vigo*, Spain. Since 2008, he has worked at the Environmental Physics Laboratory (*Universidad de Vigo*), building, maintaining, and writing software for their large-scale cluster computing facility. He is presently pursuing his Ph.D degree in Computer Science. His research interests include the study of musical signal processing and cluster computing. <orlando@uvigo.es>

Silvana Gómez-Meire holds a PhD from the *Universidad de Vigo*, Spain. She was born in Ourense, Spain, in 1972. She works as full-time lecturer in the Computer Science Department of the *Universidad de Vigo*, collaborating as a researcher with the research group SING (New Generation Computer Systems) belonging to the *Universidad de Vigo*. Regarding her field of research, she has worked on topics related to audio signal analysis and music transcription software development, although at present she is centred on the study of hybrid methods of Artificial Intelligence and their application to real problems. <sgmeire@uvigo.es>

David Olivieri was born in the USA. He received his BSc, MSc and PhD degrees in Physics (1996) from the University of Massachusetts, Amherst (USA). From 1993-1996 he was a doctoral fellow at the Fermi National Accelerator Laboratory (Batavia, IL, USA) studying accelerator physics. From 1996-1999 he was a staff engineer at Digital Equipment Corporation for the Alpha Microprocessor product line. Since 1999, he has been an Associate Professor at the *Universidad de Vigo*, in the School of Computer Engineering (Spain). He has publications in pure and applied physics, computer science, audio signal processing, and bioinformatics. His present research interests focus on signal processing and applications in sound, images and videos. <olivieri@uvigo.es>

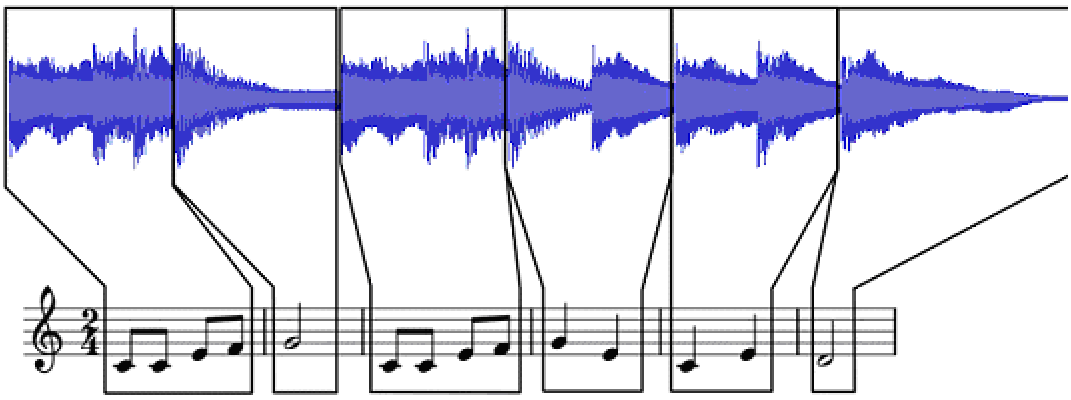


Figure 1: Relationship between an Audio Segment and its corresponding Musical Notation.

ferred time signature and rhythm [2]. Given this definition, Figure 1 shows a short audio signal waveform segment, where different measures with corresponding notes have been identified from the well defined beats or rhythm extracted from the signal. This schematic mapping from the audio signal to the musical notation is referred to as musical transcription.

In the field of music analysis [3], the automatic transcription of monophonic melodies has been widely studied [4] and essentially is considered to be a solved problem. Although more sophisticated machine learning based methods may be applied, simple algorithms for extracting monophonic melodies based on peak-tracking techniques [5] have been shown to be quite effective. This success, however, is not true for the general polyphonic music transcription problem [6]. Indeed, even for the case of a single polyphony instrument such as the piano, automatic transcription methods still perform poorly. For the more general polyphony case, consisting of several different instruments (for example in an orchestra) recorded in the same channel, it is far beyond the capabilities of present transcription systems or, at best, success is limited to special cases.

Thus, while automatic transcription may help in certain situations, manual transcription remains the gold standard for music practitioners wishing to document their own performances or transcribe performances of others. This traditional manual transcription, however, is a time intensive task that strongly depends on the training, musical knowledge, and experience of the person undertaking the process. Indeed, transcription consist of an iterative process: listening to (and normally repeating) short time segments of an audio recording, transcribing the notes in this segments, and then moving on to the next short segment, and often returning to transcribed sections in order to qualitatively evaluate the overall consistency. For those not possessing nearly perfect musical memory and pitch this involves tedious and repetitive interaction with the input audio signal as well as some music notation editor.

