

Beyond open source music software: extending open source philosophy to the music with CODES

**Evandro Manara MILETTO,
Luciano Vargas FLORES,
Daniel Eugênio KUCK and
Marcelo Soares PIMENTA**

Instituto de Informática
Universidade Federal do Rio Grande do Sul
Caixa Postal: 15064
91501-970 Porto Alegre, Brazil,
{miletto,lvflores,dekuck,mpimenta}@inf.ufrgs.br

Jérôme RUTILY

Institut Nat. Polytechnique de Grenoble
Domaine Universitaire
38402 St Martin D'Herès, France,
jerome.rutily@ensimag.imag.fr

Abstract

This paper presents CODES – COoperative Music Prototype DESign – a Web-based cooperative music composition environment. This means it allows any person to connect with other users, through the Web, and cooperate with them to draft simple musical pieces, in a prototyping way. Besides describing our main design decisions and overall implementation solutions, we will also briefly discuss our belief that CODES is not just open source music software, but extends the open source notion as it reuses publicly available code (frameworks) and tools, and allows open music production, in a way as collective as open source development.

Keywords

Music prototyping, cooperation, WWW, open source, free music

1 Introduction

Music technology has undergone considerable changes over the last decades, mainly with an increasing use of the Internet. An outcome of these changes is *networked music* [1] [2], which allows experimental artists to explore the implications of interconnecting their computers. Indeed, lately, Internet-based networked music has attracted a wider interest from the music technology community, and the existing applications have evolved towards more sophisticated projects and concepts including, for example, real-time distance performance systems, and various systems for multi-user interaction and collaboration.

Cooperative music composition environments are such a case, which early became an interest to our computer music research group [3]. The CODES project associates Computer Music with concepts from the fields of Human-Computer

Interaction (HCI) and Computer Supported Cooperative Work (CSCW), with the intent to design interaction so that the system can be useful and usable even to non-musicians. Besides musicians, novices are also probably interested in creating music and participating in musical experiments, but they lack environments oriented to their profile. On the other hand, musicians have not yet developed the tradition of sharing their musical ideas and collaborating while composing, and so they would equally benefit from the experience in a cooperative environment.

CODES – COoperative Music Prototype DESign – is similar to other Internet-based systems for music composition, such as those in a survey by Weinberg [4]. They also enable users to contribute their own material and to manipulate (listening, altering, refining, etc.) others' contribution, usually through asynchronous interaction and off-line material manipulation. But in addition to common characteristics existing in other systems, CODES addresses several other important aspects to be considered in a collaborative environment for music composition/prototyping, such as group awareness, different levels of interaction, capability to export/import alternative sound formats, and support for music prototyping rationale and for long prototyping sessions. With this set of additional features, one can track back some part of the music prototype and understand the intention and the process followed by its author to compose it, as an alternative way of learning by example.

Also, since we want CODES to be used by anyone, it should not rely on traditional music notation, nor should demand knowledge of music theory for its users to prototype music. So, we developed mechanisms to represent sound patterns as icons, and the option to intelligently suggest them to the user, or to offer him an easier access to

those patterns which could fit well in his music prototype.

Attributes such as these reflect the free/libre/open-source philosophy we follow in this project. This does not apply only to the software being open-source, but extends throughout the overall requirements of the project, such as for free access, availability, accessibility to non-musicians, portability, reuse of code, collective development approach, support to a collective/cooperative composition approach based on experimentation, etc. Moreover, music is made within CODES in the same collective/cooperative/prototyping way as in open source development. This kind of music will be the result of teamwork, will be a collective product, will be available on the Web and may be always open to further modifications. So, it cannot be dealt with according to the usual authoring laws of the traditional music market (for instance, we must find alternative licensing mechanisms that suit more in this case). It must be seen and treated by us as what some communities are already calling “free music” or “open music” [5] [6], and thus CODES extends the open source notion all the way to the music itself.

This paper is organized as follows. Our main design decisions for the CODES project are presented in section 2. Section 3 describes brief details on the architecture and implementation of the CODES environment. Some aspects of the CODES user interface and of how it is used are presented in section 4. Section 5 discusses briefly why we believe that CODES extends the open source notion beyond its source code development. Finally, section 6 presents some conclusions and future goals.

