Flow of Creative Interaction with Digital Music Notations

Chris Nash and Alan F. Blackwell

The Oxford Handbook of Interactive Audio
Edited by Karen Collins, Bill Kapralos, and Holly Tessler

Print Publication Date: May 2014 Subject: Music, Applied Music
Online Publication Date: Jul 2014 DOI: 10.1093/oxfordhb/9780199797226.013.023

Abstract and Keywords

This chapter highlights disjunctions between usability approaches and the needs of creative music practices, drawing on research into creativity and human–computer interaction (HCI) to integrate concepts of flow, virtuosity, and liveness into the design of digital notations. While computers support the production and transcription of creative ideas, current user interfaces are less suited to exploratory creativity, sketching, and the early stages of the creative process. The chapter discusses properties of interfaces and notations that influence such support. It then presents both a set of usability heuristics for virtuosity, to aid the design of user interfaces supporting skill and learning, and a technique for modeling aspects of flow and liveness within the creative user experience, emphasizing user focus and system feedback. Findings and theories are discussed in the context of examples from desktop and studio music software, such as sequencers and trackers, but they can also be generalized to other scenarios in digital creativity.

Keywords: notation, flow, liveness, virtuosity, HCI, computer music, sequencers, trackers, user experience, digital creativity, feedback

---

PRACTICE-BASED research into digital audio technology is the source of many new and exciting interactions, instruments, and sonorities. However, the nature of the technologies used raises significant challenges for traditional conceptions of musical practice. The disjunctions between composition, performance, and improvisation, between the use of common score notation and other graphical representations, and between discrete and continuous expressive scales can be compared to long-standing debates in human–computer interaction (HCI) regarding direct manipulation (e.g., mouse-based point-and-click, drag-and-drop, etc.) and abstract programming (e.g., keyboard-based notation editing), graphical user interfaces (GUIs) and command lines, and visual (e.g., Max/MSP) versus textual (e.g., SuperCollider) programming languages (see also Chapter 24 in this volume).

Our HCI research group has a long-standing program of work understanding the characteristics of notational systems in the broadest sense (Blackwell and Green 2003). We consider any visual, textual, or symbolic user interface to be a notation, which can be treated as directly analogous to music notation in the sense that it guides the future operation of the computer, just as music notation guides the “operation” of a performance. Performances can be more or less literal, more or less improvised, more or less edited and so on. All of these variations are found in both digital music systems and other digital systems, and raise theoretical challenges for computing as they do for music.

Nevertheless, the tools provided by traditional HCI theories and usability techniques have found only limited utility in catering for musicians (Paradiso and O'Modhrain 2003), especially in guiding the design of notation-based interactions (Church, Nash, and Blackwell 2010). In music, these debates are often framed in terms of the personal style of artists and practitioners, or within broad traditions and communities of practice (for example, individual preferences for SuperCollider or Max/MSP). However, this approach to analysis can obscure useful commonalities. In this chapter, we therefore combine research perspectives from HCI with those of digital music production. Our intent is to document the theoretical considerations and issues that emerge when designing and evaluating interfaces for musical expression and creativity. Drawing from other fields, such as psychology and programming practice, we discuss models of the creative process, notation use (Green 1989), skill development (virtuosity) (Nash and Blackwell 2012), flow (Csikszentmihalyi 1996), and the "liveness" (Tanimoto 1990) of musical feedback (Church, Nash, and Blackwell 2010), to highlight limitations in the use of HCI models and theories for music. We propose design heuristics for the support of virtuosity in music systems, to complement those more generally used to provide *usability* (Nielsen 1993), and present a modeling framework for considering these issues within the creative user experience, in the context of real-world music applications.

The concepts, themes, and theories behind the models and recommendations presented in this chapter are the product of a large-scale, two-year study of over one thousand sequencer and tracker users, using a variety of HCI techniques, including interaction logging, video studies, and user surveys. Our findings, which are presented elsewhere (Nash and Blackwell 2011, 2012), complement the theoretical work presented here. Wider applications of the model and details of flow and liveness in programming activities, which may be relevant to live coding practices, have also been published (Church, Nash, and Blackwell 2010).

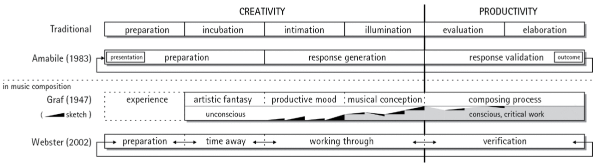
23.1 The Creative Process

Most theories of creativity attribute the creation of novel ideas to the unconscious mind, where an individual's experiences and stimuli are aggregated into new forms, ultimately surfacing into conscious awareness (Sternberg 1999). Wallas's *stage theory* (1926), based on the earlier reflections of Helmholtz and Poincaré, forms the basis of many recent descriptions of the creative process, describing distinct stages in this process (Csikszentmihalyi 1996; Sternberg 1999) (Table 23.1).

Table 23.1 Overview of the creative process

preparation	conscious, active work to thoroughly familiarize oneself with the problem or task
incubation	unconscious processing of the problem, often over time, away from the task
intimation	where the individual becomes aware that a solution is close at hand
illumination	the moment when a solution emerges into conscious thought
evaluation	a period of critical, conscious work, to verify the suitability of the solution
elaboration	a final period where refinements are made to an otherwise verified solution

Stage theory's linearity and apparent focus on goal-oriented, creative *problem solving*, rather than the more exploratory examples of creative self-expression found in art and music (Sternberg 1999), have encouraged recent theorists to consider more iterative, recursive, parallelized, and less directed forms of the model, as



Click to view larger

Figure 23.1 Stage-based theories of the creative process (Csikszentmihalyi 1996, Wallas 1926), and two descriptions of the music composition process (Graf 1947; Webster 2002), in the context of broader "creativity" and "productivity" phases of "innovation," as characterized by Amabile (1983). See references for detailed descriptions.

shown in Figure 23.1. In this way, artistic expression, such as music composition, is often characterized as an ill-defined creative problem, where the creativity rests as much in finding problems, as solving them (Amabile 1983).

