



# Sociedad de Ingeniería de Audio

## Artículo de Congreso

Congreso Latinoamericano de la AES 2011  
30 de Agosto a 1º de Septiembre de 2011  
Montevideo, Uruguay

*Este artículo es una reproducción del original final entregado por el autor, sin ediciones, correcciones o consideraciones realizadas por el comité técnico. La AES Latinoamérica no se responsabiliza por el contenido. Otros artículos pueden ser adquiridos a través de la Audio Engineering Society, 60 East 42<sup>nd</sup> Street, New York, New York 10165-2520, USA, [www.aes.org](http://www.aes.org). Información sobre la sección Latinoamericana puede obtenerse en [www.americalatina.aes.org](http://www.americalatina.aes.org). Todos los derechos son reservados. No se permite la reproducción total o parcial de este artículo sin autorización expresa de la AES Latinoamérica.*

## Una nueva iteración de YARMI

Bruno Azzinnari, Pablo Bounous, Gastón Caldeiro, Tomás Laurenzo  
Laboratorio de Medios, InCo, Facultad de Ingeniería, Universidad de la República  
Montevideo, 1300, Uruguay  
[brazzinnari@gmail.com](mailto:brazzinnari@gmail.com), [paviko@gmail.com](mailto:paviko@gmail.com), [chugas488@gmail.com](mailto:chugas488@gmail.com), [tomas@laurenzo.net](mailto:tomas@laurenzo.net)

### RESUMEN

YARMI es un instrumento musical colaborativo, tangible, que funciona en red y opera en un espacio de realidad aumentada compartido entre los músicos y el público. Presenta una interfaz de sobremesa donde se interactúa con varios secuenciadores musicales y máquinas de efecto en tiempo real.

En este artículo se presentan los cambios y mejoras introducidos en la nueva versión del instrumento, a partir del diseño original realizado por Laurenzo, Castro y Rodríguez. En esta nueva instancia se hizo hincapié en los aspectos gráficos, de usabilidad y de interacción, introduciéndose nuevas funcionalidades —como un secuenciador orientado a notas además de uno de muestras y el soporte de *loops* de audio pregrabados—, rediseñándose también arquitectura del sistema, previo a su construcción.

### 1 INTRODUCCIÓN

YARMI<sup>1</sup> es un instrumento musical diseñado originalmente por Tomás Laurenzo, Ernesto Rodríguez y Juan Fabrizio Castro [1]. En este artículo se presenta la profundización del diseño y la implementación del instrumento realizado por Bruno Azzinnari, Gastón Caldeiro y Pablo Bounous en el marco del proyecto de Grado: YARMI, de la Facultad de Ingeniería, Udelar<sup>2</sup>, bajo la tutoría de Tomás Laurenzo.

YARMI surge de una amplia discusión dentro de los laboratorios del grupo de investigación CECAL<sup>3</sup>, acerca de qué aspectos pueden hacer a un instrumento realmente ejecutable por un músico, y no una interfaz musical más a la que sea difícil sacarle un provecho artístico. Qué patrones de diseño pueden aplicarse para lograr la correcta

ejecución del instrumento, la identificación con él, y ayudar a generar un genuino sentido musical. También, cómo el público puede entender los gestos del músico sobre el instrumento si no hay previamente un conocimiento social de los mismos, como sí acontece con los instrumentos musicales clásicos.

Esta discusión está enmarcada también dentro del trabajo realizado por Tomás Laurenzo durante el desarrollo de su Maestría: *New Media Art* [2]. Siendo el diseño de este instrumento uno de los tres ejemplos presentados como trabajos orientados a la producción artística, buscando utilizar las nuevas tecnologías como un medio y no solamente como una herramienta.

Dentro de la producción de instrumentos musicales con computadoras es posible encontrar ejemplos simples, como los clásicos *step sequencers*<sup>4</sup> o *drum machines*<sup>5</sup>, o

<sup>1</sup> YARMI: YARMI is an Augmented Reality Musical Instrument.

<sup>2</sup> Universidad de la República Oriental del Uruguay.

<sup>3</sup> Centro de Cálculo, Instituto de Computación, Facultad de Ingeniería, Udelar.

<http://www.fing.edu.uy/inco/grupos/cecal>

<sup>4</sup> Un *step sequencer*, es una clase de secuenciador musical con una subdivisión de tiempo fija, llamada “paso”.

