A User-Friendly Graphical System for Room Acoustics Measurement and Analysis[*]

Leo Kazuhiro Ueda[1], **Fábio Leão Figueiredo**[2], **Fernando Iazzetta**[2], **Fabio Kon**[1]

[1]Department of Computer Science – University of São Paulo

[2]Department of Music – University of São Paulo

`lku@ime.usp.br, fabioflf@hotmail.com, iazzetta@usp.br, kon@ime.usp.br`
`http://gsd.ime.usp.br/acmus`

Abstract. *AcMus is an integrated software platform for musical room acoustics. A preliminary version of the software was developed as a MATLAB prototype of acoustical analysis functions. The prototype processes measurements recorded from a room and outputs calculations that help us assessing the quality of the room for musical performance. We have measured a number of musical rooms ourselves and are currently analyzing the data. Meanwhile, we are working on the final implementation of the AcMus integrated platform, which is based on a flexible and extensible Java plug-in framework. This new, open source version of the system incorporates our experiences with the preliminary prototype and tries to maximize the usability and effectiveness of the user interface.*

1. Introduction

In 2002, we started the AcMus project [Iazzetta et al., 2004, Yili et al., 2003], an effort to consolidate a research community in São Paulo, Brazil, devoted to issues related to musical room acoustics. One of the main goals is to build an open-source extensible software for estimation, measurement, analysis, and simulation of rooms especially designed for musical performance. The software is divided in three modules.

1. **Measurement Module**: helps the user to perform and analyze acoustical measurements.
2. **Utilities Module**: offers tools that can be useful for the design of rooms, acoustical measurements, and audio processing.
3. **Simulation and Optimization Module**: helps the design of rooms by performing computer simulations and applying optimization techniques.

This paper focuses on the results obtained so far concerning the Measurement Module. We first built a MATLAB prototype [Masiero, 2004], which we describe in Section 2.1, that allowed us to analyze measurements taken from various concert halls located in the city of São Paulo. This data has been used to carry out an investigation on how to correlate objective and subjective acoustic parameters [Figueiredo et al., 2004], we talk about this in Section 2.

The prototype consists of a number of MATLAB DSP functions with no special user interface other than the MATLAB's own graphical development environment. During the work on the field, we also used some general-purpose audio commercial tools. With this experience, we confirmed the necessity of an efficient, integrated, and easy-to-use computer system that would help us take and analyze the measurements. So, in

[*]Supported by FAPESP, Brazil, proc. numbers 02/02678-0, 04/05396-0, and 02/10660-3.

parallel to this work, we are applying what we learned from this practical experience to build a graphical user interface for the Measurement Module in Java. We have described the ongoing work of this interface in a previous paper [Ueda et al., 2005], so in Section 4 we show how it will be useful.

2. Measurement Module Prototype

With our MATLAB prototype, we are able to process the data from the rooms we measured. This analysis gives us acoustical parameters that tells us something about the quality of the room. Although many acoustical parameters may be determined by specific measurements and calculations, our interest lays in those that shows correlation with sensitivity and perception of subjects in a certain environment, particularly in rooms designed for musical execution and listening.

Such parameters play a crucial role in the artistic quality of a musical event. Each of the subjective parameters [Beranek, 2004] (Liveness, Clarity, Definition, Spatial Impression, among others) is related to specific physical phenomena that are responsible for the acoustical impressions that define the characteristics of a music room. The most important parameters are represented by mathematical expressions that generate objective values, so we may see them as measurable physical quantities [Beranek, 2004]. For example, RT60 for reverberation, C80 for clarity, and BR for bass ratio.

The measurement process consists of playing specific generated signals inside of the room and then record the room response to this signal in different positions. The recorded signal is mathematically compared with the original generated one, from which we obtain the room impulse response (IR) [Kuttruff, 1991]. Figure 1 shows two impulse responses obtained from the main floor of the *Teatro Municipal de São Paulo*.

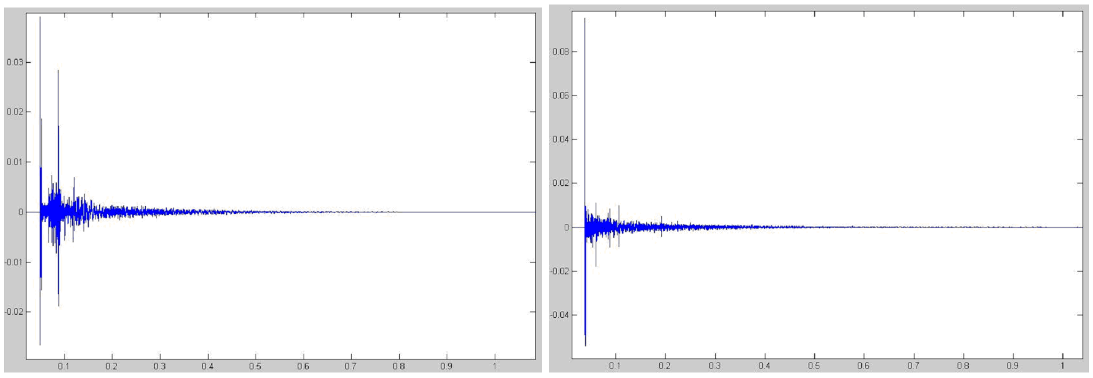


Figure 1: Impulse responses from the *Teatro Municipal de São Paulo*

The most efficient method to calculate the impulse response is the one that uses a sine sweep. The sine sweep is a sinusoid in which its instant frequency varies in time. This variation may be linear or logarithmic. We chose to use the logarithmic sweep, which has a pink spectrum, that is, its amplitude decays 3dB per octave. This means that each octave of the signal contains the same amount of energy. The frequency doubles at a fixed rate. The impulse response is obtained by deconvolving [Müller and Massarani, 2001] the generated signal with the recorded one.

