UNIVERSIDAD AUTÓNOMA DE MADRID

ESCUELA POLITÉCNICA SUPERIOR

TRABAJO FIN DE MÁSTER

Herramienta de visualización de itinerarios de aprendizaje en MOOCs

Máster en Ingeniería Informática

Lidia París Cabello

Junio 2017

Herramienta de visualización de itinerarios de aprendizaje en MOOCs

AUTOR: Lidia París Cabello

TUTOR: Estrella Pulido Cañabate

Dpto. Ingeniería Informática
Escuela Politécnica Superior
Universidad Autónoma de Madrid

Resumen

En los últimos años, la mayoría de las personas utilizan un *smartphone* o *tablet* a diario con diversos fines tales como interaccionar en redes sociales, realizar consultas a buscadores, visualizar tutoriales, realizar cursos, etc. La tecnología facilita realizar estudios oficiales o cursos online de formación complementaria a distancia, teniendo disponible una conexión a internet y un dispositivo electrónico.

Este trabajo está relacionado con el uso de estas nuevas tecnologías en el ámbito de la educación, en especial la educación a distancia a través de cursos online. Los MOOCs son cursos online masivos y abiertos que permiten el acceso a nueva formación a muchas personas a la vez. Como consecuencia de la interacción de los estudiantes con los recursos disponibles en un MOOC, se genera un registro que almacena cada acción realizada.

Estos datos son una fuente de información valiosa para los responsables de los cursos. Sin embargo, el registro generado no es fácil de interpretar a simple vista. Este trabajo utiliza herramientas del ecosistema *Apache Spark* y la librería *D3JS* para visualizar gráficamente el comportamiento de los estudiantes en la primera edición del curso online *Jugando con Android – Aprende a Programar tu Primera App* ofertado a través de la plataforma *edX*.

Palabras clave

MOOC, Big Data, visualización, gráficas

Abstract

In the last few years, most people use a smartphone or tablet daily for various purposes such as interacting in social networks, searching engine queries, visualizing tutorials, taking courses, etc. The technology facilitates performing official studies or online distance learning courses, only having an Internet connection.

This work is related to these new technologies application in the education field, especially in distance education through online courses. MOOCs are massive, open online courses that allow access to new training for many people at a time. Because of the student's interaction with the resources available in a MOOC, a record is generated that stores each action performed.

These data are a source of valuable information for those responsible of the course. However, the generated record is not easy to interpret at first glance. This work uses ecosystem *Apache Spark* tools and the D3JS library, allowing to visualize the student behaviors in the first edition of the online course *Jugando con Android – Aprende a Programar tu Primera App* through the edX platform.

Keywords

MOOC, Big Data, visualization, graphics

Tabla de contenido

Índice de tablas

Índice de figuras

1. Introducción

1.1. Motivación

En la actualidad, la forma de vivir en sociedad, de relacionarnos y de obtener nueva información está evolucionando a pasos agigantados debido a las nuevas tecnologías. En muy poco tiempo se está produciendo un gran cambio al surgir el ordenador y posteriormente internet, generando un giro inesperado en la vida de las personas. Además, la evolución de los dispositivos desde ordenadores de tamaños industriales a dispositivos electrónicos de tamaño portátil, en especial los *smartphones* y *tablet,* ha facilitado el acceso de las personas a una gran cantidad de información [1].

La cantidad de información disponible y la posibilidad de acceso desde cualquier lugar, están facilitando utilizar estas nuevas tecnologías en diversos ámbitos como investigación, medicina, educación, etc.

Este trabajo está relacionado con el uso de estas nuevas tecnologías en el ámbito de la educación. Dichas tecnologías están permitiendo realizar avances tanto en la educación presencial como la educación a distancia. Dentro de la educación presencial, la tecnología facilita el uso de *tablets* en las aulas, pizarras electrónicas o la creación de repositorios online para almacenar recursos como documentos o ejercicios, entre otros.

En relación con la educación a distancia, la tecnología facilita realizar estudios oficiales utilizando herramientas de apoyo al proceso de aprendizaje que permiten compartir vídeos, imágenes, grabaciones, etc. Además de estudios oficiales, también existe la posibilidad de realizar cursos online de formación complementaria de diversas temáticas como idiomas, salud, marketing, programación, etc.

Estos cursos online permiten que la educación sea accesible a cualquier persona con conexión a internet. El cambio desde las plataformas educativas cerradas a entornos de aprendizaje abiertos supone la posibilidad de que miles de personas de cualquier parte del mundo puedan acceder a diferentes iniciativas de aprendizaje. Es ahí donde reside su característica más destacada, y es por lo que se ha producido un auge de los cursos online.

Estos cursos se están desarrollando sobre plataformas que facilitan la compartición del material. Las más populares son *Coursera, Udemy, Udecity* y *edX.*

Los MOOCs son impartidos por profesores universitarios, que se encargan de compartir el material necesario para realizar el curso. Los materiales disponibles del

curso suelen ser documentos, vídeos, actividades, exámenes, etc. que son utilizados por los estudiantes matriculados en el mismo. Cada estudiante realiza diferentes interacciones con los distintos recursos disponibles en el curso. La interacción es la acción que realiza un estudiante con los distintos recursos disponibles en un curso online. Por ejemplo, la interacción con los vídeos es la visualización que engloba acciones como “play”, “stop”, “pause”, “avanzar”, “retroceder”, etc. La interacción con un documento es abrir el documento para realizar una lectura o la interacción con el foro es crear un hilo, responder a una pregunta, realizar una consulta, etc.

Las plataformas realizan un registro de todas las interacciones de los estudiantes. Además, como los MOOCs no tienen límite de matrículas y son gratuitos, el volumen de estudiantes matriculados en estos cursos suele ser muy elevado. Teniendo en cuenta todo esto, el volumen de datos que se genera es muy elevado. El problema es que la información no está estructurada, por lo que no es fácil obtener conclusiones o realizar un seguimiento de las interacciones de los estudiantes.

La Universidad Autónoma de Madrid también organiza cursos online en este tipo de plataformas de distintas materias como química, medicina, informática, etc. Dentro de la temática sobre informática, los profesores de la Escuela Politécnica Superior de la Universidad Autónoma de Madrid imparten el curso “Jugando con Android - Aprende a Programar tu Primera App”. Este curso consiste en aprender los fundamentos de programación en Android mediante el desarrollo de una aplicación. Como en la mayoría de MOOCs, el número de alumnos matriculados es muy elevado y el registro de las interacciones no está estructurado, lo que complica el seguimiento del alumnado. Como el curso es impartido por profesores de la facultad, es posible el acceso a los datos registrados por la plataforma para dicho curso.

Figura 1. Curso “Jugando con Android - Aprende a Programar tu Primera App”

En la Cátedra UAM/IBM de la Escuela Politécnica Superior de la Universidad Autónoma de Madrid se han desarrollado diversos proyectos para explotar la

información disponible y poder utilizarla. En concreto, se han desarrollado dos proyectos: predicción y análisis de interacciones de usuarios en plataformas de enseñanza online [2] y análisis del abandono en cursos online [3]. El primero de estos proyectos analiza las interacciones realizadas por los estudiantes en un curso online y predice el comportamiento de los mismos utilizando su patrón de acceso a la plataforma. El segundo proyecto predice el abandono o la permanencia de los usuarios en un curso online, utilizando diferentes algoritmos de clasificación y herramientas del ecosistema *Big Data Apache Spark*,

El trabajo que se describe en esta memoria se integra dentro de esta línea de trabajo sobre *learning analytics,* centrándose más en la parte de la visualización.

1.2. Objetivos

El principal objetivo de este trabajo fin de máster es crear una herramienta de visualización de itinerarios de aprendizaje (*learning paths*) de los alumnos matriculados en un MOOC. Los itinerarios de aprendizaje son la secuencia de interacciones relacionadas con los recursos disponibles en el MOOC que realiza un alumno desde el inicio del curso hasta que finaliza o lo abandona.

La representación de los itinerarios de los estudiantes va a permitir a los profesores hacer un seguimiento individualizado y observar patrones de conducta en los comportamientos de los estudiantes. Esta representación va a facilitar la visualización de la dedicación de tiempo a cada tarea de una manera muy visual. Además, se podrá ver el tiempo que dedican los alumnos a cada recurso, los períodos en los que trabajan, el tiempo que permanecen conectados, los tipos de recursos que más utilizan, si hay períodos de días, semanas o meses que el estudiante no entra en la plataforma, etc. Realizando observaciones sobre este gráfico se podrá concluir cuáles son los recursos más importantes, los itinerarios que facilitan al estudiante finalizar el curso con éxito, los itinerarios de los estudiantes que abandonan el curso, etc.

Para poder llevar a cabo el objetivo principal ha sido necesario la realización de varios objetivos subyacentes. Estos nuevos objetivos que se plantean a raíz del principal son: (i) adquirir conocimientos relacionados con herramientas que procesen gran cantidad de datos y (ii) aprender sobre librerías o software que permita mostrar dichos datos de manera muy visual. Para ello, se necesitó conocer las herramientas de Big Data para procesar el gran número de datos, estructurarlos, analizarlos, limpiarlos, etc. Estas herramientas se utilizarán durante la primera parte del proceso de este proyecto, fase de procesamiento de los datos. Además, para mostrar los datos procesados en

gráficos o grafos, fue necesario adquirir conocimientos en librerías de visualización de datos.

1.3. Organización de la memoria

La memoria se divide en los siguientes capítulos:

- **Introducción**: se expone la motivación que ha surgido para llevar a cabo este proyecto y los objetivos del mismo.

- **Estado del arte:** en este capítulo se expondrá el contexto de los cursos online y las plataformas sobre las que se desarrollan. También se detallará el contexto sobre las distintas maneras de visualizar la información.

- **Herramientas:** en este apartado se describe el estudio realizado sobre las herramientas disponibles para el procesamiento de los datos y la visualización de los mismos. Además de una comparativa entre dichas herramientas y cuál se ha escogido y por qué.

- **Implementación:** en este capítulo se detalla todo el proceso llevado a cabo para conseguir el objetivo final desde el procesamiento de los datos hasta la visualización de la información. Explicando el proyecto desarrollado, cómo utilizarlo y los resultados obtenidos.

- **Conclusiones y trabajo futuro:** se resumen las ideas obtenidas tras la conclusión del trabajo y cómo se han conseguido los objetivos establecidos al comienzo del mismo. Además, se indicarán las posibles líneas de trabajo futuro.

2. Estado del arte

En este apartado se profundiza en los tipos de cursos online que hay disponibles en Internet, las características de cada uno y una comparativa entre ellos. Además, se detallan las plataformas más importantes sobre las cuales funcionan los cursos online. Por otro lado, se define el término Big Data, cuáles son sus principales características y los cambios que ha producido la gran cantidad de datos disponibles que existen en la actualidad. Asimismo, se discute el término nuevo *dashboard*. Estos conceptos son independientes entre sí, debido a sus diferentes funciones, pero son los pilares en los que se basa la idea original de este proyecto. Por todo ello, es necesario profundizar en su conocimiento antes de desarrollar este trabajo.

2.1. Cursos

2.1.1. Cursos Online

Los cursos online son cursos que se realizan de forma no presencial haciendo uso de dispositivos electrónicos conectados a Internet. Los estudiantes pueden trabajar desde cualquier sitio con conexión a Internet. A través de un campus virtual, el curso proporciona una colección de contenidos que el estudiante puede consultar y, además, interactuar con los demás estudiantes y con el docente [4].

Las comunicaciones entre los estudiantes y docentes suelen ser asíncronas, ya que permiten distintos ritmos de trabajo y compatibilizar los horarios de todos los estudiantes.

Los materiales y el tipo de actividad utilizados en estos cursos son muy diversos. Por norma general se componen de documentos escritos, vídeos, actividades grupales con otros estudiantes, debates en el foro, ejercicios prácticos sobre la unidad, etc.

2.1.2. Cursos Masivos (Massive Open Online Course)

Los MOOC (acrónimo en inglés de Massive Open Online Course) o COMA en español (Curso Online Masivo Abierto), son cursos online dirigidos a un amplio número de participantes a través de Internet para permitir una educación abierta y a gran escala (*massive*).

Según el periódico *The New York Times* publicado en el artículo "*The Year of the MOOC*" [5], el año 2012 fue el año de los MOOCs debido a la amplia atención que había recibido este nuevo término por parte de la comunidad educativa mundial. Desde

entonces los MOOCs se han convertido en un referente en la educación online, como complemento a la educación presencial.

Aunque los MOOCs existían con anterioridad, su uso se limitaba a usuarios con un perfil concreto. Es a partir del 2012 cuando el acceso a estos cursos se amplía a todo el público.

2.1.2.1. Características de los MOOCs

Para comprender qué es un MOOC es necesario definir cada una de las palabras que forman parte del nombre [6].

Curso

Un MOOC tiene unos objetivos de aprendizaje definidos que deben obtener los alumnos al finalizar ciertas actividades en un periodo de tiempo determinado. Además, debe de contar con evaluaciones que permitan medir que los conocimientos han sido adquiridos.

Abierto

Abierto tiene varios significados en este tipo de cursos [7]. Primero, que a estos cursos puede acceder cualquier estudiante fuera de la universidad y no exigen requisitos previos para poder inscribirse. En los MOOC es necesario registrarse, para realizar un seguimiento y conocer cómo el alumno realiza el curso y los resultados que obtiene, llevando un registro personalizado del progreso.

En segundo lugar, “abierto" indica que los recursos que se están utilizando para el curso son “contenidos abiertos” (*open content)* y que los contenidos que genera el curso se publican en abierto (*open licence*) para que puedan ser reutilizados. Esta definición de “abierto” es la menos utilizada ya que hay muchos cursos MOOC sobre plataformas privadas que no comparten los recursos de esta manera, como por ejemplo Coursera.

Otra interpretación de “abierto” es que los cursos se implantan sobre una plataforma de código abierto (*open source*), que permite la adaptación de la plataforma modificando el código original [8].

En línea

El curso se realiza a distancia a través de Internet y no requiere la asistencia física a un aula, sólo es necesario un dispositivo móvil conectado a la red. Esta característica es esencial para que cualquier persona desde cualquier parte del mundo con una conexión a Internet pueda participar en estos cursos y así lograr que se cumpla la siguiente característica.

Gran escala

Un curso MOOC permite el acceso a un número muy grande de estudiantes, mucho mayor que una clase presencial o un curso en línea tradicional. Además, el curso debe estar preparado para aceptar cambios en el número de estudiantes en varios órdenes de magnitud, por ejemplo, pasar de 1.000 a 100.000 estudiantes, sin que eso suponga un problema importante para su funcionamiento.

Figura 2. Características de los MOOC [9]

En general, los MOOCs tienen una estructura semanal definida con vídeos, diferentes tipos de actividades y un foro para comunicarse con el equipo de profesores y entre los estudiantes. Una vez completado el curso satisfactoriamente, la plataforma facilita un certificado gratuito a los usuarios. Además, si se pagan unas tasas se puede obtener un diploma oficial. Actualmente, esta estructura está evolucionando a series de cursos temáticos, cursos más cortos, con menos carga de actividades y más proyectos prácticos.

2.1.3. Comparativa entre los cursos online y los MOOCs

En la Tabla 1 se detallan las principales diferencias entre los dos tipos de cursos.

Curso Online	MOOC
Se desarrolla en una única plataforma con funcionalidades acotadas	Puede desarrollarse en múltiples plataformas
Dirigido a un entorno cerrado	Dirigido a un público abierto
Acceso con pago de matrícula	Acceso gratuito
Grupo limitado	Participación masiva
Apoyo directo del profesor	Apoyo de la comunidad
Comunicación mediante foros de debate	Diversidad de herramientas de comunicación, uso de redes sociales
Más enfocado a la obtención de resultados	Mayor relevancia al procedimiento de aprendizaje
Orientado hacia la evaluación y acreditación	Énfasis en el proceso de aprendizaje más que en la evaluación y acreditación
Sin problemas de seguridad debido a ser un entorno cerrado	Potenciales problemas de seguridad

Tabla 1. Comparativa entre los cursos online y los MOOCs [10],[11]

2.2. Plataformas de MOOCs

En los últimos años se está produciendo un auge de los MOOCs, ya que son una pieza clave para la formación en todos los niveles y a cualquier edad. Para que los MOOCs sean accesibles a gran escala, es necesario un sistema que permita la ejecución de diversas acciones y el acceso a través de Internet. Esto es lo que se conoce como plataformas de enseñanza online. Hay múltiples plataformas que ofertan MOOCs (Figura 3)

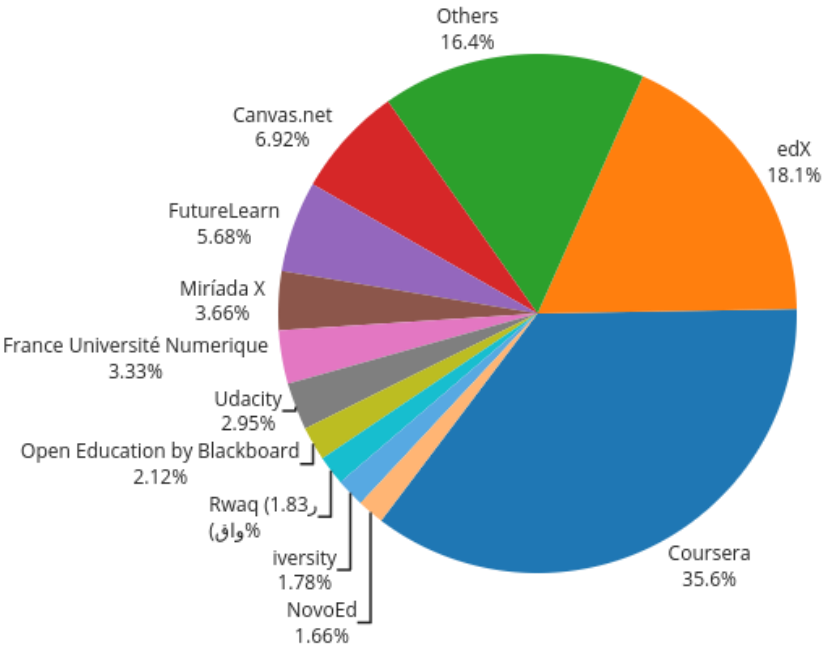

Figura 3. Plataformas que distribuyen los MOOCs (año 2015) [12]

Como se observa en la Figura 3, las más destacadas son Coursera y edX, por lo que se profundiza en ellas a continuación.

