Une station de travail audio-numérique open-source pour la plate-forme Web

Michel Buffa, Antoine Vidal-Mazuy

Une station de travail audio-numérique open-source pour la plate-forme Web

Michel Buffa, Université Côte d'AZUR, CNRS, INRIA. Nice, France, michel.buffa@univ-cotedazur.fr

Antoine Vidal-Mazuy, Université Côte d'AZUR, CNRS, INRIA. Nice, France, antoine.vidal-mazuy@etu.univ-cotedazur.de

Résumé

Cet article présente WAM Studio (Figure 1), une station de travail audio numérique (DAW) en ligne open source qui tire parti de plusieurs APIs et technologies standards du W3C, telles que Web Audio, WebAssembly, Web Components, Web Midi, Media Devices, etc. WAM Studio s'appuie également sur le standard Web Audio Modules (WAM), qui a été conçu pour faciliter le développement de plugins audio inter-opérables (effets, instruments virtuels, claviers virtuels de piano comme contrôleurs, etc.) sortes de "VSTs pour le Web". Les DAWs sont des logiciels riches en fonctionnalités et donc particulièrement complexes à développer en termes de conception, d'implémentation, de performances et d'ergonomie. Aujourd'hui, la majorité des DAWs en ligne sont commerciaux alors que les seuls exemples open source manquent de fonctionnalités (pas de prise en charge de plugins par exemple) et ne tirent pas parti des possibilités récentes offertes (comme WebAssembly). WAM Studio a été conçu comme un démonstrateur technologique pour promouvoir les possibilités offertes par les innovations récentes proposées par le W3C. L'article met en évidence certaines des difficultés que nous avons rencontrées (par exemple, les limitations dues aux environnements sandboxés et contraints que sont les navigateurs Web, la compensation de latence quand on ne peut pas connaître le hardware utilisé, etc.). Une démonstration en ligne, ainsi qu'un *repository* GitHub pour le code source sont disponibles. Wam Studio a également donné naissance à Attune, une version spéciale adaptée pour un projet de recherche en collaboration avec l'équipe MERI du CCRMA de Stanford, pour l'étude et la facilitation de l'écoute de musique multipiste par des personnes malentendantes équipées d'implants cochléaires.

1. Introduction

La Musique Assistée par Ordinateur (MAO) est un domaine en constante évolution qui utilise des ordinateurs pour enregistrer, éditer et produire de la musique. Les stations de travail audio numérique (Digital Audio Workstations en anglais, ou "DAWs") sont des logiciels spécialement conçus pour la MAO, permettant aux utilisateurs de créer et de manipuler du contenu audio numérique ainsi que du contenu MIDI. Les plugins audio sont des modules logiciels qui ajoutent des fonctionnalités supplémentaires aux DAWs, offrant aux utilisateurs une plus grande flexibilité et un meilleur contrôle sur leur production musicale. Le marché de la MAO est né avec l'Atari ST (1985) -premier ordinateur supportant le standard MIDI- et le DAW Cubase proposée par Steinberg (1989). Peu après, le standard VST pour les plugins audio a été proposé (1997) et depuis lors, des milliers de plugins ont été développés, pouvant être utilisés dans les principaux DAWs natifs disponibles sur le marché.

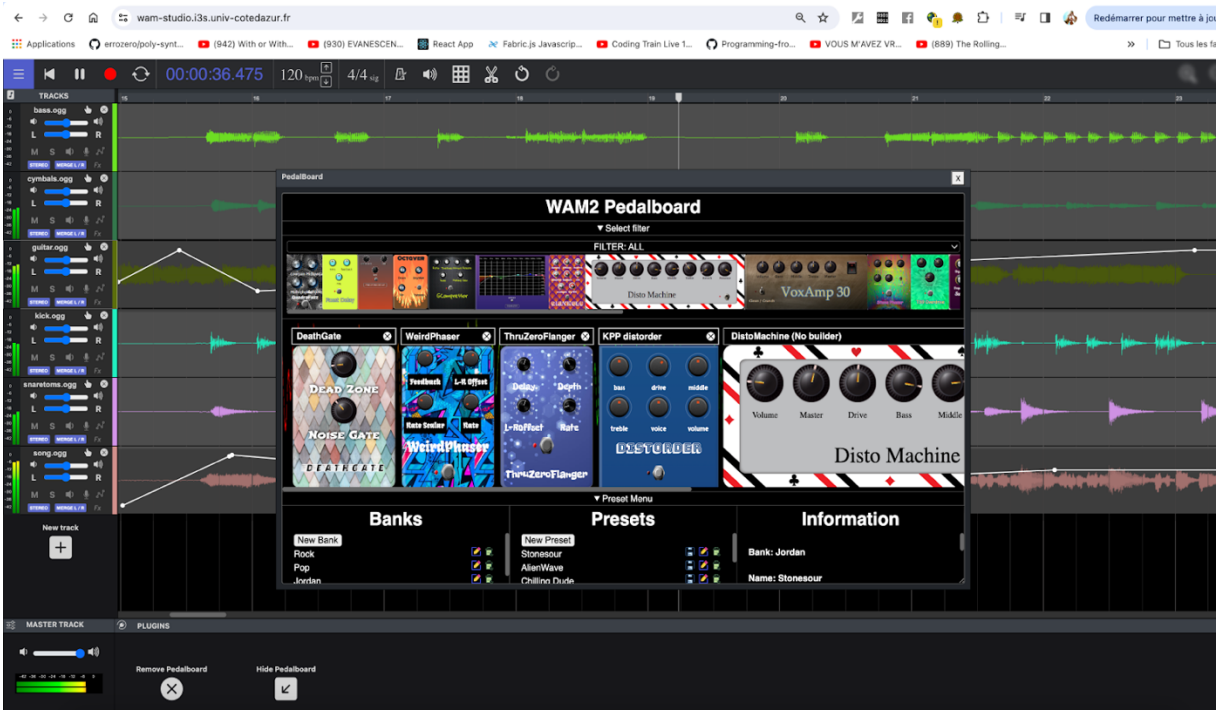

Figure 1 : Image typique de la station de travail audio-numérique Wam Studio.

Un DAW est un logiciel riche en fonctionnalités et donc particulièrement complexe à développer en termes de conception, d'implémentation et d'ergonomie. Il permet la création de pièces multipistes en utilisant directement des échantillons audio (par exemple en incorporant

dans une piste un fichier audio ou en enregistrant à partir d'un microphone ou d'une entrée de carte son), en les mélangeant, en appliquant des effets sonores à chaque piste (par exemple, réverbération, égalisation de fréquence ou auto-tune sur les voix), mais également en utilisant des pistes avec des instruments virtuels (reproduction logicielle d'un piano, d'un violon, d'un synthétiseur, d'une batterie, etc.). La piste est ensuite enregistrée au format MIDI sous forme d'événements correspondant aux notes et à des paramètres supplémentaires (comme la vélocité avec laquelle on a appuyé sur les touches d'un clavier de piano, par exemple). Ces événements permettent à la piste d'être jouée en demandant à un instrument virtuel de synthétiser le signal. Une DAW permet donc la lecture, l'enregistrement et le mixage à la fois de pistes audio et MIDI, l'édition de ces pistes (copier/couper/coller dans une piste ou entre les pistes), la gestion des effets audio en temps réel et des instruments virtuels, le mixage et l'exportation du projet final dans un format simple (par exemple, un fichier WAVE ou MP3). Dans le monde natif, ces effets et instruments sont les "plugins audio" qui étendent les capacités des DAWs standard. Depuis 1997, un marché important pour les développeurs de plugins tiers s'est développé. Quatre DAWs dominent le marché (Logic Audio, Ableton, Pro Tools, Cubase)[1] et l'existence de plusieurs formats de plugins propriétaires complique la tâche des développeurs.

D'un autre côté, le marché de la MAO sur le Web est encore émergent, les premiers DAWs en ligne sont apparus en 2008 et exploitaient la technologie Flash. Ceux utilisant HTML5 et l'API Web Audio ne sont apparus que dans la période entre 2015 et 2016 car les technologies sont beaucoup plus récentes (les premières implémentations de l'API Web Audio dans les navigateurs datent de 2012). En 2023, plusieurs DAWs basés web sont disponibles, principalement commerciaux. Un format de plugins inter-opérables appelé Web Audio Modules (WAM) existe et est soutenu par au moins un DAW en ligne.

2. L'API Web Audio

Depuis 2018, l'API Web Audio du W3C est une "recommandation" (une norme figée). Elle offre un ensemble de "nœuds audio" qui traitent ou produisent du son et on utilise ces nœuds depuis du code JavaScript en instanciant des classes fournies par l'API. Ces nœuds peuvent être connectés pour former un "graphe audio". Le son se déplace à travers ce graphe à la fréquence d'échantillonnage (valeur par défaut 44100Hz, cette valeur peut être modifiée) et subit des transformations (Buffa et al., 2018). Certains nœuds sont des générateurs d'ondes ou des sources sonores correspondant à une entrée de microphone ou à un fichier sonore chargé en mémoire, d'autres transforment le son. La connexion de ces nœuds dans le navigateur se fait

[1] . https://www.mordorintelligence.com/industry-reports/digital-audio-workstation-market

via JavaScript et permet de concevoir une large gamme d'applications différentes impliquant le traitement audio en temps réel .

Les applications musicales ne sont pas les seules à nécessiter du traitement audio complexes, l'API est également conçue pour répondre à d'autres besoins, par exemple pour le développement de jeux vidéo, pour le multimédia, la visio-conférence, etc. L'API est livrée avec un ensemble limité de nœuds "standards" pour des opérations courantes telles que le contrôle de gain, le filtrage de fréquences, l'ajout de *delay*, de réverbération, pour des traitements sur la dynamique, sur la spatialisation du son 2D et 3D, etc.