From the impulse response, the functions we implemented give us the desired parameters. We calculate the energy decay curve from the impulse response by using the Schroeder Integration method [ISO 3382, 1997]:

$$E(t) = \int_t^{\infty} p^2(\tau)d\tau = \int_0^{\infty} p^2(\tau)d\tau - \int_0^{t} p^2(\tau)d\tau \tag{1}$$

where p is the impulse response. By manipulating this integration, we can calculate the main acoustical parameters. Figure 2 shows a graphical output example of this manipulation. The measurement was taken from the *Teatro São Pedro*.

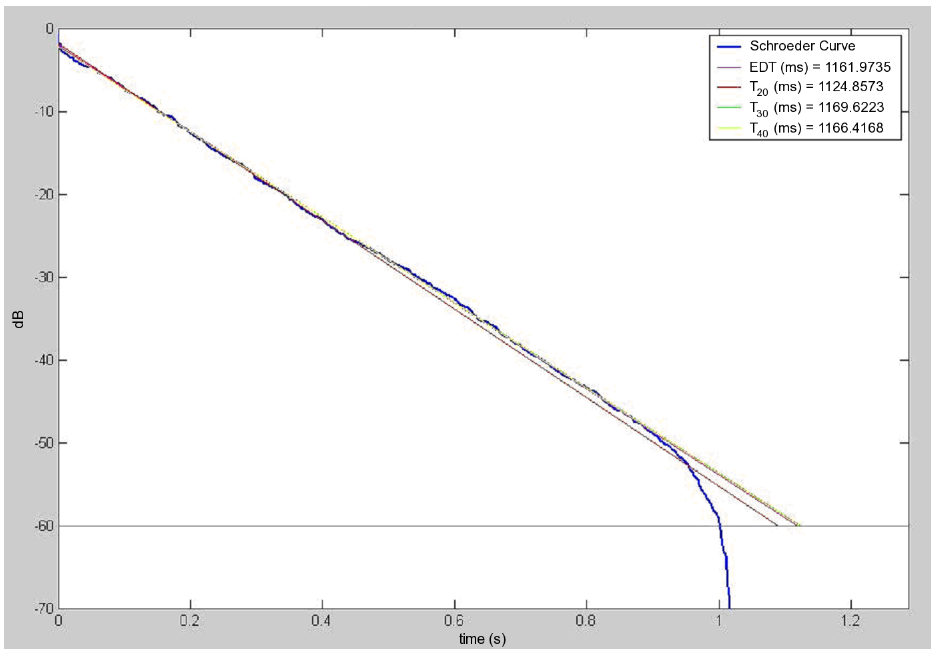


Figure 2: Decay curve (*Teatro São Pedro*)

We measured and compared some concert rooms in São Paulo where stable symphonic groups perform regularly. Figure 3 shows some of the parameters obtained from the *Anfiteatro Camargo Guarnieri* (CG), *Teatro de Diadema* (TD), *Teatro do Memorial da América Latina* (ME), *Teatro Municipal* (TM), *Teatro São Pedro* (SP), and *Teatro Sérgio Cardoso* (SC).

The theaters showed quite distinct acoustical behaviors due to different architectural characteristics and the acoustical treatment. Although we are still processing the measurement data, preliminary analysis leads to important conclusions about the acoustical performance of the rooms.

For instance, we observed that the *Teatro do Memorial da América Latina* has too much absorption material on the lateral surfaces. This resulted in lower values of the Brilliance and Lateral Fraction parameters. We detected acoustical distortions (echo and excessive bass) at the back of the main floor. The analysis of the impulse response suggests that this distortions are due to the peculiar shape of the ceiling: a parabolic curve that reflects and confines the sound waves in the distortion areas.

The *Teatro Sérgio Cardoso* has a very large scenic space, resulting in relatively long reverberation times. This is certainly a negative aspect for the orchestra performance.

The parameters for the *Teatro Municipal* indicate that it is suitable for operistical presentations. Its various possible positions for the audience (main floor, balcony, gallery) provides diversified listening conditions. The measurements show the stage is excessively dry, confirming the opinion of the musicians we interviewed. The superior floors present higher Clarity levels than the main floor. However, in the gallery we observe excessive diffusion and low Clarity, this may be a problem for certain repertoires.