<sup>5</sup> Una *drum machine*, es un instrumento musical electrónico diseñado para imitar el sonido de una batería y otros elementos de percusión. Comúnmente contienen un *step sequencer*.

instrumentos de difícil o limitada ejecución práctica. También sucede que suelen ser muy diferentes a los instrumentos musicales clásicos, y, por lo tanto, se descarta un gran volumen de conocimiento social relacionado al instrumento, como son las técnicas particulares de ejecución y los gestos típicos [3].

Para abordar estos temas planteados en la discusión, dos patrones de diseño fueron utilizados: el de asociar sonidos a objetos físicos, y el de los secuenciadores musicales.

Se puede hacer una división básica del control de música por computadora de acuerdo a la inmediatez de la respuesta musical tras la interacción: en un extremo se encuentran los secuenciadores musicales, que tienen una respuesta auditiva mediata luego de la interacción. Esta interacción consiste básicamente en agregar sonidos en una línea de tiempo.

En el otro extremo se encuentran los controles continuos, como los objetos físicos de una interfaz tangible<sup>6</sup>, que son capaces de generar gestos musicales inmediatos.

Esta última estrategia cae dentro de un estilo de interacción llamado *manipulación directa*. La idea central en la manipulación directa consiste en ofrecer al usuario la sensación de manipular directamente los objetos de la interfaz. Los dos patrones de diseño son muy importantes para los objetivos planteados en las discusiones: los secuenciadores son herramientas tradicionales para la construcción de música digital [4], y la manipulación directa incentiva la identificación con la interfaz, la familiaridad y la expresividad en tiempo real.

Un aspecto importante es que ambos patrones de diseño son fáciles de aprender y pueden ser fácilmente entendidos por el usuario [2], pudiéndose establecer rápidamente una correspondencia entre los gestos realizados y la música producida. Esto es muy importante para lograr la identificación y el sentido musical en el instrumento, tanto en el propio músico como dentro de los espectadores.

Los secuenciadores musicales son estándares que, aunque no tienen una metáfora de la realidad, ofrecen una interfaz reconocible dentro del área de la música por computadora.

La interacción a través de la manipulación directa de objetos físicos, se enmarca también dentro de un paradigma establecido en las interfaces tangibles (con ejemplos relacionados exitosos como *reacTable* [5], *levelHead* [6], *ISS Cube* [7], *Audiopad* [8] [9] y *Block Jam* [10] [11], entre otros).

## 2 YARMI

El instrumento consta de una o varias **estaciones**. Una estación consiste en una superficie de trabajo, un conjunto de fichas de madera, una cámara, una computadora y, potencialmente, un proyector. Cada estación es operada por uno o varios músicos interactuando con ella por medio de sus interfaces tangible y visual.

El instrumento musical es colaborativo tanto dentro de una misma estación—debido a que las fichas pueden ser manipuladas por más de una persona— como al formar la red de estaciones: el instrumento puede mantenerse en comunicación con otras instancias cercanas para hacer música en conjunto, de una forma coherente.

## 2.1 La interfaz tangible y la salida visual

La interfaz tangible está formada por un conjunto de fichas de madera (*tokens*<sup>7</sup>) con marcadores visuales reconocibles por el sistema (*fiducials*<sup>8</sup>) pegados en una de sus caras y símbolos identificables por un ser humano en la otra. Estas fichas son manipuladas por el músico, y su ubicación y rotación dentro del espacio de trabajo son captadas por el sistema mediante una cámara.



Figura 1 – Tokens y sus distintas caras.

La salida visual de la interfaz se construye en base a la información recibida a través de la interfaz tangible, y puede proyectarse sobre una superficie grande, visible tanto para el músico como para la audiencia.

En la superficie de trabajo se presentan distintos secuenciadores musicales —que llamamos *pistas*— y máquinas de efectos de sonido de tiempo real, que serán operadas por los músicos para generar la salida de audio esperada, en sincronía con la de los músicos de las demás estaciones.

## 2.2 Pistas

YARMI cuenta con dos tipos de pistas, que se crean utilizando dos *tokens* particulares: con la rotación del *StartToken* se selecciona el tipo de pista deseada y cuando un *EndToken* aparece en el área de trabajo, la nueva pista se crea.

Las pistas representan un compás de cuatro pulsos (o *beats*), que pueden subdividirse en hasta cuatro partes —modificando el parámetro de **cuantización**, mediante la rotación del *EndToken*— transformándola en un secuenciador de dieciséis pulsos.

