Understanding Interaction in Contemporary Digital Music: from instruments to behavioural objects

OLIVER BOWN, ALICE ELDRIDGE and JON McCORMACK

Centre for Electronic Media Art, Monash University, Australia, 3800
E-mail: Oliver.Bown@infotech.monash.edu.au

Throughout the short history of interactive digital music, there have been frequent calls for a new language of interaction that incorporates and acknowledges the unique capabilities of the computational medium. In this paper we suggest that a conceptualisation of possible modes of performance–time interaction can only be sensibly approached in light of the ways that computers alter the social–artistic interactions that are precursive to performance. This conceptualisation hinges upon a consideration of the changing roles of composition, performer and instrument in contemporary practice. We introduce the term *behavioural object* to refer to software that has the capacity to act as the musical and social focus of interaction in digital systems. Whilst formative, this term points to a new framework for understanding the role of software in musical culture. We discuss the potential for behavioural objects to contribute actively to musical culture through two types of agency: *performative agency* and *memetic agency*.

1. INTRODUCTION

Within the interactive computer music community, digital systems are commonly conceptualised in terms of the existing elements of music culture: software instruments, artificial improvisers, algorithmic composers, virtual listeners, and score-followers (Winkler 2001). This collection of metaphors draw from the classical triumvirate of *composition*, *performer* and *instrument*. Such a paradigm provides a framework for understanding the formative roles and relationships between the principle elements of a musical culture: how musical ideas develop; how they are propagated; and how they are actualised as sound. We refer to this existing paradigm as the *acoustic paradigm*.

Although countless works subvert this perspective, the acoustic paradigm outlines intuitive distinctions between the formative activities of musical culture: composing, performing and instrument making. In addition, clear roles are defined for both the people (performers, composers and luthiers) and objects (scores and instruments) implicated in these activities. It is recognised that such terms are particular to a specific (European) period of musical history, and the distinctions between them have been the subject of recent critical attention (Schroeder 2006; Waters 2007). Nevertheless, these terms persist as the principle conceptual currency for discussions of current musical practice.

Dissatisfaction with the language of the acoustic paradigm is evident in discussions of computer music production and performance. In recent years there have been calls for new interactive metaphors that take the *active* nature of the computational medium into account. The dominant metaphor of instrumental interaction emphasises a one-way reactivity that many feel is inappropriate for the conceptualisation, design and analysis of digital audio software. Hankering after a more collaborative form of interaction, we see discussion of *conversation models* (Winkler 2001; Chadabe 2002; Paine 2002) or systems with a degree of *cognition* (Bongers 2006). Such phrases capture a sense of mutual engagement in performance, but do not have the power of the acoustic paradigm in terms of linking object, activity and role.

In this paper we consider what work the metaphors of composer, instrument and performer do for us as software designers and users, and argue that they overlook important aspects of contemporary music practice. Software is helping to dissolve the distinctions between the objects, activities and roles of the acoustic paradigm, and is altering the ways in which we interact socially and musically. We attempt to provide a frame through which we can clearly apprehend both the objects and activities – the software and the social and artistic interactions it mediates – which are driving musical culture.

We argue that musical software introduces three types of interaction that are not covered by the acoustic paradigm: software acts as a new and distinct medium for interaction between people; software development in creative contexts involves a new and distinct cycle of interaction between the developer and the software; and software elements can also interact directly with each other in musically significant ways. This analysis leads us to construct a new term to view interconnected pieces of software: *behavioural objects*. This concept is aimed at emphasising the active nature of software (its behaviour) at the same time as its role

Organised Sound **14**(2): 188–196 © 2009 Cambridge University Press. Printed in the United Kingdom. doi:10.1017/S1355771809000296

as a tangible unit of social exchange, and as a creative tool (i.e. as an object). Behavioural objects can potentially exhibit complex behaviours like machines and organic structures, but can also be exchanged between people as rapidly and effortlessly as ideas. We consider the effect that this has on the process of change in musical culture. In addition, we define two types of agency that are relevant to interaction with behavioural objects: *performative agency*, which refers to the capacity for autonomy in musical software specifically in performance contexts, and *memetic agency*, which refers to the potential for musical software to exert influence in musical change over time.

The following section discusses how the language of the acoustic paradigm is applied in current software-based music production and performance. We look at a number of contemporary areas of activity and consider how the terms composition, performer and instrument are used, and the implications of this terminology. In Section 3 we discuss how digital technology in general, and software in particular, has driven a shift in the relations between people and the tools they use, and propose that this new dynamic needs to be included in the way that we understand music software systems (Section 3.1). We then introduce and discuss the term *behavioural object* (Section 3.2). We discuss agency in software systems in both performance contexts and longer-term cultural processes (Section 3.3).