2.2.1. Coursera

Fue fundada en 2012 por dos profesores de Stanford Computer Science que querían compartir sus conocimientos y habilidades con el mundo [13]. Los profesores Daphne Koller y Andrew Ng pusieron sus cursos en línea para cualquiera que los quisiera realizar y enseñaron a más estudiantes en pocos meses de lo que podrían tener en toda una vida en el aula. Desde entonces, es una plataforma donde cualquier persona, en cualquier lugar puede aprender y obtener credenciales de las mejores universidades y proveedores de educación del mundo.

2.2.2. edX

Estudiar en universidades tan prestigiosas como el MIT, la Universidad de Boston e incluso Harvard es complicado por la distancia a EEUU y el alto precio de las matrículas que no está al alcance de todas las personas. Sin embargo, desde hace unos años la Plataforma edX [14] oferta estudios que proporcionan estas grandes universidades de prestigio, ofreciendo cursos online con profesores y participantes de reconocida influencia, con un coste que se puede considerar casi simbólico.

Figura 4. Logo plataforma edX [14]

Para poder participar en los cursos online no es necesario ningún requisito, simplemente tener acceso a internet. El proceso para participar en los cursos online de la plataforma edX es sencillo: solo hay que registrarse en el área de usuarios del curso online que se haya elegido.

Los cursos online de la plataforma edX tratan de diversos temas, desde las energías renovables, la inteligencia artificial o la informática, hasta la filosofía, la comunicación y la literatura. Su duración depende del curso online, y puede ir de semanas a meses. En ellos hay disponibles recursos académicos elaborados por profesionales de prestigio y profesores de las universidades que los organizan; como la Universidad de Berkeley, el MIT, la Universidad de Boston o la Universidad de Washington, por mencionar algunas.

2.3. Big Data

Big Data es el proceso de recolección de grandes datos (estructurados, no estructurados y semi-estructurados) y su análisis para encontrar información oculta, patrones recurrentes, correlaciones, etc [15]. Estos conjuntos de datos son tan grandes y diversos que los medios tradicionales de procesamiento son ineficaces. El término Big Data hace referencia a toda aquella información que no puede ser procesada o analizada utilizando procesos o herramientas tradicionales. Sin embargo, Big Data no se refiere a una cantidad en específico, ya que es usualmente utilizado cuando se habla en términos de petabytes y exabytes de datos.

Es necesario recoger y organizar todo este gran **volumen** de información para disponer de todos los registros completos y que las conclusiones que se obtengan sirvan eficientemente para la toma de decisiones. Además, este volumen de datos se genera a gran **velocidad** y es necesario analizarlo y reaccionar inmediatamente para proporcionar respuestas. Los datos generados son heterogéneos porque provienen de diferentes orígenes y pueden ser estructurados, semi-estructurados o no estructurados. Para procesar la **variedad** de información, es necesario combinar todos los datos

generados y crear un conjunto homogéneo para dar uniformidad a la misma, siendo uno de los puntos fuertes del Big Data.

Figura 5. Big Data. Característica volumen y variedad [16]

Las características mencionadas corresponden a una interpretación academicista. Desde la visión de un analista de datos, hay dos características más: veracidad y valor. La característica asociada a la calidad de los datos es la veracidad de éstos. La **veracidad** puede definirse como el grado de confianza que se establece sobre los datos a utilizar. Esta cualidad es de gran importancia ya que determinará la calidad y la confianza de los resultados. El esfuerzo que se realiza al utilizar Big Data, se considera que tiene que servir para mejorar y ampliar el valor a los individuos, a los negocios, la ciencia, la sociedad, etc. y por ello se añade una nueva característica: **valor**. Procesar todos estos datos tiene que fomentar la innovación, la competitividad y promover una mejor calidad de vida [17].

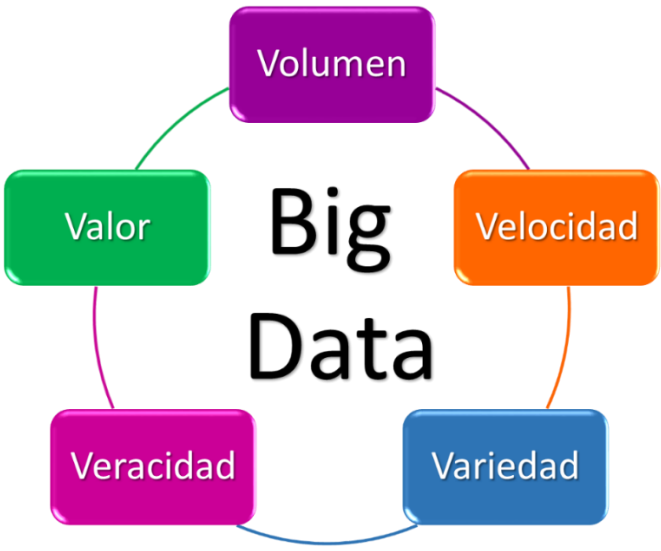

Figura 6. Las cinco Vs de Big Data

2.4. Dashboard

Un *dashboard* es un panel de datos en el que las empresas visualizan la información más importante relacionada con los objetivos del negocio [18]. La función de un *dashboard* es transformar los datos en información valiosa. Por ello, su principal objetivo es orientar al usuario para facilitar la toma de decisiones sobre la empresa basándose en los gráficos que está visualizando. Un *dashboard* permite hacer un seguimiento del negocio y puede ofrecer información sobre determinadas problemáticas. La detección de estos problemas facilitará la realización de acciones correctivas por parte de la empresa para conseguir sus objetivos.

El objetivo final de un *dashboard* es agregar mucha información en un espacio reducido y que las personas que lo vean lo entiendan y puedan tomar decisiones en función de la información representada. Los *dashboard* correctamente diseñados proporcionan información que está:

- Perfectamente organizada
- Resumida, comprimida e identificando las excepciones
- Orienta a las personas que ven el *dashboard* para ayudar a tomar decisiones en relación con los objetivos representados
- Muestra la información con mecanismos concisos pero muy claros

No existe un *dashboard* que abarque las diversas necesidades de todos los destinatarios, pero si se puede diseñar un *dashboard* perfecto que se adecue a cada destinatario. Por lo tanto, lo ideal no es tener un único dashboard, sino hacer un diseño a medida, un *dashboard* para cada necesidad (ver Figura 7).

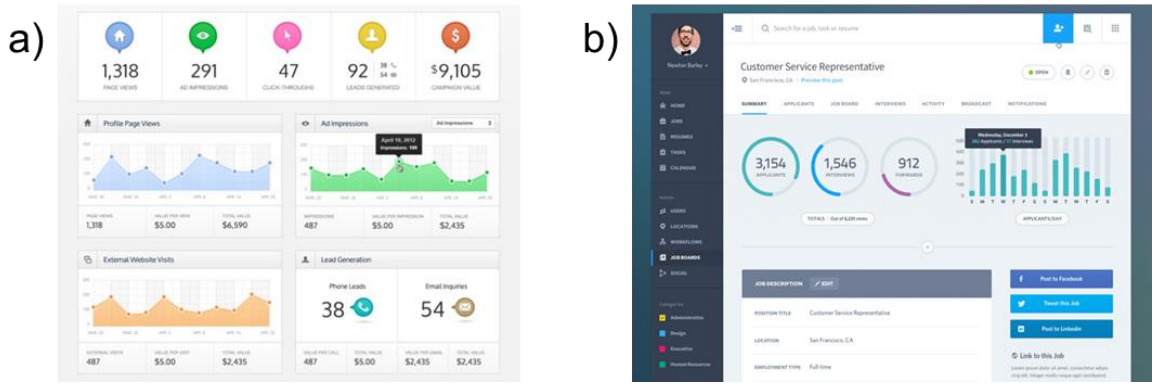

Figura 7. Ejemplos de dashboard [19]

Un *dashboard* será visualizado de una sola vez, por lo que se debe escoger la información más relevante. Este hecho permite no sobrecargar el *dashboard* de

información y facilita la presentación y el entendimiento de la misma por parte de los lectores.

Los *dashboard* convierten una gran cantidad de información incomprensible al ojo humano en datos estructurados en forma gráfica muy visuales. Esto, facilita por ejemplo obtener patrones de conducta o toma de decisiones sobre los datos representados. Por consiguiente, sería interesante realizar un diseño de la herramienta desarrollada durante este proyecto, basándose en la utilidad que aportan los *dashboard*.

3. Herramientas

Como se ha comentado en los objetivos, una parte de este trabajo se basa en estudiar y conocer las herramientas que permiten procesar gran cantidad de información. Por otro lado, también son importantes las herramientas que ayudan a visualizar gran cantidad de información como datos importantes. Además, se realiza una comparativa entre estas herramientas y los motivos por lo que se han elegido las herramientas que se utilizan en este proyecto.

3.1. Big Data

Uno de los objetivos de las tecnologías Big Data es transformar los datos en conocimiento útil para diferentes funcionalidades, como puede ser la investigación, la empresa, la educación, entre otros. Para conseguirlo, se necesitan herramientas que ayuden a procesar y analizar todos los datos recogidos. Un alto porcentaje de herramientas de Big Data son de acceso abierto (*open source*), lo que ha facilitado el aumento de popularidad de éstas. Las herramientas más utilizadas y conocidas que permiten procesamiento y análisis son *Apache Hadoop* y *Apache Spark*. En este punto se describirán las principales características de cada una, las ventajas e inconvenientes y las herramientas escogidas en función de las necesidades del proyecto.

3.1.1. Apache Hadoop

Apache Hadoop [20] es un *framework* de código abierto que permite el procesamiento distribuido de grandes conjuntos de datos en *clusters* de servidores básicos. Está diseñado para ampliar un sistema de un único servidor a miles de máquinas, cada una ofreciendo computación, almacenamiento local y con un alto grado de tolerancia a fallos.

Figura 8. Logo Apache Hadoop [20]

Las cuatro características principales de Hadoop son:

- **Redimensionable**: pueden agregarse nuevos nodos sin cambiar el formato o carga de los datos.
- **Rentable**: incorpora la computación paralela en servidores básicos, por lo que se reduce el coste de almacenamiento por terabyte.

- **Flexible**: Hadoop puede trabajar con cualquier tipo de dato, estructurado o no, provenientes de cualquier tipo de fuente. Estos datos de diversas fuentes pueden agruparse y así permitir análisis más profundos que los proporcionados por cualquier otro sistema.
- **Tolerante a fallos**: si se cae un nodo, el sistema redirige el trabajo a otra localización de los datos y continúa procesando.

Arquitectura Hadoop

La arquitectura de Hadoop (Figura 9) se puede dividir en cuatro módulos:

- *Hadoop Common*: este módulo contiene las utilidades comunes en las que se apoyan el resto de módulos. Permite abstracción a nivel del sistema de archivos o sistema operativo y contiene los ficheros necesarios para iniciar Hadoop.
- *Hadoop YARN*: es un marco de trabajo para la planificación de tareas y gestión de recursos de clúster.
- *Hadoop Distributed File System (HDFS)*: sistema de archivos distribuido que proporciona acceso de alto rendimiento para los datos de aplicación.
- *Hadoop MapReduce*: sistema basado en YARN para el procesamiento paralelo de grandes conjuntos de datos.

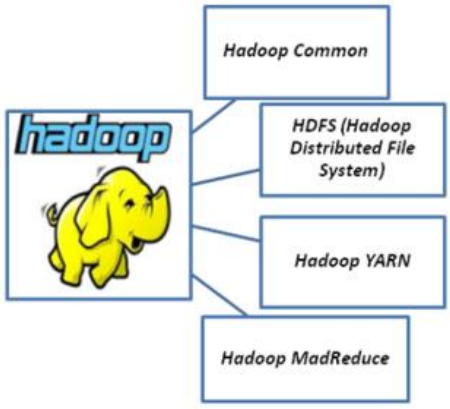

Figura 9. Arquitectura Hadoop [21]

3.1.2. Apache Spark

Apache Spark [22] es un sistema de computación distribuida a través de clusters. Fue creado en la Universidad de Berkeley en California y es considerado el primer software de código abierto que mejora los rendimientos de otros softwares privados. Este sistema es un modelo relativamente simple para escribir programas que se pueden ejecutar paralelamente en miles de máquinas a la vez. Esto revolucionó la manera de trabajar con grandes conjuntos de datos. Gracias a su arquitectura, MapReduce hace

viable la escalabilidad del sistema, ya que si los datos aumentan es posible añadir más máquinas sin aumentar el tiempo.

Figura 10. Logo Apache Spark [22]

Plataforma unificada para gestionar datos

Una de las principales características de *Apache Spark* es que es una plataforma de plataformas (Figura 11). Al unificar diversos *frameworks*, programas y herramientas facilita el funcionamiento y mantenimiento de sus soluciones. *Apache Spark* es un ecosistema que combina:

- *Spark SQL* [23]: incluye un optimizador basado en costes, almacenamiento en columnas y generación de código para hacer consultas rápidas utilizando lenguaje SQL.
- *Spark Streaming* [24]: gestiona grandes cantidades de datos en tiempo real. Esto permite analizar los datos según van entrando y gestionarlos de forma continua.
- *MLib* (Machine Learning) [25]: incluye una librería con algoritmos de clasificación, regresión, *clustering*, árboles de decisión, etc.
- *GraphX* [26]: es un *framework* de procesamiento gráfico. Proporciona una API para la elaboración de grafos con los datos. Además, tiene una librería con algoritmos de grafos.

Figura 11. Ecosistema Spark [27]

La API de *Spark* permite desarrollar código en distintos lenguajes de programación entre ellos *Scala, Java y Python* que son los más comunes como se puede ver en la Figura 12.

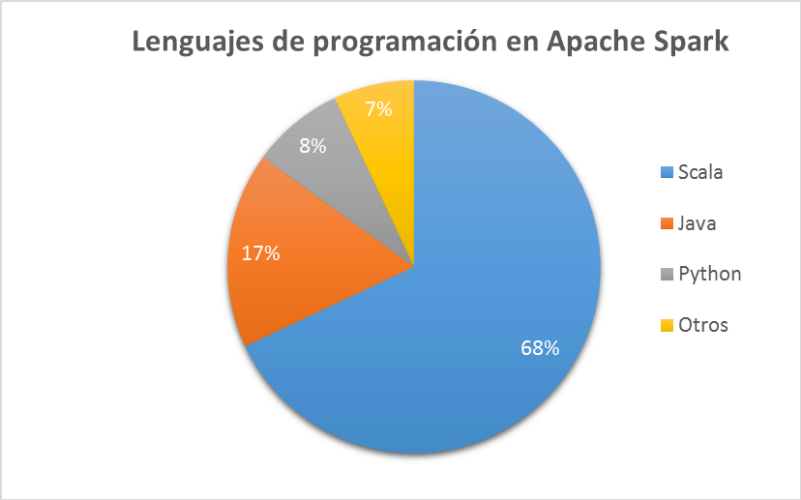

Figura 12. Lenguajes de programación en Apache Spark [28]

Funcionamiento de Apache Spark

Spark mantiene la escalabilidad lineal y la tolerancia a fallos de MapReduce, pero es más eficiente gracias a varias funcionalidades: DAG y RDD [29].

- DAG (Directed Acyclic Graph) es un grafo dirigido que no tiene ciclos, es decir, un vértice se conecta a otro, pero nunca a sí mismo. Cada tarea de Spark crea un DAG de etapas de trabajo para que se ejecuten en un determinado *cluster*. Spark con DAG no tiene que escribir en disco los resultados obtenidos en etapas intermedias del grafo, por ello es más rápido que MapReduce
- RDD (Resilient Distributed Dataset) permite a los programadores realizar operaciones sobre grandes cantidades de datos en *clusters* de una manera rápida y tolerante a fallos. Una vez que los datos han sido leídos como RDD en Spark, pueden realizarse dos tipos de operaciones.
    - Transformaciones: al aplicar una transformación se obtiene un nuevo RDD basado en el original.
    - Acciones: consiste en aplicar una operación sobre el RDD y obtener un valor como resultado, que dependerá del tipo de operación.

Para tener una alta eficiencia al aplicar transformaciones y acciones a un conjunto de datos, es recomendable almacenar los RDDs en memoria para realizar diversas acciones o transformaciones sobre un conjunto en particular y que el acceso a los datos sea más rápido que si estuvieran en disco.

Los RDDs son inmutables, una vez creados no pueden cambiar. Sobre los RDDs se aplican transformaciones para crear nuevos RDDs pero esto no significa que en cada transformación se está creando una nueva copia de los datos. Spark utiliza un

mecanismo de “evaluación perezosa” de las transformaciones. Esto significa que no se generan nuevos datos hasta que no se aplique una acción. Es en ese momento es cuando se evalúa la secuencia de transformaciones que se aplican al RDD, se combinan para optimizar y se ejecutan para generar una nueva versión de los datos. Es decir, los RDDs que se crean en cada transformación no se llegan a plasmar a no ser que una acción lo solicite, por lo que no se está creando múltiples copias del RDD.

Además de los RDDs, Spark ha añadido nuevas estructuras que permiten realizar operaciones con grandes conjuntos de datos: *Dataframes* y *Datasets* (versión de Spark 1.6) [30].

