AISB'01 Convention

21st- 24th March 2001
University of York

Proceedings of the AISB'01 Symposium on Artificial Intelligence and Creativity in Arts and Science

Published by

The Society for the Study of Artificial Intelligence and the Simulation of Behaviour

http://www.aisb.org.uk

ISBN 1 902956 18 9

Printed at the University of York, Heslington, York, Y010 5DD, England

Contents

A TKS Framework for Understanding Music Composition Processes and its Application in Interactive System Design

Richard Polfreman ; Martin Loomes
Faculty of Engineering and Information Sciences,
University of Hertfordshire, College Lane, Hatfield, Herts. AL10 9AB.
r.p.polfreman@herts.ac.uk, m.j.loomes@herts.ac.uk

Abstract

We present research directed at deriving a better understanding of the processes of musical composition through the application of Task Analysis techniques. The research was motivated by the need for improved software tools for composers that do not require expertise in signal processing, sound synthesis or computer programming, but take advantage of composers' existing knowledge and experience. An outline is presented of the Generic Task Model developed and its application in the development of two software systems, Modalys-ER and FrameWorks. Potential areas for further application of the GTM and future research directions are given.

1 Introduction

This paper describes on-going research aimed at providing composers with improved software tools for music creation on computer. The initial proposition was that through gaining a better understanding of the processes by which composers produce musical works, we could design software that would better match the needs, knowledge and expertise of composers. The research involved taking user-centred design techniques from the field of Human Computer Interaction (HCI) and applying these to the complex, creative task of music composition. Specifically, a method of Task Analysis (TA) known as Knowledge Analysis of Tasks (Johnson 1992) was used to produce a generic description of music composition tasks that could then be used to assist the software design process. This Generic Task Model (GTM) aided our understanding of the nature of music composition tasks, the environment within which tasks are typically carried out and how these tasks are organised collectively. While the motivation behind the research was to improve user-interface design, we believe that the GTM has potentially wider applications as a framework for cognitive musicological research relating to music composition, in educational areas such as curriculum design and developing assessment criteria for creative musical tasks, as a starting point for developing artificial intelligence systems for specific compositional tasks.

One of the key aims of the work was to encapsulate many forms of music composition and so be independent of stylistic concerns and compositional medium. Thus the study included composition works for acoustic instruments, tape pieces created by sound manipulation on computer and mixed works for live performer and tape/electronics. Aspects of the model may not be particularly new or radical, but where the model does agree with previous ideas, it serves as confirmation by experimental study of composition task performances. It is hoped that this application of generic TA techniques to music composition will serve as a useful example for those investigating other tasks involving human creativity.

While much of this paper concentrates on TA of creative composition, we briefly outline two prototype systems that have been developed on the basis of this work. The first deals with sound synthesis using physical models, allowing composers with little or no knowledge of Digital Signal Processing (DSP) to create complex and interesting musical sounds and gestures using their own virtual acoustic instruments. The second and more recent system attempts to begin dealing with higher level musical structures and reducing *viscosity*[1] (Green, 89) as well as providing support for extra-musical information. Viscosity is problematic in software tools for creative tasks since it reduces the scope for important experimentation and exploratory behaviours.

2 Task Analysis

2.1 KAT/TKS

Task Analysis methods are generally aimed at producing structured models of how people carry out particular tasks. In our research, KAT/TKS was applied to the problem domain of music composition. Otto Laske is one of the few researchers to have undertaken task

[1] Viscosity refers to the level of resistance to change of an artefact. In this case it is a question of how easily potentially large scale changes can be made to a musical composition.

analysis work on music composition at the time and while parallels can be drawn between task models in the form of Johnson's Task Knowledge Structures (Figure 2) and Laske s model of Musical Activity (Figure 1) the approaches are very different in the two cases Our approach has been to look at real-world composition tasks with minimal interference, rather than to use directed tasks in controlled environments This research is more concerned with what Laske refers to as *empirical task analysis* rather than the *conceptual task analysis* with which much of his work deals To put it very crudely, we are aiming to capture more of *what* composers are doing during the composition processes rather than the much more difficult question of *why* We also wanted a more generic result than most studies of music composition, which typically look at Western pitch based music and don't include more sound based electroacoustic music However, having developed such a task model we hope to provide a framework that is complimentary to, and useful for, cognitive musicology research