2. THE ACOUSTIC PARADIGM

Early on in interactive music discourse, Winkler suggested that consideration of the interactive relationships that occur in traditional performance ensembles may be a useful starting point to evolve 'new modes of thought based on the computer's unique capabilities' (Winkler 2001: 21). Focusing on the issue of control and influence (who is in charge, who follows, who leads?), he offered four models based on different types of musical ensemble and their associated idioms: the conducted classical orchestra, the string quartet, the traditional jazz combo and a free improvisation ensemble. These models draw on the established roles of performer, composition and instrument in performance and use them to signify different modes of interaction. Continuums are drawn out between unidirectional *control* and mutual *influence*, fixed composition and collaborative improvisation. Similar dimensions are established by Rowe's dichotomies of player vs. instrument and score-driven vs. performance-driven paradigms (Rowe 1992).

2.1. Software instruments

Software is created using abstract, immaterial code, which is interpreted by a machine. This immateriality means that direct, physical control of software requires some form of *interface*. Here, the metaphor of the instrument is instructive in highlighting specific attributes of acoustic instruments that afford expressive performance potential. Design principles for new instruments (of the sort presented at the New Interfaces for Musical Expression conference (NIME), for example) are often derived directly from specific features of acoustic instruments: fine-grained continuous control, non-linear or coupled mappings between controller and sound-engine, pressure sensitivity, gestural relevance, and so on. In contrast, the Hyper-instrument group at MIT aim to extend the physical interfaces of traditional instruments in ways that preserve the same sensory-motor experience from the performer's perspective.

> Our approach emphasises the concept of 'instrument', and pays close attention to the learnability, perfectibility, and repeatability of refined playing technique, as well as the conceptual simplicity of performing models in an attempt to optimise the learning curve for professional musicians.
>
> (Machover and Chung 1989: 186)

On the premise that feedback – proprioceptive and vibrotactile as well as auditory – is key to achieving expressive control in acoustic instruments, other groups design physical interfaces for digital systems that produce a comparable response (e.g. Bongers 2006).

The basic physicality of instrumental interaction also shapes the instrument's design and performance aesthetics. For example Michel Waiswisz takes the fact that acoustic instruments are energised through physical effort as the premise for the 'energy' project, which pioneers the design of electronic instruments powered by the player's body (Waiswisz 2008). A similar commitment to a 'hands-on' approach is not uncommon among electronic musicians where a fundamental stipulation is that no digital signal processing (DSP) process is active unless the performer is manipulating the instrument in some way (Wessel 2006; Dahlstadt, personal communication). Aesthetic divergences aside, in all of these cases the instrument paradigm serves to guide the development of digital performance systems, which replicate the affordances of traditional acoustic instruments: learnability, perfectability and expressivity through direct control.

2.2. Virtual performers

At the other end of the scale, the potential of computers to become *virtual performers*, or *artificial improvisers* inspires the development of systems that take an active role in generating as well as actualising musical ideas. Many musicians in this area describe their motivations in terms of wanting to capture characteristics of interaction with other players. George Lewis, creator of the *Voyager* system,

phrased this succinctly by describing his desire to make machines that he can not only 'play', but that will 'play with him' (Lewis 2006). David Plans-Casals more recently described his motivation for making *Frank* as wanting to have 'someone to play with that wouldn't get bored of me and vice versa' (Waters 2007). Blackwell and Young have provided a basic framework for theorising about algorithmic performance systems, isolating particular characteristics of human improvisation as goals for the development of *live algorithms*: 'a live algorithm can collaborate actively with human performers in real-time performance without a human operator; a live algorithm can make apt and creative contributions to the musical dimensions of sound, time and structure' (Blackwell and Young 2006).

These characteristics are abstract and do not require human-like implementations, as Blackwell and Young's own interest in swarm dynamics indicates (Blackwell and Young 2004). Nevertheless, these systems are conceived, designed and discussed in terms of achieving characteristic aspects of interaction with another performer. Notions of control are replaced with influence and the exchange strives to be mutually influential. Here the cycle of interaction takes place through sound (rather than physical interfaces) and its content is musical ideas rather than physical actions or gestures. This is the sort of interaction that Winkler's free jazz model outlines, that Rowe's player paradigm strives for, and that many refer to as a form of *conversation* (Winkler 2001; Chadabe 2002; Paine 2002).

