Director Quick Start Manual

Managing a game audio project with

Contents

Introduction

About this manual

This manual is intended for audio directors, game directors and project managers. It will give you the basic knowledge essential to start managing a game audio project with ADX2 from CRI Middleware. You will learn about the main features of ADX2, how to organize and maintain your project, and how members of the audio department can collaborate and communicate with the game team to create truly immersive audio.

About ADX2

ADX2 is a leading game audio middleware. It has already been used in more than 3200 games in all genres and on all platforms.

ADX2 offers a user-friendly, DAW-like, authoring tool and a fully-optimized audio engine, including high-performance proprietary codecs.

About CRI

CRI Middleware is a Japanese company headquartered in Tokyo, with offices in San Francisco. Founded in 1983, it specializes in audio and video software solutions for the digital entertainment industry.

Popular products include Sofdec2, a versatile movie encoding and playback solution and ADX2, an intuitive and powerful game audio middleware. Both are part of the CRIWARE SDK which is available for Unity, for Unreal Engine and in native mode for the main game platforms.

What is a game audio middleware?

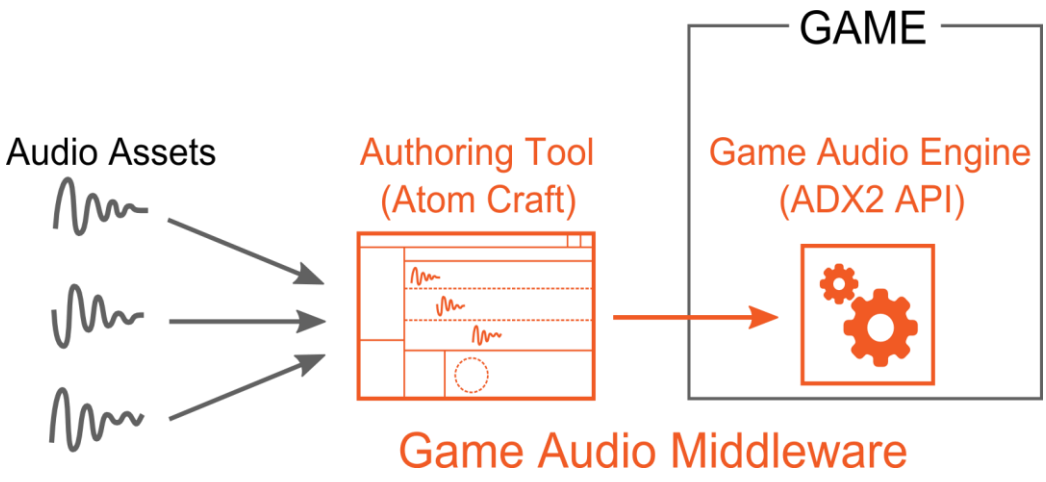
If you come from project management and are not familiar yet with what a game audio middleware is, we will quickly explain it now. A game audio middleware is composed of an authoring tool and a run-time component (the game audio engine), in this case AtomCraft and the CRI Atom library.

The authoring tool allows the sound designer to import audio assets (wave files), transform them into dynamic events that can be triggered and controlled, organize them into sound banks and export them to the game.

A game audio middleware provides specific features for sound effects (e.g. randomization and 3D positioning), interactive music (e.g. beat synchronization and transitions) and dialog (e.g. support for multiple languages).

A game audio middleware also offers mechanisms for the game and the audio engine to communicate (e.g. through game audio variables), so that the audio can truly be dynamic and react to what is happening in the game.

Last but not least, a game audio middleware usually provides in-game previewing from within the authoring tool, lowering the time required to debug and iterate on the sound design.



Overview of ADX2

The AtomCraft tool offers a DAW-like experience on top of which it adds interactive features to allow for the creation of immersive soundscapes. The main screen, shown below, displays the project and audio assets trees on the left, the details pane (here showing a timeline) in the middle, and the parameters at the bottom (property lists can also be shown on the right).

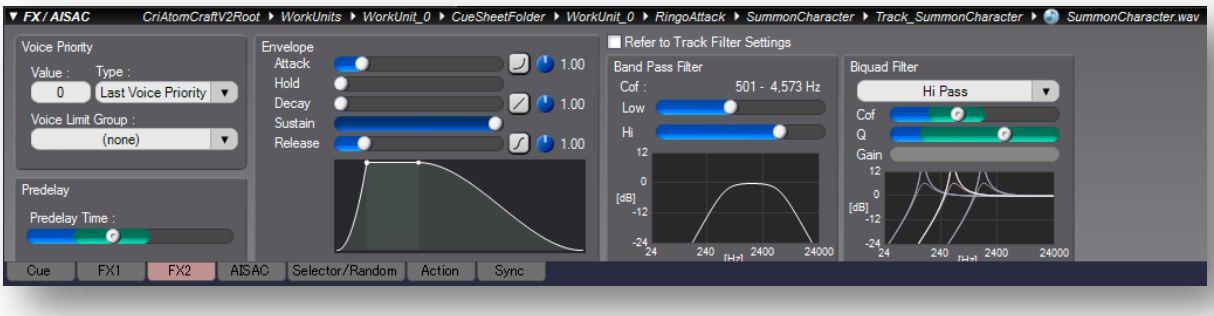
Data hierarchy and main features

As visible in the tree, a project is divided into Work Units. These are non-run-time objects used to organize the audio in the project by type or function (game location / character / weapons etc…). It makes it easier to distribute the work between several sound designers and use versioning as we will see later.