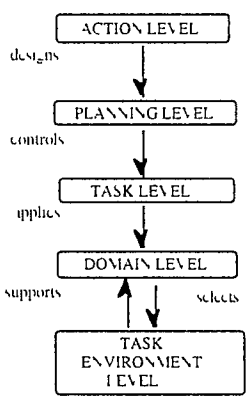

Figure 1 Laske s Musical Activity (taken from Laske 1992)

The GTM should also allow the incorporation of results from other studies that have looked at specific individual composition tasks in more detail in order to improve the model as a whole The current state of the model does provide much detail in many areas, although some remain speculative in nature and require further investigation One area that the model specifically does not include is that of the aesthetic concerns of the composer, i e the basis on which composers' decisions are made regarding what is or is not a satisfactory artistic result at any particular point

Knowledge Analysis of Tasks (KAT) is a suggested methodology for producing a generic task model expressed in terms of Task Knowledge Structures (TKS's) These are organised into goal, procedural and taxonomic structures The goal structure contains *goal* and *subgoal* elements and the control relations between them, where goals and subgoals are states of the environment to be achieved, e g 'edit sound', 'enter note', etc The procedural structure contains the *procedures* for achieving goals/subgoals in terms of *actions* acting on *objects* An object in this model is defined by its set of attributes (data) and actions (methods) that can be applied to it The taxonomic structure contains object definitions and other useful object related information (such as typical instances, which procedures use the object, etc) Figure 2 shows a summary of the structure of TKS's

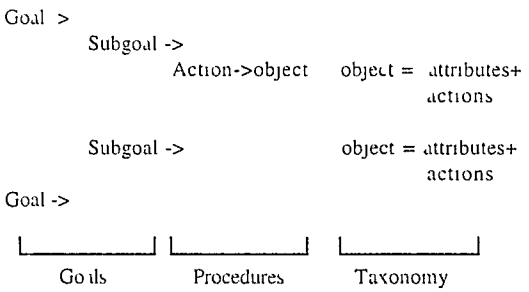

Figure 2 Task Knowledge Structures

2.2 Problems of Method

A TKS is built up on the basis of information gathered from various sources In this research questionnaires, interviews with composers and direct observation of composers at work were used as the main sources of data for the analysis The observations were also followed by retrospective protocols Early versions of the task model were taken back to composers involved in the study to verify that the model represented the composers tasks adequately and to make amendments where necessary Composers of different musical styles and using different technologies were used as subjects so as to produce as generic a task description as possible Music composition presents several problems when targeted as a subject for task analysis three of which are outlined below Similar problems are likely to occur in other creative tasks

2.2.1 Duration

The process of music composition may take place over weeks or more and so require the recording of vast amounts of data In this research detailed observations were carried out on relatively short compositions while others were studied periodically over the duration of the composition (e g once a week over a few months) In both cases the composers were questioned regarding any intervening work on their compositions and any changes made to their plans etc

2.2.2 Data Capture

Given that we were studying composers using a variety of technologies the question of data capture was a challenge While composers using computers could be recorded in various ways such as videos of display output or scripting tools used to auto-save documents periodically into different files, such techniques were found to be problematic For example, the use of keyboard combinations rather than pull-down menus in applications meant that a display video would not show which operation had just been carried out Improved computer based capture methods could be designed for future research, but for generic capture these must operate with a variety of standard music applications

In the event most sessions were recorded by a pre-interview, video of the composer at work with the task observer taking notes and a post-interview with retrospective protocol Composers not using computers/electronics were much less open to being observed while composing and this was abandoned for these composers and replaced by periodic interviews with discussion of artefacts produced, including rough sketches/diagrams as well as the music itself

2.2.3 Diversity

Music composition is a highly idiosyncratic task that varies not only across composers and technologies but also between individual compositions Moreover the final artefact may exist in many forms e g printed score, digital recording, set of computer programs Music composition is often regarded as an 'ill-defined problem' and generally opportunistic in nature Given that we wanted to be as generic as possible composers using different technologies and composing for different scenarios (tape live performers, soloist and live electronics etc) were used in the study We then attempted to generalise the information across all the tasks involved The use of questionnaires sent to a number of composers also helped in finding generic terms for use in the model and for providing rough outlines for how other composers (claim they) approach composition

