NOTATION DISTRIBUÉE DANS LE NAVIGATEUR, UN APERÇU

Jonathan Bell, Alexandre Craman

NOTATION DISTRIBUÉE DANS LE NAVIGATEUR, UN APERÇU

Jonathan Bell, Alexandre Craman
Aix Marseille Univ, CNRS, PRISM
Perception, Representations, Image, Sound, Music
Marseille, France
belljonathan50@gmail.com
alexandre.craman@etu.univ-amu.fr

ABSTRACT

Cet article examine quelques technologies disponibles pour la notation musicale, du point de vue de l'utilisateur, en se concentrant sur la "notation distribuée" : notation musicale affichée en temps réel sur plusieurs appareils simultanément. L'évolution rapide des navigateurs et le coût moindre des systèmes hétérogènes (multiplateformes) nous incitent à nous concentrer sur des solutions basées sur les navigateurs. Un bref aperçu rappelle quelques concepts clés de l'écosystème JavaScript, destiné aux compositeurs ayant peu de connaissances en développement web. L'étude compare ensuite trois frameworks : INScore (INScoreWeb), DRAWSOCKET, et SmartVox, en se concentrant sur leurs architectures respectives (bas niveau), ainsi que sur leur interface utilisateur. Après avoir mis en évidence les points de convergence entre INScore et DRAWSOCKET - mais aussi leurs spécificités, comme l'API de DRAWSOCKET, ou le modèle temporel d'INScore, cet article conclut sur une étude de cas : le concert choral de Tenor 21.

1. INTRODUCTION: LA NOTATION MUSICAL À L'ÈRE NUMÉRIQUE

1.1 La notation animée

De nombreux compositeurs et universitaires australiens ont composé et écrit sur les formes numériques de notation : Vickery voit dans la notation animée sur écran '*une solution importante pour visualiser un large panel de phénomènes et de techniques musicales, y compris les changements paramétriques continus, la synchronisation avec un traitement audio préenregistré ou en direct, et l'organisation formelle non linéaire.*' [1]. Hope, citant Winkler, voit la notation animée comme '*une troisième voie entre l'improvisation et la partition fixe*' [2]. Wyatt questionne le rôle du chef d'orchestre [3] quand, par exemple, la temporalité est dictée par l'avancement d'un curseur sur une page. Kim-Boyle, enfin, mène des recherche pionnières sur la rencontre entre la notation et les technologies de *réalité augmentée* [4] (voir section 4.1.1).

Ces approches de la notation présentent un changement important dans l'organisation du temps musical, qui suit encore aujourd'hui des principes hérités de l' *ars nova*. [1] Pour paraphraser Vickery, la notation animée simplifie la synchronisation d'interprètes humains avec le multimédia, ainsi que l'affichage de partitions génératives ou interactives dans lesquelles l'œuvre peut restituer un contenu différent à chaque fois, [5] ou de s'adapter à la performance de l'interprète. [6] Parmis ces technologies, trois environnements permettent l'affichage des partitions en temps réel dans le navigateur.

- INScore, opérationnel depuis 2012 [7], sort en 2020 sous sa version WEB (INScoreWeb).
- DRAWSOCKET [8, 9], qui prend racine dans divers développements de ses auteurs dans le champs des technologies pour la notation (Tenor) [10, 11, 12, 13]
- SmartVox, que l'on peut décrire comme un lecteur multimédia distribué. [14].

Alors que SmartVox est dédié à une tâche très spécifique (la diffusion et la synchronisation de vidéos mp4), INScore et DRAWSOCKET - qui supportent eux aussi le format vidéo - partagent avec le Decibel Score Player (see [15], 2. The Canvas scoring mode) la capacité d'afficher en temps réel des commandes graphiques SVG [2] en envoyant des message à des systèmes distribués (i.e. plusieurs appareils) par Open Sound Control (OSC) [16].

1.2 Solutions Web (embarquées dans le navigateur

La présente étude se limite strictement aux solutions web (tournant dans un navigateur), c'est pourquoi le *Decibel Score Player* [15] ne sera pas discutée ici, bien qu'il incarne ici une application iOS difficilement contournable dans un domaine communément appelé *animated notation* [3] .

Par définition, la communication réseau est évidemment simplifiée par les applications web, cependant le *Decibel Score Player* et quelques autres projets prennent pleinement en charge cette caractéristique essentielle de la notation distribuée : le projet de Pedro Louzeiro *Comprovisador*, qui distribue une notation en temps réel à divers

[1] La division de larges unités (*tempus*) en plus petites (*prolatio*), correspond à peu près aux indications de rythme et de mesure actuelles, voir par exemple : https://en.wikipedia.org/wiki/Prolation

[2] https://en.wikipedia.org/wiki/Scalable_Vector_Graphics

[3] http://animatednotation.com/

clients via *bach* [17] dans Max/MSP, communiquant par UDP (voir section 2.1.3), ou *Zscore* de Slavko Zagorac [18], qui est construit sur une version native d'INScore [7].

Après un bref aperçu des technologies web les plus communément utilisées par ces environnements de notation, voici une liste non exhaustive de frameworks et pièces récentes impliquant la notation musicale basées sur les navigateurs. (Section 3.1), afin de comparer quelques-unes de leurs caractéristiques générales (Section 4). Cet article se concentre particulièrement sur DRAWSOCKET, INScore et SmartVox, les environnements les mieux connus des auteurs.

2. APERÇU DES TECHNOLOGIES ASSOCIÉES

2.1 Protocoles de communication

2.1.1 HTTP

Hypertext Transfer Protocol (HTTP) est un protocole de couche d'application pour la transmission de documents hypermédias, tels que le HTML. S'appuyant sur le protocole TCP, il a été conçu pour la communication entre les navigateurs et les serveurs web.

2.1.2 WebSocket

WebSocket est différent de HTTP. WebSocket est un protocole de communication qui fournit des canaux de communication en duplex intégral sur une seule connexion TCP. Il est devenu un standard pour l'interaction en temps réel avec le navigateur.

2.1.3 TCP-UDP

TCP est un protocole orienté connexion. L'UDP est un protocole sans connexion. Comme TCP fournit un support de contrôle d'erreur et garantit également la livraison des données au routeur de destination, cela le rend plus fiable que UDP. Cependant, l'UDP est plus rapide et plus efficace que TCP.

2.1.4 OSC

OSC est un format de contenu développé au CNMAT par Adrian Freed et Matt Wright. Il est généralement destiné au partage de données d'exécution musicale (gestes, paramètres et séquences de notes) entre instruments de musique, via UDP (scénario le plus courant) ou TCP.

2.1.5 odot

Les développements de l'OSC au CNMAT ont ensuite conduit à odot [19] qui permet d'étiqueter les données avec du texte lisible par l'homme (tableaux associatifs). [4] Bien qu'elle soit abordée plus en détail dans les sections suivantes, la figure 2 montre la traduction opérée par DRAWSOCKET, à partir d'un paquet OSC (odot) en JSON.

