Departamento de Ingeniería Telemática

PROYECTO FIN DE CARRERA

Visualización comparativa de niveles de actividad para promover la reflexión del alumnado

Autor: David Sánchez de Castro

Tutor: Derick Leony

Leganés, Febrero de 2013

Título: Visualización comparativa de niveles de actividad para promover la reflexión del alumnado

Autor: David Sánchez de Castro
Director: Derick Leony

EL TRIBUNAL

Presidente: _____

Vocal: _____

Secretario: _____

Realizado el acto de defensa y lectura del Proyecto Fin de Carrera el día __ de _____ de 2013 en Leganés, en la Escuela Politécnica Superior de la Universidad Carlos III de Madrid, acuerda otorgarle la CALIFICACIÓN de

VOCAL

SECRETARIO PRESIDENTE

Agradecimientos

Quiero agradecer y dedicar este proyecto principalmente a mi novia, María Martín Facal, por estar a mi lado en los buenos y malos momentos. Junto a ella he pasado toda la carrera y sin su apoyo no me hubiera resultado tan fácil llegar a escribir hoy estas palabras. Vivir con ella ha sido la mejor decisión que he tomado en toda mi vida. María ha sabido comprender todos los momentos en que el estudio nos ha mantenido separados y aun así me ha apoyado en cada paso que he ido dando durante todo este tiempo. No ha sido fácil trabajar y estudiar un segundo ciclo y, a la vez, sacar tiempo para seguir creciendo como pareja.

Agradezco a mis padres José A. Sánchez Sánchez y Montse de Castro Corral por el apoyo constante e incondicional que me han dado siempre; ellos me animaron a estudiar este segundo ciclo y el hecho haber llegado aquí es el resultado de que hacerles caso fue lo mejor que pude llegar a hacer. Siempre han estado conmigo, motivándome y dándome una razón para seguir estudiando, me han apoyado en todo y no me han dicho nada que me hiciera flojear en ningún momento. Tampoco se me olvida mi querido hermano Alberto, con esos toques de humor que involuntariamente tenía y que te hacen cambiar la cara en los momentos más difíciles.

A mi tía Mercedes de Castro Corral por la lectura final que le ha dado al proyecto para intentar darle ese sentido que sólo la mejor editora sabría dar.

A mis tutores, en un principio Luís de la Fuente Valentín y luego Derick Leony por brindarme la oportunidad de hacer este proyecto, por su guía y su consejo, y por ofrecerme la oportunidad de retomar mi formación profesional como investigador en la universidad. Tampoco quiero olvidar a Abelardo Pardo por ser la referencia principal en este proyecto, porque de sus ideas salieron muchas de las decisiones que todo el proyecto GLASS ha seguido, y porque siempre encontraba solución a las dudas que se planteaban.

Y al resto de mis amigos, que han hecho de todos estos años de universidad una época imposible de olvidar.

Resumen

Se pretende diseñar un módulo Web dinámica para GLASS (Gradient's Learning Analytics SyStem), que aglutine, para un único usuario, toda la información extraíble de una base de datos no relacional. Esta información no sólo tiene que mostrar la situación del usuario, sino también la información con la que comparar a éste dentro de una población, para que se pueda conocer no sólo lo que el usuario hace, sino también el uso de su entorno de trabajo. Aunque se tenga mucha información, el módulo debe ser diseñado para que resulte lo más sencillo posible por lo que no puede poseer un gran número de filtros, pero sí tiene que ser capaz de contener la información justa que al usuario le gustaría tener para conocer su situación o rendimiento. Además, debe adaptar todas las características y posibilidades que ofrece GLASS, por lo que debe saber gestionar los diferentes permisos de los diferentes modelos de usuario, ofrecer la posibilidad de compartir un resumen que pueda ser incluido en una página exterior o en el mismo tablón de la herramienta, o ser lo amigable o reutilizable que GLASS pretende ser.

Partiendo del esquema de módulo que propone GLASS, 2 APIS para la generación gráfica y tecnologías como AJAX, JavaScript, HTML5, CSS3, PHP, Mongo y MySQL se desarrolla un módulo que es capaz de evitar las tediosas recargas que tanto incomodan a los usuarios, ofreciendo la movilidad y calidad gráfica que hasta hace poco sólo se podía conseguir a través de Flash.

Palabras claves: Apache, CSS3, diseño, e-learning, GLASS, HTML5, implementación, modulo, Mongo, PHP, plataforma, tecnología Web.

Abstract

The goal is to design a GLASS (Gradient's Learning Analytics SyStem) dynamic Web module, which will combine for a single user, all the extractable information from a non-relational database. The information not only shows the user's situation, it also shows the information needed to compare him against a population. This helps find out not only what the user is doing, but also his use of his work environment. Although it contains large quantities of information the module must be designed to be as simple as possible, therefore it can't possess large quantities of filters. Still it has to be capable of containing the exact information needed so that the user can know their situation or performance. Also it must cover all the features and possibilities offered by GLASS, this includes the ability to manage the different permissions of the different user models, it must be able to share a Widget or summary on an external web page or in the Dashboard tool and must be friendly and reusable as GLASS is meant to be.

Starting from the module structure that GLASS suggests, with the use of 2 APIs for graphics generation and technologies such as AJAX, JavaScript, HTML5, CSS3, PHP, Mongo and MySQL a module is developed to avoid the tedious recharges that are so bothersome for the users, offering the mobility and graphic quality that until now could only be achieved with Flash.

Keywords: Apache, CSS3, design, e-learning, GLASS, HTML5, implementation, module, Mongo, PHP, platform, web technology.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción y objetivos

1.1. Introducción

La comunicación es un fenómeno inherente a la relación grupal de los seres vivos por medio del cual éstos obtienen información acerca de su entorno o de otros entornos y son capaces de compartirla haciendo partícipes a otros de esa información. Las formas de comunicación han pasado de utilizar dibujos en las paredes, señales o gestos hasta poder ver lo que está realizando una persona que se encuentra a cientos de kilómetros de distancia haciendo sólo un simple clic en un ordenador o en un teléfono móvil.

Desde que Internet ha llegado a todos los hogares, la necesidad de comunicarse desde lugares inimaginables ha crecido notablemente. La posibilidad de trabajar y aprender se ve facilitada por las diferentes tecnologías y herramientas que aparecen día a día.

Una forma de utilizar esta comunicación es la de poder recibir formación desde casa con el único esfuerzo que supone manejar un ordenador. De esta manera, es posible asistir a una clase de un colegio o universidad, comunicarte con cientos de personas para debatir un tema o gestionar proyectos sin tener que moverte de casa.

Así, diferentes comunidades y empresas han visto un futuro en la creación de ciertas plataformas de software configurables, ya sea libres o de pago, donde poder gestionar tales actividades y muchas más, y de esta manera mejorar su producción, mejorar sus servicios u ofrecer mejores prestaciones.

Pero qué pasa si se quiere controlar la actividad de estas plataformas o de cualquier entorno de aprendizaje acotado, ¿es posible analizar la información de la actividad que se genera, y a través de los resultados ofrecer información orientada tanto al alumno como al profesor, para que tengan un conocimiento de la evolución del aprendizaje y así obtener patrones que ayuden a mejorar tanto la calidad la enseñanza como la del aprendizaje?

La solución a esta pregunta se encuentra en GLASS. Todas las características que sugiere son facilidades que poco a poco se han ido incluyendo, pero es necesario que siga creciendo, que siga aportando cada vez más información; es necesario seguir desarrollando módulos.

En el inicio de este proyecto, GLASS ya disponía de 2 módulos, el primero, bastante complejo debido al múltiple número de filtros (tantos como variables diferentes tenga la base de datos), permitía ver la actividad de cualquier usuario, grupo o población; el segundo, más sencillo y con menos información, comparaba cada usuario gracias a unas puntuaciones que generaba para establecer un conjunto de ránquines. De este modo, surgía la necesidad de tener un nuevo módulo orientado a cada usuario, que fuese capaz de ofrecer tanto la actividad del usuario como las calificaciones que obtenía por ser estudiante, y que aportara información diferente a la ya existente. Como resultado, se propone diseñar e implementar un nuevo módulo, concretamente la visualización 3, capaz de mostrar toda la información en tiempo real, bajo la premisa de simplicidad y reutilización constante.

1.2. Objetivos

El objetivo fundamental del proyecto es desarrollar un módulo en GLASS capaz de mostrar la información relativa a un usuario y su situación con respecto a la población, según la propuesta realizada. En base a ese objetivo principal, se proponen los siguientes objetivos parciales:

- La módulo debe ser lo suficientemente configurable como para poder entender los eventos del usuario, perdiendo la menor cantidad de información posible, abstrayéndose de aquella información que no es necesaria, independientemente del usuario que la utilice, pero de la forma más sencilla posible. Este es un punto crucial del proyecto, ya que normalmente se entiende que si algo es muy configurable es porque tiene muchas opciones y por lo tanto es complicado de manejar.
- El módulo debe ser lo más amigable y reutilizable posible. Se intenta que sea una aplicación que despierte el interés por si sola. Despierte la curiosidad por lo que ofrece y sea capaz de mostrar los resultados de una forma rápida, fácil y concisa. Para ello la configuración y el filtrado se deben realizar de forma clara, intentando que el usuario realice el menor número de pasos posible.
- GLASS es escalable gracias a los módulos, por lo que estos deben ser completamente integrable a GLASS. El módulo debe diseñarse para ser totalmente adaptable a la herramienta independientemente de lo que ya esté instalado en ella. Además debe tener todas las funcionalidades operativas que se esperan de un módulo.
- Por último debe ser una base para la investigación, es decir, que gracias a él se puedan sacar buenas conclusiones del entorno de trabajo independientemente de éste y a su vez ofrecer información personalizada de los usuarios que lo utilicen promoviendo la motivación.

1.3. Fases del desarrollo

A continuación se describen las 9 fases involucradas en el desarrollado el proyecto fin de carrera:

- **Fase 1:** Para empezar se hace un análisis del módulo. Tras la propuesta, presentada por Abelardo Pardo, se debaten las primeras modificaciones respecto al boceto presentado y se definen todas las características de cada una de las partes en que debería estar constituido el módulo. También se discuten las características que debería tener el módulo con respecto a la herramienta donde corre, tales como el resumen.
- **Fase 2:** Desarrollo Software. Se implementa el módulo casi en su totalidad y se dejan por hacer las características funcionales que no han quedado claras.
- **Fase 3:** Primera revisión. Se lleva a cabo un segundo análisis tras poder interactuar con los primeros resultados. En esta fase se solucionan los problemas presentados y se analizan posibles alternativas a las características, que una vez puestas en funcionamiento, no cumplen con las expectativas generadas.
- **Fase 4:** Desarrollo software. Se implementan todas las características y funcionalidades acordadas en la fase anterior.
- **Fase 5:** Segunda revisión. El módulo está ya casi terminado pero todavía requiere verificar algunas facilidades. Los cambios tras esta reunión ya serán mínimos. En esta fase también se discute la estructura que debería tener el informe.
- **Fase 6:** Retoques finales. Se concluye con el desarrollo del módulo y se afinan aquellas características, tales como el idioma, que tiene una prioridad de desarrollo menor debido a su simplicidad.
- **Fase 7:** Documentación. Se realiza el informe con los apartados acordados en la fase anterior.
- **Fase 8:** Revisión final. El tutor revisa la documentación con fin de clarificar las anotaciones finales antes de imprimir el informe.
- **Fase 9:** Cierre y presentación del proyecto. Se corrigen los posibles errores o recomendaciones que pudieran haber surgido en la fase anterior y se imprime y encuaderna el informe. Dentro de esta fase se incluye la presentación del proyecto.

1.4. Medios empleados

Para la ejecución del proyecto únicamente se ha necesitado de un ordenador personal con conexión a internet. El hardware utilizado para el desarrollo del módulo es el siguiente:

- Procesador: Intel(R) Core(TM) i7 CPU 950 @ 3.07GHZ
- Memoria RAM: 12GB

- Tipo de sistema: Sistema operativo Windows 7 de 64 bits

Dicho Hardware tiene instalado los siguientes paquees, plugins y herramientas:

- APPServ: paquete WAMP (Windows, Apache, MySQL y PHP).
- LDAP plugins: plugin para PHP para poder establecer conexiones con el LDAP de autentificación del que se nutre GLASS.
- PHPdesigner: herramienta para facilitar la labor de programación de todas las tecnologías utilizadas.
- GLASS: última versión instalada y operable.
- Google Chrome: navegador web seleccionado para comprobar el funcionamiento.
- MongoDB: base de datos donde se almacenan los eventos. Posee la estructura de datos definida por GLASS.

1.5. Estructura de la memoria

La memoria está compuesta por 9 capítulos con el siguiente contenido:

El capítulo 1 encierra la introducción y objetivos de la memoria como se ha podido leer.

En el segundo capítulo se realiza un breve estudio teórico de todos los conceptos en los que se basa el proyecto. Se empieza hablando de la visualización narrativa, es decir, lo que cuenta una visualización. A continuación se describen las diferentes técnicas para visualizar la información. Y se termina hablando brevemente de los Mashups y mostrando ejemplos sobre aplicaciones de visualizaciones narrativas donde se describe y ejempla, entre otros, el concepto de learning analytics.

Antes de hablar del módulo objeto del proyecto, es necesario describir brevemente la plataforma dónde va instalado para entender un poco más su funcionamiento. En el capítulo 3 se realiza una pequeña introducción de la plataforma con los objetivos que pretende cumplir, seguida de una descripción del sistema del que forma parte y del subsistema que genera. Por último se muestra la interfaz de usuario donde se describen algunas de las características y facilidades de la plataforma.

En el capítulo 4 llega el momento de hablar del módulo, es decir, de la nueva visualización. Para empezar se hace una pequeña introducción describiendo los conceptos clave. El módulo nace de una propuesta, por lo que es interesante mostrar el boceto origen y hacer una comparación con los resultados finales tanto de la visualización como de su resumen. Seguidamente se describe cada una de las partes del módulo y los archivos que lo conforman. En este apartado se definen las funciones que involucra cada archivo para poder entender la implementación. Para comprender el nivel de desarrollo del módulo, se explican a continuación la mayoría de los scripts y funciones ya enunciadas y se adjuntan los diagramas de flujo que siguen. Estos diagramas no entran mucho en detalle de lo contrario el informe se extendería demasiado; sólo pretenden mostrar el funcionamiento principal y las dependencias de archivos y funciones. También se añade un pequeño manual de usuario con las nociones básicas de instalación y uso, y se enumeran algunos de los posibles problemas que pueden surgir en su utilización.

El capítulo 5 recoge el presupuesto del proyecto. Pretende dar una aproximación de la repercusión económica que pudiera tener el proyecto. En esta sección se describen los recursos utilizados y el precio que estos tienen, pero no sólo de los totales sino los ocasionados en cada una de las fases ya descritas en el proyecto.

Para terminar, en el capítulo 6 se enumeran las principales conclusiones y se describen las líneas futuras que pudieran surgir a partir del proyecto.

Las secciones 7, 8 y 9 recogen el glosario con los acrónimos y conceptos más importantes; las referencias mostradas en el documento; y por último los anexos, donde se describen algunas de las tecnologías utilizadas y un conjunto de buenas prácticas de seguridad.

Capítulo 2

Estado del arte

2.1. Visualización narrativa

En los últimos años, se ha hablado mucho sobre el potencial narrativo de visualización de datos. Las empresas de periodismo tales como *New York Times*, *El Washington Post*, *El mundo* o *Marca* regularmente incorporan gráficos dinámicos en sus páginas Web. Los políticos, activistas y reporteros de televisión utilizan visualizaciones interactivas como telón de fondo para las historias acerca de la salud mundial, la economía y los resultados electorales. Un ejemplo claro son los gráficos interactivos que incorpora el diario Marca en el inicio de un evento deportivo importante cuyo desarrollo implica conocimientos de informática, estadística, diseño artístico y narración.

Desde siempre, se han utilizado visualizaciones estáticas para apoyar la narración, por lo general en forma de diagramas y gráficos insertados en un grupo mayor de texto. En este formato, el texto transmite la historia, y la imagen general proporciona evidencia que apoya o da detalles relacionados. Pero una nueva clase de visualizaciones trata de combinar narraciones con gráficos interactivos. Los periodistas, especialmente los que escriben en la red, están integrando cada vez más complejas visualizaciones en sus narraciones.

Para llegar al éxito dedicándose a la historia de los datos, se requiere de un diverso conjunto de habilidades. Por ejemplo, el caso de un director de cine que aparte de ser un buen contador de historias, debe tener conocimientos técnicos avanzados de informática y ciencia. Si bien las técnicas de oración, prosa, cómics, videojuegos, y la producción de películas son aplicables a la visualización narrativa, también se espera que este medio

emergente posea atributos únicos. Lo que los datos quieren comunicar difiere en muchos aspectos en relación a lo que nos cuentan los relatos. Las historias en las películas o los libros suelen presentar una serie de eventos en una progresión muy controlada. Mientras que un recorrido por los datos se puede visualizar de manera similar organizándose en una secuencia lineal, también pueden ser interactivos, invitar a la verificación, promover nuevas cuestiones o dar lugar a explicaciones alternativas.

En la actualidad, la mayoría de las herramientas más sofisticadas de visualización se centran en la exploración y análisis de datos. Las aplicaciones, como hojas de cálculo y herramientas de visualización, son compatibles con una gran variedad de rutinas de análisis y codificaciones visuales, pero más allá de exportar imágenes para presentaciones suelen ofrecer poco apoyo para la elaboración de historias con los resultados del análisis. Como tales, proporcionan vehículos poderosos para descubrir "historias", pero hacen poco para ayudar a comunicar la narrativa de estos hallazgos a los demás. Como herramientas maduras y con buena integración en la Web Multiples Eyes [4], Tableau Public [5] y GeoTime Historias [6] están permitiendo la publicación de gráficos dinámicos con niveles limitados de interactividad. Queda abierta la cuestión de cómo se desarrollaría el diseño de dichas herramientas para apoyar formas de narración más ricas y diversas.

La narración de cuentos y expresión visual son parte integral de la cultura humana; la narración de cuentos ha sido incluso referida como "la segunda profesión más antigua del mundo" [7]. A continuación se describen algunos de los conceptos clave sobre la visualización narrativa.

2.1.1. Estructura Narrativa

El Diccionario de la real academia española [3] define la narrativa como "habilidad o destreza en narrar o en contar algo". Desde tiempos antiguos, la gente ha tratado de entender y formalizar los elementos de la narración. Por ejemplo, los escritores (por ejemplo, [8][9][10]) han desarrollado tipologías de situaciones dramáticas y han identificado líneas comunes a muchas narraciones, como "viaje del héroe" [8] .

Esta investigación distingue entre el contenido de la historia y la forma en la que se cuenta. Si bien las historias a menudo se refieren a la interacción de personajes, también pueden presentar una secuencia de hechos y observaciones unidos entre sí por un hilo conductor o argumento.

Las estrategias de los cuentacuentos varían entre los medios de comunicación y otros géneros. Por ejemplo, las historias contadas a través de la escritura tienen acceso a un conjunto formal de mecanismos y estructuras narrativas (por ejemplo, la corriente de la conciencia) diferentes a las historias contadas a través del cine (por ejemplo, secuencias de pantalla dividida [11]). Blundell [12] describe los recursos narrativos para el periodismo como la *anecdotical lead*, una historia inicial, con el diálogo y los personajes que presenta el microcosmos de las noticias de mayor historia, y *the nut graf* describe explícitamente el valor de la noticia de un artículo. Estos dispositivos son en gran medida únicos para el periodismo, al contrario que para la ficción literaria o el cine. Las visualizaciones se pueden incorporar en los medios de comunicación incluyendo texto, imágenes y vídeo, y también pueden ser interactivas, lo que permiten historias cuya narración se basa tanto en el lector como el autor.

2.1.2. Narrativas visuales

Artistas, diseñadores, y psicólogos han estudiado la forma en que los medios visuales pueden organizarse para generar una experiencia narrativa. Se han desarrollado técnicas para dirigir secuencialmente la atención de los espectadores a través de transiciones orientadas. Aunque se podría realizar un estudio completo de estas técnicas, para el entendimiento de este trabajo es suficiente con presentar los principios más destacados.

Muchos relatos tienen sus raíces en un punto de partida claro. En los medios visuales a menudo se utiliza un plano general o visión general [9][11] para introducir la escena. Por supuesto, no todos los elementos de una escena son de igual importancia a lo largo de la historia, por lo que los autores suelen manipular la escena para dirigir la atención a un punto de interés. Los psicólogos tienen extensamente estudiado los fenómenos de prominencia visual [13][14][15], que muestran que una cantidad externa de características visuales tales como el color, el tamaño y la orientación pueden conseguir llamar la atención. La fuerza de esta atracción se modula por múltiples factores [14]. En cuanto a la escena en sí, por ejemplo, un objeto de color brillante es menos relevante cuando está rodeado por otros objetos de colores brillantes. En cuanto a la tarea del espectador, por ejemplo, las expectativas y la búsqueda de arriba hacia abajo pueden afectar a lo que se percibe como más o menos sobresaliente.

Los factores culturales, en particular la forma de lectura (por ejemplo, de izquierda a derecha) tienden a establecer lo que la gente mira primero y la forma en la que escanea la imagen [9]. Técnicas visuales pueden establecer la forma en el que el ojo observa los elementos de la escena. Por ejemplo, el agrupamiento Gestalt [16] a través de características tales como la proximidad espacial, la contención, o la conexión, puede conducir a la percepción de agrupamiento. La referencia vectorial [17][18], o lo que más comúnmente se entiende como las flechas de dirección, es una potente técnica para dirigir la atención secuencialmente.

Los medios visuales tienden a provocan cambios de escena, como ocurre en los paneles entre viñetas de los cómics o los cortes de edición del cine. El número de dispositivos se desarrolla para orientar a un espectador durante las transiciones. Continuas técnicas de edición en el cine [11], como la equiparación de los objetos o acciones, sugieren la relación entre las escenas y pueden sostener un enfoque de atención. Del mismo modo, el diseño de animación [5][19] a menudo se basa en la constancia de los objetos y resta importancia a detalles secundarios para mantener la orientación visual; los animadores también podrán subdividir una transición en etapas para facilitar la aprehensión. Dentro de los cómics, McCloud [9] propone una taxonomía de los tipos de transición que consiste en “momento a momento” (una tema, período de tiempo corto), “la acción a la acción” (un sujeto, mayor tiempo de período), “de sujeto a sujeto” (diferentes temas, la misma escena), “escena a escena” (cambio de escena), “de aspecto a aspecto” (los aspectos de un lugar, una idea o del estado de ánimo), y todo sin desconectar las transiciones. Además, para seguir los objetos o acciones, se utilizan elementos pictóricos [18] tales como llamadas (por ejemplo, inserciones o líneas para denotar el zoom) y anotaciones para enriquecer una narración. Como era de esperar, muchas de estas técnicas son también aplicables a la visualización narrativa.

2.1.3. Elementos y características utilizados en visualizaciones

Aunque la visualización de datos a menudo evoca comparaciones con los cuentos [20][21], la relación entre ambos raramente es articulada con claridad. Jonathan Harris, el creador de *Nos sentimos bien* y *Caza de Ballenas*, se considera a sí mismo, en primer lugar, un narrador y, en segundo lugar, un diseñador de visualizaciones: "Pienso que la gente ha comenzado a olvidar como son las poderosas historias humanas, cambiando su sentido de empatía por una fascinación fetichista con los datos, redes, patrones e información... en definitiva, los datos son sólo una parte de la historia. La parte humana es la parte principal, y los datos lo enriquecen". Sin embargo, cuando se le pregunta que describa lo que él entiende por "historia", responde con sólo una aproximación: "Yo defino la historia sin prisa. Para mí, una historia puede ser tan pequeña como un gesto o tan grande como la vida. Pero los elementos básicos de una historia, probablemente, se puedan resumir con Quién / Qué / Dónde / Cuándo / Por qué / Cómo".

Otros han tratado de articular la conexión de manera más concisa. Gershon y Page [7] observan que las historias comunican información en un formato psicológicamente eficaz. Con el guion de un escenario militar ficticio como caso de estudio, examinan las tácticas utilizadas para comunicar eventos narrativos, incluyendo continuidad de edición, resaltamiento (por ejemplo, el parpadeo) y mensajes redundantes a través de los medios de comunicación (por ejemplo, audio y vídeo). Sin embargo sigue siendo difícil de alcanzar una comprensión más profunda de la narrativa de las visualizaciones, ya que hay que entender mejor las interacciones propias de cada género con cada audiencia en particular, sus ventajas y desventajas, y cómo podría afectar al contenido y al aprendizaje. Wojtkowski y Wojtkowski [22] argumentan que lo que hace la visualización de datos diferente de otros tipos de narración visual es la complejidad del contenido que se debe comunicar. Llegan a la conclusión de que "la narración visual, a su vez, podría ser de gran importancia para una exploración rápida e intuitiva de los grandes recursos de datos", pero de nuevo no se detalla la forma en que podría ser mejor "adaptar los sistemas de visualización para adaptarse a contar historias".

Algunos sistemas de visualización han comenzado a incorporar la narración en su diseño. Por ejemplo, GeoTime Stories [23] permite a los analistas crear historias comentadas en las visualizaciones utilizando un editor de texto y la interfaz de *bookmarking*. El sistema sense.us [24] permite a los usuarios crear senderos de marcadores de visualización que se utilizan regularmente para contar historias. Tableau's graphical histories [25] permite rebajar, cotejar y explorar los puntos claves de su análisis visual. Más recientemente, Tableau's graphical histories [5] apoya la construcción y la publicación de visualizaciones interactivas basadas en Web, apoyando la narración en dominios ricos en datos, tales como las finanzas y el periodismo deportivo. Estos sistemas suponen los primeros pasos hacia la fabricación de ricas y accesibles capacidades de narración de cuentos.

En resumen, muchos han observado el potencial narrativo de la visualización de datos y han trazado semejanzas con los medios más tradicionales. Sin embargo, todavía no ha surgido una comprensión completa del espacio de diseño para la visualización narrativa. Mientras tanto, los profesionales, tales como artistas y periodistas, han estado forjando caminos a través de este espacio, y pueden proporcionar conocimientos a partir de sus exploraciones.

En “Narrative visualization: telling stories with data” [1], documento consultado para la redacción de este apartado, Edward Segel y Jeffrey Heer, intentan mejorar la comprensión de la visualización a través del análisis narrativo.

2.2. Visualización de la información

El éxito de las interfaces de manipulación directa es un primer paso hacia el uso potencial de los ordenadores de una forma más visual o gráfica [2]. Se suele decir que una imagen vale más que mil palabras, y para algunas tareas, una presentación visual, tales como un mapa o una fotografía, es mucho más fácil de usar o comprender que una descripción textual o un informe oral. Con el aumento de la velocidad del procesador y la resolución de pantalla, los diseñadores están descubriendo cómo presentar y manipular grandes cantidades de información de forma compacta y controlada por el usuario. Ahora se puede argumentar que una interfaz vale más que mil imágenes. La visualización de información se puede definir como el uso interactivo de representaciones visuales de datos abstractos para amplificar la cognición. La característica abstracta de los datos es lo que distingue a la visualización de información de la visualización científica. Para la visualización científica, son necesarias tres dimensiones porque las preguntas típicas incluyen variables continuas, volúmenes y superficies (interior / exterior, izquierda / derecha y arriba / abajo). Sin embargo, para la visualización de la información, las preguntas típicas incluyen más variables categóricas y el descubrimiento de patrones, tendencias, grupos atípicos y lagunas en los datos como en los precios de las acciones, los registros de pacientes o las relaciones sociales.

Los investigadores de la visualización de la información pretenden proporcionar presentaciones gráficas e interfaces de usuario compactas para manipular interactivamente una gran cantidad de elementos, posiblemente extraídos de conjuntos de datos mucho más grandes. Algunas veces llamada minería de datos visual, utiliza el enorme ancho de banda visual y el extraordinario sistema preceptuar humano para que el usuario pueda hacer descubrimientos, tomar decisiones o proponer explicaciones acerca de los patrones, grupos de artículos o elementos individuales. La información de la visualización permite a los usuarios responder a las preguntas que no sabían que tenían.

Los seres humanos tienen notables capacidades perceptivas que son muy poco utilizadas en la mayoría de los diseños actuales. Los usuarios pueden escanear, reconocer y recordar imágenes rápidamente, y pueden detectar cambios sutiles en el tamaño, el color, la forma, el movimiento, o la textura. La información básica, presente en las interfaces gráficas de usuario se ha mantenido prácticamente orientada al texto (aunque mejorada con iconos atractivos y elegantes ilustraciones). Nuevos psicólogos preceptúales, estadísticos, y diseñadores gráficos ofrecen valiosos consejos sobre la presentación de información estática, pero los avances en la velocidad de procesación, las herramientas de diseño y las pantallas dinámicas permiten diseños de interfaz de usuario que llegan mucho más allá de la sabiduría actual.

A medida que el campo evoluciona, los prototipos en investigación están encontrando su camino para convertirse en productos comerciales. Sin embargo, los diseñadores de la visualización de información deben entender los principios que les ayuden a cruzar el abismo a un mayor éxito. Se tiene que superar la obsesión por la novedad y la integración de las herramientas de visualización en soluciones para los

problemas de negocio reales. Esto significa que se tendrá que facilitar la importación de datos desde muchas fuentes, hacer frente a grandes volúmenes de datos, y permitir a los usuarios integrar otras herramientas, como las de minería de datos. Luego, cuando un usuario formula hipótesis acerca de las relaciones y los patrones del punto de interés, tendrá que ser capaz de guardar, enviar, imprimir y compartir estos conocimientos con los demás a través de interfaces de coordinación bien integradas.

Muchos usuarios se resisten a aproximaciones visuales y están satisfechos con potentes aproximaciones textuales para la búsqueda de metadatos, tales como menús múltiples y consultas numéricas. Su resistencia puede ser apropiada, hasta que aquellas herramientas textuales usen presentaciones compactas ricas en información. Pero para el éxito es necesario que estas herramientas de visualización de información sean más "cool"; tienen que proporcionar beneficios medibles en las tareas reales. Tienen también que ser construidas para satisfacer los principios de usabilidad, para trabajar en una variedad de plataformas, tamaños de pantalla, anchos de banda de la red, etc., al tiempo que permiten el acceso a usuarios con discapacidad y diferente lengua.

Las directrices, principios y teorías que emergen sobre la visualización de la información y que proporcionan un conocimiento más maduro, recogen el principio, ampliamente citado, conocido normalmente como el mantra de la búsqueda de información visual:

Resumen primero, zoom y filtrado, luego detalle bajo demanda.

Resumen primero, zoom y filtrado, luego detalle bajo demanda.

Resumen primero, zoom y filtrado, luego detalle bajo demanda.

Resumen primero, zoom y filtrado, luego detalle bajo demanda.

Primero la descripción general, el zoom y el filtro, a continuación detalle bajo demanda. La repetición sugiere la frecuencia con la que el principio ha sido aplicado y la naturaleza recursiva del proceso de exploración. Los investigadores y desarrolladores en visualización de la información, pueden ser capaces de desarrollar numerosas herramientas e identificar nuevas oportunidades utilizando tipos de datos por taxonomía de tareas (Tabla 1).

Tipos de datos:	**Ejemplos de aplicaciones:**
1D lineal	Lente de documentos, SeeSoft, Mural de la Información, TextArc
2D mapa	Sistemas de información geográfica, ESRI ArcInfo, ThemeView, diseño de periódicos, la autoorganización de mapas.
3D mundo	Escritorios, WebBook, VRML, Web3D, arquitectura, el diseño asistido por ordenador, medicina, moléculas.
Multidimensional	Coordenadas paralelas, matrices, diagrama de dispersión, agrupación jerárquica, Table Lens, lnfoZoom
Temporal	Perspective Wall, análisis exploratorio de datos secuencial (ESDA), gestores de proyectos, líneas de vida, TimeSearcher
Árbol	Outliners, Superbook, árboles de grados de lnterest, Hiperbolic, SpaceTree, diagramas de árbol.
Red	NetMap, netViz, SemNet, SeeNet, Butterfly.

Tareas:	Descrpción:
Visión	Obtener una visión general de toda la colección.
Zoom	Zoom sobre temas de interés.
Filtro	Filtrado de los artículos interesantes.
Detalle bajo demanda	Seleccionar un elemento o grupo y obtener información cuando sea necesario.
Relación	Ver relaciones entre elementos.
Historia	Mantener un historial de acciones de apoyo a deshacer, reproducir y refinamiento progresivo.
Extraer	Permitir la extracción de subcolecciones y de los parámetros de consulta.