2.3. Composed instruments

Testimony to both the deep appeal of these terms and their fluidity when applied to contemporary practice is their deployment in various concatenations. The term *composed instrument* has been used since the early days of electronic music to signify the merging of composition and instrument into a single object. Pioneering composer–engineer Gordon Mumma recognised this feature of his electronic performance systems:

> … I consider that my designing and building circuits is really 'composing' … my 'instruments' are inseparable from the compositions themselves.
>
> (Mumma 1967)

Composing instruments is facilitated for electronic and digital systems by virtue of the fact that the sound-producing mechanism and control surface are decoupled, unlike traditional acoustic instruments (Chadabe 2002). In electronic and digital instruments, controllers, generators and the sets of mappings that link them are independent units that can be combined with great structural and functional flexibility. There is no longer a fixed and direct correspondence between the interface and sound production mechanism. This leads to several distinct (but not mutually exclusive) senses in which an *instrument* can be *composed*: a software instrument can combine a control interface with a generative algorithm, such that explicit temporal structure can be predefined; the instrument itself can be composed, in the sense that modules (controllers, mappings and generators and their subcomponents) are arranged with specific musical intent; in both these cases, the software system embodies some aspect of the maker's musical intent, and acts (like a score) as a vehicle for sharing musical ideas across culture. Figure 1 summarises this distinction.

Schnell and Battier use the term *composed instrument* to signify the fact that digital systems can 'carry as much the notion of an instrument as that of a score' (2002: 1). This highlights the fact – implicit in algorithmic composition – that computers can be used to predetermine specific structural aspects of a musical work as much as they can be used to realise as sound a musician's action in performance.

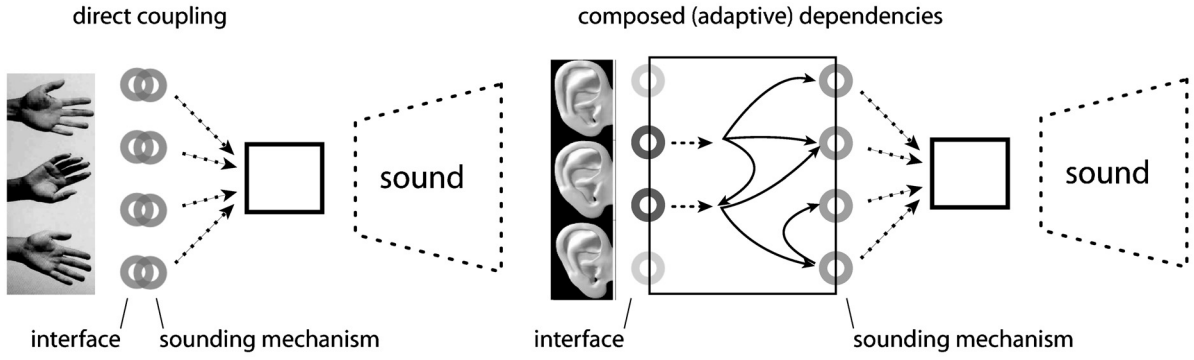

Figure 1. The decoupling between a control interface and the process of sound production in electronic instruments (right), as compared to acoustic instruments, which are inherently tightly coupled (left). This allows musicians to work compositionally with the design of instruments, with complex behaviours mediating control and sound output.

The decoupled nature of electronic and digital systems also requires the creative design of the software instrument itself. The functional units must be arranged in configurations that afford playability, and this also becomes drawn into the realm of composition in its broadest sense. As discussed above, events like NIME focus on the design of physical controllers for digital processes to facilitate expressive control. Others such as *ixi-software* (Enrike Hurtado and Thor Magnusson) focus on the link between standard interfaces (mouse, keyboard, tablet, etc.) and existing sound engines such as Reaktor (http://www.native-instruments.com), PureData (http://puredata.info) and MaxMSP (http://www.cycling74.com), designing environments in which the mapping itself can be controlled in real time (Magnusson 2007). For researchers such as David Wessel, the instrument (interface, DSP modules, mappings, feedback, etc.) must be built so that it can be played like an acoustic instrument, supporting learnability, precise control and so on, by providing the necessary structural support for sensory-motor and cognitive mastery (Wessel 2006).

The flexible modularity that facilitates 'composing instruments' in this sense implies an inversion of the relationship between instrument and performer. The instrument is no longer necessarily something that the performer learns to play by honing their motor skills and musical sensibilities. The programmer–performer, as individual or group, also learns to build instruments to fit performance needs, adapting system designs to fit cognitive–aesthetic predilections or sensory-motor competencies (see figure 2).