[4] Un tableau associatif, une carte, une table de symboles ou un dictionnaire est un type de données abstrait composé d'une collection de paires (clé, valeur), tel que chaque clé possible apparaît au plus une fois dans la collection.

2.2 JavaScript

Javascript est communément appelé le langage du web. À l'origine côté client, les moteurs Javascript sont désormais intégrés à certains serveurs, généralement via node.js. La documentation la plus complète sur ce langage se trouve à l'adresse suivante : developer.mozilla.org.

2.2.1 JSON

Le format natif de JavaScript, JSON - JavaScript Object Notation - a éclipsé XML et est ainsi devenu le format d'échange de données le plus courant pour tout service Web. JSON prend en charge les tableaux associatifs ainsi que les listes ordonnées de valeurs (arrays).

2.2.2 Node.js

Node.js est un environnement d'exécution JavaScript open-source, multiplateforme et back-end qui fonctionne sur le moteur V8 et exécute du code JavaScript en dehors d'un navigateur web. Node.js doit son succès à sa plateforme d'échange de modules *npm* [5] , ainsi qu'à la commodité de développer le client et le serveur dans le même langage.

2.2.3 Node for Max

La version de Max 8 en 2018 présente un nouveau support Javascript avec *Node for Max*. Le module Max API permet d'interagir et de communiquer avec Node à partir de Max. [6]

2.2.4 Frameworks

Node.js est de bas niveau, c'est pourquoi la plupart des applications utilisent des frameworks pour traiter les fonctionalités communes des serveurs. Le routage, principalement, est utilisé pour diriger les utilisateurs vers différentes parties des applications web en fonction de la requête effectuée. Le framework Express est devenu l'un des frameworks les plus populaires parmi Electron, koa, meteor et vue.js, pour n'en citer que quelques-uns.

2.2.5 WebAssembly

WebAssembly (abrégé Wasm) est un nouveau type de code qui peut être exécuté dans les navigateurs web modernes. Fournissant des langages tels que C/C++, C# et Rust avec une cible de compilation afin qu'ils puissent s'exécuter sur le web, l'objectif principal de WebAssembly est de permettre des applications performantes sur des pages web conçues pour s'exécuter parallèlement à JavaScript.

3. UN APERÇU DES TECHNOLOGIES WEB POUR LA NOTATION DISTRIBUÉE

La '*Distribution de partitions musicales sur des plateformes mobiles et sur Internet*' [20], une idée exprimée par Fober en 2015, bénéficie aujourd'hui d'un intérêt croissant, comme en témoignent deux frameworks majeurs (voir tableau 1) ainsi que de multiples initiatives indépendantes de compositeurs et développeurs (voir tableau 2), tous brièvement décrits dans la liste ci-dessous.

[5] https://www.npmjs.com/

[6] https://docs.cycling74.com/nodeformax/api/module-max-api.html

3.1 Description

3.1.1 INScore

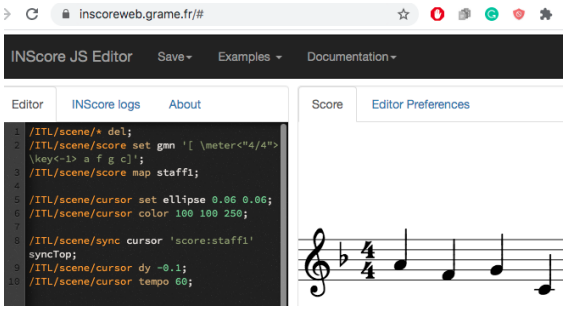

Figure 1. L'IDE INScore est disponible à l'adresse suivante : https://inscoreweb.grame.fr/.

INScore [7] est un environnement pour la conception de partitions musicales interactives augmentées, ouvert à des utilisations non conventionnelles de la notation et de la représentation musicale, y compris des capacités de notation symbolique en temps réel. Il peut être contrôlé en temps réel à l'aide de messages Open Sound Control [OSC] et d'un langage de script basé sur OSC, qui permet de concevoir des partitions de manière modulaire et incrémentielle. INScore supporte les partitions musicales étendues, combinant la notation symbolique avec des objets graphiques arbitraires. Tous les éléments d'une partition (y compris les éléments purement graphiques) ont une dimension temporelle (date, durée et tempo) et peuvent être manipulés à la fois dans l'espace graphique et temporel. Le moteur INScore est basé sur une architecture MVC (Model View Controller). Le modèle abstrait est conçu en C++ et peut être déployé sur n'importe quelle plateforme (Windows, MacOS, iOs, Android), y compris sur le Web où il est compilé en tant que module WebAssembly (voir la section 2.2.5. INScore Web dispose désormais d'un IDE (Integrated Development Environment) disponible en ligne (voir la figure 1 ci-dessus), et le package INScore Web est disponible sur *npm* en 2021.

3.1.2 DRAWSOCKET

DRAWSOCKET est précisemment décrit dans [8], raison pour laquelle seules les références retraçant sa généalogie sont fournies ici. Certaines racines de DRAWSOCKET se retrouvent dans *quintet.net* de Georg Hajdu [11], une référence dans le domaine de la notation distribuée et des performances musicales en réseau, alors que l'Internet était encore une technologie émergeante. Les travaux plus récents de Gottfried sur le "transcodage SVG vers OSC" [13] révèlent également certains aspects de ses prémisses.

Le transcodage des commandes de notation OSC vers JSON (voir la figure 2) décrit bien DRAWSOCKET, et met en évidence son potentiel pour des notations polyvalente en temps réel dans le navigateur. Tirant parti de la capacité de la bibliothèque odot [19] à formater et étiqueter dynamiquement les paquets OSC, le cadre node.js est intégré à Max (via Node for Max, voir la section 2.2.3), mais peut également fonctionner indépendamment de l'environnement

Odot (OSC) Bundle :	JSON object (or Python dictionary)

/violin : {
/key : "svg",
/val : {
/new : "circle",
/id : "bob",
/cx : 100,
/cy : 100,
/r : 20,
/fill : "green"
}
}
	{
	"violin" : {
	"key" : "svg",
	"val" : {
	"new" :"circle,
	"id" : "bob",
	"cx" : 100,
	"cy" : 100,
	"r" : 20,
	"fill" : "green"
	}
	}
	}

Figure 2. Comparaison entre un bundle odot (OSC), et un objet JSON, selon la syntaxe de lÀPI DRAWSOCKET.