Tabla 1: Tipo de datos por taxonomía de tareas

Como en el caso de las búsquedas, los usuarios están viendo colecciones de artículos donde los artículos tienen varios atributos. El tipo de datos por taxonomía de tarea incluye siete tipos de datos básicos y siete tareas básicas. Los tipos de datos básicos son de uno, dos, tres, o varias dimensiones, seguidos de otros tres tipos de datos estructurados: árbol, temporal y red. Esta simplificación es útil para describir las visualizaciones que han sido desarrolladas y para caracterizar las clases de problemas con los que los usuarios se encuentran. Por ejemplo, los usuarios de datos temporales analizan los eventos e intervalos refiriéndose al antes de, después de, o durante. Por el contrario, con los datos de estructura de árbol, los usuarios se refieren a los nodos internos y los valores de los nodos finales con preguntas relacionadas con las rutas, niveles y subniveles. Las siete tareas básicas son: visión, zoom, filtrado, detalle bajo demanda, relación, historia y extracción. A continuación se procede a comentar los siete tipos de datos con los siete tipos de tareas.

2.2.1. Datos lineales en 1D

Los datos lineales son unidimensionales. Dentro de este tipo de datos se encuentran la visualización del código fuente de un programa, documentos de texto, diccionarios y listas alfabéticas de nombres, todos los cuales se pueden organizar de manera secuencial. Si se quiere visualizar el código fuente de un programa reduciendo a menos de un píxel por carácter se forma una sola pantalla compacta de decenas de miles de líneas de código, como se puede ver en la Figura 1. Para facilitar el entendimiento al usuario en el diseño de interfaces se utilizan colores, tamaños, formas de disponer los elementos, "scrolling" o vistas completas. La tarea del usuario podría ser la de encontrar el número de elementos, ver los artículos que tienen ciertos atributos, ver las palabras más comunes en un libro, o ver un elemento con todos sus atributos.

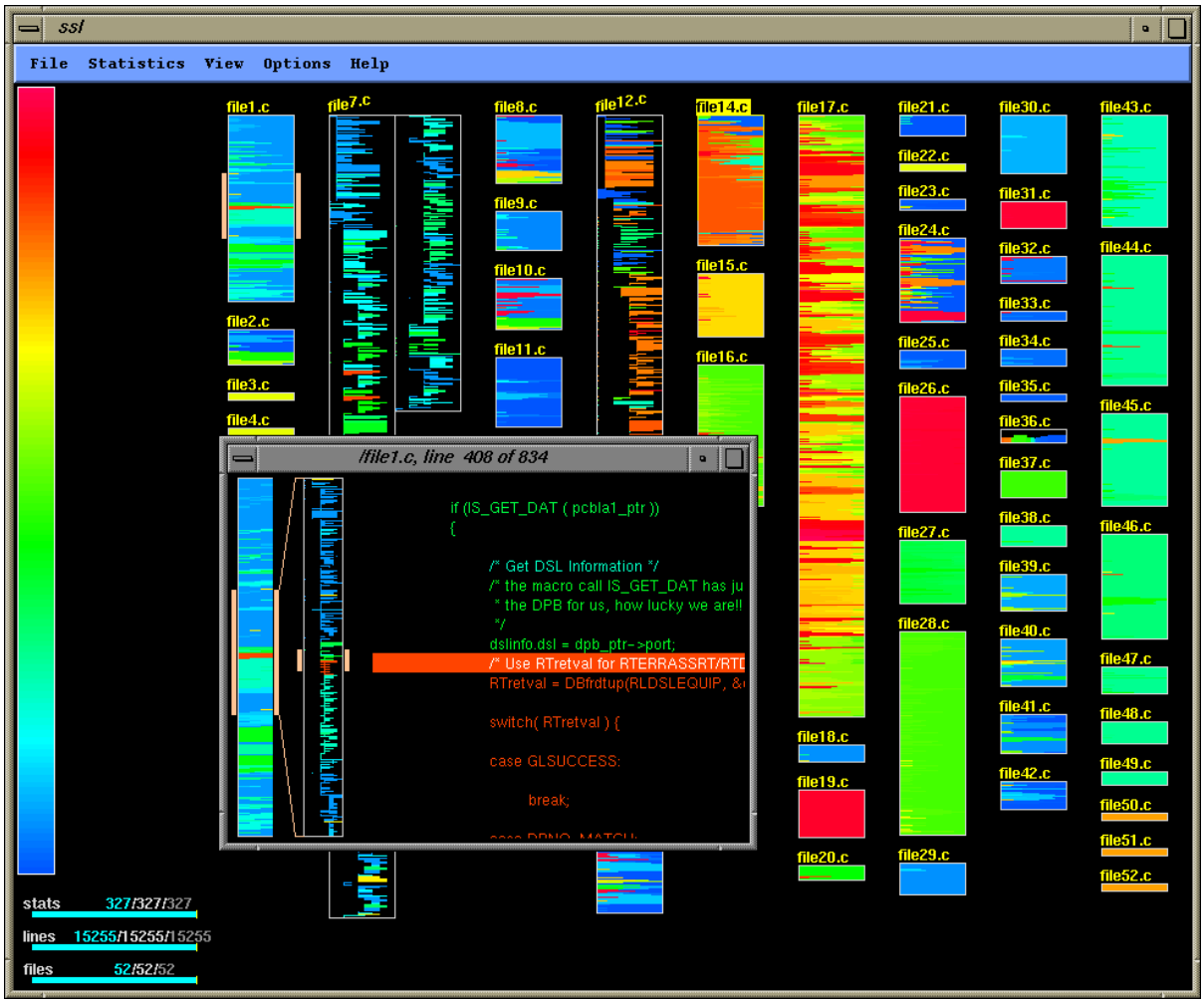


Figura 1: SSL. Aplicación de 15000 líneas de código.

2.2.2. Mapas de datos en 2D

Los datos planos incluyen por ejemplo los mapas geográficos, los planos o la disposición de un periódico. Cada elemento de la colección cubre una parte de la superficie total y puede o no ser rectangular. Cada elemento tiene los atributos de dominio de tareas, como el nombre o el propietario, y el valor y las características de dominio de la interfaz, como el tamaño, el color o la opacidad. Muchos sistemas adoptan un enfoque multicapas al tratar con los datos de un mapa, pero cada capa es de dos dimensiones. Las tareas de usuario son para encontrar los elementos adyacentes, las regiones que contienen objetos, y los caminos entre los elementos, y para realizar las siete tareas básicas. Otros ejemplos son los sistemas de información geográfica, el cual necesita de un estudio amplio y de dominio comercial. Dichos dispositivos parecen útiles al ofrecer al usuario una visión general de la colección, pero puede que necesariamente no sean tan útiles si no van acompañados de una representación textual que pueda enriquecerse de detalles.

2.2.3. Datos reales en 3D

Los objetos del mundo real, tales como las moléculas, el cuerpo humano, y los edificios tienen volumen y complejas relaciones con otros elementos. Las simulaciones por ordenador con ayuda de imágenes médicas, los dibujos arquitectónicos, los diseños mecánicos, el modelado de estructuras químicas y las simulaciones científicas se construyen para manejar estas complejas relaciones tridimensionales. Las tareas que los usuarios suelen hacer frente a las variables continuas, tales como la temperatura y la densidad de los resultados, se presentan a menudo como volúmenes y superficies, y así los usuarios se centran en las relaciones entre izquierda/derecha, arriba/abajo, e interior/exterior. En las aplicaciones tridimensionales, los usuarios al ver los objetos deben atender a la posición u orientación y deben manejar los posibles problemas de oclusión y navegación. Existen múltiples soluciones que utilizan técnicas mejoradas en 3D, vistas generales o monumentos. Las interfaces tangibles de usuario están encontrando su camino en prototipos de investigación y sistemas comerciales. Entre los éxitos figuran las imágenes de ecografías médicas que ayudan a los médicos en la planificación de la cirugía y tutoriales de arquitectura o las vistas virtuales que dan una idea a los compradores de vivienda de cómo será el edificio tras su construcción.

Son numerosos los ejemplos de gráficos en tres dimensiones y los diseños asistidos por ordenador, pero es aún compleja la información visual que ofrece una visualización en tres dimensiones. Algunos investigadores del medio ambiente virtual y diseñadores gráficos han tratado de presentar la información de estructuras en tres dimensiones, pero estos diseños a veces necesitan de varios pasos a la hora de navegar por ellos, lo que contribuye a que los resultados sean difíciles de interpretar.

2.2.4. Datos multidimensionales

Los contenidos relacionales de las bases de datos se pueden manipular como datos multidimensionales en los que elementos con *N* atributos se convierten en un punto en un *espacio n-dimensional*. La representación en una interfaz puede ser un diagrama de dispersión dinámica de dos dimensiones, con cada dimensión adicional controlada por un control deslizante. Los botones se pueden utilizar para los valores de atributo cuando la cardinalidad es pequeña, digamos de menos de 10. Las tareas incluyen la búsqueda de patrones, como las correlaciones entre pares de variables, las agrupaciones, las lagunas, y los valores extremos. Los datos multidimensionales también se pueden representar por un diagrama de dispersión tridimensional pero desorientado (sobre todo si el punto de vista del usuario es desde el interior de la nube de puntos) y la oclusión (especialmente si los puntos de cierre se representan como si fuera más grande) puede tener problemas. La utilización de diagramas de colores a la vez que se utilizan técnicas de 3 dimensiones es una manera de alcanzar un entorno multidimensional. Las parcelas paralelas de coordenadas son una de las pocas técnicas multidimensionales que es verdaderamente compacta (Figura 2). Cada eje vertical representa una dimensión paralela, y cada elemento se convierte en una línea que une los valores en cada dimensión. La formación y la práctica son particularmente útiles para convertirse en un detective "multidimensional".

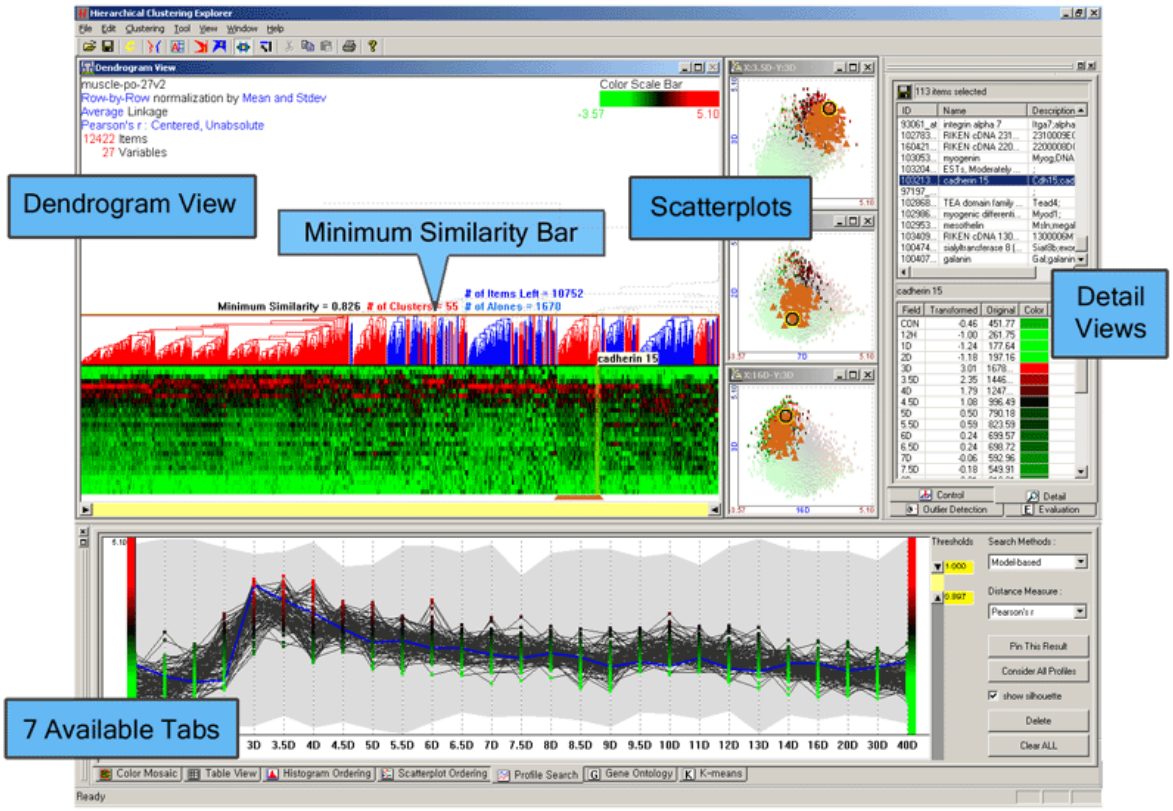


Figura 2: Human-computer interaction lab. Explorador de agrupación jerárquica para la exploración interactiva de datos multidimensionales. [26]

Otras técnicas podrían ser las que aparecen en la Figura 3, que incluye una matriz que combina una serie de pequeñas representaciones bivariadas, Table Lens (Figura 4), que utiliza una metáfora de hoja de cálculo, y InfoZoom (Figura 5), que utiliza una tabla de contenido con zoom que muestra la distribución de valores para cada dimensión y permite el filtrado progresivo de los datos haciendo clic en dichos valores. Un enfoque cada vez más común para buscar datos multidimensionales es utilizar la agrupación jerárquica o el algoritmo de agrupamiento *k-means* para identificar elementos similares. La agrupación jerárquica identifica pares cercanos de artículos y foros que son cada vez más grandes hasta que cada punto se incluye en un clúster (Figura 2). La jerarquía *k-means* comienza pidiendo a los usuarios que especifiquen cuántos grupos quieren crear, a continuación, el algoritmo pone cada elemento en el grupo más apropiado. Con estas técnicas se pueden identificar relaciones sorprendentes y valores atípicos.

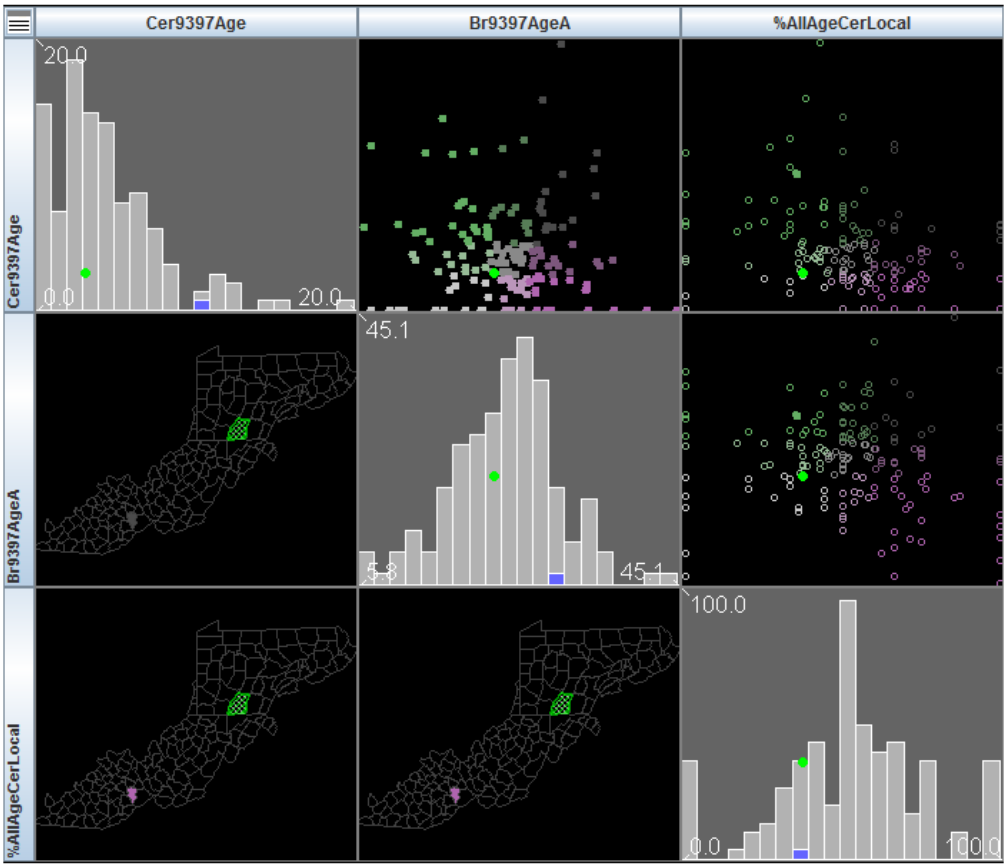


Figura 3: GeoVISTA. Matriz bivariada multiforme con mapa y diagrama de dispersión. [27]

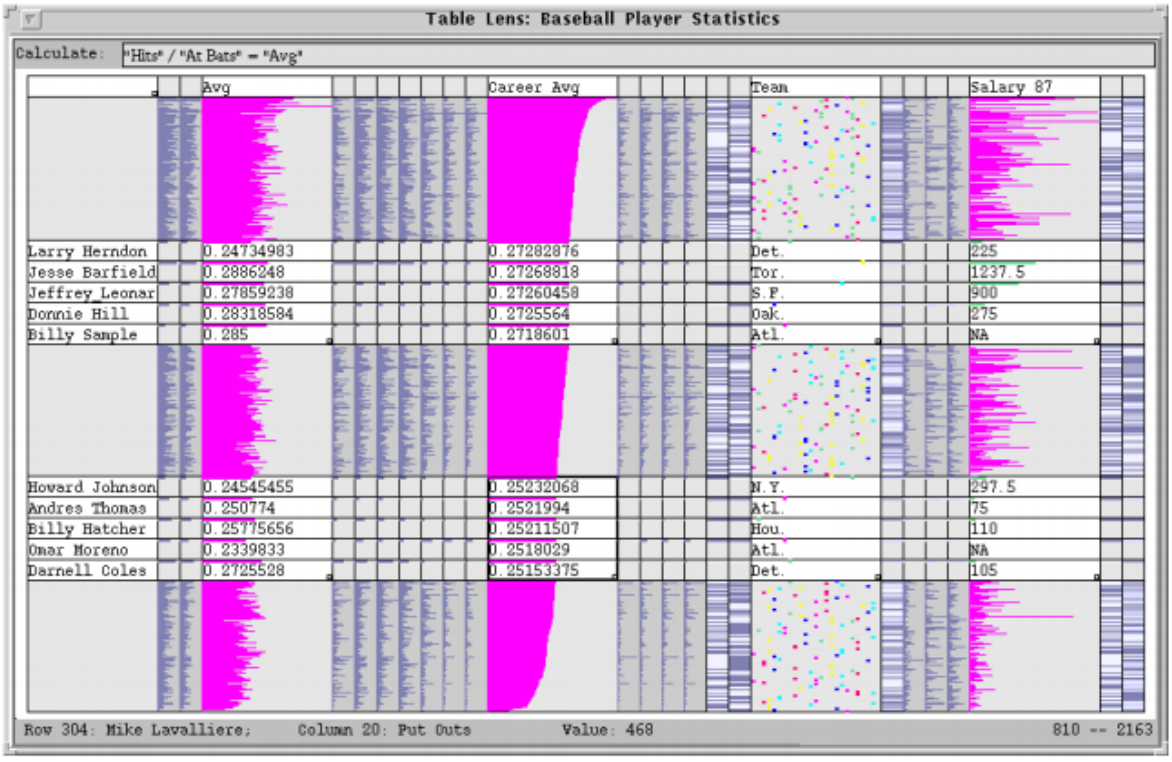


Figura 4: Table Lens [28]

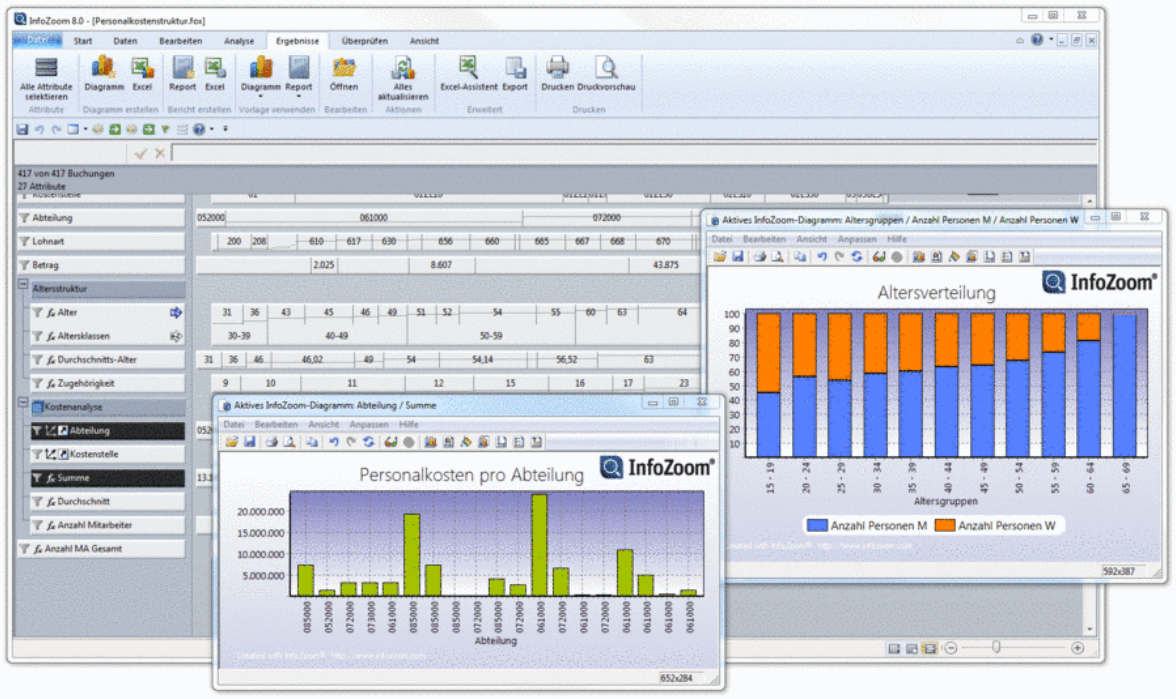


Figura 5: InfoZoom 8.0 [29]

2.2.5. Datos temporales

Las series temporales son muy comunes (por ejemplo, los electrocardiogramas, los precios del mercado de valores, o los datos meteorológicos) y merecen un tipo de datos que es independiente de los datos unidimensionales. Las distinciones de datos temporales son los elementos (eventos), tienen una hora de inicio y finalización, y los elementos se pueden solapar. Tareas frecuentes incluyen la búsqueda de todos los eventos antes, después, o durante algún período de tiempo o momento, y en algunos casos se comparan fenómenos periódicos, además de las siete tareas básicas. Algunos ejemplos son muchas de las herramientas de gestión de proyectos existentes o visualizadores de línea de vida (Figura 6). Las visualizaciones de componentes temporales de datos se incluyen en las aplicaciones de edición de vídeo, música o para preparar animaciones con Flash. Ajustar el espacio temporal de datos ha sido el centro de atención de la geovisualización. La búsqueda temporal combina múltiples series, como los precios del mercado de valores, u otras series de datos lineales como podría ser la temperatura de una región. Los usuarios dibujar cuadros en la pantalla para especificar combinaciones de rangos, y el buscador temporal muestra la serie cuyos datos caen dentro del rango.

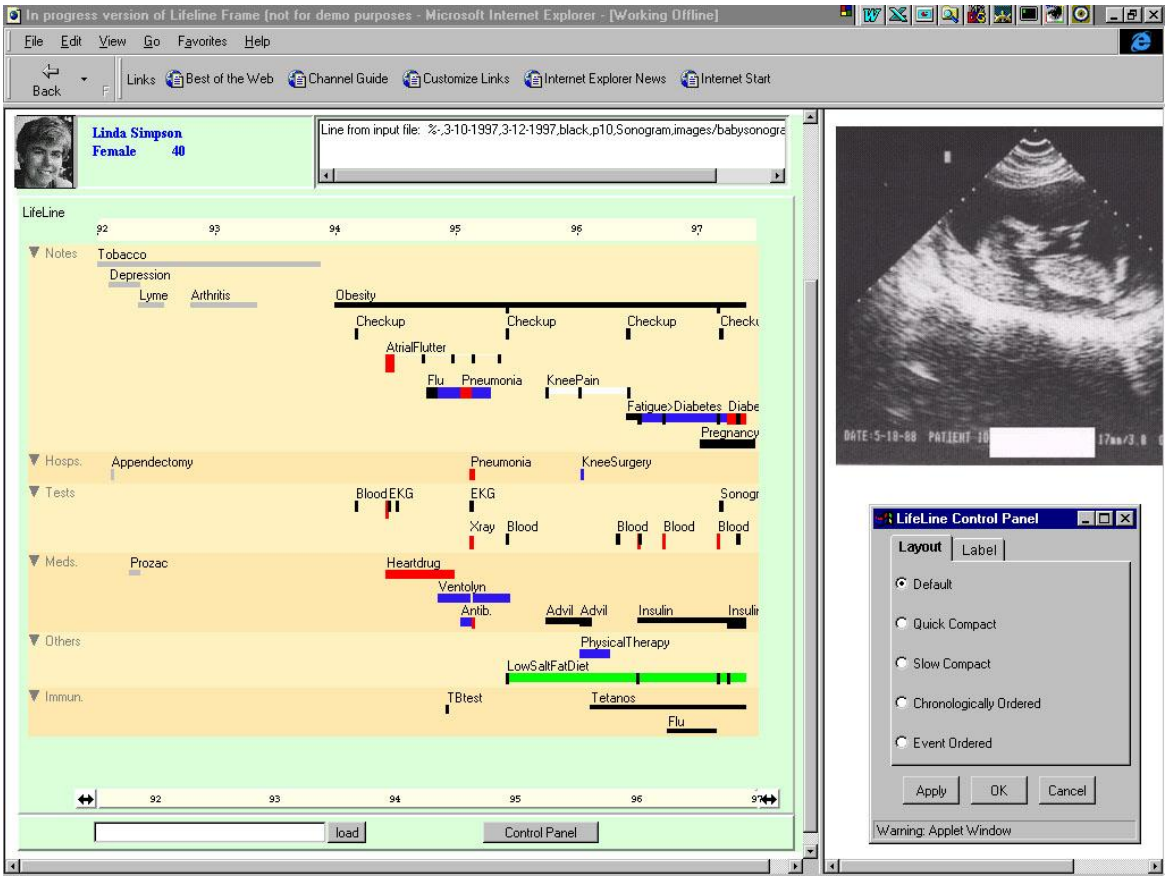


Figura 6: Lifeline. Línea de vida médica para la visualización de eventos [30]

2.2.6. Árbol de datos

Las estructuras en árbol o jerárquicas son colecciones de artículos en los que cada elemento (excepto el elemento raíz) tiene un enlace a un artículo principal. Los elementos y las relaciones entre padres e hijos pueden tener varios atributos. Las funciones básicas se pueden aplicar a los artículos y enlaces, y las tareas relacionadas con las propiedades estructurales se vuelven interesantes, por ejemplo, para un organigrama de una empresa, puede ser una jerarquía profunda o superficial, y a través de ella se puede saber cuántos empleados son supervisados por su manager. Las interfaces gráficas suelen utilizar el esquema utilizado por el explorador de archivos de Windows, o esquemas de etiquetas con diferentes sangrías, o vínculos tales como el árbol de grados de interés, SpaceTree (Figura 7), o el navegador hiperbólico (Figura 8).

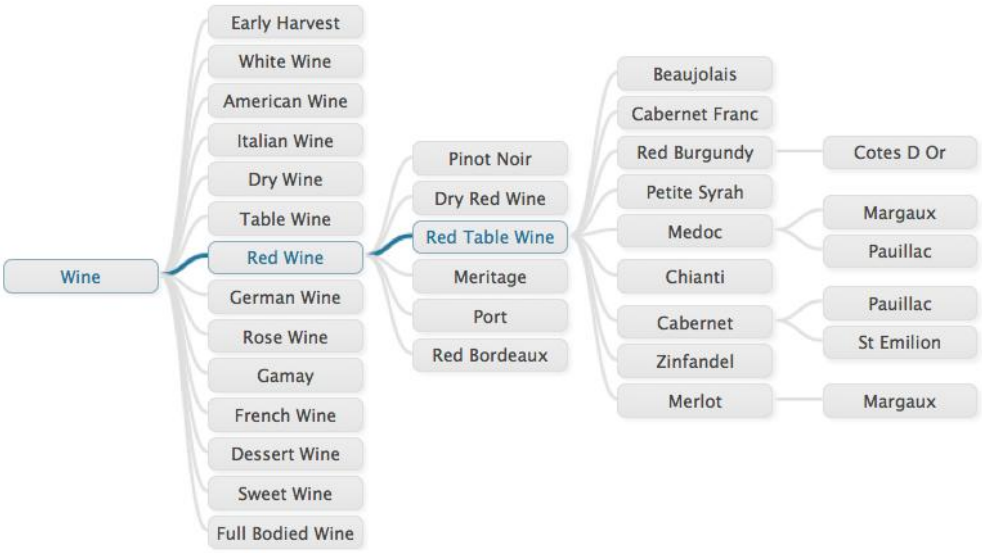


Figura 7: Space Tree

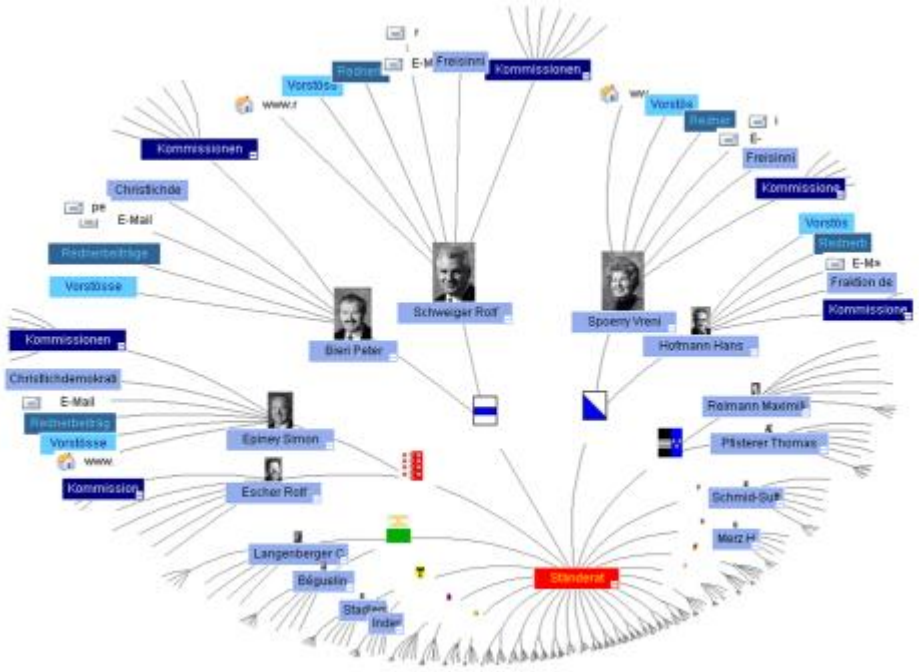


Figura 8: Buscador hiperbólico

Este enfoque se aplica para muchas aplicaciones como pueden ser la visualización de datos del mercado de valores, la supervisión de la producción de petróleo o la búsqueda de productos en los catálogos electrónicos.

2.2.7. Network data

Es útil cuando las relaciones entre los elementos no pueden ser capturados convenientemente con una estructura de árbol porque están vinculados a un número arbitrario de otros elementos en una red. Aunque existen muchos casos especiales de red (acíclico, celosías, ruteado versus no ruteado, dirigido versus no dirigido), es conveniente considerar a todos ellos como un tipo de datos. Además de realizar las tareas básicas

aplicadas a los artículos y enlaces, los usuarios de la red a menudo quieren saber acerca de las rutas más cortas o menos costosa de conexión entre dos elementos o recorrer toda la red. Las representaciones de interfaz incluyen diagramas de nodo en vínculo (pero a menudo son tan complejos que la interacción del usuario es limitado cuando se muestran redes grandes) y matrices de elementos con cada celda representando un vínculo potencial y sus valores de atributo. La visualización de una red es un arte antiguo, pero todavía imperfecto, debido a la complejidad de las relaciones y las tareas de usuario. Se pueden diseñar visualizadores especializados para ser más eficaces para una tarea dada como para aplicaciones geográficas. El nuevo interés por este tema se debe sobre todo a los constantes intentos por visualizar la World Wide Web (Figura 9).