Un Dataframe es un conjunto de datos organizado en columnas con nombre. Es conceptualmente equivalente a una tabla en una base de datos, pero con optimizaciones incluidas que permiten realizar otras operaciones. Los DataFrames pueden construirse a partir de una amplia gama de fuentes como archivos de datos, tablas Hive, bases de datos externas o consultas de SparkSQL a RDDs existentes.

Un Dataset es una colección distribuida de datos que proporciona las ventajas de los RDDs (tipado fuerte, capacidad de usar potentes funciones lambda) con los beneficios del motor de ejecución optimizado de SparkSQL. La API Dataset está disponible para Scala y Java. Python no tiene soporte para dicha API, pero debido a la naturaleza dinámica de Python, muchos de los beneficios de la API de Dataset ya están disponibles.

3.1.3. Comparativa

En este apartado se va a realizar una comparativa entre las dos herramientas detalladas con anterioridad, *Apache Hadoop y Apache Spark.*

Apache Hadoop

Hadoop fue desarrollado originalmente para apoyar la distribución del proyecto de motor de búsqueda, denominado Nutch. Debido a la cantidad de información que había que indexar parecía un problema irresoluble. Gracias a que Hadoop implementa una solución basada en sistema de archivos distribuidos, se consigue procesar enormes cantidades de datos. Estos datos crecen exponencialmente, hasta convertirse en un ecosistema que engloba distintos productos de gran popularidad que hoy se utilizan para almacenar, procesar y analizar datos masivos.

¿Qué ventajas existen al utilizar Hadoop para desarrollar un proyecto Big Data [31]?

- Bajo coste: Hadoop es una tecnología de código abierto, lo que significa que las organizaciones pueden descargar e implementar el software sin tener que pagar la licencia. Esto facilita y rentabiliza que las empresas experimenten con esta tecnología sin invertir en software costoso.
- Datos soportados: Hadoop fue creado para manejar grandes volúmenes de datos. Permite almacenar archivos de mayor tamaño que los que se pueden almacenar en un nodo o en un servidor en particular.
- Manejo de nuevos tipos de datos: en la década anterior los datos almacenados por organizaciones empresariales eran datos estructurados. Actualmente, debido al Internet of things (Internet de las cosas), las redes sociales y otras nuevas fuentes de datos, es necesario almacenar grandes volúmenes de datos no estructurados. Los sistemas de gestión de bases de datos relacionales no fueron construidos para albergar este tipo de datos. Hadoop fue construido para poder almacenar este nuevo tipo de dato

¿Qué desventajas existen al utilizar Hadoop [31]?

Con respecto al almacenamiento de datos (capa HDFS):

- Latencia para el acceso de datos: la latencia de cualquier operación entrada/salida no ha sido optimizada y sistemas de archivos tradiciones (ext4, XFS) suelen ser más rápidos en estos aspectos.
- Cantidades grandes de ficheros pequeños: cada fichero, directorio y bloque ocupa un tamaño entre 150 y 200 bytes. Esto significa que millones de ficheros pequeños van a ocupar mucho más espacio en la RAM que si hay menos cantidad de ficheros, pero de mayor tamaño.
- Escribe una vez, lee varias: en HDFS los ficheros solo se pueden escribir una vez.

Con respecto al procesamiento paralelo de grandes conjuntos de datos (MapReduce):

- Es muy difícil depurar: al procesarse el programa en los nodos donde se encuentran los bloques de datos, no es fácil encontrar los fallos de código. Tampoco es recomendable utilizar funciones de escritura de logs en el código ya que eso podría suponer un gran aumento en la ejecución del proceso.

- No todos los algoritmos se pueden ejecutar de manera paralela. Por ejemplo, el algoritmo de Dijkstra, al ser un procesamiento secuencial, no se puede implementar con MapReduce.
- Latencia: Cualquier tarea de MapReduce suele tardar por lo menos 10 segundos. Por lo tanto, si el volumen de información es pequeño, es posible que Hadoop no sea la solución más rápida.

Apache Spark

El nacimiento de Spark [22] surge en la Universidad de Berkeley en 2009 y su evolución ha sido espectacular, incrementándose notablemente la comunidad y el número de contribuciones. Pero fue en 2014 cuando Spark fue acogido como un proyecto de Apache Software Foundation y se creó la compañía Databricks para dar soporte al desarrollo de Spark. Spark surge con la motivación de mejorar los problemas que presenta Hadoop.

La desventaja de Spark es que no tiene una capa de almacenamiento. En cambio, tiene APIs nativas que facilitan el desarrollo de aplicaciones para interactuar con los datos del sistema de archivos de Hadoop (HDFS), la base de datos NoSQL de código abierto HBase y la base de datos NoSQL de código abierto Apache Cassandra.

Las ventajas de Spark son:

- Procesamiento en memoria de los resultados parciales
- Soporte para múltiples lenguajes (Java, Python o Scala)
- Tolerancia a fallos implícita
- Código abierto
- Consola interactiva
- Ecosistema de Spark está formado por módulos que se utilizan para streaming, Machine Learning, acceso a datos y grafos
- Comunidad activa que facilita documentación de manera online
- Spark se integra perfectamente con Hadoop y añade nuevas funcionalidades y en muchos proyectos se almacenan datos en el sistema de ficheros de Hadoop. Además, puede funcionar con otros productos de Big Data

¿Por qué Spark en lugar de Hadoop?

Spark ha sido diseñado con el objetivo de solucionar muchos problemas que se presentaban con Hadoop. Además, es rápido y eficaz. Una de las características del diseño de Spark es soportar en memoria algoritmos iterativos que se pudiesen desarrollar sin escribir un conjunto de resultados cada vez que se procesaba un dato. Esta habilidad para mantener los datos en memoria es una técnica de computación de alto rendimiento aplicada al análisis avanzado. Esto permite que Spark tenga velocidades de procesamiento que sean 110 veces más rápidas que las obtenidas mediante MapReduce (Figura 13).

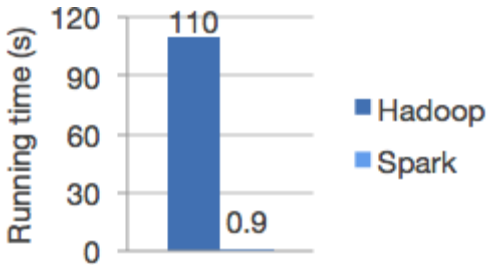

Figura 13. Velocidad de procesamiento de Hadoop y Spark [22]

Spark es un ecosistema que incluye SparkSQL, Spark Streaming, la librería MLib y el motor gráfico GraphX. Esta plataforma asegura a los usuarios la consistencia en los resultados de una manera rápida y eficaz. Con este ecosistema Spark se pueden desarrollar nuevos proyectos de Big Data con menos presupuesto y soluciones más completas.

Como conclusión se ha escogido Spark porque es un ecosistema formado por distintos módulos. Cada elemento del ecosistema tiene una función específica para afrontar conjuntos de datos más heterogéneos y sin necesidad de instalar otras herramientas o realizar un estudio de compatibilidad entre ellas. Debido a que Spark procesa en memoria los resultados parciales es cien veces más rápido que Hadoop, lo que permite procesar más cantidad de datos en menos tiempo. Además, tiene tolerancia a fallos implícita y soporta múltiples lenguajes.

Uso de Spark en este proyecto

Para el desarrollo de este proyecto se ha utilizado la versión 1.6.1, sobre la API para Python (pyspark). Para el procesamiento de los datos, además de RDD, actualmente Apache Spark soporta dos API: DataFrames y los recientes DataSets para la versión 2.0.

Debido a que se han utilizado datos semiestructurados para este proyecto, se ha trabajado con los DataFrames y SparkSQL para conseguir una alta estructuración y estandarización de la información. Para conseguir la compatibilidad con múltiples formatos y la escalabilidad de la aplicación se utilizan métodos similares a SQL para la selección, ordenación, filtrado, etc. de los datos.

Después de obtener las tablas estructuradas, se ha procedido a la limpieza de datos y el pre-procesado de éstos. Para ello, se han utilizado *pandas* [32] y *numpy* [33] librerías proporcionadas por Python [34], por la cantidad de métodos aplicados a estructuras de datos que poseen. Se ha utilizado la versión 3.5 de Python en versión Anaconda.

El uso de estas herramientas y estructuras permite mantener tiempos de ejecución rápidos. Esto es muy importante ya que se quiere conseguir un proyecto escalable, aunque actualmente hay poca cantidad de información, posteriormente se pueden añadir nuevos cursos o ediciones por lo que el volumen aumentará.

En resumen, Apache Spark ha permitido convertir datos semiestructurados en datos estructurados, obteniendo una organización que permita explotar la información de una manera más sencilla y legible.

3.2. Visualización

La visualización de los datos es uno de los puntos más importantes ya que es imprescindible para investigar, obtener patrones y definir comportamientos con antelación. Además, se puede orientar a los estudiantes para mejorar el rendimiento y para que obtengan mejores resultados durante el curso. La visualización deberá mostrar la información de manera comprensible y fácil de interpretar. Para la visualización de datos se deberán tener en cuenta las herramientas que nos pueden facilitar la representación. Entonces, se pueden destacar las más conocidas que se detallan a continuación.

3.2.1. Tableau

Es una de las herramientas de visualización de datos más utilizada por la facilidad de uso de las funciones [35]. Debido a la sencillez y rapidez de manejo, permite ser utilizada por personas que no tengan conocimientos de programación. Hay una versión gratuita que permite realizar grafos profesionales, pero con una limitación para introducir datos.

3.2.2. Gephi

Es un programa de código abierto para la visualización y consulta de grafos. Gephi da la posibilidad de agrupar nodos del grafo, colorearlos, dotarlos de tamaños proporcionales a un parámetro, etc [36]. Se carga un fichero de texto en el programa y el resultado es un grafo. Después, se puede exportar el grafo en PDF o SVG, este último permite ser reutilizado.

3.2.3. R

Es una herramienta estadística que se ejecuta en plataformas como Mac, UNIX o Windows [37]. Es un programa que puede soportar grandes volúmenes de datos y librerías. R es una herramienta que requiere conocimientos de programación.

3.2.4. D3JS

Es una librería de JavaScript para la manipulación de documentos basados en datos [38]. D3JS es un programa de código abierto que permite trabajar con los estándares web como HTML, SVG y CSS. Es rápido y permite comportamientos dinámicos de interacción y animación.

3.2.5. Comparativa

Una vez definidas las herramientas más populares o utilizadas para la visualización, se ha realizado una comparativa para escoger la que mejor se adapte a este proyecto.

¿Qué ventajas tiene cada herramienta de visualización?

Tableau

- Sencillez y rápido manejo
- No es necesario tener conocimientos de programación
- Permite representar gráficos muy profesionales

Gephi

- Visualización y consulta de grafos
- Posibilidad de agrupar nodos del grafo, colorearlos, personalizarlos, etc.
- Exportación en PDF o SVG

R

- Soporta grandes cantidades de datos

¿Qué desventajas tiene cada herramienta de visualización?

Tableau

- En la versión gratuita tiene una limitación para introducir datos
- No permite representar grafos

Gephi

- La carga de datos es mediante un fichero. Para cada grafo que se quiera representar se necesita un fichero distinto que se suba a la herramienta de manera manual. No es posible utilizar un filtro por alumno o algún parámetro, representa el fichero completo.
- No hay forma de representar itinerarios de aprendizaje

R

- Es una herramienta estadística, no visual que es lo que se está buscando para este proyecto
- Requiere conocimientos de programación
- Costosa de aprender

	Tableau	Gephi	R	D3
Costoso de aprender a utilizar	✗	✗	✓	✓
Necesarias nociones de programación	✗	✗	✓	✓
Lenguaje de programación	-	-	R	JavaScript
Limitación de datos	✓	✓	✗	✗
Permite representar grafos	✗	✓	✓	✓
Permite representación libre para realizar cualquier tipo de gráfico	✗	✗	✓	✓
Refresco estático o dinámico	dinámica	estática	dinámica	dinámica
Enfoque estadístico o de visualización	visualización	visualización	estadístico	visualización

Tabla 2. Resumen de las principales características de las herramientas de visualización

¿Por qué escoger D3JS en vez de otra herramienta de visualización?

De las herramientas estudiadas (Tabla 2), Tableu y Gephi solo permiten la representación de los datos de una manera determinada, gráficos de cualquier tipo o grafos respectivamente. Además, son herramientas estáticas ya que la información procede de un fichero que se lee completamente y del cual no se puede escoger qué leer. Por este motivo estas dos herramientas fueron descartadas.

La elección de la herramienta a escoger para realizar la visualización sería entre R y D3JS. R es una herramienta con una visualización más estática, que para este proyecto no es lo que se busca, y tiene un lenguaje propio para utilizarla (R). Por último, D3JS es la herramienta más versátil sin limitación de datos, que permite representar grafos e itinerarios de aprendizaje. Con D3JS se pueden crear y modificar diversos tipos de gráficas, representaciones a medida según las necesidades utilizando colores para que el resultado sea visual y fácil de entender. Para representar gráficos con D3JS se utilizan lenguajes como Javascript, HTML y CSS. Debido a la versatilidad, la amplia variación de tipos de gráficos y el resultado visual que se obtiene, se ha escogido D3JS para realizar la parte de visualización de este proyecto.

3.3. Otras herramientas

3.3.1. JSON

Como ya se ha comentado anteriormente, la información sobre las interacciones de los estudiantes con un MOOC se puede almacenar en ficheros JSON. En este apartado se describe el formato de este tipo de archivos.

JSON es el acrónimo de *JavaScript Object Notation* [39]. La sintaxis JSON se deriva de la sintaxis de la notación de objetos JavaScript. Los datos se escriben por pares que corresponden a la etiqueta y el valor. Cada dato se separa del siguiente por comas y cada objeto se diferencia del siguiente porque se define entre llaves (ver Figura 14).


```
"Usuario": "6307349",
"Eventos": [
  {
    "id_documento": "1.7. Recursos",
    "evento": "textbook.pdf.chapter.navigated",
    "tiempo": "2015-02-27T17:30:13.801254+00:00"
  },
  {
    "evento": "load_video",
    "tiempo": "2015-02-27T17:31:56.107121+00:00",
    "id_video": "1"
  },
  {
    "evento": "play_video",
    "tiempo": "2015-02-27T17:32:40.392104+00:00",
    "id_video": "32",
    "currentTime": "0"
  },
  {
    "evento": "pause_video",
    "tiempo": "2015-02-27T17:33:22.392781+00:00",
    "id_video": "32",
    "currentTime": "41.84"
  },
```


Figura 14. Fragmento del fichero JSON

Es un formato de texto ligero que se utiliza para el intercambio de datos. JSON es “auto-descriptivo” y fácil de entender para un humano. Además, es independiente del lenguaje de programación que se esté utilizando. El formato JSON es sólo texto, por lo que puede ser enviado fácilmente desde y hacia un servidor. Todos estos factores lo convierten en un formato muy utilizado para compartir información entre proyectos.

4. Implementación

En este apartado se describe el proceso de implementación de la herramienta de visualización. Se detalla la procedencia y estructura de los datos a visualizar, así como el procesamiento realizado a dichos datos para estructurarlos. Por último, se describen los resultados obtenidos en el proyecto.

4.1. Datos

Los datos utilizados para realizar esta herramienta proceden de la primera edición del curso “Jugando con Android - Aprende a Programar tu Primera App”, impartido por profesores de la Escuela Politécnica Superior (Universidad Autónoma de Madrid) en la plataforma edX durante los meses de febrero, marzo y abril de 2015.

El curso dura siete semanas y está formado por un conjunto de tareas que el estudiante tiene que realizar para conseguir los objetivos del curso. La plataforma realiza un registro de todos los eventos que realiza cada uno de los estudiantes con el curso y almacena esta información en *logs*.

A partir de estos *logs*, se ha extraído la información que se ha considerado relevante para cada estudiante colocándola en un formato de texto, en este caso un fichero *JSON*. En este fichero, cada estudiante ha quedado representado mediante su identificador de estudiante y una lista de eventos realizados por dicho estudiante, todos ellos ordenados cronológicamente. Este proceso de conversión de los *logs* a un fichero de *JSON* forma parte del trabajo realizado por Miguel González Gallego [2]. El archivo original que se obtiene contiene un registro de la actividad realizada por todos los estudiantes inscritos en el curso que ha realizado alguna acción en el mismo. Este proyecto comienza a partir de la información contenida en dicho fichero *JSON.*

```
"Usuario": "6307349",
"Eventos": [
  {
    "id_documento": "1.7. Recursos",
    "evento": "textbook.pdf.chapter.navigated",
    "tiempo": "2015-02-27T17:30:13.801254+00:00"
  },
  {
    "evento": "load_video",
    "tiempo": "2015-02-27T17:31:56.107121+00:00",
    "id_video": "1"
  },
  {
    "evento": "play_video",
    "tiempo": "2015-02-27T17:32:40.392104+00:00",
    "id_video": "32",
    "currentTime": "0"
  },
  {
    "evento": "pause_video",
    "tiempo": "2015-02-27T17:33:22.392781+00:00",
    "id_video": "32",
    "currentTime": "41.84"
  },
```


Figura 15. Ejemplo de fichero JSON para un estudiante

Como se puede ver en el fichero (Figura 15), para cada estudiante se detalla un registro de los eventos que ha realizado durante el curso. A su vez, todos los eventos tienen un atributo que es el *timestamp* que ordena cronológicamente los eventos e indica el momento en el que se realizaron. Por lo tanto, con este fichero se puede conocer qué evento realiza cada estudiante y cuándo se ha realizado dicho evento.