Amabile's *componential theory of creativity* (1983) expanded stage-based accounts to reflect the ongoing iterative process within creativity, as well as the crucial roles of expertise and intrinsic motivation, which enable an individual to progress and persevere within a domain. In music, Webster's model (2002) echoes this cyclic process, but also accounts for the tendency to jump between stages, observable in many composers' less formally structured, sometimes erratic, working practices.

Graf's review of composition practices (1947), a rare example of the limited canon of composition research, describes the stages more as moods, and emphasizes the importance of the *musical sketch*, as a tool composers use to probe and elicit musical ideas from their unconscious. Sketches, by virtue of their low-fidelity and exclusively personal use, enable the artist to very quickly experiment with novel ideas, without more formal verification or external oversight, economically trialing a more involved creative process. They allow an individual to explore more ideas, which can be accepted or rejected without significant penalty; facilitating creativity through greater ideation (Sternberg 1999), as illustrated in Figure 23.1.

23.2 Performance-based Music Production

While the score was once the only method of distributing music, the introduction of recording technologies allowed live performances to be captured, thus partly obviating the need for formal notation and literacy. The audio-processing model of music production became even more widespread when computer technology brought the digital studio to the desktop, in the form of the sequencer and *digital audio workstation* (DAW). These programs used visual metaphors (Blackwell 2006), drawing analogies to pianos, mixers, tape recorders, and even dangling wires, to support and preserve the working methods of studio musician, allowing the recording of live performances from acoustic or digital (MIDI) musical instruments (Duignan et al. 2004).

Though these packages offer a multitude of editing and postprocessing tools, the sequencer user interface is principally designed around the manipulation of recorded data, reflecting a division in the creative process—the creativity supported by the live performance of musical instruments, and the productivity supported by subsequent *windows, icons, menus*, and *pointer* (WIMP)-based editing, which is considerably less live (Nash and Blackwell 2011). Consequently, studies have observed a tendency for music software to support only the final, refinement stages of the creative process (Blackwell and Green 2000), and not the generation of new ideas (Smith, Mould, and Daley 2009).

23.3 Feedback and Liveness

In Marc Leman's compelling argument for more engaging *embodied cognition and interaction* in music technology (2008), he cites inherent limitations in any attempt to interact with music *indirectly* through an abstract layer of notation such as a score, piano roll, waveform, or graphic user interface. His perspective implicitly rationalizes the focus on live, real-time performance (and its discrete capture) and the peripheral role of computer editing, in the use of software such as sequencers, DAWs, Max/MSP, and the like to create music. The process of sketching, however, illustrates how notations can be used to support creativity and encourages us to think with greater optimism about the opportunities afforded by notation-mediated music interaction. A central element of Leman's thesis centers on supporting fast *action-reaction cycles* between the individual and music, replacing abstract visual modes of feedback (notation) with more direct real-time modes, such as haptics and sound itself.

In other work (Church, Nash, and Blackwell 2010; Nash and Blackwell 2011, 2012), we explored the role of feedback and interaction rates, looking at the specific interaction issues resulting from the use of *direct manipulation* and *WIMP* interfaces (e.g., sequencers, DAWs), which focus on continuous visual representations of musical parameters in real-time, in comparison to *programming-like* notation-based interfaces, like soundtracking (MacDonald 2007), which revolve around the very fast keyboard editing of scripts for future events, similar to live coding (Blackwell and Collins 2005). Borrowing from programming, we adapted Tanimoto's concept of "liveness" (1990), which describes the level of availability of feedback about the end product (the program or piece of music) from within the development environment (a code editor, sequencer, or tracker).

We found that although the sequencer architecture supported the highest level of liveness, through live performance capture (Level 4, *stream-driven*: continuous, real-time manipulation of the domain, e.g., sound), subsequent visual and mouse-based editing activities were significantly less live (Level 2, *executable*: interaction with a visual specification of the domain). By comparison, the rapid interaction rate and broad availability and prominence of musical feedback during editing in the tracker provided greater overall liveness in the user experience (Level 3, *edit-triggered*: feedback from the domain is available after any user input).

The speed with which the tracker user interacts is aided by the ergonomics and motor-learning supported by the computer keyboard, leading some to describe "the art of tracking" as "some sort of musical touch-typing" (MacDonald 2007). A tight edit–audition feedback cycle is possible because the keyboard is used not only for note entry, but also music editing, program navigation, and playback control. At the same time, the focus provided by the editing cursor provides an implicit playback marker, from which edits can be quickly auditioned, without having to consciously move a song pointer. The motor and keyboard skills learned by the user mean that, with practice, many interactions become *ready-to-hand*, and can be executed without reflecting on the physical action. In this sense, at least part of the interaction becomes *embodied*.

23.4 Virtuosity in Computer Music Interaction

Much of the speed advantage demonstrated in the tracker user experience is enabled by the development of expertise; motor skills and program knowledge learned and practiced over an extended period of time. Supporting expert use in a program can introduce learning curves that conflict with the goals of natural and intuitive usage by novices that dominate mainstream approaches to design for usability (e.g., Nielsen 1993).

Usability approaches are prominent in the sequencer and DAW, and their use of visual metaphor, which allows the user to apply knowledge learned elsewhere, thus minimizing the need for further learning (Duignan et al. 2004). However, controlling virtual representations of physical devices allows only a limited transfer of the associated procedural knowledge learned with the original device: motor skills, built on the learning of spatial schemata and haptic feedback, cannot be transferred, nor easily redeveloped using the mouse (Smyth et al. 1994). Moreover, dynamic layouts and windowing can impede learning of the interface, requiring a visual search before most interactions to locate the window, icon, menu, or pointer.