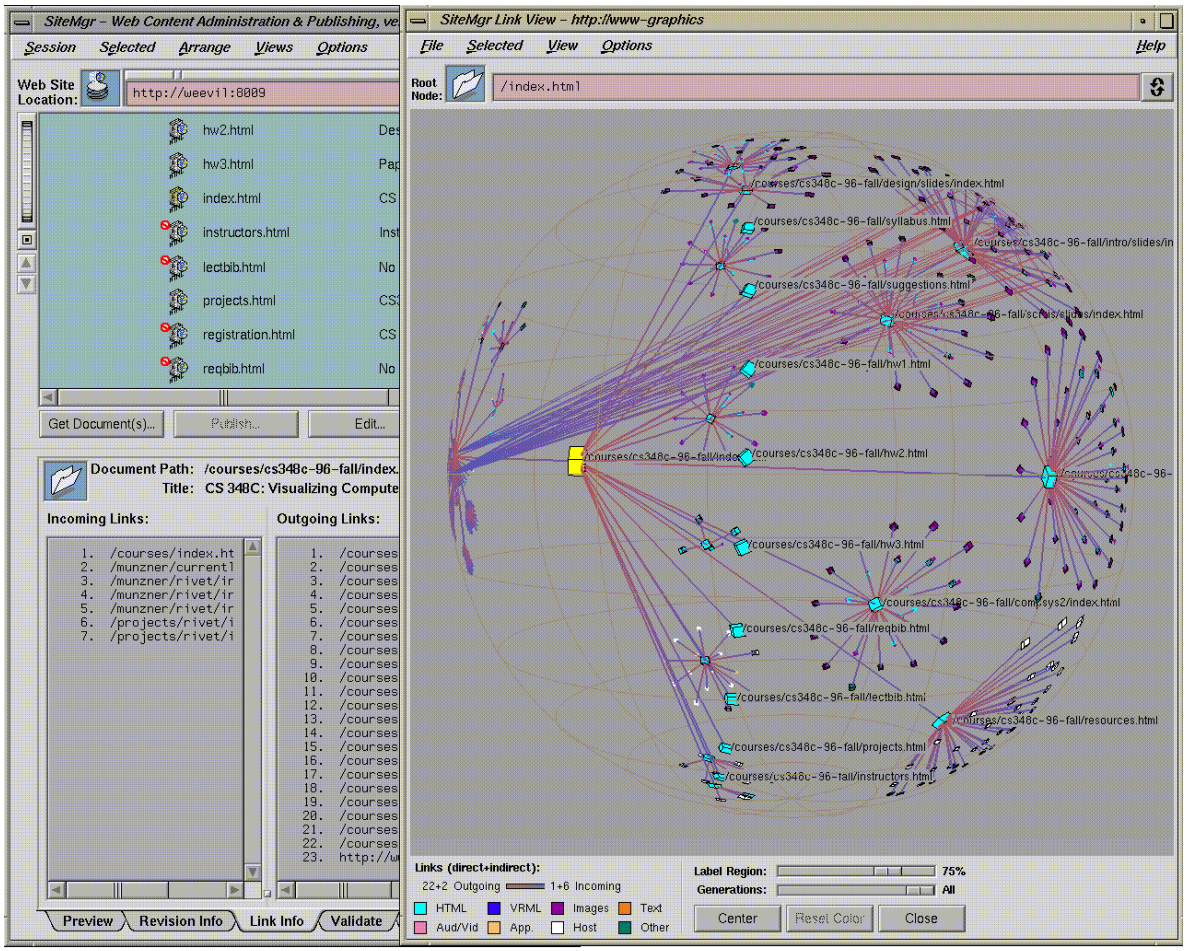


Figura 9: SiteMgr. Estructura de enlace de una Web [31]

Los siete tipos de datos reflejan una abstracción de la realidad. Hay muchas variaciones sobre cada uno de ellos (muchos prototipos utilizan combinaciones de estos tipos de datos). El segundo marco para el análisis de las visualizaciones de información abarca siete tareas básicas que los usuarios suelen realizar:

2.2.8. Visión general de tareas

Es posible obtener una visión general de toda la colección de datos. Mediante zoom se puede pasar de ver la colección completa a ver una vista detallada contigua. El resumen en cada vista puede contener un campo movible con el que los usuarios

controlan el contenido en detalles. La repetición de esta estrategia con vistas intermedias permite a los usuarios llegar a un mayor factor de zoom. Otra aproximación es la estrategia de ojo de pez, cuya distorsión aumenta una o más áreas de la pantalla, aunque no se debería realizar más de 5 veces para ser realmente utilizable (Figura 4). Dado que la mayoría de los lenguajes de consulta hacen que sea difícil obtener una visión general de una colección, un criterio útil para juzgar una herramienta es que esta disponga de una estrategia adecuada para mostrar una vista general.

2.2.9. Tareas de Zoom

Se puede hacer un zoom sobre los puntos de interés. Los usuarios suelen tener interés en alguna parte de una colección, por lo que necesitan herramientas que les permitan controlarlo. Es recomendable aplicar además un zoom suave que ayude a los usuarios a mantener su sentido de la posición y el contexto. Los usuarios son capaces de ampliar en una dimensión a la vez moviendo los controles de la barra de zoom o ajustando el tamaño de la caja de campo de visión. Una manera satisfactoria para ampliar la imagen es apuntar a una ubicación y ejecutar un comando de zoom, generalmente presionando un botón del ratón.

2.2.10. Tareas de filtrado

Se pueden filtrar los elementos de interés. Las consultas dinámicas aplicadas a los elementos de la colección constituyen una de las ideas clave en la visualización de la información. Cuando los usuarios controlan el contenido de la pantalla, es posible enfocar aquello que es interesante y omitir lo que no. Los deslizadores, botones u otros elementos de control, junto con una rápida actualización (menos de 100 milisegundos) de visualización es el objetivo a conseguir, incluso cuando aparecen decenas de miles de elementos. Del mismo modo, las técnicas de cepillado y la vinculación permiten a los usuarios seleccionar dinámicamente los elementos de interés en diferentes pantallas.

2.2.11. Detalles bajo demanda

Se puede seleccionar un elemento o un grupo de ellos para profundizar en detalle. Una vez que se ha reducido una colección a una docena de artículos, se hace fácil navegar sobre el grupo o un elemento. El método usual es simplemente hacer clic en un elemento y revisar los detalles en una ventana separada o pop-up.

2.2.12. Relacionar tareas

Es posible relacionar elementos o grupos dentro de la colección. Se utilizan indicadores visuales en vez de pantallas de texto, debido a la notable capacidad humana de percibir la información visual. Dentro de pantallas visuales se puede mostrar las relaciones de proximidad mediante líneas conectadas, o mediante un código de colores. Además se pueden utilizar técnicas de realce para llamar la atención sobre ciertos elementos en un campo de miles de artículos. En representaciones visuales el ojo, la

mano y la mente parecen funcionar sin problemas y rápidamente a medida que los usuarios realizan acciones. En un diagrama de Gantt, por ejemplo, los usuarios pueden hacer clic en una tarea y consultar los recursos asociados a dicha tarea. El diseño de la interfaz de usuario sigue siendo un desafío a la hora de especificar las acciones que manifiesta la relación. En la actualidad ya se están desarrollando herramientas que permiten a los usuarios especificar las visualizaciones que necesitan y cómo se debe controlar la interacción entre las diferentes visualizaciones.

2.2.13. Historial de tareas

Se puede mantener un historial de acciones por si prefiere deshacer algún cambio. Es raro que la intervención del usuario al principio produzca el resultado deseado. La exploración de la información es inherentemente un proceso con muchos pasos, por lo que es importante mantener el historial de acciones que permita a los usuarios volver sobre sus pasos. Sin embargo, la mayoría de los productos no trabajan adecuadamente este requisito. Los diseñadores deberían utilizar modelos de sistemas de recuperación de información, para que la búsqueda de información se pueda combinar o refinar.

2.2.14. Extraer tareas

Se permite la extracción de subcolecciones y parámetros de consulta. Es útil para que sean capaces de extraer ese conjunto y guardarlo, enviarlo por correo electrónico, o insertarlo en un paquete estadístico o presentación. También puede darse el caso de que se desee publicar esos datos para que otros puedan visualizar el contenido con una versión simplificada de la herramienta de visualización.

2.2.15. Retos para la visualización de información

El tipo de datos por taxonomía de tarea ayuda a organizar nuestra comprensión de la diversidad de problemas, pero hay todavía muchos otros retos a tener en cuenta si se quiere diseñar una herramienta de éxito:

- **Importar datos.** La decisión sobre la forma de organizar los datos de entrada para lograr un resultado deseado a menudo conlleva más tiempo de reflexión y ejecución de lo esperado. Por lo tanto, puede ser tediosa la obtención de datos en el formato correcto, el filtrado adecuado, la normalización de los valores de atributos, y hacer frente a los datos que faltan.
- **Combinar la representación visual con etiquetas de texto.** Las representaciones visuales son potentes, pero las etiquetas textuales tienen un papel importante. Éstas deben ser visibles sin sobrecargar la pantalla y confundir a los usuarios. A menudo pueden ser de gran ayuda los métodos que permiten el control de etiquetas por el usuario.
- **Sacar información relacionada**. La información adicional es a menudo necesaria a la hora de tomar decisiones. Una asesoría de patentes podría querer ver las patentes relacionadas, otras presentaciones que ha realizado una persona, o limaduras recientes con empresas de la competencia. La investigación genómica

quiere ver cómo grupos de genes trabajan en armonía durante las fases de los procesos celulares, y luego ver genes similares en la Ontología de Genes o leer artículos de investigación relevantes sobre las vías biológicas. La búsqueda de significado durante la investigación requiere un acceso rápido a una fuente rica de información relacionada.

- **Ver grandes volúmenes de datos.** Un desafío general a la visualización de información es el manejo de grandes volúmenes de datos. Muchos prototipos innovadores sólo pueden hacer frente a unos pocos miles de artículos, o tienen dificultades para mantener interactividad en tiempo real cuando se trata de un gran número de artículos. Las visualizaciones dinámicas que muestran millones de datos demuestran que la visualización de información todavía no está aún cerca de alcanzar los límites de la capacidad visual humana, y los mecanismos de control de usuario empujan cada día más. La utilización de pantallas más grandes con píxeles adicionales permiten a los usuarios ver más detalles, manteniendo al mismo tiempo una visión razonable.
- **Integrar la minería de datos.** La visualización de la información y la minería de datos se originó a partir de dos líneas independientes de investigación. Los investigadores que tratan con la visualización de información creen en la importancia de permitir que los sistemas visuales les lleven a hacer hipótesis, mientras que en minería de datos, creen que los algoritmos estadísticos y de aprendizaje automático se pueden utilizar para encontrar patrones interesantes. A veces, los patrones de compra de los consumidores destacan cuando se visualizan correctamente; como los picos de demanda que se dan antes de las tormentas de nieve o las correlaciones entre la cerveza y las compras de galletillas pretzel. Sin embargo, las pruebas estadísticas pueden ser útiles en la búsqueda de las tendencias del consumidor o los vínculos demográficos para la compra de productos. Los investigadores están combinando estos dos enfoques cada vez más. Los resúmenes estadísticos llaman la atención por su carácter objetivo, pero pueden ocultar los valores atípicos o discontinuidades (como los puntos de congelación o ebullición). Por otra parte, la minería de datos puede dirigir a los usuarios a partes más interesantes de los datos que podrían ser inspeccionados visualmente, por ejemplo, mostrando fuertes correlaciones lineales.
- **Colaborar con los demás.** El descubrimiento es un proceso complejo que depende de saber lo que busca, verificar los supuestos de colaboración con los demás, las anomalías, y convencer a otros de la importancia de un hallazgo. El apoyo a los procesos sociales es esencial para la visualización de información; las herramientas de software deberían hacer más fácil registrar el estado actual y enviarlo o publicarlo en un sitio Web con anotaciones y datos.
- **Lograr la facilidad de uso universal.** La fabricación de herramientas de visualización accesibles a diversos usuarios, independientemente de su contexto, las desventajas técnicas, o incapacidades personales, es necesaria cuando las herramientas son para ser utilizadas por el público, pero sigue siendo un gran desafío para los diseñadores. Por ejemplo, los usuarios con discapacidad visual pueden necesitar utilizar alternativas a las basadas en texto utilizando la sonificación de gráficos, diagramas de dispersión y tablas, y en el futuro, el sonido espacial podría ayudar a representar datos más complejos. Las pantallas táctiles empiezan a tener ya gran resolución pero puede ser útil también cumplimentarlas con descripciones de audio. Por desgracia, no están ampliamente disponibles. Para

los usuarios con deficiencias en la percepción de colores se pueden proveer paletas alternativas o herramientas para personalizar los colores. Otro motivo de preocupación es que en sistemas Web, los diseñadores deben trabajar con las limitaciones de las conexiones lentas. Sin embargo, a menudo es posible implementar soluciones inteligentes. La utilización de algoritmos adecuados a la hora de desarrollar una aplicación puede dar lugar a tiempos de descarga que pueden llegar a ser de 5 a 10 veces más potentes.

2.3. Mashup

En desarrollo Web, un Mashup [53] es una página Web o aplicación que usa y combina datos, presentaciones y funcionalidad procedentes de varias fuentes para crear nuevos servicios. El término implica integración fácil y rápida, usando a menudo APIs abiertas y fuentes de datos para producir resultados enriquecedores muy diferentes de la razón original para la que fueron producidos los datos en crudo originales.

Las principales características del Mashup son la combinación, la visualización y la agregación. Es importante transformar los datos existentes en otros más útiles tanto para uso personal como profesional.

2.3.1. Concepto

El contenido usado en Mashup se obtiene de otra fuente vía una interfaz pública o API (*web services*), aunque hay personas en la comunidad que consideran que los casos en que las interfaces son privadas no deberían contar como Mashups. Otros métodos de obtener contenido para Mashups incluyen Web Feeds (i.e RSS o Atom) y screen scraping.

Mucha gente experimenta con Mashups usando las APIs de Amazon, eBay, Flickr, Google, Microsoft, Yahoo o YouTube lo que ha llevado a la creación de editores de Mashup.

La arquitectura de los Mashups está siempre compuesta de tres partes:

- **El proveedor de contenidos:** fuente de los datos. Los datos están disponibles vía una API y diferentes protocolos Web como RSS, REST y Web Service.
- **El sitio Mashup**: es la nueva aplicación Web que provee un nuevo servicio utilizando diferente información y de la que no es dueña.
- **El "Web browser" cliente:** es la interfaz de usuario del Mashup. En una aplicación Web, el contenido puede ser mezclado por los "Web browser clients" usando lenguajes Web del lado del cliente. Por ejemplo JavaScript.

Los Mashups deben ser diferenciados de simples embebidos de datos de otro sitio para formar un documento compuesto. Un sitio que permite al usuario embeber vídeos de YouTube, por ejemplo, no es un sitio Mashup. Como ya se dijo, el sitio mismo debe acceder a la información externa a él usando una API y procesar esos datos de modo que incremente su valor para el usuario.

2.3.2. Tipos de Mashup

Los Mashups se presentan actualmente en tres formas: Mashups de consumidores, Mashups de datos y Mashups empresariales.

El tipo más conocido es el de Mashup de consumidores, que está muy bien ejemplificado por muchas aplicaciones que utiliza Google Maps. Los Mashups de este tipo combinan datos de fuentes varias, escondiéndolos ello tras una interface gráfica simple.

Un Mashup de datos es muy utilizado para recopilar información de muchas fuentes para así enriquecerla, resumirla, compararla o analizarla.

Un Mashup de negocio es una combinación de todo lo anterior, enfocando en agregación de datos y presentación y agregando adicionalmente una funcionalidad colaborativa, haciendo que el resultado final sea una aplicación de negocio apropiada.

Mashups dentro de Mashups son conocidos como “Mashups monstruos”.

Es importante reconocer que los Mashups ayudan o facilitan la integración de aplicaciones orientadas a arquitecturas SOA.

2.4. Aplicación de visualizaciones narrativas

Debido a su flexibilidad en la difusión y el acceso en tiempo real, la utilización del e-learning ha sido ampliamente adoptada en los últimos años. El uso de tecnologías en educación ocurre tanto en entornos e-learning como en entornos tradicionales que cada vez hacen más uso del ordenador u otros dispositivos electrónicos. En las aplicaciones de e-learning, los estudiantes son alentados a aprender a través de la interacción con una amplia gama de recursos para adquirir y construir su conocimiento. Si bien este ambiente de aprendizaje autorregulado y con abundantes recursos permite a los estudiantes una gran libertad y flexibilidad en la búsqueda, selección y recopilación de información, los alumnos pueden sufrir una sobrecarga cognitiva y una desorientación conceptual y de navegación. El reto es aún mayor cuando los contenidos de aprendizaje se encuentran dispersos bajo temas dispares y complejas estructuras de conocimiento. Cuando un alumno se enfrenta a este problema, no es capaz de entender las características y patrones significativos de diversos tipos de información, y son fácilmente obstaculizados por la memoria de trabajo limitado. Esto se debe principalmente a que los novatos no tienen suficiente conocimiento y una comprensión profunda del tema de dominio, que es crucial para organizar la información y el conocimiento para la retención en la memoria a largo plazo. Además, la educación tradicional rompe con la totalidad y se divide en partes, y se centra por separado en cada una de ellas, y los alumnos son a menudo incapaces de agrupar todas ellas y llegar a un conocimiento global. Como resultado, la mayoría de los estudiantes online, especialmente los novatos, se convierten en "perdidos en el hiperespacio".

En [32] se tiene como objetivo mejorar el diseño de los actuales sistemas de e-learning para resolver el problema antes mencionado. Para facilitar el procesamiento cognitivo y el aprendizaje autorregulado, los estudiantes deberían apoyarse en estrategias de aprendizaje adecuadas, dentro de las cuales las cognitivas y metacognitivas ya han sido bien identificadas [34][35][36]. A los estudiantes se les ayuda en su aprendizaje independiente si tienen conocimiento conceptual, y los alumnos pueden llegar a ser más independientes si tienen conciencia de su propio conocimiento y habilidad para entender, controlar y manipular los procesos individuales de aprendizaje. Si bien se han encontrado estrategias con el objetivo de ser eficaces, pocos estudios han examinado cómo pueden ser implementadas en el diseño instruccional, sobre todo en los entornos de aprendizaje online. Mientras que las teorías o estrategias de aprendizaje ofrecen pautas para mejorar el diseño de los actuales sistemas de e-learning, es mucho más difícil y se necesita un esfuerzo adicional para explorar métodos eficaces de instrucción [37].

En [32] se investiga la visualización del conocimiento (KV) enfoque para apoyar el aprendizaje en línea de recursos abundantes y autorregulados, que consta de tres componentes. En primer lugar, una representación explícita de la estructura del conocimiento conceptual se construye mediante la captura de los conceptos clave del conocimiento y sus relaciones en un formato visual. Esta estructura de conocimiento visualizado sirve como un mapa cognitivo para facilitar la construcción del conocimiento y el pensamiento de alto nivel de los estudiantes en línea. En segundo lugar, los conceptos abstractos se conectan con contenidos concretos, vinculando los conceptos de conocimiento con recursos de aprendizaje. De este modo, el procesamiento de la información y construcción de conocimiento, los dos aspectos fundamentales del proceso de aprendizaje, están bien integrados. Los estudiantes pueden navegar fácilmente a través de los abundantes recursos, espacio de conocimiento no-lineal, con la ayuda de una hoja de ruta cognitiva. En tercer lugar, se proporciona a los alumnos el apoyo al aprendizaje meta-cognitivo para regular y planificar su proceso de aprendizaje. Se proporcionan los materiales de evaluación relacionados con los conceptos de conocimiento para la auto-evaluación de los resultados del aprendizaje con el fin de generar un sistema de retroalimentación y una guía a lo largo de su proceso de aprendizaje.

2.4.1. Visualización de la estructura del conocimiento

Al facilitar el procesamiento cognitivo de los estudiantes y retener el conocimiento en memoria a largo plazo, la agrupación o agrupamiento (es decir, la organización de diferentes partes de la información en unidades significativas) es considerado como un enfoque generalizado [38]. Según las teorías en psicología, el conocimiento de la memoria semántica se organiza en redes, construidas pieza por pieza con pequeñas unidades de conceptos interactivos y con marcos proposicionales. Estas redes semánticas mentales representan una estructura cognitiva, que pueden ser utilizadas como una herramienta de aprendizaje para los procesos de aprendizaje constructivos [39]. Más importante aún, la estructura cognitiva debe estar representada en un formato externo con la descripción explícita. Esto se debe a que los métodos visuales ayudan a exteriorizar y obtener la estructura abstracta de conocimiento [40], y los cerebros humanos tienen capacidades de procesamiento rápidos para adquirir y retener imágenes visuales (Paige y Simon, 1966). Basados en computadoras tecnologías, ayudan aún más por lo que es fácil para los estudiantes construir, recuperar y modificar las representaciones visuales, y conservar durante un largo período de tiempo [41].

El enfoque KV incorpora representaciones visualizadas de la estructura de dominio del conocimiento en los sistemas de e-learning. Se han desarrollado funciones relevantes para la creación, almacenamiento, presentación y revisión de los mapas del conocimiento. En lugar de memorizar los contenidos, los alumnos pueden utilizar mapas de conocimiento para identificar los conceptos importantes y sus relaciones, y generar redes semánticas para la revisión y la reflexión. Por otra parte, un mapa de conocimiento muestra los procesos intelectuales que intervienen en la adquisición y construcción del conocimiento. Estos se convierten en la base para la investigación sistémica, la construcción del conocimiento y el pensamiento de alto nivel [42].

2.4.2. Integración del proceso de información y la construcción del conocimiento

El Tratamiento de la información y la construcción del conocimiento están estrechamente entrelazados en el proceso de aprendizaje. Los estudiantes necesitan tener acceso a información para adquirir conocimiento del contenido y formular hipótesis [43]. El conocimiento se construye a través de un aprendizaje significativo, que tiene lugar cuando los alumnos buscan deliberadamente relacionar la información (nueva), e incorporar al conocimiento que ya han poseído [44].

El objetivismo y el constructivismo ofrecen diferentes perspectivas sobre el proceso de aprendizaje, y proporcionan directrices complementarias sobre cómo debe ser engendrado el aprendizaje [43][45]. La teoría de procesamiento de la información se refiere a un ser humano como un procesador de información o una "mente como la del ordenador" [46] se basa en el paradigma objetivista, que asume que el conocimiento tiene una existencia objetiva e independiente cuyos atributos, relaciones y estructura pueden ser conocidas, y que el aprendizaje es la transmisión de los conocimientos de los profesores a los alumnos [47]. El paradigma constructivista, por otra parte, se propone que el conocimiento no es independiente del alumno, pero está construido internamente por el estudiante como una manera de hacer sentido de ejercicio [48]. En [32] se intenta integrar y facilitar tanto el procesamiento de la información objetiva como la construcción del conocimiento subjetivo para e-learning mediante la vinculación de los conceptos de conocimiento en mapas visuales con recursos de aprendizaje.

2.4.3. Facilidad en la autorregulación del aprendizaje

El conocimiento avanzado se adquiere con años de experiencia y se deriva de las actividades del pensamiento, de la acción y de la reflexión. Los expertos han adquirido un gran conocimiento de los contenidos bien organizados, y su organización refleja un profundo conocimiento de la materia [49]. Aunque los modelos de pares se han utilizado para guiar el aprendizaje autorregulado, la creación de estructuras cognitivas estables y bien conocidas para cimentar el aprendizaje avanzado se ha denotado como objetivo de instrucción primario [50]. El reconocimiento de expertos en conocimiento se ha reflejado en las teorías del aprendizaje, tanto objetivista como constructivista. El tratamiento de la información, que se basa en la teoría objetivista de aprendizaje, requiere un proceso eficaz y eficiente de la información e indica que las estructuras de los conocimientos ayudan a los estudiantes a adquirir información precisa. Al mismo tiempo, el constructivismo sugiere que la orientación y estrategias (por ejemplo, el modelado,

entrenamiento y andamiaje) proporcionan el apoyo necesario para que los alumnos construyan el conocimiento [45]. Este estudio utiliza las estructuras de conocimiento para guiar y comprender los conocimientos iniciales, el pensamiento y la investigación en su aprendizaje autorregulado. La comprensión conceptual de un dominio a menudo no se expresa plenamente en los libros o materiales de aprendizaje, y los mapas de conocimiento pueden ser utilizados para articular y manipular el conocimiento tácito con mayor eficacia. Se puede utilizar el mapa de conocimiento para reducir la posibilidad de error y las ideas erróneas. Aunque los gráficos altamente estructurados pueden parecer restrictivos, a veces, estas plantillas son un buen punto de partida para los principiantes, que tienen problemas para organizar su comprensión y se confunden en su aprendizaje autorregulado [51].

Además de facilitar los procesos cognitivos de los alumnos, los mapas de conocimiento meta-cognitivo proporcionan apoyo. Como se ha mencionado, los alumnos pueden llegar a ser más independientes si son conscientes de su proceso de aprendizaje y tienen la capacidad para regularlo. Las representaciones visuales son formas de la metacognición que muestran gráficamente el proceso de pensamiento [52]. Los mapas de conocimiento visualizan los procesos intelectuales que se representan por secuencias, alternativas, sucursales, puntos de elección, y las vías que intervienen en la adquisición y construcción del conocimiento.

2.4.4. Learning Analytics

El análisis del aprendizaje es la posibilidad de medir, recoger, analizar y representar los datos relativos al aprendizaje con el fin de evaluar el progreso académico, mejorarlo, tomar decisiones, predecir el rendimiento futuro y detectar posibles problemas. Puede ser la próxima revolución dentro de la formación.

George Siemens propone un marco para el análisis de datos del aprendizaje llamado Open Learning Analytics [54] que recoge y muestra datos desde tres puntos de vista: El alumno, el profesor y el administrador.

Se pueden definir tres tipos de dato [55]:

- Los basados en números puros.
- Los datos derivados de las interacciones sociales, el grafo social
- Análisis de calidad. Para medir la calidad de las aportaciones e interacciones.

2.4.4.1. Obteniendo y explotando los números puros

Una buena estrategia debería partir de analizar qué datos, basados en números puros, interesa recoger. Por ejemplo el número de veces que se visualiza un contenido, el tiempo que un usuario permanece viendo un video, el número de veces que el usuario accede al sistema, el tiempo de permanencia en el sistema.

Una vez que una aplicación registra lo que se quiere que registre, se debe decidir cómo explotar los datos. Se puede optar por:

- Desarrollar un asistente para la creación de informes avanzados. No consiste en unos informes predefinidos solamente si no dar la posibilidad al administrador de diseñar su propio informe.

- Utilizar una aplicación externa para explotar los datos duros, como Talend [56], Blackboard Analytics [57] o Pentaho [58].

Figura 10: Blackboard analytics

2.4.4.2. El grafo social

El grafo social es un concepto que busca describir de forma gráfica cómo son las relaciones entre nodos de una red.

Almacenarlo en una base de datos relacional no es nada fácil, al final se trata de almacenar una serie de relaciones complejas entre nodos en una base de datos que no está pensada para ello, pero existen aproximaciones excelentes como la que explica Lorenzo Albertón [59].

Otra posibilidad es almacenar el grafo en una base de datos no relacional como mongo.

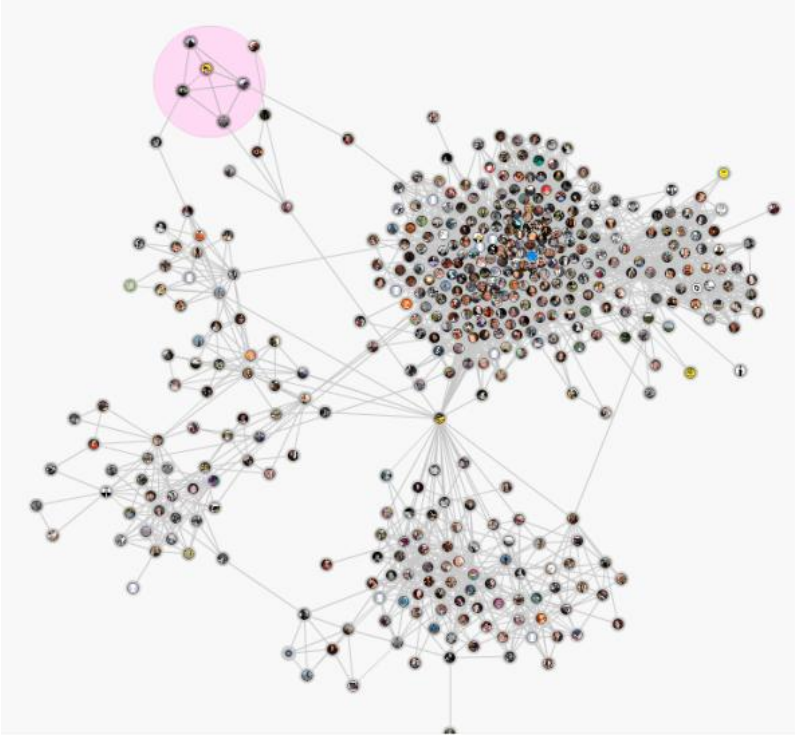


Figura 11: Ejemplo de grafo social

2.4.4.3. Análisis de calidad

No basta con saber la cantidad de participaciones e interacciones de un alumno (los números duros), es igual de importante saber la calidad de sus participaciones e interacciones. ¿Lo que comparte el alumno es interesante para otros compañeros? ¿Cuántos comentarios hay en cada uno de sus post?

En este sentido es importante planificar un buen sistema que valore los recursos (enlaces, mensajes en foros, documentos, artículos, etc.). Pero no todos los recursos son fáciles de valorar; se puede dar la oportunidad a los alumnos de que valoren cada recurso con un sistema de estrellas (por ejemplo otorgando de 1 a 10 estrellas a cada recurso) pero es muy posible que no todos los alumnos colaboren con este sistema.

Una buena forma de fomentar que el alumno colabore valorando los recursos es "gamificar" el entorno y darle puntos simplemente por gastar cada mes sus "50 estrellas" repartiéndolas entre los recursos que le resulten más interesantes

2.4.4.4. ¿Para qué?

Este análisis de datos debe llevar a comprender mejor lo que está sucediendo en el proceso formativo pero también debe facilitar la toma de decisiones y recomendaciones con carácter predictivo

Moodle tiene un interesante bloque (Figura 12) llamado Analytics and Recommendations [60] que, entre otras cosas, permite al alumno ver en qué situación está respecto al resto.

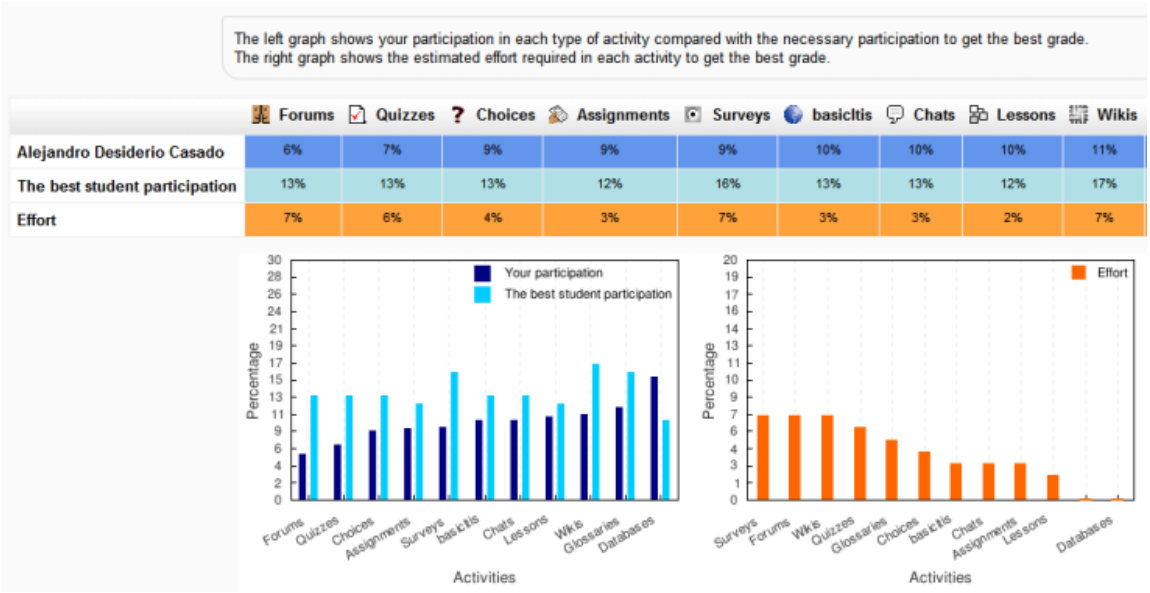


Figura 12: Analytics and Recommendations; Moodle.

Es importante entender que el análisis de datos contiene información interesante para todos los implicados en el proceso: el administrador, el profesor y el alumno. Proporcionar información en tiempo real al alumno sobre si su interacción está siendo buena, si se relaciona, si aporta cosas de calidad, cuáles son sus puntos fuertes y en cuáles debe mejorar es clave en el modelo.

El grafo social puede ayudar a detectar alumnos “desconectados” y ayudarles a volver a participar del proceso formativo.

Por otro lado un buen análisis de calidad puede facilitar el desarrollo de un buen sistema de recomendaciones para el alumno. Por ejemplo, se puede recomendar a un usuario que siga al máximo experto en la red en las materias que a él más le interesen.

2.5. Conclusiones

Las interfaces de usuario mejoradas como las bibliotecas digitales y bases de datos multimedia han dado lugar a atractivos productos. En contra de un texto complejo, se están utilizando consultas flexibles, sonido, gráficos, imágenes, bases de datos y vídeos. Es ahora posible que junto a los paquetes estadísticos y la formulación de consultas se añadan aproximaciones gráficas de manipulación directa.