Max [7]. DRAWSOCKET fournit à l'utilisateur une API [8] dédiée à la communication en temps réel avec les clients connectés. Comme le montre la figure 2, le routeur de niveau supérieur envoie le paquet à un client spécifique (ici le violon). En remplaçant /violon par /*, on envoie le paquet à tous. Ensuite, chaque mot-clé (/key ou "key" dans la figure 2) obéit à une syntaxe légèrement différente - bien que cohérente : "svg" dessinera un svg sur la page, "tween" animera le svg en fonction de sa valeur /id cible (dans la figure 2 l'id du svg cible est "bob"), "pdf", "sound" et "file" chargeront les fichiers correspondants (par ex. par exemple pdf, mp3 ou JSON). Le dépôt de partitions du Tunnel de l'Elbe fournit un bon exemple pour observer DRAWSOCKET en action. [9]

3.1.3 SmartVox

Développé à l'IRCAM en 2016 dans le cadre de SoundWorks [21] du projet Cosima (ANR-13-CORD-0010), SmartVox [14] consiste à distribuer et synchroniser des partitions audiovisuelles mp4 sur les navigateurs des smartphones des interprètes (typiquement sur un réseau local hétérogène c'est-à-dire sur des OS différents ou multiplateformes), pour les aider à chanter en polyphonie, dans des performances *in situ* (par exemple en se déplaçant dans le public), et dans un langage spectral (c'est-à-dire microtonal). Écrit en Javascript, SmartVox utilise le format natif JSON pour attribuer à chaque chanteur un fichier unique (voir figure l'exemple de script ci-dessous).

```
const score = {
  duration: 20 * 60, // seconds

  // define the different parts
  parts: {
   'soprano-1': {
      file: 'videos/soprano-1.mp4',
    },
    'soprano-2': {
      file: 'videos/soprano-2.mp4',
    },
    // ...
  },

  // define the different sections
  sections: {
    alpha: {
      time: 0,
      label: 'First_section',
    },
```

[7] Voir https://drawsocket.github.io/index.html
[8] Voir : https://drawsocket.github.io/api.html
[9] https://quintetnet.hfmt-hamburg.de/tunnel_webviewer/index.html

```
    beta: {
      time: 117,
      label: 'Second section',
    },
    // ...
  },
};
```

3.1.4 Prolonged Into the Latent (PITL)

Parmi un nombre sans cesse croissant de solutions pour les partitions synchronisées sur le Web, la solution de Justin Yang, *Prolonged into the latent (PITL)* [10] constitue un bon point de départ pour étudier comment deux types de clients (le chef d'orchestre [11] et un des chanteurs [12]) peuvent communiquer via WebSockets (voir la section 2.1.2). La mise en page de Yang rappelle le jeu vidéo *Guitar Hero* par son déroulement du temps à la verticale.

3.1.5 John

John, le semi-chef d'orchestre : Un outil de comprovisation [13] [22] de Goudard, est un logiciel de notation distribué '*conçu pour aider l'improvisation libre collective*'. Le terme *Comprovisation* fait référence au compositeur et universitaire Sandeep Bhagwati [23].

3.1.6 Anna und Marie (A&M)

Pirchner [14] [24] a développé un système de partition en temps réel pour la composition *Anna & Marie (A&M)* de Marko Ciciliani. SuperCollider envoie des messages OSC à l'environnement dédié, générant des symboles et des instructions de jeu pour chaque interprète, et les restituant sur des écrans de tablettes.

3.2 Comparaison

Les six frameworks susmentionnés vont maintenant être comparés selon les catégories suivantes : 1) Sur quel type de serveur repose l'installation. 2) Côté client, le framework cible-t-il le web ou plutôt un système d'exploitation spécifique ? 3) Supporte-t-il WebSocket (voir section 2.1.2). 4) Utilise-t-il un framework tel qu'Express (voir la section 2.2.4). 5) S'agit-il d'un cadre, ou l'architecture a-t-elle été construite pour un seul élément ? 6) Supporte-t-il le son ? 7) Supporte-t-il la notation traditionnelle (abrégée CMN pour Common Music Notation) 8) Le framework repose-t-il sur une horloge partagée/quelle bibliothèque utilise-t-il à partir de celle-ci 9) Supporte-t-il Scalable Vector Graphics (SVG) ? 10) Utilise-t-il des bibliothèques graphiques préexistantes ?

3.2.1 node.js/WebSocket

Les deux premières lignes des tableaux mettent en lumière la forte présence de node.js du côté serveur, ainsi que des WebSockets, des technologies aujourd'hui bien connues pour la mise en œuvre de communications en temps réel dans le web moderne. Le cas d'INScore est différent dans le sens où il privilégie l'autonomie à l'architecture client/serveur : la page peut donc être délivrée par des serveurs de tout type tels que python, Apache ou node.js.

La deuxième ligne du tableau 2 montre que trois technologies utilisent socket.io, [15] DRAWSOCKET et certaines parties de PITL utilisent le cadre express.js, *SmartVox* utilise *SoundWorks*[21].

	INScore	**Drawsocket**
Server	any server	node.js
Web/Native	any OS	Web
Websockets	ws	ws
uses a framework	no	express
is a framework	yes	yes
Sound	faust	tone.js
CMN	Guido	Bravura
Sync	native	timesync
SVG support	yes	yes
Graph. lib., animation	qt	tween d3js

Table 1. Comparaison d'architectures de deux frameworks majeurs pour la notation distribuée.

3.2.2 Son et notation conventionnelle/CMN (Common Music Notation)

En ce qui concerne leurs capacités sonores, INScore présente l'avantage d'intégrer *faust* [25] [16] , et promet ainsi une écriture en parallèle de la partition et des traitements DSP dans le domaine de la musique mixte. *PITL* utilise l'API audio web et DRAWSOCKET utilise tone.js, une bibliothèque simple construite par-dessus. Pour la notation traditionnelle (CMN), INScore supporte nativement GUIDO. [17]

3.2.3 Synchronisation

DRAWSOCKET et PITL montrent que *timesync* est de fait la bibliothèque la plus utilisée pour, par exemple, s'assurer qu'un message arrive à tous les clients exactement en même temps, selon une horloge partagée (Smartvox utilise une bibliothèque de l'IRCAM [26], basée sur un concept similaire).

3.2.4 Graphisme et animation

Alors que la plupart des applications abordées jusqu'ici supportent le format SVG (Scalable Vector Graphics étant la solution la plus courante pour définir les graphiques pour le web), seuls INScore et DRAWSOCKET supportent la manipulation SVG en temps réel.

DRAWSOCKET, comme son nom l'indique, *dessine* (en SVG) sur une page html, en utilisant GSAP-tween pour l'animation [18] , qui peut être conçu comme un 'définisseur

[10] Le code est disponible à l'adresse : https://github.com/elosine/prolonged_into_the_latent

[11] L'interface du chef d'orchestre est accessible via l'url suivante : https://pitl.justinyang.net/?parts=0;1;2;3;4;5;6;7;8;9;10;11;12;13;14;15&controls=yes

[12] L'interface du chanteur est accessible via l'url suivante : https://pitl.justinyang.net/?parts=0, en appuyant sur 'start' sur le conducteur, la partie du chanteur commencera à être jouée. En appuyant sur le bouton "ready" de l'interface du chanteur,

[13] Code disponible sur : https://github.com/vincentgoudard/ReactiveJohn

[14] Voir dépôt en ligne : https://github.com/asa-nerd/Anna-und-Marie

[15] Socket.IO utilise principalement le protocole WebSocket avec *polling* comme option *fallback*, tout en fournissant la même interface. Bien qu'il puisse être utilisé comme un simple wrapper pour WebSocket, il offre de nombreuses autres fonctionnalités, notamment la diffusion vers plusieurs sockets, le stockage des données associées à chaque client et les E/S asynchrones, une bibliothèque JavaScript construite sur WebSocket.