In these cases, various activities that were once reasonably peripheral to composition have become central to it. Software systems become compositions in that they embody some representation of their maker's musical idiolect: they have become a vehicle for sharing musical ideas rather than simply performing them.

The flexible modularity of software also facilitates the composition of instruments *as* performance. Assembling instruments as performance is by no means unique to electronic musicians. Improviser Eddie Prévost, for instance, plays with the re-construction of drum kits in performance – accumulating small cymbals on the head of a large kick drum (documented in Bowers and Villar 2006). The composition of ad hoc instruments as performance is explored explicitly by musician–theorists Bowers and Villar (2006) in their *Pin&Play&Perform* system, in which physical electronic components are assembled into circuits during the time of performance. A similar idea is explored in the performance-installation system *reactable* (Kaltenbrunner, Jordà, Geiger and Alonso 2006), and in perhaps the purest form in live patching of hardware and software synths, and new modes of software-based performance such as live coding (see http://www.toplap.org).

The language of the acoustic paradigm is easily reconfigured around various practices in computer music production and performance, and we readily refer to bits of software variously as instruments, composition systems and co-performers. But, as this discussion illustrates, the defining characteristics of instrument, performer and composition are weakened through an erosion of the traditionally recognised relations between them, particularly in terms of the roles taken by people in defining these relations.

3. TOWARDS A DIGITAL PARADIGM

3.1. Software and cultural interaction

The description of musical software systems in the language of the acoustic paradigm does not accommodate the role that software plays in the broader process of musical change and development, along with the respective shift that has taken place in the roles played by different individuals in a musical context. This requires a broader view of the interaction between people and software as a process happening as much outside of performance as within it. In this section we attempt to characterise interaction taking place over cultural development, with the goal

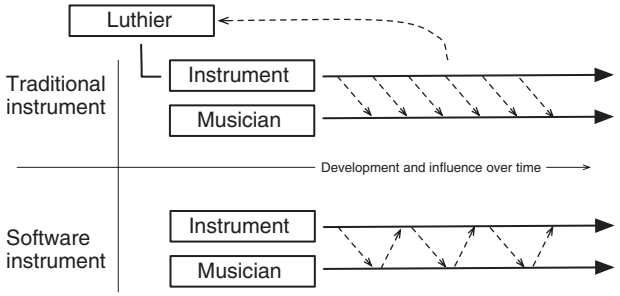

Figure 2. Two models for the developing relation between instrument and musician. At the top, a musician is adaptive towards an instrument (the direction of influence shown by dotted lines) as they interact over time. Influence feeds back into instrument design as a result of such interactions. At the bottom, the musician adapts to the instrument, but also modifies or reconfigures it during development.

of constructing a more inclusive view of music software systems, in Section 3.2.

Our observations here come from two broad categories of electronic music. The first is music that can be best described as experimental and includes academically and commercially produced music, such as electroacoustic music production and performance, experiments with artificially intelligent music software, and improvised electronic music inheriting from areas such as free improvisation, noise music and sonic art. Here, software developers commonly play an active part in the development of the musical concepts and the production of the music itself, and artist–programmers are common. The other style is best described as commercially produced electronic music (although this is intended to cover much music that is not mainstream or commercially viable), including much dance music and electronica. Here, musicians are less likely to be found developing software themselves, but their work is nevertheless tightly coupled with the development of new software through the exploration of new styles and technical possibilities (establishing new targets for commercial music technology).

It is now common for electronic musicians working with code or development environments such as Max/MSP to publicly release programs that embody aspects of their own compositional and performance practice. Leafcutter John's *Forester* (Burton 2008) is one example of a popular software download that many other musicians have subsequently employed in their work. The program processes samples loaded in by the user, and could be understood according to a number of established categories: effect, compositional tool, DJ tool or generative music system. The AV performance group Mabuse (Mick Grierson and Herbert Daniel, see http://mabuse.co.uk) have gone a step further by releasing a commercial software product of the same name, which embodies their compositional and performance practice. Similarly, ixi-software have released a software improvisation suite, *ixiQuarks* (Magnusson 2007), that embodies their aesthetic. These systems are geared towards an audience who are interested in using live audio and visual performance tools.

As idiosyncratic artistic creations, such software releases verge on musical works in their own right. They can also be understood as tools in the hands of their creators and their potential 'end users'. As such they are increasingly defining a channel through which interaction takes place.