Considering now the Brilliance (TR) and Bass Ratio (BR) parameters, the *Anfiteatro Camargo Guarnieri* and the *Teatro São Pedro* are more balanced than the others. The *Teatro de Diadema* and the *Teatro Sérgio Cardoso*, on the other hand, presented ex-

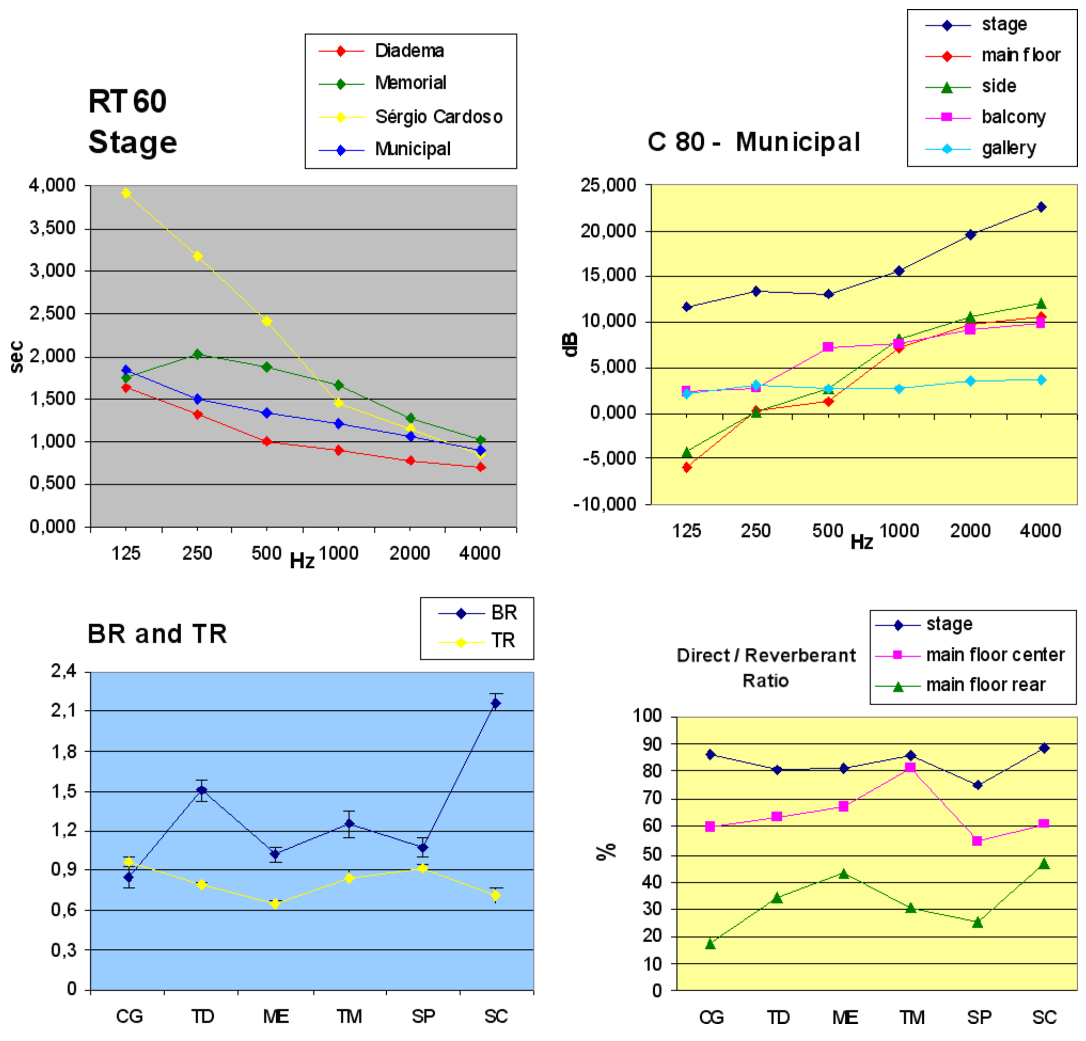


Figure 3: Some acoustical parameters of rooms in São Paulo

cessive Bass Ratio. A possible explanation for this behavior in the *Teatro de Diadema* is that it has wide openings at the stage sides, making an additional space in the area where the sound is produced. The *Teatro Sérgio Cardoso* has a huge reverberant chamber behind the orchestra position and the panels placed on the stage are not efficient enough to solve this problem.

When we finish processing all the data, we will be able to diagnose precisely the acoustical behavior of the rooms and to suggest possible solutions to their acoustical problems.

2.1. MATLAB Implementation

MATLAB (`http://www.mathworks.com/products/matlab`) is a commercial package that includes a high-level interpreted programming language and a development environment. Unlike the operators in the more common programming languages, which usually only deal with scalars, MATLAB built-in functions and operators let the programmer work with matrices and complex numbers in a quick and easy way. Along with its development environment and its comprehensive DSP and math libraries, this makes it a good prototyping language, suitable for our needs to test and validate the audio processing algorithms we studied.

The prototype we implemented is a set of MATLAB functions that provides most of the analysis functionality of the Measurement Module for two methods: Maximum-Length Sequence (MLS) and Log Sweep FFT (LSF). Let us take a look at them.