2 The CODES project: design decisions

The major motivation underlying our proposal is allowing non-musicians to access a virtual space in order to interact with each other, explore sounds together, discuss about this exploration, and retrieve all the discussed information anytime they want. Therefore, we soon discussed some aspects that could be essential to consider in the design of such an environment, which we still use to guide the development of CODES:

- It must run in a *virtual space only*, via Web browser, to ensure that the barrier of geographic distance among partners (physical presence) does not become an issue.
- It must *run on a great diversity of platforms* and browsers (at least, all W3C compliant [7]), minimizing requirements of use and thus increasing accessibility.
- It should allow *independence of a tutor*, and

non-structured groups, despite the possibility of supporting structured groups with a tutor role, what is usually necessary in learning situations.

- Due to the exploratory nature of how it is used, a very important characteristic should be the users’ possibility to perceive and analyze group members’ actions on music prototypes, and to know the reasons behind each one of these actions. These are aspects related respectively to *awareness* and to *prototyping rationale*, for which CODES then must provide support. The concept of “awareness”, from the CSCW literature, cannot be precisely and uniquely defined [8]. In the context of CODES, the adopted notion of awareness is “the understanding of the actions of other users, what provides for a user a context for his own actions”. Prototyping rationale is a mechanism which allows each user to justify his actions, in order to make clear to the others the idea or reasons that guided him to make that decision. As a result, collaborators in a prototype will have access to an explicitly recorded track of all steps that led to the current prototype state.
- It must support *long prototyping sessions*: an important mechanism in any design activity is the ability to interrupt the session and to resume it in order to continue the process from the last break point. A music prototyping session can take many days or even weeks before a final result is reached.
- It should offer *alternative music encoding formats*, making it easy for users to export/import their music between different systems, thus integrating CODES into a wider context of music systems. Standard MIDI was chosen here due to its easy manipulation and compatibility. Although the sounds of synthesized MIDI files played on most PCs are still low quality, it yields some future possibilities, like the conversion from MIDI to conventional music notation. We are investigating the use of some mark-up languages for music – like MusicXML [9], Music Mark-up Language (MML) [10], and the Music Encoding Initiative (MEI) [11] – as interesting alternatives to be explored. We believe that in a near future one of them (or some variation thereof) will be the standardized format of choice for music content on the Web.

As we can see, several design decisions for the CODES project were targeted on allowing more freedom and “openness”. We will discuss that

further on section 5. Next we will present briefly some system implementation details.

3 Architecture and implementation of the CODES environment

CODES implementation follows an open source philosophy, aiming as well at providing easy access to its software development and supporting tools. We went after various publicly available software frameworks and design patterns, to reuse well-known solutions. As a consequence, we chose to build CODES respecting the classic Model-View-Controller (MVC) architectural model (see Figure 1). On the server side, it is implemented through the *WebWork framework* (Java) [12]. As Web server, CODES uses the Apache Tomcat Servlet Container [13], a reference for Java applications. Data persistence is done following the DAO design pattern for data access, implemented by the *Hibernate framework* [14], which stores data in a MySQL database [15]. The system is organized in a few modules (such as the Prototype Manager, the Users Manager, and the Cooperation Manager), and to coordinate them, CODES implements a Façade design pattern, that is an interface which provides modularity. Finally, for the user interface on the client side we are using AJAX (Asynchronous JavaScript and XML), which is supported by this architecture, and we implement it with the *Dojo JavaScript framework* [16].

The Java Sound API allows us to focus system development on both graphical user interface (GUI) and cooperation aspects, making the sound handling part easier because of its components that already offer sound control.

4 The CODES user interface

The user interface was designed to meet requirements related to interaction flexibility, robustness, and easiness of use, as well as to

present adequate support when complex musical information is displayed, thus providing an effective interaction between users and the environment. We wanted to reach a balance between user interfaces that are so “easy” for the user that they end up depleting his expressiveness, and others that are so complicated that they discourage beginners.

In music, some peculiarities make the creation and conception processes different from those carried out in other fields. Musical composition is a complex activity where there is no general agreement about what activities have to be done and in which sequence: each person has his own style and way of working. So, the process of music composition is difficult to understand and, therefore, also to learn.