3 The GTM

3.1 Overview

There is not space here to present the entire GTM, but we summarise the main areas Further details can be found in (Pollfreman, 1999) and a complete description in (Pollfreman, 1997a) We do not claim this to be a single definitive task model for music composition - the model presented here is how we have analysed the task and others may produce very different task descriptions

In analysing such a creative and often opportunistic task, it was necessary to add two important features to the goal structure *partial completion* and *goal hopping* Partial completion implies that it is not always necessary for a goal to be entirely satisfied before moving on to a subsequent goal, i e that partly completing a goal can allow partial completion of further goals This typically takes one of two forms where the goal requires the production of several items and only one or a few are produced, where not all specifics of the result are defined, but only some key features or general category An example would be deciding one instrument for a piece and writing some music for that part before deciding the rest of the instrumentation Goal hopping indicates that once sufficient conditions are satisfied for work to proceed towards a goal, the composer may jump to that goal at any time, providing that the sufficiency conditions remain satisfied Figure 3 shows a network of the top-level goals of the GTM, with three main areas

Design framework involves the setting out of what can be seen as a set of (musical and practical) constraints within which the piece will be composed This goal can often be the most important part of the composition process while in other cases much of this goal is determined with little conscious effort immediately a work is begun

Research may be necessary before a work can be completed It includes many topics that are typically of interest to composers These goals are not analysed any further and it would be difficult to do so in a generic way However awareness of these various research areas could be useful in providing support tools within a computer based music system In particular, it may be useful to incorporate well-known products of these research areas into software tools (e g the standard pitch ranges of acoustic instruments in a score typesetting program, frequency to MIDI pitch charts in a sound synthesis program)

Produce music is the goal of creating the music and setting it down in an external form so that it can be performed or tested This goal involves the creation of the final deliverable product of the composition process Within this goal systems for generating musical material are activated and methods of manipulation are applied This goal includes the production of musical sketches and rough drafts that do not necessarily form part of the final artefact In the past this task has usually been over-emphasised in software tools at the cost of *research* and *design framework* areas

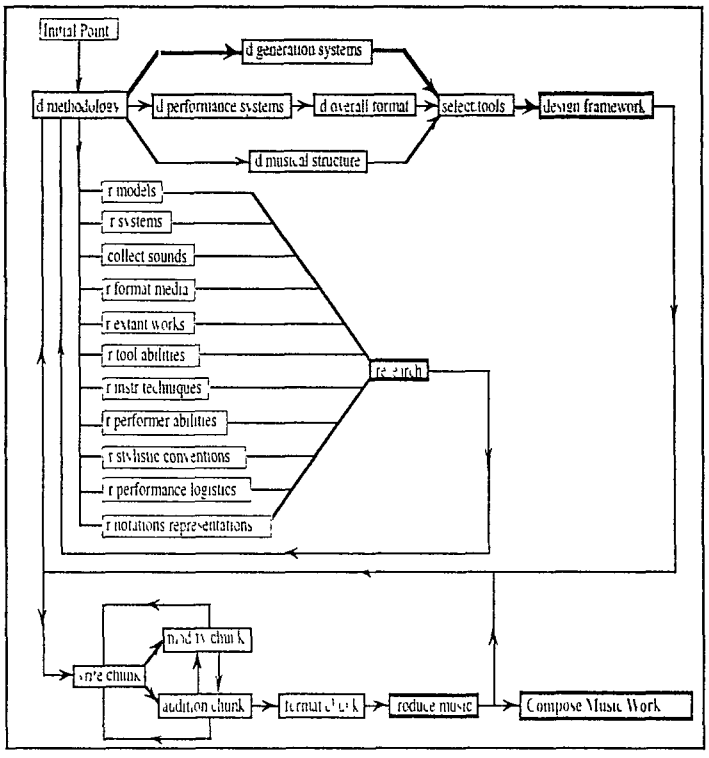

Figure 3 The GTM Top Level Goals Structure

3.2 Design Framework

The starting point is *design methodology*. This involves the formation of a *plan* of action in order to carry forward the process. A *plan* is essentially a list of goals to be completed in a certain order (with possible parallelism). A minimal *plan* may consist of only one target goal. *Plans* are likely to change during the task - the composer often returning to this goal. Initially, from *design methodology* the composer moves to either a research goal, or to one of designing *generation systems*, *performance system* or *musical structure*.