Order	Length	Class	Tap
2	3	a	2, 1
3	7	a	3, 1
		b	3, 2
4	15	a	4, 1
		b	4, 3
5	31	a	5, 2
		b	5, 3
6	63	a	6, 1
		b	6, 5
7	127	a	7, 1
		b	7, 6
8	255	a	8, 6, 5, 1
		b	8, 5, 3, 2
9	511	a	9, 4
		b	9, 5
10	1023	a	10, 3
		b	10, 7
11	2047	a	11, 2
		b	11, 9
12	4095	a	12, 7, 4, 3
		b	12, 11, 8, 6
		c	12, 11, 10, 2
13	8191	a	13, 4, 3, 1
		b	13, 12, 10, 9
14	16383	a	14, 12, 11, 1
		b	14, 13, 8, 4
		c	14, 13, 12, 2
15	32767	a	15, 1
		b	15, 14
15	32767	c	15, 11
		d	15, 8
16	65535	a	16, 5, 3, 2
		b	16, 15, 13, 4
17	131071	a	17, 3
		b	17, 14
		c	17, 14, 13, 9
18	262143	a	18, 7
		b	18, 11
19	524287	a	19, 6, 5, 1
		b	19, 18, 17, 14
20	1048575	a	20, 3
		b	20, 17
21	2097151	a	21, 2
		b	21, 19
22	4194303	a	22, 1
		b	22, 21
23	8388607	a	23, 5
		b	23, 18
24	16777215	a	24, 4, 3, 1
		b	24, 23, 22, 17
25	33554431	a	25, 3
26	67108863	a	26, 8, 7, 1
27	134217727	a	27, 8, 7, 1
28	268435455	a	28, 3
29	536870911	a	29, 2
30	1073741823	a	30, 16, 15, 1
31	2147483647	a	31, 3
32	4294967295	a	32, 28, 27, 1

Table 1: Feedback taps for the MLSs of order 2 to 32

Signal generation

In order to perform a measurement, we first need to generate the input signal to be played inside the room. The user should use the right function for the chosen method.

```
[mls,row,col] = mls2tap(N,tap0,tap1)
[mls,row,col] = mls4tap(N,tap0,tap1,tap2,tap3)
```

Generates the `N`-order MLS with two or four feedback taps.

Receives:		Returns:	
`N`	MLS order.	`mls`	the MLS.
`tap*`	feedback taps, according to Table 1.	`row`, `col`	permutation arrays to be used in the IR calculations.

```
[c,B,A] = varredura(fs,t,f0,f1)
```

Generates the logarithmic sine sweep with sample frequency `fs`, duration `t`, start frequency `f0`, and end frequency `f1`.

Receives:		Returns:	
`fs`	sample frequency.	`c`	the logarithmic sine sweep.
`t`	duration.	`B`, `A`	butterworth bandpass filter coefficients to be used in the IR calculations.
`f0`	start frequency.		
`f1`	end frequency.		

From the vectors `mls` or `c`, we may use the function `wavwrite` to output a Microsoft WAVE sound file so that they can be played by an external program. The MLS signal should be generated with at least two repetitions of `mls`.

Impulse Response Calculation

The generated signal should be played inside the room and the response recorded simultaneously. Figure 4 shows the equipment setup we used. It produces a stereo recording

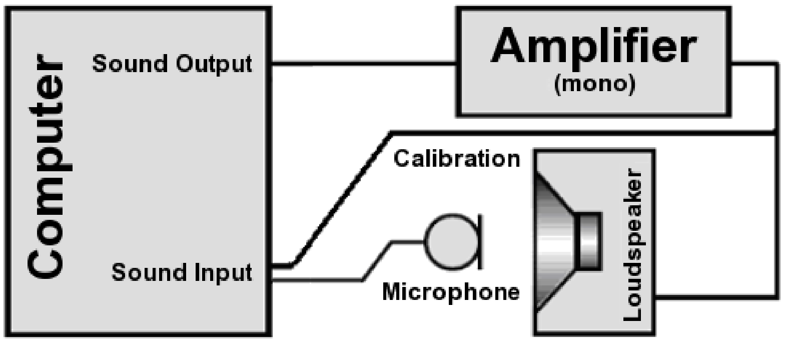


Figure 4: Diagram of the equipment setup

where the first channel is the room response and the second is the calibration. This configuration is important for the processing functions below.

Before we proceed, we must read the recorded samples to a MATLAB matrix. This may be done by the `wavread` function, which reads a Microsoft WAVE file. By now we should have a 2-column matrix where the first column contains the samples of the room response channel and the second column contains the samples of the calibration channel. We can now calculate the impulse response.

`ir = demls(signal,row,col,reps)`	
Calculates the average of the `reps` repetitions of the generated MLS in `signal`, excluding the first one.	
Receives: `signal` the recorded response. `row,col` permutation arrays returned by `mls2tap` or `mls4tap`. `reps` number of repetitions of the generated MLS played.	Returns: `ir` the impulse response.

`ir = dechirp(signal,B,A,N)`	
Deconvolves the sine sweep signal with the room response to it using the FFT.	
Receives: `signal` the recorded response. `B,A` filter coefficients returned by `varredura`. `N` desired length for the resulting IR.	Returns: `ir` the impulse response.