4.1.1. Tareas que componen el curso

Es importante definir las diferencias entre las tareas y los eventos. Las tareas son los distintos tipos de materiales disponibles en el curso MOOC, a través de los cuáles el estudiante adquiere conocimientos. Estas tareas son: vídeos, problemas, foro, documentos y autoevaluación (ver Figura 16). Asociado con cada tipo de tarea existen los eventos que son las acciones que se realiza sobre el recurso. Por ejemplo, los eventos relacionados con un vídeo son visualizar un vídeo (*play_video*), parar un vídeo (*pause_vídeo*) o cargar vídeo (*load_video*) (ver Figura 15). A continuación, se detalla más información sobre cada tarea y los eventos relacionados con cada una de ellas.

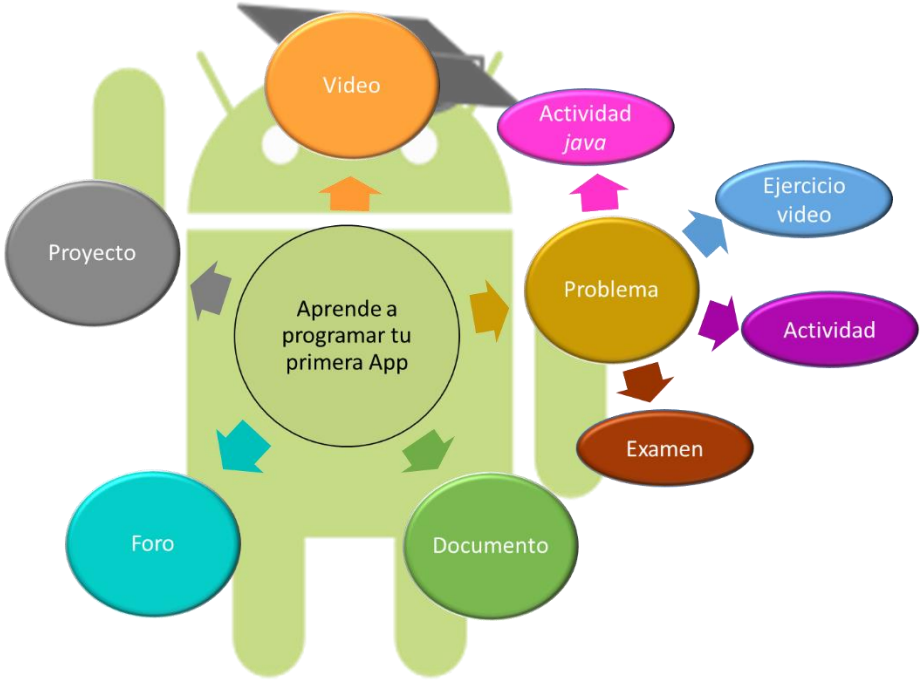

Figura 16. Tipo de tareas que forman el curso

Vídeo

Cuando un estudiante interacciona con los vídeos del curso generan una serie de eventos. Los eventos relacionados con los vídeos son pausar un vídeo, *play* vídeo, cargar vídeo, desplazarse en la línea de tiempo de un vídeo, aumentar o disminuir la velocidad de reproducción. Para cada tipo de evento, la plataforma registra el identificador del vídeo, el identificador del estudiante que ha realizado dicha acción y el momento determinado en el que lo realiza.

Foro

Cuando el estudiante interacciona con el foro del curso genera unos eventos diferentes a los de los demás recursos. Los eventos de los foros son: creación de un hilo, respuesta de un hilo, comentario o búsqueda en el foro.

Documento

Cuando el estudiante abre un documento del curso, se produce un evento relacionado con los documentos. El evento relacionado con este recurso almacena el identificador del documento, el identificador del estudiante y la fecha de lectura.

Problema

Los recursos de tipo problema se pueden clasificar en cuatro tipos (ver Figura 16). El primer tipo se refiere a los problemas que son de tipo test y se utilizan para evaluar, al principio del curso, los conocimientos de los estudiantes sobre el lenguaje de programación *Java*. El siguiente tipo engloba los problemas de tipo actividad, que son problemas genéricos relativos a los contenidos de cada semana y están formados por varios ejercicios. Otro tipo agrupa los problemas cuya respuesta está relacionada con la visualización de un vídeo. Por último, el tipo de problemas de examen que se realizan al finalizar el curso y sirve para evaluar los conocimientos adquiridos por el estudiante. Los eventos relacionados con el recurso problema almacenan el identificador del problema, el identificador del estudiante, la respuesta proporcionada y la fecha.

Proyecto o autoevaluación

Estos son los eventos que se desencadenan cuando el estudiante realiza una autoevaluación de un proyecto que ha realizado. Para este tipo de eventos quedan registrados el identificador de la autoevaluación y las preguntas de las que consta la autoevaluación. Para cada pregunta, se registra también la calificación máxima posible en cada una de las mismas, la nota que se asigna el estudiante y un *feedback*.

4.2. Desarrollo

Este proyecto se ha dividido en dos fases consecutivas: la fase de procesamiento y la fase de visualización.

4.2.1. Fase de procesamiento

El objetivo de esta fase es estructurar los datos proporcionados en el fichero JSON. A partir de este fichero se genera un conjunto de datos homogeneizado y estructurado para facilitar el acceso y la utilización de la información en la fase de visualización.

4.2.1.1. Procesamiento inicial

El procesamiento de datos consiste en ejecutar operaciones sobre la información para convertirla en información significativa. Este procesamiento puede involucrar diversas funciones como validar, clasificar, recapitular, agregar, analizar, etc. Durante el desarrollo de esta fase se han realizado validaciones, clasificaciones y análisis para estructurar la información. Esta nueva estructuración facilita extraer los datos para que la visualización sea lo más eficiente, visual y sencilla posible.

Los datos proceden del fichero JSON (ver apartado 4.1 Datos) son, la fuente de información de la que se va a nutrir este proyecto. Durante esta fase se ha realizado un estudio sobre los datos que proporcionaba el fichero y los datos que eran relevantes para este proyecto.

Revisando la información se encontraron registros repetidos, eventos que no estaban relacionados con ningún estudiante, interacciones repetidas para un mismo estudiante, etc. Estos datos no son información fiable por lo que se eliminaron.

En una segunda fase se realizó un análisis de los datos ya validados. Durante este análisis se observó que hay estudiantes que realizan interacciones diarias, estudiantes con interacciones puntuales y otros que apenas interaccionan con el curso. Solo se han considerado aquellos estudiantes que realizan más de 100 eventos de cualquier tipo durante el curso.

Una vez restringida la información a aquellos datos que aportan valor al proyecto, el siguiente reto a superar fue estructurar la información. Esta estructuración es necesaria para simplificar y optimizar el procesamiento de los datos que se utiliza para la visualización gráfica.

4.2.1.2. Procesamiento programado

Para la implementación se ha utilizado el intérprete interactivo *Pyspark* con *Apache Spark 1.0.* Se ha instalado la distribución *Anaconda* de *Python* que integra un gran número de paquetes de código abierto de *Python*. Entre estos paquetes se encuentra el intérprete interactivo *Ipython* que se ha ejecutado mediante *Jupyter Notebook*, un entorno interactivo web de ejecución de código. Además, para las consultas realizadas se utiliza *SparkSQL* (ver 3.1.2).

Para continuar con este proceso, se realizó la lectura del fichero *JSON* utilizando el intérprete interactivo *Pyspark*. Como resultado de la lectura, la información se organiza en dos columnas en una estructura abstracta de *Spark* que se conoce como *RDD* (ver 3.1.2).

A partir de este RDD se realiza el procesamiento mediante programación en *Python* con la herramienta *Spark*. Para seleccionar la información relevante para el proyecto, se llevó a cabo el procesamiento de aquellos estudiantes que interaccionaron con el curso generando más de 100 eventos. Se realizó un proceso de eliminación de aquellos estudiantes que no cumplen dicho requisito. Como resultado de este filtro se obtiene un conjunto de algo más de 1600 estudiantes.

El RDD está dividido en dos columnas: eventos e identificador de usuario, que están estructuradas como aparece en la Figura 17. El recuadro verde señaliza las características de los eventos que ha realizado un estudiante. Estos datos se representan como una estructura dentro de un *array* y contienen la información de todos los tipos de eventos. En color naranja se señala el identificador del estudiante que ha realizado esas interacciones.

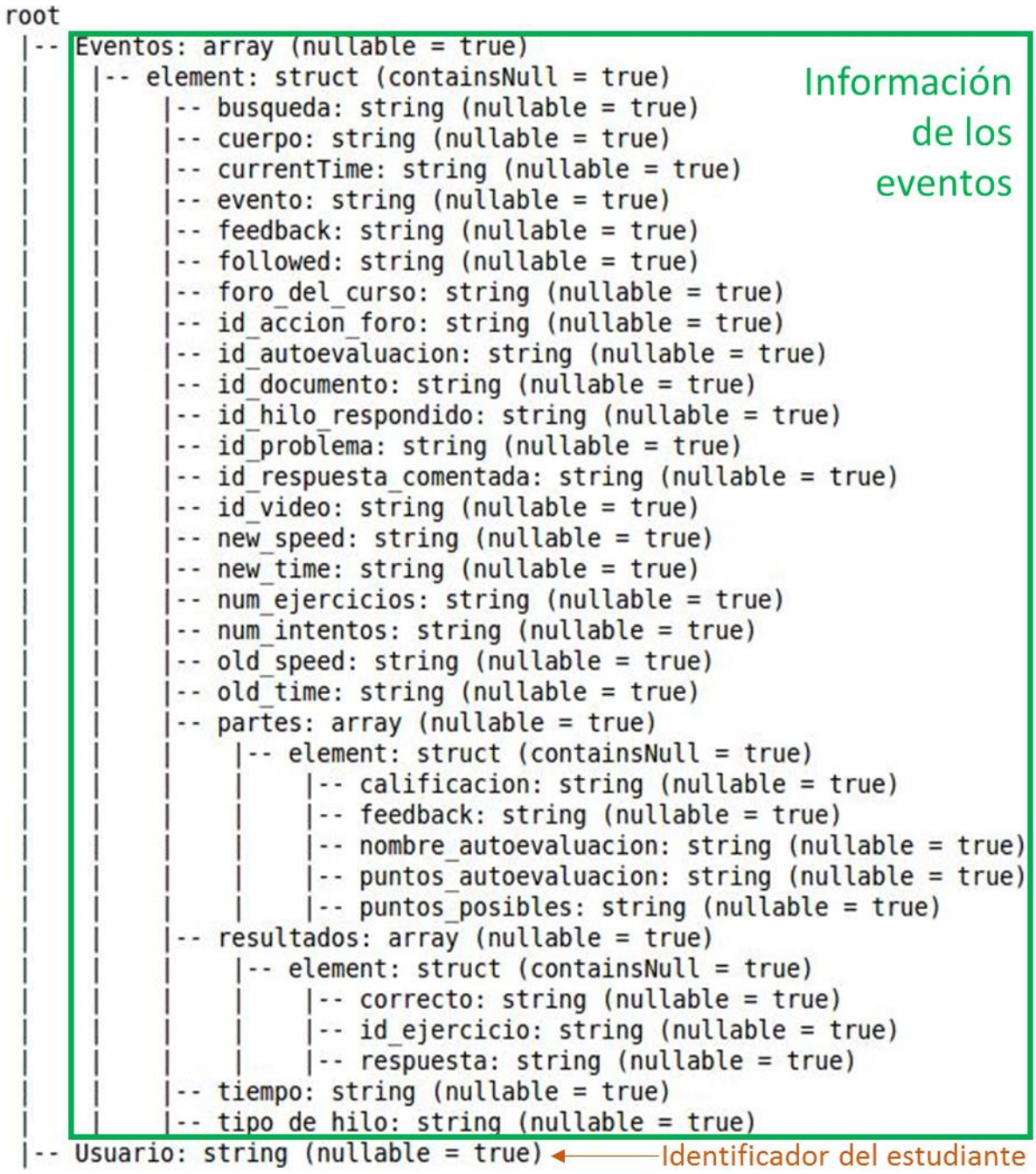

Figura 17. Esquema de la tabla resultante de la lectura del fichero JSON

La columna *Eventos* contiene todas las características de todos los tipos de eventos. Las características que se pueden observar en la Figura 17 pertenecen a diferentes eventos, por ejemplo, *búsqueda* y *cuerpo* pertenecen a los eventos de tipo foro, *id_documento* pertenece al evento de tipo documento y *partes* pertenece al evento autoevaluación, etc. En el caso de que el evento no utilice esa característica, el valor de ese campo es *null*. Si por el contrario esa característica pertenece a ese tipo de evento, el campo contiene el valor correspondiente para ese estudiante. Todos los eventos

contienen un valor de tipo fecha para la etiqueta *tiempo* que corresponde al momento en el que el estudiante realizó dicha interacción.

Como cada evento solo completa algunas de las características, para evitar valores nulos y organizar la información se decidió crear una tabla para cada tipo de evento. Los datos de cada evento se almacenan en distintas tablas, que son lo que se conoce como *dataframes* en *Spark*. La idea general es que para cada tabla se filtran aquellos eventos relacionados con cada recurso utilizando *SparkSQL.* Se genera una consulta seleccionando las características de cada tipo de evento y el resultado es un *dataframe* con toda la información de ese tipo de evento. Además, se calcula la fecha de finalización del evento. A continuación, se explica detalladamente esta idea para cada tipo de recurso.

Vídeo

Como se ha comentado anteriormente, todas las características de los diferentes eventos están mezcladas en una única columna del RDD resultante de la lectura del fichero JSON (ver Figura 17). Por ello, para poder crear una tabla para cada tipo de evento, el primer paso es seleccionar las características de cada uno de éstos. Para realizar esta selección se ha utilizado *SparkSQL*¸

"SELECT Eventos.id_video, Eventos.new_speed, Eventos.new_time, Eventos.old_time, Eventos.old_speed, Eventos.currentTime FROM rdd"

Una vez seleccionadas las características específicas de este evento, se realiza un filtro para que solo aparezcan los registros relacionados con los vídeos. Los tipos son los siguientes: *"load_video", "play_video", "seek_video", "stop_video", "pause_video", "speed_change_video"*. El código desarrollado con SparkSQL para realizar el filtro es el siguiente:

```
video = videoConcat[(videoConcat['evento']=="load_video")
        | (videoConcat['evento']=='play_video')
        | (videoConcat['evento']=='pause_video')
        | (videoConcat['evento']=='stop_video')
        | (videoConcat['evento']=='seek_video')
        | (videoConcat['evento']=='speed_change_video')]
```

Como resultado se obtiene la tabla de la Figura 18.

	evento	Usuario	tiempo	id_video	new_speed	new_time	old_time	old_speed	currentTime	tipo	tiempo_fin
0	load_video	5343437	2015-02-24 00:01:20.635487	7	NaN	NaN	NaN	NaN	NaN	V	NaN
1	play_video	5343437	2015-02-24 00:01:24.190024	7	NaN	NaN	NaN	NaN	NaN	V	NaN
2	pause_video	5343437	2015-02-24 00:01:45.718590	7	NaN	NaN	NaN	NaN	NaN	V	NaN
3	stop_video	5343437	2015-02-24 00:01:45.720604	7	NaN	NaN	NaN	NaN	NaN	V	NaN

Figura 18. Registros ejemplo tabla vídeo

Además, es necesario calcular una fecha final para calcular cuánto tiempo está interaccionando el estudiante con cada vídeo.

Para ello, se crea una tabla nueva que agrupa los registros que corresponden al mismo estudiante, sobre el mismo vídeo (registros con el mismo identificador de vídeo) y que estén continuos en el tiempo. La información relevante que se necesita es el intervalo de tiempo que el estudiante está interaccionando con el recurso de tipo vídeo, ya sea mientras se carga, se reproduce, busca un minuto en el vídeo, lo para, lo pausa o cambia la velocidad de éste. Como resultado de la agrupación se obtiene un único registro que indica el tiempo total que el estudiante ha interaccionado con el vídeo (Figura 19).

	evento	Usuario	tiempo	id_video	new_speed	new_time	old_time	old_speed	currentTime	tipo	tiempo_fin
0	video	5343437	2015-02-24 00:01:20.635487	7	NaN	NaN	NaN	NaN	NaN	V	2015-02-24 00:01:45.720604

Figura 19. Agrupación de la información sobre el uso del vídeo por el estudiante

Foro

Para los eventos del foro seguimos el mismo patrón que para los vídeos. Se seleccionan las características relacionadas con el foro utilizando *SparkSQL* con la siguiente consulta:

"SELECT Eventos.busqueda, Eventos.cuerpo, Eventos.followed, Eventos.foro_del_curso, Eventos.id_accion_foro, Eventos.id_hilo, Eventos.id_respuesta_comentada FROM rdd"

Con esta consulta obtenemos las características, pero solo se quieren guardar aquellos registros que sean interacciones con el foro. Para ello, se filtran aquellos registros del log cuyo tipo de evento contenga las siguientes definiciones: *"edx.forum.thread.created","edx.forum.response.created", "edx.forum.comment.created", "edx.forum.searched"*. Estos tipos se corresponden con eventos de creación de hilos, respuesta a hilos, comentarios a respuestas y búsquedas

en foro, respectivamente. Para realizar este filtro se ha utilizado *SparkSQL* y el código es el que aparece a continuación.

```
foro = foroConcat[(foroConcat['evento']=="edx.forum.thread.created")
| (foroConcat['evento']=='edx.forum.comment.created')
| (foroConcat['evento']=='edx.forum.response.created')]
```

Como resultado de la selección y el filtro se ha creado una tabla que contiene las interacciones relacionadas con el foro como se puede ver en la Figura 20.

	evento	Usuario	tiempo	busqueda	cuerpo	followed	foro_del_curso	id_accion_foro	id_hilo_respondido	tiempo_fin	tipo
0	edx.forum.thread.created	6419893	2015-03-10 13:09:17.872363	NaN	NaN	NaN	NaN	NaN	NaN	2015-03-10 13:19:17.872363	F
1	edx.forum.comment.created	6419893	2015-03-10 14:53:54.758131	NaN	NaN	NaN	NaN	NaN	NaN	2015-03-10 15:03:54.758131	F

Figura 20. Dos registros a modo de ejemplo de la tabla foro

Documento

Siguiendo la misma lógica que para las tareas anteriores, primero seleccionamos las características del evento:

"SELECT Eventos.id_documento FROM allTableDfr"

A continuación, filtramos los registros relacionados con los documentos. Estos eventos vienen clasificados por el tipo de evento *"textbook.pdf.chapter.navigated"* que indica que el estudiante ha estado navegando por un documento identificado.

```
doc = docConcat[(docConcat['evento']=="textbook.pdf.chapter.navigated")]
```

Como resultado se obtiene la tabla de la Figura 21.

	evento	Usuario	tiempo	id_documento	tiempo_fin	tipo
0	textbook.pdf.chapter.navigated	5474606	2015-02-24 00:02:07.770791	1.1. El entorno de desarrollo de Android	2015-02-24 00:04:19.801970	D
1	textbook.pdf.chapter.navigated	5360253	2015-02-24 00:03:15.727770	1.1. El entorno de desarrollo de Android	2015-02-24 00:13:15.727770	D

Figura 21. Dos registros a modo de ejemplo de la tabla documento

Problema

Una vez más, primero seleccionamos las características relacionadas con los problemas, para seguidamente filtrar aquellos registros del log cuyo tipo de evento sea *"problem_check".* El código de *SparkSQL* que permite realizar estas acciones es el siguiente:

```
"SELECT Eventos.id_problema, Eventos.num_ejercicios,
Eventos.num_intentos FROM rdd"
```

problema = problemConcat[(problemConcat['evento']=="problem_check")]

Este tipo de evento ha sido dividido en cuatro subtipos diferentes que son ejercicios sobre *java*, actividades, ejercicios sobre un vídeo y ejercicios de examen.