Many principles of usability design are outlined by Nielsen (1993), in his set of *usability heuristics*, used in the design and evaluation of user interfaces. While advocating minimizing a user's memory load ("recognition rather than recall"), he also suggests "shortcuts" for experienced users ("unseen by the novice user"). Similar design principles, which treat the computer as a fundamentally visual medium, are evident in most modern consumer software, including audio software like sequencers and DAWs, in contrast to those for hardware audio interfaces, which focus on skilled interaction, motor learning, and nonvisual feedback modes, such as haptics and sound (Paradiso and O'Modhrain 2003). Consequently, in the next section, we propose design heuristics for computer music interfaces, which specifically account for virtuosity and nonvisual feedback, and which are designed to aid the development of user experiences supporting the creative process, drawing on concepts of feedback, liveness, and direct involvement.

23.5 Design Heuristics for Virtuosity

Following the principles presented above, we suggest design heuristics for interfaces to support virtuosity. Designing *multilayered* interfaces that suit both novice and expert users presents design challenges (Shneiderman et al. 2005). A distinction is made in the targeting of expert users; a virtuosity-enabled system enables a novice user to become expert. It does not rely on domain expertise learned elsewhere (e.g., music literacy), but should consider the transferability of skills learned.

Some of these heuristics draw upon and develop the recommendations of a recent workshop report on creativity support tools (Resnick et al. 2005). Various aspects of computer-based notations are also discussed in the context of the *cognitive dimensions of notations* (CD) framework (Green 1989), which has been previously used to highlight interaction issues in music software (Blackwell and Green 2000).

Heuristic$_1$ (H_1): Support learning, memorization, and prediction (or "recall rather than recognition")

Expert methods can be enabled by the use of memory (Smyth et al. 1994). Although some interface widgets allow both novice and expert interaction (e.g., the use of mnemonics, in menu accelerators), provisions for usability (e.g., "recognition rather than recall"; Nielsen 1993) can hamper experts (Gentner and Nielsen 1996) and their impact should be considered carefully in systems designed for virtuosity. Using memory, interaction is no longer mediated through visual metaphors fixed by the interface designer, but by schema derived from physical interaction and personal experience.

Notations should not aim solely to be "intuitive," rely heavily on domain-specific knowledge, or otherwise devalue the learning experience. Instead, they should provide a rewarding challenge that scales with user experience (Csikszentmihalyi 1996). Shneiderman and others (2005) describe a similar requirement that creative support systems should have "low threshold, high ceiling, wide walls," respectively offering: a minimal initial learning barrier to support novice use (see H_3); a maximal scope for advanced and more complex edits to facilitate the greater ambitions of experts; and the opportunity for users to define their own paths and working processes, without being constrained to established systems or practices.

Unfortunately, HCI methodologies provide limited account of "learnability" (Elliot, Jones, and Barker 2002), either assuming prior user expertise or explicitly obviating the learning requirement. Although the CD framework (Green 1989) reserves judgment as to the desirability of various aspects (dimensions) of a notation, the presence of *hard mental operations* is invariably viewed as a negative quantity, in HCI. In the context of virtuosity, perhaps we have found a context in which such mental challenges are actually beneficial.

H_2: Support rapid feedback cycles and responsiveness

To master a system, its behavior must be "transparent" (Holtzblatt, Jones, and Good 1988; Kitzmann 2003), allowing the user to easily equate cause with effect in their interactions. Reducing the delay between action and reaction is an effective way to achieve this (Leman 2008).

In computer interaction, basic control feedback should be provided within approximately 100 ms (Nielsen 1993) to appear instantaneous. Complicated operations should complete within roughly 1s (~300 ms to 3 s), or otherwise risk interrupting the flow of thought. After 10 s of idleness, users actively become restless, and will look to fill the time with other tasks. As such, longer delays, especially those requiring wait cursors or progress meters, should be avoided; and are "only acceptable during natural breaks in the user's work."

To support live performance and recording, there are even stricter criteria for a music system, which must respond within a few milliseconds (Walker 1999). Dedicated low-latency sound drivers (e.g., ASIO, WDM) have been developed to provide such latencies, typically confining delays to under 25 ms, and potentially as low as 2 ms. Even below this threshold, musicians and professional recording engineers are sensitive to jitter (the moment-to-moment fluctuations of clock pulses, measured in nanoseconds), but the impact is perceived in terms of sound quality (the introduction of noise and enharmonic distortions, and deterioration of the stereo image), rather than system responsiveness. While less "live" interactions such as playback control and general UI responses tolerate higher latencies, longer delays nonetheless affect the perceived directness of the user experience. Table 23.2 summarizes these requirements for interaction in a musical system. A relationship between timing and control emerges; the finer the required control, the tighter the demands on responsiveness.

As much as the timing, the quality of feedback also affects perceived "liveness" of a system (Church, Nash, and Blackwell 2010; Nash and Blackwell 2012). Liveness, in the context of notation use,[1] is a quality of the design experience that indicates how easy it is for users to get an impression of the end product during intermediate stages of design. UI designers should apply the timing constraints in Table 23.1 to both visual and musical feedback, delivering them in synchrony, where possible. At the same time, increased liveness can reduce the opportunity for useful abstraction and increase the skill required

Table 23.2 Timing of feedback in a music system, listing the changing perceptions of delays at different timescales, and consequences for interaction if they are exceeded (Nielsen 1993; Walker 1999).