Executable software joins many other types of digital data that have long been objects of exchange in electronic music production (particularly dance music and electronica): samples, grooves, patterns and presets. The remarkable diffusion of the Winstons' 'Amen break' through portions of our contemporary musical landscape illustrates this process (see http://en.wikipedia.org/wiki/Amen_break). The Amen break is a drum sample taken from the b-side of a record released by the Winstons in 1969. The sample has now been used in thousands of musical works, appearing on hip-hop tracks in the mid-1980s, and later, at a massively sped-up tempo, in drum and bass and jungle in the 1990s, where it effectively came to define a musical style. Here, musical culture can be seen to involve more than just the rapid social exchange of ideas, but also of immaterial but nevertheless tangible *objects*. The Amen break, and other samples like it, are immaterial objects (information or data) that are passed between individuals (perhaps not quite literally 'given' or 'exchanged', but nevertheless used by new musicians as a result of their experience of its use in other musical works). Mass production and manual duplication of audio recordings makes this process possible, and a musical culture that favours the process of reproducing and collaging existing musical samples creates the context in which one recording can play a role in so many musical works. The speed with which we can share such material through duplication, and now through digital networks, has clearly impacted on the organisation and development of musical genres.

In part, software simply epitomises this process of social interaction, which exists outside of musical performance. In the case of software, however, interaction does not involve the sharing simply of passive ideas or content, but of potentially active machines that can be employed for musical tasks. Whereas musical ideas may once have developed and circulated far more rapidly than the inanimate physical objects that define traditional musical instruments, software objects can now evolve and move around at just as fast a pace.

We see this as a major transition that is currently unfolding. What is not clear is what kinds of shared entities might become prevalent in different musical cultures. That specific drum samples would become objects of profuse reproduction was not an obviously predictable phenomenon, and it could have turned out that people would be averse to embedding somebody else's music in their own creative work. Although sampling is subject to the default authority of copyright control, its prevalence demonstrates its acceptance as part of a creative process. Likewise, there is little reason to believe that the adoption of increasingly active software objects, such as those made available by Leafcutter John, Mabuse and ixi-software, will be seen as a threat to creative practice, and every reason to believe that such objects could become part of the currency of musical culture.

The process of bringing system design into the remit of composition, discussed in Section 2.3, is therefore coupled with the process of networked sharing

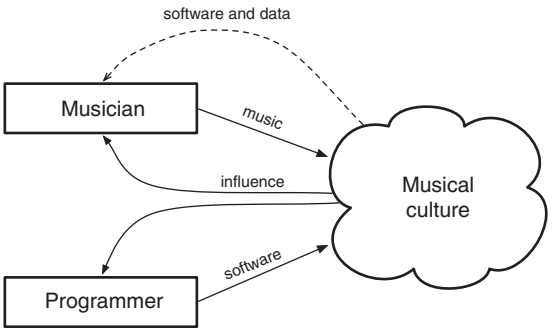

Figure 3. Interaction between musicians and programmers (possibly the same person) and a wider system of musical culture. Musicians and programmers both modify their goals and interests in response to the changing musical culture (influence). Musicians also draw new types of software and data (such as samples) from that culture.

in much contemporary music practice. An idealised diagram of this process is presented in figure 3, which shows the relationships between both musicians and programmers and their musical culture. This social element is evident in the popularity of computer music systems, such as Max/MSP, that maximise the potential for sharing objects at different levels of utility and complexity via mechanisms for easily organising and structuring components. This is by no means limited to music, but is an essential feature of all modern software development. In creative contexts, however, there may be greater potential for feedback: from usage to design goals. Creative contexts, for example, offer a great deal of potential for reappropriation of function.

3.2. Behavioural objects

Thus, in addition to the various modes of performance–time interaction between people and software systems (such as instrumental and conversational interaction), interaction between people via software objects plays a significant role in musical change, and may do so increasingly in the future. In this form of interaction, software is not just a communicative medium, but a tangible entity that can be interacted with and manipulated creatively. Similarly, in Section 2.3, we suggested the importance of interaction between people *and* software objects in a process of mutual development (developing, learning, configuring). Whilst traditional performer–instrument interaction is typified by the heteronomy and long-term stability of the instrument and the adaptation of the performer towards this structured musical environment, these are no longer points of stability: software systems may be partially autonomous, rapidly changing (over historical time), and adapted by the musician as part of his or her musical character.