While not directly providing automatic transcription, several software tools exist whose purpose is to assist the task of transcription. Some of these tools, such as Noteedit [7], focus more upon facilitating a notational WYSIWYG editor and provide no tools for directly interacting with the audio signal while transcribing. Other systems, such as Transcribe [8], provide direct frequency analysis from the segments of the time domain audio signal, thereby indicating fundamental tones, yet offer no facilities for simultaneously writing the musical score.

Motivated by the shortcomings of presently available software in this domain, the work described in this paper grew out of the need to create a new software application that could aid the process of transcribing music from recorded digital music that would merge the strengths of editing software and those of audio signal analysis. The novelty of our software application and fundamental design criteria is based on the inter-reaction between the notation editor and the audio signal being analyzed, which we directly linked in time. For the user, the present working measure in the note editor is highlighted in a different color on the rendered audio waveform to show this direct correspondence. Not only does this provide for an intuitive user experience, but it really helps transcription, since the user always knows which part of the audio signal corresponds to parts that have been transcribed and which to those parts that are yet to be transcribed.

2 State of the Art

As described in the previous section, other music transcription software is focused either on music score editing or on the implementation of different analysis tools, but not both together.

> “In recent years, computers have become an essential part of music production”

> “In this paper, we describe a software package, called Katmus, whose design allows transcribing complex musical passages from audio to musical scores”

There are currently many software solutions, both commercial and open source, for music notation editing, and they are often presented in the context of a much larger and more encompassing computer music tools suite. However, software applications in the particular domain of automatic music transcription have only modest success and lack many features to make them useful to the music practitioner. Since the number of computer music software and applications domains is large and beyond the scope of this paper, we provide a brief review of software that we consider to be forerunners to Katmus, for the specific purpose of aiding manual music transcription. Thus, following an analysis of these applications, we provide a comparative table summarizing the features of each that are of interest for this task.

Sibelius [9] is a commercial music editing software that is both popular and easy to use. It supports playback of complete polyphony arrangements by using backend instrument synthesis with the use of standard sound fonts, and the basic functionality can be extended through the use of plugin libraries. The Sibelius suite provides a wide array of tools, useful for both the experienced musician as well as the amateur. Moreover, several features are also useful for teaching through functions that allow for the creation of lessons. However, it is not directly designed for manual transcription and does not provide any facilities for frequency analysis or connection with an external audio file.

Finale [10] is another proprietary software tool which is quite complete and widely used by musicians. Like Sibelius, it does not provide the ability to simultaneously interact with the time-domain signal and the notion editor at the same time, nor does it provide frequency analysis of the time-domain signal. It is exclusively geared towards the high quality publication of sheet music scores.

Noteedit [7] is an open source software application that provides similar functionality as its commercial counterparts, just described, for editing scores, as well as saving and exporting midi files. The application is a hybrid between a full sequencer and a simple notation editor in that it requires a real time audio server (Jack on Linux) to provide a patchbay input/output for midi based instruments, that can interface directly with the notation editor. However, for the purpose of transcribing music, this application does not provide useful tools that link the audio file with the editor, nor does it provide the ability for project management.

Rosegarden [11] is a more general purpose open source audio workstation software that provides audio and MIDI sequencing, in the same way as Noteedit, and also provides notation editing. It is focused on providing a wide range of complex audio sequencing functionality and interaction with MIDI inputs found in high end commercial computer music suites. For compositional purposes, Rosegarden contains a complete music score editor which can be connected to MIDI input devices; however, Rosegarden does not have specific facilities for manual transcription.

Transcribe [8] is a software tool specifically designed for aiding manual music transcription. The application supports different audio file formats and provides a graphical interface for the input audio waveform and corresponding frequency analysis. The main window consists of an active piano keyboard with note synthesis so that a user can compare the tone of a synthesized note with that in the input audio signal. The main window also contains the rendered audio waveform and a plot of the frequency domain is superposed on the scale of the piano keyboard so that the peaks in frequency are centered on the corresponding notes of the piano keyboard. Another useful feature of Transcribe is the ability to replay small segments as well as change speed without sacrificing the tone, called time-warping.

With its intuitive design, Transcribe is a lightweight program useful for monophonic melodies, yet practically useless for polyphonic music. Another shortcoming of this tool for more serious transcription tasks is that it does not provide a notation integrated editor that collects the results of the frequency analysis, nor is it possible to add extensions and the software is not open source, so it cannot be extended to include new features.