Timing	Perception	Consequence if violated
< 1 ms	sound quality ("tightness," "jitter")	user hears noise artifacts, inharmonic distortions, muddied stereo image
< 25 ms	real-time audio ("low latency")	user has difficulty keeping musical time, maintaining sync during performance
< 100 ms	"instantaneous" UI response	system feels slow and unwieldy, harming user's performance and sense of control
< 1 s	noticeable delay	user has difficulty planning ahead or maintaining "flow of thought" and continuity of action
< 10 s	tolerable delay	user loses focus, attention wanders to other tasks

to interact with a program. For example, music programs usefully abstract time, allowing the editing of events at arbitrary points in the music, whereas real-time music input requires performance skills.

In programming, promoting liveness is an example of the push to accelerate the feedback cycle in software design, complimenting the philosophies of similar moves toward *rapid application development*. In rapid application development, the early availability of testable prototypes allows more flexible targets, and supports experimentation and ideation (Resnick et al. 2005), both of which facilitate creativity (Sternberg 1999).

Authoring software, including music production software, has seen the provision of increasingly complex functionality, which can not only be difficult for the user to understand, but can also take time to execute, reducing the perceived responsiveness of the system (Resnick et al. 2005). An interaction designer, in automating trivial yet laborious tasks, must also take note of the period a user must then spend idle (see Table 23.1). In both cases, the goal is to keep the user active and engaged, and avoid interrupting the flow of action.

H_3: Minimize musical (domain) abstractions and metaphors

In HCI, UI designers try to reify the user's "mental model" to represent and operationalize a task domain (Norman 1993), using formal abstractions and metaphor (Duignan et al. 2004; Blackwell 2006) for processes, properties, states, relationships. The formalisms of any notation determine the expressive flexibility it allows, shaping a user's perspective of the domain, or even a culture's at large (Sloboda 1985). It is difficult for an interface designer to match the user's internal representation of musical expression without inadvertently shaping or even limiting the creative output (Blackwell and Collins 2005; Kitzmann 2003). This is particularly the case when designing a unified interface for an artistic audience, who define themselves by their uniqueness and innovation.

Although classically trained or musically literate users will share many perceptions of musical structure, there are few widely accepted formalisms encapsulating the full gamut of computer music capabilities, and nondigital abstractions (such as wires, pots, pedals, mixers, or other metaphors from electronic and acoustic music; see Duignan et al. 2004) can be confusing, confining, and cumbersome. In such cases, "the *closeness of mapping* to conventional audio processing equipment…is indicative of a corresponding reduction in potential for creative exploration" (Blackwell and Collins 2005).

Resnick and his colleagues (2005) advocate avoiding the use of higher-level abstractions, in favor of low-level primitives that can be layered and combined to produce equivalent or greater functionality by the user. At the same time, the simpler functionality of each primitive makes it easier to understand and learn. As more primitives are layered and combined, the challenge increases, providing a scalable learning experience, toward the development of broader mastery. Turkle and Papert (1990) call this approach "soft-mastery," observing that it encourages "closeness to the object," and that such bottom-up perspectives are common in fine artists and musicians.

A visual UI presents explicit abstractions of the music, but the interface designer should also not overlook the implicit abstractions a user will make by simply listening to the sound. In this capacity, simply the broad availability of such musical feedback may be the best scaffolding enabling users to form their own abstractions. In this sense, such "audibility" may be seen as a correlate of the *visibility* dimension in the CD framework, which concerns the degree to which systems "bury information in encapsulations" (Blackwell and Green 2003).

H_4: Support consistent output and focused, modeless input

An interface that remains consistent, from moment to moment, can be more easily remembered and predicted. Fixed, static layouts enable the development of not only spatial schemata, but also motor learning (Smyth et al. 1994), both of which allow a degree of interaction to be handled unconsciously (see H_1). Changeable, dynamic screen layouts, such as floating windows, require conscious reflection, interrupting thought processes and hampering the performance of experienced users. Users should not have to visually search through different windows, modes, or other views, to locate information or effect minor edits, but "should be able to perform any task at any time" (Gentner and Nielsen 1996).

Most programs present a primary notation, which is kept in view for the majority of the time, and which constitutes the focus of activity, for example the source code in an integrated development environment (IDE), the document in a word processor, or the waveform in a sound editor. Such a focal point allows a user to maintain concentration. By contrast, a decentralized interface split across multiple subdevices hides functionality, complicating learning and use of the program.

Hardware controllers (e.g., MIDI devices, control surfaces, digital mixers) can make visual metaphors tangible and enable peripheral interaction (Paradiso and O'Modhrain 2003), often presenting fixed, physical layouts that can aid motor learning. These devices support more direct involvement with music in a live context, but potentially draw attention away from the software environment, and may have limited use in non-real-time edits.

H_5: Support informal interaction and secondary notation.

Blackwell, Church, and Green (2008) noted that formalism is an unavoidable consequence of encoding processes in the digital domain, but also that UI conventions have, at times, restricted the use of design software. Even when using digital tools, users often support more complex interaction with pen and paper, to make notes, reminders, calculations, or sketches of representations not easily or quickly executed in the UI (Sellen and Harper 2001). Earlier, we discussed the benefit this medium brought to music, in the composer's use of the musical sketch. However, there is no reason why similar informal, low-fidelity representations can't be supported digitally (Blackwell et al. 2008).

In the CD framework, this provision is known as a secondary notation and specifically describes those aspects of a programming environment that allow a user to record information without using the formalisms in the main notation. In a score editor, for example, this might involve allowing the user to annotate any part of the score with text or drawings. Because such notes needn't be interpreted by the program in any way (e.g., as music), users are not limited in what they can write or draw, allowing them not only to quickly scribble down ideas while fresh in their mind, but also record information and ideas that the program's designers never thought to support explicitly.

