

Universidades de Burgos, León y
Valladolid

Máster universitario

Inteligencia de Negocio y Big Data en Entornos Seguros



**Arquitectura *Big Data* de colas
para el procesamiento de vídeo en
tiempo real**

Presentado por José Luis Garrido Labrador
en Universidad de Burgos — 25 de abril
de 2020

Tutor: Dr. Álgvar Arnaiz González y Dr. José
Francisco Díez Pastor

Universidades de Burgos, León y Valladolid



Máster universitario en Inteligencia de Negocio y Big Data en Entornos Seguros

Dr. D. Álgvar Arnaiz González, profesor del departamento de Ingeniería Informática.

Expone:

Que el alumno D. José Luis Garrido Labrador, con DNI 71707244Y, ha realizado el Trabajo final de Máster en Inteligencia de Negocio y Big Data en Entornos Seguros titulado Arquitectura *Big Data* de colas para el procesado de vídeo en tiempo real.

Y que dicho trabajo ha sido realizado por el alumno bajo la dirección del que suscribe, en virtud de lo cual se autoriza su presentación y defensa.

En Burgos, 25 de abril de 2020

Vº. Bº. del Tutor:

Vº. Bº. del co-tutor:

Dr. D. Álgvar Arnaiz González

Dr. D. José Francisco Díez Pastor

Resumen

En este primer apartado se hace una **breve** presentación del tema que se aborda en el proyecto.

Descriptores

Palabras separadas por comas que identifiquen el contenido del proyecto Ej: servidor web, buscador de vuelos, android ...

Abstract

A **brief** presentation of the topic addressed in the project.

Keywords

keywords separated by commas.

Índice general

Índice general	III
Índice de figuras	V
Índice de tablas	VI
Memoria	1
1. Introducción	3
2. Objetivos del proyecto	5
2.1. Objetivos generales	5
2.2. Objetivos técnicos	5
2.3. Objetivos personales	6
3. Conceptos teóricos	7
3.1. Teoría de computación distribuida	7
3.2. Teoría de procesamiento de imágenes	7
4. Técnicas y herramientas	9
4.1. Gestión de flujo	9
4.2. Infraestructura de bajo nivel	10
5. Aspectos relevantes del desarrollo del proyecto	11
5.1. Desarrollo de la aplicación web	11
6. Trabajos relacionados	15

7. Conclusiones y Líneas de trabajo futuras	17
Apéndices	18
Apéndice A Plan de Proyecto Software	21
A.1. Introducción	21
A.2. Planificación temporal	21
A.3. Estudio de viabilidad	23
Apéndice B Especificación de Requisitos	25
B.1. Introducción	25
B.2. Objetivos generales	25
B.3. Catalogo de requisitos	25
B.4. Especificación de requisitos	25
Apéndice C Especificación de diseño	27
C.1. Introducción	27
C.2. Diseño de datos	27
C.3. Diseño procedimental	27
C.4. Diseño arquitectónico	27
Apéndice D Documentación técnica de programación	29
D.1. Introducción	29
D.2. Estructura de directorios	29
D.3. Manual del programador	29
D.4. Compilación, instalación y ejecución del proyecto	29
D.5. Pruebas del sistema	29
Apéndice E Documentación de usuario	31
E.1. Introducción	31
E.2. Requisitos de usuarios	31
E.3. Instalación	31
E.4. Manual del usuario	31
Bibliografía	33

Índice de figuras

5.1. Funcionamiento del mando de SNES para el control de la aplicación web por parte del paciente.	12
5.2. Menú del paciente	12
5.3. Menú del terapeuta	13

Índice de tablas

A.1. Tareas del <i>sprint</i> 0	22
A.2. Tareas del <i>sprint</i> 1	22
A.3. Tareas del <i>sprint</i> 2	23
A.4. Tareas del <i>sprint</i> 3	23
A.5. Tareas del <i>sprint</i> X	23

Memoria

Introducción

Descripción del contenido del trabajo y del estructura de la memoria y del resto de materiales entregados.

Objetivos del proyecto

Los objetivos del proyecto se han dividido en tres apartados siendo estos los objetivos generales, los técnicos y los personales.

2.1. Objetivos generales

- Exploración de las diferentes herramientas para el procesado de vídeo en tiempo real a través de las fases de emisión, recogida, encolado, ingestión, procesado, enriquecimiento y almacenamiento.
- Estudio del estado del arte en análisis de imagen para problemas de salud ante distintos escenarios tanto en aspectos físicos (iluminación, enfoque...) como en aspectos lógicos (resolución, tasa de refresco...).
- Implementación del software necesario para la recogida de vídeo en tiempo real sobre sistemas de videoconferencia.

