FACULTAD DE INGENIERIA
Universidad de Buenos Aires

El Fogón

Sistema distribuido para músicos con estado compartido del compás, la canción y el repertorio

Segundo Cuatrimestre de 2025

Tutor: Diego Montaldo

Luis Waldman
Padrón: 79279

Índice

Palabras Clave

- Web
- Vue.js
- WebSocket
- WebRTC
- Diseño responsivo
- Golang
- NUnit
- Playwright
- Sincronismo
- Compensación de *jitter*
- Música
- Letras
- Acordes
- Partituras

Abstracto

Español

Fogón es un sistema distribuido orientado a facilitar la práctica musical y la sincronización entre músicos. A cada uno le ofrece, desde una aplicación web progresiva, vistas para su instrumento: letras, acordes o partituras. Permite crear listas y editar canciones. También, crear sesiones en la que los participantes comparten el estado de la canción, el repertorio y hasta el compás exacto que están tocando.

El sistema ayuda al aprendizaje y la enseñanza musical al mostrar cómo realizar los acordes en cada instrumento y al permitir afinarlos.

El principal desafío técnico fue el de la reproducción en dispositivos distribuidos: un *delay* de 20 ms empieza a ser perceptible por el oído humano y la latencia en internet es mayor. Se implementó un protocolo que sincroniza los dispositivos usando WebSocket y WebRTC (con compensación de *jitter*) a través de un servidor Golang.

La arquitectura soporta numerosas vistas complejas y responsivas para distintos instrumentos, y permite agregar nuevas vistas para otros instrumentos. Para las partituras emplea la librería VexFlow, mientras que para las vistas de letra y acordes se usó un desarrollo propio.

Para probar y "debuggear.[el] sistema de sincronización hubo que desarrollar algunas vistas y controles. Las pruebas de aceptación se desarrollaron de punta a punta con NUnit y Playwright.

Palabras Clave: Web, Vue.js, WebSocket, WebRTC, Diseño responsivo, Golang, NUnit, Playwright, Sincronismo, Compensación de *jitter*, Música, Letras, Acordes, Partituras.

English

Fogón is a distributed system designed to facilitate musical practice and synchronization among musicians. It provides each musician with a progressive web application featuring instrument-specific views: lyrics, chords, or sheet music. It allows users to create playlists and edit songs. Additionally, it enables the creation of sessions where participants share the song state, repertoire, and even the exact bar they are playing.

The system aids in musical learning and teaching by showing how to perform chords on each instrument and allowing for tuning.

The main technical challenge was distributed device playback: a delay of 20 ms starts to become perceptible to the human ear, and internet latency is typically higher. A protocol was implemented that synchroni-

zes devices using WebSocket and WebRTC (with jitter compensation) through a Golang server.

The architecture supports numerous complex and responsive views for different instruments, and allows for the addition of new views for other instruments. For sheet music, it employs the VexFlow library, while for lyrics and chord views, a custom development was used.

To test and debug the synchronization system, several views and controls had to be developed. End-to-end acceptance tests were developed with NUnit and Playwright.

Keywords: Web, Vue.js, WebSocket, WebRTC, Responsive design, Golang, NUnit, Playwright, Synchronization, Jitter compensation, Music, Lyrics, Chords, Sheet Music.

Agradecimientos

A mi madre y a mi padre,

A la Educación Pública y en particular a las cátedras de Arquitectura de Software, Base de Datos, Introducción a Sistemas Distribuidos y Sistemas Distribuidos de la Universidad de Buenos Aires.

A Pitágoras, a Newton, a Turing y a todos los que asumen la heroica tarea de descubrir y transmitir ciencia.

A los artistas que prefieren estructuras rigurosas.

A las Cadenas de Márkov y la proliferación de IAs.

A vos que estás leyendo esto.

1. Introducción

Tanenbaum y Van Steen definen: Ün sistema distribuido es una colección de computadoras independientes que aparece ante sus usuarios como un sistema único y coherente"(1)

La definicion coincide con la de un grupo musical que combina armonías, melodías y ritmos de modo que se escuche como un tema único y coherente.

Para hacer esto los músicos se nutren de protocolos y mecanismos de coordinación que les permiten sincronizarse y compartir información.

Los avances en la ciencia y la ingeniería fueron incorporados por los músicos: Pitágoras sintetizó la Matemática y la Armonía, la imprenta permitió la publicación de partituras, la revolución industrial el metrónomo de Maelzel.