23.6 Systems of Musical Flow

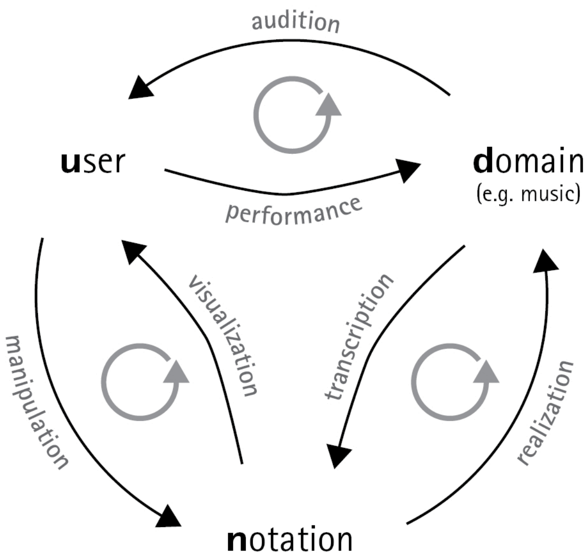
Similar to stage theory (see section above), many HCI methodologies (e.g., GOMS/KLM, Cognitive Walkthrough) address scenarios characterized by a well-defined problem or task, entailing established interactive procedures and known solutions (e.g., "correct actions") (see Sharp, Rogers, and Preece 2007). Artistic creativity, by comparison, can be likened to a "wicked problem": ongoing, ill-defined, with no "correct" solution or definitive end condition (Rittel and Webber 1973). Consequently, the creative user experience can be difficult to model using standard HCI methods. In this section, we propose a modeling approach to the creative user experience, which combines our previously discussed theories of feedback, liveness, and virtuosity, with Csikszentmihalyi's theory of *flow*. This theory describes a mental state, where a delicate *balance of challenge and ability* leads to a *feeling of control* and a *loss of self-consciousness*, engendering a working environment that can benefit creativity (Csikszentmihalyi 1996). Over time, ability increases, requiring greater challenges to maintain flow, ultimately leading to the development of mastery in a domain. In this context, the flow concept describes an intrinsically rewarding path to building ability, through enjoyable and fulfilling challenges matched to the individual, engendering a scalable learning experience.

Earlier sections identified the central role of feedback in creative user experience, as recognized in flow theory's call for *direct and immediate feedback* (Csikszentmihalyi 1996). Timely, unambiguous modes of feedback, in interfaces that favor fast audio feedback rather than abstract visual representations, also help a user to identify *clear goals* during interaction, and the actions taken to meet them, enabling them to learn the system through tinkering (Scripp, Meyaard, and Davidson 1988). Further components of flow have also been touched on, such as *concentration and focus* and the *feeling of control* (*autonomy*). More generally, feedback enables users to track their progress, allowing them to reflect on their musical creativity, and providing either a source of satisfaction or prompting further editing. In this way, interaction with the music system can be *intrinsically rewarding*, if progress is maintained and distractions or interruptions are avoided.

This section presents a model of interaction within interactive systems designed for authoring or creative design, such as music. Rather than explicitly identifying goal-oriented, stepwise, or linear processes, the model seeks to characterize the moment-to-moment feedback, flow, and liveness properties of the system to highlight issues in the user experience that might be detrimental to the performance and motivation of users, and thus also their creativity.

23.6.1 Overview

We model the creative user experience as a combination of closed feedback loops (Figure 23.2), between the *user, domain*, and a system's *notation* (i.e., interface). Musical performance is modeled as *direct involvement* with the domain, where the user's expertise with the performance device (e.g., instrument) supports *embodied interaction* (Leman 2008). At the same time, a user can interact indirectly with the domain, through the notation. In this latter scenario, the speed of notation manipulation and the speed of audio realization and musical feedback determine the liveness of the experience. A system that enables fast manipulation of the notation and fast audio feedback, synchronous with visual feedback, involves the user more directly in the domain (see Section 23.3), without sacrificing the utility of a visual notation.



Click to view larger

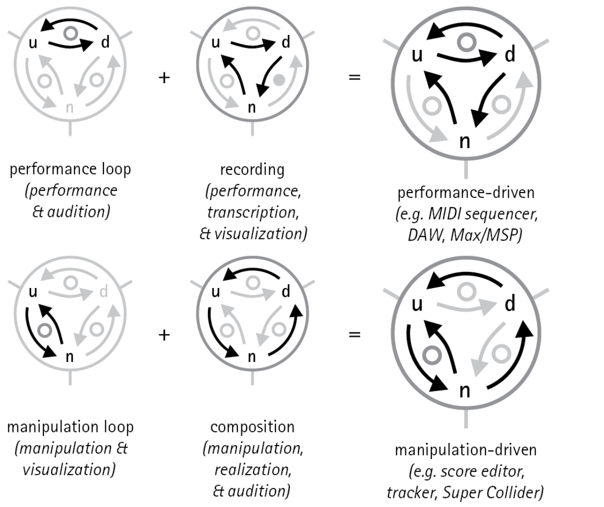
Figure 23.2 The systems of musical flow model of interactive music, characterizing the user experience as a network of feedback loops based on interactions with and between the task domain (e.g. music) and notation.

Systems are modeled by evaluating the emphasis of each arc or loop in a given music interaction or interactive system. In addition to the three basic loops indicated in Figure 23.2, larger cycles emerge within the combinations of arcs and loops, notably defining broad user-interaction paradigms that span the user, notation, *and* domain: a *recording* paradigm (*performance → transcription → visualization*) and a *composition* paradigm (*manipulation → realization → audition*). Combinations of these cycles with the smaller loops can be used to define basic system archetypes, which can be used to characterize most real-world music systems, such as sequencers, score editors, and trackers.