[16] Un exemple est consultable depuis l'IDE(https://inscoreweb.grame.fr/), dans le menu exemples/faust.

[17] https://guidodoc.grame.fr/#welcome-to-guido

[18] Disponible sur : https://greensock.com/docs/v3/GSAP/Tween

de propriétés hautes performances'. En ce qui concerne l'utilisation de bibliothèques externes pour les rendus graphiques aucune convergence claire ne peut être trouvée parmi les six frameworks considérés ici.

	PITL	A&M	John	SmartVox
Server	node.js	node.js	node.js	node.js
Web/Native	Web	Web	Web	Web
Websockets	socket.io	socket.io	∅	socket.io
uses a fr.	express	electron	Meteor	SoundWorks
is a fr.	no	no	yes	yes
Sound	webaudio	∅	∅	(mp4)
CMN	∅	∅	∅	∅
Sync	timesync	∅	∅	@ircam
SVG support	yes	yes	yes	no
Graph. lib.	Three.js	snap.svg	d3js	∅

Table 2. Comparaison des caractéristiques architecturales de quelques frameworks pour la notation distribuée dans le navigateur.

Après ce rapide tour d'horizon technique du paysage de la notation en temps réel/distribuée, les considérations techniques ainsi que les expériences des utilisateurs avec INScoreWeb, SmartVox et DRAWSOCKET seront discutées plus en détail, afin de mettre en évidence ce qui les rapproche et ce qui les différencie..

4. ÉTUDES COMPARATIVES

Lorsqu'on les compare deux à deux, les trois environnements susmentionnés révèlent quelques similitudes frappantes (voir section 4.3/Figure 3), mais soulèvent également des questions lorsqu'ils abordent différemment les mêmes domaines.

4.1 INSCORE et SmartVox

4.1.1 Augmented Reality

Les auteurs ont documenté certains travaux artistiques dans le domaine de la notation AR avec SmartVox : [27, 28, 29]. Des travaux émergents tels que [4, 30] permettent d'envisager des développements rapides dans ce domaine, où la prescription d'un geste dans l'espace devrait révéler des représentations bien plus intuitives (souvent appelées prescriptives, ou de type tablature, par opposition à descriptives c'est-à-dire reposant sur un système abstrait tel que la portée à 5 lignes) que celles utilisées sur papier depuis des siècles, ou celles développées sur des écrans animés depuis quelques décennies.

Bien qu'elle ait été réalisée par [31], dans laquelle quatre artistes-interprètes étaient guidés par Hololens, l'évolution rapide et le caractère éphémère de ce type de technologie nous incitent à privilégier des solutions moins coûteuses sur smartphone. Jusqu'à présent, la méthode simple utilisée par l'auteur consistait en un simple écran au-dessus de la tête de l'interprète. Cependant, avec des installations aussi bon marché, l'affichage holographique - qui nécessite une image différente pour chaque œil - ne permettait souvent pas un affichage confortable en raison de problèmes de calibrage liés à la taille du téléphone de l'interprète et à la longueur de sa distance inter-pupillaire.

INScore peut fonctionner comme une application Android native, qui a donné des résultats très prometteurs sur les lunettes EPSON bt-350 dans ce domaine. Le mécanisme de transfert d'INScore [19] permet le transfert en temps réel d'une instance INScore (sur un ordinateur portable) vers une autre (sur des lunettes), en écrivant le script suivant, dans lequel '$glasses' correspond à l'adresse ip du dispositif cible.

```
glasses = "192.168.0.26:7000";
/ITL forward $glasses;
```

Cela s'est avéré pratique à des fins de débogage, et a révélé un affichage confortable pour l'utilisateur. La version native Android de INScore a obtenu des résultats stables. L'inconvénient des lunettes EPSON bt-350, pour les tests basés sur un navigateur, est qu'elles ne parviennent pas actuellement à charger les pages html servies par DRAWSOCKET, INScoreWeb ou SmartVox.

Figure 3. INScore embarqué sur des lunettes EPSON bt-350 (à gauche), dans la pièce "*Fantaisie, Querying Schumann's op.73*" (J. Bell, 2021)

.

4.1.2 Son

Le son peut ne pas sembler être une fonctionnalité centrale pour les logiciels dédiés à la notation musicale. Pourtant, il a été démontré dans [32, 33, 34] que de nombreuses œuvres reposent sur l'utilisation de fichiers sons comme partition (pour des chanteurs le plus souvent). SmartVox, essentiellement un lecteur mp4 distribué, tire parti de la balise vidéo HTML, dont les médias intégrés supportent aussi bien l'audio que la vidéo.

L'implémentation du navigateur INScore [35], grâce à la publication récente du package *npm* Faust [20] . (Faust compilé en tant que bibliothèque WASM), étudie actuellement les possibilités d'exploiter les capacités DSP de la page web qui rend la partition. Cette fonctionnalité permettra donc (entre autres possibilités) une synchronisation précise entre les transformations électroniques et la partition, par exemple l'ouverture du microphone du navigateur lorsque le curseur atteint une certaine note.

[19] Décrit ici : https://inscoredoc.grame.fr/refs/11-forwarding/
[20] https://www.npmjs.com/package/@grame/libfaust

4.2 DRAWSOCKET et SmartVox : gestion de la mémoire cache et des délais

4.2.1 Rejoindre la performance à tout moment

Par rapport aux applications natives, les pages web sont fragiles, et les problèmes suivants méritent d'être pris en compte : 1) les pages web peuvent parfois se charger de manière incorrecte en raison de problèmes de réseau ; 2) certaines fonctionnalités peuvent ne pas fonctionner selon le navigateur et le système d'exploitation utilisés ; 3) les comportements par défaut des smartphones (sourdine, veille, mode sommeil...), ou les interactions involontaires de l'utilisateur (comme les balayages, les clics ou le branchement des écouteurs) peuvent avoir des effets périlleux dans des situations de performance.

Dans ce cas, un rechargement est souvent nécessaire. Ensuite, des problèmes peuvent survenir si quelque chose ne va pas au milieu d'une performance musicale, et si l'état actuel de l'application n'est pas stocké dans le cache (par exemple, à quelle mesure de la ligne de temps de la composition sommes-nous "maintenant" ?) En DRAWSOCKET, le système de cache stocke automatiquement toutes les commandes de dessin, de sorte que le rafraîchissement de la page côté client garde la trace de toutes les commandes de dessin depuis le démarrage du serveur, ou depuis que la page a été effacée. Lorsqu'un client est en retard, un mécanisme lui permet de rattraper son retard lorsqu'un message est reçu trop tard par un client [21] . Plus précisément, pour les animations tween (voir note de bas de page n° 15), avec la fonction 'cmd', DRAWSOCKET vérifiera la différence entre l'heure de début et l'heure actuelle (selon l'horloge partagée abordée dans la section 3.2.3) et sautera en avant si elle est en retard. [22]

L'une des forces de SmartVox consiste en sa capacité à mettre à jour le 'currentTime' (l'instant actuellement affiché dans la vidéo) à chaque tick du serveur : lorsque les clients interrogent périodiquement l'horloge partagée pour vérifier si la dérive n'est pas trop importante [23] . Cette fonctionnalité s'est avérée très robuste dans plusieurs performances [24] , si par exemple, au milieu d'un morceau, un problème survient et que l'interprète doit rafraîchir sa page, ou s'il n'était pas prêt lorsque le morceau a commencé.