En général les bibliothèques/API audio répartissent le travail entre un thread de contrôle et un thread de rendu (Déchelle et al., 1999), (McCartney 2002), (Puckette, 2002). L'API Web Audio ne fait pas exception et utilise également ce modèle : un thread de rendu appelé "audio thread" est chargé exclusivement du traitement du son par le graphe audio et de la livraison des échantillons sonores au système d'exploitation pour qu'ils puissent être lus par le matériel. Ce thread est soumis à des contraintes de temps réel strictes et a une priorité élevée - s'il ne parvient pas à fournir le prochain bloc d'échantillons audio à temps (128 échantillons par défaut dans le cas de l'API Web Audio), il y aura des anomalies audibles. D'autre part, le thread de contrôle est généralement en charge d'exécuter le code JavaScript de l'interface utilisateur et d'effectuer les appels à l'API Web Audio. Il gère aussi toutes les modifications apportées au graphe audio : il permet par exemple à l'utilisateur de connecter/déconnecter des nœuds et d'ajuster leurs paramètres.

Les nœuds standards sont tous des instances de sous-classes de la classe AudioNode. A une exception près, ils fournissent des traitements prédéfinis codés en C++ ou en Rust (Firefox) et exécutent le traitement du son dans le thread audio, mais les algorithmes utilisés ne peuvent pas être modifiés, seules les modifications de paramètres sont autorisées depuis du JavaScript. L'exception citée est l'ajout récent (2018) du nœud AudioWorklet qui fournit une solution pour implémenter un traitement audio personnalisé de bas niveau s'exécutant dans le thread audio (Choi, 2018), avec de nombreuses contraintes (pas d'opérations asynchrones, d'imports de fichiers, pas d'accès au DOM de la page HTML, etc.).

Les nœuds de l'API Web Audio peuvent être assemblés dans un graphe permettant aux développeurs de créer des effets ou instruments audio plus complexes. Voici quelques exemples construits de cette manière : effet *delay* (DelayNode, BiquadFilterNode, GainNode), *auto-wah* (BiquadFilterNode, OscillatorNode), *chorus* (multiples DelayNodes et OscillatorNodes pour la modulation), *distorsion* (GainNode, WaveShaperNode), *synthétiseurs* (OscillatorNodes), *samplers* (AudioBufferNodes), etc.

Du code DSP existant dans d'autres langages tels que C/C++ ou écrit en utilisant des langages spécifiques à un domaine tels que FAUST (Ren et al., 2020), peut être compilé en WebAssembly et exécuté dans un seul nœud AudioWorklet. Au fil des années, de nombreux effets et instruments audio de haut niveau ont ainsi été développés (Buffa et al., 2020).

Cependant, il est souvent nécessaire de chaîner de tels effets et instruments audio (par exemple, le pédalier d'un guitariste est composé de pédales d'effets connectés entre elles) et lors de la composition/production musicale dans des DAWs, plusieurs effets et instruments sont en général utilisés. Ce sont des cas où les nœuds de l'API Web Audio sont de trop bas niveau, d'où la nécessité d'une unité de plus haut niveau pour représenter l'équivalent d'un plugin audio natif (Buffa et al., 2022). Pour la plate-forme Web, un tel standard de haut niveau pour les "plugins audio" et les applications "hôtes" n'existait pas avant 2015 (Kleimola, Larkin, 2015). Plusieurs initiatives ont été lancées et l'une d'entre elles est devenue un "standard communautaire" que nous détaillons dans la section suivante.

3. Web Audio Modules

En 2015, Jari Kleimola et Olivier Larkin ont proposé une norme pour des plugins Web Audio sur le Web, intitulée "Web Audio Modules" (WAM) (Kleimola, Larkin, 2015). Peu après, Jari Kleimola a participé à la création d'ampedstudio.com, l'un des premiers DAWs en ligne utilisant l'API Web Audio. En 2018, le projet WAM initial a été étendu par des chercheurs avec l'aide de développeurs de l'industrie de la MAO, ce qui a donné lieu à une norme plus polyvalente (Buffa et al., 2018). Finalement, la version 2.0 des Web Audio Modules a été publiée en 2021. Cette version vise, en plus d'établir une norme communautaire (API, SDK), à apporter plus de liberté aux développeurs (support d'outils de build, TypeScript, frameworks comme React, etc.), à améliorer les performances, à simplifier l'accès aux paramètres de plugin et à faciliter l'intégration dans les DAWs (Buffa, Ren et al., 2022).

Nous reviendrons sur cette fonctionnalité plus tard, mais le standard WAM 2.0 utilise une conception originale pour la gestion de la communication entre les plugins et les applications hôtes qui ne repose pas sur la gestion des paramètres fournie par l'API Web Audio. La raison principale est de permettre des performances élevées dans le cas où à la fois un DAW et des plugins sont implémentés en tant qu'AudioWorklet. En effet, au moment de la conception de l'API Web Audio, les AudioWorklets n'existaient pas et certains cas d'utilisation ne pouvaient pas être pris en compte. Si le DAW est construit en utilisant des nœuds AudioWorklet pour le traitement audio, alors certaines parties du code s'exécutent dans le thread audio à haute priorité. En outre, si un plugin WAM est associé à une piste donnée dans un projet DAW, et si le plugin est lui-même construit en utilisant un nœud AudioWorklet, il dispose également de

code personnalisé s'exécutant dans le thread audio. Le standard WAM a été conçu pour gérer ce cas particulier et permet une communication DAW/plugins *sans quitter le thread audio*.

Prenons un exemple : pendant la lecture, une piste MIDI envoie des notes à un plugin d'instrument virtuel et modifie certains des paramètres de ce plugin à la fréquence d'échantillonnage (a-rate). N'oublions pas qu'un DAW peut avoir plusieurs pistes, chacune associée à des dizaines de plugins, et que chaque plugin peut avoir des dizaines de paramètres. Le standard WAM détectera automatiquement le cas où DAW et plugins sont des AudioWorklets et optimisera en coulisse la communication (avec utilisation de mémoire partagée et de buffer tournant), sans franchir la barrière du thread audio. Pas besoin d'envoyer des événements à partir du thread de contrôle/GUI, ce qui aurait été obligatoire si la gestion des paramètres de l'API Web Audio était utilisée.

En résumé, le standard WAM simplifie la création de plugins et d'applications hôtes et permet une communication optimale entre les hôtes et les plugins.

4. Analyse de l'existant : DAWS Commerciaux et Open Source

		bandlab	ampedstudio	soundtrap	soundation	arpeggi	audiotool
Target	Desktop	X	X	X	X	X	X
	Mobile (native or responsive webapp)	X		X		X	
	Oriented to users familiar with Computer Music		X	X			
	Premium (free, with premium functionalities)	X	X	X	X		X
functionalities	Basic DAW features (recording, editing, mixing etc.)	X	X	X	X	X	X
	Audio effects and virtual instruments	X	X	X	X	X	X
	Commercials plugins support (not documented)	X	X	X	X		
	Open plugins support (WAM)		X				
	Cloud (saving project , loading, sharing, etc.)	X	X	X	X	X	X
	Collaborative (asynchronous)	X	X	X	X	X	X
	Collaborative (synchronous)			X			X
	Call in app		X	X			X
Technologies	Using AudioWorklet	X	X	?			X
	C++/wasm audio engine	X	X				
	Native mobile application	X					
	Blockchain					X	
Open Source							

Figure 2: Table de comparaison des principaux DAWs commerciaux disponibles.

Les premiers DAWs en ligne basés sur l'API Web Audio sont apparus entre 2015 et 2018 et sont encore disponibles aujourd'hui : Audiotools (Lazarevic, Scepanovic, 2010), BandLab, AmpedStudio, SoundTrap (Lind, MacPherson, 2017), Soundation. Ces DAWs ont en commun le fait qu'ils sont commerciaux, avec un code source fermé et ont fait l'objet de très peu de publications académiques. Ils facilitent la collaboration à distance sur des projets musicaux, notons que la pandémie de COVID-19 (2021-2022) leur a été profitable et a augmenté leur popularité. Le mode de collaboration varie (synchrone comme Google Docs ou asynchrone grâce au partage de liens), ainsi que les outils de communication fournis (chat, vidéo-conférence), mais tous ces DAWs proposent les fonctionnalités classiques : enregistrement audio et MIDI, édition de pistes, mixage, support pour effets et instruments virtuels, etc. Certains ont été conçus principalement pour les ordinateurs de bureau, tandis que d'autres sont particulièrement adaptés aux appareils mobiles.

Ils diffèrent principalement en termes d'ergonomie : certaines de ces applications sont destinées à un public très large et ont mis l'accent sur la simplicité (BandLab, SoundTrap), tandis que d'autres ont choisi un aspect plus "professionnel" et sobre, comme AmpedStudio. De son côté, Audiotools est davantage un "studio virtuel" qu'un DAW pur et trouve son marché principalement dans le domaine de l'éducation. Les détails de mise en œuvre de ces applications sont inconnus car le code source n'est pas accessible au public. Cependant, à travers des publications académiques limitées et des présentations/interviews, on sait qu'AmpedStudio utilise un moteur C++ cross-compilé en WebAssembly et des AudioWorklets, tandis que Soundtrap a principalement utilisé les nœuds standard de l'API Web Audio. BandLab est censé avoir un noyau écrit en C++ et s'appuie également sur WebAssembly/AudioWorklet, avec peut-être une version mobile spécifique. Audiotools fonctionne également dans des nœuds AudioWorklet avec un portage du moteur audio initialement écrit en Flash. Récemment, un nouveau DAW appelé Arpeggi.io est apparu, mettant en avant l'utilisation de la technologie blockchain (pour suivre qui, quand et comment les sons et la musique sont utilisés) et des NFTs pour monétiser les créations.