Figure 23.3 shows the synthesis of two popular archetypes, but space constrains us from exhibiting the full (super)set in this chapter, which extends the model to encapsulate applications such as augmented instruments, algorithmic composition, Max/MSP, or even *Guitar Hero*-like programs (see Nash 2012) and thus may form the basis for a taxonomy of computer music interaction.

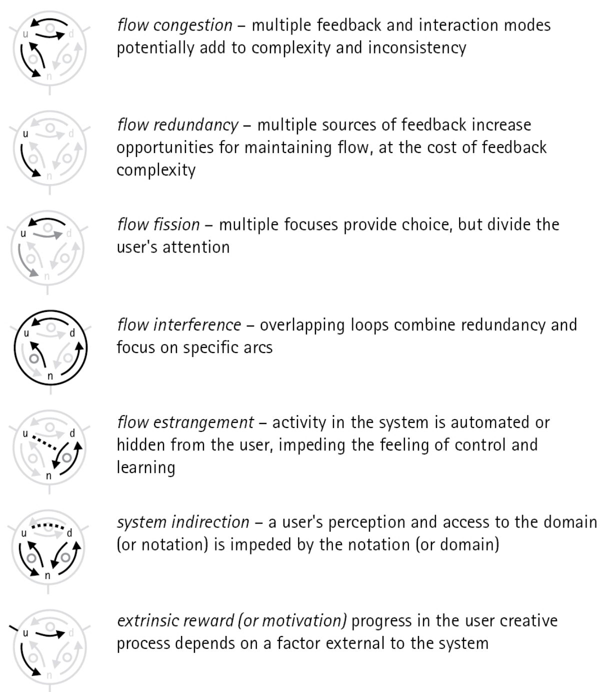
23.6.2 Application

The capacity for flow is indicated by the level of liveness of each interaction loop, as annotated in Figure 23.5, but is also influenced by the basic configuration of the program. Real-world systems may be many times more complicated than their underlying archetype, involving more than one notation, or even domain. Features of the modeled system can highlight interaction issues that may impact the associated user experience, as illustrated and briefly described in Figure 23.4.



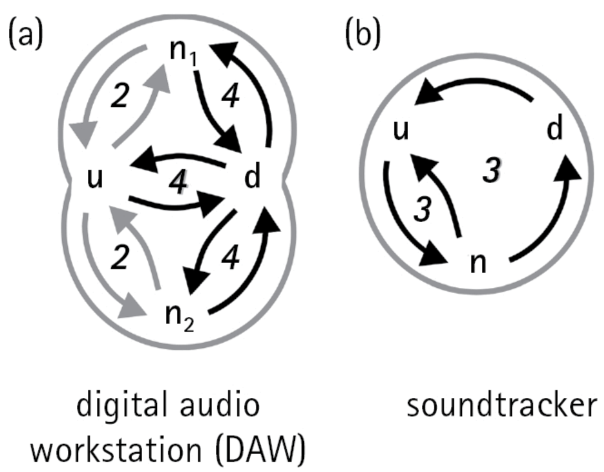
Click to view larger

Figure 23.3 Example system archetypes, characterized as the synthesis of feedback loops coupling notation and domain, with real-world examples. See also Nash (2012) for further examples and discussion.



Click to view larger

Figure 23.4 Interaction issues highlighted by system features.



For example, a digital audio workstation (DAW) (Figure 23.5a) provides many features and interfaces beyond the basic sequencer model. The system is split between multiple visual notations ("devices," in the program), arranged around a central performance loop. As established in our earlier discussions, Level 4 liveness is offered during performance, but interaction with the various notations is more cumbersome. No interaction scenario that includes the user, a notation, and the domain exhibits liveness above Level 2. Moreover, the congestion of arcs intersecting the user, caused by multiple partial-notations (or "subdevices"), highlights issues with divided attention and focus switching (Flow Congestion and Flow Fission in Figure 23.4), also affecting the consistency and complexity of the user interface or interaction.

By comparison, the simpler *soundtracker* (Figure 23.5b) model supports rapid edit–audition cycles through a single, more-encompassing notation. Naturally, this unified representation of the domain is likely more complicated than any one of the DAW's subdevices, and architects of a system must balance usability and virtuosity considerations; for example, using standard usability techniques (e.g., Nielsen 1993) and the heuristics proposed in Section 23.5. Naturally, there is nothing to prevent designers integrating a faster, more powerful, expert-oriented notation as an *extra* subdevice in the DAW, with minimal impact on the current usability provided by existing notations, which could be used by novices and ignored by experts.

Click to view larger

Figure 23.5 Real-world system examples, annotated with respective liveness levels for constituent feedback loops.

23.7 Conclusions

Musical creativity relies on exploration, expertise, and intrinsic motivation, contrasting goal-based usability approaches in mainstream HCI practice and analysis. In music, creative individuals not only develop virtuosity with instruments, but also notations, and while new performance devices provide new modes of real-time musical expression, relatively little research has looked at the composer's use of notation, and how it can be supported by the computer, as a tool for abstracting, sketching, and exploring creative ideas. Moreover, the linear recording process predominant in existing performance-based music software, such as sequencers and DAWs, can be seen to serve only an ancillary role in creativity, focusing on the transcription or refinement of a musical idea, and not the earlier, more exploratory stages of creativity (see Blackwell and Green, 2000; Duignan 2004; Smith, Mould, and Daley 2009).

Integrating psychologies of expertise and intrinsic motivation, this chapter introduces flow theory (Csikszentmihalyi 1996) as a framework for identifying and analyzing the properties of a notation-based music system that supports learning and creativity, integrating it with HCI theories of liveness (Tanimoto 1990) and the cognitive dimensions of notations (Green 1996), working toward a theoretical foundation for HCI in digital music.