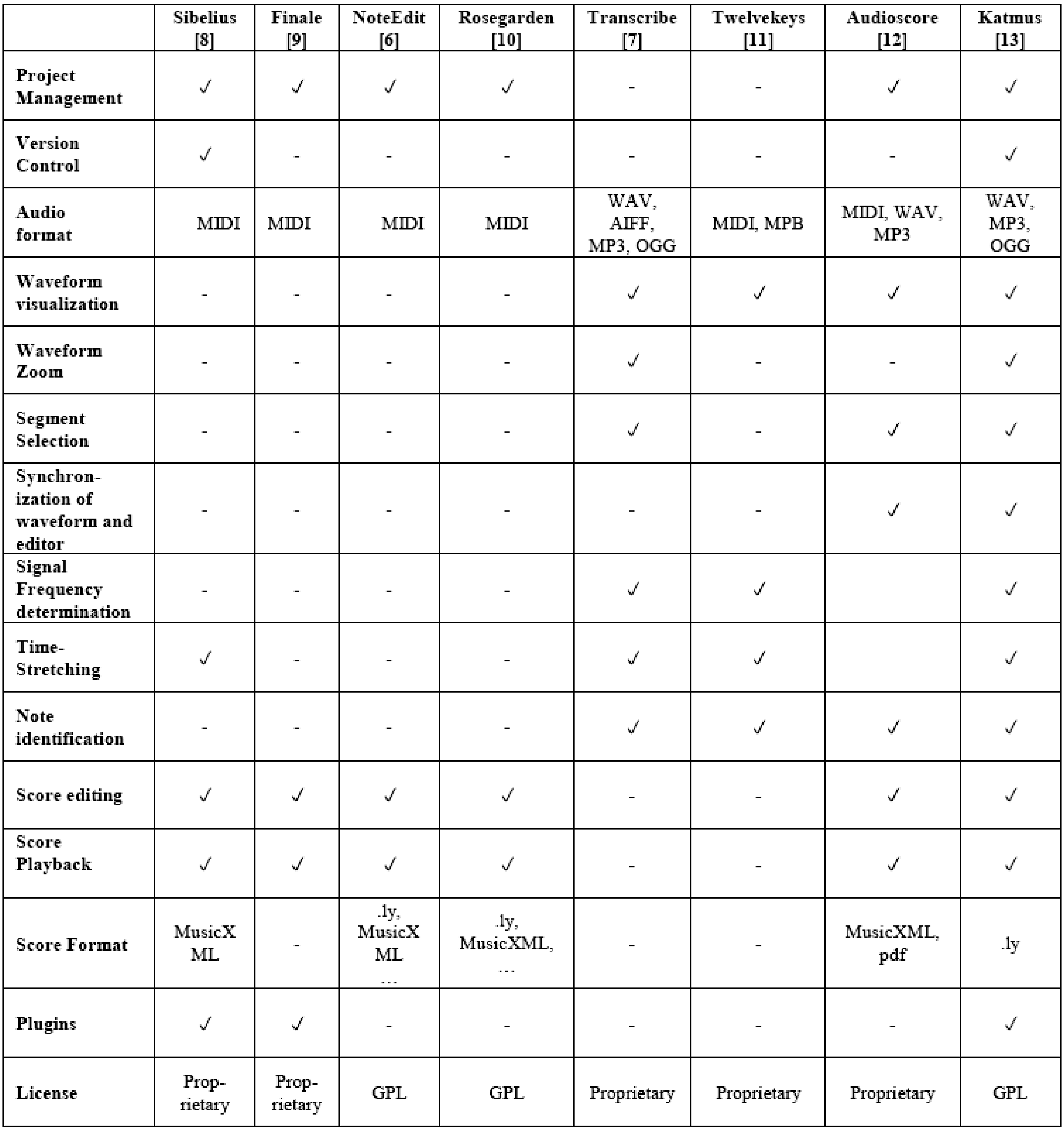
Like the previous tool, TwelveKeys [12] is a proprietary software tool that provides analysis of monophonic and polyphonic music recordings in order to identify notes through the display of the frequency analysis of short time domain segments of the audio file.

> “Musical transcription can be defined as the act of listening to a melody, or polyphony arrangement, and translating it into its corresponding musical notation”

Once again, there is no provision for annotation of the notes identified, so external editing software must be used.

AudioScore [13] is another commercial transcription tool that displays the signal in the time domain together with the frequency domain analysis for note identification. This software does provide an editing environment but, as with all the other software tools described, the audio file is not directly linked to the notation editor, so the user does not know which time domain segment corresponds to a particular measure in the score.