Ces DAW diffèrent également dans la manière dont ils gèrent les effets et les instruments audio. Il est évident que certains prennent en charge des plugins externes : AmpedStudio prend en charge la norme WAM -les développeurs de la société ont contribué à sa création- et le site web a une boutique pour "activer" des plugins premiums. Les autres DAWs en ligne commerciaux semblent utiliser un format propriétaire, tout en intégrant des portages tiers de plugins natifs (SoundTrap propose le célèbre plugin "Auto-tune" d'Antares). Propellerheads, une entreprise bien connue pour son DAW natif Reason, a également publié des versions web de son synthétiseur Europa sous forme de plugin Web (disponible maintenant dans AmpedStudio et dans Soundation, il s'agit d'un portage de Plugin Rack, un standard de plugins propre à

Propellerheads dans le monde natif). Jusqu'à présent, AmpedStudio est le seul DAW à prendre en charge une norme de plugins ouverte : Web Audio Modules. La figure 2 illustre les similitudes et les différences entre les DAW en ligne commerciaux cités.

Dans le monde de l'open source, le choix est limité : GridSound est un DAW développé depuis 2015[2] qui prend en charge l'audio et le MIDI. Bien que les effets et instruments proposés demeurent simples, GridSound est un DAW entièrement fonctionnel. Il ne prend pas en charge de plugins. Il ne semble plus maintenu activement.

Il existe également des bibliothèques JavaScript open source populaires pour développer un lecteur/enregistreur multipiste, telles que wavesurfer.js[3] , peaks.js[4] ou waveformplaylist[5] , et des éditeurs de fichiers audio de type Audacity tels que AudioMass[6].

Aucune de ces initiatives open source n'utilise de traitement DSP personnalisé de bas niveau (pas d'AudioWorklets), ni ne se soucie d'optimiser la communication entre DAW/plugin, ni n'utilise de plugins externes. C'est la raison principale pour laquelle nous avons développé WAM Studio : en tant que démonstrateur de ces techniques.

5. WAM Studio, design et implémentation

Wam Studio est un outil en ligne pour créer des projets audio que l'on peut imaginer comme de la musique multipistes. Chaque piste correspond à une "couche" différente de contenu qui peut être enregistré, édité, joué ou simplement intégré (en utilisant des fichiers audio, par exemple). Certaines pistes peuvent être utilisées pour contrôler des instruments virtuels : dans ce cas, elles ne contiennent que les notes qui doivent être jouées, avec quelques métadonnées. Des pistes peuvent être ajoutées ou supprimées, jouées isolément ou avec d'autres pistes. Elles peuvent également être "armées" pour l'enregistrement, et lorsque l'enregistrement commence,

[2] https ://gridsound.com/. Voir également la présentation vidéo de GridSound à la conférence ADC 2021 :

https ://www.youtube.com/watch?v=ejTtENwRxnA

[3] . https ://wavesurfer-js.org/
[4] . https ://github.com/bbc/peaks.js/
[5] . https ://github.com/naomiaro/waveform-playlist
[6] . https ://audiomass.co/

toutes les autres pistes joueront simultanément, tandis que les pistes armées enregistreront un nouveau contenu.

5.1 Les pistes

Dans un DAW, chaque piste est un conteneur de données liées à l'audio qui s'accompagne d'une représentation interactive de ces données, de fonctionnalités d'édition et de traitement ainsi que de quelques paramètres par défaut tels que le volume et le panoramique gauche/droite de la piste.

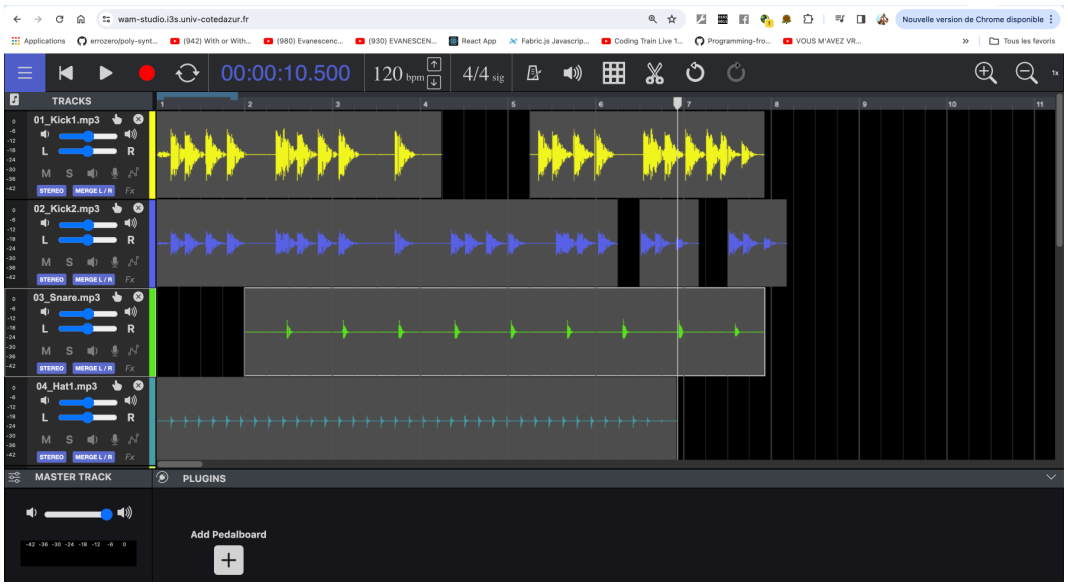

Figure 3 : Pistes et régions dans la GUI du DAW Wam Studio.
Les régions peuvent être déplacées, supprimées, etc.

La Figure 3 montre des pistes audio dans Wam Studio avec l'affichage de la forme d'onde des buffers audio associés et les contrôles/paramètres par défaut des pistes, sur le côté gauche (mute/solo, armement pour l'enregistrement, volume, panoramique stéréo, affichage des courbes d'automation, plugins d'effets). Comme de nombreuses pistes peuvent être affichées, scrollées pendant la lecture, zoomées et éditées, nous avons utilisé la bibliothèque pixi.js pour gérer de manière efficace le dessin et les interactions dans un canvas HTML. Cette bibliothèque utilise un rendu WebGL accéléré par GPU et offre de nombreuses fonctionnalités pour la gestion de plusieurs calques sur un seul canvas HTML5 (Figure 3). On voit sur la Figure 5 que

chaque piste peut également être associée à un ensemble de plugins pour ajouter des effets audio ou générer de la musique (dans le cas de plugins instrumentaux).

La Figure 4 montre le graphe audio correspondant à la chaîne de traitement d'une piste audio. Le son va de gauche à droite : d'abord le "lecteur/enregistreur/éditeur de piste" est implémenté en tant que nœud AudioWorklet, en utilisant du code personnalisé pour le rendu d'un buffer audio, puis le signal de sortie a son gain et son panoramique stéréo ajustés, ensuite nous avons un autre nœud AudioWorklet pour le rendu du volume dans un canvas (VU-mètre), et enfin nous trouvons une chaîne de plugins WAM pour ajouter des effets audio.

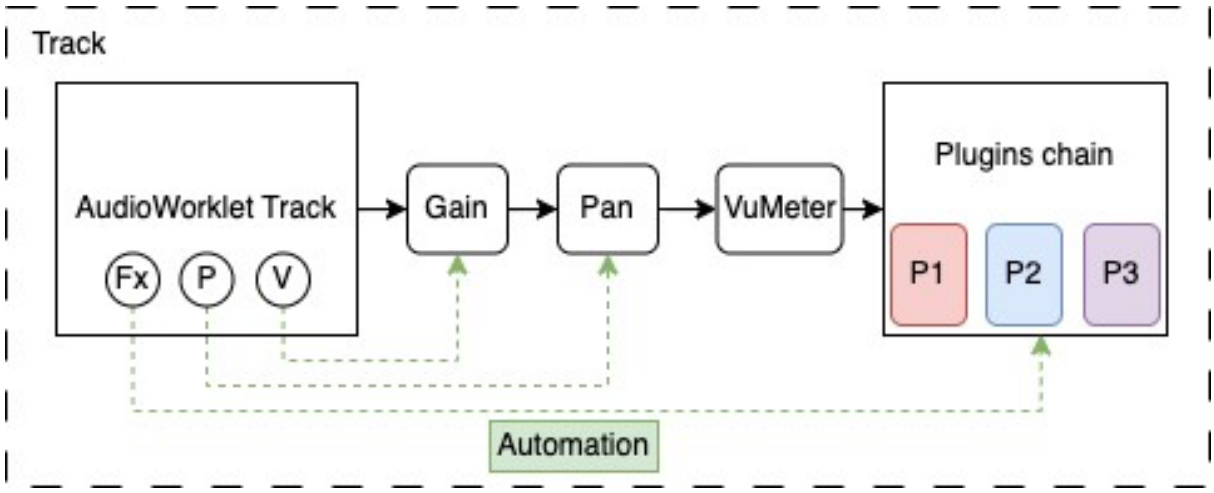

Figure 4 : Graphe audio d'une piste dans Wam Studio

Il existe deux types de pistes : les pistes "audio" : elles contiennent de l'audio enregistré, tel qu'une prise vocale, un enregistrement de guitare ou tout autre type de signal sonore, qui sont généralement rendus graphiquement avec une forme d'onde représentant le signal stocké (sous forme d'un ensemble d'échantillons sonores). Ces pistes peuvent être éditées, traitées et mélangées en copiant/coupant et collant des échantillons dans le buffer audio associé à la piste.