La visualización de información se está moviendo fuera de los laboratorios de investigación con un número cada vez mayor de productos comerciales, añadidos a los paquetes estadísticos, y al desarrollo de ambientes comerciales, y como en el caso de este proyecto, para la mejora de la educación y el e-learning. Si se busca por la Web se pueden encontrar muchos recursos para diseñadores y desarrolladores, como XMD vTool o Xgobi. Los nuevos productos proporcionan una integración sin problemas con el software existente y apoyan el conjunto de las tareas mencionadas: información general, zoom, filtrado, detalles a la carta, relación, historia y extracción. Estos productos son atractivos porque presentan información de manera rápida y controlada por el usuario

permitiendo la exploración. Para ser plenamente eficaz, se requieren estructuras de datos avanzadas, pantallas de alta resolución en color, recuperación rápida de datos, y nuevas formas de capacitación de usuarios. Se deben hacer pruebas con cuidado para ir más allá del deseo de conseguir una interfaz "cool" e implementar diseños que lleguen a ser beneficiosos para las tareas reales.

Aunque el ordenador contribuye a la explosión de la información, es también potencialmente la lente mágica para buscar, ordenar, filtrar, y presentar los productos en cuestión. La necesidad de buscar en documentos estructurados complejos, gráficos, imágenes y archivos de sonido o video presenta grandes oportunidades para el diseño de interfaces de usuario avanzadas. Potentes motores de búsqueda serán capaces de encontrar las agujas en los pajares y los bosques más allá de los árboles. Los métodos que pueden utilizar las herramientas de exploración de información, tales como consultas dinámicas, diagramas de árbol, interfaces de usuario, interfaces jerárquicas y coordenadas paralelas, son sólo algunos de los inventos que tendrán que ser domesticados y validados por los investigadores de la interfaz de usuario. Es necesaria una mejor integración con la psicología perceptual (comprensión preatencional de los procesos y el impacto de la codificación o variado destacando técnicas) y con la toma de decisiones empresariales (identificando tareas y procedimientos que se dan en situaciones reales), como lo son los fundamentos teóricos y prácticos para la elección de los puntos de referencia entre las diversas técnicas de visualización. La exploración de la información puede resultar abrumadora en interfaces complejas para los nuevos usuarios y pueden ser útiles los nuevos métodos de demostración. Los estudios empíricos ayudarán a resolver la situación específica en la que la visualización es de gran ayuda.

Hay que tener cuidado en la forma de mostrar esta información para no saturar a un estudiante, hay que saberle dirigir en el aprendizaje, remarcándole aquello que es más importante para que vaya adquiriendo la información de una manera eficaz. Es muy importante no desmotivarlo ni que caiga en la falsa sensación de tener todo controlado. La información tiene que llegar de una manera simple pero a su vez llegar a ser detallada si se quiere profundizar más. Una manera de conseguir todo esto será gracias al diseño adecuado de visualizaciones (siempre se ha dicho que una imagen vale más que mil palabras). Por ello las visualizaciones tienen que ser simples; con un rápido vistazo hay que ser capaz de ver lo relevante pero que gracias a su potencial ofrezca la posibilidad de poder indagar más en los datos. Además tienen que ser fáciles de comprender e intuitivas, y si la información que muestran va cambiando a lo largo del tiempo, es importante que muestre su evolución y que refleje el cambio; una buena visualización tiene que incitar a que la vuelvan a ver, y si es interactiva, que su atractivo incite a ser reutilizada, que no sea un simple gráfico del que una vez visto te puedas olvidar de ella.

Como concepto actual surge el análisis del aprendizaje para medir, recopilar, analizar y presentar los datos de los alumnos y su contexto. Muchas de las técnicas que se han expuesto a lo largo del apartado pueden utilizarse para tal concepto. Dependiendo de la visualización, el tópico que se trate, la persona y el aprendizaje habrá que utilizar una diferente o intentar innovar aún más.

Capítulo 3

GLASS

3.1. Introducción

Partiendo de una estructura de datos creada en MongoDB se representan los eventos resultantes de la monitorización de usuarios en un entorno de aprendizaje. Así se capturan tantos tipos de eventos como registros sea posible no sólo para conocer lo que hacen los usuarios, sino para ir un paso más, para conocer el comportamiento seguido en el uso de su entorno de trabajo. El problema surge en que, por muchos datos que se capturan, carecen de significado sin una adecuada visualización.

Con motivo de solucionar este problema, se concibe el proyecto GLASS (Gradient's Learning Analytics SyStem), donde se pretende diseñar e implementar una herramienta que permita visualizar el conjunto de eventos capturados siguiendo un esquema flexible definido para GLASS, con el fin de conseguir una interpretación satisfactoria de los mismos.

A lo largo de este apartado se procede a especificar los requisitos necesarios que debe contener el sistema de visualización, reuniendo los diferentes aspectos claves que conforman la plataforma. En él se discuten los diversos factores que pueden ser críticos en la implementación del mismo, así como el aspecto general que debería tener la plataforma.

Este apartado se divide en 3 secciones. En la primera, se describen los aspectos generales el sistema. Para ello se describen los objetivos generales de la herramienta y se hace una descripción del sistema y de las bases de datos que utiliza. En la segunda sección se describen el hardware y el software necesario, las limitaciones que puede tener

la plataforma y una descripción de los archivos que forman la herramienta. En la tercera sección se describe brevemente la interfaz de usuario donde se describen todos los puntos de contacto entre el usuario y el equipo, tanto para el manejo de GLASS como para su instalación.

3.2. Descripción general del sistema

3.2.1. Objetivos

El objetivo general de la herramienta consiste en poder visualizar de forma clara y concisa, los eventos recogidos en un formato definido en MONGO [61], ofreciendo en todo momento la capacidad de guardar y compartir las visualizaciones.

La herramienta es lo suficientemente configurable como para poder entender los eventos, perdiendo la menor cantidad de información posible, abstrayéndose de aquella información que no es necesaria, independientemente del usuario que la utilice. Éste siempre ha sido un punto crucial en GLASS, ya que según el usuario y los intereses del mismo, querrá conocer un comportamiento que para otro usuario pueda que no tenga ningún significado.

La herramienta se ha diseñado para ser lo más amigable y reutilizable posible. Se ha intentado que sea una aplicación que despierte el interés por si sola. Despierta la curiosidad por lo que ofrece y sea capaz de ofrecer resultados de forma rápida, fácil y concisa. Para ello la configuración y el filtrado se realizan de forma clara intentando que el usuario realice el menor número de pasos posible.

La aplicación es escalable. Se basa en una tecnología modular para que vaya creciendo con el tiempo. Cada módulo se diseña de forma fácil y es completamente adaptable a la herramienta, independientemente de lo que ya esté instalado en ella. Conocer la forma de instalar un módulo es un punto clave para que futuros desarrolladores puedan familiarizarse con la tecnología de una manera fácil y rápida, y así poder desarrollar a partir de ésta.

Por último se ha implementado la plataforma con el fin de ser una base para la investigación, es decir, que gracias a ella se puedan sacar buenas conclusiones del entorno de trabajo independientemente de éste. Además ofrecer la posibilidad de poder compartir visualizaciones entre usuarios con el fin de poder guardar configuraciones.

3.2.2. Descripción del sistema

3.2.2.1. Arquitectura

GLASS es un elemento integrado en un sistema compuesto por los siguientes elementos:

- **Sensor:** envían metadatos de las acciones realizadas a la base de datos**.**

- **Base de datos:** guardan la información de los eventos capturados. El formato o estructura de datos se explicará más adelante
- **GLASS:** Objeto del proyecto. Es necesario un visualizador que sea capaz de representar la información capturada por los sensores y almacenada en la base de datos.
- **Usuario:** Persona que comprende e interpreta razonadamente los datos obtenidos con el visualizador.

El resultado que se pretende tener al final es un conjunto de usuarios que se conectan desde sus navegadores Web al visualizador. Se pueden tener diferentes visualizaciones, ya sea por las que tiene configurado cada usuario, o por la instalación que haya hecho el usuario del mismo. Cada visualizador a su vez se puede conectar a un número de bases de datos Mongo, o simplemente a una sola. Esto dependerá del entorno de visualización creado para su representación y la información necesaria para la visualización que en esos momentos esté pidiendo cada usuario. Por último, es necesario que la base de datos vaya creciendo con el tiempo, no que sea estática; es necesario que haya una serie de sensores que forman el entorno de captura de eventos y que van llenando poco a poco la base de datos. En la Figura 13 se puede ver un esquema de cómo interactúan los diferentes elementos que conforman el sistema total.

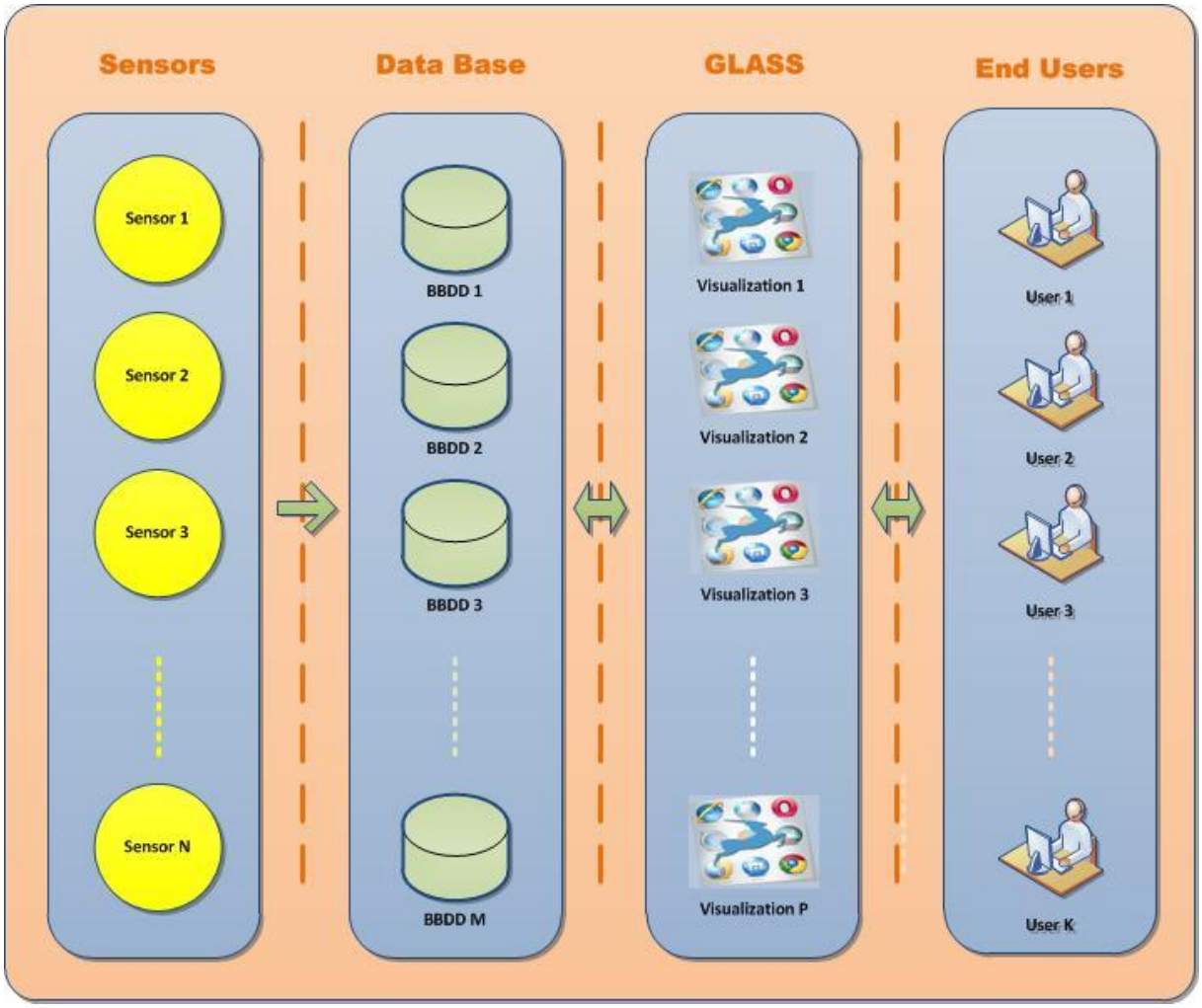


Figura 13: Sistema

GLASS define dos estructuras JSON para interpretar los datos, una para los eventos y otra para los usuarios de los eventos. Por un lado, los eventos se almacenan de acuerdo a la estructura JSON, como se puede ver en la Figura 14. Se pueden ver 3 campos claves que deben guardar el formato especificado y varios campos opcionales como se muestra a continuación:

- **datetime:** fecha y hora en la que el evento ha tenido lugar.
- **event_type:** tipo de evento capturado.
- **user:** entidad que ha realizado el evento. Este elemento se define como un array ya que un evento puede estar compuesto por varios usuarios. Cada elemento del array debe necesariamente contener un id y todos los campos opcionales que se deseen. Cada campo opcional puede tener una clave (cuyo nombre es especificada por el usuario) y sólo un valor.
- **Campos opcionales:** siguen una estructura de clave-valor. El nombre con que se define la clave puede ser especificado por el usuario.

```
event={
   datetime:'_______',
   event_type:'_______',
   user:[
      {id:'____',optional_A:'____', optional_B:'____',…},
      {id:'____',optional_A:'____', optional_B:'____',…},
      …
   ],
   optional_1:'_______',
   optional_2:'_______',
   …
}
```

Figura 14: Esquema de eventos en MONGO

Por otro lado, los usuarios se almacenan con la estructura JSON que se puede ver en la Figura 15. El usuario sólo debe contener un campo obligatorio definido como "id" el cual debe ser el mismo identificador que se especificó en la estructura de los eventos y es utilizado para enlazar ambas "tablas". A parte de "id" se pueden añadir tantos campos como sea necesario, manteniendo la estructura clave-valor. El nombre de la clave pude ser especificado por el usuario.

```
user={
   id:'_______',
   optional_I:'_______',
   optional_II:'_______',
   …
}
```

Figura 15: Esquema de usuario en MONGO

Para instalar una nueva base de datos en MONGO, sólo es necesario especificar los siguientes campos:

- **Nombre de usuario:** nombre del usuario con acceso a la base de datos de Mongo.
- **Clave:** clave secreta del usuario anterior.

- **Equipo:** equipo donde se localiza la base de datos.
- **Base de datos:** nombre de la base de datos que almacena las estructuras ya comentadas.

Una vez conocida la estructura de la base de datos se procede a comentar la estructura de la herramienta. En la Figura 16 se muestra un diagrama básico de la estructura de visualizador, dónde se representan las relaciones que los diferentes elementos tienen entre ellos.

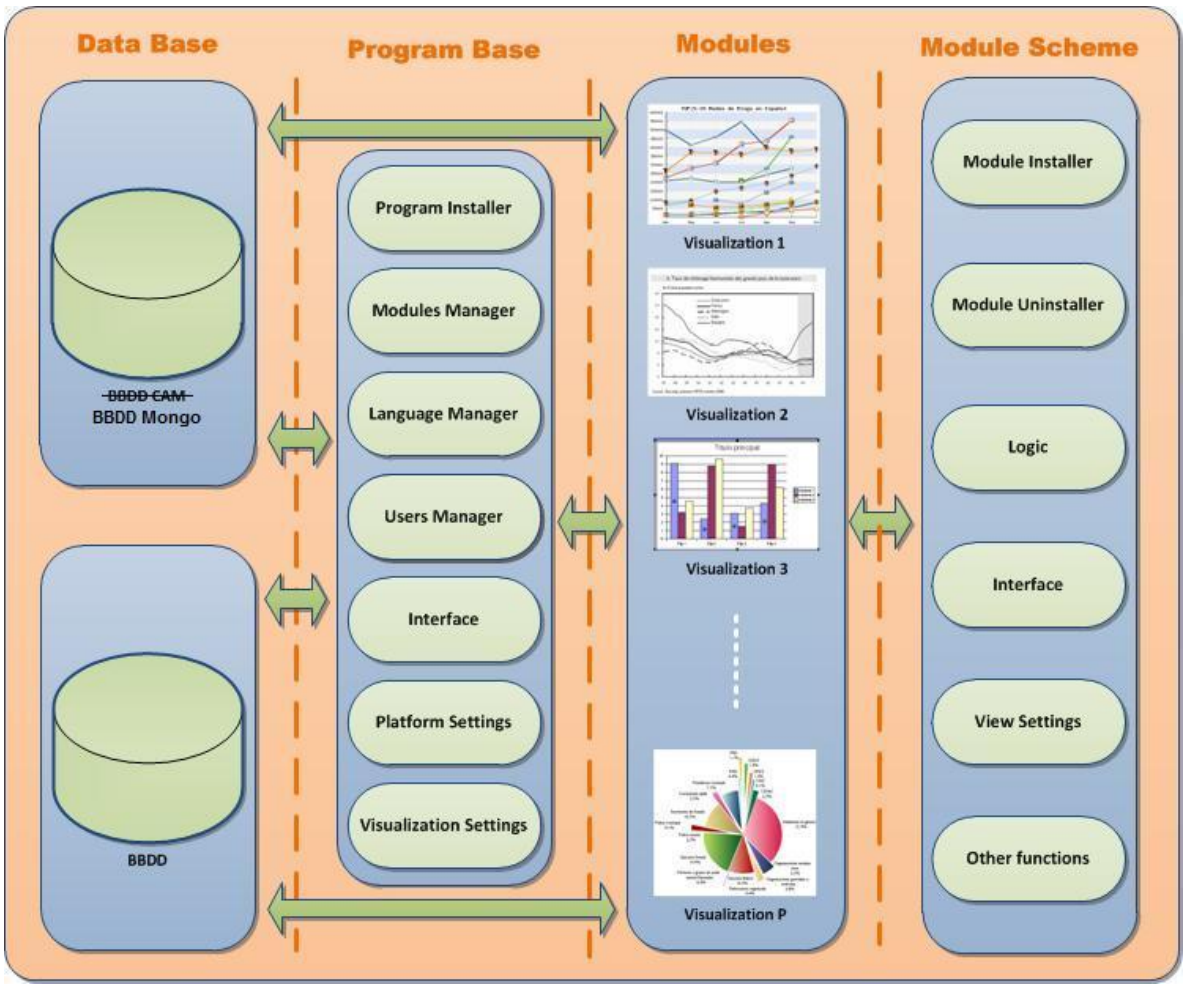


Figura 16: Estructura de la herramienta

Se ha hablado que la herramienta tiene que conectarse a la base de datos MONGO para poder así generar las visualizaciones, pero ésta no es la única base de datos necesaria para el funcionamiento de la herramienta, ya que se necesita de una base de datos adicional que utilice la herramienta para la configuración y funcionamiento de la misma. El diagrama de entidad-relación de dicha base de datos es el que aparece reflejado en la Figura 17.

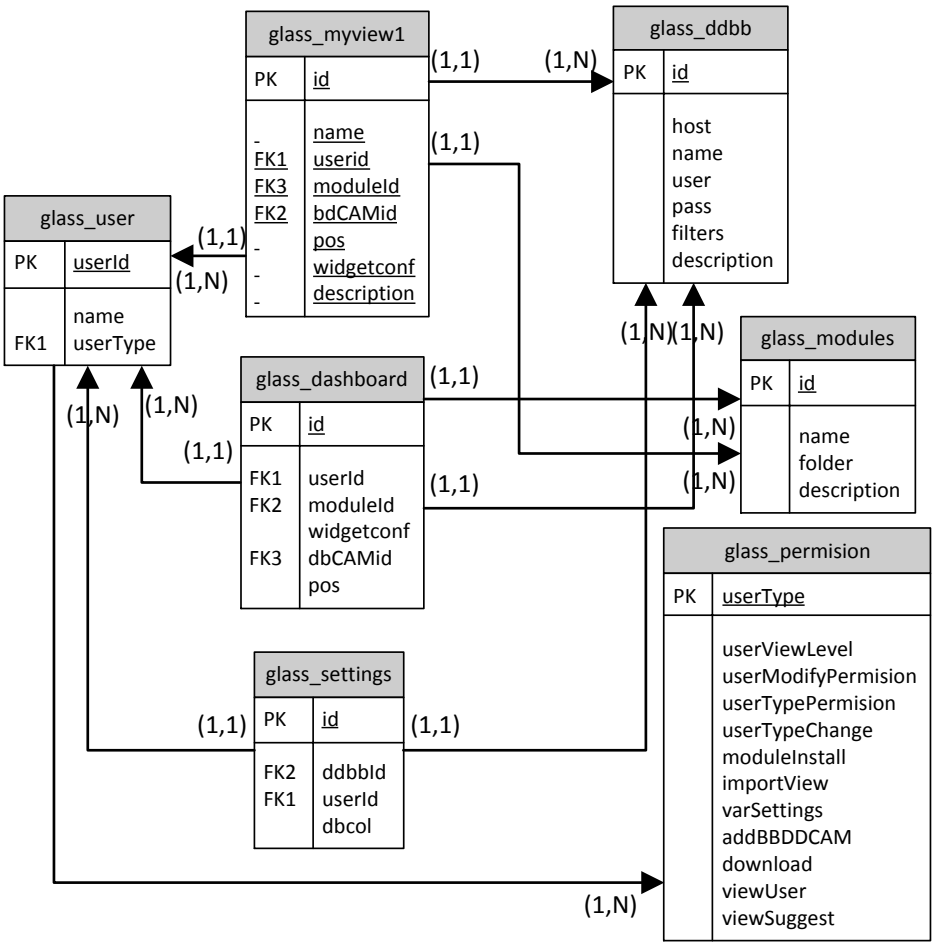


Figura 17: Diagrama ER base de datos operativa de GLASS

La tecnología utilizada para la plataforma debe ser lo más modular posible. Por un lado tiene que haber un código base donde se sustenten los diferentes módulos de la aplicación. En un principio´, cada módulo va a ser una visualización, pudiéndose instalar tantos módulos como visualizaciones estén disponibles. Por eso, el código base de la herramienta se encargará de las siguientes tareas:

- **Instalador del programa:** Secuencia que hace posible la instalación en el servidor de la herramienta tanto de la base de datos MONGO como de la base de datos creada para el funcionamiento del mismo.
- **Gestor de módulos:** Se encarga de instalar o desinstalar los módulos disponibles en la herramienta. Como requisito previo, será necesario que la plataforma localice los módulos disponibles automáticamente.
- **Gestor de idiomas:** Gestiona todo lo referente al cambio de idioma.
- **Gestor de usuarios:** Se encarga de instalar o desinstalar los módulos disponibles en la herramienta. Como requisito previo, será necesario que la plataforma localice los módulos disponibles automáticamente.
- **Interfaz:** Se encarga de mostrar la interfaz que tiene el programa.
- **Ajustes de la plataforma:** Recoge todo aquello que sea necesario para establecer las configuraciones básicas del programa.

- **Ajustes de la visualización:** Reúne el conjunto de filtros que son comunes para todos los módulos, es decir, para todas las visualizaciones.

Por último, se muestra la arquitectura que debería tener cada módulo. Cada uno debe estar constituido por las siguientes partes:

- **Instalador:** Recoge la secuencia de procesos que tiene que tener el módulo para que pueda operar en la herramienta, en su mayoría se encargará de insertar los nuevos elementos y tablas en la base de datos.
- **Desinstalador:** Se encarga de borrar las entradas que se hayan creado en la base de datos para dejar de estar operativas en la herramienta.
- **Filtros visuales:** Reúne la configuración necesaria para cada vista.
- **Interfaz:** Todo lo relativo a la representación por pantalla de la visualización.
- **Lógica de visualización:** Lógica necesaria para mostrar la visualización.
- **Otras funciones:** Cualquier otra función que pueda reunir una visualización.

3.2.2.2. Modelo de usuario

Se pueden definir 4 modelos de usuario cuyos permisos pueden ser configurados en todo momento.

- **Administrador:** Tiene control total de la plataforma, puede agregar y borrar visualizaciones, configurar la plataforma, etc.
- **Instructor:** Tiene sólo control de visualización. No puede configurar nada de la plataforma pero puede configurar y ver todas las visualizaciones que quiera dentro de las disponibles.
- **Observador:** Posee permisos especiales de visualización; puede ver la plataforma pero no puede conocer datos personales. Los permisos que tenga serán, de los cuatro roles disponibles, los más cambiantes, dentro de que todos son configurables.
- **Estudiante:** Sólo puede ver la información relativa a su identidad y algunas estadísticas generales.

La plataforma, según el usuario que esté conectado, muestra ciertos elementos de interfaz en función de los permisos de cada uno, así, un usuario tendrá mayor cantidad de herramientas disponible que otro. Además, según el usuario que esté conectado, se muestra la información guardada, su última configuración y aquellas visualizaciones que se centren en datos personales se adaptan a dicho usuario.

La herramienta se ha desarrollado para que se autentifique vía LDAP, pero viene diseñada de acuerdo a que con algunos conocimientos de programación se implemente un acceso mediante cualquier otro método. Cuando se realiza un acceso toda la parte de gestión de usuario corre a cargo de LDAP y la plataforma sólo se encarga de guardar la configuración de perfil de cada usuario. Debido a que se trata de una plataforma de código abierto, la persona nombrada como administrador tiene la responsabilidad de encargarse de su gestión e instalación, de administrar los usuarios y mantener la seguridad.

3.2.3. Descripción de las tablas de la base de datos operativa

El esquema de cada una de las tablas es el siguiente:

3.2.3.1. glass_dashboard

Esta tabla guarda la información necesaria para imprimir cada *Widget* en el *Dashboard*.

Field	Type	Null	Default	Comments
id	int(11)	No		Auto incremental
userId	int(11)	No		Id en glass_user
moduleId	int(11)	No		Id en glass_modules
widgetconf	varchar(500)	Yes	NULL	Aquí va la información necesaria para imprimir el *Widget*. Debería ser un String Json
bdCAMid	int(11)	No		Id en glass_ddbb
pos	int(11)	No		Posición del *Widget* en el *Dashboard*

Tabla 2: Información del Dashboard

3.2.3.2. glass_ddbb

Esta tabla guarda las diferentes bases de datos Mongo que GLASS puede analizar.

Field	Type	Null	Default	Comments
id	int(11)	No		Auto incremental
host	varchar(30)	No		Dirección del host
name	varchar(30)	No		Nombre de la base de datos
user	varchar(30)	No		Nombre del usuario
pass	varchar(30)	No		Clave secreta de la base de datos
filters	varchar(500)	No		Cadena en Json con los filtros seleccionados en la instalación
description	varchar(500)	Yes	*NULL*	Descripción de la base de datos

Tabla 3: Información de las bases de datos Mongo

3.2.3.3. glass_modules

Esta tabla guarda la información necesaria de cada módulo.

Field	Type	Null	Default	Comments
id	int(11)	No		Auto incremental
name	varchar(30)	No		Nombre del módulo
folder	varchar(30)	No		Nombre del directorio dentro del directorio de módulos
description	varchar(200)	Yes	*NULL*	Descripción del modulo

Tabla 4: Información de los módulos

3.2.3.4. glass_myview

Esta tabla guarda la información necesaria para poder generar las diferentes vistas favoritas de cada usuario.

Field	Type	Null	Default	Comments
id	int(11)	No		Auto incremental
name	varchar(60)	No		Nombre
userid	int(11)	No		Id en glass_user
moduleId	int(11)	No		Id en glass_modules
bdCAMid	int(11)	No		Id en glass_ddbb
pos	int(11)	No		Posición en la lista
widgetconf	varchar(500)	No		Here goes the necessary information to print the widget. It should be a Json string
description	varchar(500)	Yes	*NULL*	Descripción

Tabla 5: Información de las vistas favoritas

3.2.3.5. glass_permision

Esta tabla guarda los diferentes permisos de los modelos de usuario de GLASS.

Field	Type	Null	Default	Comments
userType	varchar(30)	No		Modelo de usuario
userViewLevel	int(11)	No		Nivel de usuario
userModifyPermision	Binar	No		Permiso de modificación de permisos de usuario
userTypeChange	Binar	No		Permiso para cambiar el tipo de usuario
moduleInstall	Binar	No		Permiso para instalar módulos
importView	Binar	No		Permisos para importar las vistas favoritas
varSettings	Binar	No		Permiso para cambiar las variables de configuración
addBBDDCAM	Binar	No		Permiso para agregar una nueva base de datos
download	Binar	No		Permiso para descargar
viewUser	Binar	No		Permiso para poder ver todos los usuarios de la base de datos
viewSuggest	Binar	No		Permiso para poder recibir una sugerencia de metadatos. Actualmente no utilizado

Tabla 6: Información sobre los permisos

3.2.3.6. glass_settings

Esta tabla guarda la información necesaria de la configuración propia de cada usuario.

Field	Type	Null	Default	Comments
id	int(11)	No		Auto incremental
ddbbId	int(11)	No		Id en glass_ddbb
userId	int(11)	No		Id en glass_user
dbcol	int(11)	No		Número de columnas en el *Dashboard*

Tabla 7: Información sobre las configuraciones de los usuarios

3.2.3.7. glass_user

Esta tabla guarda la información de cada usuario

Field	Type	Null	Default	Comments
userId	int(11)	No		Auto incremental
name	varchar(30)	No		Nombre de usuario
userType	varchar(30)	No		Tipo de usuario en GLASS. Se utiliza para asignar permisos

Tabla 8: Información del usuario

3.3. Entorno de la herramienta

3.3.1. Herramienta

Las tecnologías utilizadas para el desarrollo de la herramienta son:

- **Linux:** aunque la aplicación puede ser instalada también en Windows debido a que utiliza como tecnologías bases las definidas por LAMP (Linux, Apache, MySQL, PHP) y WAMP (Windows, Apache, MySQL, PHP).
- **Apache, PHP5 y MySQL:** muy útil la utilización de paquetes WAMP (Windows) o LAMP (Linux) para la integración de dichas tecnologías con el sistema operativo. Entorno del lado del servidor para todo lo referente a las consultas a la base de datos.
- **HTML5, JavaScript y Ajax:** Contenido gráfico y dinámico con tecnologías de última generación con lenguajes del lado del cliente para limitar la carga en el servidor y procesar lo datos en tiempo real.
- Editores de textos que entiendan los lenguajes de programación. Los más utilizados han sido UltraEdit, Notepad++ y PHPdesigner en entorno Windows y Geany y Kedit para entorno Linux.

Como ya se ha comentado, el software es código abierto y posee un código de licencia básico adjuntado como archivo de texto.

3.3.1.1. Limitaciones

A la hora de desarrollar la herramienta o cualquier modulo es necesario tener en cuenta las siguientes limitaciones:

- **Tiempo de respuesta:** el tiempo de respuesta a la hora de la realización de consultas a la base de datos tiene que ser lo más rápido posible y nunca superior a unos pocos segundos, si no la experiencia del usuario con la aplicación se verá notablemente comprometida. A la hora de la realización de las diferentes consultas a la base de datos, es necesaria la mayor optimación posible por lo que se ha dedicado un principal interés a la forma de hacer las consultas. Se han utilizado medidas como almacenamientos por clave para disminuir el tiempo de búsqueda una vez almacenados los datos de la base de datos en arrays o la utilización de *map-reduce* en MONGO.
- **Aplicación Web:** el navegador puede ser una limitación en el momento de ser la ventana para el uso de la aplicación. Además se supone que el usuario dispone de un acceso a Internet, o un acceso a la red en el caso de que el servidor esté instalado en una red local y se quiera acceder a través de ella.
- **Mongo:** es necesario conocer el formato de la base de datos y ceñirse a él para poder realizar consultas. Se ha especificado una estructura lo más versátil posible, para que pueda ser adaptado al mayor número posible de tipos de datos. Los filtros se generan en tiempo real según la base de datos que está corriendo en ese momento.
- **Tecnología cliente-servidor:** La aplicación corre en un servidor y se puede acceder a ella gracias a una interfaz Web, por lo que hay que tenerlo mucho en cuenta en el momento de su instalación, puesta en marcha y utilización de la herramienta. El acceso a una tecnología de este tipo ofrece menor rendimiento que una aplicación que se ejecute en modo local ya que el acceso múltiple puede llegar a saturar el servidor.
- **Lenguajes de programación:** es necesario tener un navegador de última generación para poder funcionar con la herramienta. Se supone por lo tanto que el usuario opera con uno de la amplia lista de navegadores que soporta los lenguajes de programación elegidas. No obstante, cuando se inició este proyecto, se recomendaba la utilización de Google Chrome para asegurar el prefecto funcionamiento de la herramienta.
- **Seguridad:** en el desarrollo de la herramienta o en futuros desarrollos se ha tenido o se debe de tener muy en cuenta la Ley de protección de datos de carácter personal (LOPD) y la sensibilidad que éstos puedan tener.