Parameters Calculation

The last step is the calculation of the acoustical parameters. Before being processed, the impulse response needs a treatment to minimize the signal disturbance. We implemented three different methods in the following functions.

`[s] = ldbparam(ir,fs,flag)`			
Implements the Lundeby method.			
Receives:		Returns:	
`ir`	the impulse response.	`[s]`	the acoustical parameters.
`fs`	sample rate.		
`flag`	if equals 1, displays the Schroeder curves.		

`[s] = chuparam(ir,fs,flag)`			
Implements the Chu method.			
Receives:		Returns:	
`ir`	the impulse response.	`[s]`	the acoustical parameters.
`fs`	sample rate.		
`flag`	if equals 1, displays the Schroeder curves.		

`[s] = hrtparam(ir1,ir2,fs,flag)`			
Implements the Hirata method.			
Receives:		Returns:	
`ir1,ir2`	two impulse responses obtained under the same conditions.	`[s]`	the acoustical parameters.
`fs`	sample rate.		
`flag`	if equals 1, displays the Schroeder curves.		

3. Usability Requirements

The task of taking measurements from a room demands the maximum of agility and organization. Considering the high number of signal takes and measurement positions, we want the software to support the work on the field. During the measurements, we dealt with the manipulation, conversion, and storage of the data without the use of any special software, we only used our MATLAB prototype and a few audio tools. That experience showed us the importance of a integrated system designed for these tasks that offers an agile and flexible interface. We then enumerated some usability requirements for the software we are building.

1. **Organization of the data**: when we measure a room, we must perform the signal takes in a lot of different positions. On top of that, each take must be repeated several times. This generates a great number of recordings, each one with its set of calculated parameters associated with it, so it is important to label and store them properly.
2. **Display of results**: we also need an efficient way to display this great amount of data. This should be done in the form of graphs and tables generated by the software.
3. **Audio tools**: tools for audio visualization may be useful when dealing with the recordings.
4. **Automated repetitions**: given that we need to repeat the same take a few times for each position, the software should be able to do it automatically.
5. **Integrated environment**: all these tools and features should be integrated in a single environment for convenience.

4. AcMus Measurement Module

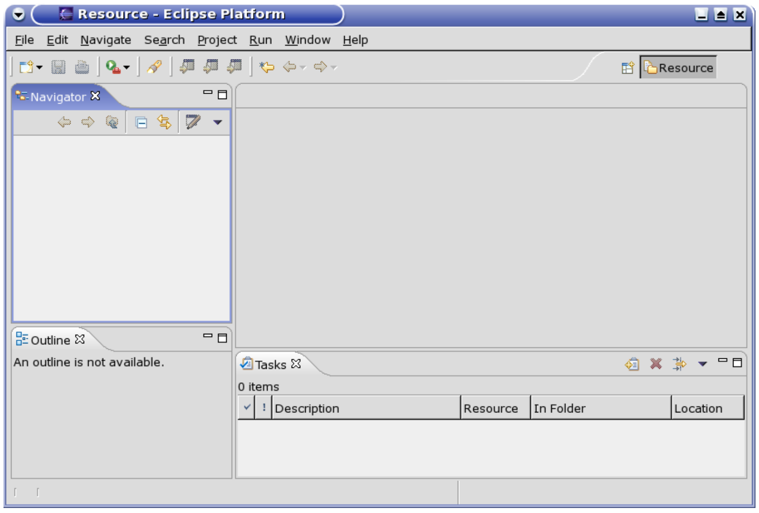
We are working on the final Java implementation of the AcMus Measurement Module taking into account the experience we gained from the measurements. We plan to implement the three modules of the AcMus software in the same environment, satisfying Requirement 5. We are doing this with the help of the Eclipse Platform.

4.1. Eclipse

The AcMus software is being implemented as a set of Eclipse plug-ins. Eclipse (`http://www.eclipse.org`) is a powerful, generic, extensible, open set of computer tools for developing programs. It is actually a general-purpose Integrated Development Environment (IDE) that can be largely extended with contributed plug-ins written in Java. It provides a foundation for constructing and running these plug-ins, allowing extensions built by different people to integrate seamlessly and become part of the Eclipse Platform. It also counts with a great number of high quality open-source plug-ins offered by the Eclipse community.

The Eclipse Platform is then used by both the plug-in programmers (who have the role of the platform developers) and the application programmers (who have the role of the platform users). Its architecture is based on the concept of *plug-in*. Plug-ins are pieces of programs with a defined structure that add functionality to the platform. A plug-in may define *extension points* to let other plug-ins extend it. Almost the entire platform is built upon this model, even its main subsystems are structured as sets of plug-ins, resulting in a highly extensible system.

When started, the Eclipse Platform opens the *workbench*, shown in Figure 5. Besides the menus and the toolbar, the workbench also contains panels, known as *views*,


Resource - Eclipse Platform
File Edit Navigate Search Project Run Window Help
Resource
Navigator
Outline
Tasks

An outline is not available.