Le signal de sortie des pistes audio peut être traité par une chaîne de plugins d'effet audio. Les pistes "MIDI" : elles contiennent des données MIDI (le pitch - la note qu'un instrument jouerait, sa vélocité et sa durée). Les données MIDI ne contiennent pas d'informations audio mais sont plutôt utilisées pour contrôler des instruments virtuels tels que des synthétiseurs, des samplers, des boîtes à rythmes, etc. Les pistes MIDI peuvent être éditées, auquel cas il est possible de modifier les événements MIDI stockés dans la piste dans un affichage de type matriciel. Les pistes MIDI sont généralement associées à un ou plusieurs plugins WAM d'instruments

virtuels. Les événements MIDI peuvent cibler tous les plugins d'instruments de la piste ou certains en particulier.

Les pistes sont généralement organisées verticalement dans l'interface utilisateur et peuvent être gérées séparément en termes de volume, de panning stéréo, de plugins et d'automatisation des paramètres de ces plugins d'effets ou d'instruments (comme le montre la Figure 5).

Figure 5: Une chaîne de plugins associés à la piste sélectionnée.

Chaque sortie de piste est connectée à une "piste maître" (master track) où il est possible d'ajuster globalement le volume et le panning stéréo du mix final. Comme pour toutes les autres pistes, il est également possible d'associer une chaîne d'effets audio à la piste maître (par exemple pour ajuster la dynamique et les fréquences finales). Tous les effets audio et instruments virtuels sont des plugins WAM dans le cas de notre DAW, et tous leurs paramètres sont automatisables[7] . Cette conception offre un degré de contrôle et d'adaptabilité étendu et permet aux utilisateurs de mélanger et de manipuler le son de chaque piste avec une grande

[7] . A la fréquence d'échantillonnage (a-rate), du quantum audio (k-rate), personnalisée : cela dépend de l'implémentation du plugin au format WAM

précision et sophistication, rendant ainsi plus facile la création de productions audio complexes, le tout sans quitter le navigateur Web. Wam Studio est le seul logiciel open source à proposer l'ensemble de ces fonctionnalités.

Une piste DAW est également capable de "rendre" la piste en un signal audio audible (lorsque la piste est en lecture). Lorsque l'on appuie sur le bouton de lecture du DAW, toutes les pistes sont rendues simultanément, et le signal de sortie final est ce que l'on appelle "le mix". Les DAWs disposent également d'une option de "rendu hors ligne" pour exporter le mix final ou des pistes individuelles sous forme de fichier(s), sur le disque dur local ou sur le cloud (en utilisant l'OfflineAudioContext [8]fourni par l'API Web Audio).

5.2 Le cœur des pistes est un AudioWorklet Processor

Concentrons-nous pour le moment sur une piste de type audio. L'API Web Audio fournit le nœud AudioBufferSourceNode qui est un conteneur pour un buffer audio, et qui permet la lecture d'un buffer audio stocké en mémoire. Cette approche simple à mettre en œuvre est néanmoins non adaptée pour un DAW devant gérer l'automation de paramètres. Cette tâche consiste à interpoler, pendant la lecture ou l'enregistrement multipiste, les paramètres des plugins et des pistes à de hautes fréquences (à la fréquence d'échantillonnage *a-rate* ou de contrôle *k-rate*). Un paramètre automatisé peut être celui de n'importe quel plugin associé à n'importe quelle piste. Il peut y avoir potentiellement des dizaines, des centaines de paramètres à automatiser. Cela représente une quantité substantielle d'informations qui peut être échangée entre le DAW et les plugins WAM, ce qui fait de la performance un aspect crucial qui nécessite une attention particulière.

En concevant notre propre lecteur audio de bas niveau (au lieu d'utiliser le nœud standard AudioBufferSourceNode), nous gagnons un contrôle total sur le comportement de lecture/enregistrement de chaque piste. Nous avons ainsi conçu le noyau de chaque piste audio comme un "AudioWorklet processor", qui permet l'exécution de code DSP personnalisé dans le thread audio à très haute priorité. Plus précisément, ce noyau est une instance d'une sous-classe de la classe WamProcessor fournie par le SDK des Web Audio Modules, qui hérite de la classe AudioWorkletProcessor. Selon l'API, chaque processeur implémente une méthode process(input, output) qui sera appelée à la fréquence d'échantillonnage. À l'intérieur de cette

[8] . Contrairement à un AudioContext standard, qui sert à créer les noeuds du graphe audio pour un rendu "live", un OfflineAudioContext sert lui, à générer aussi rapidement que possible un graphe audio et à envoyer le résultat dans un AudioBuffer, qui peut ensuite être converti en format WAVE par exemple.

méthode, nous traitons les échantillons sonores et procédons au rendu du buffer audio, mais nous pouvons également envoyer des données aux plugins WAM. La classe WAMProcessor dont nous héritons fournit en effet des méthodes pour gérer efficacement la communication entre le DAW et les plugins.

Avec l'API Web Audio, les AudioWorklets sont composés de deux composants, reflétant la nature multi-threadée de l'environnement : le nœud AudioWorkletNode et l'AudioWorkletProcessor. Le standard WAM les étend avec les classes WamNode et WamProcessor, qui s'exécutent respectivement dans le main thread du navigateur (thread de la GUI) et dans le thread audio. Wam Studio est une application hôte WAM, et ses pistes sont donc mises en œuvre en tant que sous-classes de WamProcessor.

Les plugins WAM associés à ces pistes peuvent également avoir leur cœur DSP implémenté en tant que sous-classes de WamNode avec un WamProcessor, mais ce n'est pas obligatoire car certaines implémentations de plugins WAM utilisent plutôt un graphe composé d'une combinaison de nœuds, dans ce cas ils étendent la classe WAM CompositeNode.

Quelle que soit l'implémentation des plugins le code de communication de l'hôte vers le plugin est le même, et les optimisations mise en œuvre se font en coulisse (dans le WAM SDK). Cette abstraction permet une interopérabilité transparente entre les hôtes et les plugins WAM, indépendamment de la mise en œuvre sous-jacente et sur tous les threads. En d'autres termes : *si une piste est un WamProcessor et que les plugins sont également des WamProcessor, leur code DSP s'exécute dans le thread audio et une communication hautement optimisée peut être réalisée* : de la mémoire partagée (Shared Array Buffers) peut être utilisée pour la communication entre DAW et plugins. Si un plugin n'est pas un WamProcessor, alors le code DAW qui "communique" avec lui reste inchangé et la mise en œuvre sous-jacente sera moins optimale et impliquera le franchissement de la barrière des threads.

Pour gérer plusieurs plugins en chaîne, le standard WAM utilise un objet singleton, appelé WamEnv, qui est attaché à la portée globale du thread audio et sert de médiateur pour les interactions entre les hôtes et les plugins.

Les plugins sont structurés au sein d'un « WamGroup » et regroupés dans un "WamEnv", qui encapsule le code de la version du SDK WAM utilisée. Si une nouvelle version du SDK sort, il sera ainsi possible de charger les plugins dans un environnement WamEnv protégé, chaque plugin utilisant sa propre version du SDK.

Chaque groupe de plugins (WamGroup) comprend les plugins créés par un hôte spécifique et il est possible d'envoyer des événements (MIDI par exemple) à tous les plugins à la chaîne.

Un effort considérable a été déployé pour réduire le nombre d'hypothèses faites par l'API WAM concernant la mise en œuvre de l'hôte, avec WamEnv et les instances de WamGroup étant les seuls objets du système WAM qui sont fournis par l'hôte. Le WamEnv est alloué lors de la création du DAW et les instances de WamGroup sont associées à chaque piste.

5.3 Gestion de la chaîne des plugins

Les chaînes de plugins sont gérées à l'aide d'un plugin WAM spécial qui agit également en tant que "mini-hôte" (Figure7). Nous l'avons appelé le WAM pedalboard (Buffa et al., 2022). Il se connecte à un ou plusieurs serveurs de plugins qui renvoie(nt) la liste des plugins disponibles sous forme de tableau JSON d'URIs (un plugin WAM peut être chargé simplement à l'aide d'une importation dynamique et de son URI, voir (Buffa, Ren et al., 2022). À partir de cette liste d'URIs, les descripteurs de plugins WAM sont récupérés, qui contiennent des métadonnées sur les plugins : nom, version, fournisseur, URI d'image miniature, etc.

Lorsque le plugin pedalboard est affiché dans le DAW, la chaîne de plugins actifs est vide, et les plugins peuvent être ajoutés à la chaîne de traitement, supprimés, réordonnés et leurs paramètres peuvent être définis.

Figure 6 : Le plugin WAM Pedalboard qui gère les chaînes de plugins associés à chaque piste. Ici chargé dans une simple page HTML hôte, en mode standalone.

Toute configuration peut être enregistrée en tant que preset nommé (par exemple, "son de guitare crunch 1"). Les presets peuvent être organisés en banques de sons ("rock", "funk",

"blues"). La gestion de l'organisation et de la nomination des banques et des presets relève de la responsabilité du pedalboard plugin. Les paramètres exposés par ce plugin correspondent à l'ensemble des paramètres du preset actif (c'est-à-dire la somme des paramètres des plugins du preset), et peuvent être automatisés par le DAW.

L'API Web Audio Modules permet d'envoyer des événements au DAW lorsque des modifications se produisent dans la configuration du plugin (ajout ou suppression), et un menu fourni dans le DAW avec chaque piste pour sélectionner les paramètres pouvant être automatisés est automatiquement mis à jour (Figure 7).

Figure 7 : Chaque piste a un menu d'automation qui se met à jour en fonction des plugins associés dans le preset courant.