4.2.2 Planificateur

Le comportement par défaut de DRAWSOCKET privilégie la réactivité instantanée à la synchronisation. Si, par exemple, un son est déclenché pour tous les clients, ce son sera joué le plus rapidement possible. Plutôt que de retarder les messages d'une valeur donnée pour s'assurer que tous les clients le reçoivent en même temps, en utilisant le message 'cmd', le son sera joué en avance s'il est en retard. Le mécanisme est comparable à celui expliqué précédemment avec les tweens (cf. note de bas de page 20).

Chaque message entrant est automatiquement horodaté en fonction d'une horloge partagée. Pendant la rédaction de cet article, deux messages différents ('del' et 'schedule') sont testés, à la fois localement et dans une configuration distante. 'schedule' retarde un message d'une valeur donnée en fonction du décalage de synchronisation de l'horloge, alors que 'del' le retarde simplement. Plus précisément, ces deux messages font partie d'un 'event' en cours de développement : https://drawsocket.github.io/api.html#event système de traitement, qui vise à fournir une API généralisée pour la synchronisation des événements. Le comportement par défaut de DRAWSOCKET privilégie la réactivité instantanée à la synchronisation. Si, par exemple, un son est déclenché pour tous les clients, ce son sera joué le plus rapidement possible. Plutôt que de retarder les messages d'une valeur donnée pour s'assurer que tous les clients le reçoivent en même temps, en utilisant le message 'cmd', le son sera joué en avance s'il est en retard. Le mécanisme est comparable à celui expliqué précédemment avec les tweens (cf. note de bas de page 20).

4.3 INSCORE - DRAWSOCKET, similarités

Afin de montrer comment, fondamentalement, INScore et DRAWSOCKET obéissent au même type de commandes de dessin pilotées par OSC, les deux scripts suivants dessinent une ligne ou un rectangle nommé "curseur", en haut à gauche de la partie de violon. Dans INScore, l'origine est au centre, c'est pourquoi x et y sont négatifs (mis à l'échelle entre -1. et 1.). Dans DRAWSOCKET l'origine est en haut à gauche, et se compte en pixels (voir Figure 4, les 3 premières lignes dans le script INScore, colonne du milieu dans DRAWSOCKET). Cependant, pour animer l'objet curseur et ainsi le déplacer sur la partition, INScore et DRAWSOCKET utilisent des méthodes différentes (voir Figure 4, ligne 4 jusqu'à la fin dans le script INScore, colonne de droite dans DRAWSOCKET), qui seront détaillées plus en détail dans la section suivante.

INScore & INScore WEB	DRAWSOCKET	

## création du curseur: 	
 	
/ITL/scene/cursor set rect 0.01 0.2;
/ITL/scene/cursor color black;
 	
## création d'une ligne (trajectoire): 	
/ITL/scene/line set line wa 1 0;
/ITL/scene/line x -0.5; ## vers la gauche
/ITL/scene/line y -0.7; ## vers le haut
/ITL/scene/line duration 1; ## 4 secondes
 	
## synchronisation du curseur à la ligne: 	
/ITL/scene/cursor tempo 60 ;
/ITL/scene/sync cursor line syncFrame;
	"// création du curseur"
	
	/violin : {
	/key : "svg",
	/val : {
	/new : "line",
	/id : "cursor",
	/x1 : 200,
	/x2 : 200,
	/y1 : 40,
	/y2 : 80,
	/style : {
	/stroke : "black"}
	}
	}
		"// définition d'un déplacement de 200 pixels vers la droite"
		
		/violin : {
		/key : "tween",
		/val : {
		/id : "score-tween",
		/target : "#cursor",
		/dur : 5,
		/vars : {
		/x : 200,
		/paused : "true"
		}
		}
		}

Figure 4. Animation du curseur dans INScore et DRAWSOCKET.

4.4 INSCORE - DRAWSOCKET, une différence significative dans la conception des animations

4.4.1 Modèle temporel INScore

La description du temps dans INScore [36] partage avec deux autres projets français (Antescofo [37, 38] et Iscore

[21] Voir le code source :https://github.com/HfMT-ZM4/drawsocket /blob/master/code/node/lib/drawsocket-client.js,
ligne 947-1069 et 1567-1613)

[22] Reloading the page in the middle of a tween animation : https://youtu.be/2ahjjbS5s2U

[23] Voir le code source ici : https://github.com/belljonathan50/SmartVox0.1 /blob/master/src/client/player/PlayerExperience.js, ligne 153).

[24] par exemple Deliciae, Common Ground, Le temps des nuages, SmartVox... Tous décrits dans d'anciennes publications TENOR du même auteur.

[39]) des préoccupations similaires concernant la capacité de la technologie à gérer à la fois le temps continu et le temps événementiel. Les objets INScore ont donc une durée et une date afin d'être ensuite synchronisés graphiquement en fonction de leur relation temporelle, ce qui permet par exemple de "surveiller" un événement pour en déclencher un autre (par exemple, déclencher un tour de page après qu'un curseur le curseur a fini de traverser la portée...). Dans l'exemple ci-dessus (voir Figure 4) le curseur est synchronisé (selon un tempo donné - 60) à une ligne (un segment d'une certaine longueur dans l'espace graphique, et d'une durée donnée).

Dans INScore, la synchronisation d'un objet (a) avec un autre (b) a une signification très particulière, qui peut être comprise comme "faire adopter à x les propriétés spatiales de y, en fonction du temps". Bien que ce ne soit qu'une façon de les comprendre, mais comme exemple de point d'entrée utile, les objets graphiques d'INScore peuvent être interprétés comme appartenant à deux catégories différentes : curseurs (a) et trajectoires (b), ou, en d'autres termes, joueurs/pointeurs (a) et score (b). Typiquement, le curseur est 'synchronisé' à une trajectoire donnée (maître). Tout comme une partition traditionnelle (fixe), 'b' peut être exécutée à une vitesse légèrement différente à chaque fois par son interprète (a), nous attribuons donc une date et une durée fixes à la partition b, et un tempo au curseur a (un exemple en *temps absolu* est fourni dans la section 5.5, *temps absolu* contraste avec *temps musical* qui est relatif au tempo).