Work Units can contain one or more CueSheets, each having one or more Cues (a CueSheet is basically the equivalent of a sound bank and Cues are the audio events / sounds that you will be triggering from the game).

Several types of Cues are available, each exhibiting a different behavior. A Cue is composed of several Tracks, each having a timeline on which Waveform Regions (sample data with some extra parameters) can be arranged. The Cue type will determine the way these Tracks are selected and played. All the typical playing behaviors are available: polyphonic, sequential, random, shuffle etc… as well as some more original ones. For example, a Cue of the combo-sequential type will play the

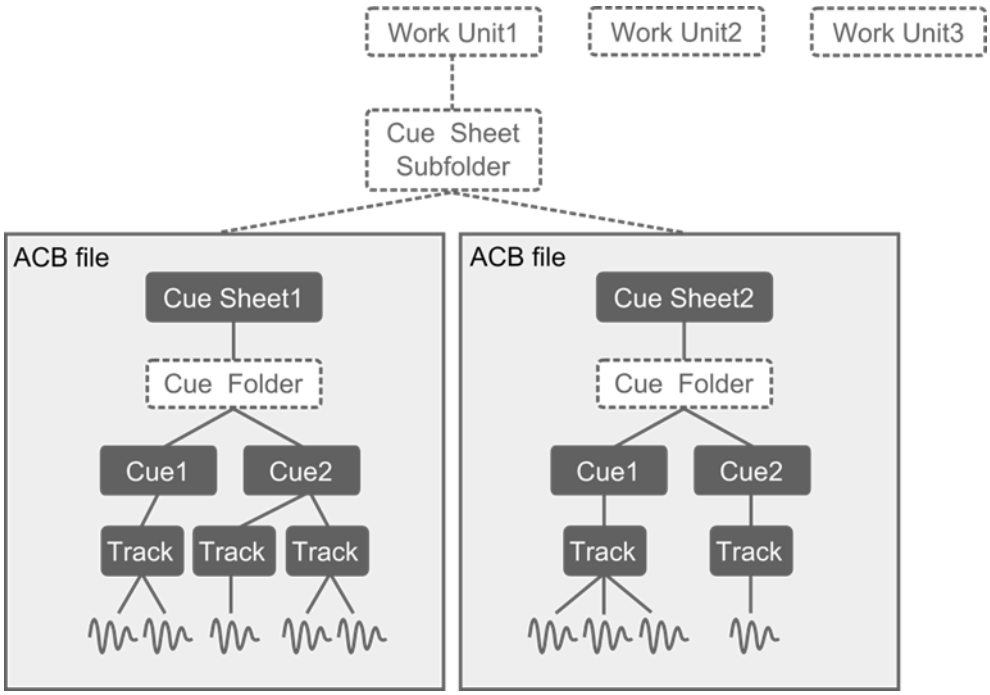
next Track each time it is triggered, but only if it happens within a given time interval, therefore allowing the simulation of combos. If the sound is not triggered in time, it will go back to the first Track.



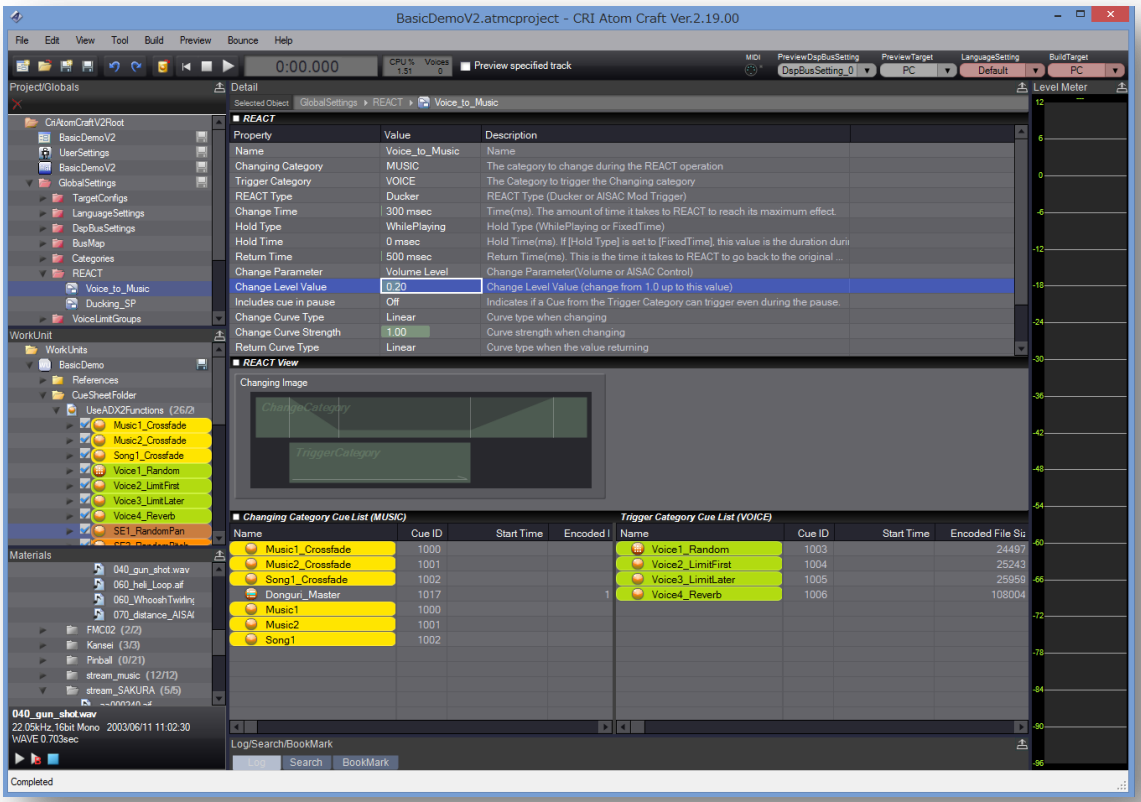
Each Track (and even each Waveform Region) can have its own parameters (such as an envelope and a biquad filter, as depicted above). Actions can also be inserted on a Track to provide looping at the Track level (by opposition to the regular sample looping at the Waveform Region level), to set parameters or call other Cues. Finally, a Cue can be organized vertically in blocks with transition rules that make it perfect for interactive music.