5.4 Enregistrement, gestion de la latence

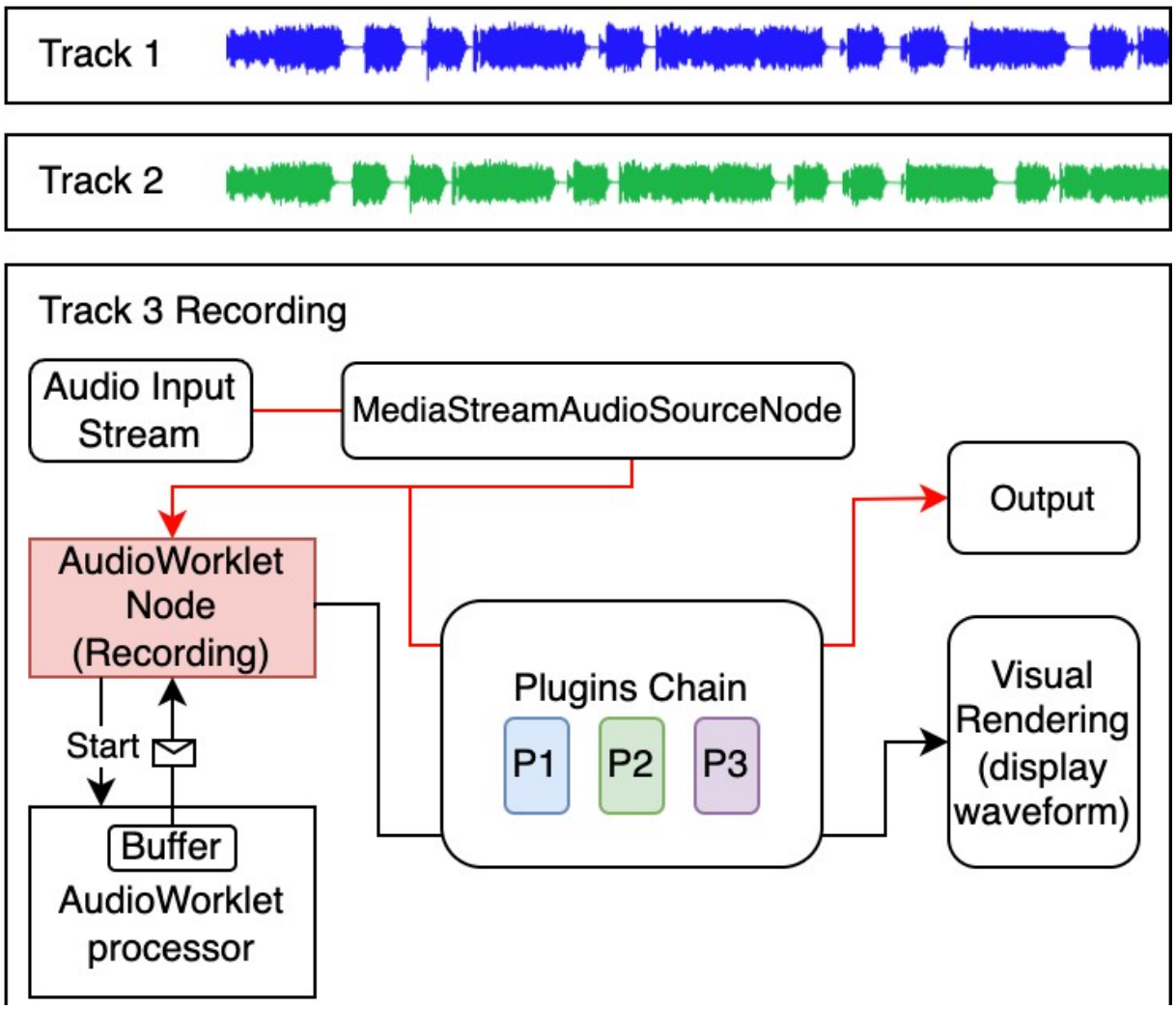

Figure 8 : Schéma logique de l'enregistrement audio dans une piste de Wam Studio.

La figure 8 montre le graphe audio correspondant au processus d'enregistrement d'une piste : nous devons d'abord obtenir un flux multimédia à partir du microphone de l'utilisateur ou d'une entrée de carte son à l'aide de l'API MediaDevices. Ensuite, nous pouvons utiliser un MediaStreamSourceNode de l'API Web Audio dans le graphe audio, en passant le flux

multimédia à son constructeur. C'est comme cela que nous exposons un flux audio live à l'API Web Audio.

Dans un deuxième temps, nous avons besoin d'un moyen d'enregistrer ce flux dans un buffer. Cela peut être fait en utilisant l'API W3C MediaRecorder ou en utilisant des solutions de bas niveau à l'aide d'un AudioWorklet.

La solution MediaRecorder a pour avantage sa simplicité, mais tous les tests effectués ont montré que c'est une solution peu fiable car le temps nécessaire pour allouer des buffers audio et commencer l'enregistrement est (voir plus loin).

Nous avons finalement opté pour une solution de bas niveau plus précise qui utilise un WamProcessor dont le rôle est d'enregistrer en continu les échantillons sonores dans un buffer tournant contenant à peu près une seconde de son numérisé. Ce buffer est situé dans une mémoire partagée avec un Worker JavaScript qui lui tourne dans un thread moins prioritaire et dont le rôle est de consommer les échantillons envoyés par le WamProcessor et de les copier à la fin d'un buffer qui grossit au fur et à mesure. On a une conception classique producteur / consommateur si ce n'est que le producteur est exécuté dans le thread audio (et donc non interruptible, à haute priorité) et que le consommateur est lui dans le thread de l'interface graphique, moins prioritaire.

Cette approche évite des allocations dans le thread audio, elle et également plus robuste contre un éventuel stress CPU. Le Worker envoie également sous forme de message le buffer audio courant à un code JavaScript exécuté dans un autre thread de la GUI. Ce dernier peut ainsi dessiner la forme d'onde du signal audio enregistré qui grandit visuellement au fur et à mesure que l'enregistrement avance. Lorsqu'on stoppe l'enregistrement, la piste est prête à être jouée puisque le buffer audio a été mis à jour au fur et à mesure.

Compensation de latence : le processus d'enregistrement avec un DAW est plus contraint et structuré que dans un simple enregistreur de mémos audio, car nous devons prendre en compte différents types de latences.

- *La latence d'entrée* est le temps entre la capture d'un signal audio par le dispositif d'entrée (carte son) et son traitement par le contexte audio. Elle dépend du système d'exploitation, de la configuration du dispositif d'entrée, du temps de traitement du système audio et de la taille du buffer utilisé par le contexte audio.
- *La latence de sortie* représente le temps entre la génération d'un son et sa perception par l'utilisateur.

- La somme de ces deux latences représente *la latence round-trip* et peut être mesurée à l'aide d'un dispositif d'enregistrement externe avec deux microphones : un sur la source du son physique (par exemple, sur le corps d'une guitare branchée sur une carte son), et un devant les haut-parleurs. Frappez le corps de la guitare, enregistrez la sortie et comparez les signaux d'entrée et de sortie. Nous avons effectué de telles mesures dans le passé (Buffa, Lebrun, 2017). On peut aussi l'estimer avec un programme émettant des sons (en général des sons de type métronome) et en enregistrant ces mêmes sons en plaçant un micro devant les haut-parleurs.

Paul Adenot propose une implémentation d'un outil permettant de mesurer cette latence et aussi la latence de sortie [9] . Le résultat est en général moins précis que lors de la mesure avec des outils externes.

Par ailleurs, l'API Web Audio expose une propriété outputLatency de l'AudioContext dont la valeur est une approximation de la latence de sortie, soit le nombre de secondes entre l'audio atteignant l'AudioDestinationNode (conceptuellement, la sortie du graphe de traitement audio) et le dispositif de sortie audio.

Si on sait mesurer la latence round-trip, alors on peut connaître la latence d'entrée puisque :

latence d'entrée + latence de sortie (connue) = latence round-trip.

Donc :

Latence d'entrée = latence round-trip – latence de sortie (connue)

Lors de l'enregistrement d'une piste de guitare, par exemple, d'autres pistes telles que des pistes de batterie et de basse peuvent être jouées simultanément. Le son de la guitare est traité en temps réel par une chaîne de plugins (simulation d'amplificateur de guitare, égaliseur, retard, etc.) et doit être entendu en temps réel par le guitariste, en synchronisation avec les autres pistes et sans latence perceptible lorsqu'il est joué (et enregistré).

C'est pourquoi la latence round-trip globale doit être aussi faible que possible !

[9] . Voir le billet de blog de Paul Adenot https://blog.paul.cx/post/audio-video-synchronization-with-the-web-audio-api/

La route permettant de s'entendre jouer pendant l'enregistrement est représentée dans la Figure 8 avec le chemin rouge : une partie du signal traverse la chaîne de plugins et peut être entendue (*wet route*) et le signal non traité est enregistré (*dry route*, vers l'AudioWorklet). En utilisant une taille de buffer de 128 échantillons (valeur par défaut lors de la création d'un AudioContext), la latence round-trip mesurée sur MacOS est de 23 ms avec le navigateur Chrome, 15ms avec Firefox, suffisamment confortable et comparable à ce que nous pouvons obtenir avec des applications natives dans des cas d'utilisation similaires (guitare, Logic Audio, plugin de simulation d'amplificateur de guitare, taille de buffer audio de 128), même avec des guitaristes jouant très vite (Buffa, Lebrun, 2017).

Néanmoins, lors de la lecture simultanée de la nouvelle piste enregistrée, un décalage de plusieurs millisecondes est visible et audible, la piste nouvellement enregistrée est "en retard" par rapport aux autres. Il est nécessaire de décaler en arrière dans le temps la nouvelle piste. Cette opération s'appelle la *compensation de latence*, et la valeur de compensation est égale à la *valeur de la latence d'entrée*.

Nous avons vu que cette valeur peut être calculée si on connaît la latence round-trip (puis que la latence de sortie est fournie par l'AudioContext).