Thus, despite the wide array of software tools for music composition and transcription described, we believe that there is a well defined conceptual gap in the way software tools have approached the problem of transcribing musical pieces, since they ignore the manner in which transcription is normally accomplished. We believe the key concept for a useful transcription software tool is to provide an explicit correspondence between the time



	Sibelius [8]	Finale [9]	NoteEdit [6]	Rosegarden [10]	Transcribe [7]	Twelvekeys [11]	Audioscore [12]	Katmus [13]
Project Management	✓	✓	✓	✓	-	-	✓	✓
Version Control	✓	-	-	-	-	-	-	✓
Audio format	MIDI	MIDI	MIDI	MIDI	WAV, AIFF, MP3, OGG	MIDI, MPB	MIDI, WAV, MP3	WAV, MP3, OGG
Waveform visualization	-	-	-	-	✓	✓	✓	✓
Waveform Zoom	-	-	-	-	✓	-	-	✓
Segment Selection	-	-	-	-	✓	-	✓	✓
Synchron-ization of waveform and editor	-	-	-	-	-	-	✓	✓
Signal Frequency determination	-	-	-	-	✓	✓		✓
Time-Stretching	✓	-	-	-	✓	✓		✓
Note identification	-	-	-	-	✓	✓	✓	✓
Score editing	✓	✓	✓	✓	-	-	✓	✓
Score Playback	✓	✓	✓	✓	-	-	✓	✓
Score Format	MusicXML	-	.ly, MusicXML …	.ly, MusicXML, …	-	-	MusicXML, pdf	.ly
Plugins	✓	✓	-	-	-	-	-	✓
License	Proprietary	Proprietary	GPL	GPL	Proprietary	Proprietary	Proprietary	GPL

Table 1: Comparison of Relevant Software Tools

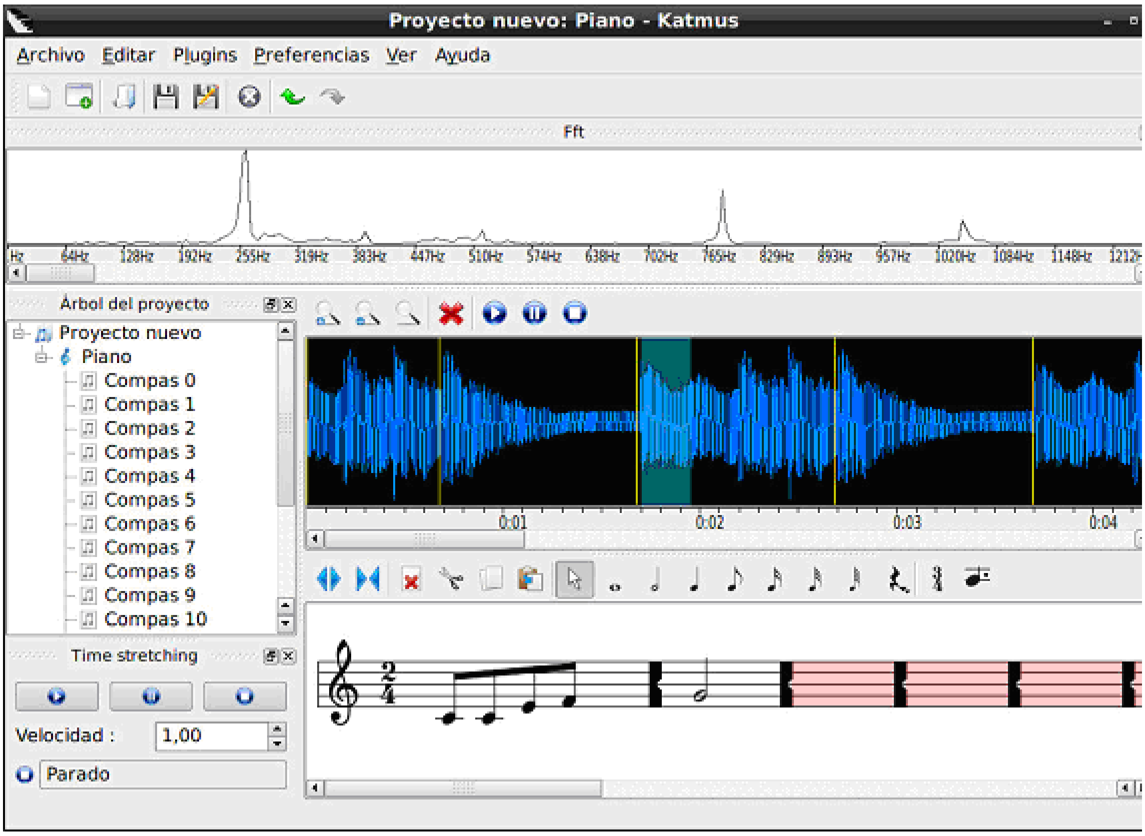


Figure 2: Main Window Workspace in Katmus.

points in audio input waveform with the associated measures in the sheet music score. Our open source software tool, Katmus [14], bridges this gap. Moreover, Katmus not only combines time-frequency analysis of the audio waveform with a powerful WYSIWYG score editor, but also introduces a project based workflow in the process of music transcription. In the Katmus environment, having synchronization between the audio waveform and the note editor means that updates to the score have a corresponding update to the state of the associated time segment of the waveform. In particular, the state of being transcribed or not is represented on the rendered waveform as a color-coded highlight. Thus, an incomplete measure in the note editor is represented on the corresponding segment of the waveform with a different color from a measure that is completely transcribed.