One further form of interaction that is not addressed by the acoustic paradigm is that between software components themselves, such as objects in a Max/MSP or Pure Data patch, in realtime performance situations (and also, potentially, in non-realtime processes such as evolutionary simulations that may generatively create software). Whilst it is reasonable to treat such internal interaction as operating inside a black box, the 'software component' of any musical performance, this clearly conceals those interactions. Yet in many cases the final sonic outcome may significantly depend upon interactions *between* software routines (di Scipio 2003; Lewis 2006). Interaction between software components that is *not* modelled on human musical interaction, such as Blackwell and Young's Swarm Granulator (Blackwell and Young 2004), may ultimately become comprehensible to a sensitive audience.

All three of these modes of interaction are conspicuously present in computer music performance and production, and we believe that a constructive conceptual framework should accommodate them.

To this extent, we consider it appropriate to bring together in our conceptualisation of software its potential to take on the form of a tangible object (that can be the focus of various different forms of interaction) and its ability to exhibit complex temporal behaviours that vary widely in the ways we might characterise them. This analysis leads us to propose the term *behavioural object* as a descriptive term with which we can examine the interactive potential of software. We define a behavioural object as an entity that can act as a medium for interaction between people through its dissemination and evolution, can develop interactively with individuals in processes of creative musical development, and can interact with other behavioural objects to produce musical output.

The term *behaviour* satisfies the need we have to capture a continuum between the active qualities associated with software such as live algorithms to the more passive characteristic of a simple software synth, all of which can be described as possessing behaviour in a musical sense.

The term *object* refers primarily to a material thing that can be seen and touched. Software objects are immaterial, and quite clearly cannot literally be seen and touched: they require interfaces to mediate this interaction. However, they have a behavioural tangibility, just as the data that constitutes a digital recording of the Winstons' Amen break does. The term *object* also refers to a computational data construct that defines its own state, method of operation and interaction with other objects. The various relationships between classes and objects in the object-oriented programming paradigm, and in other forms of software development, provide an extensible set of

ways in which behavioural objects can engage in the set of interactions described above.

Behavioural objects can act as mediators between people in the development of musical styles and ideas (discussed in Section 3.1), moving through social networks and developing at the pace of ideas, unbounded by the materiality of physical objects. Behavioural objects can be extensively reconfigured by people, allowing the flexible adaptation of systems to performance contexts. Behavioural objects can interact with each other with generative consequences. In each of these forms of interaction, behavioural objects may be *active* in driving the overall process.

Understanding software systems as behavioural objects addresses these forms of interaction by replacing the distinction between active agent (performer) and passive object (instrument) with a continuum. Accordingly, the behavioural object perspective should be viewed as having a flexible scope. At the widest level, the scope of what objects can be understood as behavioural objects extends without limit to include material objects: all objects are tangible, have some form of active behaviour (resonance, for example), can mediate interactions between people, can engage in mutual development with people in musical creativity (although in both cases mostly driven by human creativity, where humans may creatively respond to the nature of the object), and can interact with other objects (again, resonance is an example). At this scale, behavioural objects provide a general framework for looking at current and past musical practices, emphasising the active role of non-human objects in determining musical outputs. For most objects (including many software objects), this poses an awkward stretch of the imagination, and is not particularly useful.

As the scope is reduced, a smaller set of behavioural objects remains, as satisfaction of the criteria for behavioural object status becomes tighter. Many present systems may exhibit some of the forms of interaction discussed above. Max/MSP and PD patches, and SuperCollider (http://supercollider.sourceforge.net) code, is shared amongst a community of users and hacked for specific uses. Autonomous performance systems, such as Lewis' *Voyager*, provide examples of systems that can develop musical ideas with other performers, and have developed through successive stages of interaction with their makers, whilst being tested in live performance contexts. Complex dynamical systems such as di Scipio's AESI (see below), demonstrate the generation of music through interactions between software elements. These works qualify as behavioural objects, but don't as yet typify them.

Towards the smallest extreme of scope, behavioural objects should be limited to systems currently only manifest as AI fantasies; computer programs which are, to all extents and purposes, human-like. In this case, the capacity for behavioural objects to mediate social relations, co-develop with others in creative musical practice, and interact musically with each other goes without saying. However, the behavioural object concept addresses a more general scenario in which future software needn't be particularly human-like in order to engage actively with human musical performance and culture.

3.3. Memetic and performative agency

Since behavioural objects are understood as being involved in forms of interaction beyond the performance context, we distinguish between two senses in which a behavioural object has agency: performative agency (in performance time) and memetic agency (out of performance time).