3.2.1 Generation Systems

A *generation system* is a scheme for producing musical material at a level above *instrument techniques* (defined below). Typical *generation systems* may include improvising in a particular key or using a mathematical rule to select notes from a defined mode or pitch set. A *generation system* contains two different elements: selection *processes* that generate values, and *constraints*[2] that decide how the values are mapped onto a set of (musical) events/parameters. A single system may determine just one element of musical material, such as pitch. In this case, other systems are used to decide durations, dynamics, etc. A *process* may be applied to a *constraint*, e.g. to shift a set of pitches being used over time. *Processes* may also be applied to *processes*, e.g. a simple repeat *process* to a graphical pattern. The interaction between several simple *processes* and *constraints* can result in complex musical material.

The processes and constraints being used by composers may be difficult to describe computationally and vary widely in nature. Given the diversity and potential complexity of generation systems it is difficult to analyse the task of creating them in a generic way. For example, if a composer is using a system with some graphical mapping, the whole range of image transformations become possible manipulations, which in other generation systems are not relevant. At this level it is better to organise systems into different categories and describe their design accordingly. In particular intuitive and improvisatory systems need to be investigated from a cognitive musicological standpoint.

3.2.2 Performance Systems

A *performance system* is defined as 'a system that given a score produces sound' - 'score' having a wide interpre-

[2] Note that the term *constraint* here is used in the general meaning of the word rather than necessarily as formally defined in constraint based programming.

tation. A *performance system* includes three major components:

Instrument: 'a system of sound production that has an associated set of potential sounds'.

Instrument Technique: 'a method for the production of a sub-set of an instrument's potential sounds'.

Performer: 'an interpreter that reads a score and activates instrument techniques'.

Figure 4 illustrates the relationships in performance systems and gives some examples. A composer may use various performance systems purely for aid during the compositional task rather than for final performance. For example, a score typesetting program and MIDI sound module may be used in the composition of a work for acoustic instruments.

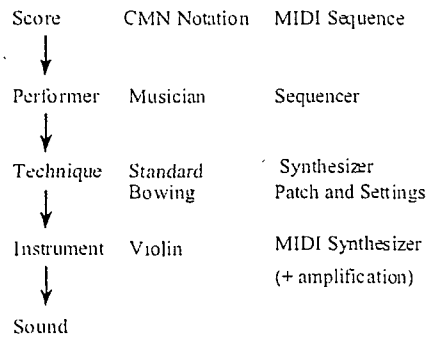

Figure 4: Performance Systems and Examples (types of corresponding scores also shown)

3.2.3 Overall Format

'Overall format' refers to the nature of the deliverable materials of the composition process - printed scores, tapes, signal processor routings, etc. This *format* is usually dependent upon the types of the instruments used, if not the actual instruments themselves. A *format* itself covers both the physical media upon which information is stored and also the information representation (and rendering thereof) used. Thus, a typical printed score *format* might consist of A3 paper, using Common Music Notation representation and a standard Visually Rendered Notation for CMN (Huron, 1992).

3.2.4 Musical Structure

Design musical structure refers to the creation of a skeletal (temporal) framework within which the musical material produced by *generation systems* is assembled. This relates to the overall shape of a piece as well as internal relationships between material occurring at different locations. The interaction between this goal and that of designing *generation systems* is very important - in fact the distinction between the two areas can become blurred. For example, processes in *generation systems* may be applied recursively to material, first defining events, then organising these progressively to form structural hierarchies. We must be clear though, that a *musical structure* can exist independently of whether any musical material has been specified or not. Here, we define a *musical structure* to be constituted from a hierarchy of *structural components* and *structural relations*. A *component* is simply a sub-unit of a *musical structure* (which may itself contain various *components* and *relations*), while a *relation* expresses some kind of aesthetic and/or musical link between components. A composer may be working with different musical structures for the same piece simultaneously, each structure looking at the work from different musical or aesthetic viewpoints.