Another important difference between Katmus and other similar software is our emphasis on a project management workflow approach. In this way, a user can save Katmus sessions, thereby saving the present state of all parameters in XML format. With this state persistence, a user can return at a later time to an unfinished transcription and continue at the point of the last session with all the previous parameters restored. Thus, there is no reason for the transcription to proceed in a linear order; large sections can be left untranscribed to be returned to at a later time. Sections that are complex can be marked as unfinished, indicating to the user that these are points which must be revised and/or require further effort. Project creation in Katmus is flexible, allowing for multiple scores per audio file, as well as selecting scores based upon single or multiple staves. As with other full feature notation editors, Katmus provides score playback with sound synthesis support and allows for exporting scores to PDF or MIDI (Musical Instrument Digital Interface).

Perhaps one of the most powerful software architecture features is the plugin management infrastructure, which allows for smaller software modules to be hot-plugged without the

> “There are currently many software solutions, both commercial and open source, for music notation editing”

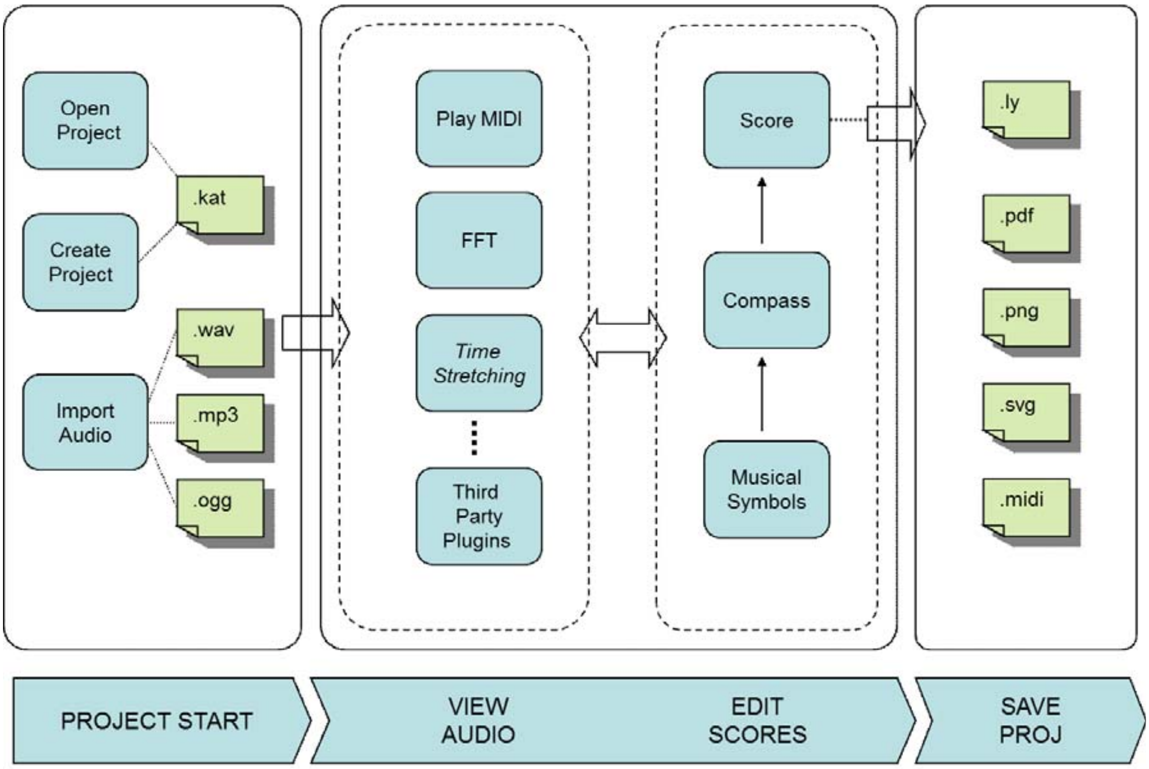


Figure 3: Typical Workflow in Katmus.

need for recompilation of the entire application. This has the advantage that experimental audio analysis and other additional features can be inserted without affecting the underlying software kernel of the application. Several modules have been written using this plugin system, including the time-stretching features found in other similar tools, frequency spectral analysis and filters, and experimental automatic transcription that can provide suggestions to the user. In this way, Katmus can act as a powerful workbench for researchers developing different audio applications related to musical analysis.