3.3.2.Archivos

GLASS se compone de un conjunto de archivos que contienen los scripts necesarios para el correcto funcionamiento de la plataforma. Los archivos se dividen de acuerdo con las diferentes operaciones que desempeñan. El conjunto de archivos es el siguiente:

- **Directorio raíz:** Aquí se encuentran los archivos de configuración, características de inicio y características menos complejas A continuación se describen algunas:

- **Login:** Está formado por los archivos *access.php* and *index.html*. Permite hacer login en la plataforma, enlazar con la consulta al LDAP, crear el usuario si es la primera vez que éste accede, e iniciar el proceso de instalación de la herramienta si no ha sido instalada previamente.
- **Logout:** Permite hacer desconexión de la sesión. El archivo que se encarga de tal cometido es *logout.php*.
- **Variables de configuración:** Contiene las variables de configuración de la herramienta. La compone el archivo *config.php*. Si hay un problema en el proceso de instalación se pueden editar estos valores para asegurarse del correcto funcionamiento de la plataforma. El archivo *install.sql* del directorio *install* contiene un script SQL necesario para instalar la herramienta. Para cualquier posible error en la instalación de la herramienta es recomendable echarle un vistazo a este archivo.
- **Documentación:** Lo componen el archivo *documentation.php* script PHP encargado de enlazar a una página HTML que muestra la documentación disponible por la plataforma, y el archivo *credits.php* que incluye información sobre el grupo de trabajo.
- **Página de inicio:** La conforma el archivo *home.php*. Este archivo se encarga del manejo del contenido del *Dashboard*.
- **Configuración:** Está formado por el archivo *settings.php*. El archivo enlaza con el directorio *install* que se encarga de la instalación como se indica más adelante.

- **Directorio *Install*:** Contiene los archivos necesarios para instalar GLASS. Este proceso principalmente guarda diferentes variables de configuración del servidor en la base de datos. El archivo *installprocess.php* es el responsable de tal tarea. Este script edita el contenido de *config.php* e *install.sql* también de ejecutar el anterior script en la base de datos MySQL. Ésta carpeta también incluye todos los procesos de configuración de la herramienta que se pueden ver en la sección de configuración del menú de GLASS.
- **Directorio *Fview*:** Contiene todos los scripts necesarios para gestionar la sección de mis vistas favoritas
- **Directorio *lib*:** Contiene un conjunto de archivos y directorios necesarios para los desarrolladores para la implementación de los diferentes módulos o procesos internos de la herramienta. Dentro de la documentación de la herramienta se puede encontrar la información necesaria de las clases y funciones más utilizadas. Estas funciones, a su vez llamar a otros archivos en esta carpeta, especialmente aquellos archivos JavaScript que utilizan Ajax y llaman a archivos PHP o archivos PHP que contienen llamadas a archivos HTML. Un ejemplo de esto último es la función que genera el menú, que tras generar las variables del menú, llama al archivo *menu.html* para su impresión.
- **Directorio *modules*:** Los archivos que contienen son responsables de gestionar los diferentes módulos y la instalación en la base de datos.
- **Directorio *resources*:** Aquí se localizan los archivos que se quieren compartir. También se localiza el esquema de un nuevo módulo.

- **Directorio *themes*:** En él se localizan los archivos y directorios para cambiar entre los diferentes temas de configuración. Aunque el cambio de temas no está implementado, si este se implementara, bastaría con interactuar con cada uno de los directorios aquí almacenados para cambiar la apariencia. Por el momento sólo aparece el directorio *classic* que incluye todas las imágenes y hojas de estilo en cascada necesarias para generar las diferentes apariencias.
- **Directorio *user*:** Este directorio contiene todos los scripts necesarios para gestionar a los usuarios y los privilegios de cada modelo de usuario.
- **Directorio *visualization*:** Aquí es donde se tiene que copiar un nuevo módulo como ya se comentará en la instalación del módulo desarrollado.

3.4. Interfaz de usuario

Se ha intentado, a la hora de desarrollar la interfaz, cumplir con dos objetivos; por un lado que la interfaz sea lo más amigable posible, es decir, que no se necesite de un extenso manual para poder manejar la aplicación, y por otro que sea muy atractiva para llamar la atención de los usuario.

Lo primero que el usuario se encuentra al entrar en la plataforma es con un identificador. Es básico que la aplicación requiera de autentificación para que el usuario pueda tener una experiencia personalizada cuando la utilice. Todo lo referente a permisos, perfiles, etc. aparece reflejado en la gestión de usuario.

Una vez autentificado el usuario, se encuentra con la interfaz básica de la herramienta. Ésta se divide en 3 partes. Por un lado la parte superior se encargará de todos aquellos aspectos corporativos, es decir, logo, título de la aplicación y cualquier cosa referente al corporativismo de la plataforma. En segundo lugar, aparece el menú con todas las herramientas de que dispone la aplicación, necesarias para su gestión, administración y control. Y por último, con la mayor cantidad de espacio disponible, una colección de *Widgets* como se puede ver en la Figura 18.

Figura 18: Interfaz de usuario; página principal.

El número de *Widgets* así como su distribución es altamente configurable, pudiendo aparecer tantos como el usuario desee y en el orden que desee; esto último aunque está hecho para que funcione correctamente no se ha añadido ninguna opción gráfica por el momento y se deja para futuras mejoras.

A las visualizaciones se puede entrar de dos formas. Por un lado pulsando en el menú correspondiente de las visualizaciones por lo que se irá a una visualización como puede ser el caso de la Figura 19 que carece de *Widget*, o pulsando el propio *Widget*, que redirige a la visualización con la configuración establecida en el resumen previo como se puede ver en la Figura 20. Como se puede ver, se trata de la visualización del W*idget* pero en él aparecerán todas las características y configuraciones de la vista, así como poder entrar en cualquier otra vista de la visualización. En esta figura se puede ver la utilización de un filtro temporal que ha sido diseñado para poder ser utilizado en todos los módulos si se requiere.

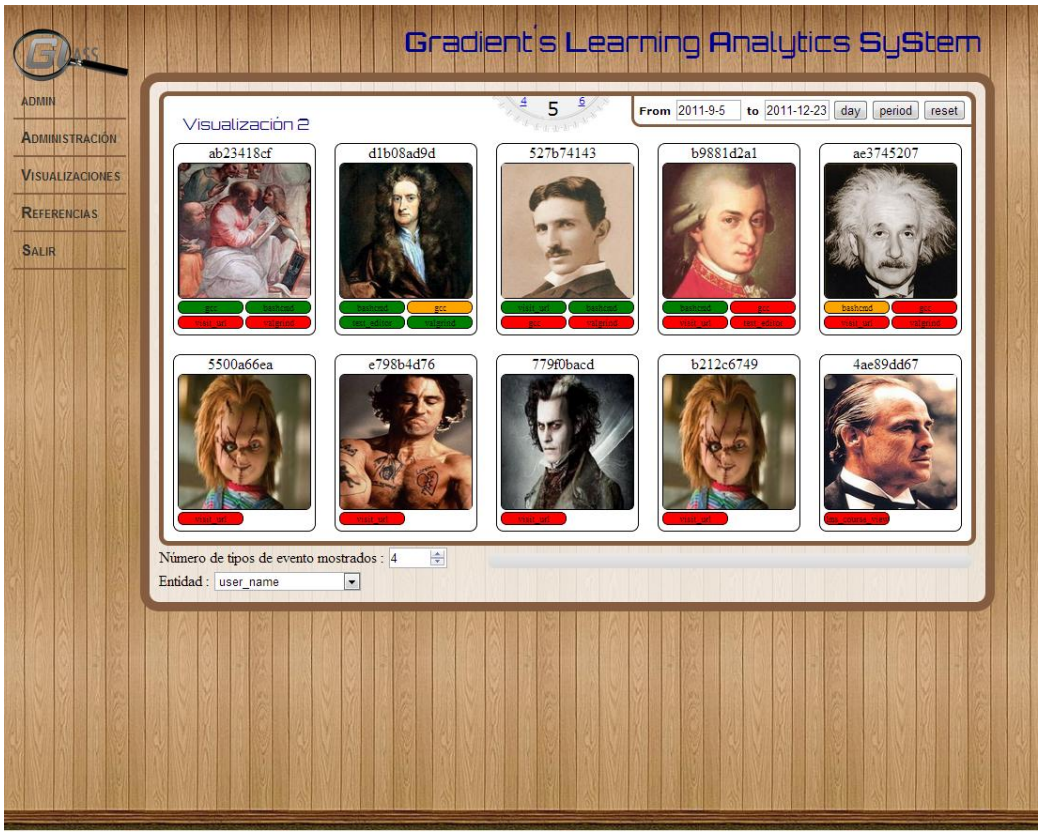


Figura 19: Interfaz de usuario; esquema de una visualización simple.

Llegado a este punto se puede ver claramente la diferencia entre una vista y una visualización. Dependiendo de la visualización y de la información que se quiera trasmitir, ésta puede tener 1 o más vistas; un claro ejemplo es la Figura 19, que solamente posee una vista, o la Figura 20, que llega a tener 5. En esta última figura se pueden ver claramente los botones que tiene disponibles para añadir un resumen al *Dashboard* como *Widget* o incluso al apartado de vistas favoritas. Quedan a elección del usuario los tipos de resúmenes que se puedan crear. Un ejemplo es la visualización 1 que permite agregar dos tipos de resumen, uno asociado a la vista 1 y otro asociado a la vista 2. La visualización 2 (Figura 19) en cambio, no permite agregar ningún resumen, aunque como futura línea se podría desarrollar uno y la visualización 3 (Figura 20), permite agregar un único resumen global, con los datos más característicos de todas las vistas.

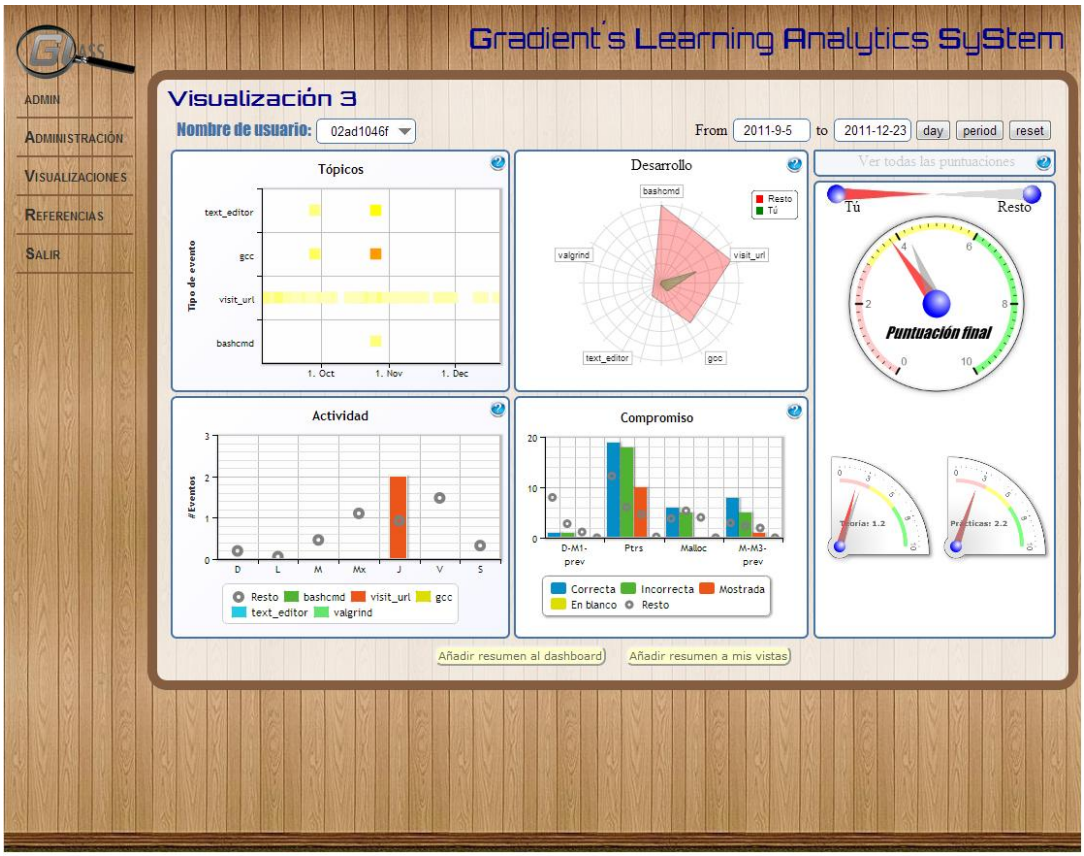


Figura 20: Interfaz de usuario, esquema de una visualización compleja.

Como se ha comentado, la plataforma va a ser altamente modular. Si queremos añadir nuevas visualizaciones (módulos de visualización), es necesario que haya una interfaz que se encargue de añadirlos o quitarlos. Una aproximación de interfaz es la que se ve reflejada en la Figura 21.

En la gestión de módulos pueden verse dos listas con los módulos instalados y con los módulos disponibles; al pasar de uno al otro se desinstalarán o se instalarán en el programa. Seleccionando un elemento de la lista se puede ver una descripción, y si se hace clic en uno, se puede ver una ventana debajo con una descripción más detallada, como se puede ver en la figura. Esta descripción es un texto HTML por lo que se pueden añadir imágenes o cualquier elemento que el diseñador vea oportuno.

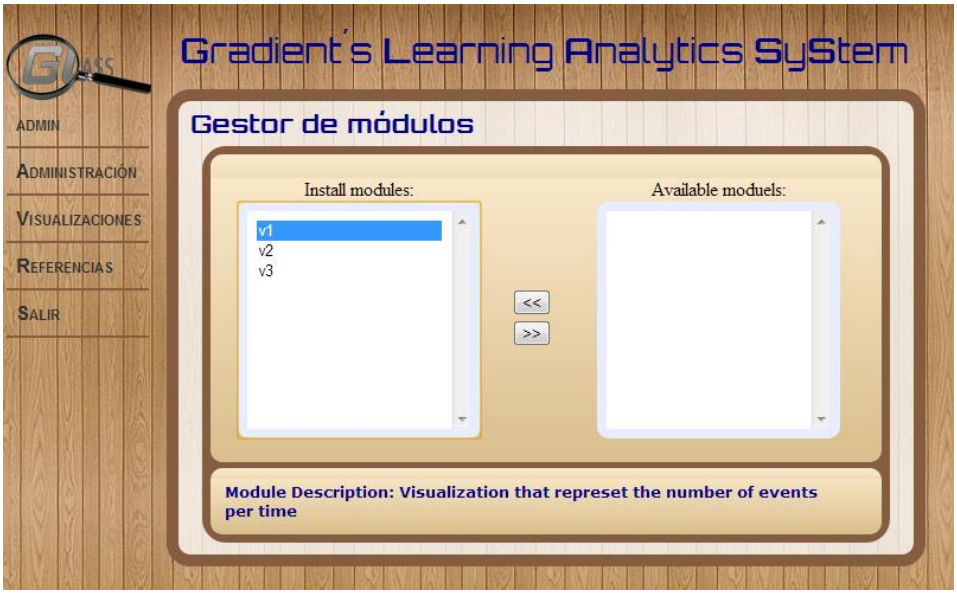


Figura 21: Interfaz de usuario; gestor de módulos.

Ya se ha hablado de que GLASS ofrece la posibilidad de guardar las vistas que el usuario guarde, como se puede ver en la Figura 22. Se trata de un cuadro donde aparece el nombre de la vista y unas opciones de gestión. Las opciones disponibles consisten en unas flechas para bajarlas y subirlas según aparecen en la lista, por si se quieren ordenar por importancia, opción de copiar, que permite seleccionar la URL al completo para importarla a otro navegador como cadena de texto, y por último una opción de borrado por si se quiere eliminar la vista de la lista de vistas favoritas. Aunque no aparece reflejado, también se podría añadir una opción para editar el nombre y el comentario puesto para la vista por si se quiere añadir algunas notas a posteriori. Si se pasa el botón por encima de un elemento de la lista se desplegará la visualización para poder así tener consciencia de lo que se ha ido guardando.

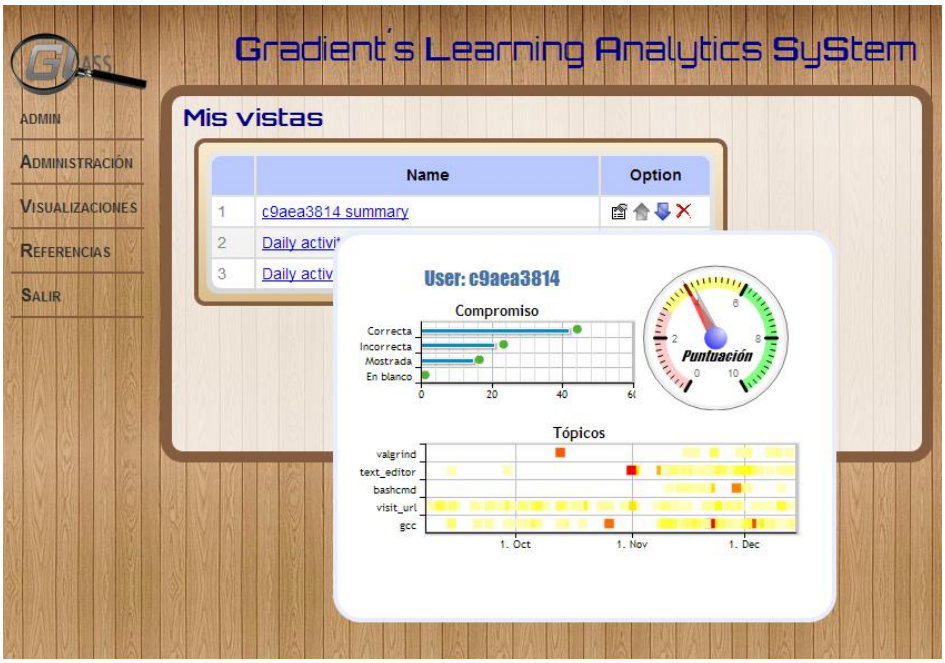


Figura 22: Interfaz de usuario; Mis vistas favoritas.

Con el fin de configurar la plataforma se ofrece al usuario la capacidad de modificar el número de columnas del *Dashboard*, añadir una nueva base a la base de datos o cambiar algunas variables de entorno global del servidor, como se puede ver en la Figura 23.

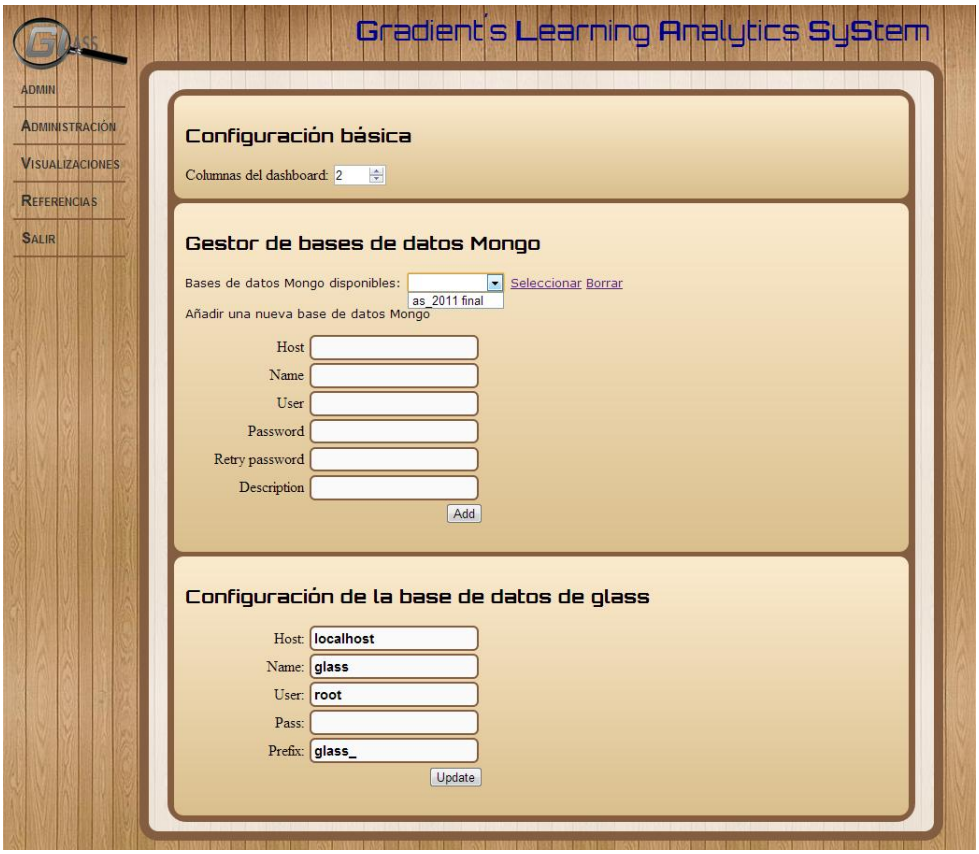


Figura 23: Interfaz de usuario; configuración de plataforma.

Del mismo modo, se dispone de una segunda ventana de configuración que permite gestionar los usuarios, como se puede ver en la Figura 24. En ella se observan dos ventanas. En la primera, un buscador con autocompletado para poder buscar un usuario en cuestión y así cambiarle su modelo de usuario y un número de botones de acceso rápido que muestran los usuarios del modelo de usuario seleccionado. En la segunda ventana se ven los diferentes permisos asociados a los 4 posibles roles antes comentados. El rol de administrador tiene todos los permisos y se prohíbe modificarlos por cuestiones de seguridad, ya que si por error se quitara el permiso de gestión de usuarios y es el único que puede otorgarlos, esta opción quedaría inhabilitada.

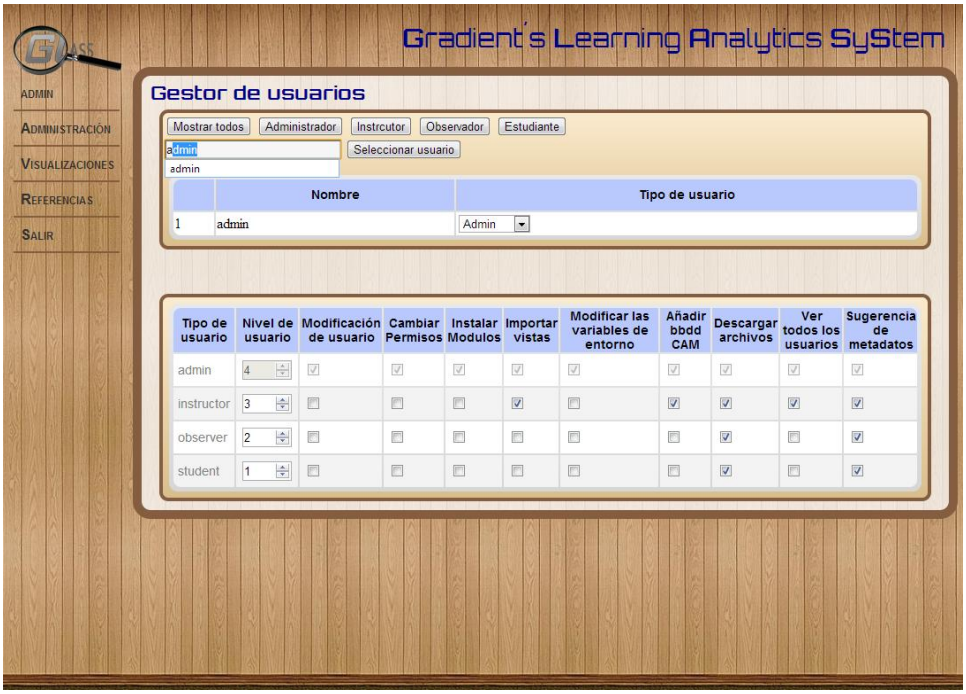


Figura 24: Interfaz de usuario; gestor de usuarios.

Capítulo 4

Visualización 3

4.1. Introducción

A lo largo de este apartado se va a entrar más en profundidad en explicar las características y funcionalidades del módulo en el que se centra este proyecto. El módulo recibe el nombre de visualización 3 ya que es el tercer módulo desarrollado para GLASS.

Su funcionalidad principal es mostrar el rendimiento de los alumnos tanto a ellos mismos como al profesor encargado de su supervisión. Se trata de un módulo que pretende ser lo más sencillo posible en cuanto a su utilización pero a la vez mostrar la mayor cantidad de información posible. Para conseguirlo se ha intentado que aparezcan el menor número de filtros posibles, por ello sólo se utiliza un filtro temporal y otro de selección de usuario. Debido a que se trata de un módulo orientado al usuario, no se muestra información de grupos y sólo la comparación con la media del resto de usuarios (resto de la población).

El módulo está compuesto por 5 vistas (*views*); la primera muestra diagrama de calor, enseñando la relación que existe entre el número de eventos realizados, el instante de tiempo en el que tuvieron lugar y el tipo de eventos del que se trata; una gráfica de radar, también conocida como un diagrama de araña, es una herramienta muy útil para mostrar visualmente los gaps entre el estado actual y el estado ideal; una primera gráfica de barras que muestra el número de eventos que han tenido lugar por instante de tiempo; y una segunda gráfica de barras que muestra los resultados de todas las pruebas realizadas a modo de test; y para finalizar, un conjunto de marcadores de las diferentes

puntuaciones del alumno junto con su nota media total y 2 notas medias parciales (práctica y teórica).

Se trata de un módulo que pretende ser, a simple vista, lo más sencillo posible pero con una gran complejidad en su interior como más tarde se explicará. El módulo mueve una gran cantidad de información por lo que se han tenido que definir algoritmos de búsqueda y procesado de la información bastante complejos para mostrar la información de una manera instantánea. Además, todas las actualizaciones se realizan en vivo por lo que el cambio de filtro no supone un tiempo de respuesta larga salvo el inicial de carga de datos del servidor, como ya se comentará.

Debido a que el módulo nace como una propuesta con un boceto definido, se empieza realizando una comparación entre los requisitos de diseños pedidos y los resultados finales obtenidos. Además se comentan los principales problemas surgidos y las soluciones que se han llevado a cabo.

A continuación se explica el módulo de una manera más exhaustiva. Para ello se van explicando cada una de las gráficas y los filtros que se pueden utilizar en ellas con mayor lujo de detalles. Además se habla de la relación de los modelos de usuario y los permisos que tienen cada uno de ellos, ya que dependiendo del modelo que utilice, el módulo restringirá o no cierta información o desactivará algún filtro.

Seguidamente, se realiza una descripción de los archivos del módulo y las funcionalidades que cada uno tiene, para continuar en el siguiente apartado describiendo la implementación de cada una de las partes. En este punto se hará una pequeña descripción de las 2 APIS utilizadas en el diseño.

Para finalizar, se añade un pequeño manual de usuario con las nociones básicas de instalación y uso, y se enumeran algunos de los posibles problemas que pudieran acontecer en su utilización.

4.2. Propuesta y resultados

A lo largo de este apartado se van a describir las principales motivaciones que surgieron para el desarrollo del proyecto y la comparación entre la propuesta presentada y el resultado final, describiendo los problemas que fueron surgiendo, las soluciones que se implementaron y las ventajas que éstas aportan.

Después del desarrollo de las 2 primeras visualizaciones creadas para GLASS, se llegó a la conclusión, sobre todo por la primera de ellas, que eran visualizaciones que ofrecían muchos datos y tenían muchos filtros, lo cual podía ser difícil de entender para usuarios no familiarizados con el entorno y podían llegar a provocar rechazo en aquellos usuarios que prefirieran algo que muestre mucha información pero con la mínima complejidad.

Por ese motivo, se propuso, tras ver otras herramientas que se asemejan a GLASS, el diseño que se puede apreciar en la Figura 25, cómo diseño de salida del módulo o visualización 3. El primer acuerdo que se tomó, que hace que este boceto difiera de la línea inicial seguida, es que, en vez de estar destinado a mostrar grupos, muestra usuarios, ya que es la entidad principal de análisis de la base de datos. De ahí que en vez mostrar lo que en la figura aparece referido como “Grupo 55”, se muestre el usuario a analizar. Y

para saber la situación de un individuo, es necesaria una referencia, de ahí que se añaden unas marcas que representen en términos medios al resto (*Rest*) de la población del mismo tipo, o lo que supone en el caso de ejemplo, la información media de todo el curso.

En este apartado no se va a entrar en detalle en el significado de cada una de las vistas, ya que más adelante se dedica un apartado entero para su descripción. Este apartado lo que pretende es demostrar la diferencia entre la propuesta inicial y el resultado final, que como se puede ver, guardan bastante semejanza.

Figura 25: Diseño original

Tras el desarrollo de la aplicación y tras varias revisiones, el resultado final del módulo es el que se puede apreciar en las capturas que aparecen representadas en la Figura 26 y Figura 29. Como se puede apreciar el diseño presenta algunas diferencias con respecto al boceto del diseño original. La primera, como se esperaba, es que aparecen los usuarios en vez del grupo. La siguiente diferencia es el filtro temporal, que en el boceto se representa simplemente como semana, mes y todo. Ahora tiene un aspecto más usable, utilizando el filtro temporal de otros módulos. El tercer cambio, es que la gráfica de radar no posee el indicador que aparece en el boceto a la izquierda, esto es debido a que se llegó a la conclusión de que mostraba información redundante. Por último, los indicadores de la derecha se han sustituido por varios indicadores divididos en indicadores de resultados finales e indicadores de resultados parciales.

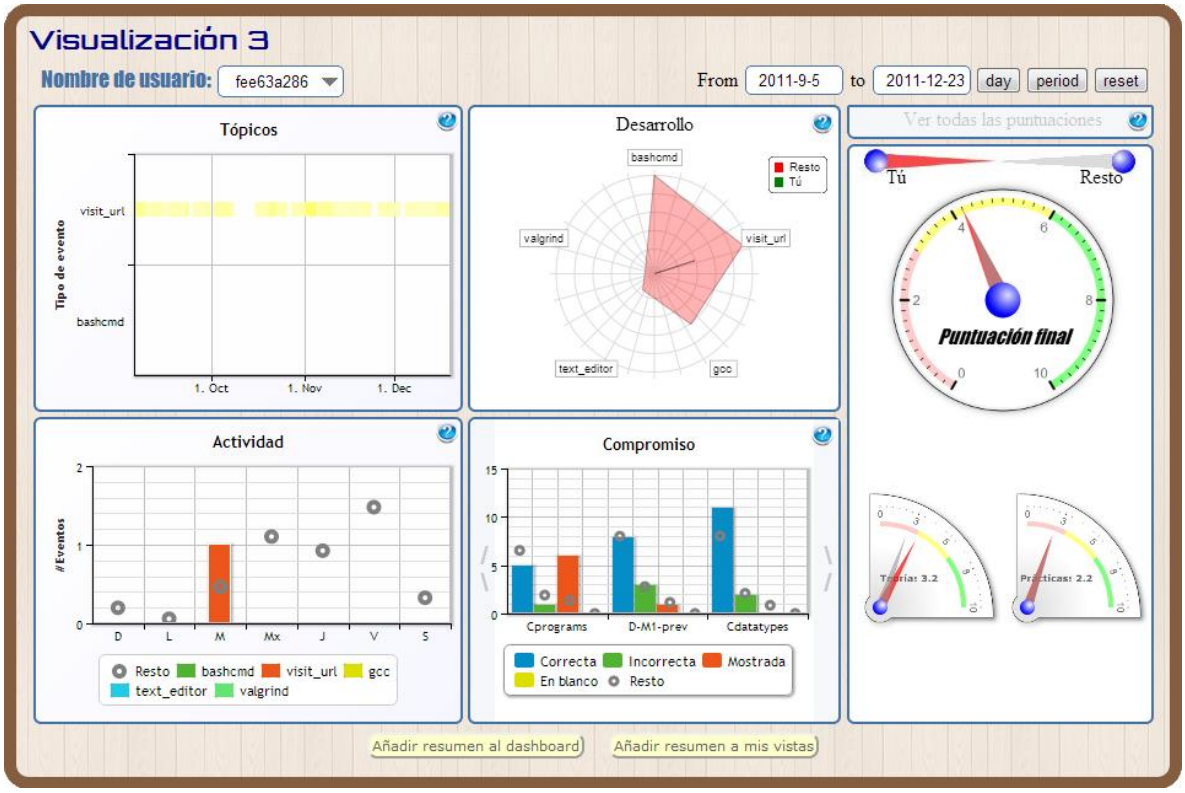


Figura 26: Diseño final con notas medias y notas parciales ocultas

Debido a que GLASS nos da la posibilidad de insertar un resumen en el tablón inicial de la aplicación, se planteó también el boceto de resumen que aparece en la Figura 27. En él, cono se puede ver, aparece un resume de cada una de las vistas de la visualización.

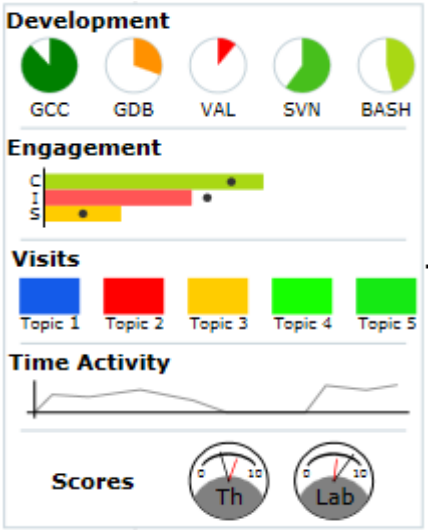


Figura 27: Boceto del Widget

Aunque en un principio se obtuvo algo similar, se llegó a la conclusión de que para un resumen eran demasiados datos y lo sobrecargaba demasiado. De este modo, como se puede ver en la Figura 28, se ha recogido sólo una gráfica similar a la que aparece en la visualización, el diagrama de calor nombrado como tópicos, ya que es la que más información posee y es sencilla de ver. Las otras dos son, la puntuación final, que recoge la nota final tanto del usuario (en rojo) como la del curso (en gris), y un resumen del

diagrama de compromiso que reúne todos los resultados de los test en una única gráfica, junto con la media de todos los usuarios.