Desde que esta internet circulan archivos con paginas de acordes que evolucionaron a paginas multimedia, archivos MIDI, aplicaciones de edición de partituras, etc.

La aplicación .El Fogón"pone a disposición de los músicos estos avances y propone un nuevo enfoque: cada músico puede acceder con su dispositivo a un Estado compartido de compás, canción, acordes, partituras, repertorio, etc.

Esto permite que varios músicos toquen juntos y en sincronía, compartiendo y actualizando información en tiempo real , pero viendo cada uno la información de su instrumento

1.1. Motivación

Busca hacer un aporte novedoso a la musica desde la informatica, incorporando soluciones anteriores y agregando un enfoque novedoso: el estado compartido entre músicos.

Ademas, busca la alta disponibilidad: Los musicos pueden acceder a la aplicacion en su dispositivo en cualquier momento, sin necesidad de conexion a la red.

1.2. Objetivos

1.2.1. Objetivo General

Fogón es una solución dirigida tanto a cantantes y guitarristas aficionados como a músicos de orquestas profesionales: una aplicación en donde puedan buscar letras de acordes y canciones de modo intuitivo y también una herramienta que los ayude a ensayar y crear cosas nuevas.

Cada músico podrá ver en su dispositivo la vista de su instrumento: el cantante, la letra; el guitarrista, los acordes; el pianista sus partituras.

El usuario podrá ver y editar una variedad de canciones publicadas en el mismo sitio. Si se loguea con su usuario, podrá compartir sus canciones con otros usuarios.

Varios usuarios logueados podrán unirse en una sesión para sincronizar la lista de canciones, la canción que se está reproduciendo y el momento en que cambia el compas actual, subrayandolo y automatizando el scroll en la letra y en los pentagramas: de este modo podrán organizar un ensayo, un concierto o una noche de karaoke entre amigos.

1.2.2. Objetivos Tecnicos

La sincronización del compás en una sesión debe ser exacta cuando un grupo de músicos está tocando: un delay de 20 ms empieza a ser perceptible por el oído humano y la latencia en internet puede ser mayor. Los distintos dispositivos se conectarán con un servidor Golang e implementarán un protocolo que combine timestamps sincronizados (basados en NTP), buffers adaptativos y compensación del jitter (variación en la latencia) para resolver esto. El mismo servidor además por medio de HTTP intercambia los archivos de las canciones con las aplicaciones. Todas las visitas deberán poder adaptarse a distintos dispositivos, ser configurables y extensibles: será posible incorporar vistas adicionales, como notación numérica para armónica, tablaturas para guitarra, y reproductores multimedia como YouTube o Midi. Edicion de letra y acordes: También podrán editar las canciones mediante una interfaz intuitiva y accesible: la compleja relación entre las letras y los acordes, que además se agrupan en partes que se repiten según una secuencia podrá modificarse de una manera natural y sencilla. Sincronicacion: Varios usuarios logueados podrán unirse en una sesión para sincronizar la lista de canciones, la canción que se está reproduciendo y el estado de la reproduccion. Construir algunas herramientas nesarias para la musica como un afinador que permita afinar distintos instrumentos. Construir herramientas para probar y "debuggear.[el] sistema de sincronización desarrollado.

2. Estado del Arte

Describir trabajos previos relacionados y el estado actual del conocimiento en el área.

2.1. Tecnologías Existentes

Análisis de las tecnologías y soluciones existentes en el mercado.

2.2. Trabajos Relacionados

Revisión de proyectos y trabajos académicos relacionados.

2.3. Comparativa de Soluciones

Comparación entre diferentes enfoques y sus ventajas/desventajas.

3. Solución Implementada

Descripción detallada de la solución desarrollada para resolver el problema planteado.

3.1. Arquitectura del Sistema

Descripción de la arquitectura general del sistema.

3.2. Componentes Principales

Detallar los componentes principales de la solución:

- Componente 1
- Componente 2
- Componente 3

3.3. Tecnologías Utilizadas

Listar y justificar las tecnologías seleccionadas.

3.4. Implementación

Describir cómo se implementó la solución propuesta.

Listing 1: Ejemplo de código

```
def funcion_ejemplo():
    print("Hola, mundo!")
    return True
```

Figura 1: Descripción de la figura

4. Metodología Aplicada

Describir la metodología empleada para el desarrollo del trabajo.

4.1. Marco Metodológico

Explicar el marco metodológico utilizado (ágil, cascada, etc.).