0 items

✓	!	Description	Resource	In Folder	Location

Figure 5: The Eclipse Workbench with the default perspective.

and an *editor* area. These elements compose a graphical interface for the development and management of *projects*. A project is mapped to a folder in a file system. Files inside this folder represent the project resources. A *perspective* defines a collection of view, menus, and buttons in the toolbar. In a way, the perspective determines the kind of tasks for which the workbench will be useful. The default perspective offers generic views for project management and basic file editing. For example, the Navigator View lets the user create, select, and remove projects. To its right is the editor area. When a document is selected in the Navigator View, the appropriate editor windows opens there. The Java perspective, accordingly, offers the functionality of a full-featured Java IDE.

Each of these GUI elements is a plug-in or a set of plug-ins that can itself be extended. The AcMus platform is being built upon this model. The Measurement Module described in this paper implements a perspective that offers customized views and editors.

4.2. AcMus Perspective

The *AcMus Perspective* is shown in Figure 6. The view on the left hand side of the window

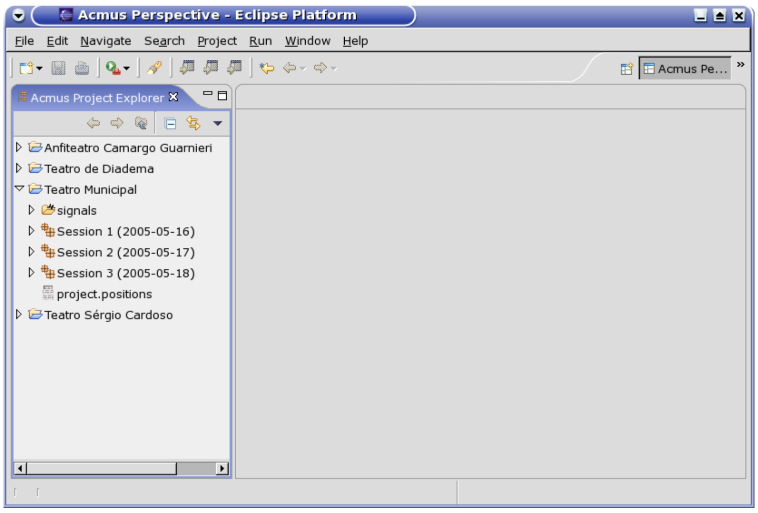


Figure 6: The AcMus Perspective.

is the *AcMus Project Explorer*, which extends the Eclipse Navigator View. It has the same purpose of managing projects. In order to create new projects and resources, we provide wizards that were built using the wizard framework offered by the Eclipse Platform. The plug-in also defines different types of resources, some of them having their own editor. For example, the resources that represent audio files may be viewed with the *Audio Player* (which is actually an editor), as shown in Figure 7. This player draws the waveform and

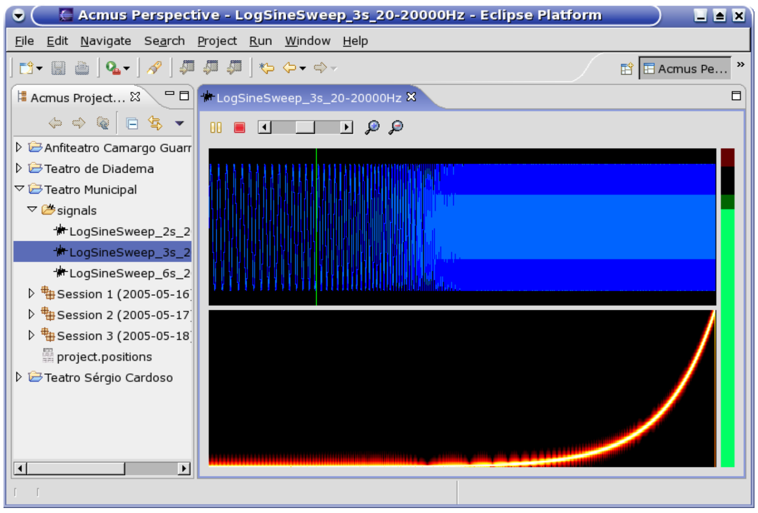


Figure 7: The Audio Player.

the spectrum of an audio file, conforming with Requirement 3.

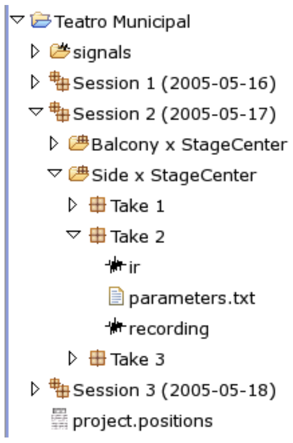
In the next sections we show more of these elements and how they work together.

Measurement Management

The Project Explorer view lets the user store measurements taken from different rooms in hierarchical folders. There are four main kinds of folders.

- **Project folder**: represents a room.
- **Session folder**: groups measurements taken at a specific period in time.
- **Set folder**: stores the repetitions of the same measurement.
- **Measurement folder**: stores the audio file of the room's response and the output of the response analysis.

The wizards for the creation of these folders let the user input additional information about the folder such as date, time, equipment, comments, and so on. This structure, designed to comply with Requirement 1, induces a certain organization. Figure 8 shows an actual example.



```
Teatro Municipal
  signals
  Session 1 (2005-05-16)
  Session 2 (2005-05-17)
    Balcony x StageCenter
    Side x StageCenter
      Take 1
      Take 2
        ir
        parameters.txt
        recording
      Take 3
  Session 3 (2005-05-18)
  project.positions
```

Figure 8: An example of the organization of the folders.