The bottom tree on the left of the screen is called the Materials tree. This is where you will drag and drop audio to import them into your project. Once an audio file is registered as a Material, it is possible to specify its looping and encoding parameters, and, of course, to use it as a Waveform Region on a Track.

To summarize, the data hierarchy can be seen below (we will talk about what the ACB file in the next section).

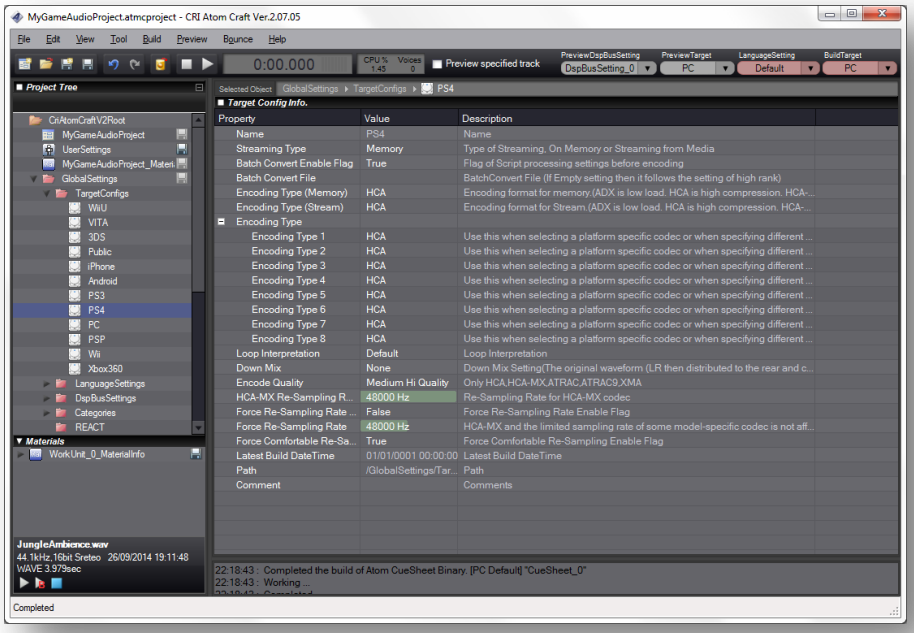


At the global level, Categories, Voice Limit Groups, REACT (automatic ducking, displayed below), AISAC (real-time control parameters), DSP buses can all be defined to control and manage groups of Cues at run-time. Finally, Languages Settings allow for the localization of dialog.



Exporting

Once you have prepared your cues for sound effects, music and dialog, it is time to export the cue sheets towards the game. First, make sure that the target platforms and target languages are correctly configured. You can setup the languages in the *LanguageSettings* folder of the project tree. Similarly, you can configure the target platforms in the *TargetConfigs* folder. In particular, you will be able to select the encoding for each platform:

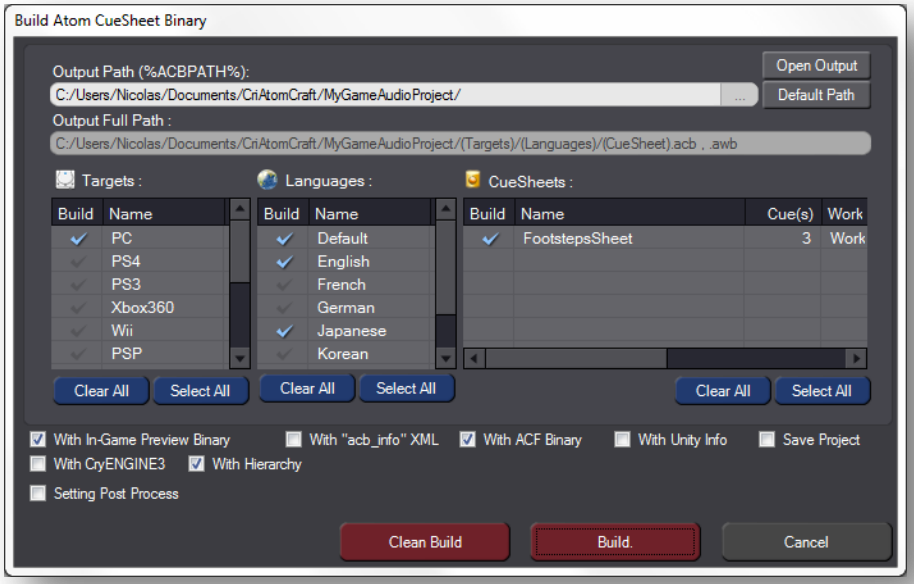


In addition to the regular offer from the console manufacturers, CRI Middleware has developed two proprietary codecs for sample data.