2.2. Objetivos técnicos

- Crear una infraestructura software basada en contenedores *Docker* para ser independientes del software anfitrión.
- Desplegar un *pipeline* sobre herramientas de la suite de *Apache* para el *Big Data* que satisfagan el flujo ETL propuesto (TODO Cita al flujo).
- Desarrollar algoritmo sobre *Spark Stream* que procese los vídeos generando los datos necesarios para los estudios posteriores.

2.3. Objetivos personales

- Contribuir a la mejora de la calidad de vida, a través de facilitar soportes para la rehabilitación, de pacientes con enfermedad de Parkinson.
- Conocer más profundamente las herramientas de la suite de *Apache* y como estas se pueden combinar para facilitar tareas de *Big Data*.
- Completar mi formación durante el máster a través de la creación de una solución que utiliza gran parte de los conocimientos adquiridos durante el mismo.

Conceptos teóricos

En este capítulo se explicarán los conceptos teóricos subyacentes a todos los apartados del proyecto. Se afrontará en dos bloques, uno respecto a la teoría sobre la computación distribuida (??) y otro sobre el procesamiento de imágenes.

3.1. Teoría de computación distribuida

3.2. Teoría de procesamiento de imágenes

Definiciones

Imagen 24 bits: codificación habitual de las imágenes a color. Cada pixel se define como la combinación de tres enteros sin signo de 8 bits. La codificación (o espacio de color) necesaria para la visualización es la combinación de las capas de color roja, verde y azul, conocida como RGB, aunque también la BGR usada por openCV.

HSV : acrónimo del inglés de matiz, saturación y valor. Consiste en un modelo de color basado en los componentes del mismo y representa una transformación no lineal del espacio de color RGB.

Matiz : grado del ángulo respecto a la rueda de color. Representa un color único y algunos ejemplos son el rojo (0°), amarillo (60°) o verde (120°).

Técnicas y herramientas

4.1. Gestión de flujo

Uno de los puntos más esenciales de este trabajo es recoger y dirigir los *streams* de vídeo que se reciben. Por tanto, escoger una correcta aplicación para la gestión de este flujo de datos es esencial.

Dentro de la suite de *Apache* existen varios componentes que se encargan de la gestión del flujo de datos. Con el objetivo de que el sistema fuese lo más robusto, y siguiendo las recomendaciones del estado del arte (TODO citas de esto) se combinarían las herramientas siguiente.

- ***Apache Kafka*** [2] es un proyecto de intermediación de mensajes que trabaja sobre el patrón publicación-suscripción funcionando como un sistema de transacciones distribuidas. Incorpora para la implementación de este patrón un sistema de colas para la distribución de mensajes. Aporta una API para el productor, el consumidor, el flujo y el conector y la conexión se realiza a través del protocolo de la capa de transporte *TCP*.
- ***Apache Spark Streaming*** [3] es la extensión sobre la API de *Spark* para la creación de aplicaciones sobre flujos de datos. Es un consumidor nativo de *Kafka*, *Flume*, los sistemas de ficheros *HDFS* y *S3* entre otras herramientas. El funcionamiento interno es a través de crear pequeños lotes de datos para pasarlo al motor de *Spark* y retornar los lotes procesados.

También se ha contemplado el uso de ***Apache Flume*** [1], herramienta de gestión de flujo diseñada para hacer una gestión distribuida de manera

fiable y altamente disponible de los datos. Proporciona un servicio eficiente para la recogida, agregación y almacenamiento de los datos. Sin embargo, aunque esta herramienta pudiese suplir las necesidades de una gestión de flujo se ha descartado debido a que está optimizado para la gestión de *logs*, en detrimento de datos serializados como vídeos además de que no proporciona sistemas de colas.

4.2. Infraestructura de bajo nivel

Otro apartado importante en el despliegue de la aplicación son las herramientas y técnicas a ser usadas para la producción. Para esto se utilizan:

- ***GNU/Linux***, el sistema operativo más extendido en el entorno de los servidores [5, 7] además de ser el disponible en los servidores prestados para la realización de este proyecto.
- ***Docker***, un software de gestión de contenedores estandarizados, semejante a los entornos *chroot* que facilita la virtualización de software en un entorno seguro y ligero. Sobre este motor se ejecutarán las aplicaciones del entorno de *Apache Spark* [4], *Apache Kafka* [6] además de la aplicación desarrollada para el cumplimiento de los objetivos.

Aspectos relevantes del desarrollo del proyecto

En este capítulo se explicarán las partes más importantes del desarrollo del proyecto.

5.1. Desarrollo de la aplicación web

Para realizar la primera fase del desarrollo, la recogida de los datos, se ha desarrollado una aplicación web que permitiese grabar a los pacientes durante la realización de los ejercicios de rehabilitación mientras son dirigidos por el personal especializado.