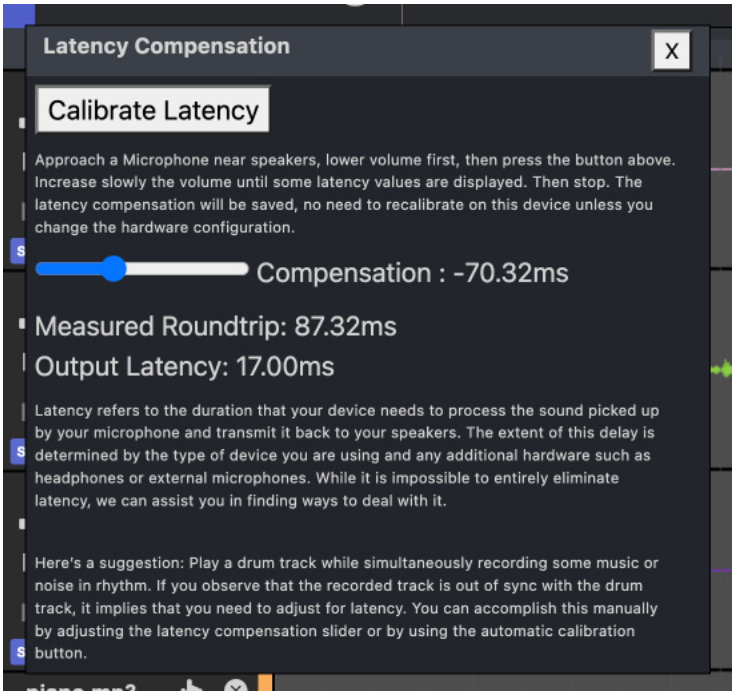

Figure 9 : Réglage automatique de la compensation de latence.

Dans WAM Studio, une option de configuration permet de calculer automatiquement la compensation de latence en approchant un micro des haut-parleurs et en lançant le processus de calibration (le DAW émet des sons, enregistre le résultat et compare les temps d'émission et de réception). Pour une configuration matérielle donnée cette opération n'est à effectuer qu'une seule fois (voir Figure 9 et aussi vidéo de démonstration[10]).

Le DAW AmpedStudio propose une approche similaire. Les auteurs de SoundTrap ont expliqué qu'ils utilisent plutôt des tables prédéfinies avec des entrées pour les paires OS/cartes son les plus courantes, dont la latence d'entrée a été mesurée à la main et codées en dur dans le code (Lind, McPherson, 2017). SoundTrap utilise ensuite des heuristiques pour déterminer la configuration (vérification de l'OS, deviner le modèle de la carte son en regardant le nombre d'entrées/sorties exposées en utilisant l'API MediaDevices, etc.).

D'autres optimisations classiques ont été mises en place dans Wam Studio : dès que les pistes sont armées pour l'enregistrement, on alloue le buffer tournant, on démarre les Workers (tâches de fond JavaScript) d'enregistrement, on pré-alloue les buffers audio et on ne commence à enregistrer que lors de l'appui sur les boutons "record" puis "play", à ce moment-là, l'enregistrement à proprement parler commence. Ceci évite de perdre du temps au démarrage, avec les allocations et le lancement des Workers.

Tous les plugins WAM implémentent également des méthodes getState/setState pour sérialiser/désérialiser leur état. Lorsqu'un projet est enregistré, l'état de chaque piste, des buffers audio, etc. est enregistré, ainsi que l'état de la configuration de chaque plugin.

5.4 Autres fonctionnalités : bouncing des pistes, rendu final du mix, boucle audio

Comme le montre la Figure 10, Wam Studio permet d'exporter le mix final (*master track*) ou chaque piste individuellement (ou les deux à la fois). L'utilisation d'un OfflineAudioContext est ici utilisée pour que le temps de rendering soit le plus rapide possible. Les chaînes d'effets et l'automation de leurs paramètres sont également pris en compte. Les formats possibles sont .wav ou .mp3.

10 https://www.youtube.com/watch?v=CAAWoTjjlB0

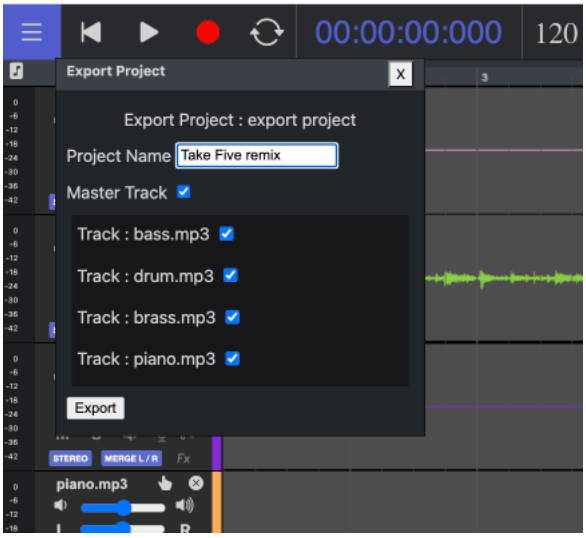

Figure 10 : Rendu du mix final ou de pistes individuelles.

Un navigateur de boucles audio a été récemment intégré et utilise une extension proposée par les Web Audio Modules : la gestion des assets par le logiciel hôte (Figure 11).

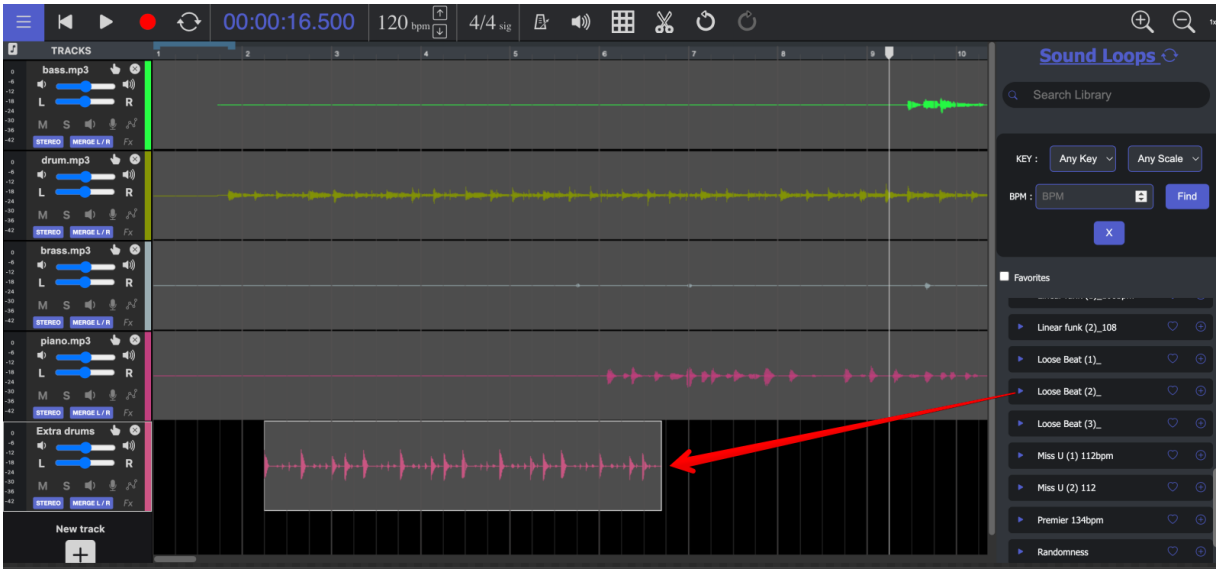

Figure 11: explorateur de boucles audio (sur la droite).

Avec l'extension « Asset », les hôtes WAM contrôlent le chargement et l'enregistrement des assets telles que des boucles audio ou des échantillons pour un plugin de type sampler. Les plugins WAM individuels peuvent s'appuyer sur l'hôte pour le stockage dans le cloud, et les utilisateurs peuvent gérer tous les fichiers liés à un projet musical dans l'hôte sans que chaque plugin WAM ne stocke les fichiers dans des services ou des comptes cloud distincts (une des craintes exprimée par les utilisateurs de Web Audio Modules, lors de tests utilisateurs, a été de voir leurs assets éparpillés sur plusieurs services cloud non contrôlés). L'explorateur de boucles audio propose une interface classique vue dans les DAWs natifs, avec des filtres par instrument, tempo, tonalité. Il suffit de drag'n'dropper des boucles sur une pistes pour ajouter une région audio à la piste. On peut aussi déposer une boucle sous la dernière piste et cela créera une nouvelle piste. A noter que le drag'n'drop de fichiers externes est également supporté, avec le même comportement (région ou nouvelle piste crée).

6. Attune : une version spéciale de Wam Studio pour aider les utilisateurs d'implants cochléaires

6.1 Contexte

Suite à la publication open source de Wam Studio, les auteurs ont été contacté par des chercheurs de l'équipe MERI du laboratoire CCRMA de Stanford travaillant sur des solutions pour améliorer l'écoute de musique par des personnes équipées d'implants cochléaire.

Ces chercheurs ont demandé aux auteurs de cet article d'adapter, dans le cadre d'une collaboration scientifique, le logiciel Wam Studio à leurs besoins spécifiques. Ainsi est née Attune[11], une version spéciale de Wam Studio permettant aux utilisateurs de créer visuellement des macros pour contrôler plusieurs paramètres de plugins et de pistes à la fois, et offrant une interface graphique asymétrique avec deux modes d'utilisation, selon que ce sont les chercheurs de l'équipe MERI qui l'utilisent, ou bien des personnes malentendantes.