Performative agency refers to the ability of a software system to influence the outcome of a specific musical performance. For those primarily interested in human–computer collaborative improvisation systems, performative agency is directly synonymous with the quality of musical interaction it affords. A live algorithm is evaluated on its interactive potential (Blackwell and Young 2006), its ability to convincingly engage a human in improvisation. But even without human interaction, the DSP routines in Agostino di Scipio's Audible EcoSystemic Interface (AESI) have performative agency, directing the course of a musical performance in interesting ways. Lewis' Voyager system achieves performative agency in a more classical manner: it is driven by a set of carefully constructed interdependent processes that embody his musical acumen as an improviser, and so keep other performers engaged in the musical interaction (truly achieved if this goes so far as to make them feel that they need to engage it in return).

In contrast, memetic agency refers to the ability of a software system, broadly defined, to influence the evolution of musical styles over historical time. Meme theory looks at cultural processes as dynamical evolutionary systems in which cultural forms such as musical styles emerge. As such, these forms are not the direct result of individual human goals, but arise independently, as an indirect result of interaction between multiple agents with differing goals. Meme theory aspires to reveal the nature of human agency as a combination of the evolved forces generated by the various memes and genes that 'inhabit' our bodies (Dawkins 1976). It was originally proposed by biologist Richard Dawkins as a way to characterise the Darwinian process apparent in culture. There are many other ways that cultural change has been described in terms of evolutionary processes, and we do not state a preference for strongly Darwinian views of culture such as those of Richard Dawkins and Daniel Dennett (e.g. Dennett 1996) over alternatives

such as Actor Network Theory (e.g. Law 1992) or Sperber's use of notions such as cultural teleofunction (Sperber 2007).

If cultural evolution has a Darwinian dimension, then concepts, artefacts, institutions and other aspects of culture can be viewed as entities that act over time to maintain their own structure within cultural systems. Accordingly, the speed of development and propagation of software, and its potential to engage actively with people, makes it a potentially potent driver of cultural change.

Thus, performative agency is related to a behavioural object's ability to influence a performance; memetic agency relates to a behavioural object's influence across longer time scales. In both cases, the influence in question is an influence over other agents, who also play a role in directing a changing process, be it within one performance or over the course of historical musical change. In terms of human interaction in an artistic domain, this can often be usefully understood in terms of establishing relevance. This process can further be understood from a system theoretic or evolutionary perspective as establishing the conditions by which a system successfully maintains itself.

We consider the distinction between performative and memetic agency interesting because of the potential for memetic agency to feed back and influence performative agency over time. Performative agency typically implies the need for an individual developer to design a system's functionality and specify its relevance at performance time. This could conceivably be replaced by an emergent relevance, where the system's relevance as a musical agent goes beyond the input of any one designer or user (i.e. as the result of a blindly evolving cultural process), leading to classes of musical objects that no longer make sense within the acoustic paradigm. Such systems may not be recognisable as virtual performers from a traditional point of view (i.e. appearing to contribute to a musical performance in the same way that a human performer would), but could still contribute to the emergence of the appropriate performance context (e.g. musical genre) in which they can exhibit novel forms of performative agency. In other words, memetic agency could establish the conditions for performative agency.

4. CONCLUSION

In this paper, we have argued that software establishes a fundamentally different relationship between people and the objects they interact with, and that the acoustic paradigm of composer, performer and instrument does not accommodate certain aspects of this interaction that are relevant to an understanding of this relationship. Whilst metaphors of instruments and performers are valuable in defining goals for computer music, they overlook certain features of contemporary music practice. We have attempted to characterise emerging aspects of music software under the term *behavioural object*. A behavioural object has various properties in common with traditional instruments, but it plays a fundamentally different role within social interaction, and has a far greater capacity to exhibit varied forms of behaviour and to drive cultural change through its active and adaptive nature. The view we have constructed in this paper is one in which networks of behavioural objects interact with people, and increasingly with each other, in a complex system of digital music culture. Whilst, in principle, the memetic nature of this interaction is the same as that of material artefacts, its speed, flexibility and potentially active nature result in a different state of affairs in which there are objects in the world that do more than simply act as media for the relations between human users.

Our distinction between performative and memetic agency implies that performance can be thought of as the framing of a musical context. At performance time the term behavioural object provides a conceptual frame for viewing and designing a locus of interaction. This object is built using code, but is capable of inviting attributions of intentionality, unravelling its internal structure in a generative composition, or reacting to a call from either a human performer or some other routine in the system. The term also names, and thus brings to attention, an entity that is exchanged in musical culture, the pieces of code that can be shared, modified and repurposed and are the currency and building blocks both functionally and aesthetically in contemporary music culture. In doing so we celebrate the fluidity of the terms *composition*, *instrument* and *performer*.