Moviendo sus *tokens* de comienzo y fin, una pista puede reposicionarse en el área de trabajo. También puede **colapsarse** sobre un *StartToken*; esto es, llevarse a un estado en el que el área que ocupa es mucho menor y con solo ocultar o volver a mostrar el *token*, la pista detiene o reinicia su reproducción, proveyendo de esta forma un método de secuenciación manual.

La forma de componer sobre una pista depende de su tipo. Una **pista de muestras** funciona colocando *SampleBankTokens* —esto es, *tokens* que representan bancos de muestras— en sus ranuras. Con la rotación de cada uno de estos *tokens* se modifica la altura tonal de la muestra seleccionada, mientras que con ayuda del

<sup>6</sup> *Interfaz tangible*, es una interfaz en la cual la interacción con el sistema es realizada mediante la manipulación de objetos físicos particulares, y no las herramientas de uso general como teclados, ratones, etc.

<sup>7</sup> *Token*, es el objeto físico de la interfaz tangible. En YARMI, éstos son fichas redondas de madera.

<sup>8</sup> *Fiducial*, es la marca o el símbolo visual que se coloca en el campo de visión de un sistema de tracking visual, con el fin de determinar su ubicación, rotación u otra información relacionada.

*SelectorToken* –que se mencionará más adelante– se elige la muestra a reproducir.



Figura 2 – Una pista de muestras con tres *SampleBankTokens*.

Una **pista de sintetizador** funciona con la ayuda de tres *tokens* adicionales, que permiten al músico asignar una nota particular a cada ranura, o eliminarla. El sonido generado por esta pista es básico: una onda de diente de sierra con una envolvente de amplitud predeterminada. Sin embargo, con ayuda del *MIDIToken* puede hacer que su salida se redirija por un canal MIDI para poder ser interpretada por un software externo.

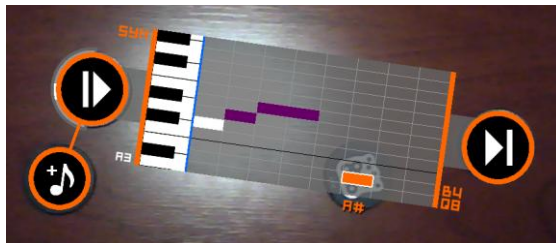


Figura 3 – Una pista de sintetizador con los *tokens* auxiliares de composición.

Finalmente, las pistas pueden ser eliminadas utilizando el *DropToken*.

### 2.3 Loops

El instrumento soporta la secuenciación continua y en bucle de un tipo particular de muestras, llamadas *loops*. Éstas son generalmente piezas rítmicas pregrabadas a una velocidad (BPM) particular, que se extienden sobre una cantidad determinada de compases.

Un *LoopBankToken* contiene un banco de *loops* previamente configurado. Con solo colocarlo sobre la mesa, la reproducción del primer *loop* del banco comienza. La selección del *loop* y de la altura tonal con el que este se reproducirá se realiza de la misma manera que para un *SampleBankToken*.

Gráficamente, un *token* de *loop* es muy similar al de una pista colapsada, permitiendo que la secuenciación de ambos tipos de construcciones se pueda realizar intuitivamente mediante el mismo mecanismo.

### 2.4 Control

No se ha mencionado hasta ahora una funcionalidad importante del instrumento: la capacidad de detener o reanudar la composición completa de la estación utilizando un *token* algo particular –un *token* con un fiducial de cada lado– que según la cara que muestre a la cámara, ejecutará uno u otro comando.

Otro *token* importante es el *BPMTToken*, que controla la velocidad de reproducción de la composición entera: tanto pistas como *loops* ajustan en todo momento su velocidad de reproducción al valor que este token modifica.

Antes de introducir el próximo *token*, es necesario introducir el concepto de *acoplabilidad*, la capacidad de un

*token* de asociarse a un objeto particular de la interfaz para afectarlo de una u otra manera.

El *VolumeToken* controla con su rotación –en su modo de funcionamiento normal– el volumen general de la estación. Sin embargo, este *token* puede también acoplarse a pistas, *tokens* de *loops* y *tokens* individuales dentro de una pista de muestras para cambiar volúmenes específicos.

Por último, existe además un *token* similar al anterior que permite controlar el balance estéreo de la salida de audio general.

### 2.5 Efectos

Los *EffectTokens* son *tokens* que pueden ser acoplados tanto a pistas como a *tokens* de *loop* para modificar la salida de audio que estos objetos generan. Si el token no está acoplado a ningún objeto, la salida de audio que se ve afectada es la global de la estación.