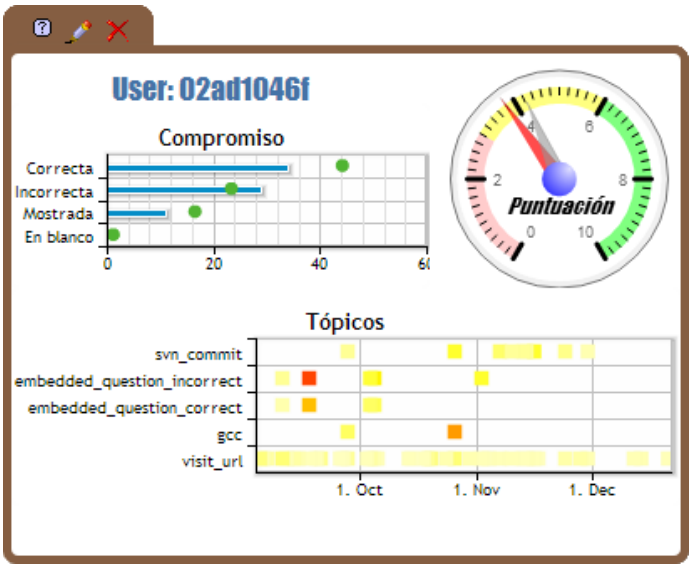


Figura 28: Widget

A lo largo del desarrollo surgieron una serie de problemas que desembocaron en algunos de los cambios ya descritos. A continuación se enumeran los problemas detectados y se describen las soluciones:

- **Compromiso:** La gráfica del compromiso, presentaba un problema relacionado con el número de test que pudieran aparecer. Cuando éste era muy alto, la gráfica se recargaba con demasiados datos por lo que se llegó a un acuerdo de diseño: el número máximo de test que debe aparecer en la gráfica es de 4 y, si hay más de 4, se reparten en gráficas sucesivas, de ahí las flechas que aparecen a los lados de la gráfica que originan que vayan apareciendo más gráficas.
- **Marcadores:** El problema de los marcadores es similar al problema presentado en la gráfica del compromiso. El número de notas llegaba a ser demasiado alto y no quedaba claro cuál era el dato más importante. La solución es que siempre se muestre la puntuación final (que es el dato más importante), y que se alterne entre las notas medias de datos y teoría y las notas de cada una de las evaluaciones. El resultado es la diferencia que puede haber en los indicadores de la derecha de las capturas que aparecen en la Figura 26 y la Figura 29. Además, como las notas pueden ser múltiples se parte de la misma solución del problema de la gráfica del compromiso.

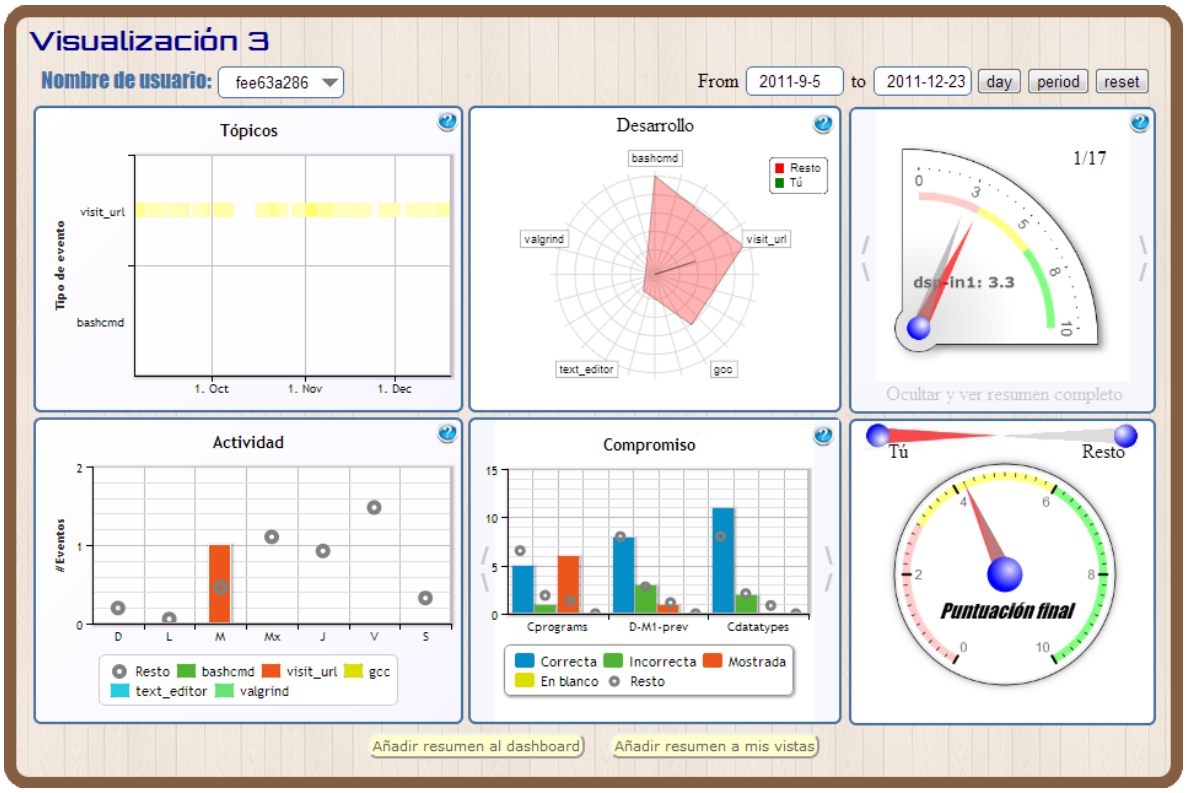


Figura 29: Diseño final con notas parciales desplegadas

- **Actividad:** Dentro de los problemas que aparecen, es el más liviano de todos. Simplemente se mostraba el problema de que, si esta gráfica muestra la actividad temporal de la última semana sería mejor mostrar la de los últimos 7 días debido a que cuando se llega a final de semana, la información estaría más desactualizada que al principio.
- **Tiempo de adquisición de datos:** Aunque sólo posee dos tipos de filtros y no es de los módulos que más recarga de datos necesite, sigue siendo un problema si cada vez que se cambie un filtro tenga que, primero obtener los datos de la base de datos y luego realizar los cálculos. Para solucionarlo, en la carga inicial del módulo se cargan todos los datos de la base de datos y lo único que se realiza en tiempo real son los cálculos, por lo que el usuario sólo percibe un pequeño retraso al entrar a la visualización por primera vez, que no es mucho mayor que si obtuviera sólo los datos particulares, y cualquier cambio que se realice en los filtros conlleva un cálculo en tiempo real y que ocasiona un movimiento lo que origina que el diseño parezca más atractivo.
- **Movimiento:** Debido al gran número de gráficas se pude ver, que si en las recargas a todas se les aplicara movimiento, en algunos equipos podría llegar a producir movimientos bruscos con bloqueos de tiempos pequeños, por lo que se han eliminado los movimientos de las gráficas menos significativas y más numerosas, es decir, en las puntuaciones.

4.3. Descripción del módulo

A lo largo de este apartado se describe cada una de las partes del módulo en más detalle. Se empieza haciendo una descripción de cada una de las vistas del módulo, se sigue con la descripción del Widget o resumen y para finalizar se describen los filtros utilizados y cómo influyen los modelos de usuarios en cada uno de ellos.

4.3.1. Vistas

El módulo está formado por 5 vistas como ya se ha comentado, a continuación se describe cada una de ellas:

4.3.1.1. Tópicos (diagrama de calor)

El diagrama de calor (Figura 30) muestra la relación que existe entre el número de eventos realizados, el instante de tiempo en el que tuvieron lugar y el tipo de eventos del que se trata. Es un diagrama que muestra 3 dimensiones de la siguiente manera:

- **Eje X:** eje temporal; instante de tiempo donde se produce el evento.
- **Eje Y:** tipo de evento; donde muestra aquellos 5 que más veces han tenido lugar.
- **Calor:** representación por medio de colores, a modo de tanto por ciento, del número de veces que se ha dado ese evento en un instante de tiempo. Se representa por una escala de colores que sigue la secuencia blanco-amarillo-rojo en el cual, blanco representa que ese evento no ha tenido lugar y rojo que es el referente en cuanto a número de apariciones. Se trata de un valor ponderado ya que se toma como referencia el número de eventos mayor que un usuario ha generado.

El diagrama depende del eje temporal en el que se ha ejecutado. Por defecto se muestran todos los días que un usuario ha producido eventos pudiéndose acotar con el filtro temporal. En el caso de no existir ningún usuario o no tener permisos para seleccionar uno, se mostrarán los datos de toda la población.

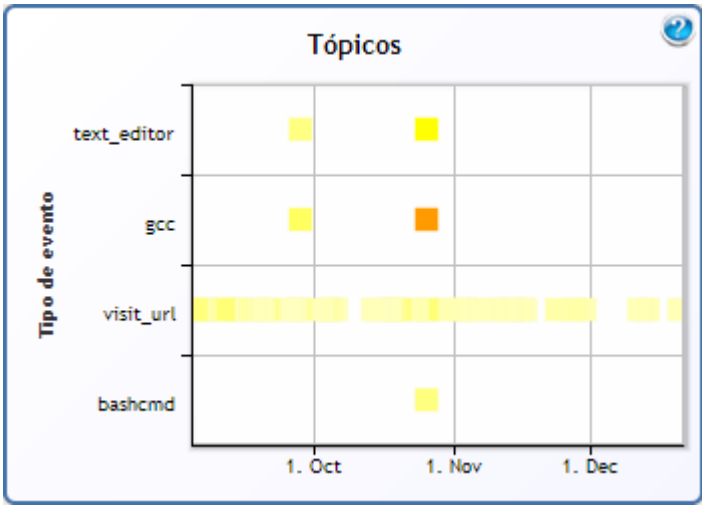


Figura 30: Tópicos (diagrama de calor)

4.3.1.2. Actividad temporal (diagrama de barras)

Esta gráfica (Figura 31) muestra el número de eventos que han tenido lugar por instante de tiempo. Además, para ampliar la información, muestra un diagrama de colores para especificar los diferentes eventos divididos por el tipo de evento especificado. El eje temporal está acotado a los 7 últimos días en relación con la cota superior del diagrama temporal mostrando los 5 eventos que más veces tienen lugar y que vienen nombrados en la leyenda siguiendo un código de colores para relacionarlos con el diagrama. Se representa, mediante un círculo, la media aritmética de todos los usuarios de esa población para poder establecer una referencia y en la leyenda viene representada como "Resto". En el caso de no existir ningún usuario o no tener permisos para seleccionar uno, sólo se muestra la media aritmética de la población.

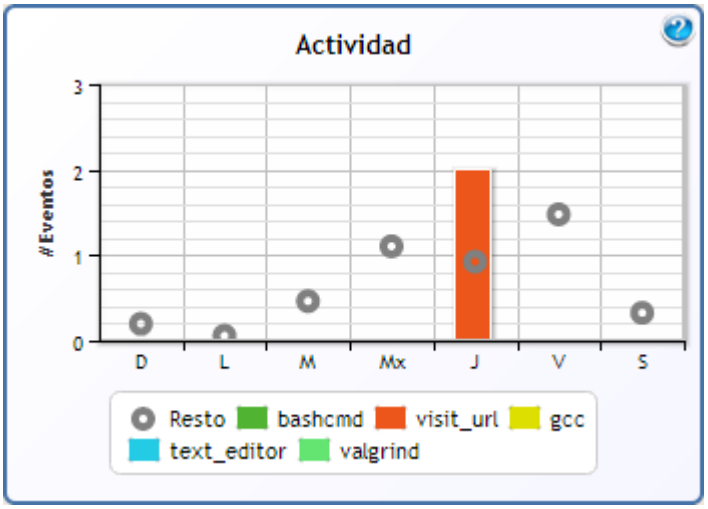


Figura 31: Actividad temporal (diagrama de barras)

4.3.1.3. Desarrollo (diagrama de radar)

Una gráfica de radar (Figura 32), también conocida como un diagrama de araña, es una herramienta muy útil para mostrar visualmente los gaps entre el estado actual y el estado ideal. Concretamente esta gráfica muestra el número de veces que un usuario ha generado un tipo de evento y se relaciona con la media aritmética de toda la población. Para la representación de la gráfica se utilizan los 5 tipos de eventos que el usuario más ha generado. Por defecto aparecen todos los eventos del usuario independientemente del tiempo donde han tenido lugar, pero gracias al filtro temporal se puede acotar el número de eventos que intervienen. En la leyenda se refiere a los datos del diagrama utilizando un código de colores para separar los datos del propio usuario (*tú*) y la media de los demás (*resto*). La parte de área que ambas entidades comparten aparece representada como la mezcla de ambos colores. En el caso de no existir ningún usuario o no tener permisos para seleccionar uno, sólo se muestra las medias aritméticas de la población.

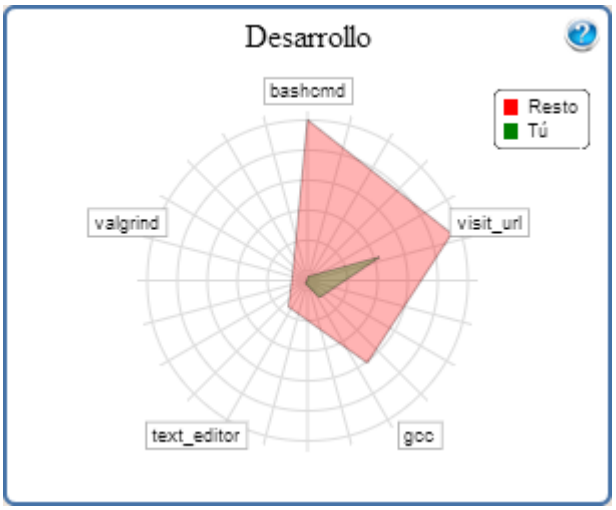


Figura 32: Desarrollo (diagram de radar)

4.3.1.4. Compromiso, preguntas insertadas (diagrama de barras)

Esta visualización (Figura 33) muestra los resultados de todas las pruebas realizadas a modo de test. Cada grupo, de 4 barras, representa la prueba realizada y cada barra representa el número de preguntas que se ha respondido correctamente, incorrectamente, que se ha dejado en blanco o que se ha mostrado la solución. En la leyenda aparece un código de colores que nombra y relaciona los 4 tipos anteriores. En el eje X, que se puede ver nombrado el grupo al que pertenece el test, y en el eje de las Y, la escala con el número de preguntas de cada tipo. Con un círculo, "resto" en la leyenda, se puede ver la media aritmética generada por toda la población. Cada gráfica muestra un número máximo de 4 test, por lo que si el usuario ha participado en más de 4 test, aparecen flechas a los lados de la visualización que se activan al pasar el ratón por encima, así se permite ver en una nueva gráfica el resto de test en los que ha participado. En el caso de no existir ningún usuario o no tener permisos para seleccionar uno, sólo se muestra la media aritmética de la población.

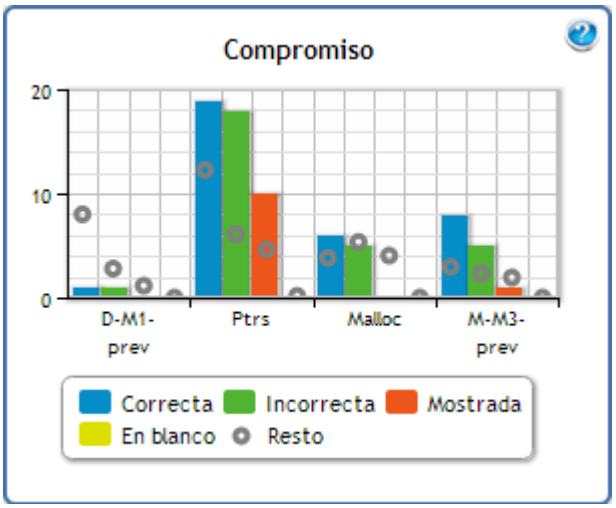


Figura 33: Compromiso (diagrama de barras)

4.3.1.5. Puntuaciones (marcadores)

En esta visualización (Figura 34) se puede ver de dos formas diferentes. Por defecto (vista de la izquierda) aparece la nota de la asignatura (nota final) y debajo de ésta, los resultados totales de prácticas y teoría. Si se quiere ver una vista más avanzada (vista de la derecha), pulsando en la caja superior se puede desplegar un nuevo cuadro donde ver cada uno de los resultados de las diferentes pruebas realizadas, lo que hará desaparecer los resultados totales de práctica y teoría. Se puede volver en cualquier momento de una forma de visualización a otra. En cada gráfico se pueden apreciar 2 flechas. En rojo, es aquella que indica los resultados del usuario (*tú*) y en gris, lo que representa la media aritmética de la población de cada prueba. En el caso de que no exista un usuario o no tenga permisos para seleccionar uno, sólo se muestra las medias aritméticas de la población.

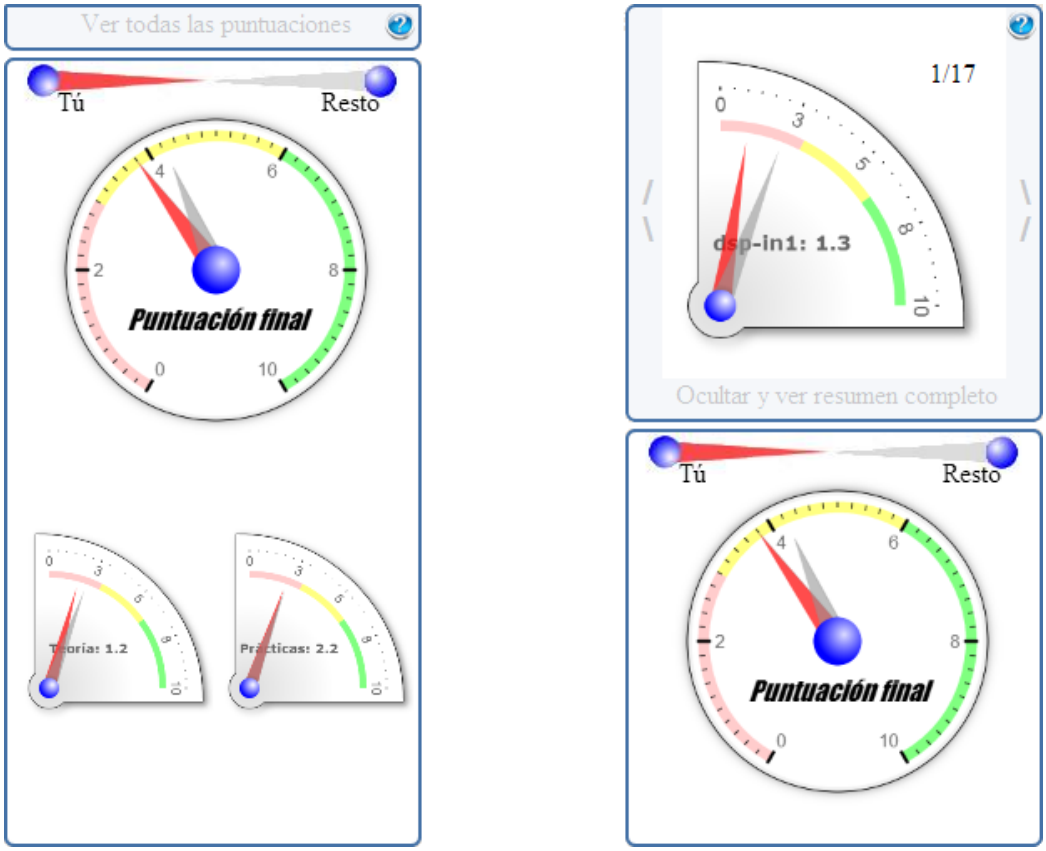


Figura 34: Puntuaciones (marcadores)

4.3.2. Widget

En la Figura 28 se ha podido ver como es el resultado final del Widget o resumen del módulo. En él se pueden ver 4 partes perfectamente diferenciadas:

- **Nombre de usuario:** Muestra el usuario al que pertenece el resumen.
- **Compromiso:** Se muestra lo mismo que en la vista del compromiso de la visualización pero difiere en que en vez de mostrar un número de gráficas según el test al que pertenecen, muestra una única gráfica con la suma de cada uno de los

tipos de resultado posibles. Esto sucede tanto para el individuo como para la población.

- **Puntuación:** Es la nota final. Se muestra este indicador ya que, como se ha comentado, es el indicador más importante.
- **Tópicos:** Esta es la única gráfica que coincide exactamente con la que aparece en la visualización, pero siempre muestra el periodo completo y no se aplica el filtro temporal.

4.3.3. Filtros y botones

En la visualización se pueden apreciar 2 filtros perfectamente diferenciados y 2 botones para interactuar con las vistas y el Dashboard de GLASS. A continuación se describen con más detalle cada una de ellos:

4.3.3.1. Filtro de Usuario

El filtro de usuario (Figura 35), permite la selección de los diferentes usuarios. Este filtro no es visible para todos los modelos de usuario debido a temas de confidencialidad por lo que sólo estará disponible para el administrador del sistema y para los profesores. Para el modelo de usuario de estudiante se selecciona automáticamente al usuario si es que existe en la base de datos; si no se muestra la información como si fuera un observador. Para el modelo de usuario de los observadores sólo se muestran los datos de medias generales y no hay ninguna información de alumnos.

Figura 35: Filtro de usuarios

4.3.3.2. Filtro temporal

El filtro temporal (Figura 36) permite cambiar el periodo de las gráficas de radar y la de tópicos. Para la gráfica de actividad temporal muestra la información de los últimos 7 días del periodo seleccionado. Este filtro posee 3 botones para la selección del tiempo de visualización:

- **Día (*day*):** Despliega un calendario que permite elegir el día a analizar.
- **Periodo (*period*):** Igual que para el día pero permite elegir 2 días para seleccionar el periodo comprendido entre ellos.
- **Reseteo (*reset*):** Resetea todos los datos y establece el periodo inicial, es decir, el inicio y fin de todos los datos guardados en la base de datos.

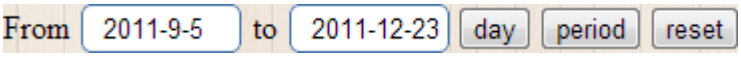


Figura 36: Filtro temporal

4.3.3.3. Botones

Para interactuar con las funcionalidades de GLASS, la visualización incluye 2 botones (Figura 37) situadas en la parte inferior central de la visualización.

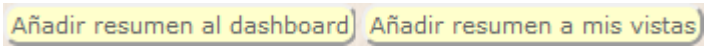


Figura 37: Botones para interactuar con GLASS

El primero de ellos (izquierda) permite agregar un resumen al tablón de inicio de GLASS. Debido a que el resumen es inherente al usuario solamente permite agregar un resumen por usuario debido a que agregar más sería tener resúmenes duplicados. Al apretar el botón se muestra información relativa a lo sucedido.

El segundo de ellos (derecha) tiene el mismo funcionamiento que el anterior, pero en vez de agregar la gráfica al tablón, lo agrega al apartado de vistas favoritas. En cuanto a las duplicidades ocurre lo mismo que en el Widget del Dashboard.

4.4. Archivos

El módulo está constituido por un conjunto de archivos los cuales tienen una funcionalidad única. A continuación se describe cada uno de ellos:

4.4.1. Archivo index.php

Contiene todo el código necesario PHP que genera la visualización. Se incluye la librería *mainlib.php* la cual incluye todas las clases y funciones de GLASS en PHP y la librería *lang.php* que gestiona el idioma de GLASS y por lo tanto del módulo. También se incluye la librería *jslib.js* la cual incluye las funciones JavaScript de GLASS.

Además también gestiona que la sesión esté activa y si no redirige al login pasando la página actual para volver directamente tras el logueo. Contiene los tags que construyen el diseño exterior al módulo (menú, cabecera y espacio para el cuerpo) y que define GLASS.

Enlaza al archivo *index.html* que se encarga de todo el cuerpo del módulo.

4.4.2. Archivo index.html

Contiene todo el código necesario HTML y JavaScript del módulo. El código se divide en 3 secciones. En la primera se incluyen todos los enlaces. Entre ellos aparecen los enlaces al código de las 2 APIs utilizadas para la generación de las gráficas, a otros archivos JavaScript que almacenan el código de otras funciones necesarias por el código que aquí se genera, y a los archivos encargados de las hojas de estilo en cascada.

En la segunda se incluye el código de las funciones JavaScript. Estas funciones, que serán descritas en el apartado de implementación, son las siguientes:

- *get_data()*
- *generate_views(all_data,score_data,eq_data,temporalFilter,entity_name,datasetId,dictionary)*
- *v3_show_help(index)*

Y en la tercera, se incluyen los tags utilizados tanto por el código JavaScript como las hojas de estilo en cascada para generar el diseño gráfico del módulo.

4.4.3. Archivo info.txt

Añade la información necesaria del módulo la cual se muestra a la hora de ser instalado. Admite tags HTML por lo que se pueden añadir imágenes, tipos de letra, etc.

4.4.4. Archivo share.php

Guarda el código necesario para compartir el Widget o resumen que se ofrece en el apartado de mis vistas favoritas de GLASS. Incluye las librerías de configuración, lenguaje y librería principal de GLASS, así como la librería v3lib.php que contiene la función que captura los datos necesarios de la base de datos para generar los datos del Widget. También contiene los tags que sirven como contenedor para la impresión de los elementos del Widget.

4.4.5. Archivo widget.html

Contiene todo el código HTML y JavaScript necesario para imprimir el Widget.

Está compuesto por 4 funciones que gestionan las 3 gráficas que se muestran el en Widget:

- *V3_print_graphs(data,numwidget)*
- *V3_scatter(data,numwidget,dictionary)*
- *V3_engagement(eng,numwidget,dictionary)*
- *V3_scores(score,numwidget,dictionary)*

Contiene los enlaces a los APIs y a las hojas de estilo en cascada.

4.4.6. Archivo getdata.php

Contiene el código PHP que realiza la consulta a la base de datos para obtener los datos necesarios para la visualización. Incluye el código de *v3lib.php* y *jslib.js* librerías principales de GLASS y *lang.php* para la gestión del idioma.

4.4.7. Archivo v3lib.php

Está compuesto por dos funciones; la primera recupera los datos de la base de datos para la visualización y la segunda recupera los datos para el Widget. Las funciones son:

- *get_visualization3_Json_data($CFG,$userid,$username,$confiId,$fconfId,$user_level,$dictionary)*

- *get_visualization3_data_widget($CFG,$conf,$userid,$datasetId,$dictionary)*

4.4.8. Archivo widget.php

Contiene todo el código PHP necesario para imprimir el resumen del módulo.

Comprueba que hay una sesión iniciada, captura la información necesaria de la base de datos, establece el *encoding* para las cadenas de texto y llama a *widget.html* para que imprima el Widget. En él se incluyen los tags necesarios para imprimir el Widget, es decir, los contenedores. Incluye la librería *v3lib.php* que contiene la función que captura los datos necesarios de la base de datos para generar los datos del Widget.

4.4.9. Directorio lang

Incluyen los archivos que definen las cadenas de caracteres que se quieran mostrar en el módulo. Cada archivo que aparece dentro del directorio corresponde a su idioma.

4.4.10. Directorio css

Incluye los archivo necesarios que definen el diseño gráfico del módulo. Está compuesto por un conjunto de imágenes y el archivo *style.css* el cual contiene la hoja de estilo en cascada del módulo.

4.4.11. Directorio js

Incluye los archivos JavaScript necesarios para el funcionamiento del módulo. Esta carpeta está formada por:

- La carpeta *modules, themes* y *RGraph* y los archivos *highcharts.js* y *highcstock.js:* contienen el código de las APIs utilizadas.
- **El archivo *maindatagen.js*:** Ordena los datos y les da un formato operativo para JavaScript y adecuado para la visualización. Está formada únicamente por la función *sort_data(all_data,temporalFilter,entity).*
- **El archivo *column.js*:** Genera la gráfica de la actividad temporal. Está formada por las siguientes funciones:
    - *get_column_chart_data(data,EventTypeToUse,dictionary)*
    - *print_column_chart(data,EventTypeToUse,dictionary)*
- **El archivo *columnpool.js*:** Genera la gráfica del compromiso. Está formada por las siguientes funciones:
    - *get_column_pool_chart_data(eqdata,entity_name,dictionary)*
    - *print_column_pool_chart(eq_data,entity_name,dictionary)*
    - *generate_column_pool_chart(graph_data,divcontainer,dictionary)*

- **El archivo *indicators.js*:** Genera los diferentes indicadores. El archivo contiene las siguientes funciones:
    - *get_metadata_and_print_indicator(all_metadata,ENTITY,dictionary)*
    - *print_indicartors(metadata,ENTITY,PREFIX,dictionary)*
    - *print_indicator_graph(index,divtoplace,values,grap_title,grapthtype)*
- **El archivo *radar.js*:** Genera la gráfica de radar. El archivo contiene las siguientes funciones:
    - *prepare_data_for_radar_chart(data,MAX_PRINTABLE_VALUES)*
    - *print_radar_chart(data,MAX_PRINTABLE_VALUES,dictionary)*
- **El archivo *scatter.js*:** Genera la gráfica de diagrama de calor. El archivo contiene las siguientes funciones:
    - *get_scatter_chart_data(data_and_biggest,eventTypeToUse)*
    - *print_scatter_chart(data,eventTypeToUse,dictionary)*

4.5. Implementación

Ya se han enumerado los diferentes archivos y las funciones que almacenan cada uno, ahora es el momento de entrar un poco más en detalle y explicar la implementación de cada una de las funciones, así como de los procesos de los scripts.

4.5.1. Generación de datos

El archivo *v2lib.php* está compuesto por 2 funciones que se encargan de obtener los datos de la base de datos Mongo. La primera función es la encargada de obtener los datos para la visualización y la segunda para el Widget que se añade al tablón (Dashboard), o a mis vistas favoritas.

4.5.2. Generar datos de la visualización

Nombre de la función

get_visualization3_Json_data

Entrada

$CFG: objeto que guarda la configuración de GLASS.

$userid: identificador del usuario.

$username: nombre del usuario.

$confiId: id de la configuración para el Widget.

$fconfId: id de la configuración para mis vistas favoritas.

$user_level: nivel de usuario según los permisos de GLASS.

$dictionary: array que guarda las cadenas de caracteres que se definen en el idioma.

Salida

Datos en formato JSON.

Descripción

Captura los datos necesarios para la visualización de la base de datos. Necesita de conexión a la base de datos de GLASS para obtener configuraciones, chequear usuario y gestionar permisos. Utiliza el formato JSON para poder pasar los datos al script principal. El flujo de funcionamiento es el que se puede ver en la Figura 38.

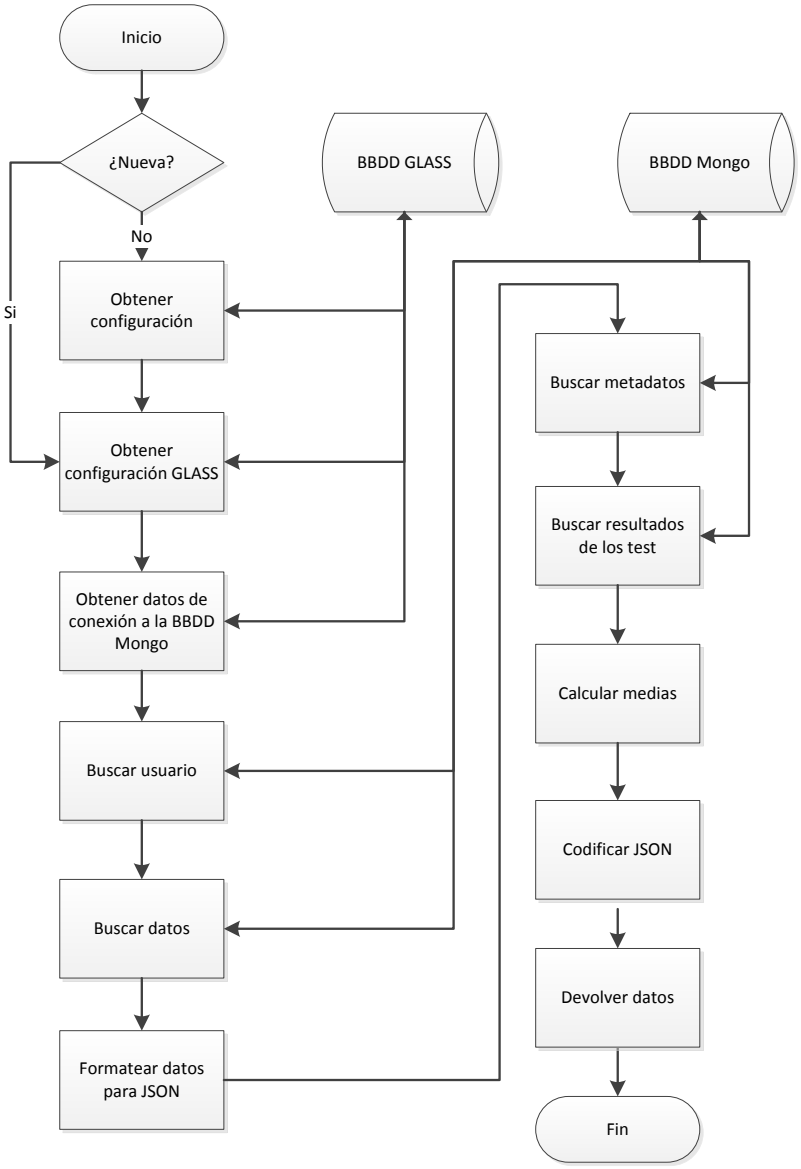


Figura 38: Diagrama flujo de la función get_visualization3_Json_data

4.5.3. Generar datos del Widget

Nombre de la función

get_visualization3_data_widget

Entrada

$CFG: objeto que guarda la configuración de GLASS.