Supporting flow in notation use entails a design shift from usability to virtuosity; advocating user interfaces for composition that facilitate a rapid edit–audition cycle and the development of skill (motor and memory) with input devices. By improving liveness and virtuosity (e.g., motor learning), programs can support embodied interaction through notation and levels of immersion and flow in music that are comparable to those found with live performance devices, as evident in studies of interaction with tracker notation (Nash and Blackwell 2011, 2012). Expert tracker users, for example, trade off editing complexity to maintain faster feedback cycles that preserve the liveness of interaction, favoring shorter editing episodes and more iterative editing styles, with more frequent playback of material (Nash and Blackwell 2012), facilitated by fluency with the QWERTY keyboard (Nash and Blackwell, 2011).

Another critical component of flow is the balance of challenge and ability. Through moving away from real-time music, users also gain control of the pace of interaction, enabling them to self-regulate the challenge involved, and preserving a sense of autonomy. For novices, this relaxes the virtuosity required to engage in the musical domain, lowering the threshold for creativity (Scripp, Meyaard, and Davidson 1988). At the same time, it gives experts the time to consider and experiment with more complex, advanced, and original musical solutions (compared with what is solvable in real-time performance or improvisation, see Sloboda 1985), ultimately raising the ceiling of creativity.

The goal of this chapter was to provide the designers of interactive audio systems with a new perspective on the musical interaction engendered by their systems, emphasizing aspects of the user experience that are not easily encapsulated using traditional HCI methods, nor widely supported in UIs for authoring music. Our hope is that the concepts, recommendations, and theoretical models presented here—including virtuosity, liveness, and flow—will encourage wider discussion and development of the HCI issues and theory facing interactive audio practitioners.

Our research was funded by the Harold Hyam Wingate Foundation. In the course of this research, Nash spent several months as a research intern with Steinberg and is grateful for input, support, and assistance from colleagues at Yamaha, Steinberg, and Cakewalk.

References

Amabile, Teresa M. 1983. *The Social Psychology of Creativity*. New York: Springer-Verlag.
[+] Find This Resource

Auslander, Philip. 1999. *Liveness: Performance in a Mediatized Culture*. Abingdon, UK: Routledge.
[+] Find This Resource

Blackwell, Alan F. 2006. The Reification of Metaphor as a Design Tool. *ACM Transactions on Computer-Human Interaction (TOCHI)* 13 (4): 490–530.
[+] Find This Resource

Blackwell, Alan. F., Luke Church, and Thomas. R. G. Green. 2008. The Abstract is "an Enemy": Alternative Perspectives to Computational Thinking. *Proceedings of PPIG 2008* 20: 34–43.
[+] Find This Resource

Blackwell, Alan. F., Luke Church, Beryl Plimmer, and Dave Gray. 2008. Formality in Sketches and Visual Representation: Some Informal Reflections. In *Sketch Tools for Diagramming*, ed. Beryl Plimmer and Tracy Hammond, 11–18. Workshop at VL/HCC 2008. Washington, DC: IEEE Computer Society.
[+] Find This Resource

Blackwell, Alan F., and Nick Collins. 2005. The Programming Language as Musical Instrument. *Proceedings of PPIG 2005* 17: 120–130.
[+] Find This Resource

Blackwell, Alan F., and Thomas R. G. Green. 2000. A Cognitive Dimensions Questionnaire Optimised for Users. *Proceedings of PPIG 2000* 12: 137–152.
[+] Find This Resource

——. 2003. Notational Systems: The Cognitive Dimensions of Notations Framework. In *HCI Models, Theories and Frameworks: Toward a Multidisciplinary Science*, ed. J. M. Carroll, 103–134. San Francisco: Morgan Kaufmann.
[+] Find This Resource

Church, Luke, Christopher Nash, and Alan F. Blackwell. 2010. Liveness in Notation Use: From Music to Programming. *Proceedings of PPIG 2010*, 22: 2–11.
[+] Find This Resource

Collins, David. 2005. A Synthesis Process Model of Creative Thinking in Music Composition. *Psychology of Music*, 33: 193–216.
[+] Find This Resource

Csikszentmihalyi, Mihaly. 1996. *Creativity: Flow and the Psychology of Discovery and Invention*. New York: HarperCollins.
[+] Find This Resource

Duignan, Matthew, James Noble, Pippin Barr, and Robert Biddle. 2004. Metaphors for Electronic Music Production in Reason and Live. In Computer Human Interaction, *Lecture Notes in Computer Science (LNCS)*, ed. M. Masoodian, S. Jones, and B. Rogers, 3101: 111–120. Berlin: Springer.
[+] Find This Resource

Elliot, G. J., E. Jones, and P. Barker. 2002. A Grounded Theory Approach to Modelling Learnability of Hypermedia Authoring Tools. *Interacting with Computers* 14: 547–574.
[+] Find This Resource

Gentner, Don, and Jakob Nielsen. 1996. The Anti-Mac Interface. *Communications of the ACM* 39 (8): 70–82.
[+] Find This Resource

Graf, Max. 1947. *From Beethoven to Shostakovich*. New York: Philosophical Library.
[+] Find This Resource

Green, T. R. G. 1989. Cognitive Dimensions of Notations. In *People and Computers V*, ed. Alistair Sutcliffe and Linda Macaulay, 443–460. Cambridge, UK: Cambridge University Press.
[+] Find This Resource

Holtzblatt, Karen A., Sandy Jones, and Michael Good. 1988. Articulating the Experience of Transparency: An Example of Field Research Techniques. *SIGCHI Bulletin* 20 (2): 46–48.
[+] Find This Resource

Kitzmann, Andreas. 2003. Transparency, Standardization and Servitude: The Paradoxes of Friendly Software, In *Perspectives on Multimedia: Communication, Media and Information Technology*, ed. Robert Burnett, Anna Brunstrom, and Anders G. Nilsson, 41–54. Chichester, UK: John Wiley.
[+] Find This Resource