“Prototype” is not a common expression in music literature. In fact, a “composition” is known to be the result of a composer’s creative activity. But the emphasis of our work is mainly on the *process* (prototyping), and not on the product itself. The repetitive cooperation cycle in CODES, where online partners refine a musical sketch until its final form is reached, clearly resembles the incremental prototyping cycle adopted in industry, and thus we call its result a *music prototype*.

Music is like an “artistic product”, which can be designed through prototyping. A musical idea (notes, chord sequences, rhythms, etc.) is created by someone (typically for a musical instrument), and afterwards cyclically and successively modified and refined according to his initial intention or to ideas that come up during the prototyping process.

We believe *the experience of this prototyping process* is what’s most interesting in using CODES. During such an experience, lots of knowledge sharing will take place, by means of the rich interaction and argumentation mechanisms associated, in this environment, to each prototype modification. So, *the process itself* will foster all

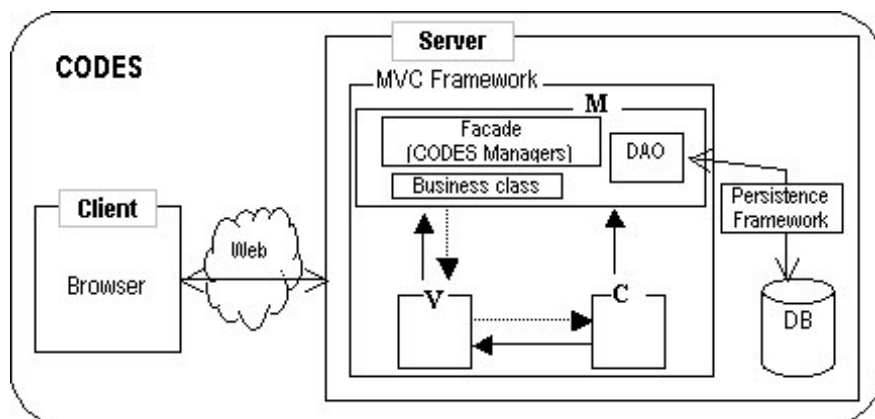


Figure 1: Architecture of the CODES environment

the participants' learning about music and music making, and that is exactly our main concern. It is not important to us, at the moment, the quality of the product (musical piece) that results from music prototyping with CODES. We are now more interested in enhancing the experience of music making, so that non-musicians may also have access to it, and we may get a better understanding of how this process works.

In CODES, a musical prototype consists of *Lines* (of instruments, arrangements, effects, etc. – see Figure 2) that can be edited. Editing is typically done by selecting sound patterns among many of the predefined patterns made available in CODES. Sound patterns are high-level musical structures (small sections of music files in MIDI format), which then make the processes of choosing sounds and prototyping easier. It will be made possible, in the near future of the project, for the user to edit the sound patterns, as a deeper level of composition.

A user can create more than one line, that is, someone can be the “owner” of more than one line (like user *Jerome* in Figure 2). By clicking the “Play” (>) button, the Sound Manipulation Manager starts the execution of all the lines enabled for playback (option *Mute* unselected). All patterns of the chosen lines, vertically grouped in

the same timeline, are mixed and played, under complete user control, which can stop and restart at any time with usual control buttons (*Play, Stop, Forward, Rewind, Pause*).

User interaction, therefore, basically includes actions such as selecting sonic patterns, dragging and dropping them into lines (what is allowed through AJAX coding) and playing them, and combining them with other lines composed by his “partners” (other users) in the same music prototype. This combination can occur in different ways: overlapping (simultaneous playing), juxtaposition (sequencing), etc.

Cooperative music prototyping is herein defined as an activity that involves people working together in a musical prototype. Cooperation in CODES is asynchronous, since it is not yet necessary to manage the complexity of real-time events, if the present goal is just to support the development of musical prototypes. Users can access the prototype, do their experiments and write comments at different times.

In CODES, a musical prototype is initiated by someone, the “prototype owner”. The prototype owner uses CODES to elaborate an initial musical prototype, and to ask the collaboration of other partners by sending explicit invitations (typically using an e-mail form from inside the system).