Esta aplicación se ha compuesto de dos partes. Una ha consistido en el diseño y consecuente implementación de una interfaz hombre-máquina fácil de usar y accesible mediante un mando sencillo¹. Esta se compone de una serie de botones de colores, directamente relacionados con los del mando (figuras 5.1 y 5.2), que permiten iniciar una comunicación mediante videoconferencia con el personal terapéutico como finalizar la reunión. Durante el proceso de la llamada el vídeo del paciente es capturado y enviado al servidor para su procesamiento posterior.

La segunda parte consiste en la interfaz del terapeuta encargado de dirigir las sesiones de rehabilitación del paciente ofreciendo una interfaz sencilla que permite conectarse a las videoconferencias de los diferentes pacientes y dirigir las sesiones de terapia además de gestionar la evolución del paciente (figura 5.3).

¹Concretamente un mando de la consola SNES adaptado para ser usado mediante

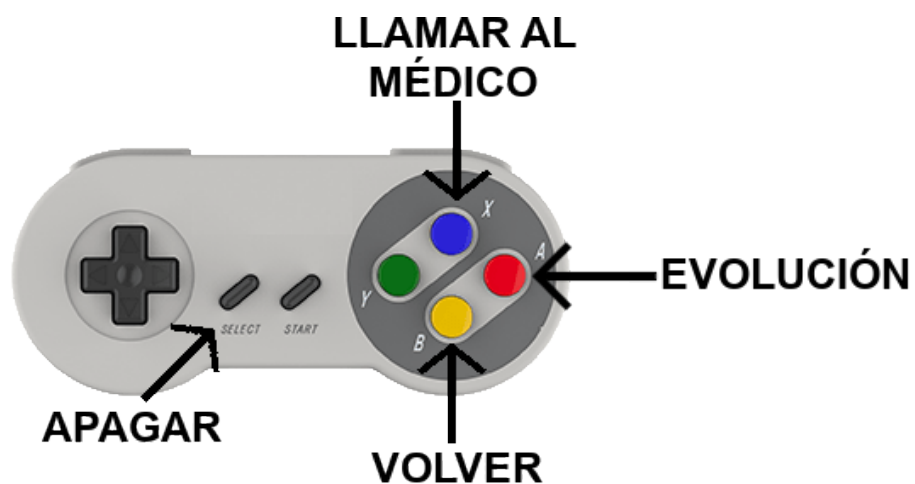


Figura 5.1: Funcionamiento del mando de SNES para el control de la aplicación web por parte del paciente.

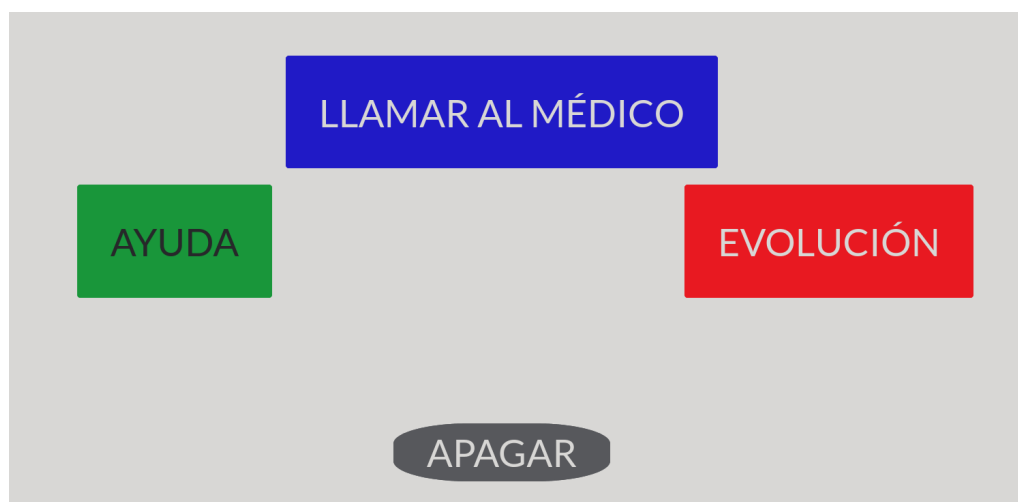


Figura 5.2: Menú del paciente



Figura 5.3: Menú del terapeuta

TODO: Quizás sería interesante explicar las valoraciones otorgadas de la aplicación si se tienen antes de la entrega del trabajo.

TODO: Comentar las características de hardware de los dispositivos de despliegue

Trabajos relacionados

Conclusiones y Líneas de trabajo futuras

Apéndice

Apéndice A

Plan de Proyecto Software

A.1. Introducción

A.2. Planificación temporal

La planificación temporal se ha realizado adaptando la metodología *Scrum*. Para poder adaptarlo a un trabajo de una sola persona para un proyecto educativo se han han considerado las siguientes indicaciones:

- El desarrollo se ha basado en iteraciones o *sprints* de dos semana de duración aproximadamente.
- Cada uno de los *sprints* contiene las tareas que se realizaron en el mismo.
- Cada tarea tiene un coste estimado dependiendo de lo que el programador estime conveniente siguiendo los parámetros de tiempo a emplear, dificultad técnica entre otros.
- Una vez concluida una tarea se especifica el coste real para poder estimar de una manera más correcta tareas de *sprints* siguientes.
- Al finalizar cada *sprint* se realiza una reunión con los tutores del proyecto.