11 Version en ligne : https://attune.i3s.univ-cotedazur.fr/, le code source est disponible dans la branche Stanford-prototype du repository GitHub de Wam Studio : https://github.com/Brotherta/Wam Studio

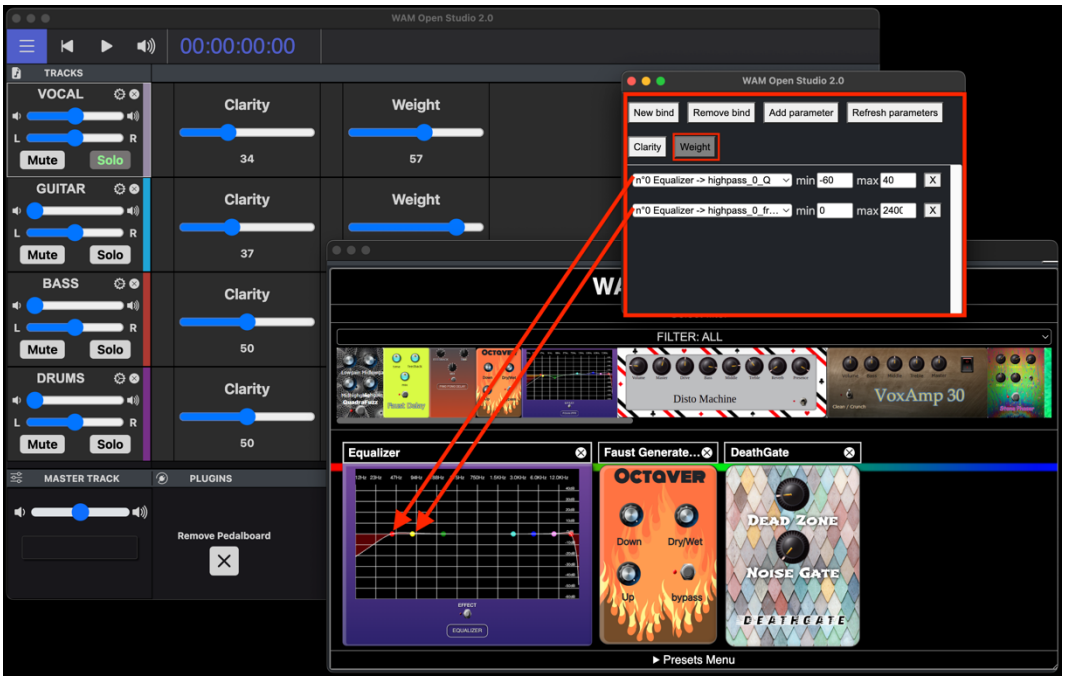

Figure 12 : Vue de l'éditeur de Macros de Attune

Dans le premier cas, c'est l'interface graphique de WAM studio qui est proposée, avec en plus des plugins spécifiquement développés pour ce projet ainsi que le gestionnaire de macros permettant de faire le mapping entre un nombre variable de paramètres de piste et de plugins et un simple contrôleur de type slider ou bouton rotatif (Figure 12).

Dans le second cas, c'est une interface graphique extrêmement simplifiée qui est proposée (Figure 13) : quatre pistes audio et deux réglages simples par piste pour que les personnes malentendantes adaptent le rendu audio de chaque piste (voix, batterie, etc.). Les réglages ont pour nom « clarté », « punch », etc. Des réglages de même nom peuvent en réalité correspondre à des paramètres de plugins très différents selon qu'ils figurent sur une piste de voix, de piano ou de batterie. Par exemple, on peut assigner la coupure d'un filtre, la résonance et la quantité d'enveloppe d'un synthé à une macro, de sorte qu'il est possible d'ajuster tous ces paramètres en même temps à l'aide d'un seul bouton appelé "timbre".

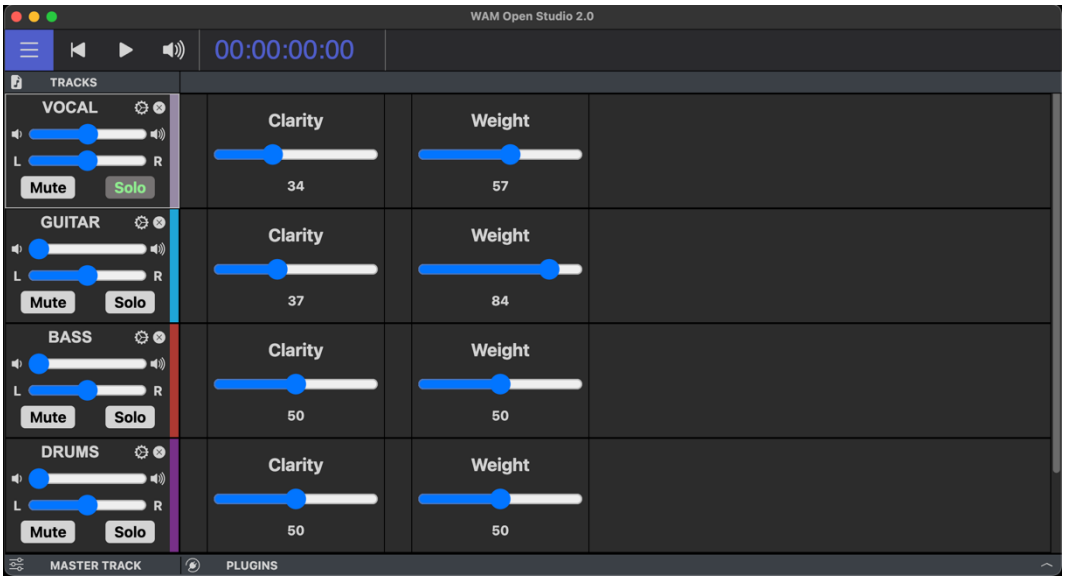

Figure 13 : Vue simplifiée de Attune, pour personnes malentendantes.

Les macros de Wam Studio/Attune sont inspirées du système disponible dans le logiciel Ableton Live[12]. Elles peuvent être créées, chargées et sauvegardées dynamiquement dans le gestionnaire de macros de Wam Studio, y compris en tant que presets, ce qui permet de rappeler rapidement des paramètres personnalisés et de les réutiliser dans des projets ultérieurs. Lorsqu'un projet est sauvegardé, l'état actuel et la configuration des macros du projet sont également sauvegardés. Une vidéo de démonstration est disponible, montrant la création de Macros et les deux modes d'utilisation[13].

6.2 Détails des expériences menées à l'aide de Attune

Plus d'une personne sur cinq dans le monde, soit environ 430 millions de personnes, sont sourdes ou malentendantes et nombre d'entre elles voient leur qualité de vie s'améliorer considérablement grâce à l'utilisation de technologies d'assistance, telles que les aides auditives

12 https://www.youtube.com/watch?v=NOufylMAEA&t = 177s

13 https://youtu.be/m52sv_5KLMk

ou les implants cochléaires (ICs) (WorldHealthOrganization, 2023). En anglais on utilise le terme DHH[14] pour nommer ces personnes (DHH pour *D/deaf or Heard-of-Hearing*).

Les implants cochléaires sont des dispositifs électroniques qui convertissent les signaux sonores acoustiques en signaux électriques utilisés pour stimuler la cochlée. Actuellement, le traitement audio interne des IC est optimisé pour la parole, ce qui fait que les expériences perceptives de la musique et d'autres stimuli auditifs complexes varient considérablement d'un utilisateur d'IC à l'autre (Spangmose et al., 2019). Les utilisateurs d'IC perçoivent une plage dynamique maximale réduite de ~40-80 dB et une résolution de fréquence environ 10-20 fois inférieure à celle des personnes ayant une audition traditionnelle (Hartmann, Kral, 2015) (Zeng et al., 2014).

Par conséquent, les utilisateurs d'IC perçoivent certaines caractéristiques musicales, telles que le rythme et le tempo, de la même manière que les personnes ayant une audition traditionnelle; cependant, la perception des informations timbrales, harmoniques et mélodiques diffère grandement (Limb, Roy, 2014).

Les chercheurs de l'équipe MERI du CCRMA ont effectué des séries de tests avec les utilisateurs d'ICs, qui ont transmis leurs commentaires et évalué divers mixages à l'aide d'un mélange de mesures qualitatives et quantitatives, ce qui a permis de déterminer les stratégies de mixage et les combinaisons de plugins les plus pertinentes (ce sont celles qui ont été bien notées). Les résultats ont montré que 'utilisation des macro-commandes de Attune pour ajuster certains paramètres sonores peut accroître le confort d'écoute de musique enregistrée pour les utilisateurs d'ICs (Buffa et al., 2023).

Afin de faciliter l'utilisation de l'application d'écoute, Attune possède en réalité deux modes d'utilisation. Dans le mode spécial pour l'écoute audio par des personnes DSS, nous avons créé une vue simplifiée des pistes et macro-commandes tout en cachant les options inutiles présentes originellement dans la version complète du logiciel (cette dernière est très similaire à celle de Wam Studio), comme le montre la Figure 12. La simplification majeure consiste à remplacer les forme d'onde des buffers audio associée aux pistes par les curseurs de macro-commande correspondants. Ces macros ont été créées à partir de tests antérieurs effectués par l'équipe MERI du CCRMA et reçoivent des étiquettes non techniques, telles que "clarté", "punch" ou "poids", permettant aux utilisateurs d'ajuster finement plusieurs paramètres d'une chaîne de plug-ins ou de la piste à l'aide d'un seul curseur. Elles sont prévues pour être utilisées avec des

14 DHH est un terme générique désignant les personnes malentendantes ou présentant une déficience auditive, y compris celles qui s'identifient comme culturellement sourdes et peuvent utiliser une langue des signes comme langue primaire.

chansons de test multipistes (typiquement quatre/cinq pistes : voix, batterie, basse, piano, autre) et avec différents genres de musique enregistrée. (pop, rock, country, jazz, etc.).