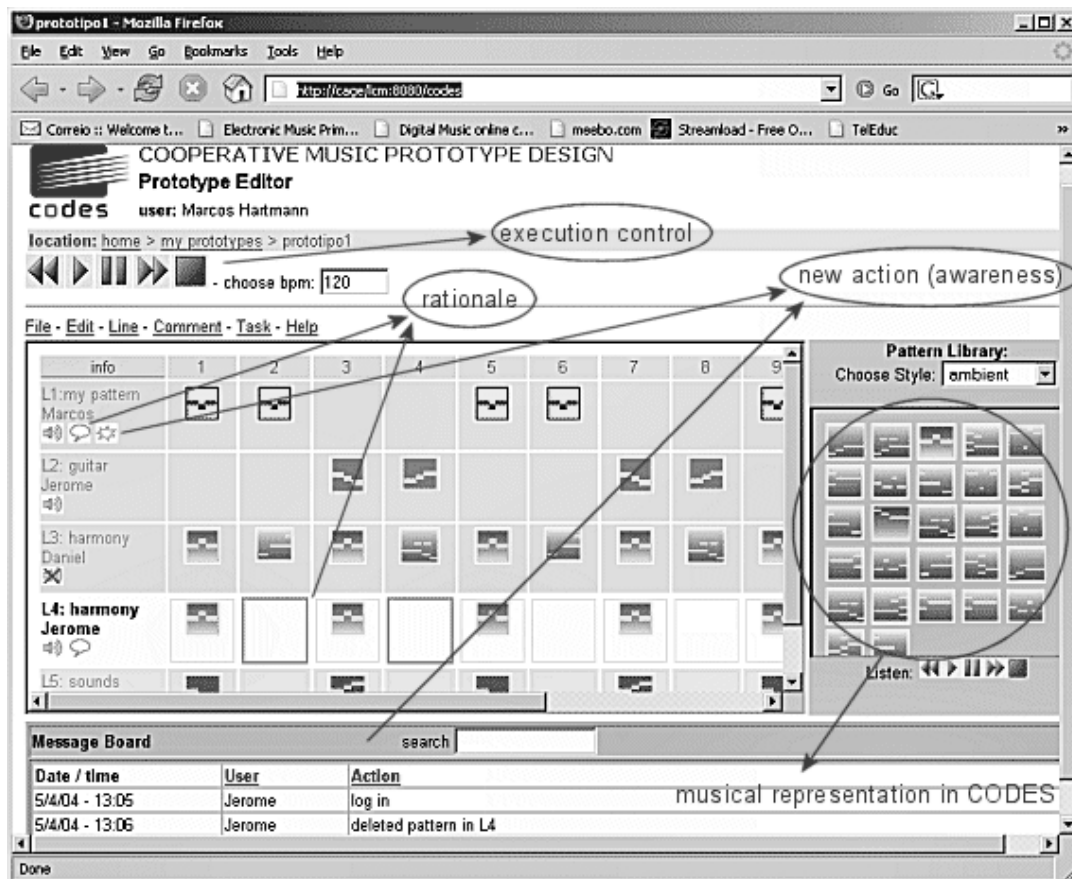


Figure 2: Elements of the CODES editing window

Partners who accept the invitation can participate in the collaborative musical manipulation and refinement of the prototype. The owner can also leave his prototype in an “open” space, in which interested users would discover it and then join the collaboration, as new partners. This way, the group of partners may evolve into a virtual community and, therefore, CODES may be classified as *communityware* [8].

For better support to cooperative musical activities, we propose in CODES three kinds of awareness mechanisms (see Figure 2) [17]:

1. *Music Prototyping Rationale*: to allow users to link their explanations with their actions on music prototypes;
2. *Action Logging*: to keep an explicitly recorded track of the steps that led to the current prototype state; and
3. *Modification Marks*: to indicate to a user that a prototype has been modified by others.

Actually, awareness mechanisms offer several advantages to music prototyping:

- keeping track of decisions;
- tracking progress in music prototyping and identifying conflicts, which may initiate negotiation processes between multiple points of view;
- supporting the construction of cumulative prototyping knowledge;
- assisting the integration of perspectives from multiple members of a group;
- “understanding” of each prototype, as there is no single answer or solution to a music prototyping problem.