Actualmente están soportados tres tipos de efectos: *reverb*, *tremolo* y *delay*. Una vez que el token de efecto está en el área de trabajo, mediante su rotación y con la ayuda del *SelectorToken* pueden modificarse –con resultados en tiempo real– sus parámetros.

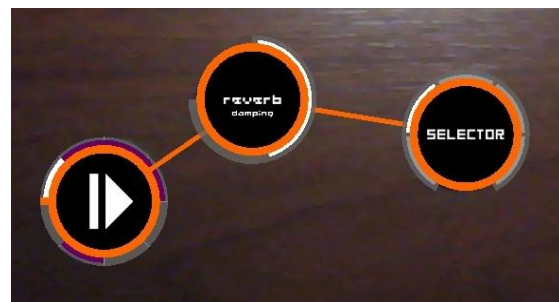


Figura 4 - Un *EffectToken* de reverberación acoplado a una pista colapsada y un *SelectorToken* acoplado al primero, seleccionando su segunda dimensión, el parámetro *damping*.

### 2.6 Herramientas

Si bien hasta ahora no se ha mencionado explícitamente, varias de las funcionalidades descritas utilizan el concepto de **dimensión**: un parámetro que ciertos tipos de *tokens* – los *tokens* **dimensionables**– poseen y que toma un valor dentro de un conjunto discreto de posibilidades (que varía de *token* a *token*) y según el cual modifican su comportamiento de alguna u otra forma.

El *SelectorToken* permite, mediante el mecanismo de acoplamiento combinado con su propia rotación, modificar la dimensión seleccionada en un *token*.

Los *tokens* dimensionables son: *SampleBankToken*, *LoopBankToken* y *EffectToken*. A todos ellos puede acoplarse el *SelectorToken* para modificar su dimensión seleccionada –que para los dos primeros es la muestra seleccionada y para el último, el parámetro del efecto que la rotación del *token* original modifica.

### 2.7 Características de red

Como se mencionó anteriormente, el instrumento tiene la capacidad de funcionar en conjunto con otras estaciones YARMI en forma coherente.

En este escenario, la estación **líder** –la última estación que haya puesto el *LeaderToken* sobre su mesa– tiene el control sobre la producción musical global, pudiendo decidir cuando ésta se detiene o reinicia, y especificando parámetros como el volumen general, BPM y el balance estéreo.

### 3 IMPLEMENTACIÓN

Cada estación está constituida de tres grandes módulos:

- un módulo escrito en C++ que utiliza openFrameworks<sup>9</sup> como framework de programación,
- un módulo de audio desarrollado en Pure Data<sup>10</sup>,
- y un sistema de reconocimiento de *fiducials* o marcadores, reacTIVision<sup>11</sup>.

La Figura 5 indica cómo se transfiere el flujo de información entre los diferentes módulos que componen la arquitectura de la aplicación.

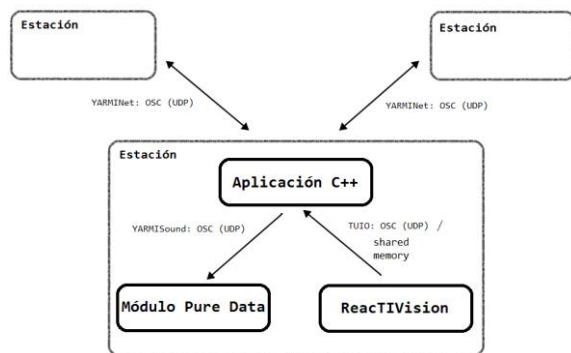


Figura 5 - Arquitectura de software de una estación

Para el reconocimiento de patrones fiduciales e implementación de la interfaz tangible usamos reacTIVision. Este software transmite información sobre la posición de los *fiducials* (capturada mediante una cámara web) al módulo principal (mediante el protocolo TUIO<sup>12</sup> basado en OSC<sup>13</sup> y enviado sobre UDP) quien maneja toda la información de la estación, se encarga del procesamiento y generación de la imagen en tiempo real. A su vez, el módulo principal transmite (mediante el protocolo propio YARMSound basado en OSC y enviado sobre UDP) información al módulo de Pure Data, el cual se encarga de generar el sonido también en tiempo real.

Las estaciones se comunican entre sí mediante el protocolo –también propio– YARMINet basado en OSC y enviado sobre UDP. Cada vez que ocurre un cambio en algunos de los parámetros de configuración en la estación líder, se realiza un *broadcast* notificando los cambios a las estaciones conectadas.