The special folder **signals** stores the generated signals to be played in the room. The current implementation provides a wizard for the creation of logarithmic sine sweeps.

Measurement Interface

A double-click on a measurement folder opens an interface for the management of this measurement, the Measurement editor. The interface, shown in Figure 9, allows the user to perform a measurement, that is, to play the chosen input signal and record the room response. After that, by clicking on the “Calculate IR” button, the user asks the software to

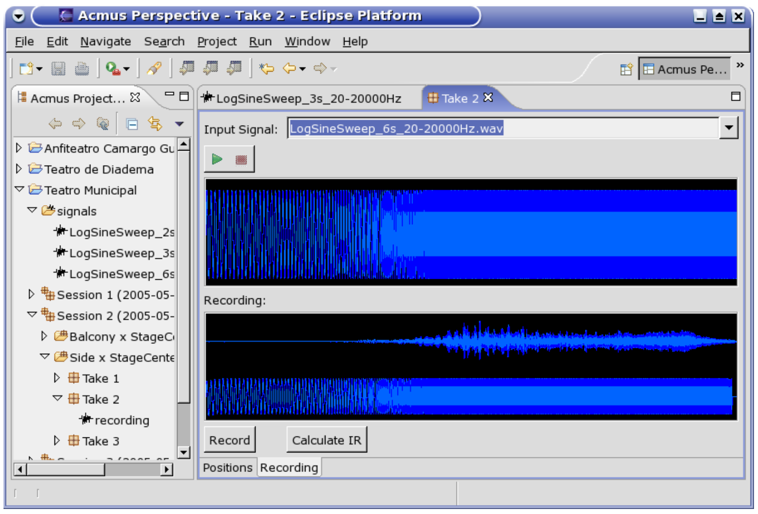


Figure 9: The Measurement Interface.

calculate the impulse response from the captured sound. The result is shown in Figure 10. Finally, the button “Calculate parameters” shows the acoustical parameters obtained from the impulse response. Figure 11 shows the parameters that the current implementation is able to calculate.

The code for this part of the software is a reimplementation of the MATLAB prototype (Section 2.1). For this we are also implementing a DSP library that currently

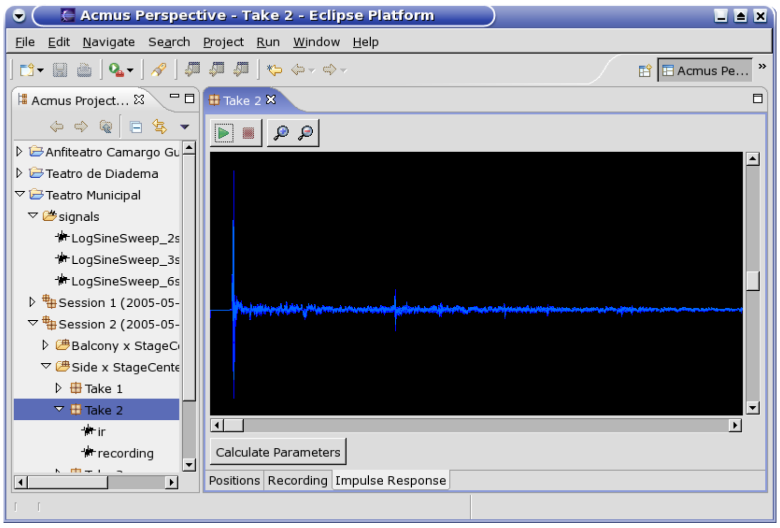
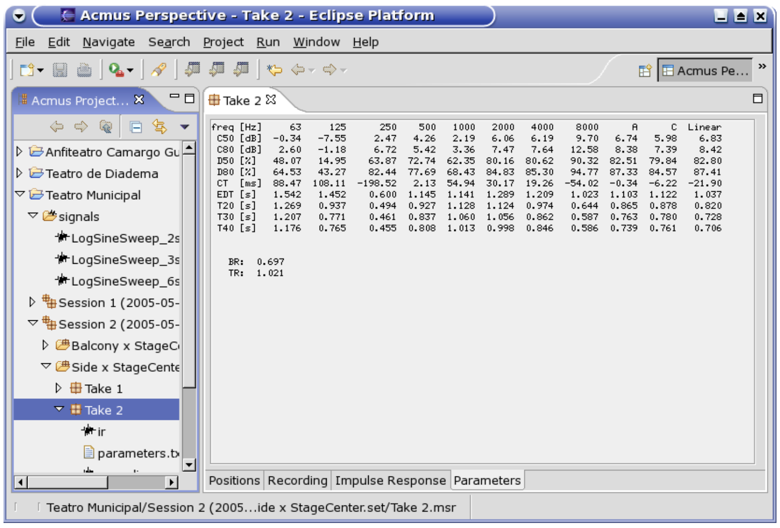