3.2.5 Select Tools

The *tools* are any items to be used in the composition task - musical instruments, writing implements, reference materials, software, etc. The idea of partial completion is important here, since tools are necessary for use in previous goals - e.g. *design generation systems*. In this case, some decision is made about the type of *generation system* to be used, *tools* are then selected for the design process and then the composer returns to *design generation systems* to complete the actual design. The *select tools* goal is placed after these other goals since normally some earlier decisions must be taken before the *tools* for the task can be chosen.

3.3 Produce Music

The primary goal of *design framework* is achieved by completion of the goals described above. From this point the composer can move on to the subgoals of *produce music, research* or return to *design methodology* or any (partially) completed subgoal covered so far.

3.3.1 Write Chunk

A *chunk* in this model is defined as 'a segment of musical material of arbitrary size, usually (but not always) bounded in some musically meaningful way'. A *chunk* may be purely conceptual in its delineation, rather than explicitly set out in the music. Also, a *chunk* may be assigned temporarily by a composer to a region of music rather than being a significant fixed feature within the structural framework of a piece. There are parallels between the *chunks* that make up a piece and the musical structure of a work, but not necessarily a direct correspondence. A chunk that has been written may real-

ise a structural component in a work, but may just be a part of a component or can be spread across several components. A chunk is produced by activating one or more *generation system* in order to create the material.

3.3.2 Audition Chunk and Modify Chunk

These are two closely connected goals. Auditioning is a checking mechanism to see if a *chunk* is correct, i.e. meets current musical requirements, and if not, to identify the precise faults. This can be done via visual (i.e. score reading), or aural (i.e. sound playback or aural imagination) means, or a combination of the two. Since 'current musical requirements' may change at any time, *chunks* are often auditioned repeatedly at different times during the composition process. Auditioning can also lead to changes in *performance systems* where the fault is identified as being in instrumentation rather than in score events. There is generally a complex interaction between writing, auditioning and modifying *chunks*, with jumps from one to another, jumps between *chunks* and sub-*chunks* and jumps back to *design framework* or *research* goals. Modifying a chunk generally involves applying a transformation (including delete) or identifying a sub-chunk and modifying that.

3.3.3 Format Chunk

In order to provide a finished product the goal *format chunk* is undertaken, normally after a final audition of the *chunk* to be formatted (which may be the entire piece). In fact, up to this point the music may only have existed in the composer's imagination. Formatting can involve saving files onto a disk, recording to tape, or writing out by hand a fair copy of a drafted score. A formatted version is usually auditioned itself in order to check for last minute errors. Once all the *chunks* that form the piece have been formatted, then the produce music task is completed and the composition is generally finished, although may be re-worked subsequently.

4 Prototype Systems

4.1 Modalys-ER

Although the GTM covers a wide range of composition tasks, our prototype systems so far have been aimed at dealing with specific problem areas. Building on the basis of these prototypes, we aim to build more complete systems in future. Modalys-ER - originally Modalyser (Polfreman, 1997b) - was aimed at dealing with sound synthesis, an area traditionally dominated by signal processing models that require specific knowledge and skills that many composers do not possess. The composer has to specify sound and sound control by the manipulation of elements such as signal generators, filters, envelopes, in order to construct a time varying spectrum that matches the desired timbre. By using physical modelling algorithms (developed at IRCAM, Paris) controlled though a graphical user-interface, we developed a system that allows users without DSP experience or programming skills to define their own 'instruments' (using objects such as plates, tubes, strings and interactions such as pluck, strike, bow) and control these instruments via simple time varying graphs. This approach opens up sophisticated sound synthesis to a wide range of potential users. The structure of the system is along the lines of the task model although a Modalys-ER instrument contains both the GTM *instrument* and the *techniques* for playing it. Also, the Modalys-ER score may be seen in the role of GTM *performer* since it contains a low level specification of technique activation and not high level structures. Figure 5 shows a Modalys-ER instrument, where the top half contains the instrument 'hardware' and the bottom the techniques for playing it.