Sprint 0

El *sprint* 0 consistió en el desarrollo de la aplicación web para la recogida de datos para el proyecto. Es el único *sprint* realizado en colaboración con el alumno José Miguel Ramírez Sanz.

Tarea	Estimado	Final
Diseño de la interfaz web	3	3
Creación de la plantilla maestra de toda la web	3	2
Creación de la plantilla base para el menú del paciente	5	5
Implementación del menú del terapeuta	2	2
Creación de la conexión para videollamada - Terapeuta	1	1
Creación de la conexión para videollamada - Paciente	1	1
Implementación del sistema de inicio de sesión	2	3
Creación de <i>plugin</i> para la captura y emisión del vídeo del paciente	8	13
Creación de la interfaz de gestión del paciente	2	2

Tabla A.1: Tareas del *sprint* 0

Sprint 1

El *sprint* 1 consistió en la exploración de herramientas para la creación y procesamiento de flujos de vídeo.

Tarea	Estimado	Final
Búsqueda de herramientas para la creación de flujos de datos	2	2
Pruebas sobre <i>Apache Flume</i>	5	3
Pruebas sobre <i>Apache Kafka</i>	5	5
Búsqueda de herramientas para el despliegue por contenedores	2	2
Prueba de despliegue de <i>Apache Kafka</i> para <i>Docker</i>	2	2
Búsqueda de herramientas para el procesamiento de flujos de vídeo	2	2
Pruebas con <i>Spark Streaming</i> con <i>OpenCV</i>	5	13

Tabla A.2: Tareas del *sprint* 1

Sprint 2

El *sprint* 2 consistió en la implementación de una infraestructura de contenedores *Docker* que conectase todos los servicios para el flujo.

Tarea	Estimado	Final
Despliegue de <i>Apache Spark</i> para el nodo <i>máster</i>	2	2
Despliegue de <i>Apache Spark</i> para varios nodos <i>slave</i>	2	1
Despliegue de aplicación <i>Java</i> para probar <i>Spark</i>	3	8
Conexión de la infraestructura de <i>Kafka</i> con aplicación <i>Java</i>	5	3
Recogida de <i>stream</i> de vídeo e ingestión en <i>Kafka</i>	8	8

Tabla A.3: Tareas del *sprint* 2

Sprint 3

El *sprint* 3 consistió en la automatización de los procesos del *sprint* 2¹.

Tarea	Estimado	Final
-------	----------	-------

Tabla A.4: Tareas del *sprint* 3

Sprint X

El *sprint* X consistió en Y².

Tarea	Estimado	Final
-------	----------	-------

Tabla A.5: Tareas del *sprint* X

A.3. Estudio de viabilidad

Viabilidad económica

Viabilidad legal

¹Pendiente de realizar

²Plantilla

Apéndice B

Especificación de Requisitos

- B.1. Introducción
- B.2. Objetivos generales
- B.3. Catalogo de requisitos
- B.4. Especificación de requisitos

Apéndice C

Especificación de diseño

- C.1. Introducción
- C.2. Diseño de datos
- C.3. Diseño procedimental
- C.4. Diseño arquitectónico

Apéndice D

Documentación técnica de programación

- D.1. Introducción
- D.2. Estructura de directorios
- D.3. Manual del programador
- D.4. Compilación, instalación y ejecución del proyecto
- D.5. Pruebas del sistema

Apéndice E

Documentación de usuario

- E.1. Introducción
- E.2. Requisitos de usuarios
- E.3. Instalación
- E.4. Manual del usuario

Bibliografía

- [1] Apache Flume. <https://flume.apache.org/>.
- [2] Apache Kafka. <https://kafka.apache.org/>.
- [3] Spark Streaming - Spark 2.4.5 Documentation. <https://spark.apache.org/docs/latest/streaming-programming-guide.html>.
- [4] Mario Juez-Gil. `mjuez/spark-cluster-docker`. <https://github.com/mjuez/spark-cluster-docker>.
- [5] MuyLinux. Red Hat lidera el segmento Linux en el mercado de servidores. <https://www.muylinux.com/2018/10/19/red-hat-lidera-mercado-linux-servidores/>.
- [6] Wurstmeister. `wurstmeister/kafka-docker`. <https://github.com/wurstmeister/kafka-docker>.
- [7] Wensong Zhang et al. Linux virtual server for scalable network services.