ADX is a high-fidelity, high compression, low CPU-load codec running on all game platforms, mobile devices, and embedded systems. With ADX, 48kHz stereo sounds can be compressed up to 1/16.

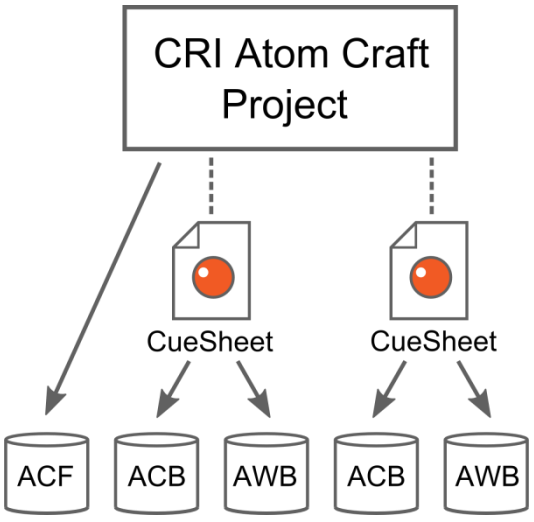
The HCA codec (for High Compression Audio) offers a compression ratio from 1/6 to 1/12. This codec has been tuned for games, the CPU load is very low (compared to MP3 and AAC) and stable during decoding. HCA also supports seeking. The HCA-MX codec is a variation of the HCA which reduces the CPU load when playing back a large number of sounds simultaneously. Thanks to this, the engine can play hundreds of sounds on consoles, and tens of sounds on mobile platforms.

The export itself consists in generating a binary representation of the cue sheets.



You will be able to select the target platforms, the target languages, and of course what cue sheets you want to export.

When AtomCraft exports a project, several files are generated: one ACF file for the whole project, and one ACB and one AWB file per cue sheet exported.



The ACF file (for *Atom Configuration File*) contains the general information about the project and what cue sheets are parts of it. There is only one ACF file outputted per project.

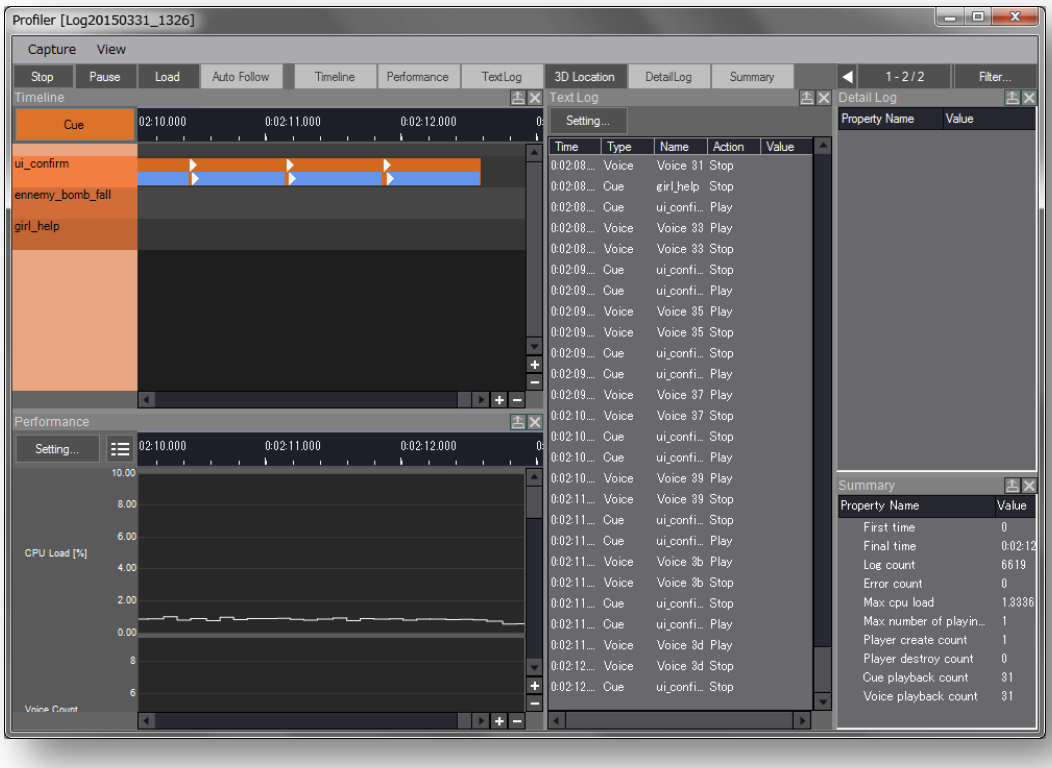
The ACB files (for *Atom Cue sheet Binary*) contain all the parameters of a given cue sheet as well as any sample data destined to be played in memory by that cue sheet. If the audio data is expected to be streamed, it is stored in the corresponding AWB file.