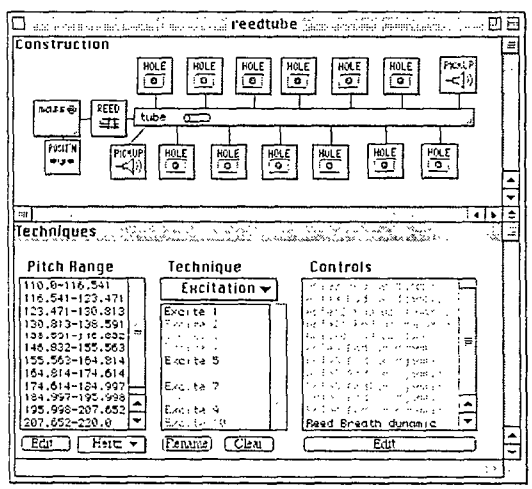

Figure 5: A Virtual Reed Instrument in Modalys-ER.

Figure 6 shows a Modalys-ER 'score'. Given the processing speed currently (the physical model algorithms do not yet approach real-time) the need so far for high level structures within the system is reduced.

One problem that remains with this system is that of instrumental control, i.e. the interface between performer and technique elements. Since the user can define arbitrary instruments with any number of controllable parameters it is difficult to produce a simple-to-use system for expressing these controls in a generic way. That is, such that a given score can effectively control any instrument in a sensible way. In future work we hope to revisit these areas in order to provide a more satisfactory solution than in the current system.

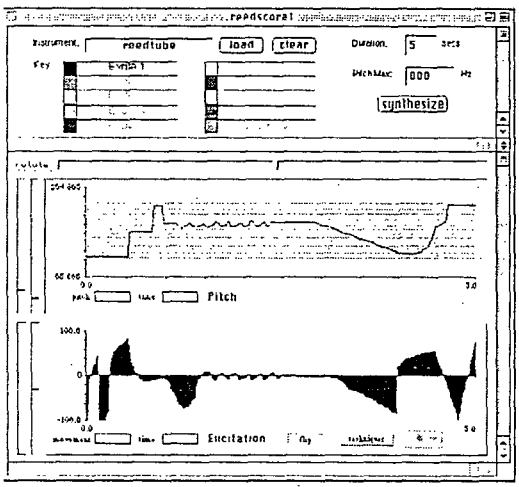

Figure 6: A Modalys-ER 'Score'

Expert systems would be useful for aiding aspects of specification and control of instruments, since although the general mechanism of physical models is well understood by many composers, the details of parameter setting are not necessarily so. Also, performance parameters are not always obvious in their behaviour, with interactions such as reeds and bowing particularly difficult to control. Modalys-ER runs on Mac OS and is available via the IRCAM Software Forum.

4.2 FrameWorks

The second prototype, FrameWorks (Polfreman, 2001), looks at more general problems of structuring and defining musical material, whether this is control data for physical or signal models, sound sample data or music notation, although FrameWorks itself only handles MIDI events. In typical extant systems there is little satisfactory support for structural concerns of composers (beyond the level of bar/measure) or defining musical relationships between materials at different points in a musical work[3]. This promotes a bottom-up approach to composing, while the GTM indicates that top-down and bottom-up approaches may be equally important. In FrameWorks composers work with a *framework* of *components* (segments of musical material of effectively arbitrary duration) that are interconnected by *relations* of various types that establish and maintain user-definable transformations of musical material from one *component* to another. These structures are dynamically maintained and so any changes made to material, components or relations are instantaneously reflected throughout the structure, thus producing a system with reduced viscosity as defined earlier. Components can be continuously stretched or compressed by the user which slows are accelerates the material inside and again these changes are propagated throughout the system. The components and relations are related to those defined in the GTM, although currently having a more limited scope. The system allows the composer to experiment with relationships and material independently and s/he can in fact define structures and relationships before any musical material has been specified. Although currently a single level framework this already offers composers much that is difficult to achieve within typical music applications. In future development we hope to extend the framework system to handle hierarchical organisation and multiple overlayed structures to improve the scaleability of the system. One of the interesting features of work produced using the system, i.e. composed using the framework structure, is that the document itself contains a useful analysis of the work which could then be studied from a musicological point of view, since much of the composer's understanding of the piece is explicitly set out in these structures.