4.2. Proceso de Desarrollo

Detallar las etapas del proceso de desarrollo:

1. Análisis de requisitos
2. Diseño de la solución
3. Implementación
4. Pruebas
5. Despliegue

4.3. Herramientas de Desarrollo

- Herramienta 1
- Herramienta 2
- Herramienta 3

4.4. Gestión del Proyecto

Describir cómo se gestionó el proyecto (sprints, reuniones, etc.).

5. Experimentación y Validación

Presentar los experimentos realizados y la validación de la solución implementada.

5.1. Diseño de Experimentos

Describir el diseño de los experimentos realizados para validar la solución.

5.2. Casos de Prueba

Detallar los casos de prueba ejecutados.

5.3. Resultados Obtenidos

Presentar los resultados obtenidos del trabajo realizado.

Caso de Prueba	**Resultado Esperado**	**Resultado Obtenido**
Prueba 1	Éxito	Éxito
Prueba 2	Éxito	Éxito

Cuadro 1: Resultados de las pruebas realizadas

5.4. Análisis de Resultados

Analizar e interpretar los resultados obtenidos.

5.5. Validación con Usuarios

Describir el proceso de validación con usuarios finales.

6. Cronogramas

Presentar el cronograma de actividades del proyecto.

6.1. Cronograma Planificado

Descripción del cronograma inicial planificado para el proyecto.

Actividad	**Duración**	**Período**
Análisis de requisitos	2 semanas	Enero
Diseño	3 semanas	Febrero
Implementación	8 semanas	Marzo-Abril
Pruebas	2 semanas	Mayo
Documentación	1 semana	Mayo

Cuadro 2: Cronograma planificado del proyecto

6.2. Cronograma Real

Descripción del cronograma real de ejecución del proyecto.

6.3. Desviaciones y Ajustes

Análisis de las desviaciones respecto al plan original y los ajustes realizados.

7. Riesgos y Lecciones Aprendidas

Análisis de los riesgos identificados durante el proyecto y las lecciones aprendidas.

7.1. Identificación de Riesgos

Listar y describir los riesgos identificados al inicio del proyecto.

Riesgo	**Probabilidad**	**Impacto**
Riesgo técnico 1	Alta	Alto
Riesgo de recursos	Media	Medio
Riesgo de tiempo	Baja	Alto

Cuadro 3: Matriz de riesgos del proyecto

7.2. Gestión de Riesgos

Describir cómo se gestionaron los riesgos durante el proyecto.

7.3. Problemas Encontrados

Detallar los problemas principales enfrentados durante el desarrollo.

7.4. Lecciones Aprendidas

Compartir las lecciones aprendidas durante la ejecución del proyecto:

- Lección 1
- Lección 2
- Lección 3

8. Impactos Sociales y Ambientales del Proyecto

Análisis de los impactos sociales y ambientales del proyecto desarrollado.

8.1. Impacto Social

Describir el impacto social esperado o generado por el proyecto:

- Beneficios para la comunidad
- Mejoras en la calidad de vida
- Accesibilidad y inclusión

8.2. Impacto Ambiental

Analizar el impacto ambiental del proyecto:

- Consumo de recursos
- Huella de carbono
- Sostenibilidad de la solución

8.3. Responsabilidad Social y Ética

Consideraciones éticas y de responsabilidad social del proyecto.

8.4. Medidas de Mitigación

Describir las medidas implementadas para minimizar impactos negativos.

9. Desarrollos Futuros

Describir posibles extensiones, mejoras y trabajos futuros relacionados con el proyecto.

9.1. Mejoras Propuestas

Listar las mejoras que podrían implementarse en el futuro:

- Mejora 1
- Mejora 2
- Mejora 3

9.2. Funcionalidades Adicionales

Describir funcionalidades adicionales que podrían agregarse.

9.3. Escalabilidad

Analizar cómo el sistema podría escalarse para soportar mayor carga o alcance.

9.4. Investigación Futura

Proponer líneas de investigación futuras relacionadas con el proyecto.

9.5. Conclusiones Finales

Presentar las conclusiones finales del trabajo, resumiendo los principales hallazgos y logros alcanzados.

Referencias

[1] Autor, A. (2024). *Título del libro o artículo*. Editorial.

[2] Autor, B. y Autor, C. (2023). Título del artículo. *Nombre de la Revista*, 10(2), 123-145.

[3] Autor, D. (2025). Documento en línea. Disponible en: `https://ejemplo.com`