The AWB files (for *Atom Wave Bank*) contain the encoded data for the streams referenced by the cue sheet (the parameters for the stream playback are in the ACB file).

Profiling and previewing

ADX2 provides an in-game preview feature that will allow you to modify the sound parameters and even the waveforms (within certain limits that you can set) within the tool while testing the game.

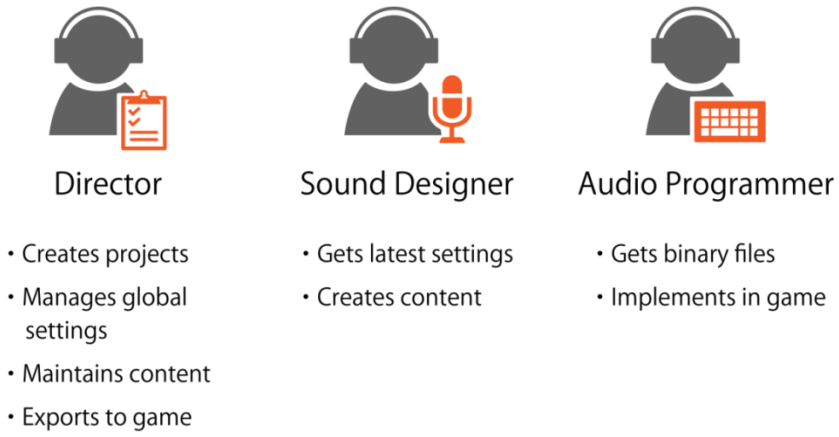
ADX2 also offers a profiler to see what is happening in the game and to optimize your game audio project.



Managing your project

Communication across audio disciplines

Like for all projects, it is important to clearly define the tasks and responsibilities of each member of the team. The picture below shows how the tasks are usually distributed. Note that the Director could simply be someone a bit more senior which is responsible for maintaining an up-to-date version of the audio project.



ADX2 truly empowers both the audio designer and the audio programmer. Depending on the wishes or skillset of your audio team, you can decide to have the sound designers control the whole audio experience, to put the audio programmers fully in charge or a mix of both.

For example, as a sound designer, you could design an automatic ducking system in a few clicks with the REACT feature in AtomCraft (based on the categories the sounds belong too). You could also do it with envelopes or AISACs. The programmer, on his side, could trigger automatic fade-ins / fade-outs with the Fader module at run-time or he could program every single change by updating the volumes of the sound effects themselves or of their respective categories.

This also means that both a sound designer and a programmer could be changing a parameter of the sound independently. The ADX2 library is pretty smart about it and values coming from the tool and the code will be combined based on what makes more sense (for example, volumes will be multiplied while pitch offsets will added and some other parameters may just be overridden by the last value).

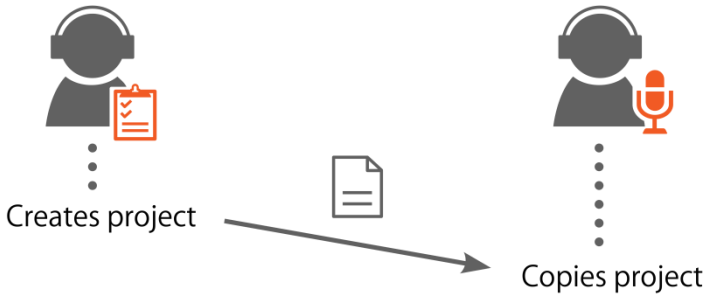
But if this was not planned, this can lead to hard to find bugs. Therefore, it is of the utmost importance that everybody on the team has a clear idea of how a sound is to be implemented, and by whom.

Working with multiple sound designers

When several sound designers are working on the same project, it becomes important to clearly divide the tasks between them. This can be done per level, type of items / character in the game, type of assets (sound effects, dialog and music) etc… ADX2 provides a way to divide and manage projects in work units. These work units can reference their own local audio materials or share common materials at the project level. It is also possible to not load some of them if they have no connection to your work, thus saving memory and allowing for faster loading times.

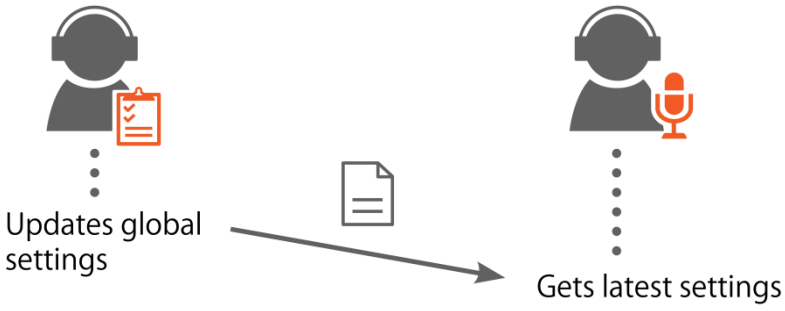
Creation of a new project

The director or project manager creates a new project and manages the master files. Sound designers copy the entire folder of the project created to have the same environment. Then they start the work. The project is created only once and the new Work Units are created by each sound designer.