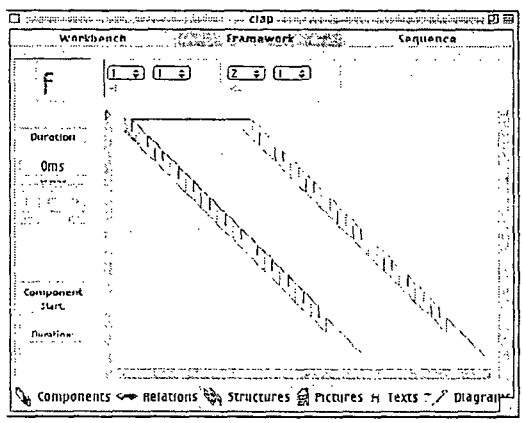

Figure 7: A Framework of Components and Relations

Figure 7 shows a framework implementing an abbreviated (i.e. without repeats) version of Steve Reich's Clapping Music as a simple example. This piece consists of two parts (to be clapped) based around a simple pattern (shown in the component editor in Figure 8). One part simply repeats this pattern throughout, while the second recursively applies a simple rule that takes the first beat of the pattern and places it at the end each time, until arriving back at the original pattern again. This simple movement of one part against the other produces complex shifting rhythmic patterns that develop through the piece with a final return to unison at the end. In Figure 8 a simple piano-roll style notation is used to specify the original pattern (ta-ta-ta ta-ta ta ta-ta) for the source component.

[3] Apart from in specialist algorithmic composition systems.

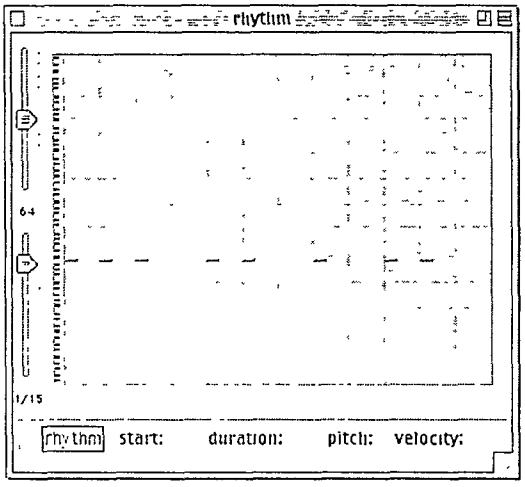

Figure 8: A Component Editor

Figure 9 shows a *time relation* implementing the shifting rule. It contains two *time maps*, the first of which takes the last eleven twelfths and plays them first, the second of which takes the first twelfth and plays that at the end.

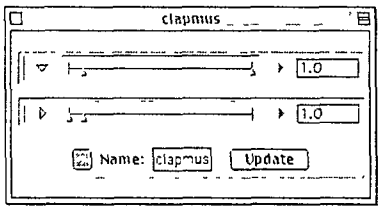

Figure 9: A Time Relation Editor

In the framework, the top left component (darker than the rest) is the source pattern which is connected to several following components by identity relations, thus creating the static part. It is also connected across to the adjacent part, here using an 'octave up' relation to make the two parts more separate. This track then applies the 'clapmus' relation successively in order to create the moving part. The advantage of the system is that once such a framework has been created, it can form a template that is very simple to adapt or change by altering the source material or redefining relations.

This prototype also includes a 'workbench' area where composers can gather together the products of any research they have carried out, including both musical and non-musical material, but not yet including any research enabling tasks (other than the ability to take notes, make diagrams, etc). Figure 10 shows a workbench containing several elements, including text, a picture, a component and diagrams. Each element has a drop-down preview and can be opened to display its contents fully and for editing in most cases.

