A "Pick and Mix" approach to the integration of Composition Tools

Barry Eaglesone
University of Bradford
Bradford, UK
B.Eaglesone@comp.brad.ac.uk

Peter Lundin
Royal Institute of Technology
Stockholm, Sweden
ludde@faor.kth.se

Abstract

TEMA is a generic environment for composition. It has been designed to host collections of software tools and environments and to provide generic services for assisting the composer with the management of resources, products and processes. This paper overviews the methods used to integrate composition tools into the TEMA environment.

1 INTRODUCTION

The experimental and individualistic nature of computer music drives many composers to be scavengers of composition tools and techniques, from both amateur and professional sources, and also experimenters with new and modified software. Composers will often design both the art work and also the tools and techniques by which it is created. This makes computer music composition unlike other design tasks, such as in engineering, where professional quality tools are utilized within the bounds of "good practice" rules and methods.

Our response to this problem is TEMA, an "open" support environment which has been designed to assist composers to collect together, utilise and experiment with sets of composition tools taken from different sources. TEMA is based on support environments enjoyed by engineers, but adapted for artistic use [E94]. This paper overviews TEMA's support for integrating composition tools (see [E95] for a fuller description), and is complemented by [EL96] which gives an overview of the architecture and object model of TEMA and implementation details.

2. COMPOSITION TOOLS

The integration of composition tools is complicated by their diversity, ambiguous rules, and volatility.

Diversity: The experimental and idiosyncratic nature of composition has led to a proliferation of tools which are more diverse and variable in quality than in the case is other design and engineering disciplines. The situation is complicated by the continued use of the older, stable and well established, but low-level tools (such as Csound). These small-scale "legacy systems" are sometimes preferred over innovative technology-driven systems because of the short life span of "new-technology-based" systems and a desire for longevity of the composition.

Ambiguity: Tools may often have a multiple role. For example, in real-time stochastic compositions, a composition tool for generating wave forms may also become part of the composition itself.

Volatility: Composers may program new or modified tools to overcome limitations of existing ones, such as their poor quality, incompatibility or functional limitations, or simply in order to try out new ideas.

The two conclusions drawn from the above observations are: (i) the usual segregation between design objects and tools is inappropriate in a composition support environment; and (ii) integration of "foreign" and "legacy" tools is a primary requirement.

3. TOOLS INTEGRATION METHODOLOGY

TEMA has a (core or test) conventional design system architecture, centred on an object-oriented database tool [EL96], the repository, within which tools and design object are stored. Tools can communicate with each other, and access design objects via a public tools interfaces, which acts as the integration agent.

TEMA supports three ways of integrating composition tools - tools may be encapsulated within the repository, implemented as applications using the repository object model to simulate the tools environment; or may
operate "along side" the TEMA system with a common process maintaining a binding between the objects processed by the tool and representations of those objects in the repository. In addition, the ambiguous, volatility and variable quality of tools is accommodated by representing the tools themselves within the repository as versioned design objects. This is to support re-programming and experimenting with tools.

Encapsulation: Encapsulation, whereby the tools become methods of the data objects upon which they operate, has long been seen as the natural means of tools integration in an object-oriented environment. It is supported by TEMA, but in a weaker form, in which data and the tools that operate upon it form complex (composite) objects. The advantage is that both tools and data can be versioned objects and can be manipulated independently, thus allowing experimentation with both data and tools. This has provided a useful forward means of integrating functions taken from the CARL, computer music library with the objects that represent sounds. We have also integrated Cosound with objects representing Cosound instruments, orchestras, score events and scores [E95]. An advantage has been the ability to implement a higher level graphical interface for Cosound programming, which shields the composer from low-level detail (e.g., score events are created by "playing" instruments, instead of coding them).

Environment simulation: This method integrates a "foreign" tool by simulating the tool's environment within the TEMA object model. In this way mismatches between the tool's environment and that of TEMA are reconciled. For example, file processing tools can be integrated via TEMA objects which simulate files. Greater benefit is achieved by simulating the information structure of the data manipulated by the tool, since this makes possible fine-grain application of the TEMA services (for example, allowing individual design objects to be versioned, rather than files of design objects). This we have done to integrate Peter Lundt's CCPL composition system [L94]. The added value for CCPL includes the TEMA services for managing the CCPL information structures (units, spaces, ...) at persistent versioned objects, and the ability to use CCPL in conjunction with other tools.

Environment binding: This third method is weaker than the previous ones, but has the advantage of allowing a tool to operate within its own environment, and thus the need to make any alterations to the tool. This method works as follows: The tool and the data objects upon which it operates are both represented within TEMA's repository, but the contents of the data objects are stored externally within the tool's environment. When the tool is executed, TEMA also executes an integration agent which ensures integrity of the binding between the representation of the data objects in the repository, and their contents within the external environment. An advantage is that various TEMA object management services can be used concurrently with the execution of the tools. This method of integration has been used to integrate standard editing tool with other composition tools integrated using the other two methods.

5 DISCUSSION AND CONCLUSIONS

The integration methods described provide a means by which composition tools can be integrated within a support environment, enabling composers to use them in combination, and also to use the support services provided by the environment. We believe that our "open" environment approach can liberate composers, by assisting their with the inevitable object and process management problems which are a by-product of computing in the individual ways in which they wish to use the tools they wish to exploit. As our research progresses, an overriding requirement will remain that of avoiding prescriptive element which may restrict creativity by forcing composers into using "good practice" tools and methods. This requirement distinguished artistic from engineering support systems.

REFERENCES