$conf: configuración del Widget. Cadena de caracteres con los datos en JSON.

$userid: identificador del usuario.

$username: nombre del usuario.

$datasetId: identificador en la base de datos GLASS de la base de datos Mongo.

$dictionary: array que guarda las cadenas de caracteres que se definen en el idioma.

Salida

Datos en formato JSON.

Descripción

Captura los datos necesarios para generar el Widget. Necesita de conexión a la base de datos de GLASS para obtener configuraciones, chequear usuario y gestionar permisos. Utiliza el formato JSON para poder pasar los datos al script principal. El flujo de funcionamiento es el que se puede ver en la Figura 39.

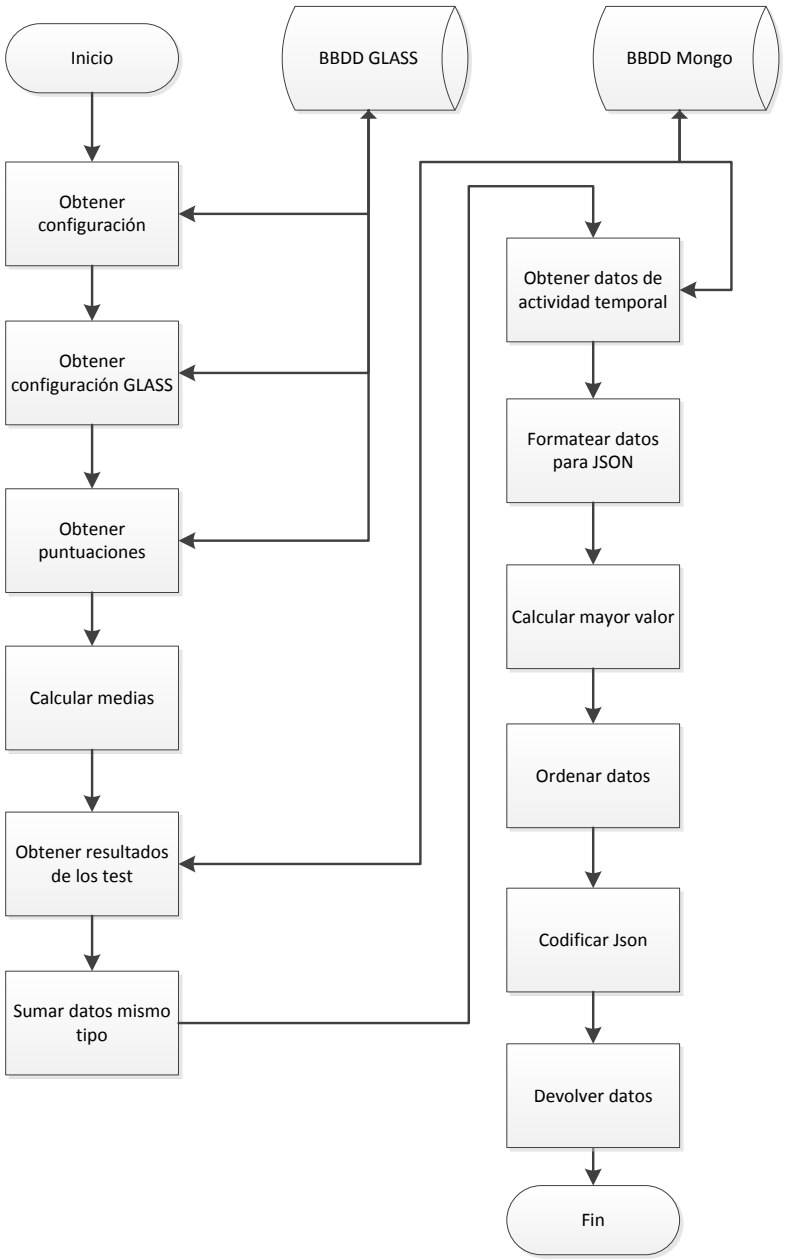


Figura 39: Diagrama de flujo de la función get_visualization3_data_widget

4.5.4. Impresión de la visualización

A lo largo de esta sección se van a describir las funciones y los procesos encargados de imprimir y actualizar las diferentes vistas del módulo. Principalmente está constituido por código HTML (para contenedores) y JavaScript salvo la adquisición de datos de la que ya se ha hablado. El archivo principal es *index.html* y la función *get_data().* A partir de ésta se van haciendo las llamadas a las diferentes funciones que se encuentran en este mismo archivo y en *maindatagen.js*, *column.js, columnpool.js, indicators.js m, radar.js* y *scatter.js*. Debido a que se trata de código JavaScript, todos los datos son precalculados sin necesidad de una recarga de la página, buena práctica para reducir la carga del servidor.

A continuación se describen el conjunto de procesos involucrados en las siguientes subsecciones.

4.5.4.1. Obtener datos

Nombre de la función

get_data

Entrada

Ninguna.

Salida

Ninguna.

Descripción

Función que mediante la tecnología AJAX configurado en modo asíncrono y pasando gracias al método POST una serie de variables, ejecuta el proceso de captura de datos de la base de datos contenido en *getdata.php*. El formato de intercambio de datos utilizado es la estructura JSON como ya se ha comentado. Mediante el uso de eventos inicia de nuevo el cálculo de los datos. El flujo de funcionamiento es el que aparece reflejado en la Figura 40. La función hace un preordenamiento y preformateo de los datos para que pueda ser procesado de una manera más sencilla por el código JavaScript y establece alguna de las opciones esenciales para el control de la visualización.

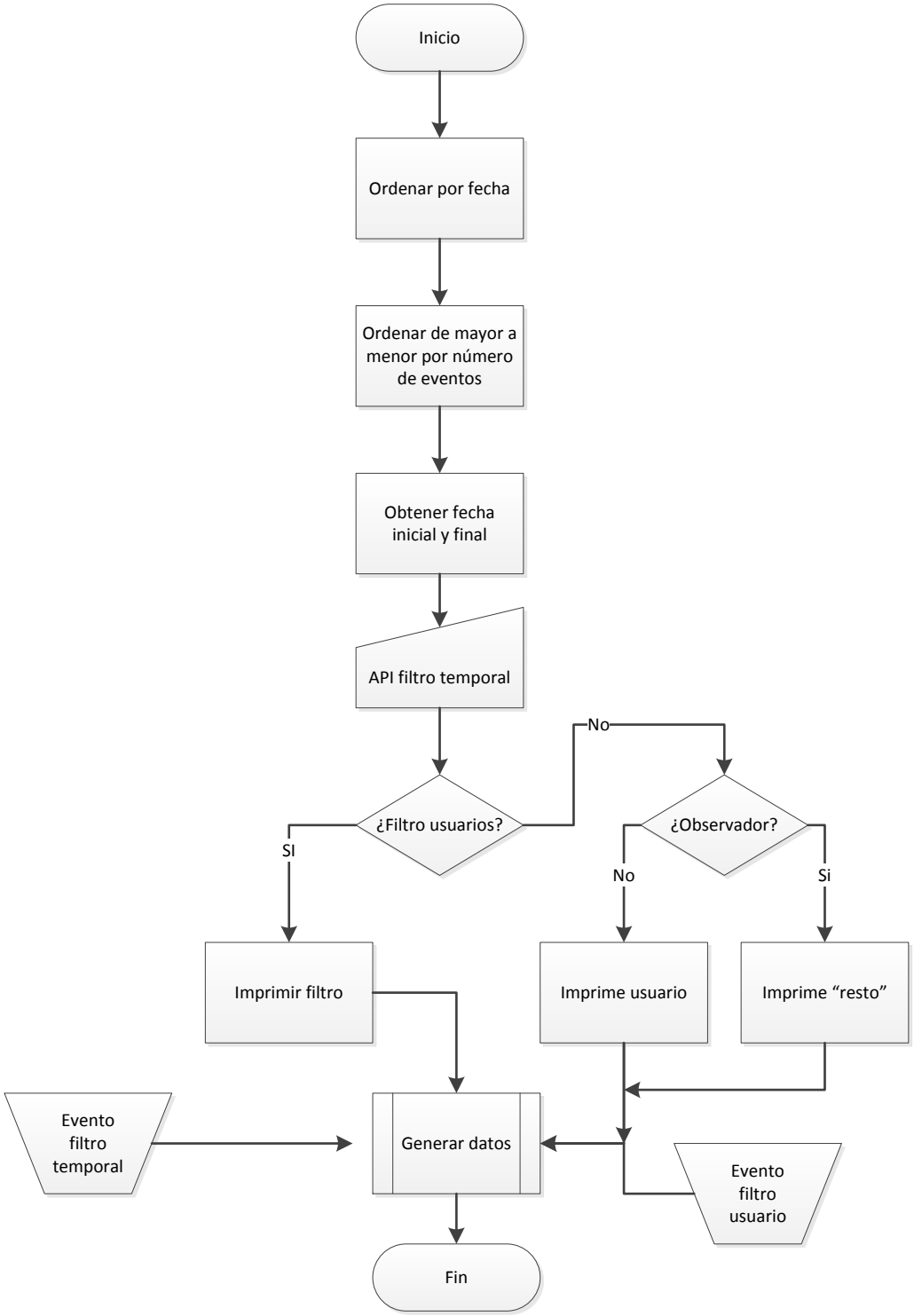


Figura 40: Diagrama de flujo de la función get_data

4.5.4.2. Imprimir vistas

Nombre de la función

generate_views

Entrada

all_data: datos necesarios para la impresión de la vista de la actividad temporal, el diagrama de calor y el diagrama de radar.

score_data: datos necesarios para la impresión de los indicadores. Estos datos se definen en la base de datos mongo como *scores* y son propios del usuario.

eq_data: datos necesarios para la impresión del diagrama de compromiso capturado. Estos datos se definen en la base de datos mongo como *embedded_question* y son propios del usuario.

temporalFilter: objeto que almacena la información relativa al filtro temporal.

entity_name: entidad de usuario seleccionada o detectada.

datasetId: identificador GLASS de la base de datos Mongo para esta visualización.

dictionary: array que guarda las cadenas de caracteres que se definen en el idioma.

Salida

Ninguna.

Descripción

Esta función se encarga de hacer las llamadas a las diferentes funciones que imprimen cada una de las vistas y recoger de estas los datos que son necesarios para la función siguiente. El flujo de la función es el que se puede ver reflejado en la Figura 41.

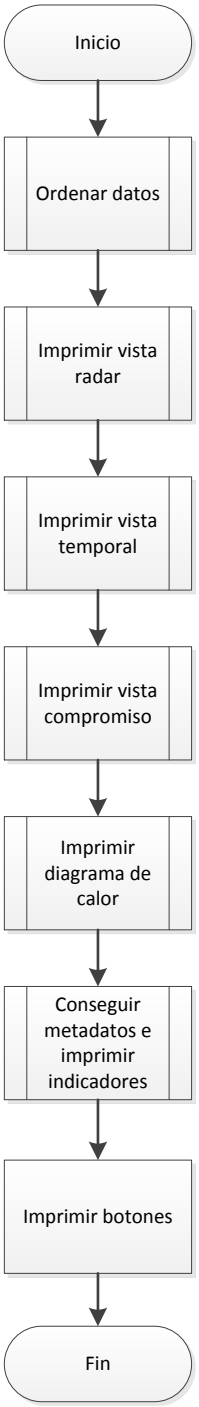


Figura 41: Diagrama de flujo de la función generate_views

4.5.4.3. Imprimir ayuda

Nombre de la función

v3_show_help

Entrada

index: gestiona si se muestra o no la ayuda.

Salida

Ninguna.

Descripción

Imprime la ayuda de cada una de las vistas y gestiona si debe ser mostrada o no por pantalla. El diagrama de flujo es el que se puede apreciar en la Figura 42.

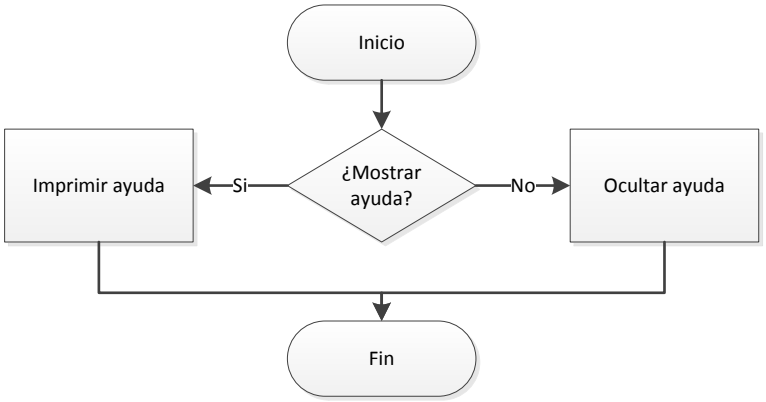


Figura 42: Diagrama de flujo de v3_show_help

4.5.4.4. Ordenar datos

Nombre de la función

sort_data

Entrada

all_data: datos necesarios para la impresión de la vista de la actividad temporal, el diagrama de calor y el diagrama de radar.

temporalFilter: objeto que almacena la información relativa al filtro temporal.

entity: entidad de usuario seleccionada o detectada.

Salida

data_result: datos y sus valores máximos generados.

-1: sin datos.

Descripción

A partir de la entidad y los filtros temporales calcula, ordena y formatea los datos para la vista de la actividad temporal, el diagrama de calor y el diagrama de radar. El diagrama de flujo es el que se puede ver en la Figura 43.

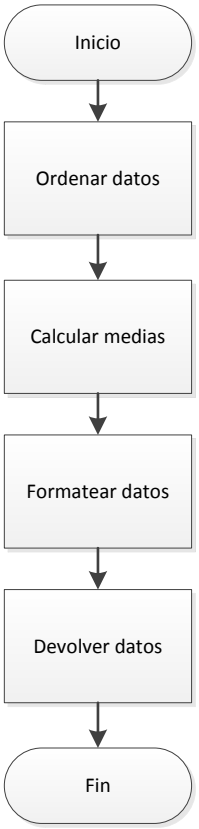


Figura 43: Diagrama de flujo de la función sort_data

4.5.4.5. Imprimir diagrama de calor

Nombre de la función

get_scatter_chart_data

Entrada

data_and_biggest: datos y sus valores máximos capturados.

eventTypeToUse: tipos de evento con mayor número de eventos

Salida

data_chart: datos generados.

Descripción

La función adapta los datos para que la API pueda generar las vistas. El diagrama de flujo es el que se puede ver en la Figura 44.

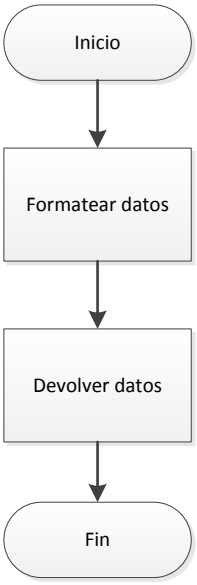


Figura 44: Diagrama de flujo de la función get_scatter_chart_data

Nombre de la función

print_scatter_chart

Entrada

data: datos y sus valores máximos capturados.

eventTypeToUse: tipos de evento con mayor número de eventos.

dictionary: array que guarda las cadenas de caracteres que se definen en el idioma.

Salida

Ninguna.

Descripción

Función que imprime el diagrama de calor. Es la función que interactúa con la API insertando sus valores de configuración. El diagrama de flujo de la función es el que se puede ver en la Figura 45.

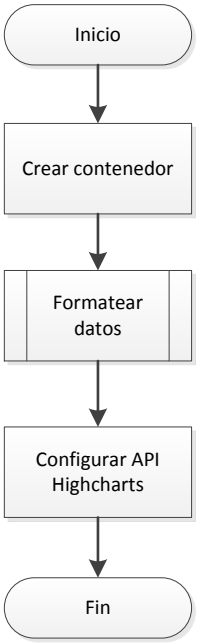


Figura 45: Diagrama de flujo de la función print_scatter_chart

4.5.4.6. Imprimir diagrama de radar

Nombre de la función

prepare_data_for_radar_chart

Entrada

data: objeto con los datos necesarios para la impresión de la vista.

MAX_PRINTABLE_VALUES: variable estática que guarda el número máximo de tipos de eventos que se pueden imprimir.

Salida

result_array: array con los datos adaptados a la API que genera la vista del radar.

-1: sin datos

Descripción

Genera los datos necesarios para el diagrama de radar. El diagrama de flujo es que se puede ver en la Figura 46.

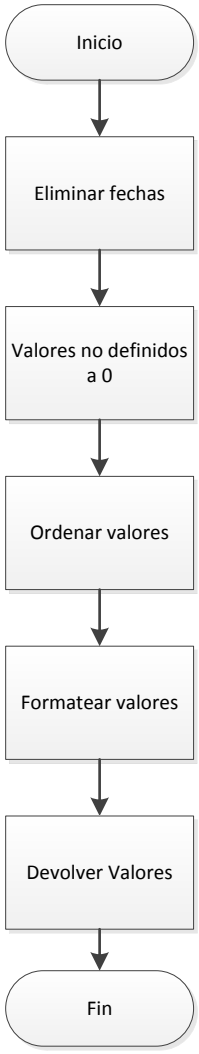


Figura 46: Diagrama de flujo de la función prepare_data_for_radar_chart

Nombre de la función

print_radar_chart

Entrada

data: objeto con los datos necesarios para la impresión de la vista.

MAX_PRINTABLE_VALUES: variable estática que guarda el número máximo de tipos de eventos que se pueden imprimir.

dictionary: array que guarda las cadenas de caracteres que se definen en el idioma.

Salida

eventTypeToUse: tipos de evento con mayor número de eventos.

Descripción

Función que formatea los datos para que puedan ser utilizados por la API que genera la visualización y que devuelve, para que así puedan ser aprovechadas

por otras vistas, aquellos tipos de eventos que poseen un mayor número de eventos. El diagrama de flujo de la función es el que se puede ver en la Figura 47.

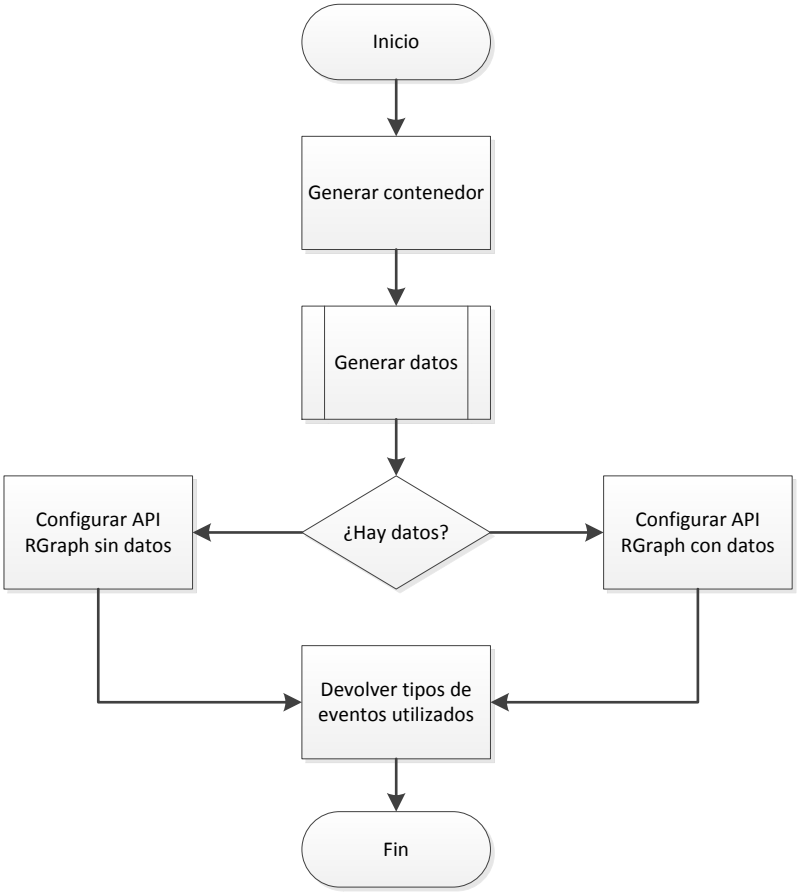


Figura 47: Diagrama de flujo de la función print_radar_chart

4.5.4.7. Imprimir indicadores

Nombre de la función

get_metadata_and_print_indicator

Entrada

all_metadata: objeto con todos los metadatos.

ENTITY: nombre de la identidad objeto de análisis.

dictionary: array que guarda las cadenas de caracteres que se definen en el idioma.

Salida

Ninguna.

Descripción

Función encargada de imprimir los diferentes indicadores. Al ser una función que depende de los indicadores de la base de datos, genera diferentes contenedores bajo demanda. El diagrama de flujo de la función es el que se puede ver en la Figura 48.

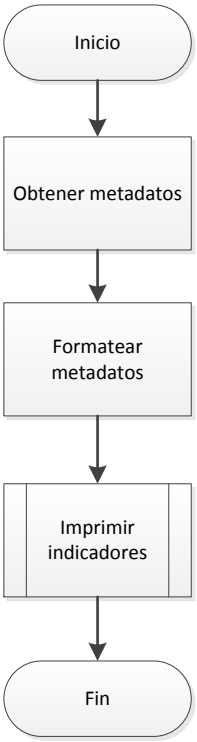


Figura 48: Diagrama de flujo de la función get_metadata_and_print_indicator

Nombre de la función

print_indicartors

Entrada

metadata: objeto con los metadatos necesarios para la impresión de los indicadores.

ENTITY: entidad objeto de análisis.

PREFIX: "*score*", es decir, la palabra clave (prefijo) que identifica los datos necesarios para esta vista.

dictionary: array que guarda las cadenas de caracteres que se definen en el idioma.

Salida

Ninguna.

Descripción

Función que imprime todas las puntuaciones según los datos que se le pasan por referencia. Es el encargado de generar el contenedor que alberga todos los subcontenedores y las opciones necesarias para producir las animaciones gráficas y las interactuaciones con la vista. El diagrama de flujo es el que se puede ver en la Figura 49.

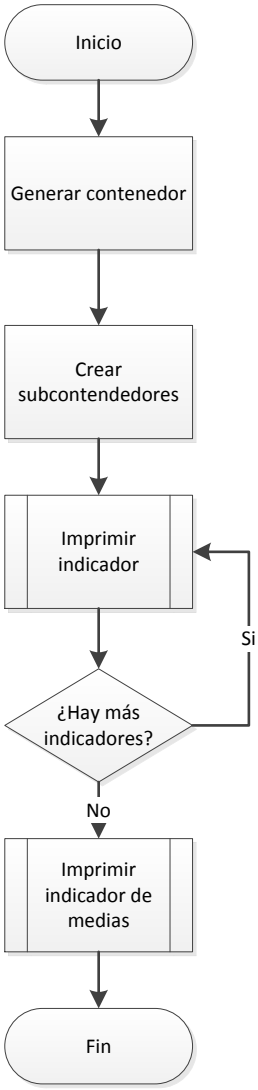


Figura 49: Diagrama de flujo de la función print_indicartors

Nombre de la función

print_indicator_graph

Entrada

index: variable que identifica inequívocamente al indicador.

divtoplace: contenedor padre donde va localizado el indicador.

values: datos del indicador.

grap_title: título del indicador.

Grapthtype: tipo de gráfica del indicador.

Salida

Ninguna.

Descripción

La función configura la API para generar gráficamente el indicador. Esta función es llamada por su función padre tantas veces como indicadores haya. El diagrama de flujo es el que se puede ver en la Figura 50.

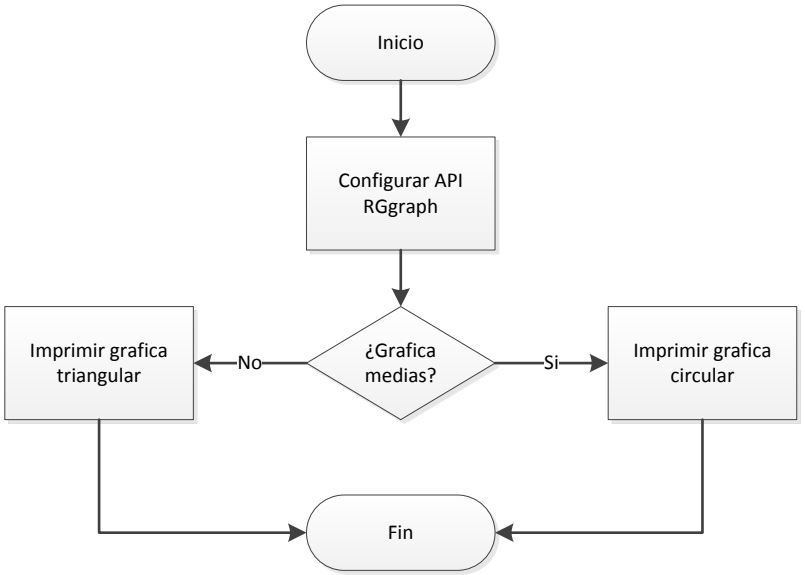


Figura 50: Diagrama de flujo de la función print_indicator_graph

4.5.4.8. Imprimir actividad temporal

Nombre de la función

get_column_chart_data

Entrada

data: objeto con el conjunto de datos necesarios para imprimir la vista.

eventTypeToUse: tipos de evento con mayor número de eventos.

dictionary: array que guarda las cadenas de caracteres que se definen en el idioma.

Salida

result_data: array con los datos formateados para que puedan ser utilizados por el API.

Descripción

Función encargada de formatear los datos para que puedan ser utilizados por el API buscando aquellos valores que son propios de la vista. El diagrama de flujo es el que se puede ver en la Figura 51.

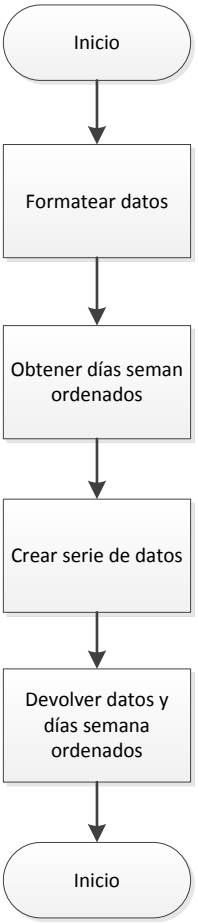


Figura 51: Diagrama de flujo de la función get_column_chart_data

Nombre de la función

print_column_chart

Entrada

data: objeto con el conjunto de datos necesarios para imprimir la vista.

eventTypeToUse: tipos de evento con mayor número de eventos.

dictionary: array que guarda las cadenas de caracteres que se definen en el idioma.

Salida

Ninguna.

Descripción

Función encargada de imprimir la vista de la actividad temporal. Esta función interactúa directamente con la API encargada de generar la gráfica. El diagrama de flujo es el que se puede ver en la Figura 52**.**

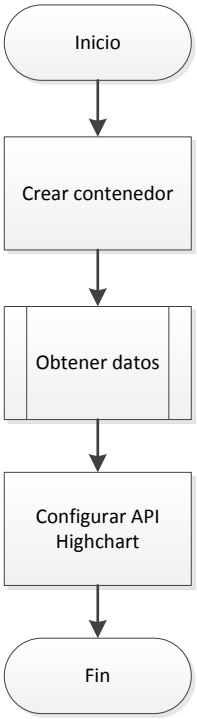


Figura 52: Diagrama de flujo de la función print_column_chart

4.5.4.9. Imprimir diagrama de compromiso

Nombre de la función

get_column_pool_chart_data

Entrada

eqdata: array con los datos necesarios para la impresión de la vista.

entity_name: nombre de la entidad objeto de análisis.

dictionary: array que guarda las cadenas de caracteres que se definen en el idioma.

Salida

result_data: array con los datos formateados y divididos para que puedan ser utilizados por el API para cada gráfica.

Descripción

Función encargada de formatear los datos para que puedan ser utilizados por el API buscando aquellos valores que son propios de la vista del diagrama de compromiso. Debido a que esta función posee varias gráficas es necesario generar varios conjuntos de datos. El diagrama de flujo es el que se puede ver en la Figura 53.

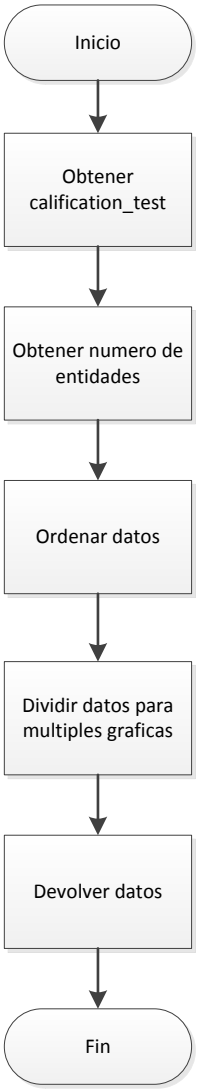


Figura 53: Diagrama de flujo de la función get_column_pool_chart_data

Nombre de la función

print_column_pool_chart

Entrada

eq_data: array con los datos necesarios para la impresión de la vista.

entity_name: nombre de la entidad objeto de análisis.

dictionary: array que guarda las cadenas de caracteres que se definen en el idioma.

Salida

Ninguna.

Descripción

Función encargada de imprimir la vista del diagrama de compromiso. Genera los diferentes contenedores para imprimir cada gráfica y las opciones necesarias para interactuar con ellas. El diagrama de flujo es el que se puede ver en la Figura 54.

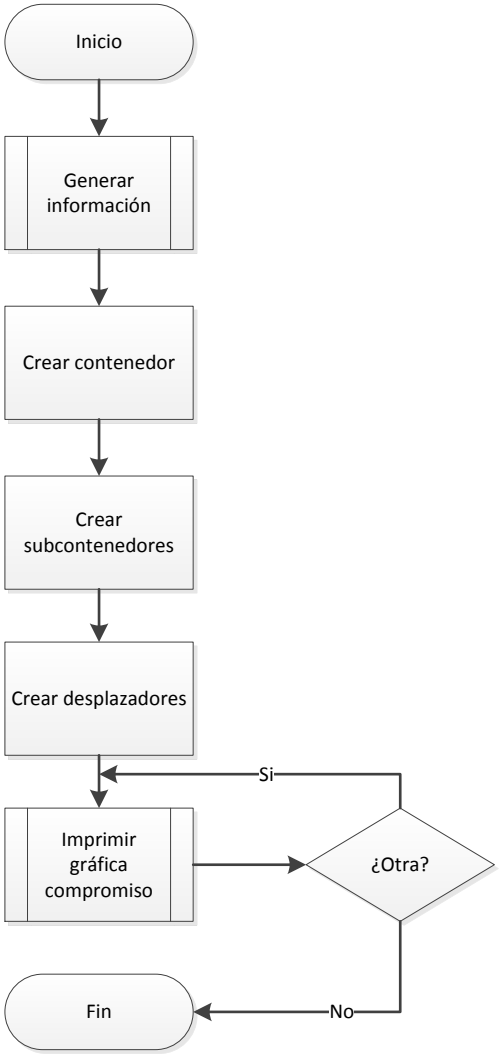


Figura 54: Diagrama de flujo de la función print_column_pool_chart

Nombre de la función

generate_column_pool_chart

Entrada

graph_data: datos de la gráfica en particular.

divcontainer: subcontenedor donde insertar la gráfica.

dictionary: array que guarda las cadenas de caracteres que se definen en el idioma.

Salida

Ninguna.

Descripción

Función encargada de imprimir la gráfica en particular. Esta función interactúa directamente con la API. La función es llamada tantas veces como gráficas se hayan generado según los datos de la entidad. El diagrama de flujo es el que se puede ver en Figura 55.