### 4 CONCLUSIONES Y TRABAJO FUTURO

Muchos de los cambios que el diseño del instrumento sufrió durante esta iteración se dieron en forma paralela a su construcción, siendo producto de pruebas de usabilidad realizadas tanto por parte del propio equipo de desarrollo

como con usuarios externos. Así se fueron estandarizando métodos de acción particulares, reglas de uso implícitas, y algunos conceptos en principio abstractos, que luego fueron tornándose más concretos.

Durante las pruebas más recientes con usuarios se comprobó que el lenguaje generado es efectivo, coherente e intuitivo; y que aparece como una base fuerte sobre la cual sería posible crear una gran variedad de construcciones adicionales, que agregarían características potencialmente muy diversas.

En vista al futuro hay varios aspectos que podrían trabajarse. Uno de ellos está relacionado con que actualmente el instrumento solo trabaja con compases de 4/4, lo cual es una gran limitante. Se pretende incorporar otras métricas generales, o particulares para diferentes pistas, o *loops*. También se podrían crear pistas de diferente longitud (en cantidad de compases), de acuerdo al diseño original.

Otra característica deseable sería la posibilidad de secuenciar diferentes pistas y *loops*, enlazándolos unos a continuación de otros.

Se pretende, también, implementar técnicas adicionales de visión por computadora que ofrezcan al usuario la posibilidad de realizar gestos manuales durante la interacción (probablemente, utilizando cámaras 3D).

### REFERENCIAS

- [1] Tomás Laurenzo, Ernesto Rodríguez and Juan Fabrizio Castro. YARMI: an Augmented Reality Musical Instrument. NIME, 2009.
- [2] Laurenzo, Tomás. Master Thesis: New Media Art.
- [3] Jordà, S. (2005). Digital Lutherie: Crafting musical computers for new musics' performance and improvisation. Ph.D. dissertation, Dept. of Computer Engineering, Universitat Pompeu Fabra, Barcelona.
- [4] Duignan M., Biddle R. & Noble J. A Taxonomy of Sequencer User-Interfaces. Barcelona: International Computer Music Conference (ICMC), 2005.
- [5] Kaltenbrunner, M. & Jordà, S. & Geiger, G. & Alonso, M. The reacTable\*: A Collaborative Musical Instrument. Manchester, U.K. : WETICE, 2006.
- [6] Oliver, Julian. levelHead. <http://selectparks.net/~julian/levelhead>.
- [7] Quarta, M. ISS Cube. Exhibited at Ars Electronica, Cybersonica, Bafta Interactive Festival. 2003.
- [8] Patten J., Recht B. & Ishii H. Audiopad: A Tag-based Interface for Musical Performance. Dublin, Ireland: Proceedings of the Conference on New Interfaces for Musical Expression, 2002.
- [9] Patten, J. & Recht, B. Audiopad. <http://www.jamespatten.com/audiopad>.
- [10] Newton-Dunn H., Nakanon H. & Gibson J. Block Jam: A Tangible Interface for Interactive Music. Montreal, Canada: Proceedings of the Conference on New Interfaces for Musical Expression, 2003.
- [11] Newton-Dunn, Henry. Block Jam. <http://www.sonyvcs.co.jp/1L/projects/blockjam>.
- [12] Openframeworks. Sitio web: <http://www.openframeworks.cc>
- [13] Pure Data. Sitio web: <http://puredata.info>
- [14] ReacTIVision. Sitio web: <http://reactivision.sourceforge.net>
- [15] TUIO. Sitio web: <http://www.tuio.org/>
- [16] OSC. Sitio web: <http://www.opensoundcontrol.org/>

<sup>9</sup> openFrameworks es una librería de código abierto, escrita en C++ y diseñada para la codificación creativa. Es desarrollada y mantenida por Zachary Lieberman y Theo Watson [12].

<sup>10</sup> Pure Data (o Pd) es un lenguaje de programación gráfico de código abierto desarrollado por Miller Puckette durante los años 90 para la creación de música computarizada interactiva y obras multimedia [13].

<sup>11</sup> reacTIVision es un framework de visión por computadora de código abierto y multiplataforma para el reconocimiento veloz y robusto de marcadores fiduciales adjuntos a objetos físicos, y para el reconocimiento de dedos sobre una superficie multitáctil [14].

<sup>12</sup> TUIO es un framework de código abierto que define un protocolo y una API para la transmisión de eventos que ocurren sobre superficies multitáctiles [15].

<sup>13</sup> OSC (Open Sound Control) es un protocolo de red que permite comunicar instrumentos de música, computadoras y otros dispositivos multimedia en tiempo real [16].