Voici quelques exemples d'implémentation de macros utilisées pendant ces tests :

- Clarté (Pop, Vocal) : L'augmentation de cette macro augmente la quantité de 2kHz et 5kHz présents dans le signal, augmente le mélange humide/sec d'un dé-esseur et ajoute un compresseur à attaque moyenne et à release lent.
- Punch (Rock, Batterie) : Augmente le taux de compression et le mixage wet/dry d'un compresseur avec une attaque et un release moyennement rapides, augmente le mixage wet/dry d'un pitch-shifter de sous-octave, ainsi que le mixage wet/dry d'un plug-in de saturation.
- Brillance (Country, Guitare) : Réduit la quantité d'informations sub-250 Hz, augmente le mixage wet/dry sur un pitch-shifter d'une octave au-dessus, et augmente le ratio et le mixage wet/dry sur un compresseur à attaque lente et release lent.

Plusieurs plugins ont été développés spécifiquement pour les utilisateurs DHH, comme un octaver et un plugin de tracking-EQ qui détecte et augmente la fréquence fondamentale du signal.

L''utilisation d'un DAW personnalisable avec interface graphique asymétrique pour permettre aux utilisateurs DHH d'améliorer l'écoute multipiste est une contribution originale dans ce domaine. Les approches actuelles visant à améliorer l'appréciation de la musique par les utilisateurs d'ICs comprennent majoritairement des ajustements du traitement interne du signal sur l'IC lui-même, ou bien la création de musique composée spécifiquement pour les utilisateurs d'IC, ou encore du prétraitement algorithmique des contenus audio (Nogueira et al., 2018).

Ces approches ont certainement des mérites, mais elles ne reconnaissent pas la grande diversité et la variance de la perception auditive et des préférences esthétiques parmi les utilisateurs d'ICs. En outre, ces processus supposent un auditeur d'IC passif avec un désir limité de jouer un rôle actif dans son expérience d'écoute.

Pour conclure, Wam Studio/Attune est aujourd'hui la seule station de travail audionumérique basée web disposant d'un système de macros, qui est open source et qui prend en charge les plugins de tiers (n'importe quel plugin au format WAM peut être utilisé).

7. Conclusion

En 2022, juste après la crise du COVID qui a vu les services collaboratifs gagner en popularité, Stickland a publié une comparaison entre un DAW en ligne (SoundTrap) et deux solutions natives (Avid Cloud Collaboration et VST Transit). Il a montré que les DAWs en ligne "*ont le potentiel d'être largement adoptés et pourraient même dépasser l'utilisation des paradigmes existants dans la pratique professionnelle du mixage audio à l'avenir*" (Stickland et al., 2022).

Quatre DAWs commerciaux existent, mais à ce jour, Wam Studio le seul DAW open source utilisant des fonctionnalités avancées telles que la prise en charge de plugins externes, des communications DAW/plugins à haute performance, la lecture, l'enregistrement et l'édition de pistes audio avec compensation de latence, le support de plugins externes avec automation des paramètres et un système de macros permettant de contrôler plusieurs paramètres simultanément. En tant que démonstrateur, il montre les capacités de l'API Web Audio et comment le standard Web Audio Modules peut être utilisé pour faciliter le développement d'applications audio basées web complexes.

La prise en charge des pistes MIDI est en cours de développement et devraient intégrer la branche principale du code source à court terme [15]. Notons qu'en 2023, les mises à jours de Wam Studio sont régulières.

Par ailleurs, les Web Audio Modules, le standard de plugin/host sur lequel est construit Wam-Studio, voit son écosystème s'agrandir chaque semaine avec de nouveaux outils de développement, de nouveaux plugins et hosts proposés en open source [Buffa et al., 2024]. Le code source du DAW peut être trouvé sur GitHub [16] et l'application est également disponible en ligne [17].

[15] . Nous avons écrit des démonstrations de lecture de pistes MIDI à l'aide de WAMs sur le site de tutoriels WAM : https://wamexamples.vidalmazuy.fr/

[16] . http://github.com/Brotherta/Wam Studio

[17] . https://Wam Studio.i3s.univ-cotedazur.fr/, la version Attune est disponible sur https://attune.i3s.univ-cotedazur.fr/

8. References

(Buffa et al., 2024)

Michel Buffa, Shihong Ren, Tom Burns, Antoine Vidal-Mazuy, Stéphane Letz. *Evolution of the Web Audio Module Ecosystem*. Submitted to the Web Audio Conference 2024, Purdue University, Lafayette, USA, 2024.

(Buffa et al., 2023)

Buffa M, Vidal-Mazuy A, May L, Winckler M. Wam Studio: *A Web-Based Digital Audio Workstation to Empower Cochlear Implant Users*. In IFIP Conference on Human-Computer Interaction 2023 Aug 25 (pp. 101-110). Cham: Springer Nature Switzerland.

(Buffa, Vidal-Mazuy 2023)

M. Buffa, and A. Vidal-Mazuy. *WAM-studio, a Digital Audio Workstation (DAW) for the Web*. In Companion Proceedings of the ACM Web Conference 2023, pp. 543-548. 2023.

(Buffa et al. 2022)

Michel Buffa, Pierre Kouyoumdjian, Quentin Beauchet, Yann Forner, and Michael Marynowic. *Making a guitar rack plugin -WebAudio Modules 2.0*. In Web Audio Conference 2022., Cannes, France, 2022. https://doi.org/10.5281/zenodo.6769098.

(Buffa, Ren and al. 2022)

Michel Buffa, Shihong Ren, Owen Campbell, Jari Kleimola, Oliver Larkin, and Tom Burns. 2022. *Web Audio Modules 2.0 : An Open Web Audio Plugin Standard*. *In WWW '22 : The ACM* Web *Conference 2022*. ACM, Virtual Event, France, 364–369. https ://doi.org/10.1145/3487553.3524225

(Buffa et al., 2020)

Michel Buffa, Jerome Lebrun, Shihong Ren, Stéphane Letz, Yann Orlarey, Romain Michon, and Dominique Fober. *Emerging W3C APIs opened up commercial opportunities for computer music applications*. In The Web Conference 2020 DevTrack. Taipei, Taiwan. https ://doi.org/10.13140/RG.2.2.16456.19202

(Buffa et al., 2018)

Michel Buffa, Jerome Lebrun, Jari Kleimola, Oliver Larkin, and Stephane Letz. 2018. *Towards an open Web Audio plugin standard*. In WWW2018 – TheWebConf 2018 : The Web Conference, 27th International World Wide Web Conference. Lyon, France. https://doi.org/10.1145/3184558.3188737

(Buffa, Lebrun, 2017)

Michel Buffa and Jerome Lebrun. 2017. Real time tube guitar amplifier simulation using WebAudio. In Proceedings of the Web Audio Conference (Queen Mary Research Online (QMRO) repository). http://qmro.qmul.ac.uk/xmlui/handle/123456789/26089). Queen Mary University of London, London, United Kingdom. https ://hal.univ-cotedazur.fr/hal-01589229

(Choi, 2018)

Hongchan Choi. 2018. *Audioworklet : the Future of Web Audio*. In Proceedings of the International Computer Music Conference. Daegu, South Korea, 110–116.

(Déchelle et al., 1999)

François Déchelle, Riccardo Borghesi, Maurizio De Cecco, Enzo Maggi, Butch Rovan, and Norbert Schnell. *jMax : an environment for real-time musical applications*. In Computer Music Journal 23, 3, 50–58. 1999.

(Kleimola, Larkin, 2015)

Jari Kleimola and Oliver Larkin. Web Audio Modules. In Proceedings of the Sound and Music Computing Conference 2015.

(Lazarevic, Scepanovic, 2010)

Bojan Lazarevic and Danijela Scepanovic. *Unrevealed Potential in Delivering Distance Courses: the Instructional Value of Audio.* In eLearning and Software for Education (2010).

(Limb, Roy, 2014)

Limb, C.J., Roy, A.T.: Technological, biological, and acoustical constraints to music perception in cochlear implant users. Hearing research 308, 13–26 (2014)

(Lind, MacPherson, 2017)

Fredrik Lind and Andrew MacPherson. Soundtrap: *A collaborative music studio with Web Audio*. In Proceedings of the Web Audio Conference 2017. Queen Mary University of London, London, United Kingdom.

(McCartney, 2002)

James McCartney. *Rethinking the computer music language: Super collider*. In Computer Music Journal 26, 4, 61–68. 2002.

(Nogueira et al., 2018)

Nogueira, W., Nagathil, A., Martin, R.: Making music more accessible for cochlear implant listeners: recent developments. IEEE Signal Processing Magazine 36(1), 115–127 (2018)

(Puckette, 2002)

Miller Puckette. *FTS : A real-time monitor for multiprocessor music synthesis*. *In Computer music journal 15, 3, 58–67*. 2002.

(Ren et al., 2020)

Shihong Ren, Stephane Letz, Yann Orlarey, Romain Michon, Dominique Fober, et al. *Using Faust DSL to Develop Custom, Sample Accurate DSP Code and Audio Plugins for the Web Browser*. Journal of the Audio Engineering Society, 68 (10), pp.703-716. 2020.

(Stickland et al., 2022)

Stickland, S., Athauda, R., Scott, N.: *A new audio mixing paradigm: evaluation from professional practitioners' perspectives*. Creative Industries Journal pp. 1–49 (2022)

(WorldHealthOrganization, 2023)

WorldHealthOrganization: Deafness and hearing loss fact sheet (Feb 2023), https://www.who.int/news-room/fact-sheets/detail/deafness-and-hearing-loss

(Spangmose et al., 2019)

Spangmose, S., Hjortkjær, J., Marozeau, J.: *Perception of musical tension incochlear implant listeners*. Frontiers in neuroscience 13, 987 (2019).

(Hartmann, Kral, 2015)

Hartmann, R., Kral, A.: *Central responses to electrical stimulation*. In: Cochlear implants: Auditory prostheses and electric hearing, pp. 213–285. Springer (2004)

(Zeng et al., 2014)

Zeng, F.G., Tang, Q., Lu, T.: *Abnormal pitch perception produced by cochlear implant stimulation*. PloS one 9(2), e88662 (2014)