About This Issue

In its early years, *Computer Music Journal* occasionally published articles under the rubric of "Machine Tongues." That moniker referred to textual programming languages for music, a topic that also forms the thematic backbone of the current issue. We are pleased to publish here the first journal article on ChucK, one of the most successful recent languages for computer music. Introduced a little over a decade ago, ChucK has become popular especially for rapid prototyping, live coding (see *CMJ* 38:1), and pedagogy, as well as for designing software instruments used by performance ensembles of laptop computers or mobile devices. In this issue's first article, the language's creators, Ge Wang and Perry Cook, along with coauthor and ChucK developer Spencer Salazar, present an overview of the language's design and features. Noteworthy among ChucK's innovations are the keyword `now`, which allows the programmer to control the flow of logical time precisely and deterministically, and a time-based concurrent programming model, both leading to what the authors have termed a "strongly timed" language.

Next, Vesa Norilo introduces a newer language, Kronos. One of Kronos's primary goals is efficiency of signal processing. Another is to provide the programmer with a high-level, expressive medium. To reconcile these criteria, Kronos includes both a dataflow language for low-level building blocks and a metalanguage whose high-level code gets compiled into the dataflow language. Norilo reviews the classic unit-generator paradigm of sound synthesis, after which he briefly surveys some languages that go beyond the model of the unit-generator interpreter: CLM, Nyquist, SuperCollider, PWGLSynth, ChucK, Extempore/XTLang, and Faust. He then outlines Kronos's design considerations and offers a series of programming examples, culminating in a polyphonic synthesizer.

The third article on recent programming languages discusses the language LC, focusing on a particular family of sound-synthesis techniques that LC was explicitly designed to support. Hiroki Nishino, Naotoshi Osaka, and Ryohei Nakatsu show how their language can handle not only the traditional unit-generator paradigm but also microsound synthesis. Microsound synthesis techniques, including granular synthesis, pose challenges for the unit-generator approach. The authors explain how the concepts of abstraction barriers and abstraction inversion are relevant to these challenges. To help solve the problems of microsound synthesis, LC gives the programmer direct access to samples, and it borrows ChucK's `now` operator for control of logical time. A set of code examples illustrate LC's approach to microsound synthesis in comparison with some other languages.

The issue's fourth article, by Reginald Harrison et al., also treats sound synthesis, but not primarily from a programming-language perspective. This article presents a mathematical approach to the physical modeling of valved brass instruments. The authors use finite-difference time-domain methods, which permit the modeling of time-varying parameters such as mouth pressure, lip dynamics, and valve depressions. (This work constitutes part of the Next Generation Sound Synthesis project at the University of Edinburgh.) The article describes how the model was prototyped in MATLAB and then implemented in C for use either at the command line or through the Soundloom graphical user interface to the Composers Desktop Project software.

Evidence of ChucK's popularity can be discerned in the final article, which discusses an application that was written in ChucK. This is Adam Linson's Odessa, an interactive system for free improvisation. The system uses an artificial agent based on Rodney Brooks's "subsumption" architecture. From Brooks's architecture, which was originally created for building mobile robots, Odessa borrows the principles of computational simplicity, interactive behavioral layers, and the avoidance of internal models of the world (in Odessa's case, the world of musical improvisation). Thus Odessa is focused on the general interactive behavior of the "free" musical style in which it performs alongside human musicians, rather than on reasoning with music-theoretical constructs such as scales, harmonies, and meter. Its layered construction allows it to both adapt to and challenge the human musicians' playing, responding in different ways to pitch, loudness, and timing. After elucidating the system's architecture, the authors summarize the assessments of expert improvisors who interacted with Odessa.

The Reviews section covers a one-day conference on computer simulation of musical creativity as well as a CD by Australian composer

doi:10.1162/COMJ_e_00328

Front cover. ChucK code example and diagrams. (Courtesy of Ge Wang.) The large symbol at lower left is the "ChucK" operator, which permits the chaining of signal-processing modules in an intuitive left-to-right order.

Back cover. Illustrations from the article by Harrison et al.

CJ Symon. Following the Products of Interest section, the issue concludes with the Program Notes for the Journal's annual Sound and Video Anthology, whose media files can be found as "supplementary content" at http://www.mitpressjournals.org/toc/comj/39/4. Our thanks go to this anthology's curator, Marco Donnarumma, for selecting these exemplars of biophysical music and providing his insights.