Table 1 shows a comparative summary of the different tools that have been described in this section, including the most relevant parameters for the specific task of transcription.

Table 1 provides a comparison which helps describe the advantages of Katmus, our application for musical transcription, showing strengths and weaknesses of other applications with respect to this problem domain.

3 Environmental Features of Katmus

As described in the previous section, the novel aspect of Katmus is the tailored workflow for helping users transcribe complex musical compositions. This workflow consists of project management, and a graphical interface that exposes a WYSIWYG notation editor coupled to a graphical representation of the time domain signal of the audio signal. Also integral to this graphical interface is the ability to listen and apply various analysis algorithms to transcribe the musical arrangement, displaying at all times the correspondence between what is written, where it appears in the audio signal and score, and playback.

Figure 2 shows the main window of the Katmus application. The top panel displays the representation of the frequency domain audio signal, the middle panel the time domain signal, and the bottom panel is the corresponding score. The representation in the frequency domain helps to identify the notes present in the audio segment. In the plot representing the audio waveform, it is possible to manually mark the limits of the measures, thus link-

> “Katmus is a software platform designed to help musicians, professionals, teachers and students with complex music transcription tasks”

	Usage Case	Background	Sequence of Events
Project	*Open*	Request opening an existing Katmus project that contains different scores containing various notes, chords, ties and other stylistic symbols, and/or an audio file with format wav, mp3 and/or ogg. This project may not contain errors, be corrupt or have associated audio files.	1. If the source file is correct, the application should properly load the project with all source elements. 2. If the source file has errors, the application should display a dialog box indicating that the file is corrupt or not found.
	Close	A request is made to close a project.	1. Confirms or cancels the closure of a project. 2. Asks the user whether to open or create a new project.
	Create	A request is made to create a new project.	1. This opens the project creation wizard. 2. Features are introduced into the new project. 3. The option is available to return and update previously entered element values. 4. Loading an invalid file will display an error message.
	Save	Create a new score in the project with different elements (time signature, notes, and other symbols).	1. The project is saved with the original elements and the updated score. 2. If the project has been previously saved, the name can be changed.
Score	*Delete*	A score can be selected and removed from the project tree.	1. It shows the dialog box for confirmation and removal. 2. If cancelled, the process is aborted. 3. If the score is the only one associated with the project, it cannot be removed.
	Export	The user wants to export a score to be rendered.	1. The user can choose the format in which to export the score. 2. If the file exists, it can be overwritten.
	New	The user wants to add a new score to the project.	1. A dialog box is displayed for confirmation. 2. The name and score type is entered.
	Rename	The user wants to change the name of a score in the project.	1. The score is selected from the project tree. 2. The rename option is chosen and the new name entered. 3. If confirmed, the score is renamed, otherwise the current name is retained.
	Playback	The user wants to play back a selected score.	1. An instrument is chosen. 2. The score is reproduced with the chosen instrument. 3. The user can choose the same controls as with the reproduction of the audio signal.
Notes	*Delete*	The user selects one or more notes of a measure, or several measures, to be deleted.	1. The selected notes are deleted.
	Copy	The user wants to select one or more notes of a measure, or several measures, to be copied to another measure.	1. All selected notes are copied and pasted into the desired measures.
	Cut	The user wants to select one or more notes of a measure, or several measures, to be cut and pasted into another measure.	1. The cut notes are removed and pasted in the desired measure.
	Insert	The user wants to select a note or a stylistic symbol to be inserted in the score.	1. If the score does not have any measures, then nothing happens. 2. If measures exist, the selected elements are inserted into the desired measure.
	Paste	The user wants to paste one or more notes previously copied.	1. An empty space is selected and the notes are copied. 2. If the user tries to paste over an existing note or outside the present measure, then the process is aborted.
	Select	The user wants to select one or more notes in one or more measures.	1. The user selects the note and the background color changes. 2. If an empty zone is pressed, the present selection is undone. 3. Selection of multiple notes is performed by selecting the first and dragging the cursor to the last note desired. 4. All notes of a measure or pentagram are selected by selecting an empty zone prior to notes desired and dragging the cursor to

Table 2.Tests based upon Scenarios.

ing the original signal with the note editor.