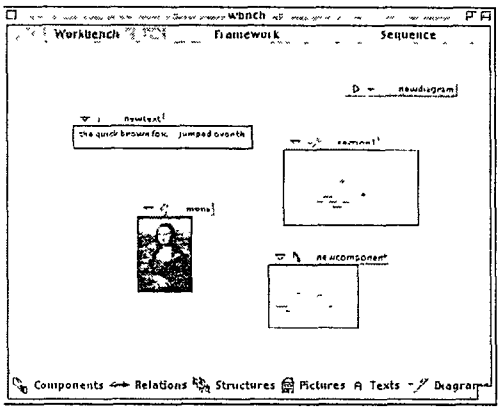

Figure 10: A 'Workbench'

FrameWorks is a Java application (1.1.5 or later with Swing/JFC) and requires Grame's free MidiShare system. It currently operates on MacOS and Windows ('95 or later) operating systems, and in future Linux. A preview release is freely available from the University of Hertfordshire from March 2001.

5 Conclusions & Further Research

The GTM was used here in assisting systems development, particularly in terms of defining the structure and conceptual levels of the user-interface. As prototypes develop, the model will be used in checking for absent functionality within the area of the GTM that the systems attempt to cover. Future work may also review other computer music systems, using the GTM to set out evaluation criteria.

A key limitation of the current model is the lack of *secondary* goal relations. The fact that composers will make various jumps within the structure means that many patterns of task performance are not directly captured in the model itself. This is to say, *primary* relationships between goals and subgoals are given, but *secondary* goal relationships are not, e.g. those between *design musical structure*, *write chunk* and *design generation systems*. These *secondary* relationships are likely to be complex and widely variable, making them difficult to encompass within a generalised TKS. Further study of composers in the light of the GTM may assist in developing our understanding of these relationships and so in representing these in the model. The absence of such relations can be detrimental to the model's application in user-interface design, since parts of an interface corresponding to areas with *secondary* relations may be not be traversable in appropriate ways.

Further TA studies investigating specific areas not sufficiently well described by the current GTM will be carried out This will include studies involving our prototype systems as part of the task-artefact-cycle (Carroll and Rosson, 1992) In particular, the separation of structure and material in FrameWorks may be useful in analysing these elements more independently, while also indicating the nature of the relationships between them FrameWorks itself is only a proof of concept system in its current form and we hope to develop these ideas much further With respect to higher level structures, we are exploring the use of 3D user-interface technologies for effective representation of complex hierarchical frameworks Provision of support systems for various generation systems at the lowest level of the structure will also be investigated, together with potential automated derivation of relations through comparisons between components Performing analysis of extant music works using FrameWorks is another possibility at a point where the system has become more sophisticated

It is hoped that the GTM provides a useful contribution to the understanding of the complex task of music composition and may be explored further by researchers in areas other than user-interface design By providing a structured analysis of music composition into specific goals and subgoals, the model could provide a basic framework for expert systems design where certain goals may be analysed further and applicable rules developed for their automatic completion by the system

Acknowledgements

We would like to thank past and present members of IRCAM for their assistance and co-operation in the development of Modalys-ER, including Hugues Vinet, Francisco Iovino, René Caussé and other members of the Musical Acoustics team Also thanks to Stephane Letz at Grame for help with MidiShare/Java in the development of FrameWorks Finally our gratitude goes to all the composers who (anonymously) took part in this research

References

J Carroll and M B Rosson Getting around the Task-Artefact Cycle How to make claims and design by scenario *ACM Transactions on Information Systems*, 10 181-212 ACM, 1992

T R G Green Cognitive dimensions of notations *People and Computers V, Proceedings of the HCI '89 Conference* 443-460 Eds A Sutcliffe and L. Macauley. CUP, 1989

D Huron Design principles in computer-based music representation *Computer representations and models in music*, Ch 1 Eds Marsden & Pople Academic Press, 1992

P Johnson *Human computer interaction psychology, task analysis and software engineering*. McGraw-Hill, 1992

O Laske Artificial intelligence and music A cornerstone of cognitive musicology *Understanding music with AI*, Ch 1 Eds Balaban Ebcioglu & Laske MIT Press, 1992

R Polfreman *User-interface design for software based sound synthesis systems* (PhD Thesis) University of Hertfordshire, 1997a

R Polfreman Modalyser A Graphical System for Creating Modalys Performances *KlangArt 97*, Osnabruck University, 1997b

R Polfreman A task analysis of music composition and its application to the development of Modalyser *Organised Sound*, 4(1) 31-43, 1999

R Polfreman *FrameWorks user documentation*. University of Hertfordshire, 2001