Updating the global settings

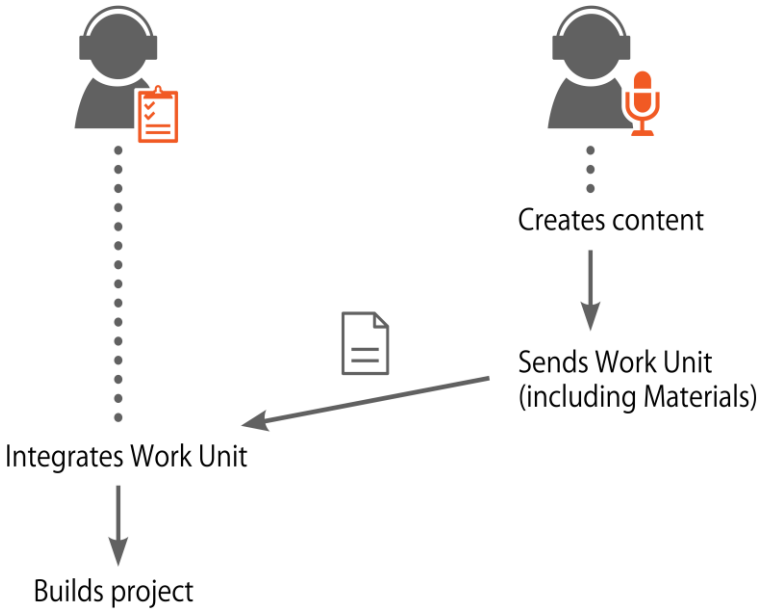
The director or project manager is responsible for building the project and managing the global settings, such as AISACs or categories. The sound designer gets the global settings file and project file from the manager / director and overwrites its own version, then starts the work.



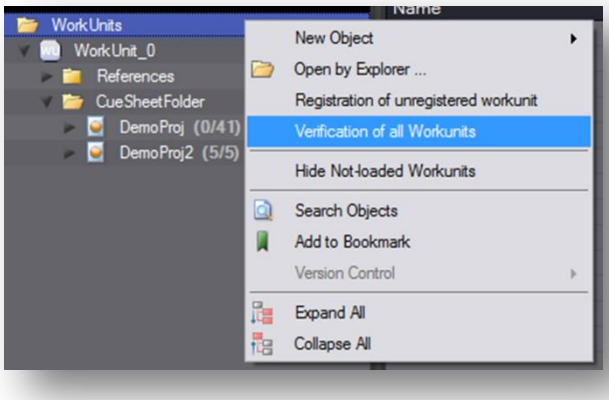
Please note that link errors may occur when the global settings are modified. For example, the manager may have removed a global AISAC that the sound designer is still referencing. In this case, the link error must be resolved. Since only the sound designer may find the link error, the project manager may not be aware of the issue.

Collecting and building all work units

Sound designers send their new Work Units to the audio director / project manager who makes them editable and then builds the corresponding Atom CueSheet binary (ACB files).



The audio director / project manager must check that the links are correct for all the data. To that effect, after collecting all the data, it is possible to use the “Verification of all Work Units” command.



To summarize, here are some rules concerning the collaboration between several sound designers on a given project.

- First, it is recommended to manage the files in the project folder using a version control system (see next section). This is to keep track of the master data and to make sure that it can be reproduced.

- A project should be created only once and each sound designer should then add new Work Units, making sure that there is no duplicate name.

- The person in charge of the project should determine the global settings. Be careful when updating them: when you modify the global settings and a Work Unit is not registered, links may be broken. This can happen when renaming Categories, AISAC links, global AISACs, when changing the Voice Limit Group etc.

- Any broken link should be resolved immediately. Indeed, if the global settings reference unknown data, the links will not be saved.

- The unknown links can be resolved by creating the target of the link or removing the objects with unknown links.

- Work Units and global settings should not be edited at the same time. Work Units cannot be merged automatically. When someone needs to edit a Work Unit and checks it out, nobody else should edit it.

Version control

In AtomCraft, the materials and the work units are saved as separate files. By using a version control system, it becomes straightforward to check the history of these files, how they have been modified, and to revert them to an older version if needed.

Usually, it is necessary to use a dedicated version control client to manage the history of the files. However, since AtomCraft has a version control feature, you can manage the files directly from within the tool, without having to switch all the time between Atom Craft and the version control client.

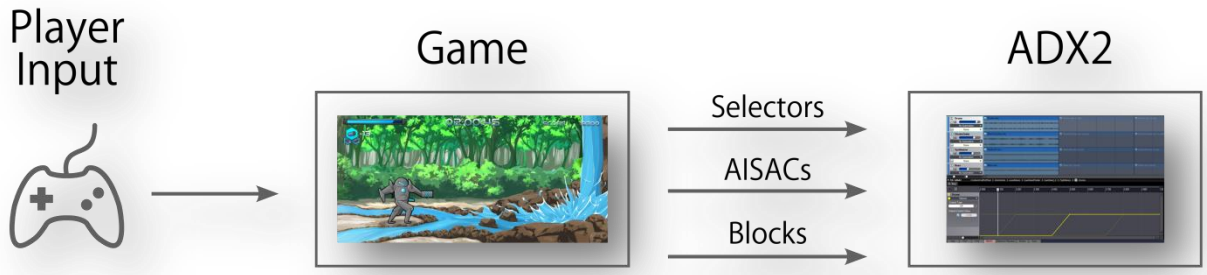
The editing operations in AtomCraft are directly linked to the version control system. For example, when dropping a waveform file onto a CueSheet, the new material is automatically submitted to the repository.

Popular version control systems such as **Subversion** and **Perforce** are supported. However, since the version control functionality is provided by a plug-in, you can easily switch between different version control systems or add new ones.