Nous trouvons intéressant de rappeler ici comment diverses améliorations d'INScore ont finalement conduit à la nouvelle spécification d'un attribut "tempo" (voir [34], fin de l'introduction), ce qui témoigne des exigences complexes du temps musical. Dans *Perspective Temporelles* [25] , l'observation de la façon dont la vitesse du curseur change au tout début pourrait être un exemple approprié pour illustrer comment la relation temps/graphique nécessite souvent des ajustements raffinés ([3], section 2.1, relation temps/graphique). En effet, dans l'exemple mentionné dans la note de bas de page n° 24, la trajectoire horizontale du curseur est divisée en deux segments [470 , 540[et [540, 2880[, [26] , chacun étant doté d'une durée correspondante, d'où l'effet perçu d'une accélération du curseur.

4.4.2 Les animations dans DRAWSOCKET

En revanche, DRAWSOCKET propose une solution d'animation simple, le "tween", issue de la bibliothèque GSAP - un standard pour l'animation javascript en HTML5. L'animation est définie en fonction du temps et de l'espace graphique à parcourir. L'approche tween est plus facile à gérer que l'INScore pour les cas simples, mais peut en revanche rencontrer des limites lorsque le déroulement temporel n'est pas linéaire par rapport à l'espace graphique. Pour gérer de tels cas, l'implémentation de l'interpolation de DRAWSOCKET prend en charge les 'lignes temporelles d'interpolation multi-segment' (accessibles dans le fichier d'aide de DRAWSOCKET, onglet animation de l'interpolation). [27]

[25] La pièce est disponible à http://berio.grame.fr/perspectives/

[26] Pour une explication de la syntaxe de définition des segments, voir : https://inscoredoc.grame.fr/refs/12-mapping/#segments-definitions

5. ÉTUDE DE CAS : LE 'REMOTE CHOIR CONCERT @TENOR 2021'

Cet article est écrit à un moment où le monde est profondément marqué par les conséquences du Covid-19. Certains ensembles vocaux constitués ont dû endurer une année entière sans aucune répétition. La pratique musicale contemporaine se voit soudain contrainte d'opérer massivement dans un domaine que Hajdu [11] avait exploré dès les premiers jours de l'internet, et qui, à l'époque, était tout sauf facile à réaliser : "Concevoir un environnement de performance en réseau, tel que mon Quintet.net, est probablement l'une des tâches les plus exigeantes auxquelles un compositeur ou un artiste visuel puisse être confronté aujourd'hui" [11]. La transmission audio en temps réel sur l'internet étant impensable au tournant du millénaire, Hajdu avait envisagé un système (quintet.net) dans lequel la musique jouée par les instrumentistes était enregistrée, puis codée en midi, pour être distribuée sur le réseau. Comme il l'a formulé dans le titre de son article "Embodiment and disembodiment in networked music performance" (Embodiment et désincodiment dans les performances musicales en réseau), Hajdu a également anticipé très tôt les nombreux défauts (techniques et artistiques) des performances dans lesquelles les musiciens sont éloignés les uns des autres. Une autre limitation bien connue des performances musicales en réseau avant l'ère covide était due au fait que le retard du réseau empêchait les musiciens de se répondre rythmiquement les uns aux autres [40].

En réponse à la crise, et à l'initiative de Hajdu, un concert choral à distance a été organisé à la Horschule de Hambourg pour le Tenor 2021 [28] , dans lequel chaque chanteur a reçu un kit audio à faible latence (Rasbperry Pi 4 + microphone + carte son), grâce à l'effort de Jacob Sello qui a configuré vingt clients JackTrip intégrés au Raspberry Pi, ainsi que deux iPads (un pour les sessions de zoom, et l'autre pour la notation distribuée).

Cinq pièces ont été répétées pour le concert, toutes utilisant DRAWSOCKET de diverses manières, ainsi qu'un système audio à faible latence. L'avantage évident consistant à permettre aux chanteurs de répéter ensemble depuis chez eux, les compositeurs situés dans quatre pays différents pouvaient également assister aux répétitions.

L'une des caractéristiques qui s'est avérée la plus utile ici est sa capacité à charger dynamiquement les différents morceaux du concert, de sorte que les chanteurs (clients) n'aient rien d'autre à faire que de rejoindre l'url qui leur est attribuée au début du concert.

5.1 Anders Lind : MmmUMmmbing

L'interface de cette pièce est à l'origine un patch Max, et a été entièrement réécrite en DRAWSOCKET, afin que le

[27] Plusieurs fonctions d'assouplissement intégrées dans GSAP font des mouvements non linéaires, et il est également possible d'écrire ses propres fonctions de temporisation avec GSAP https://greensock.com/docs/v3/Eases.

[28] L'enregistrement du concert est consultable ici: https://youtu.be/1sumbS_c8Y4

compositeur puisse télécommander la notation générée en direct depuis la Suède alors que le chœur répétait à Hambourg. [29]

5.2 Justin Yang : Prolonged into the latent (PITL

Comme nous l'avons vu précédemment dans le tableau 2, *Prolonged into the latent (PITL)* possède son propre environnement, et DRAWSOCKET a donc été utilisé ici uniquement pour encapsuler le site Web de Justin Yang dans un *iframe*.

```
/bas4 : {
  /key : "html",
  /val : {
    /new : "iframe",
    /parent : "forms",
    /id : "iframe_ex",
    /style : {
      /position : "absolute",
      /top : "0px",
      /left : "0px",
      /width : "100vw",
      /height : "100vh"
    },
    /src :
"https://pitl.justinyang.net/?parts=0"
  }
}
```

DRAWSOCKET redirige ici vers la voix la plus grave de la composition ('part 0' correspond à la voix 'basse 4', accessible via l'url DRAWSOCKET concaténée avec '/bas4') vers la partie correspondante du site web de Justin Yang. [30]

5.3 Richard Hoadley : Unthinking Things

Unthinking Things a été initialement écrit en INScore. La pièce algorithmique a été composée par Hoadley à l'aide de SuperCollider envoyant des messages de contrôle à INScore : [41]. Le portage sur DRAWSOCKET pour le concert consistait en une vidéo de la pièce générée par INScore, servie et synchronisée par DRAWSOCKET. [31]

5.4 Jonathan Bell : Common Ground

Common Ground [28] est écrite à l'origine en bach [17], et plus particulièrement dans sa plus récente *bell* extension textuelle [42, 43]. La performance originale impliquait des chanteurs dansant dans un espace immersif [28], avec des partitions distribuées et synchronisées par SmartVox embarqué sur un Raspberry Pi. Comme pour Hoadley, la performance finale était composée de vidéos synchronisées via DRAWSOCKET. [32]

5.5 Palestrina : O crux Ave

L'œuvre de Palestrina [33] était la seule dont le langage est basé sur une pulsation régulière - bien que très lente - ce qui a soulevé les questions mentionnées précédemment : le rôle du chef d'orchestre lorsque la pulsation peut être transmise par des moyens de notation animés. Les répétitions ont permis des essais itératifs d'animations basées sur la pulsation [34] et des approches contrastées avec des curseurs défilants réalisés avec INScore, avec lequel la correspondance entre le temps et la position du curseur sur l'écran, au pixel près, peut être notée avec une grande précision : dans l'exemple suivant par exemple, le curseur parcourt la distance entre x1 (208) et x2 (249) en une seconde (t2 - t1), puis la distance entre x2 (251) et x3 (305) en 1 seconde (t3 - t2). [35]

```
 # x1   x2    y1    y2      t1          t2
([208,  249[  [93,  394[) ([0:0:0,  0:1:00[)
 # x2   x3    y2    y3      t2          t3
([251,  305[  [97,  394[) ([0:1:00,  0:2:00[)
```

La perspective envisagée pour les tentatives futures consistera à capturer le geste du chef d'orchestre à l'aide d'une technologie de suivi des gestes [44] pour surmonter les limitations causées par le décalage actuel des plateformes de vidéoconférence actuelles (comme le zoom), qui rend actuellement la conduite impraticable.

