A Human Factors Approach to Computer Music Systems User-Interface Design

Richard Polleman & John Sapsford-Francis
John Lill Centre for Music Studies, Computer Science Division, School of Information Sciences, University of Hertfordshire, College Lane, Hatfield, Herts. AL10 9AB U.K.
R.P.Polleman@herts.ac.uk, J.Sapsford-Francis@herts.ac.uk
Vox: +44 01707 284411, 284309
Fax: +44 01707 285098, 284303

Abstract: Software sound synthesis has failed to gain wide usage amongst professional composers. We highlight two major causes: the lack of available real-time facilities and user interfaces ill-matched to non-specialist composers. We describe Human Factors work at the University of Hertfordshire (UK) that is developing new user interfaces to Computer Music Systems. A major factor is the use of empirical Task Analysis in evaluation of extant systems and requirements capture for the design of new systems.

1. Introduction

There are a number of popular uses of computers in music e.g. MIDI sequencing, digital audio and score typesetting. Software sound synthesis has been in existence for nearly 40 years but, despite its longer history, it remains little used outside the avant-garde or academic research. We posit two major reasons for this situation:

- The processing power required for real-time software synthesis has not been supplied by readily available computer systems (i.e. PCs). This has made software synthesis unattractive compared with electronic synthesis that provides real time, polyphonic, multi-timbral sound. However, processing speeds are now approaching real-time on RISC based PCs - e.g. a PowerMacintosh running 'native' (sound) and sound direct to a DAC after a short delay (there are some glitches however).

- An alternative approach has been to provide additional external processors to allow real-time sound synthesis (e.g. Kyma - Scalaletti, C.). We would expect that soon PC technology will be capable of complex real time sound synthesis with minimal additional technology. Although essentially a hard-ware problem, there is scope for designing highly efficient synthesis algorithms to reduce processing times.

- The provision of user-interfaces that are a poor match to the needs, knowledge and expertise of composers not expert in computer programming and sound synthesis techniques. Software synthesis systems have generally presented the user with a wide range of tasks and representations unfamiliar to composers outside the field. "Traditional" music composition involves the creation of a notated score by the composer that is then given to musicians for performance. In a computer system the composer must specify performance details that are generally absent from CMX scores i.e. those provided by a musician's interpretation. In addition the composer must design the 'instrument' itself and decide how it is to be controlled in performance. This design process often requires an extensive knowledge of DSP techniques and sound synthesis methods. The composition task is often complicated by user interfaces based on computer programming languages, together with little use of musical concepts and structures familiar to the composer. Oppenheim has stated: "Systems for creating music were too often designed by engineers who did not have a sufficient understanding of the musical issues such as: what is music? what is a musical idea? how does a composer express a musical idea? and what is the compositional process?" In our human factors approach to user interface design we hope to gain some insight into these questions, concentrating on aspects of the compositional process.

We focus on the second of these causes, that of user interface design, via the application of empirical Task Analysis methods. These methods have been successfully applied to user-interface design in other software domains.
2. Task Analysis

2.1 Knowledge Analysis of Tasks

Human factors work has identified analysis of users' tasks as an important aspect of user interface design and evaluation (e.g., Casey, Stammers & Astery; Shackel). There are many approaches to task analysis suited to different applications (see Johnson, P. for an overview). Our work requires a method that is system independent, appropriate to the early stages of software design, captures the cognitive nature of tasks and is applicable to software design and evaluation. We selected the Knowledge Analysis of Tasks (KAT) method using Task Knowledge Structures (TKS), a full description can be found in (Johnson, P).

Briefly, TKSs are organised into three main structures - a goal, a procedural and a taxonomic structure. The goal structure contains goal and subgoal elements and the control relations between them (plan). Goals and subgoals are states of the environment to be achieved e.g., 'create timbre', 'play note' etc. The procedural structure contains the procedures for achieving goals/subgoals in terms of actions acting on objects. An object is defined by its set of attributes (data) and actions (methods) that can be applied to it. The taxonomic structure contains object definitions, including the class hierarchy.

KAT provides a set of guidelines and suggested methods for both acquiring the information required to derive TKS elements and for combining information from a set of task performers into a Generalised Task Model (GTM). The methods suggested include questionnaires, interviews, protocols, direct observation, etc.

The task analysis of music composition presents various practical problems. We cannot discuss them here, but major difficulties are: the long duration of the composition process, data capture where composers are using a wide variety of equipment during composition, the idiosyncratic nature of music composition, minimizing the impact of data capture methods on the composers.

2.2 Users

There are many potential users of a CMS who have a variety of purposes in mind. We have restricted tasks in music composition and have defined 3 categories of user based on the technology they currently use for sound generation during composition. This does not necessarily imply musical style, or the technology used for final performance. The categories are acoustic, electronic, and computer composer. Thus the acoustic composer uses acoustic (or no) instruments during composition, the electronic composer uses electronic synthesis/sounds (and perhaps instructions) and the computer composer uses software based synthesis/DSP. Composers may correctly regard themselves as members of more than one category. We can project the knowledge that the different categories of composers may be expected to possess, e.g. the acoustic composer generally has no knowledge of DSP techniques, while the computer composer generally has expert knowledge of this field - see (Pottfam et al.) for further discussion. In our work we analyse the composition tasks of members of each category in order to construct a GTM and to consider which tasks will be carried out by different types of composer in that generic model. This is useful in designing multi-level systems to provide for the knowledge sets of different potential users.