Communication with the game

By opposition to movies which are linear, games are interactive products whose sounds must respond to the actions of the player and to what is happening in the game. In order to achieve this, the game programmers must be able to communicate the status of the game to the audio engine.

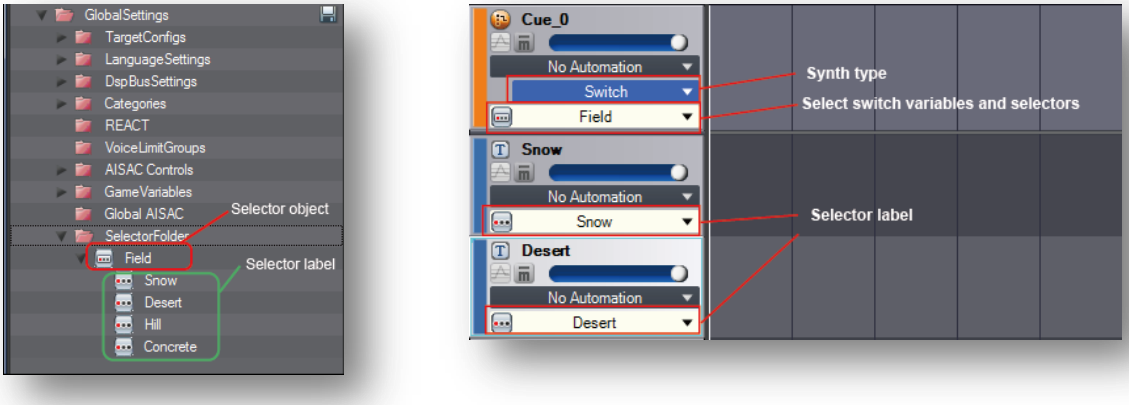
ADX2 provides many ways to do this. Some are semi-automatic, like for example the 3D positioning of the sound source based on its coordinates in the game (once the right parameters have been set in the Atom Craft tool). Others will require more consultation between the audio department and the game team. Here we describe 3 ways programmers and sound designers can collaborate to implement sounds that react to the game context: selectors, AISACs, and blocks.



Selectors

Think about the selectors as variables that can take several predefined values (the selector labels). At run-time, the Cues of type “Switch” will check the value of the selector and play only the Tracks that are associated with that value (i.e. selector label).

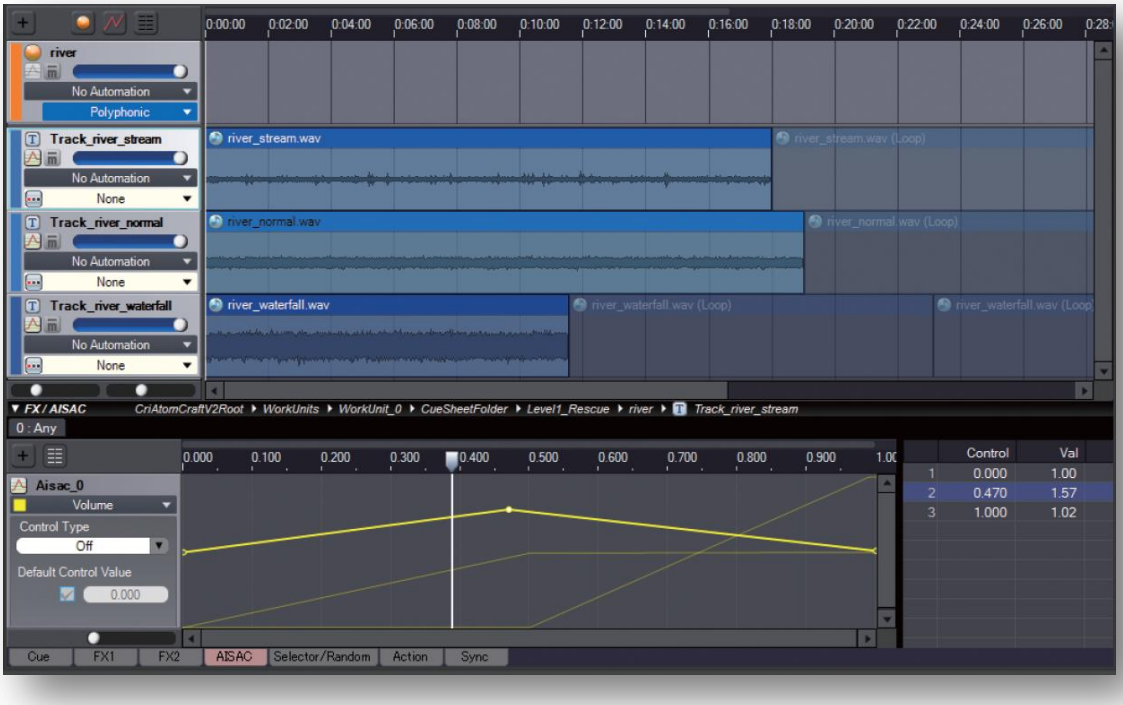
This is very useful for example to play footsteps on various materials with a single Cue. You could have a Selector called “Material” and selector labels such as “Concrete”, “Metal”, ”Grass”, “ Snow”, “Wood” etc... Once the selector labels assigned to the right Tracks, if the Cue is triggered the footstep sound effect corresponding to the type of ground in the game would automatically play.