Figure 10: The calculated impulse response.



```
freq [Hz]     63     125      250    500    1000   2000   4000    8000      A      C  Linear
 C50 [dB]  -0.34   -7.55     2.47   4.26   2.19   6.06   6.19    9.70   6.74   5.98    6.83
 C80 [dB]   2.60   -1.18     6.72   5.42   3.36   7.47   7.64   12.58   8.38   7.39    8.42
 D50 [%]   48.07   14.95    63.87  72.74  62.35  80.16  80.62   90.32  82.51  79.84   82.80
 D80 [%]   64.53   43.27    82.44  77.69  68.43  84.83  85.30   94.77  87.33  84.57   87.41
 CT  [ms]  88.47  108.11  -198.52   2.13  54.94  30.17  19.26  -54.02  -0.34  -6.22  -21.90
 EDT [s]   1.542   1.452    0.600  1.145  1.141  1.289  1.209   1.023  1.103  1.122   1.037
 T20 [s]   1.269   0.937    0.494  0.927  1.128  1.124  0.974   0.644  0.865  0.878   0.820
 T30 [s]   1.207   0.771    0.461  0.837  1.060  1.056  0.862   0.587  0.763  0.780   0.728
 T40 [s]   1.176   0.765    0.455  0.808  1.013  0.998  0.846   0.586  0.739  0.761   0.706


   BR:   0.697
   TR:   1.021
```

Figure 11: The calculated parameters.

has a few FFT and filtering functions. At this early stage of development, we do not know yet if the Java performance will be an issue, since we have been doing a straightforward translation from the MATLAB functions. If this is the case, we may have to implement some of these functions in C++ and integrate them with the JNI [Liang, 1999] at the cost of the full portability.

5. Conclusions

The AcMus software offers the main acoustic calculations and processing tools available in similar commercial systems such as WinMLS (`http://www.winmls.com`), Aurora (`http://pcfarina.eng.unipr.it/aurora/home.htm`), ETF5 (`http://www.etfacoustic.com`), and Sample Champion (`http://purebits.com/`). However, some additional tools are not implemented in AcMus, for instance, signal convolution and deconvolution, and STI (Speech Transmission Index) and ST1 (support) calculations, but they may be added in future releases.

Besides, in order to share the results of our research with other groups more easily, the AcMus software is an open-source project. All the source code and papers we produced are available at our Web site: `http://gsd.ime.usp.br/acmus`. None of the mentioned systems are completely freely available and open-source.

5.1. Next Steps

In addition to the analysis of the theaters we measured, we have work to be done on the final implementation of the Measurement Module. There is a lot of room for enhancements, especially in the measurement interface, so that we could properly meet Requirements 4 and 2. Also, the parameter calculation functionality is not complete yet.

All this work is of great importance for the last stage of the project: the final implementation of the Simulation and Optimization Module. We have been working on this module since the beginning of the project [de Queiroz, 2003, de Avelar Gomes et al., 2004], so now we are getting ready to incorporate this research to the AcMus integrated platform.

References

Beranek, L. L. (2004). *Concert halls and opera houses: music, acoustics, and architecture*. Springer-Verlag, New York.

de Avelar Gomes, M. H., Vorländer, M., and Gerges, S. N. Y. (2004). Measurement and use of scattering coefficients in room acoustic computer simulations. In *Proceedings of the European Acoustics Symposium*, Portugal.

de Queiroz, M. G. (2003). Some optimization models for listening room design. In *Proceedings of the 9th Brazilian Symposium on Computer Music*, pages 149–154, Campinas, Brazil.

Figueiredo, F. L., Masiero, B. S., and Iazzetta, F. (2004). Análise de parâmetros acústicos subjetivos: Critérios para avaliação acústica de salas de música. In *Anais da 4ta. Reunion Anual de la Sociedad Argentina para las Ciencias Cognitivas de la Música*, Tucumã, Argentina.

Iazzetta, F., Kon, F., de Queiroz, M. G., da Silva, F. S. C., and de Avelar Gomes, M. (2004). AcMus: Computational tools for measurement, analysis and simulation of room acoustics. In *Proceedings of the European Acoustics Symposium*, Portugal.

ISO 3382 (1997). Acoustics – measurement of the reverberation time of rooms with reference to other acoustical parameters.

Kuttruff, H. (1991). *Room Acoustics*. Elsevier Applied Science, London.

Liang, S. (1999). *Java Native Interface: Programmer's Guide and Reference*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Masiero, B. S. (2004). Estudo e implementação de métodos de mediação e resposta impulsiva em salas de pequeno porte. Technical report, Fapesp. Available at `http://gsd.ime.usp.br/acmus/publi/relat_medicao.pdf`.

Müller, S. and Massarani, P. (2001). Transfer function measurements with sweeps. *J. Audio Eng. Soc.*, 49:443.

Ueda, L. K., Kon, F., and Iazzetta, F. (2005). An open-source platform for musical room acoustics research. In *Proceedings of the 2005 International Congress and Exposition on Noise Control Engineering*, Rio de Janeiro, Brazil.

Yili, Y., Silva, F., Iazetta, F., and Kon, F. (2003). Estimadores de qualidade para pequenas salas destinadas a atividades musicais. In *Anais do IX Simpósio Brasileiro de Computação Musical*, pages 163–170, Campinas, Brazil.