La división en cuatro subtipos, en general, ha sido por dos motivos. El primero es por el momento del curso en el que se realiza ese tipo de evento y el segundo porque ese tipo de *"problem_check"* está relacionado con otro recurso del curso. Esto significa que los ejercicios sobre *java* (Figura 22) son ejercicios que se realizan al comienzo del curso para evaluar las nociones que poseen los estudiantes sobre el lenguaje de programación de *java.* Por otro lado, los ejercicios de examen (Figura 23) corresponden a la parte final del curso para evaluar los conocimientos adquiridos durante el curso. El subtipo de ejercicios sobre vídeos (Figura 24) son una serie de preguntas que están relacionadas con la visualización de un vídeo en particular. El último subtipo son las actividades (Figura 25) que son preguntas sobre los conceptos que se han trabajado durante toda la semana.

	evento	Usuario	tiempo	id_problema	num_ejercicios	num_intentos	tiempo_fin	tipo	semana
0	problem_check	5343437	2015-02-24 00:07:09.670677	1	1	1	2015-02-24 00:07:27.334447	J	1
1	problem_check	5343437	2015-02-24 00:07:27.334447	2	1	1	2015-02-24 00:09:38.225932	J	1

Figura 22. Dos registros a modo de ejemplo de la tabla de ejercicios sobre java

	evento	Usuario	tiempo	id_problema	num_ejercicios	num_intentos	tiempo_fin	tipo	semana
0	problem_check	5474606	2015-03-31 00:16:57.705142	135	1	1	2015-03-31 00:17:53.797859	X	7
1	problem_check	6520242	2015-03-31 00:45:10.058108	135	1	1	2015-03-31 00:48:04.754432	X	7

Figura 23. Dos registros a modo de ejemplo de la tabla ejercicios de examen

	evento	Usuario	tiempo	id_problema	num_ejercicios	num_intentos	tiempo_fin	tipo	semana	num_video
0	problem_check	6304192	2015-02-24 00:45:20.024106	18	1	1	2015-02-24 00:45:25.022235	O	1	3
1	problem_check	6304192	2015-02-24 00:45:25.022235	17	1	1	2015-02-24 00:45:27.344850	O	1	2

Figura 24. Dos registros a modo de ejemplo de la tabla ejercicios relacionados con un vídeo

	evento	Usuario	tiempo	id_problema	num_ejercicios	num_intentos	tiempo_fin	tipo	semana	num_actividad
0	problem_check	6548623	2015-02-24 03:45:55.189919	29	1	1	2015-02-24 03:46:29.244108	A	1	1
1	problem_check	6548623	2015-02-24 03:46:29.244108	30	1	1	2015-02-24 04:07:45.101946	A	1	1

Figura 25. Dos registros a modo de ejemplo de la tabla actividades

Proyecto o autoevaluación

En los eventos de autoevaluación se han seleccionado el identificador y las respuestas proporcionadas por el estudiante con el siguiente código:

"SELECT Eventos.id_autoevaluacion, Eventos.partes FROM rdd"

Además, se ha realizado el filtro de eventos que contienen la cadena *"openassessmentblock.self_assess"* como tipo de evento.

partesConcat = partesConcat[partesConcat['evento'] ==
"openassessmentblock.self_assess"]

En la columna *partes* están disponibles las respuestas proporcionadas por el estudiante y la puntuación obtenida en función de las respuestas dadas (Figura 26). Utilizando esta columna, se ha calculado la puntuación que ha obtenido el estudiante, y la puntuación máxima a obtener. Además, se calcula el tiempo de finalización de la tarea, para obtener el período que ha estado el estudiante realizando esta tarea.

	id_autoevaluacion	evento	Usuario	tiempo	partes	tipo	puntos_autoevaluacion	puntos_posibles	tiempo_fin
0	i4x://UAMx/Android301x/openassessment/bee5ad6b...	openassessmentblock.self_assess	6441582	2015-03-04 08:06:11.239467	[(Fair, , Recursos, 1, 2), (Good, , Atractivo ...	P	7	8	2015-03-04 08:16:11.239467
1	i4x://UAMx/Android301x/openassessment/bee5ad6b...	openassessmentblock.self_assess	5816514	2015-03-04 13:11:31.594503	[(Good, , Recursos, 2, 2), (Good, , Atractivo ...	P	8	8	2015-03-04 13:21:31.594503

Figura 26. Dos registros a modo de ejemplo de la tabla proyecto

La información más relevante para este proyecto son los eventos que se han dividido en varias tablas relacionadas con cada estudiante que haya realizado dicha tarea. Los filtros aplicados a cada tabla han sido consultas realizadas a la tabla principal. Para realizar estas consultas se ha utilizado también *SparkSQL*.

La idea principal de la base de datos relacional que se ha construido se basa en tres conceptos: los estudiantes, las tareas y las acciones realizadas. Los estudiantes son los usuarios que acceden a la plataforma para realizar el curso MOOC. Las tareas se corresponden con los distintos recursos disponibles del curso (video, foro, proyecto, etc.). Entre los estudiantes y las tareas se establece la relación "realiza" que registra el momento en el cual cada estudiante inicia y finaliza la tarea, además de las características de cada tarea como las respuestas dadas o el texto escrito, por ejemplo.

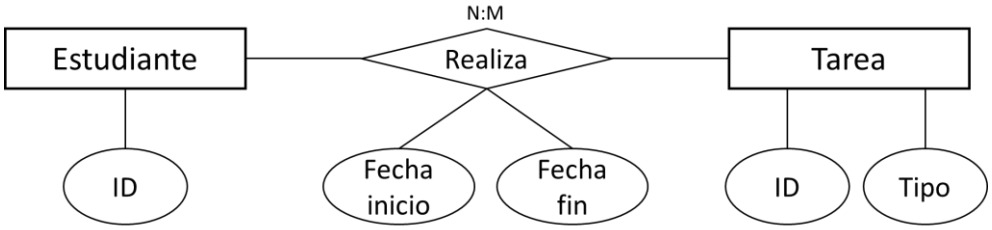

Figura 27. Idea principal de la base de datos relacional

Sin perder de vista el objetivo de esta fase, es necesario generar una base de datos que sirva como entrada de información para la generación de los gráficos en una página web. De todas las tablas que componen la base de datos, la más importante es "realiza" debido a que contiene toda la información que vamos a utilizar para representar los gráficos. Esta tabla contiene el identificador de cada estudiante, el identificador de la tarea, el tipo de tarea y las características de dicha tarea y está directamente relacionada con otra tabla que contiene todas las tareas con su identificador y su tipo.

4.2.2. Fase de visualización

El objetivo de esta fase es crear una página web que permita visualizar la información que hay almacenada en la base de datos de una manera clara, visual, amena y fácil de interaccionar. Para ello, se ha utilizado la herramienta *D3JS*, programación *JavaScript*, programación en *PHP* y la base de datos resultante de la fase de procesamiento.

La página web creada para la visualización (Figura 28), se divide principalmente en dos partes: la parte superior que contiene los filtros que se pueden emplear para visualizar la información (recuadro gris de la Figura 28) y la parte inferior que contiene los gráficos resultantes de los filtros seleccionados (recuadro morado de la Figura 28).

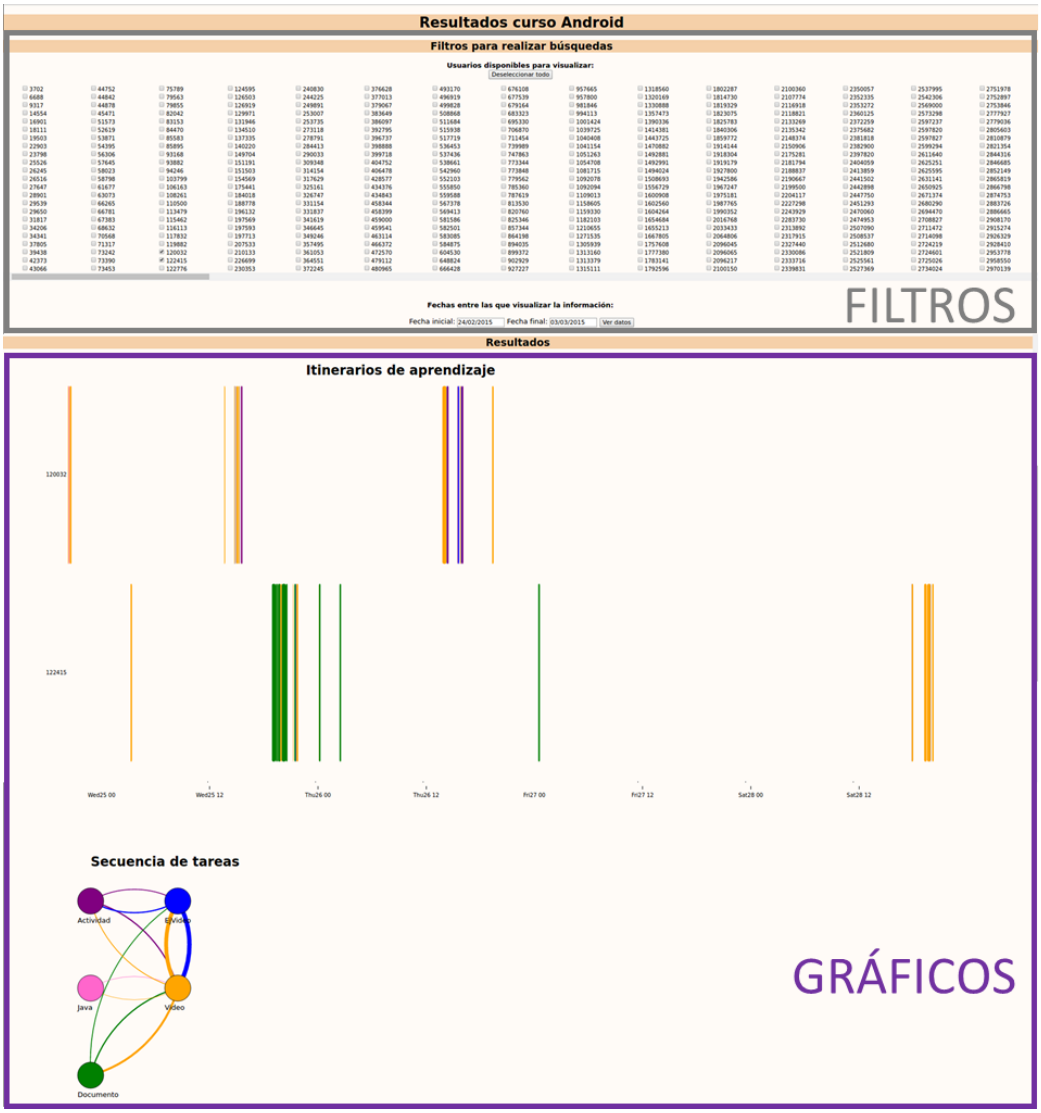

Figura 28. Página web

La parte superior contiene dos filtros (Figura 29). El primer filtro (recuadro verde de la Figura 29) sirve para seleccionar a los estudiantes sobre los que se quiere visualizar la información. Este filtro es un *checkbox* en el cuál, seleccionando la casilla a la izquierda del identificador del estudiante, queda escogido para ver los resultados de su interacción durante el curso. En la parte superior, justo encima del listado de identificadores de estudiantes, hay disponible un botón “Deseleccionar todo” que sirve para borrar los *checks* de los estudiantes seleccionados.

El segundo filtro (recuadro morado de la Figura 29), justo debajo del primero, permite seleccionar una fecha inicial y una fecha final. Estas fechas acotan el tiempo sobre el que se quiere visualizar los resultados. Este periodo puede variar desde un día hasta los tres meses que dura el curso. El resultado que se visualiza en los gráficos es

el conjunto de la selección de los dos filtros, es decir, las interacciones de los estudiantes seleccionados en el periodo de tiempo indicado.

Figura 29. Filtros de la página

Debido a la cantidad de información resultante de la selección en el filtro de fechas y el filtro de estudiantes, la representación de la información puede llegar a ser abrumadora y no comprensible. Por ello, se ha programado una limitación de selección de estudiantes, en el que hay que indicar como mínimo un estudiante y un máximo de diez. En el caso de no seleccionar a ningún estudiante o más de 10, sale un mensaje de advertencia como se puede ver en la Figura 30.

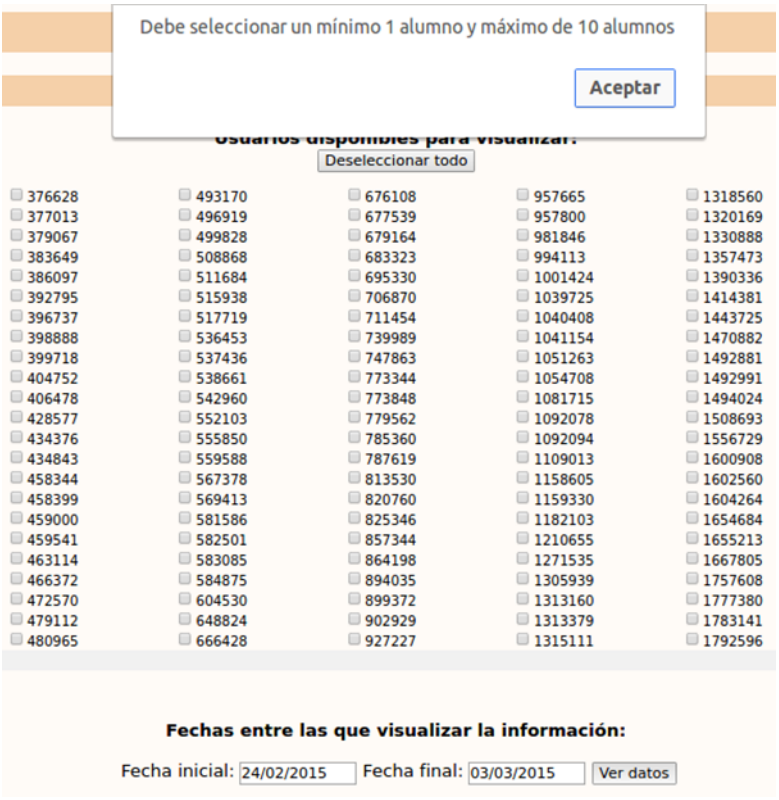

Figura 30. Mensaje de advertencia

Una vez seleccionados los estudiantes y el período de fechas, hay que pulsar el botón “Ver datos”. Al pulsar el botón se realiza todo el proceso para recuperar la información y mostrarla gráficamente. Puede ocurrir que no exista información para el periodo o estudiantes seleccionados. En este caso aparecerá un mensaje informando que no hay información disponible (Figura 31).

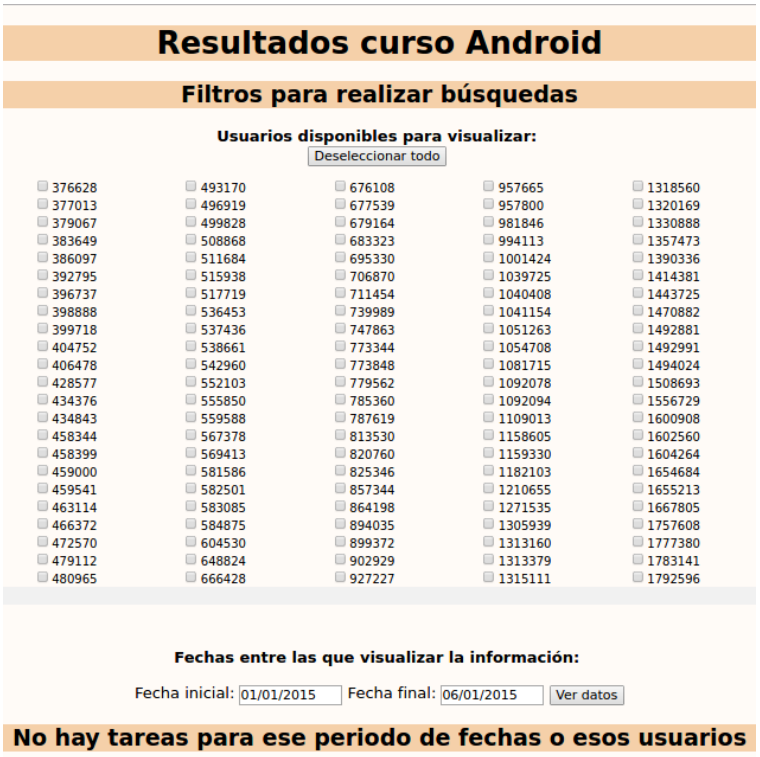

Figura 31. No hay información disponible

En caso contrario, es decir, si hay información disponible, se visualizan dos gráficas (Figura 32) que representan la interacción de los estudiantes con las tareas del curso.