CODES uses *icons* to represent musical information, as an alternative to the conventional music notation (score). Icons allow users to rely on both audio and visual clues more than on music theory to choose the right sound patterns. The idea here is to favor experimentation rather than theoretical knowledge. Each pattern can be individually listened to, before being selected and incorporated into a line on the prototype. Each icon traces its own sound pattern in a sort of “Cartesian plane”, where the horizontal direction means duration of notes and the vertical means pitch variation. See an example in Figure 3, which shows the CODES notation and the correspondent musical staff.

This loose representation of pitch and duration is inspired in the early “piano roll” metaphor, largely used in music editing software. This way the user has a visual feedback of the sound even without listening to it in beforehand. In fact, we believe no previous musical knowledge should be required from any user to create music prototypes using CODES. The user does not need to know

conventional music notation to create prototypes: he may select, play and combine such patterns in an interactive way, by direct manipulation and experimentation, without taking into account the formal representation format. However, the capability to convert from musical prototype to the score version is one of our next goals, in order to better support pedagogical uses and further music theory learning possibilities.

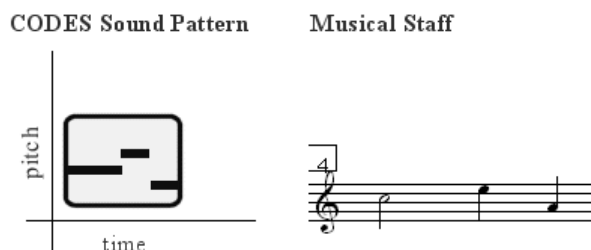


Figure 3: Musical representation in CODES

5 Beyond the open source music software

CODES is open-source from the beginning, since it is a scientific/academic research, conducted at a public university, and under public funding. But we want our project to take this quality further beyond the mere software that is being developed. As we saw in sections 1 and 2, requirements and design decisions such as reuse of code, free access and availability even to non-musicians, cooperative music prototyping approach, and alternative music encoding formats, all show that the CODES project is being entirely conducted under a free/libre/open-source philosophy. In other words, the system is being built to be itself a tool for open content production.

Sure there will be certain parts in such a project where achieving the desired freedom/openness will not be entirely possible. But we believe that just taking this open source notion as a principle is already important.

Going further with the free philosophy of our research, we are also studying *alternative licensing* options for all the intellectual products of this project: the software that forms the system; the sound patterns available in the system; and even for all music that will result from using CODES. These more flexible intellectual work licensing options are vital in the current global “information age” context where, according to Esther Dyson, “the Net dramatically changes the economics of content” [18]. Traditional copyright or intellectual property laws cease to be applicable, because they are too limiting, or they simply do not manage to survive.

Interesting alternatives may be those of the

Creative Commons initiative [19], and for the software, the CC-GNU-GPL, which is being suggested by government efforts in Brazil [20]. As far as it concerns the users, it will be made clear why they should “open their work to the world”, at least to agree with the philosophy of the project. This discussion will be an important part of the CODES project website, contributing to increase general consciousness about the “free/libre/open-source work” issue, which is becoming more and more important in present times.

6 Conclusion

CODES software is still under development, but we have already conducted a number of small scale studies with a partially functional prototype, in order to evaluate its use, identify and correct problems, and determine new requirements. The most interesting result of this preliminary assessment was that, in addition to the “conventional” edit-listen-publish-cooperate-refine procedure for which we have initially developed CODES functions, users were able to find other creative applications for it:

- as an effective support for music learning;
- as an entertainment tool (DJ-like performance/experiments);
- as accompaniment system for human live performance.

This flexibility to accept new uses, together with several features to support the use by non-musicians, indicate both that we are applying a free/libre/open-source philosophy in all facets of the whole CODES project (not just in open-sourcing the software code), and that this notion can indeed be extended to other fields besides source code development. This discussion was our aim with this paper, along with presenting briefly the architecture, implementation and user interface details of the CODES environment.