6. CONCLUSION

Avec la présente étude, les auteurs espèrent avoir fait la lumière sur le domaine émergent de la notation distribuée dans le navigateur. Avec l'API DRAWSOCKET et les capacités de synchronisation d'INScore, si l'on pense aux capacités de mise à l'échelle de telles technologies (la performance du tunnel de l'Elbe à Hambourg St-Pauli [9] impliquait 144 musiciens [36] , Le temps des nuages a été créé avec 80 chanteurs [37]), ou les situations de performance imprévisibles auxquelles elles peuvent conduire lorsqu'elles sont combinées à des technologies de RA de plus en plus accessibles, nous espérons que les cas discutés ici inciteront davantage de compositeurs à s'intéresser à ce domaine passionnant.

Nous tenons à remercier Dominique Fober, pour son temps et son soutien généreux, ainsi que l'équipe de Hambourg pour l'expérience du concert choral Tenor 21.

7. REFERENCES

[1] L. Vickery, "The limitations of representing sound and notation on screen," *Organised Sound*, vol. 19, no. 3, pp. 215–227, 2014.

[2] C. Hope, "Electronic scores for music: The possibilities of animated notation," *Computer Music Journal*, vol. 41, no. 3, pp. 21–35, 2017.

[3] A. Wyatt and C. Hope, "Conducting animated notation: Is it necessary?" in *Proceedings of the International Conference on Technologies for Music Notation*

[29] Travail original d'Anders Lind : https://youtu.be/4iePLi5uQzU. Version portée sur DRAWSOCKET par Jonathan Bell : https://youtu.be/sdSyHIbK5FY. Résultat final: https://youtu.be/1sumbS$_c$8Y4?$t = 3906$

[30] Partition : https://pitl.justinyang.net/?parts=0&controls=yes Captation 'live': https://youtu.be/1sumbS$_c$8Y4?$t = 3906$

[31] La partition est disponible sur : https://youtu.be/gLWvjR8vPHw. Captation 'live': https://youtu.be/1sumbS$_c$8Y4?$t = 2439$

[32] Une première version est disponible sur : https://youtu.be/ZrLgbBw4xfU. Reprise du concert Tenor: https://youtu.be/1sumbS$_c$8Y4?$t = 1416$

[33] Captation: https://youtu.be/1sumbS$_c$8Y4?$t = 2279$

[34] une version de la pièce animée par la bibliothèque GSAP-tween de DRAWSOCKET est disponible ici : https://youtu.be/3SS9Cb0AtU0, concert TENOR: https://youtu.be/1sumbS$_c$8Y4?$t = 2274$

[35] Pour une explication de la syntaxe de définition des segments, voir : https://inscoredoc.grame.fr/refs/12-mapping/#segments-definitions

[36] Score : https://quintetnet.hfmt-hamburg.de/tunnel_webviewer/index.html Performance : https://youtu.be/cdnA_ZijYUI

[37] Recording with the score : https://youtu.be/SyFdR2HiF00 Performance : https://youtu.be/7j2_D-nQAHY?t=6424

and Representation – TENOR'20, Hamburg, Germany, 2020, pp. 169–174.

[4] D. Kim-Boyle, "3d notations and the immersive score," *Leonardo Music Journal*, vol. 29, pp. 39–41, 2019.

[5] J. Freeman, "Extreme sight-reading, mediated expression, and audience participation: Real-time music notation in live performance," *Computer Music Journal*, vol. 32, pp. 25–41, 09 2008.

[6] A. Cont, "ANTESCOFO: Anticipatory Synchronization and Control of Interactive Parameters in Computer Music." in *International Computer Music Conference (ICMC)*, Belfast, Ireland, Aug. 2008, pp. 33–40.

[7] D. Fober, Y. Orlarey, and S. Letz, "INScore - An Environment for the Design of Live Music Scores," in *Linux Audio Conference*, Stanford, United States, 2012, pp. 47–54.

[8] R. Gottfried and G. Hajdu, "Drawsocket: A browser based system for networked score display"," in *Proceedings of the International Conference on Technologies for Music Notation and Representation – TENOR'19*, Melbourne, Australia, 2019, pp. 15–25.

[9] G. Hajdu and R. Gottfried, "Networked music performance in the old elbe tunnel," in *Proceedings of the International Conference on Technologies for Music Notation and Representation – TENOR'19*. Melbourne, Australia: Monash University, 2019, pp. 55–60.

[10] G. Hajdu and N. Didkovsky, "Maxscore: Recent developments," in *Proceedings of the International Conference on Technologies for Music Notation and Representation – TENOR'18*, Montreal, Canada, 2018, pp. 138–146.

[11] G. Hajdu, "Quintet.net: An environment for composing and performing music on the internet," *Leonardo*, vol. 38, no. 1, pp. 23–30, 2005.

[12] R. Gottfried and J. Bresson, "Symbolist: An open authoring environment for user-defined symbolic notation," in *Proceedings of the International Conference on Technologies for Music Notation and Representation – TENOR'18*, Montreal, Canada, 2018, pp. 111–118.

[13] R. Gottfried, "Svg to osc transcoding as a platform for notational praxis and electronic performance," in *Proceedings of the First International Conference on Technologies for Music Notation and Representation – TENOR'15*, Paris, France, 2015, pp. 154–161.

[14] J. Bell and B. Matuszewski, "Smartvox. a web-based distributed media player as notation tool for choral practices," in *Proceedings of the International Conference on Technologies for Music Notation and Representation – TENOR'17*, A Coruña, Spain, 2017, pp. 99–104.

[15] A. Wyatt, L. Vickery, and S. James, "Unlocking the decibel scoreplayer," in *Proceedings of the International Conference on Technologies for Music Notation and Representation – TENOR'19*, Melbourne, Australia, 2019, pp. 61–68.

[16] M. Wright, "Open sound control: An enabling technology for musical networking," *Org. Sound*, vol. 10, no. 3, pp. 193–200, Dec. 2005.

[17] A. Agostini and D. Ghisi, "A max library for musical notation and computer-aided composition," in *Computer Music Journal*, 2015.