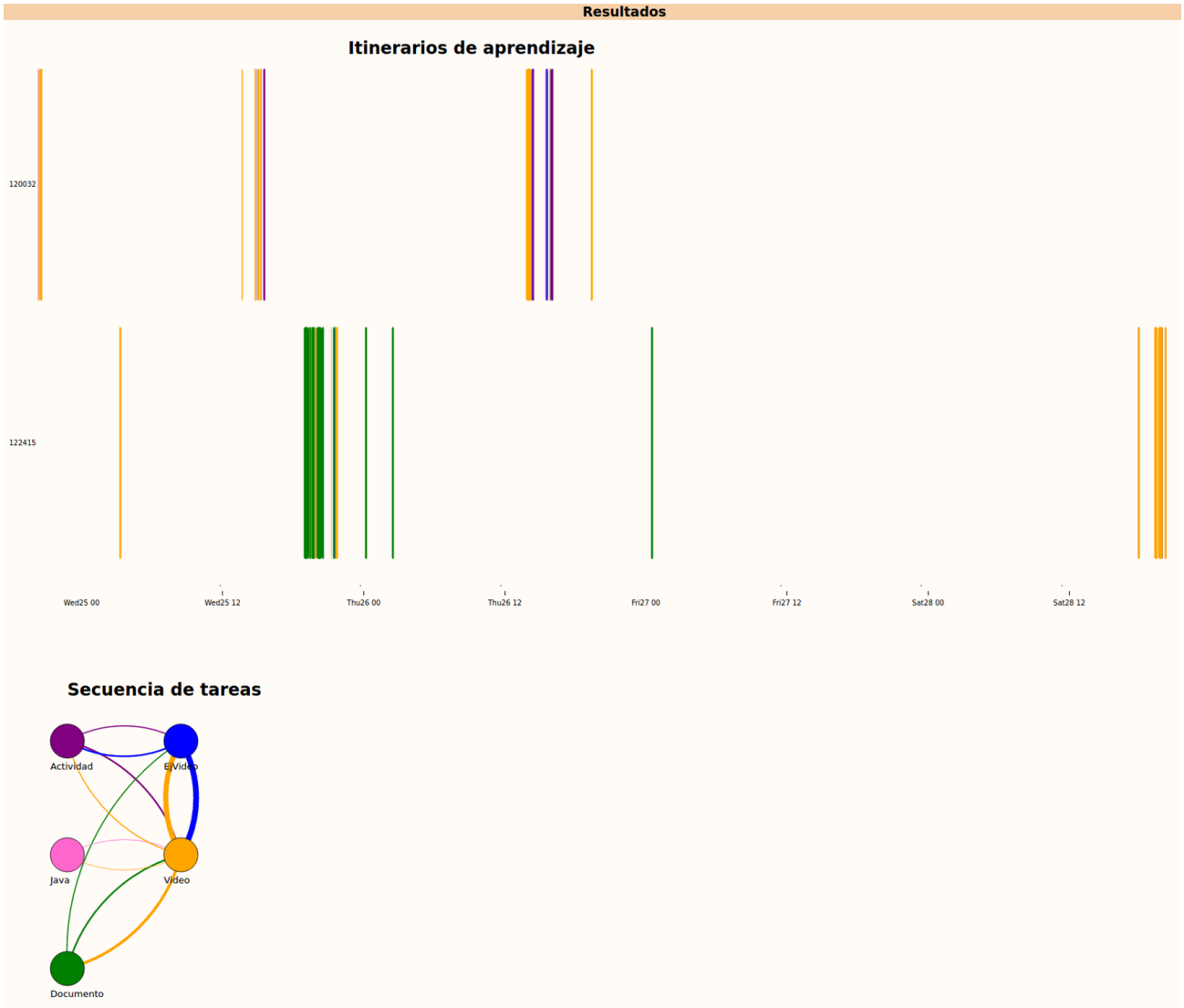

Figura 32. Graficas resultante de los filtros

4.2.2.1. Itinerario de aprendizaje

El primer gráfico que se visualiza, representa los itinerarios de los estudiantes. Se ha llamado “itinerarios de aprendizaje” al conjunto de tareas que van realizando los estudiantes en un periodo de tiempo, es decir, a la secuencia de las tareas que realizan y las transiciones de un tipo a otro. Cada tarea se representa con un color (ver Figura 33) y la longitud de cada barra representa el tiempo que el estudiante ha estado realizando ese tipo de tarea.

Figura 33. Tipo de tareas y su color correspondiente

Este gráfico permite visualizar la dedicación de tiempo a cada tarea de una manera muy visual. Al mostrar los resultados obtenidos con diversos estudiantes, se ha observado que, en general, los estudiantes no estaban más de media hora interaccionando con cada recurso. Por este motivo, este gráfico parece tener limitaciones si los filtros seleccionados son muy amplios (ver Figura 34).

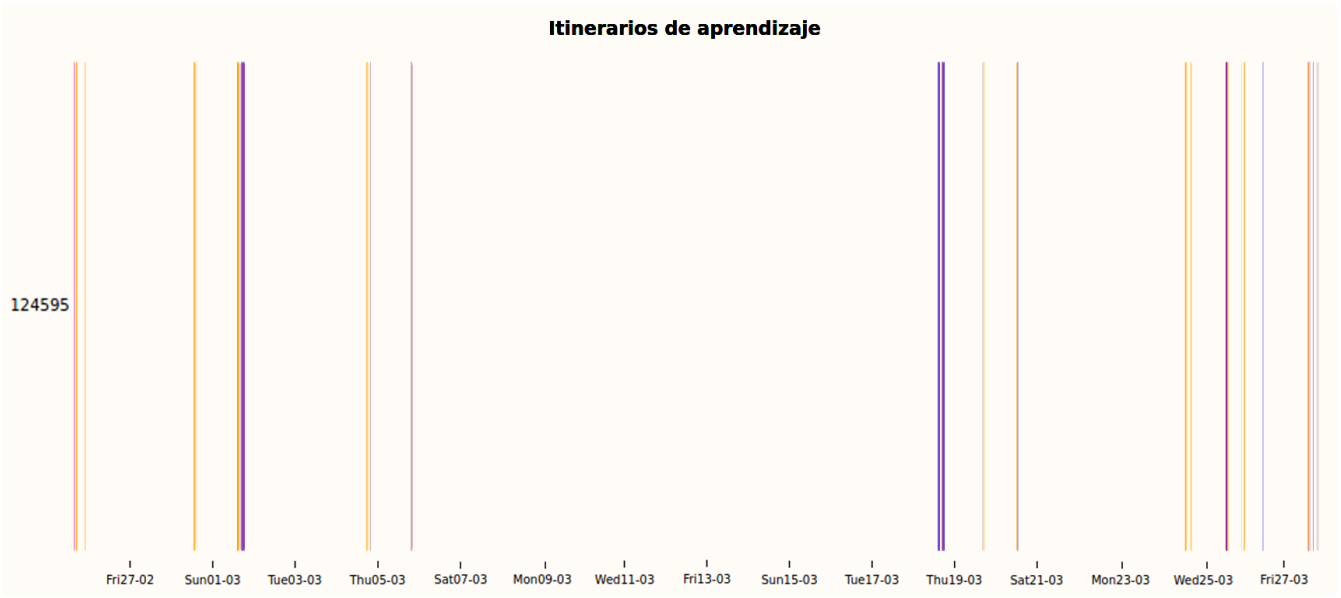

Figura 34. Itinerario de aprendizaje de un estudiante durante un mes

En cambio, si el periodo de tiempo seleccionado en el filtro se reduce por ejemplo a un día o una semana, está información es interesante (ver Figura 35). En este gráfico se puede ver el tiempo que dedican los estudiantes a cada recurso, los períodos en los que trabajan (los fines de semana o de lunes a viernes), los tipos de recursos que más utilizan, si hay períodos de días, semanas o meses que el estudiante no entra en la plataforma, etc.

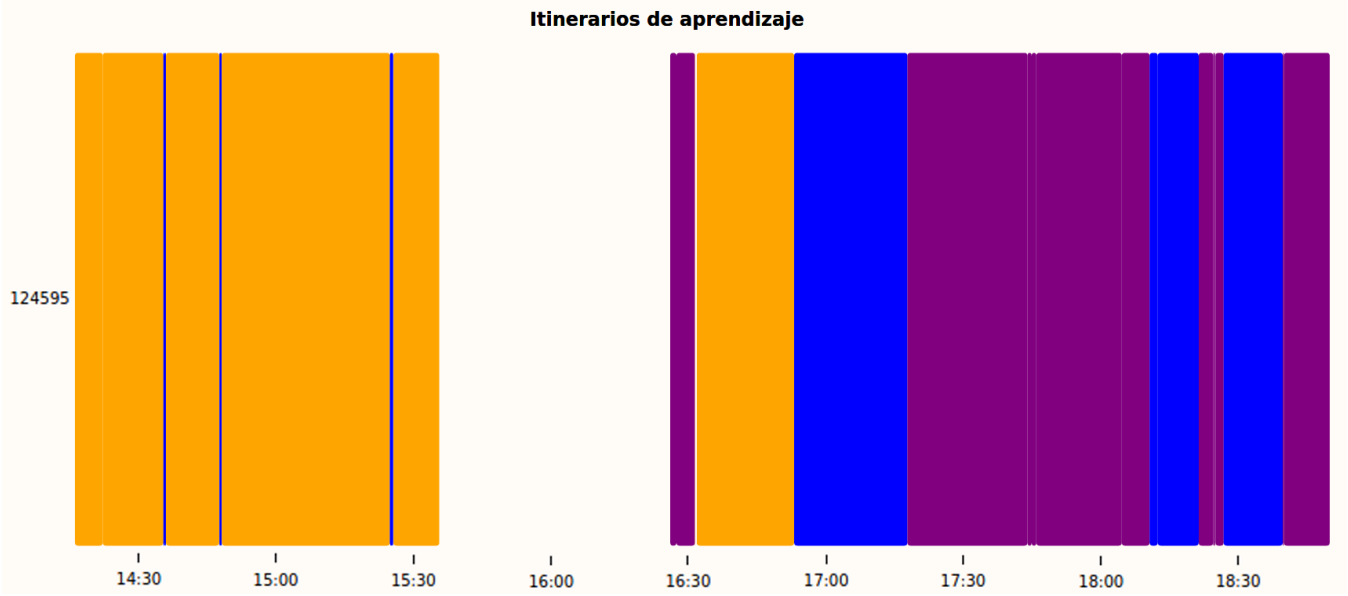

Figura 35. Itinerario de aprendizaje de un estudiante durante un día

De todas formas, debido a la limitación del gráfico anterior se ha pensado en la realización de otro gráfico para conseguir una visualización en un periodo corto o extenso de tiempo.

4.2.2.2. Secuencia de tareas

El segundo gráfico representa la secuencia entre las tareas. Para representar dicha secuencia se ha diseñado un grafo. Cada nodo del grafo representa un tipo de tarea y las aristas indican que existen estudiantes que primero han realizado la tarea de dónde parte la arista, y a continuación, a la que llega la arista. Las aristas no tienen dirección porque la dirección se ha representado con un color. Las aristas que corresponden con el mismo color de un nodo, indican que ese nodo es el origen. Además, cada arista tiene un grosor distinto que indica la frecuencia con que se ha realizado esta secuencia entre los estudiantes seleccionados. El objetivo de este gráfico es que, a simple vista, se puede ver la secuencia más común entre los distintos recursos disponibles en el curso.

Con estas gráficas se consigue el objetivo de permitir visualizar la información de una manera sencilla y clara. Además, gracias a los filtros programados se permite interaccionar y poder modificar el periodo de tiempo o el conjunto de estudiantes que se quiere visualizar.

El grafo se ha desarrollado utilizando la librería D3JS. Esta librería facilita funciones para desarrollar gráficos de diversos estilos que después se pueden personalizar o modificar según las necesidades. En la Figura 36 aparece un ejemplo de grafo básico que consiste en dos nodos unidos por una arista. Los elementos se representan sobre un contenedor *SVG* y, a continuación, se dibujan los nodos y las aristas dentro del *layout*.

Figura 36. Grafo básico realizado con D3JS [40]

A partir del código correspondiente al grafo básico se empezaron a realizar modificaciones para adecuar el grafo a la información que se quiere observar. La personalización del código comenzó por generar tantos nodos como tipos de recursos existen en el curso MOOC. Para ello, se realizó la lectura de un fichero de prueba que contiene distintos tipos de recursos y cómo los estudiantes interaccionan con ellos. Se asignó un color distinto a cada tipo de recurso del fichero y se indicó el nombre del tipo de recurso dentro del nodo. El resultado se puede visualizar en la Figura 37.

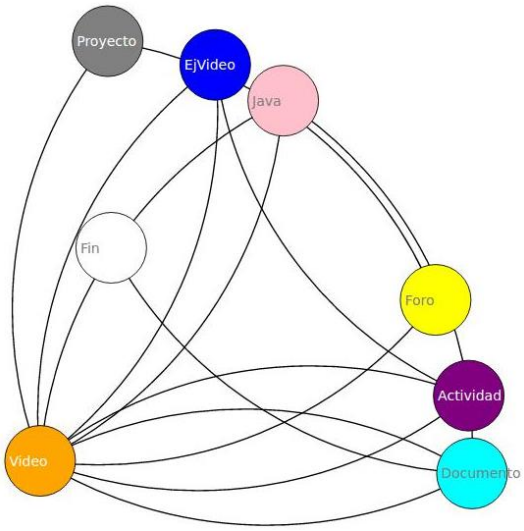

Figura 37. Grafo con los nodos de distintos colores

Al ejecutar el código, cada nodo se colocaba de manera aleatoria y se observa que con poca información es complicada la interpretación porque todas las aristas se entrecruzan. Esta ubicación aleatoria complica la búsqueda de los recursos y la interpretación de los datos representados.

Se modificó la visualización para que la posición de cada nodo quedara fija en una coordenada. Después de varias ejecuciones se observa que el recurso más relevante del curso son los vídeos porque todos los demás elementos se relacionan con éstos. Por este motivo, se ubicó en el centro del grafo. Además de la posición, se modificó también el grafo para colocar el nombre de cada recurso a un lado del nodo en vez de en su interior. A su vez, se modificó el grafo para que mostrase las aristas con distintos colores y grosores. Las aristas que corresponden con el mismo color de un nodo, indican que ese nodo es el origen. El grosor de la arista indica la frecuencia con que los estudiantes realizan primero una tarea y, a continuación, la tarea que está en el otro extremo de la arista.

En la Figura 38 se puede ver el resultado de las modificaciones sobre el código. En la imagen a) los nodos quedan en una posición fija facilitando la visualización de la información, mientras que en la imagen b) se puede ver que también se ha añadido color y grosor a las aristas.

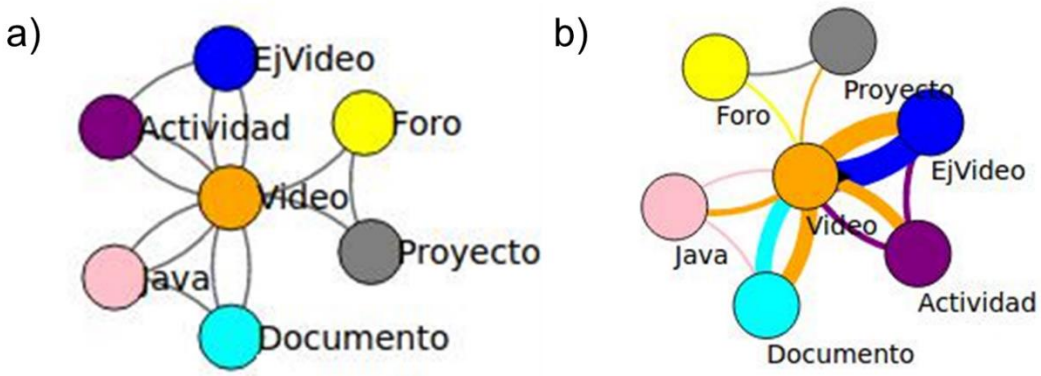

Figura 38. a) grafo con los nodos en posición fija. b) grafo con la representación de las aristas

Para finalizar las modificaciones de este grafo se amplió el tamaño global del grafo (Figura 39), lo que implica el aumento de distancia entre los nodos. Esta modificación permite que se puedan visualizar mejor las aristas que relacionan los nodos.

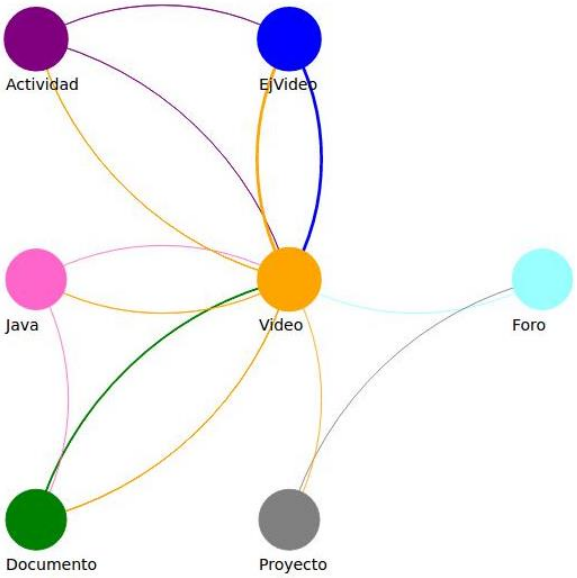

Figura 39. Representación final del grafo

Por último, se incluyó la consulta a la base de datos que contiene la información de los estudiantes matriculados en el MOOC. La comunicación entre el cliente (navegador) y la base de datos (ubicada en el servidor) se realiza mediante el uso de programación en PHP.