Leman, Marc. 2008. *Embodied Music Cognition and Mediation Technology*, Cambridge, MA: MIT Press.
[+] Find This Resource

MacDonald, R. 2007. Trackers! *Computer Music* 113: 27–35.
[+] Find This Resource

Nash, Christopher. 2012. Supporting Virtuosity and Flow in Computer Music. PhD diss., University of Cambridge. **http://research.nashnet.co.uk**.
[+] Find This Resource

Nash, Christopher, and Alan Blackwell. 2011. Tracking Virtuosity and Flow in Computer Music. *Proceedings of ICMC 2011*, 575–582. Huddersfield, UK 31 July–5 August.
[+] Find This Resource

——. 2012. Liveness and Flow in Notation Use. *Proceedings of NIME 2012*, 28–33. **http://www.eecs.umich.edu/nime2012/Proceedings/NIME2012WebProceedings.html**.
[+] Find This Resource

Nielsen, Jakob. 1993. *Usability Engineering*. Cambridge, MA: AP Professional.
[+] Find This Resource

Norman, Donald. 1993. *The Psychology of Everyday Things*. New York: Basic Books.
[+] Find This Resource

Paradiso, Joseph A., and Sile O'Modhrain. 2003. Current Trends in Electronic Music Interfaces. *Journal of New Music Research* 32 (4): 345–349.
[+] Find This Resource

Resnick, Mitchel, Brad Myers, Kumiyo Nakakoji, Ben Shneiderman, Randy Pausch, Ted Selker, and Mike

Eisenberg. 2005. Design Principles for Tools to Support Creative Thinking. In *NSF Workshop Report on Creativity Support Tools*, ed. Ben Shneiderman, Gerhard Fischer, Mary Czerwinski, Brad Myers, and Mitch Resnick **http://www.cs.umd.edu/hcil/CST/**.
[+] Find This Resource

Rittel, Horst W. J., and Melvin Webber. 1973. Dilemmas in a General Theory of Planning. *Policy Science* 4: 155–169.
[+] Find This Resource

Scripp, Larry, Joan Meyaard, and Lyle Davidson. 1988. Discerning Musical Development: Using Computers to Discover What We Know. *Journal of Aesthetic Education* 22 (1): 75–88.
[+] Find This Resource

Sellen, Abigail J., and Richard H. R. Harper. 2001. *The Myth of the Paperless Office*. Cambridge, MA: MIT Press.
[+] Find This Resource

Sharp, Helen, Yvonne Rogers, and Jenny Preece. 2007. *Interaction Design: Beyond Human-Computer Interaction*. Chichester, UK: Wiley.
[+] Find This Resource

Shneiderman, Ben, Gerhard Fischer, Mary Czerwinski, Brad Myers, and Mitch Resnick. 2005. *NSF Workshop Report on Creativity Support Tools*. **http://www.cs.umd.edu/hcil/CST/**.
[+] Find This Resource

Sloboda, John A. 1985. *The Musical Mind*. Oxford: Oxford University Press.
[+] Find This Resource

Smith, Jeff, David Mould, and Mark Daley. 2009. Constructures: Supporting Human Ingenuity in Software. *Digital Creativity* 20 (1–2): 79–94.
[+] Find This Resource

Smyth, Mary M., Alan F. Collins, Peter E. Morris, and Philip Levy. 1994. *Cognition in Action 2nd edn*. Hove, UK: Lawrence Erlbaum.
[+] Find This Resource

Sternberg, Robert, ed. 1999. *Handbook of Creativity*. Cambridge, UK: Cambridge University Press.
[+] Find This Resource

Tanimoto, Steven L. 1990. VIVA: A Visual Language for Image Processing. *Journal of Visual Languages and Computing* 1: 127–139.
[+] Find This Resource

Turkle, Shelley, and Seymour Papert. 1990. Epistemological Pluralism. *Signs* 19 (1): 128–157.
[+] Find This Resource

Walker, Martin. 1999. Mind the Gap: Dealing with Computer Audio Latency. *Sound On Sound*, April. Cambridge, UK: SOS. **http://www.soundonsound.com/sos/apr99/articles/letency.htm**.
[+] Find This Resource

Wallas, Graham. 1926. *The Art of Thought*. New York: Harcourt-Brace.
[+] Find This Resource

Webster, Peter. 2002. Creative Thinking in Music: Advancing a Model. In *Creativity and Music Education*, ed. Timothy Sullivan and Lee Willingham, 16–34. Toronto: Britannia.
[+] Find This Resource

Notes:

(1) . Distinct from other uses of the term "liveness" in music, used to describe the immersive quality of a live performance viewed from the social perspective (e.g., that of the audience; Auslander 1999). See Nash and Blackwell (2012).

Chris Nash

Chris Nash is a professional programmer and composer, who completed his PhD on music HCI at the University of Cambridge. His research looks at theoretical and analytical methods for modeling and designing interfaces for composition, supported by a longitudinal study of over 1,000 DAW users, empirically investigating user experience with respect to flow, learning, virtuosity, creativity, and liveness. His work has attracted interest from Yamaha, Cakewalk, and Steinberg, whom he joined for 6 months in 2009. Around his research, he is the developer of the award-winning reViSiT plugin, and has written music for TV and radio, including the BBC.

Alan F. Blackwell

Alan Blackwell is Reader in Interdisciplinary Design at the University of Cambridge Computer Laboratory. He is an authority on visual representation and notation, especially with regard to the usability of programming languages. He collaborates regularly with music researchers, especially through Cambridge's Centre for Music and Science, and has a specific research interest in notations for artistic production and performance, working with a wide range of contemporary choreographers and composers. Together with his students and collaborators, he has a long-standing interest in the tools and practices of Live Coding.