The actual forms that this interaction will take are constantly and slowly becoming revealed, and the framework we have developed here is no more than an attempt to characterise the nature of the crisis of musical language and to propose a more appropriate frame. These ideas are formative, but aim to capture the range of different forms of interaction that underpin musical culture, both within and beyond musical performance.

ACKNOWLEDGEMENTS

This research is supported by Australian Research Council Discovery Project grants DP0772667 and DP0877320.

REFERENCES

Blackwell, T. and Young, M. 2004. Swarm Granulator. *Proceedings of The 2004 European Workshop on Evolutionary Music and Art*, Coimbra, Portugal.

Blackwell, T. and Young, M. 2006. Live Algorithms for Music Manifesto. Available from http://www.timblackwell.com.

Bongers, B. 2006. *Interactivation: Towards an e-cology of People, our Technological Environment, and the Arts*. PhD thesis, Vrije Universiteit, Amsterdam, Holland.

Bowers, J. and Villar, N. 2006. Creating Ad Hoc Instruments with pin&play&perform. *Proceedings of the 2006 Conference of New Interfaces for Musical Expression*, Paris. NIME.

Burton, J. 2008. Forester: Magical Sound Creation for Mac and PC. Available from http://leafcutterjohn.com.

Chadabe, J. 2002. The Limitations of Mapping as a Structural Descriptive in Electronic Instruments. *Proceedings of the 2002 Conference on New Instruments for Musical Expression*, Dublin, Ireland.

Dawkins, R. 1976. *The Selfish Gene*. Oxford: Oxford University Press.

Dennett, D. C. 1996. *Darwin's Dangerous Idea: Evolution and the Meanings of Life*. London: Penguin.

Di Scipio, A. 2003. Sound is the Interface: From Interactive to Ecosystemic Signal Processing. *Organised Sound* **8**(3): 269–77.

Kaltenbrunner, M., Jordà, S., Geiger, G. and Alonso, M. 2006. The Reactable*: A Collaborative Musical Instrument. *Proceedings of the Workshop on 'Tangible Interaction in Collaborative Environments' (TICE), at the15th International IEEE Workshops on Enabling Technologies (WETICE)*, Manchester, UK.

Law, J. 1992. Notes on the Theory of the Actor Network: Ordering, Strategy and Heterogeneity. Available from http://www.lancs.ac.uk/fass/sociology/papers/law-notes-on-ant.pdf.

Lewis, G. 2006. Improvising with Creative Machines. Talk given at Improvising with Computers, pre-NIME workshops. IRCAM, Paris.

Machover, T. and Chung, J. 1989. Hyperinstruments: Musically Intelligent and Interactive Performance and Creativity Systems. *Proceedings of the 1989 International Computer Music Conference*. Columbus, USA.

Magnusson, T. 2007. The ixiQuarks: Merging Code and GUI in one Creative Space. *Proceedings of the 2007 International Computer Music Conference*, Copenhagen, Denmark.

Mumma, G. 1967. Creative Aspects of live Electronic Music Technology. *Papers of 33rd National Convention*, New York. Audio Engineering Society.

Paine, G. 2002. Interactivity, Where to from Here? *Organised Sound* **7**(3): 295–304.

Rowe, R. 1992. *Interactive Music Systems: Machine Listening and Composing*. Cambridge, MA: MIT Press.

Schnell, N. and Battier M. 2002. Introducing Composed Instruments, Technical and Musicological Implications. *Proceedings of the 2002 Conference on New Instruments for Musical Expression*, Dublin, Ireland.

Schroeder, F. (ed.) 2006. *Contemporary Music Review. Special Issue: Bodily Instruments and Instrumental Bodies* **25**(1/2).

Sperber, D. 2007. Seedless Grapes: Nature and Culture. In E. Margolis and S. Laurence (eds.) *Creations of the Mind: Theories of Artefacts and Their Representation*. Oxford: Oxford University Press.

Waiswisz, M. 2008. Text and image archive for Michel Waisvisz. http://crackle.org/.

Waters, S. 2007. Performance Ecosystems: Ecological Approaches to Musical Interaction. *Proceedings of EMS: The 'Languages' of Electroacoustic Music (EMS07)*, Leicester, UK.

Wessel, D. 2006. An Enactive Approach to Computer Music Performance. In Y. Orlarey (ed.) *Le feedback dans la création musicale*. Lyon: Studio Gramme, 93–8.

Winkler, T. 2001. *Composing Interactive Music: Techniques and Ideas Using Max*. Cambridge, MA: MIT Press.