Hasta ahora se ha descrito la fase de procesamiento y la fase de visualización, pero también es importante explicar que la conexión entre dichas fases se realiza mediante la base de datos resultante de la fase de procesamiento. Para esta conexión se ha diseñado un modelo cliente-servidor, donde en el servidor se ha ubicado la base de datos y el cliente se conecta a dicho servidor.

El proyecto de visualización está formado por diversos módulos:

- Módulo de visualización, donde se encuentran los ficheros de estilos (*css*).
- Módulo de datos, donde están disponibles los ficheros que contienen el código *PHP* para conectar con la base de datos del servidor.
- Módulo de *D3JS*, donde se encuentran los ficheros de la librería de *D3JS.*
- Módulo de código, dónde están disponibles todos los ficheros desarrollados en *JavaScript* que utilizan los datos obtenidos mediante *PHP* y utiliza las herramientas de la librería *D3JS* para representar las gráficas. A este módulo se accede con el cliente.

En la Figura 40, se puede visualizar un esquema que resume los pasos que se ha realizado durante la implementación de este proyecto. El primer paso fue realizar un procesamiento de los datos utilizando *PySpark* con *Apache Spark*, como resultado de esta fase se obtuvo una base de datos con toda la información relevante para este proyecto. En la siguiente fase se ha realizado la parte de visualización, se ha desarrollado un módulo con código *JavaScript* que se conecta con el módulo de datos programados en *PHP* para obtener la información y la visualiza utilizando la librería D3JS.

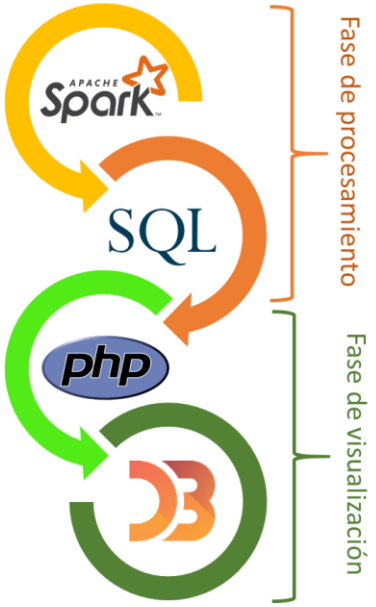

Figura 40. Esquema de las fases del proyecto

4.3. Resultados

El resultado final de este proyecto es una aplicación web para realizar un seguimiento de los itinerarios de aprendizaje realizados por los estudiantes. El contenido de la web se divide en dos partes, los filtros y las gráficas como hemos visto en el punto anterior (4.2.2 Fase de visualización).

Una vez seleccionado los estudiantes y las fechas, se pulsa el botón “Ver datos” y se obtiene el resultado en dos gráficas.

4.3.1. Itinerario de aprendizaje

La primera gráfica representa los itinerarios de aprendizaje de los estudiantes seleccionados mediante los filtros. En el eje *x* se detalla la fecha y en el eje *y* se define el identificador del estudiante, que es un identificativo único para cada estudiante del MOOC. Según la fecha inicial y final que se seleccione, el eje *x* modifica la máscara

para visualizar las fechas representadas. Si se escoge los resultados en un único día, la máscara indica las horas (ver Figura 41).

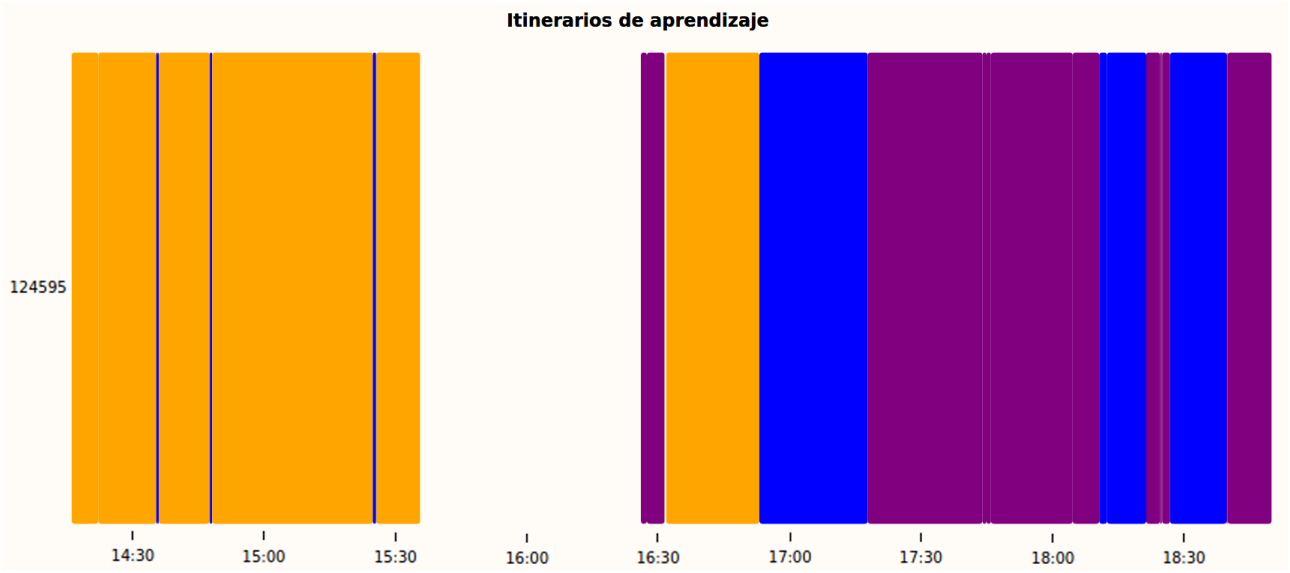

Figura 41. Resultados de un único día para un estudiante

Si los resultados son una semana, la máscara indica los siete días de la semana y cada día indica las 12 de la mañana y de la noche (Figura 42).

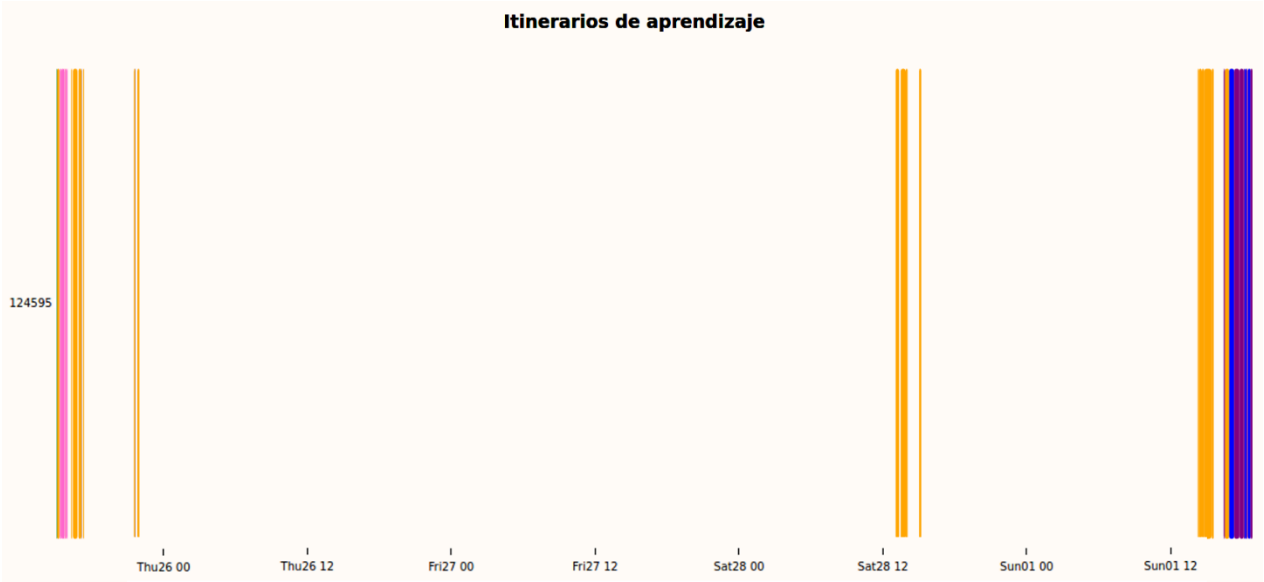

Figura 42. Resultado de una semana para un único estudiante

Si los resultados son un mes o más de un mes, la máscara indica el día de la semana junto con el día del mes (Figura 43).

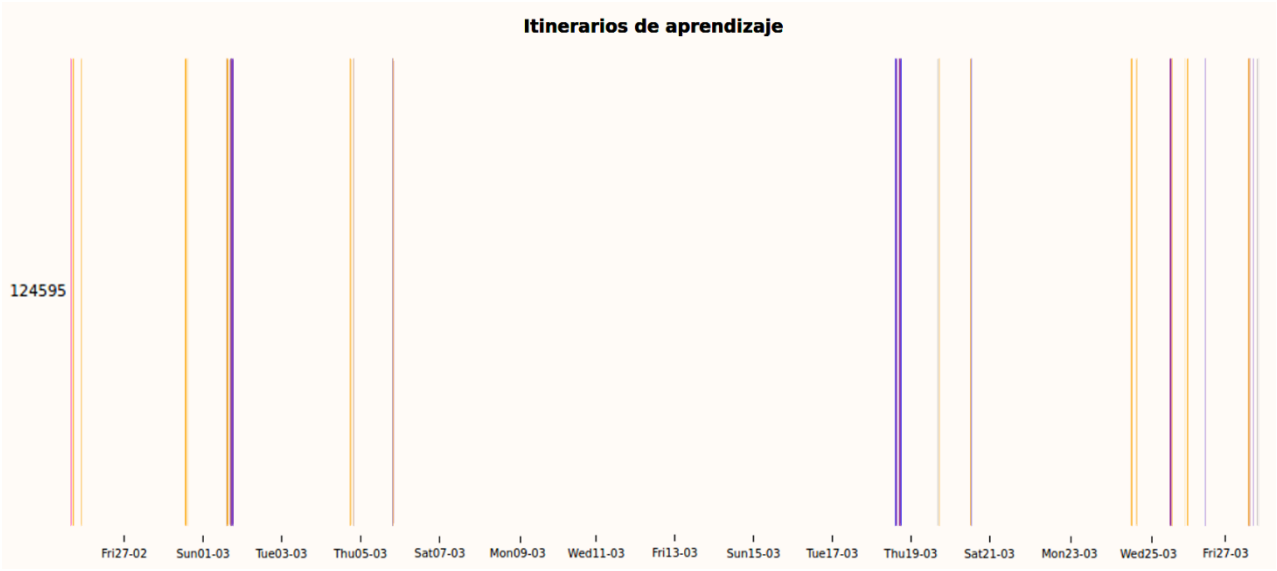

Figura 43. Resultado de un mes para un único estudiante

Cada barra indica un tipo de tarea y el grosor de éste es el tiempo que el estudiante ha realizado esa tarea. Para hacer más visual y fácil de entender la gráfica, cada tarea que se puede realizar durante el curso está representada por un color distinto como se ha comentado anteriormente (ver Figura 33).

Este gráfico permite visualizar una gran cantidad de información de un golpe de vista. Entre la información que se puede extraer del gráfico se encuentra: el tipo de tareas que realizan los estudiantes, el tiempo que le dedican, los períodos en los que suelen trabajar, si los estudiantes trabajan de manera continua o intermitente, si le dedican tiempo los fines de semana o a diario. Obtener esta información es tan fácil como indicar los estudiantes en el listado e indicar un período de tiempo entre dos

fechas. De esta forma se puede visualizar los itinerarios de aprendizaje de cada uno de ellos (ver

Figura **44**).

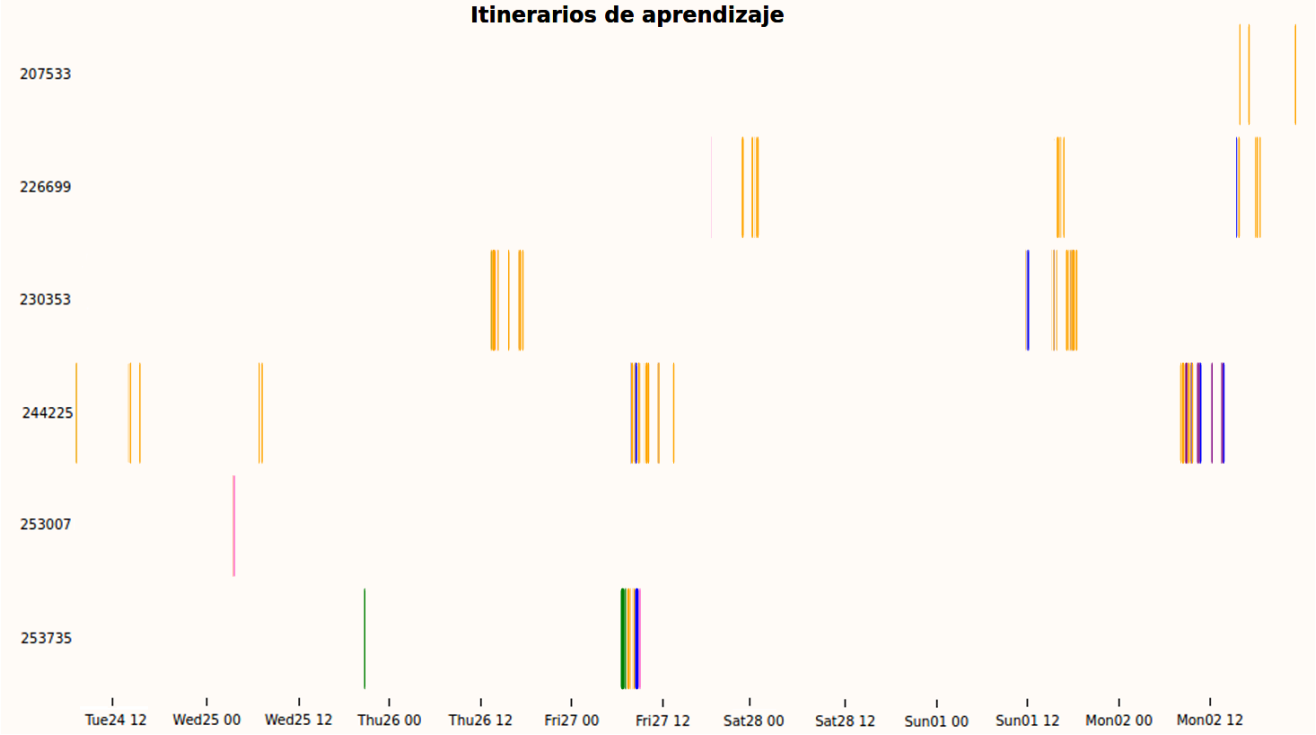

Figura 44. Itinerario de aprendizaje obtenido con la aplicación web para seis estudiantes escogidos

4.3.2. Secuencia de tareas

El segundo gráfico es un grafo donde los nodos representan las tareas pertenecientes al MOOC (vídeos, actividades, foro, etc.) y las aristas indican la secuencia temporal entre las distintas tareas. Cada nodo y arista tiene un color que es el color representativo de esa tarea. Para facilitar la visualización, este color es el mismo que en el gráfico de itinerarios de aprendizaje descrito anteriormente (Figura 33**¡Error! o se encuentra el origen de la referencia.**). Si existe una arista entre dos nodos significa que con una cierta frecuencia los estudiantes realizan una tarea y a continuación la otra.

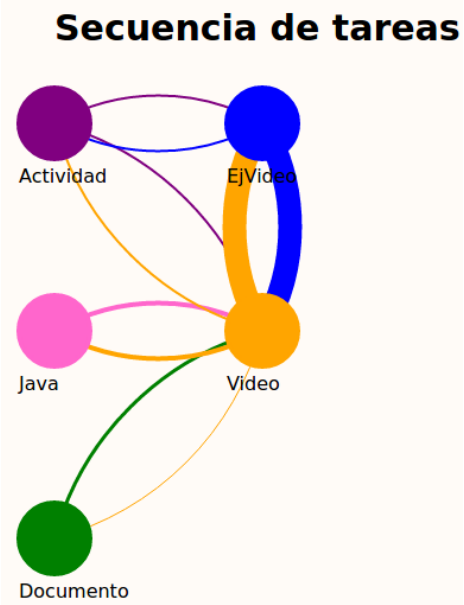

Figura 45. Grafo obtenido con la aplicación web para tres estudiantes

Si se realiza zoom sobre la Figura 45 y se focaliza en la tarea de ejercicios de Java y visualización de vídeo, se visualiza la Figura 46. En esta figura se observa un nodo rosa que representa las actividades de tipo Java y un nodo naranja que representa las actividades de visualización de videos. Entre los dos nodos se puede ver una arista rosa y otra naranja, la arista rosa representa la frecuencia con que los estudiantes realizan un ejercicio de tipo Java y a continuación visualizan un vídeo. Por el contrario, la arista naranja representa la frecuencia con la que los estudiantes visualizan vídeos y después realizan ejercicios de Java.