[18] S. Zagorac and M. Zbyszynski, "Networked comprovisation stratagies with zscore," in *Proceedings of the International Conference on Technologies for Music Notation and Representation – TENOR'20*, Hamburg, Germany, 2020, pp. 133–140.

[19] J. MacCallum, R. Gottfried, I. Rostovtsev, J. Bresson, and A. Freed, "Dynamic Message-Oriented Middleware with Open Sound Control and Odot," in *International Computer Music Conference*, ICMA, Ed. Denton, United States: University of North Texas, 2015.

[20] D. Fober, G. Gouilloux, Y. Orlarey, and S. Letz, "Distributing music scores to mobile platforms and to the internet using inscore," 2015.

[21] N. Schnell and S. Robaszkiewicz, "Soundworks – A playground for artists and developers to create collaborative mobile web performances," in *'Proceedings of the Web Audio Conference (WAC'15)*, Paris, France, 2015. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01580797

[22] V. Goudard, "John, the semi-conductor: A tool for comprovisation," in *Proceedings of the International Conference on Technologies for Music Notation and Representation – TENOR'18*, Montreal, Canada, 2018, pp. 43–49.

[23] S. Bhagwati, "Notational perspective and comprovisation," *Sound & Score. Essays on Sound, Score and Notation*, pp. 165–177, 2013.

[24] A. Pirchner, "Ergodic and emergent qualities of realtime scores. anna and marie and gamified audiovisual compositions," in *Proceedings of the International Conference on Technologies for Music Notation and Representation – TENOR'20*, Hamburg, Germany, 2020, pp. 189–197.

[25] Y. Orlarey, D. Fober, and S. Letz, "FAUST : an Efficient Functional Approach to DSP Programming," in *New computational paradigms for computer music*, E. D. FRANCE, Ed., 2009, pp. 65–96.

[26] J.-P. Lambert, S. Robaszkiewicz, and N. Schnell, "Synchronisation for Distributed Audio Rendering over Heterogeneous Devices, in HTML5," in *2nd Web Audio Conference*, ser. Proceedings of the 2nd Web Audio Conference (WAC-2016), Atlanta, GA, United States, Apr. 2016.

[27] J. Bell, "Networked Head-Mounted Displays for Animated Notation and Audio-Scores with SmartVox," NIME - New Interfaces for Musical Expression, Jun. 2019.

[28] J. Bell and A. Wyatt, "Common ground, music and movement directed by a raspberry pi," in *Proceedings of the International Conference on Technologies for Music Notation and Representation – TENOR'20*, Hamburg, Germany, 2020, pp. 198–204.

[29] J. Bell and B. Carey, "Animated notation, score distribution and ar-vr environments for spectral mimetic transfer in music composition," in *Proceedings of the International Conference on Technologies for Music Notation and Representation – TENOR'19*, Melbourne, Australia, 2019, pp. 7–14.

[30] G. Santini, "Action scores and gesture-based notation in augmented reality," in *Proceedings of the International Conference on Technologies for Music Notation and Representation – TENOR'20*, Hamburg, Germany, 2020, pp. 84–90.

[31] D. Kim-Boyle and B. Carey, "Immersive scores on the hololens," in *Proceedings of the International Conference on Technologies for Music Notation and Representation – TENOR'19*, Melbourne, Australia, 2019, pp. 1–6.

[32] J. Bell, "Audio-scores, a resource for composition and computer-aided performance," Ph.D. dissertation, Guildhall School of Music and Drama, 2016.

[33] S. Bhagwati, "Elaborate audio scores: Concepts, affordances and tools," in *Proceedings of the International Conference on Technologies for Music Notation and Representation – TENOR'18*, Montreal, Canada, 2018, pp. 24–32.

[34] C. Sdraulig and C. Lortie, "Recent audio scores: Affordances and limitations," in *Proceedings of the International Conference on Technologies for Music Notation and Representation – TENOR'19*, Melbourne, Australia, 2019, pp. 38–45.

[35] D. Fober, "A web based environment embedding signal processing in musical scores," in *Proceedings of the International Conference on Technologies for Music Notation and Representation – TENOR'21*, R. Gottfried, G. Hajdu, J. Sello, A. Anatrini, and J. MacCallum, Eds. Hamburg, Germany: Hamburg University for Music and Theater, 2021.

[36] D. Fober, Y. Orlarey, and S. Letz, "INScore Time Model," in *International Computer Music Conference*, Shanghai, China, 2017, pp. 64–68. [Online]. Available: https://hal.archives-ouvertes.fr/hal-02158977

[37] J.-L. Giavitto, J.-M. Echeveste, A. Cont, and P. Cuvillier, "Time, Timelines and Temporal Scopes in the Antescofo DSL v1.0," in *International Computer Music Conference (ICMC)*, ser. Prooceedings of the 2017 ICMC / EMS conférence : Hearing the self, vol. ISBN 978-0-0845274-6-5. Shanghai, China: ICMA, Oct. 2017. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01638115

[38] J.-L. Giavitto and J. Echeveste, "Real-Time Matching of Antescofo Temporal Patterns," in *PPDP 2014 - 16th International Symposium on Principles and Practice of Declarative Programming*. Canterbury, United Kingdom: ACM, Sep. 2014, pp. 93–104. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01054667

[39] A. Allombert, M. Desainte-Catherine, and G. Assayag, "Iscore: A system for writing interaction," in *Proceedings of the 3rd International Conference on Digital Interactive Media in Entertainment and Arts*, ser. DIMEA '08. New York, NY, USA: Association for Computing Machinery, 2008, pp. 360–367.

[40] C. Chafe, J.-P. Cáceres, and M. Gurevich, "Effect of temporal separation on synchronization in rhythmic performance," *Perception*, vol. 39, no. 7, pp. 982–992, 2010, pMID: 20842974. [Online]. Available: https://doi.org/10.1068/p6465

[41] R. Hoadley, "Homenaje a cervantes," in *Proceedings of the International Conference on Technologies for Music Notation and Representation – TENOR'17*, H. L. Palma, M. Solomon, E. Tucci, and C. Lage, Eds. A Coruña, Spain: Universidade da Coruña, 2017, pp. 229–238.

[42] J.-L. Giavitto and A. Agostini, "Bell, a textual language for the bach library," in *ICMC 2019 - International Computer Music Conference*, New York, United States, Jun. 2019. [Online]. Available: https://hal.archives-ouvertes.fr/hal-02348176

[43] J. Bell, "Improvements in bach 0.8.1, a User's Perspective," in *SMC - Sound and Music Computing*, Turin, France, 2020. [Online]. Available: https://hal.archives-ouvertes.fr/hal-02773883

[44] F. Bevilacqua, F. Guédy, E. Fléty, N. Leroy, and N. Schnell, "Wireless sensor interface and gesture-follower for music pedagogy," in *International Conference on New Interfaces for Musical Expression*, New York, United States, 2007, pp. 1–1, cote interne IRCAM: Bevilacqua07a. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01161378