The application can import audio files that can be played and will be used for analysis during the transcription process. The interface also allows you to incorporate Katmus imported audio files corresponding to transcriptions in the form of scores. The major features that Katmus provides the user with are the following:

- Import and associate an audio file to be transcribed within the project. Several audio file formats are supported, including uncompressed wav, mp3 and ogg vorbis.
- Associate different transcriptions to the same audio file within a single project. This feature allows the user to save and maintain several transcriptions of the same audio file or to have individual transcriptions for different instruments.
- Zoom capability in both the audio waveform and notation editor, which allows the positioning of precise selections of audio segments for fast musical passages.
- Synchronization of the audio waveform with the measures in the score, thereby associating each audio segment with a compass. Combined with a color coding, this provides a powerful functional advantage since completed and uncompleted parts of the audio file and/or score are indicated, saved and restored for multi-session work.
- Playback of the audio signal. The user can replay the entire signal or certain segments, selected by dragging the mouse over the graphical representation of the audio signal. A powerful feature for transcription is pitch invariant time-stretching, where time domain segments can be slowed down without effecting the pitch. This feature is especially interesting for rapid musical segments or in cases where complicated chords (polyphony) need to be resolved.
- Edit scores. The integrated notation editor provides basic functionality for the editing of musical symbols.
- Play back the score. This functionality makes it possible to compare the original melody with the tune of the current work.
- Export scores. Supporting formats are PDF, MIDI, Lilypond, SVG or PNG.

4 System Description and Technical Specifications

One of the fundamental aspects of the Katmus development and philosophy has been the use of open source software for its implementation, thereby encouraging future contributions from a wider community of developers. The application is written in C++ and makes extensive use of the open source Qt4 graphical interface library (originally developed by Trolltech and now owned by Nokia). The significant advantages of the Qt4 library are that it is cross-platform, object-oriented, and provides extensive technical documentation.

> “The novel aspect of Katmus is the tailored workflow for helping users transcribe complex musical compositions”

Since Qt provides a complete framework for developing applications, the core capabilities and functionality of Katmus rely heavily upon the standard and advanced features of the library. Some noteworthy features provided by Qt in the Katmus application are: (i) the use of the Plugin Manager API for developing shared modules, which extends the basic functionality of an application and encourage third party contributions and experimentation, (ii) the use of the specialized Qt thread classes, which can greatly accelerate the applications performance on computer architectures that can take advantage of multi-threading, (iii) the use of interoperability with the use of XML document exchange through standard SAX and DOM technology, and (iv) a clean implementation of object event callback handling with the use of the signals and slots paradigm, characteristic of the Qt framework.

Project management in Katmus is implemented with the use of XML through a DOM implementation offered explicitly in the Qt4 library. In order to produce high quality score rendering, scores are exported using a Lilypond based file generator [15] which produces the specific language syntax for post-processing by the Lilypond compiler that produces the desired output format (PS, PDF or MIDI). Within the application, the audio signal is played by invoking the libao library [16]. This library is cross-platform and provides a simple API for audio playback that can be used internally or through different standard audio drivers such as ALSA or OSS. Playback of scores is done with the use of the Lilypond syntax generator to generate MIDI files and uses Timidity++ [17] for sound synthesis. The slow motion playback is programmed using the Rubberband library [18], which implements a phase-vocoder that can change the speed of original musical audio in real time without affecting the pitch. Finally, to obtain the frequency domain from the time domain of the audio signal, the popular open source Fourier transform library, fftw3 [19], is used.

Figure 3 shows the typical workflow of Katmus, which is a standalone application with a complete user interface. The first action when launching the application is either to create a new a project, select an existing project saved on disk, or start a default project by directly importing an audio file.

Since many different options can be used to instantiate a new project, a graphical wizard guides the user through the process of creating a new project. The information queried during this process includes: (i) the type of audio file to be imported, (ii) the channel (if stereo), (iii) the type of score (one or two staves) and (iv) the name of the project. Once the project is successfully created, the standard work area of the application is instantiated, which consists of three discrete parts, shown in Figure 2, and described as follows:

1. *Display window for audio signal:* Provides zoom capability to change both the time as well as amplitude scale of the audio waveform, which is an important feature for transcription. Thus, the user can focus upon short time scale segments of the audio waveform. An important feature is the ability to select the time intervals corresponding to the different measures of the score, which are marked in the display window by vertical lines. Since there can be variability of time meters within a musical composition, Katmus offers two different ways of making the correspondence between measures and the times in the audio signal: (i) manual selecting the limits of each measure in the signal display window by mouse point/click events, or (ii) assigning a constant time duration for all measures, and then making small tweaks to the duration of individual measures where necessary. The user may also interacts with the waveform display window by selecting small segments with mouse drag events. In this way, all the points included in the selected waveform segment can be used for subsequent analysis with built-in functions, plugin functions, or repetitive playbacks.