3. Results

Space limits us to a few key points:

3.1 Questionnaires

Questionnaires have been returned by composers of various styles who use different technologies.

• Many composers, of all types, use informal diagrams or sketches in both musical and non-musical notations during the composition process. These tend to be developed early in the compositional process (although subject to later changes) and they tend to describe the form or high level structure of the piece.
• Some composers indicated an initial development of form that would then be fleshed out with materials, others the development of materials leading to an emergent form. Others again indicated an interaction of the two processes. The separate construction of form and material and the manipulation of both would seem to be an integral part of the composition process.

• A substantial number of electronic composers do not create their own sounds - they use presets provided or make minor adjustments or 'tweaks'. Some of these composers are closely related to acoustic composers (often composing for acoustic ensembles) in that they use fixed instruments, that is presets, that replace the use of acoustic instruments directly.

3.2 Task Models

• There are various composition tasks that are carried out either in the 'background' or when composers are not specifically 'composing'. These tend to be idea generating processes.

• Certain tasks seem well structured, although many have optional components. A common structure is the iterative performance of a task or tasks with some 'judgement' by the composer to allow exit from the loop. This type of structure conforms to a Pirerian view of the compositional process as an evolutionary loop of problem identification, trial solution and error elimination (Magne).

• As predicted tasks were often interrupted by other tasks, that on completion were followed by a return to the original task or to a second interrupting task. Nesting of interrupts has not gone beyond 2 in observations so far. Interrupting tasks usually bore some relationship to the interrupted tasks.

• A common task used throughout the composition process is an auditing process i.e. that of listening back to some section of music or sound and making some judgement as to its quality. Skilled users of CMM are able to do this via reading the notation and translating this internally into some aural form i.e. using the 'minds ear'. Auditioning tasks require aural feedback as quickly as possible and so extended processing times are problematic.

• Composers' tasks can be considered at many different levels - a high level conceptual level, a notational level, (a software level) and a hardware level. The mapping of composition tasks from the conceptual level to the hardware level can be assisted by effective notation and software designs.

4. Summary and Further Work

Work so far has suggested several guidelines for CMS user-interface design, including:

• CMS designers should take into account the wide variation in composers' knowledge, the types of composer they are designing for and the characteristics of those composers. It may be desirable to design multi-level systems where complexity can be hidden from users who do not require it, and to design knowledge based systems for taking care of details a user does not wish to specify directly or precisely.

• The classification of tasks within the GTM according to types of user can define boundaries between levels in a multi-level interface approach. Classification of tasks into those involving knowledge, rule or skill based problem solving (Rasmussen, J. and Jensen, A.) can aid the design of systems that make use of composers' existing skills, and indicate where possibilities for automated processes lie.

• Composers are often skilled at manipulating both formal and informal graphic representations of musical information. These representations generally do not contain the complete specification of music performance but describe only certain aspects. Informal notations tend to be used for description of high level structure or form. Few current systems support informal notations (for an exception - which doesn't support synthesis however - see Kistler & Howard) or flexible graphic manipulation of multi-level forms. Further study of informal notations used by composers may be useful here.
• The development and manipulation of musical form is often carried out prior to, or concurrent with, development of musical materials. Most systems require a bottom up approach with the creation of material that is then shaped into different forms. This means that the composer will often develop ideas of form outside of the CMS and then import these ideas after construction of the materials. A complete system should aid in supporting work on form independently of materials within it.

• Few composers we classify as electronic or acoustic have any experience of computer programming languages or techniques. Systems built around languages such as Lisp, Smalltalk or dedicated music languages such as Csound are therefore problematic for these potential users. Even graphical abstractions of these languages (e.g. IRCAM's Patchwork) carry inherent characteristics of the language and so can be difficult to learn. Computer language based systems do have advantages of extensibility and flexibility, however they also have problems of learnability and over functionality.

• Systems such as Csound have the problem of high viscosity [Green, TRG]. Changes to one aspect of the price require that many changes are made elsewhere. For example, in Csound changing an instrument definition generally requires all events in the score to be altered accordingly. In creative composition tasks the composer often wishes to make changes (perhaps on a large scale) very quickly. This means that systems should be designed to have a low viscosity where possible, or that mechanisms for swift propagation of changes through the system should be provided.

• No single system is likely to fulfill all user demands. Therefore it is desirable to provide a system that is not only modular and customisable but that can also be used in conjunction with other software. This requires that musical information be shared and/or communicated between applications. This has happened with MIDI software on the Macintosh for example. However MIDI is inadequate for many purposes. ZIPI may provide sufficient synthesis and sound information for future systems.

Future work will include the continuation of task analysis on a wider range of composers, development of metrics for evaluation of CMS's from generalised task models, reviews of music representations, notations and synthesis models, selection of particular interface problems and prototyping solutions.

References
Kail, M. 1993. Patchwork™ Introduction (user manual), IRCAM.