AISACs

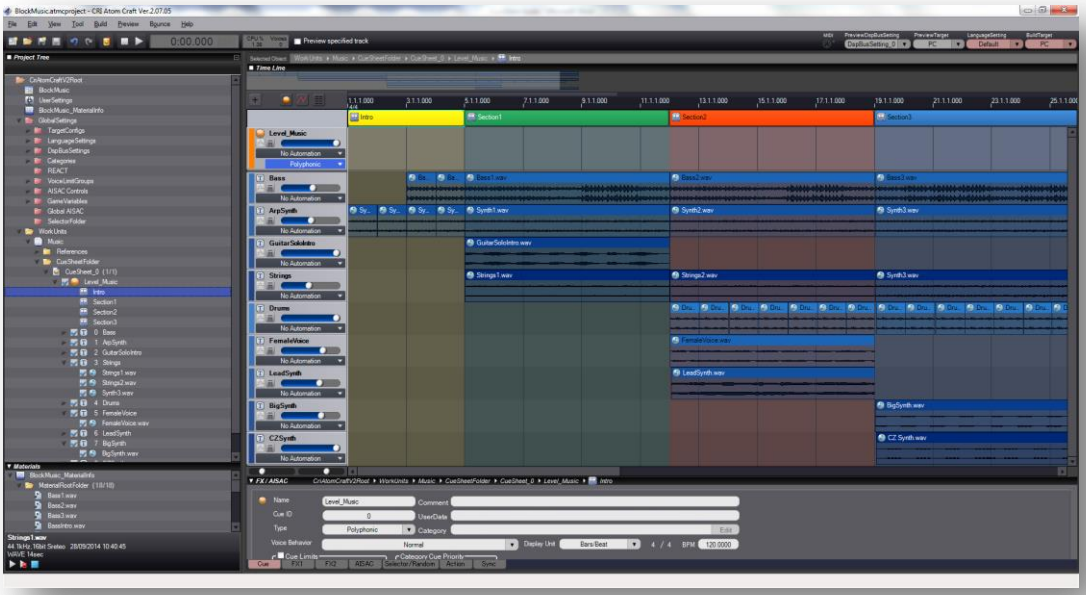
An AISAC is the equivalent of a real-time parameter control (RTPC) in other systems. One or more audio parameters of a Cue can be controlled by an AISAC, which can be either local to that Cue or global. An AISAC can also control another AISAC and it can be used as a randomizer or as a LFO (auto modulation). AISACs can control almost any parameter of a Cue.

The picture below shows a simple example where an AISAC controls the volume of the different tracks of a Cue. Here, each Track corresponds to a different layer of a river sound. The AISAC makes it possible to change the sound of the river's flow simply by passing a value between 0.0f and 1.0f from the game to the Cue.



Blocks

Blocks are usually well suited for interactive music but can also be used advantageously for sound effects. Indeed, they make it possible to jump between different sections of a Cue's timeline and allow the design of complex sonic behaviors. The timing at which the block transitions can occur - as well as the rules to which they obey - can be set in the authoring tool but also overridden at run-time.



Of course, there are also ways to control groups of sounds from the game. You can use Categories for example. Categories can be used to update the parameters of all the sounds that belong to them at the same time. They make it possible to mute a specific type of sounds, to fade them in or out, to apply an AISAC on them etc…

Integrating ADX2 with your workflow

There are several ways to integrate the CRI AtomCraft tool in your current game audio / sound design workflow.

Importing AAF files

In AtomCraft, you can import files in the AAF format. This type of file is generated by DAW such as Logic or ProTools for example. They contain information like the time offset of each waveform and the track names.

This allows you to precisely copy a track arrangement in AtomCraft without having to manually adjust the position of all the waveforms on the timeline. However, please note that the waveform data is not imported from an AAF file so it is necessary to copy the corresponding audio files in the material folder.

Importing / Exporting CSV files

In addition, a number of objects can be exported / imported in CSV format, opening the door to the development of proprietary tools that generate or convert data automatically.

It is for example possible to import / export CueSheets, AISACs and lists of materials.

Sound Forge Batch Processing

If you need to process many files, for example to remove leading / trailing silence or to apply a radio effect on dialog recordings, it is possible to use the batch processing function of Sound Forge 10 on the materials before their encoding in AtomCraft.

To use the batch processing feature, you have to prepare a batch job file (.bj file) which contains a plug-in chain you edited in Sound Forge. Then, in AtomCraft, specify the path to the Sound Forge 10 executable in the options, as well as the batch job file to use for each specific material folder you want to process. Finally, you can run the batch processor from the context menu of the material folder.

Command line tools

If a large number of audio files are needed, the CRI Atom Encoder or the console tool can also be used separately. This data must be managed independently from the CRI Atom Craft projects though.

Where to go from here?

Congratulations! You learned about the main features of ADX2, the type of data involved and how to export it. You also know how to organize your game audio project in work units and use version control within the AtomCraft tool. You understand what information must be communicated across the audio disciplines and what features will allow you to successfully create immersive audio in collaboration with the game team.

This Quick Start manual is only the beginning though. The current version of ADX2 is the result of decades of research and experience in game audio. Whatever the challenges game development brings, whatever the type of your project, you will find ways to implement the audio features you need with ADX2. Please refer to the full SDK documentation to learn more about this game audio middleware.