2. *Intelligent Score Editor:* Once the measures and the time signature are defined, the user can insert the various musical notes and symbols corresponding to the transcription. An important feature of the editor is that the time computation is automatically validated so that only measures with the correct time signature can be marked as complete.

3. *Project tree:* Displays the different elements, such as scores, measures and audio file, which are part of the complete transcription project. As described previously, an advantage of the project paradigm is that it provides an intuitive way of allowing Katmus to contain many transcription scores for a single audio file, thereby containing different versions of a transcription or assigning different musical instruments to each separate score.

> “ Katmus is available at SourceForge since July 2009. Since then, hundreds of users have successfully downloaded and installed the application ”

In order to ensure the proper functioning of the system described and the quality of the software, several evaluation tests were performed. There are numerous methods of evaluation in the literature that ensure the quality of software development based upon the product type and metrics [20]. For Katmus, both functional and structural tests were performed focusing specifically on object-oriented systems.

The method chosen for these tests is based upon scenarios, since it focuses upon actions that the user performs in order to discover interaction errors. This means that tasks performed by users must be captured in a series of user cases and any possible variants which may arise. Tests are then performed on this set of cases.

Table 2 shows set of tests based on usage scenarios of the application. Applying each of these scenarios to the Katmus software system resulted in a thorough method for debugging the application.

Functional, or black box testing, was applied to the user interface for testing usage cases. The evaluation was based on informal handling tests following the evaluation cycle of the interface [21] during which users evaluated beta versions of the software in order to provide informal feedback for debugging the final design.

Once extensive usage tests and bug fixes were performed, Katmus was made available to the wider user community at SourceForge in July 2009. Since then, hundreds of users have successfully downloaded and installed the application without significant incidence.

5 Conclusions and future work

This paper presents the architecture, implementation, and philosophy of Katmus, which is an easy to use software platform designed to help musicians, professionals, teachers and students with complex music transcription tasks. For the expert musician, the Katmus philosophy and implementation provides a natural mapping of the transcription process which is a great aid to producing the final arrangement. For the more novice users, Katmus provides facilities that reinforce the recognition of musical notes and chords and may serve as an educational tool.

The end result is open source transcription software that is both intuitive and easy to use. Moreover, the application is easily extendible with the use of a plugin architecture that simplifies the addition of enhancements or experimental algorithms. By taking advantage of open source philosophy, future code enhancements and extendible modules could be provided by community contributions in the following areas:

- Extending the capabilities of the notation editor.
- Using beat detection algorithms for accurately associating measures.
- Extending the support for sequencing MIDI to handle more complex polyphony output.
- Extending the labeling system for measures.
- Providing support for multi channel audio analysis.
- Module development for audio signal editing.

References

[1] R. Bennett. Elementos básicos de la música. Ed. Jorge Zahar, 1998.

[2] K.D. Martin. Automatic transcription of simple polyphonic music: a robust front end processing. MIT Media Laboratory Perceptual Computing Section. Technical Report No. 385, 1996.

[3] M. Pizczalski. A computational model of music transcription.

PhD. Thesis, University of Michigan, Ann Arbor, 1986.

[4] A. Sterian. Model based segmentation of time-frequency images for musical transcription. PhD. Thesis, University of Michigan, 1999.

[5] W. Hess. Pitch determination of speech signals. Springer-Verlang, New York, 1983.

[6] A. Klapuri and M. Davy. Signal processing methods for music transcription, Springer cop., New York, 2006. doi=10.1.1.1.4071

[7] Notedit. <http:// noteedit.berlios.de/>.

[8] Transcribe. <http:// www.seven thstring.com/>.

[9] Sibelius. <http://www.sibelius.com>.

[10] Finale. <http://www.finalemusic.com/>.

[11] Rosegarden. <http:// www.rose gardenmusic.com/>.

[12] Twelvekeys. <http:// twelvekeys.softonic.com/>.

[13] Audioscore. <http:// www. neuratron.com/audioscore.htm>.

[14] Katmus. <http:// katmus. source forge.net/>.

[15] Lilypond. <http://www.cs.wisc.edu/condor/>.

[16] Libao. <http:// www.xiph.org/ao/>.

[17] Timidity++. <http:// timidity.sourceforge.net/>.

[18] Rubberband. <http://www. breakfast quay.com/rubber band/>.

[19] Fftw3. <http://www.fftw.org/>.

[20] A. R. Hevner, S. T. March, J. Park, S. Ram. Design Science in Information Systems Research Management Information Systems Quarterly, Vol. 28 No. 1, 2004

[21] R.S. Pressman. Software Engineering: A Practitioner's Approach. McGraw Hill, 6th edition, 2006.