We have shown throughout this paper that the open source notion can even be extended to the very main purpose of a music software, and equally to the music that will result from its use. In this case, developers must care to treat such music as “free music”, or open content, and to take the proper measures to make it practicable (e.g. dealing with alternative licensing options).

By allowing non-musicians to have access to a musical creative experience, we are again practicing a free philosophy, since we are “freeing” users from the need to know music theory and to know how to play a musical instrument. As in open-source software philosophy, where source code is available to the public, we believe music should not be held exclusively by those who know the “secrets” (theory) of how to make it.

We also want to make clear that, despite focusing on non-musicians, we are not discarding the use of CODES by actual musicians too. In fact, it will be even of interest to assess, in the near future, what kind of uses that experienced musicians may find for this cooperative music prototyping environment.

Another aspect remaining to be explored is the implementation of real-time interaction in the CODES architecture. Interaction among users in a synchronous fashion may bring up new features and properties not considered until now.

7 Acknowledgements

This project is being partially supported by the Brazilian research funding councils CNPq and CAPES.

References

- [1] A. Barbosa. Displaced soundscapes: a survey of network systems for music and sonic art creation. *Leonardo Music Journal* 13: 53–60, 2003. MIT Press, Cambridge, Massachusetts.
- [2] *Organised Sound*, 10(3) [issue on Networked Music]. Cambridge University Press, Cambridge, UK, Dec. 2005.
- [3] UFRGS Computer Music Research Group. *LCM – Computer Music Lab*. <http://www.inf.ufrgs.br/lcm/>, accessed in Dec. 2006.
- [4] G. Weinberg. The aesthetics, history, and future challenges of interconnected music networks. In *Proceedings of the International Computer Music Conference* (Göteborg, Sweden, 2002). ICMA, 2002.
- [5] R. Samudrala. *The Free Music Philosophy*. 1998. <http://www.ram.org/ramblings/philosophy/fmp.html>, accessed in Dec. 2006.
- [6] *Free the sounds... and you free the music*. <http://freethesounds.org/>, accessed in Dec. 2006.
- [7] W3C. *World Wide Web Consortium*. <http://www.w3.org/>, accessed in Dec. 2006.
- [8] O. Liechti. Awareness and the WWW: an overview. *ACM SIGGROUP Bulletin* 21(3): 3–12, Dec. 2000.
- [9] M. Good. MusicXML: an Internet-friendly format for sheet music. In *Proceedings of the XML Conference and Exposition* (Orlando, 2001).
- [10] J. Steyn. *Music Markup Language*. <http://www.musicmarkup.info/>, accessed in Dec. 2006.

- [11] P. Roland. The Music Encoding Initiative (MEI). In *Proceedings of the International Conference on Music Applications using XML* (Milan, Italy, 2002).
- [12] OpenSymphony. *WebWork*. <http://www.opensymphony.com/webwork/>, accessed in Dec. 2006.
- [13] The Apache Software Foundation. *Apache TomCat*. <http://tomcat.apache.org/>, accessed in Dec. 2006.
- [14] Red Hat. *Hibernate*. <http://www.hibernate.org/>, accessed in Dec. 2006.
- [15] MySQL AB. *MySQL: The world's most popular open source database*. <http://www.mysql.com/>, accessed in Dec. 2006.
- [16] Dojo Foundation. *Dojo, the Javascript toolkit*. <http://dojotoolkit.org/>, accessed in Dec. 2006.
- [17] E. M. Miletto et al. CODES: supporting awareness in a Web-based environment for collective music prototyping. In *Proceedings of the Brazilian Symposium on Human Factors in Computer Systems, IHC 2006* (Natal, Brazil, Nov. 2006).
- [18] E. Dyson. Intellectual value. *Wired* 3(7): 136–141 and 181–185, Jul. 1995. <http://www.wired.com/wired/archive/3.07/dyson.html>, accessed in Dec. 2006.
- [19] *Creative Commons*. <http://creativecommons.org/>, accessed in Dec. 2006.
- [20] Technical Committee for the Implementation of Free Software in the Federal Government. *Portal – Free software licenses*. <http://www.softwarelivre.gov.br/Licencas/>, accessed in Dec. 2006. (In Portuguese.)