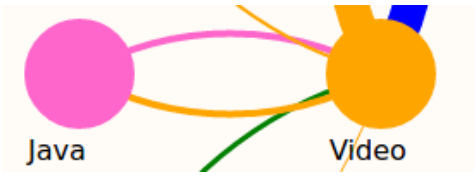

Figura 46. Zoom en el grafo sobre las tareas Java y Video

Además, mediante el grosor de la arista que une los dos nodos, se representa la frecuencia con que se ha realizado esa secuencia temporal. Si nos fijamos en la Figura 45, se puede determinar que los recursos entre los que más se ha interactuado en esas fechas y esos estudiantes, son los vídeos y los ejercicios relacionados con esos vídeos. Deducimos esto por el grosor de las aristas que es mayor que el del resto de las aristas que aparecen en el nodo. También se observa que las actividades de *java* y la visualización de vídeos también están relacionadas. En cambio, en este caso los vídeos y las actividades no son tareas que hayan ido consecutivas para estos estudiantes.

La Figura 45 corresponde a fechas iniciales del curso y solo se han seleccionado tres estudiantes. Si queremos visualizar los tres meses de curso o si queremos ver la información de diez estudiantes en amplio período, el resultado puede ser un grafo difícil de visualizar (Figura 47). Puede ocurrir que las aristas entre los diversos nodos sean muy gruesas o que el grafo se convierta en un grafo completo (cada par de vértices está conectado por una arista) y esto dificulte la visualización de la secuencia. Además, dependiendo de la frecuencia con que ocurre una secuencia entre dos tareas se representa el grosor de la arista y también puede dificultar la visualización de las aristas que tienen menor frecuencia.

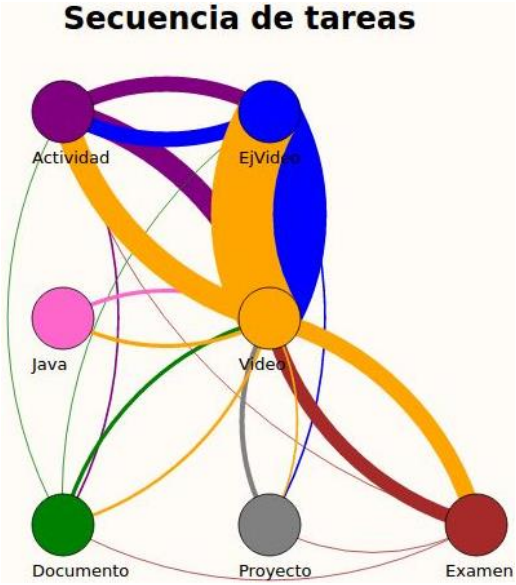

Figura 47. Ejemplo de un grafo con todas las tareas y con diversas secuencias entre ellas

Para realizar observaciones sobre los comportamientos de los estudiantes, es necesario que sea evidente la secuencia de tareas que realizan. Para facilitar esta visualización, se ha desarrollado una mejora sobre el grafo que consiste en subrayar de un color llamativo el nodo y las aristas cuyo origen es dicho nodo para visualizar las conexiones. Para visualizar el grafo con el color llamativo indicando las conexiones es necesario ubicar el ratón sobre el nodo que se quiera visualizar.

En la siguiente figura se puede visualizar el grafo desarrollado en la aplicación seleccionando a cinco estudiantes durante todo el curso. Siendo un grafo complejo como se puede ver en la Figura 47, se ha utilizado el subrayado para visualizar las secuencias de tareas. Todas las imágenes corresponden al mismo grafo, la diferencia entre ellas es que según las secuencias de la tarea que nos interesa ver se ha ido moviendo el ratón al nodo correspondiente a dicha tarea. En la Figura 48, cada imagen desde la a) hasta la g) corresponde a las secuencias de cada tipo de tarea. De izquierda

a derecha y de arriba abajo, el ratón se ha situado sobre los siguientes nodos: actividad (a), ejercicios relacionados con un vídeo (b), ejercicios sobre *java* (c), visualización de vídeo (d), lectura de documento (e), realización de proyecto (f) y realización de examen (g).

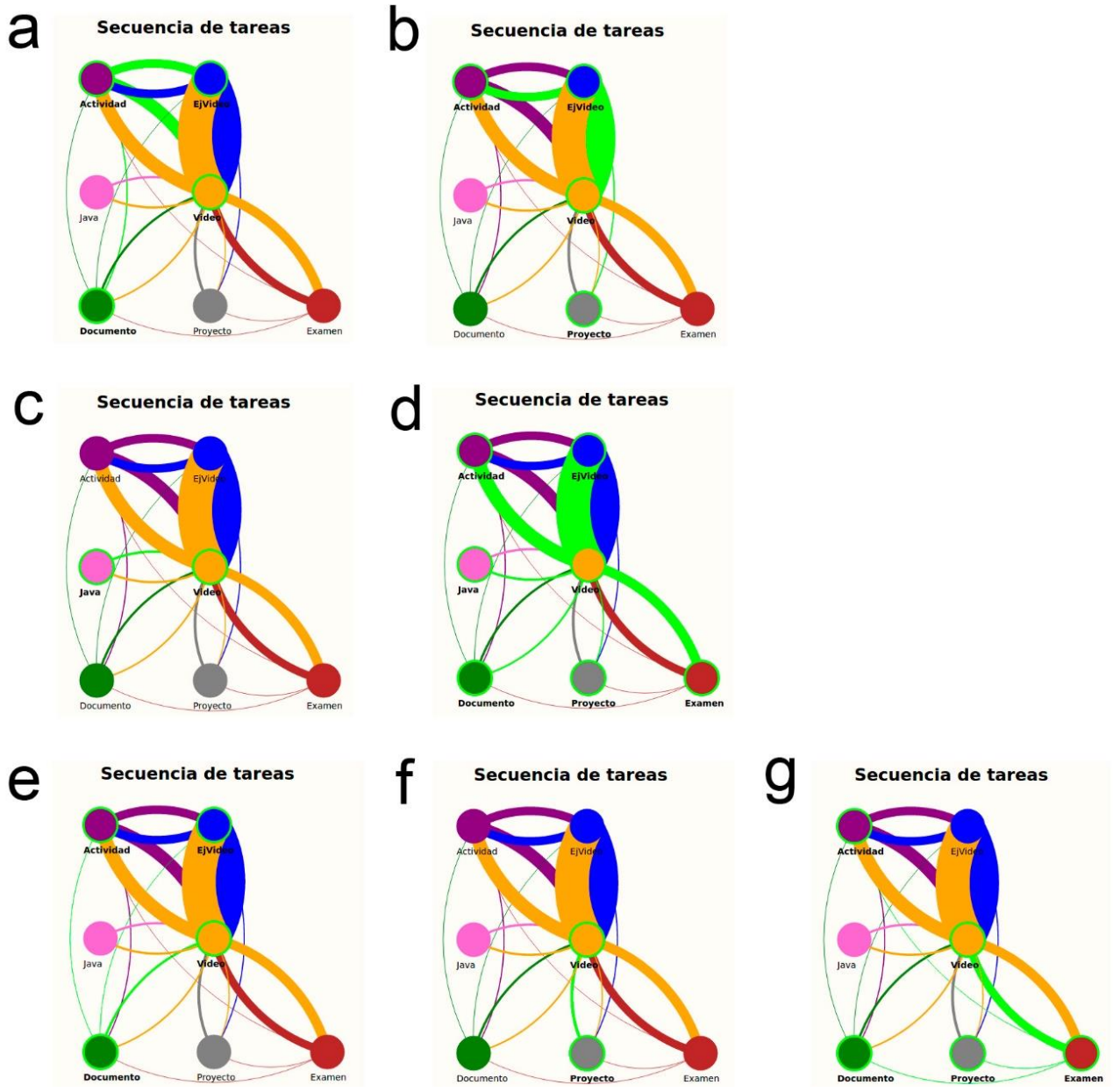

Figura 48. Ejemplo de grafo indicando las transiciones según el nodo elegido

5. Conclusiones y trabajo futuro

5.1. Conclusiones

En este trabajo se ha desarrollado una herramienta que permite observar y evaluar la interacción de los estudiantes con un curso MOOC. Para implementar esta herramienta el proceso se ha dividido en dos fases: (i) la fase de procesamiento y (ii) la fase de visualización.

En la fase de procesamiento se ha utilizado el *framework Spark* para convertir los datos almacenados en un fichero *JSON* en una base de datos estructurada, utilizando los módulos de dicho *framework* y el lenguaje de programación *Python.*

Para la conversión de estos datos a una información estructurada, primero se ha realizado la lectura del fichero *JSON* para almacenar la información en un RDD. La información del RDD está dividida en dos columnas: el identificador del usuario y el registro de todas las interacciones realizadas por ese usuario. Una vez que la información está en memoria, se divide el RDD en múltiples *dataframes* utilizando *SparkSQL*. Cada *dataframe* contiene la información imprescindible de cada tipo de evento. A continuación, esta información se organiza en una base de datos que contiene los datos relevantes para realizar la presentación de los gráficos. En la base de datos, hay una tabla con la información de los usuarios y otra tabla que contiene la información de las tareas. Además, existe otra tabla que contiene todos los datos de las tareas que ha realizado cada usuario, permitiendo relacionarlos entre sí.

La siguiente fase del proceso ha consistido en representar la información estructurada gráficamente para que sea sencillo observar y analizar los comportamientos de los estudiantes. Para el desarrollo de esta fase se ha utilizado la librería *D3JS* que proporciona un conjunto de funciones que permite representar diversas gráficas y modificarlas para adaptarlas a los requisitos de cada proyecto. También se ha utilizado los lenguajes de programación *Javascript, HTML, CSS y PHP*.

Utilizando la librería y *Javascript* se han representado dos gráficos:

(i) Un gráfico que representa la secuencia de las tareas que realiza un estudiante en un periodo de tiempo.

(ii) Un grafo donde se dibuja la relación existente entre las tareas, en función de la interacción del estudiante con éstas.

Para representar estos gráficos es necesario acceder a la información de la base de datos implementada durante la fase de procesamiento. Para consultar los datos de

las tablas se utiliza el lenguaje *PHP.* Además, los lenguajes de programación *HTML* y *CSS* junto con *Javascript* se han utilizado para crear la página web desde la cual se accede a la información gráfica.

La unión de estas dos fases da como resultado una herramienta que permite visualizar los itinerarios de aprendizaje de los estudiantes matriculados en un MOOC y la frecuencia con la que realizan las secuencias entre las diversas tareas. Asimismo, la representación de dichos itinerarios permite realizar un seguimiento de las interacciones y observar los patrones que los estudiantes llevan a cabo.

Por todo esto, la herramienta desarrollada durante este trabajo permite visualizar una información, inicialmente desestructurada y difícil de interpretar, de manera muy sencilla e intuitiva basándose en la observación de unos gráficos.

En relación a todo lo anterior, se ha conseguido el objetivo principal propuesto en este trabajo: desarrollar una herramienta que permita interpretar las interacciones de los estudiantes de una manera sencilla y fácil de comprender visualmente.

Este trabajo, además, ha contribuido al desarrollo de mis conocimientos y formación en herramientas para procesar grandes cantidades de información como es Spark y herramientas que facilitan la visualización de dicha cantidad de datos, como es la librería D3JD. Gracias al *software* utilizado para el desarrollo de este proyecto he adquirido conocimientos sobre el ecosistema de *Apache Spark.* También he aprendido a trabajar con evaluación "perezosa" aplicando diversas transformaciones sobre el mismo *RDD,* pero sin ver un resultado hasta aplicar una acción. Además, debido a la cantidad de información a procesar ha sido interesante evaluar la diferencia de tiempo entre realizar un bucle o aplicar funciones como *map, filter, sort, reduce*, etc… Esta forma de trabajar ha sido novedosa, muy eficiente y me ha proporcionado una nueva visión de la programación.

También he adquirido conocimientos sobre la librería *D3JS*. Esta librería permite representar los datos utilizando HTML, CSS, SVG y JavaScript. D3 facilita la creación de una tabla HTML a partir de unos datos o utilizarlos para crear un gráfico SVG interactivo con transiciones e interacciones. Aunque tenía conocimientos de HTML, CSS y JavaScript, ha sido necesario formarme en SVG para poder representar los gráficos y que estos sean compatibles en el uso online y flexibles.

5.2. Trabajo futuro

Este apartado define las posibles líneas futuras de investigación relacionadas con este proyecto. Estas líneas están relacionadas con (i) el análisis de nueva información, (ii) la utilización de otras máquinas para procesar una mayor cantidad de información y de una forma más eficiente y (iii) la inclusión de gráficas adicionales en la herramienta.

(i) En relación a la primera línea, sería interesante agregar los datos registrados de ediciones posteriores del MOOC. Al incorporar nuevos datos al conjunto inicial, se duplicaría la cantidad de datos por cada nueva edición incluida al conjunto. Debido a que la cantidad de datos aumentaría, la fase de procesamiento vería reducida su eficiencia.

(ii) Por este motivo, otra línea de trabajo consistiría en ejecutar el programa escrito de forma paralela en varias máquinas simultáneamente para procesar la gran cantidad de información de forma más rápida.

(iii) Por otro lado, al tener información disponible de distintas ediciones, una posible nueva línea de trabajo sería añadir nuevas gráficas a las herramientas. La incorporación de estas gráficas permitiría realizar comparativas visuales entre las interacciones de los estudiantes de las distintas ediciones de un curso.

6. Bibliografía

1. Coello Coello, C. A. *Breve historia de la computación y sus pioneros*. (Fondo de Cultura Económica, 2003).
2. González-Gallego Sosa, M. Á. & Ángel, M. Predicción y análisis de interacciones de usuarios en plataformas de enseñanza online. (2016).
3. Torres Pascual, D. Spark para Learning Analytics: análisis del abandono en cursos de formación online. (2016).
4. García Areito, L., Ruiz Corrbella, M. & Domínguez Figaredo, D. *De la educación a distancia a la educación virtual*. (Ariel, 2007).
5. Pappano, L. The Year of the MOOC. (2012).
6. Pernías Seco, P. & Luján-Mora, S. Los MOOC: orígenes, historia y tipos. *Comun. y Pedagog.* 269**–**270**,** 41–47 (2014).
7. Luján Mora, S. ¿Qué son los Recursos Educativos Abiertos? (2013).
8. Daniel, J. Making Sense of MOOCs: Musings in a Maze of Myth, Paradox and Possibility. *J. Interact. Media Educ.* 2012**,** 18 (2012).
9. Massive Open Online Course (MOOC) a Promising Option for Distance Learning | CalSWEC. Available at: http://calswec.berkeley.edu/massive-open-online-course-mooc-promising-option-distance-learning.
10. López Zamorano, C. Los MOOC como una alternativa para la enseñanza y la investigación. in (2013).
11. Guía del profesor para la planificación, diseño e impartición de MOOCs - Tabla de Contenidos. Available at: http://docubib.uc3m.es/MOOCS/Guia-metodologica-MOOC-Wimba/index.htm.
12. Shah, D. By The Numbers: MOOCS in 2015 — Class Central. (2015). Available at: https://www.class-central.com/report/moocs-2015-stats/.
13. Coursera. Available at: https://about.coursera.org/.
14. edX. Available at: https://www.edx.org/about-us.
15. Barranco Fragoso, R. ¿Qué es Big Data? *2012* Available at: https://www.ibm.com/developerworks/ssa/local/im/que-es-big-data/.
16. Stock Fotos, Imágenes, Vídeos, Vectores | iStock. Available at: http://www.istockphoto.com/es.
17. Ortoll, E. Big Data se escribe con V. *Rev. los Estud. Ciencias la Inf. y la Comun.* 37**,** (2014).
18. Few, S. *Information dashboard design : the effective visual communication of data*. (O'Reilly, 2006).
19. Rivas, M. 15 ejemplos de diseños de dashboards web - Blog de Diseño Web Vida MRR. Available at: http://www.vidamrr.com/2015/12/15-ejemplos-de-disenos-de-dashboards-web.html.
20. Welcome to Apache™ Hadoop®! Available at: http://hadoop.apache.org/.
21. Arquitectura Hadoop | El entorno de Hadoop. Available at: https://elentornodehadoop.wordpress.com/tag/arquitectura-hadoop/.

22. Apache Spark™ - Lightning-Fast Cluster Computing. Available at: http://spark.apache.org/.

23. Spark SQL & DataFrames | Apache Spark. Available at: http://spark.apache.org/sql/.

24. Spark Streaming | Apache Spark. Available at: http://spark.apache.org/streaming/.

25. MLlib | Apache Spark. Available at: http://spark.apache.org/mllib/.

26. GraphX | Apache Spark. Available at: http://spark.apache.org/graphx/.

27. What is Apache Spark? Available at: https://databricks.com/spark/about.

28. The Apache Spark Open Source Project on Open Hub. Available at: https://www.openhub.net/p/apache-spark.

29. Karau, H. *et al. Python Notes / Cheat Sheet*. *Igarss 2014* 1**,** (2015).

30. Spark SQL and DataFrames - Spark 2.1.1 Documentation. Available at: http://spark.apache.org/docs/latest/sql-programming-guide.html.

31. Davis, J. Hadoop: Pros And Cons For Enterprise Users - InformationWeek. *Inf. Week* (2016).

32. Python Data Analysis Library — pandas: Python Data Analysis Library. Available at: http://pandas.pydata.org/.

33. NumPy — NumPy. Available at: http://www.numpy.org/.

34. McKinney, W. *Python for Data Analysis*. (O'Reilly Media, 2012).

35. Business Intelligence and Analytics | Tableau Software. Available at: https://www.tableau.com/.

36. About. Available at: https://gephi.org/about/.

37. R: What is R? Available at: https://www.r-project.org/about.html.

38. D3.js - Data-Driven Documents. Available at: https://d3js.org/.

39. JSON Introduction. Available at: https://www.w3schools.com/js/js_json_intro.asp.

40. Understanding D3.js Force Layout - 1: The Simplest Possible Graph - bl.ocks.org. Available at: http://bl.ocks.org/sathomas/11550728.