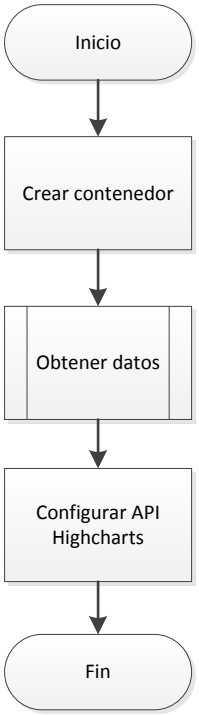


Figura 55: Diagrama de flujo de la función generate_column_pool_chart

4.5.5. Compartir Widget

GLASS ofrece la posibilidad de poder compartir los resúmenes al exterior. Para ello es necesario que no se haga un control de sesión pero se interactúe con la base de datos para capturar la información de un Widget. Aunque el encargado de imprimir por pantalla propiamente el Widget no es este script, puesto que llama a widget.html para ello, es el encargado de todo lo demás. Aparece contenido en *share.php* y tiene el diagrama de flujo que se puede ver en la Figura 56.

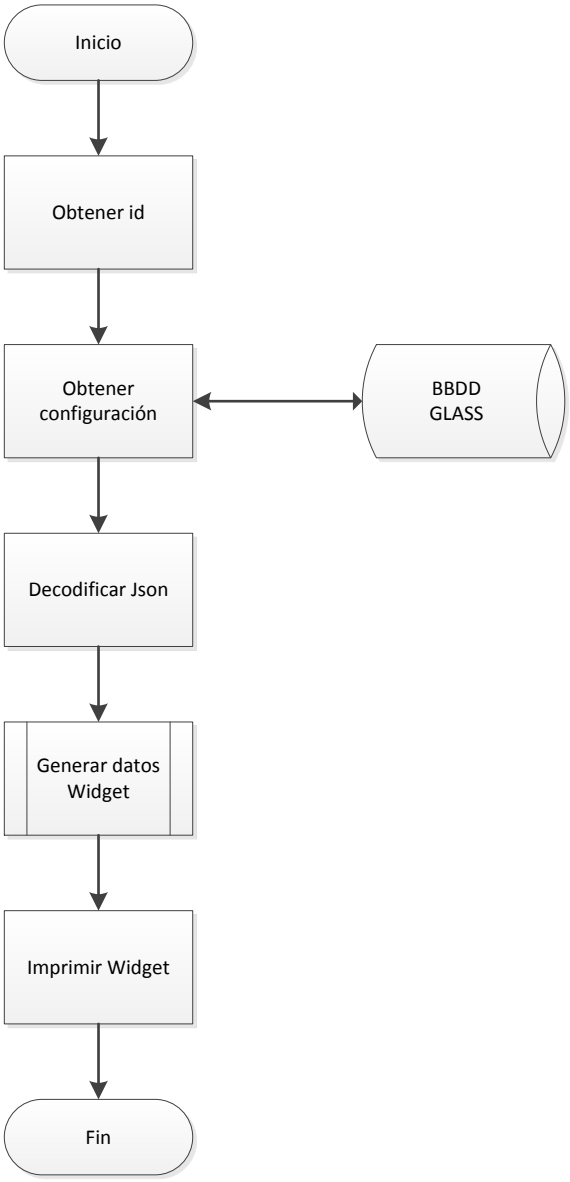


Figura 56: Diagrama de flujo del script que contiene el archivo share.php

4.5.6. Imprimir Widget

Está formado por los archivos *widget.php* y *widget.html*. El primero es un script que llama a la función que obtiene los datos del Widget del mismo modo que lo hacía *share.php*, para a continuación llamar al segundo archivo que genera los contenedores e imprimir las diferentes gráficas por pantalla. A continuación se describen y detallan las 4 funciones contenidas en el fichero HTML.

Nombre de la función

V3_print_graphs

Entrada

data: objeto con todos los datos del Widget.

numwidget: identificador que define inequívocamente al Widget para el usuario registrado

Salida

Ninguna.

Descripción

Función encargada de imprimir cada una de las gráficas que conforman el resumen o Widget. Para ello ser sirve de 3 funciones. El diagrama de flujo es el que se puede ver en la Figura 57.

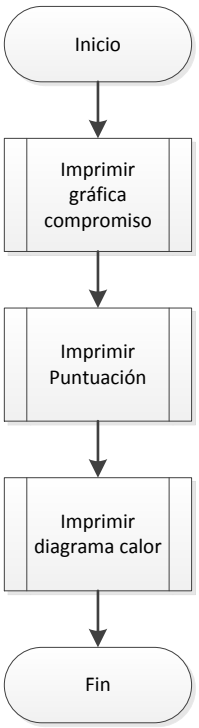


Figura 57: Diagrama de flujo de la función V3_print_graphs

Nombre de la función

V3_scatter

Entrada

data: objeto con todos los datos que generan la gráfica de la actividad temporal.

numwidget: identificador que define inequívocamente al Widget para el usuario registrado.

dictionary: array que guarda las cadenas de caracteres que se definen en el idioma.

Salida

Ninguna

Descripción

Función encargada de imprimir el diagrama de calor del Widget. Para ello necesita formatear los datos para que puedan ser interpretados por la API. El diagrama de flujo es el que se puede ver en la Figura 58.

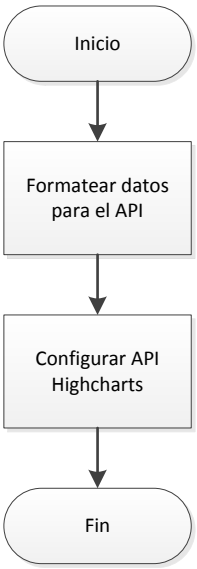


Figura 58: Diagrama de flujo de la función V3_scatter

Nombre de la función

V3_engagement

Entrada

Eng: array con los datos de la gráfica del compromiso.

numwidget: identificador que define inequívocamente al Widget para el usuario registrado.

dictionary: array que guarda las cadenas de caracteres que se definen en el idioma.

Salida

Ninguna.

Descripción

Función encargada de imprimir la gráfica de compromiso del Widget. Sólo configura la API ya que los datos le llegan directamente formateados. El diagrama de flujo es el que se puede ver en la Figura 59.

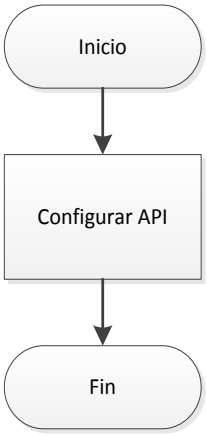


Figura 59: Diagrama de flujo de las funciones V3_engagement y V3_scores

Nombre de la función

V3_scores

Entrada

score: array con las puntuaciones.

numwidget: identificador que define inequívocamente al Widget para el usuario registrado.

dictionary: array que guarda las cadenas de caracteres que se definen en el idioma.

Salida

Ninguna.

Descripción

Función encargada de imprimir el indicador de la puntuación global que posee el Widget. Sólo precisa de configuración de la API ya que los datos ya le llegan formateados. El diagrama de flujo es el que se puede ver en la Figura 59.

4.5.7. API's gráficas

A continuación se proporciona una pequeña descripción de las dos API's utilizadas para la generación de los distintos gráficos que utiliza el módulo. Para más información las API's poseen una extensa documentación con diversos ejemplos que ayudan en gran medida a comprender su funcionamiento. Se han utilizado ambas librerías y no sólo una de ellas debido a que hay tipos de gráficos y funcionalidades que son soportados por una API que para la otra no están disponibles.

4.5.7.1. Highcharts

Highcharts [62] es una librería gráfica escrita en JavaScript que ofrece gráficas intuitivas e interactivas a cualquier aplicación Web. Por el momento, Highcharts soporta gráficas de tipo lineales, área, columnas, barras, tarta, scattering, galgas angulares, rangos de área, rangos de columnas y polar.

Funciona en la mayoría de los exploradores, incluyendo el del iPhone/iPad e Internet Explorer desde la versión 6. Los exploradores estándares utilizan SVG para la renderización gráfica. Para los gráficos de Internet Explorer se utiliza VML.

No se necesitan de los permios del auto para su utilización, sólo es necesario adquirir una de las dos licencias disponibles. Una de las características claves de Highcharts es que, bajo cualquier tipo de licencias, ya sea libre o no, se puede descargar el código fuente y que el usuario haga sus propias ediciones. Esto permite una gran flexibilidad ofreciendo personalizaciones personales.

Highcharts únicamente se basa en tecnologías de navegador nativo y no requiere plugins de lado del cliente como Flash o Java. Además, no se necesitas instalar nada en el servidor. No utiliza PHP o ASP.NET. Highcharts necesita sólo dos archivos JS para ejecutar: la base de highcharts.js y un marco jQuery, MooTools o Prototype. Seguramente uno de estos marcos sea utilizado en la página Web del usuario.

Para la configuración de las opciones de configuración Highcharts, no se requiere ninguna habilidad especial de programación. Las opciones se establecen siguiendo una estructura de objetos propia de JavaScript, que es básicamente un conjunto de claves y valores conectados por dos puntos, separados por comas y agrupados por llaves.

A través de una completa API se puede agregar, quitar y modificar la serie, sus puntos o ejes, en cualquier momento después de la creación de la tabla. Numerosas fuentes de eventos son programables contra la tabla. La combinación de jQuery, MooTools o Ajax, abre la posibilidad de soluciones como cartas en vivo actualizando constantemente con valores que provienen del servidor, como los datos de usuario y mucho más.

A veces se desea comparar variables que no están en la misma escala. Por ejemplo, la temperatura versus presión de aire y la lluvia. Highcharts permite asignar un eje para cada serie o un eje X si desea comparar conjuntos de datos de diferentes categorías. Cada eje puede colocarse a la derecha o izquierda, parte superior o inferior de la tabla. Todas las opciones pueden ajustarse individualmente, incluyendo inversión, estilo y posición.

Highcharts puede mostrar un texto de descripción con información sobre cada punto y la serie al que pertenece. La descripción sigue mientras el usuario mueve el ratón sobre el gráfico, y han hecho grandes esfuerzos para que aparezca el punto más cercano, así como lo que no es fácil de leer, como puede ser algo que está por debajo de otro punto.

El 75% de todos los gráficos formados por los ejes X e Y tiene una fecha y hora en el eje X. Por lo tanto Highchart es muy inteligente operando con valores de tiempo. Con las unidades del eje en milisegundos, Highcharts determina dónde colocar las marcas que denotan el inicio del mes o semana, medianoche y mediodía, la hora completa, etc.

Con el módulo exportador habilitado, los usuarios pueden exportar la tabla a formato PNG, JPG, PDF o SVG con sólo un botón, o imprimir la tabla directamente desde la página web.

Haciendo zoom en un gráfico se puede examinar más de cerca una parte interesante de los datos. El zoom se puede realizar tanto en la dimensión del eje X o Y, como el de ambos.

Highcharts toma los datos en una matriz de JavaScript, que puede ser definida en el objeto de configuración local, en un archivo separado o incluso en un sitio diferente. Además, los datos pueden tratarse sobre Highcharts en cualquier formato, y se utiliza una función de devolución de llamada para analizar los datos en una matriz.

Es ideal para proporcionar tableros de instrumentos, medidores angulares o velocímetro como gráficos, fáciles de leer en un solo vistazo.

Tipos de gráfico cartesiano en línea o área se pueden convertir en un gráfico polar o radial mediante una simple opción.

A veces necesita voltear su carta para que el eje X aparezca en vertical, como por ejemplo en un diagrama de barras. Invertir el eje, con los valores más altos que aparecen más cercanos a origen, es también compatible.

Todas las etiquetas de texto, incluyendo los rótulos de los ejes, las etiquetas de datos y el título de puntos y ejes, se pueden girar en cualquier ángulo.

4.5.7.2. RGraph

RGraph [63] es una librería HTML5 de gráficos que utilizan la etiqueta canvas para dibujar alrededor de 20 tipos diferentes de gráficos. Con la nueva etiqueta canvas de HTML5, RGraph crea estos cuadros en el navegador Web con JavaScript, lo que da lugar a páginas más rápidas y menos carga en el servidor web. Esto conduce a tamaños de página más pequeños, menores costes y sitios web más rápidos.

RGraph es libre de usar en sitios web no comerciales, tales como blogs personales, educativos o sitios de caridad bajo la licencia Creative Commons (no comercial) por lo que no necesita comprar una licencia, sólo añadir un enlace a la Página Web de RGraph en el sitio desarrollado. El código fuente viene completamente incluido en la descarga y puede ser modificado como sea necesario.

Las características principales de RGraph son:

- Tooltips
- Efectos visuales
- Zooming
- Menús contextuales
- Ajustes interactivos
- Eventos personalizables
- Soporta AJAX
- Soporta JSON
- Actualización dinámico
- Redimensionado
- Soporta MSIE 7/8
- Anotaciones
- Combinar diferentes tipos de gráficas

Todos los gráficos y características están documentados con ejemplos y líneas de código que se pueden utilizar. Las guías HOWTO guían al usuario realizando tareas comunes paso a paso con ejemplos de código JavaScript.

El código fuente está incluido en la descarga y el usuario es libre de realizar todos los cambios que vea oportuno. La API está documentada y existen varios ejemplos básicos que ayudan a comprender los diferentes tipos de gráficas con sus diversos tipos de funcionalidades

Uno se puede olvidar de Flash o Silverlight, se están incorporando JavaScript y HTML5 con canvas en todos los navegadores modernos y permiten fácilmente generar dibujos en 2D. RGraph permite producir sus gráficos facilitando el proceso para así obtener rápidamente las funcionalidades que se deseen. Canvas es compatible con todos los navegadores modernos y dispositivos móviles, lo que significa que las tablas y gráficos sean vistas por el mayor número posible de usuarios.

4.6. Manual de usuario

A lo largo de esta sección se va a proceder a describir el manual de usuario del módulo. Debido a que el objeto del proyecto es la visualización 3 no se va a entrar en detalle en la descripción de GLASS más allá de lo que pueda ser útil para el módulo, debido a que en secciones anteriores ya se ha hablado extendidamente de ello.

La visualización 3, como ya se ha comentado, es un módulo diseñado para funcionar en GLASS, que permite la visualización de una manera sencilla del rendimiento de los alumnos y como se ven con respecto a su clase, curso o población objeto de análisis. Como se trata de una aplicación Web, requiere de un servidor que puede funcionar tanto en Linux como en Windows con un paquete LAMP o WAMP dependiendo del sistema donde se ejecute, una o varias bases de datos Mongo y tener instalada la herramienta GLASS. Una vez instalado en el servidor, sólo se precisa de un navegador Web y una conexión a Internet para poder interoperar con él. GLASS y el módulo están diseñados para que corran bajo Chrome y Firefox, aunque las revisiones sobre este último han sido menos exhaustivas.

En esta sección se va a empezar enumerando los requisitos del sistema, necesarios para poder instalar el módulo. Seguidamente, se describen las instrucciones para instalar el módulo en GLASS. Y para finalizar, se dan unas nociones básicas para manejarse con el módulo.

4.6.1. Requisitos del sistema

A continuación se describen los requisitos que deben tener tanto en el servidor como el cliente. Los requisitos del servidor son:

- Herramienta GLASS en perfecto funcionamiento
- Una o varias bases de datos Mongo con metadatos del tipo *score* y *embedded_question* y con una estructura de datos entendible para GLASS.

Los requisitos del cliente son:

- Navegador Web Google Chrome con conexión a internet o conexión a la red donde se encuentre el servidor.

4.6.2. Instalación del módulo

Para la instalación del módulo sólo es necesario copiar el directorio raíz del módulo en la carpeta *visualizations de GLASS*. En cuanto esto se haya hecho, GLASS detectará automáticamente la presencia de nuevos módulos y los tendrá listos para instalar en la opción de gestión de módulos de la herramienta. Si en el momento de la copia se está visualizando la sección de gestión de módulos, se deberá refrescar la página o volver a entrar en tal opción para que GLASS detecte la nueva visualización.

4.6.3. Guía de uso

Una vez que el usuario ha hecho login en GLASS puede acceder al módulo de las 3 formas que GLASS nos propone:

- Por el menú de visualizaciones.
- A través del resumen o Widget, siempre y cuando haya uno insertado en el tablero o Dashboard.
- A través de mis vistas favoritas, siempre y cuando se haya agregado un resumen a dicha sección.

Lo primero con lo que el usuario se encuentra tras acceder a la visualización es con la captura de pantalla que se puede ver en la Figura 60 o en la Figura 61.

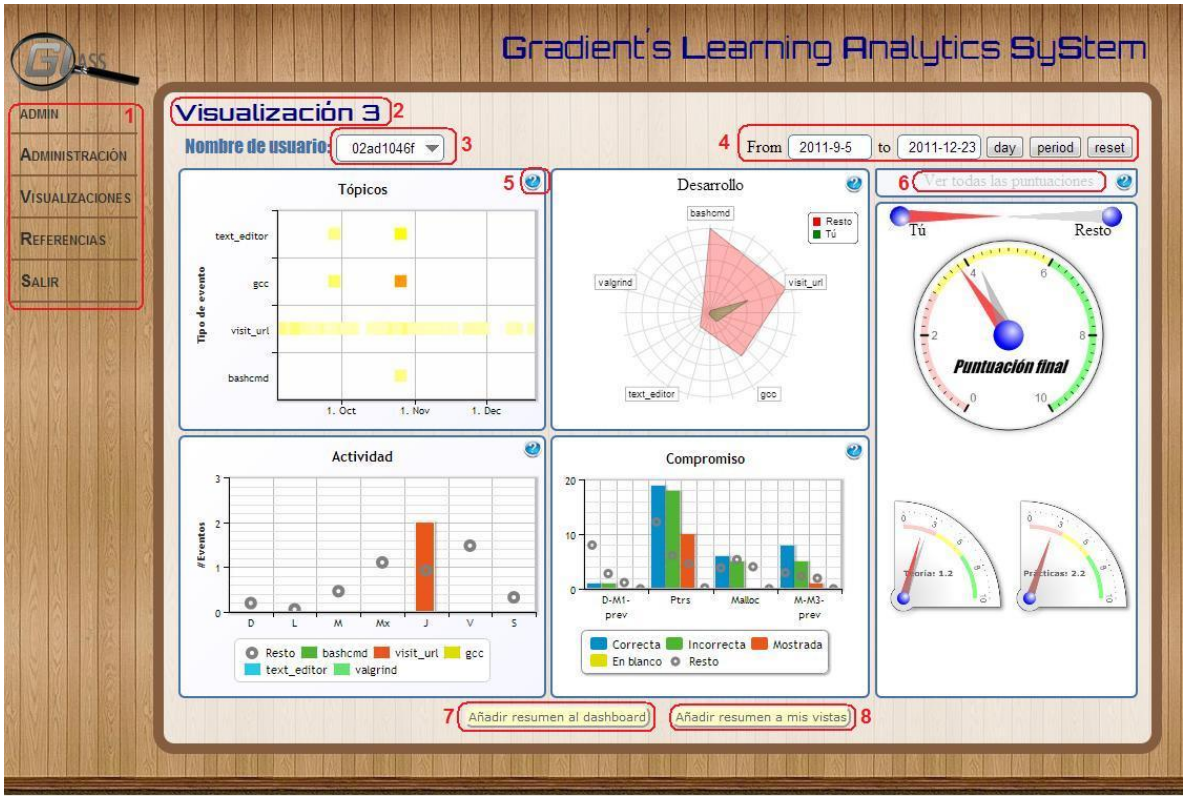


Figura 60: Manual de usuario 1

En la Figura 60, Figura 61 y Figura 62 se pueden apreciar en rojo los siguientes elementos:

- 1: Menú de GLASS: permite al usuario dirigirse a cualquier facilidad de la herramienta.
- 2: Nombre del módulo.
- 3: Filtro de usuarios.
- 4: Filtro temporal.
- 5: Ayuda de la vista.
- 6: Desplegar todas las calificaciones.
- 7: Agregar un resumen al Dashboard.
- 8: Añadir un resumen a mis vistas favoritas.
- 9: Logo de GLASS.
- 10: Significado del acrónimo GLASS.
- 11: Desplazadores hacia la izquierda (a) o derecha (b) para recorrer las diferentes gráficas del diagrama de compromiso.
- 12: Desplazadores hacia la izquierda (a) o derecha (b) para recorrer los diferentes indicadores.
- 13: Ocultar las calificaciones parciales.

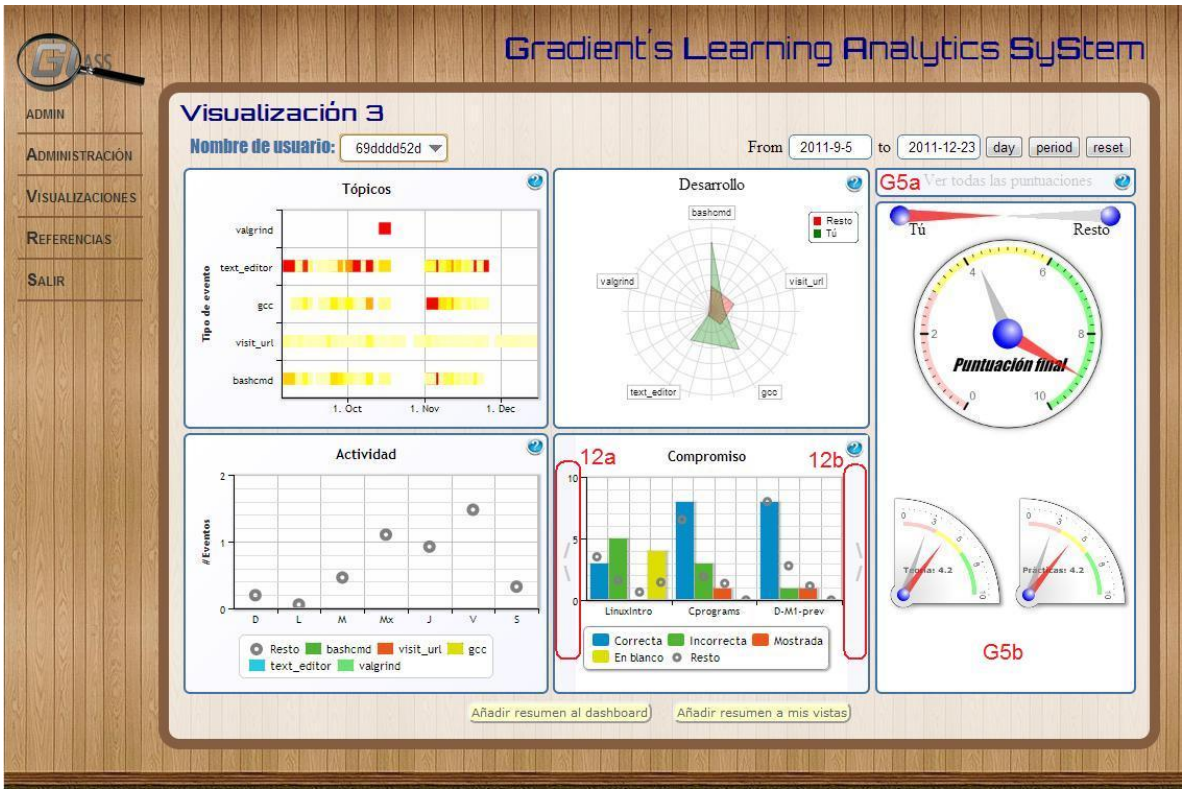


Figura 61: Manual de usuario 2

En la Figura 61 y Figura 62 se enumeran en rojo los diferentes tipos de vistas:

- G1: Diagrama de calor con los diferentes tópicos.
- G2: Diagrama de radar.

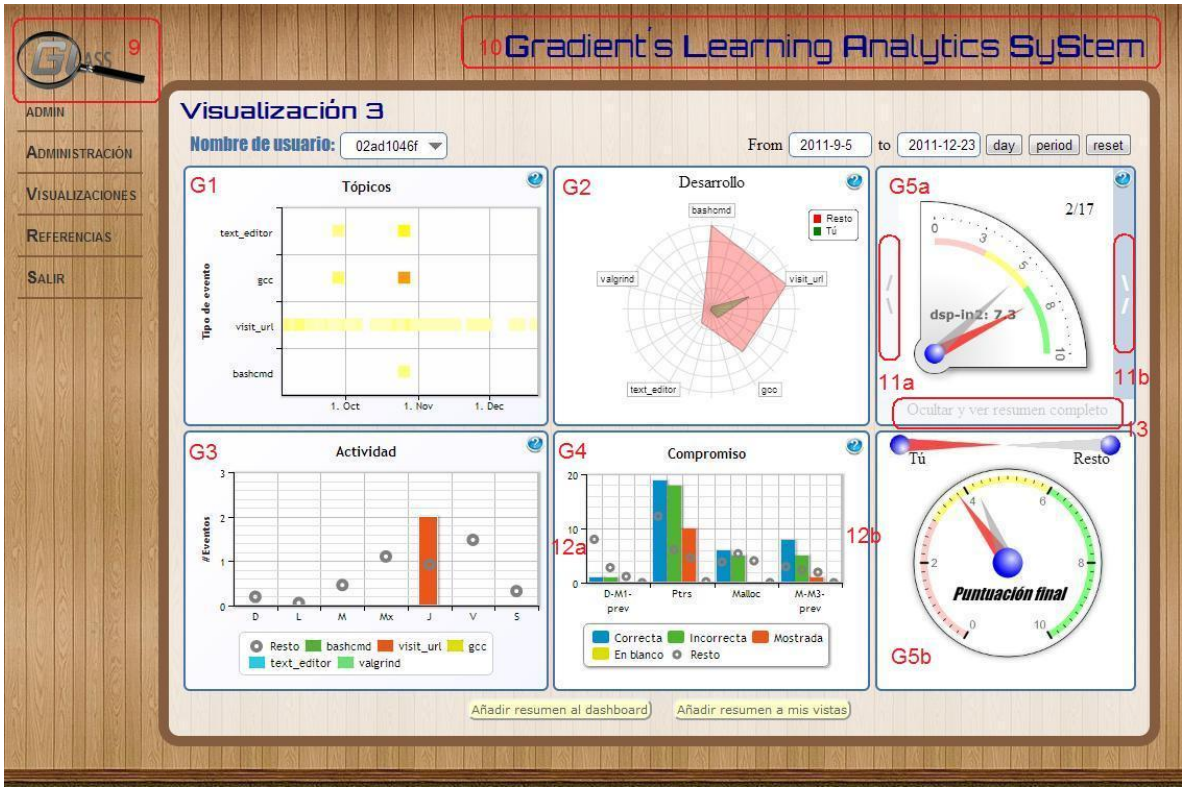
- G3: Actividad temporal.
- G4: Diagrama de compromiso.
- G5: Puntuaciones, tanto parciales (a) como las medias (b). Estas últimas se dividen en media total, la cual se representa como un indicador circular, y la media de laboratorio y media de teoría, representadas como dos gráficas triangulares más pequeñas, justo debajo de la anterior, y que desaparecen al desplegar las puntuaciones parciales pulsando en 6. Si se pulsa en 13, las puntuaciones parciales se ocultarían nuevamente.
- 

Figura 62: Manual de usuario 3

Para finalizar, indicar que los desplazadores están desactivados si sólo existiese una gráfica de compromiso o si sólo apareciese un indicador parcial.

En todas las gráficas se añaden leyendas para el mejor entendimiento de éstas y la mayoría proporcionan información si se pasa el ratón por encima de alguno de sus puntos o barras.

4.6.4. Solución de problemas

Debido a que el módulo se ha intentado simplificar para el usuario de la mejor forma posible y ha ido sujeto a constantes revisiones para la detección de bus, los problemas con los que el usuario se puede encontrar son bastante reducidos.

Si se selecciona a través de los filtros una configuración que no posee datos, las gráficas aparecerán vacías y en muchas de ellas aparecerá (no data) para indicar que no se han capturado datos.

Un problema que puede surgir es que cuando se entra en la visualización, ésta no tenga datos incluso dándole al reseteo. Debido a que GLASS, cuando se accede por el menú de visualizaciones, carga el periodo total de datos de la misma forma que cuando se da al reseteo del filtro temporal, si las gráficas aparecen con el indicador de que no tienen datos, indica que probablemente se deba a que la base de datos Mongo está mal instalada o se presentan problemas de acceso, por lo que habrá que hablar con algún técnico de GLASS. También puede ser debido a que Mongo haya tirado la petición, lo cual se solucionará refrescando el navegador. También se puede dar cuando se prueba por primera vez una base de datos que acaba de ser instalada debido a que esté haciendo operaciones de reducción de datos, o por qué este proceso se haya interrumpido por algún problema en el servidor.

Si aparece algún mensaje en rojo es que se están produciendo problemas relativos a GLASS que normalmente son de la base de datos MySQL que necesita para funcionar. Como ocurre con el caso anterior, es recomendable consultar a algún técnico de GLASS.

También se puede dar el caso que a un estudiante sólo le aparezca la información media de todos los usuarios pero no la suya propia. Esto se debe a que probablemente el usuario no se encuentre en la base de datos, y por lo tanto, al no poder mostrar la información de otro usuario debido a sus permisos, para garantizar la confidencialidad GLASS sólo muestra datos medios de población.

Capítulo 5

Presupuesto

En este capítulo se presentan los costes globales de la realización de este Proyecto Fin de Carrera. Para ello, se desglosan los gastos de la siguiente forma: organización y sueldos del personal involucrado, los materiales requeridos para la consecución y realización del mismo. Se finaliza con el resumen del presupuesto total del Proyecto.

5.1. Recursos humanos

Para la realización de este proyecto se definieron los siguientes roles:

- **Profesor:** Abelardo Pardo, Derick Leony y Luis de la Fuente Valentín. Todos ellos fueron o son participantes en el desarrollo de GLASS.
- **Investigador:** David Sánchez de Castro, autor material del proyecto.

Además, dependiendo de la fase de trabajo se definen los siguientes grupos:

- **Grupo analista:** formado por los miembros que desarrollaron GLASS. Está compuesto por Abelardo Pardo, Derick Leony, Luis de la Fuente Valentín y David Sánchez de Castro. Para este proyecto sólo se reúnen para realizar el análisis inicial del módulo tras la propuesta presentada por Abelardo Pardo.
- **Tutor:** Derick Leony. Se encarga de las revisiones del proyecto y de la ayuda al análisis de los diferentes problemas surgidos.

- **Desarrollador:** constituido por el autor del proyecto y por lo tanto encargado de realizar la completa ejecución del mismo.

5.2. Planificación temporal

Para el desarrollo del proyecto se definen las siguientes fases:

- **Fase 1:** Análisis del módulo.
- **Fase 2:** Desarrollo software.
- **Fase 3:** Primera revisión.
- **Fase 4:** Desarrollo software.
- **Fase 5:** Segunda revisión.
- **Fase 6:** Retoques finales.
- **Fase 7:** Documentación.
- **Fase 8:** Revisión final.
- **Fase 9:** Cierre y presentación del proyecto.

En la Tabla 9 se pueden ver las diferentes fases del proyecto con los roles involucrados y las horas totales que se han dedicado a cada fase.

Fase	Descripción	Grupo de trabajo	Tiempo [horas]
1	Análisis	Grupo analista	2
2	Desarrollo software	Desarrollador	166
3	Primera revisión	Profesor y desarrollador	4
4	Desarrollo software	Desarrollador	84
5	Segunda revisión	Profesor y desarrollador	4
6	Retoques finales	Desarrollador	44
7	Documentación	Desarrollador	80
8	Revisión final	Tutor	8
9	Cierre y presentación del proyecto	Desarrollador	8

Tabla 9: Fases y recursos del proyecto

Se define un día de trabajo laboral como 8 horas de trabajo y una semana como 5 días laborales. Además se establece que el sueldo de un profesor es de 30€/hora y el de un investigador 20€/hora. El diagrama temporal en forma de diagrama de Gantt resultante es el que se puede ver en la Figura 63.

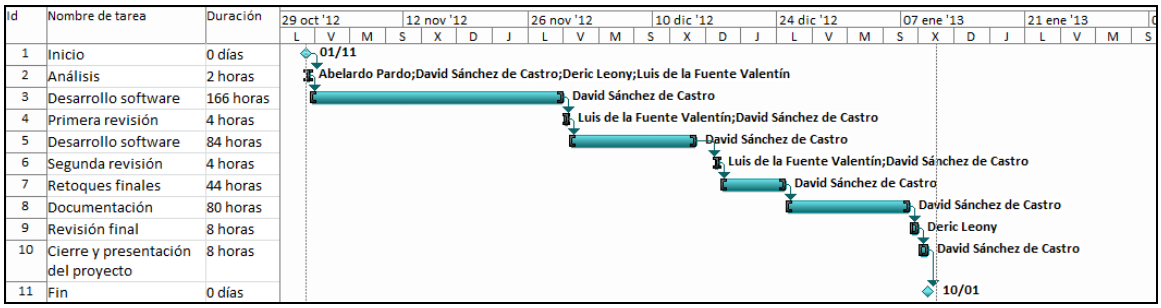


Figura 63: Gantt del proyecto

5.3. Costes humanos

Gracias a la herramienta MS Project de Microsoft se han podido generar como se aprecia en la Tabla 10, los costes relativos a cada una de las tareas. Se ha dividido el coste por mes en vez de por semana para no recargar en exceso la tabla.

	noviembre	diciembre	enero	febrero	Total
Inicio					
Análisis	220,00 €				220,00 €
Desarrollo software	3.320,00 €				3.320,00 €
Primera revisión	200,00 €				200,00 €
Desarrollo software	80,00 €	1.600,00 €			1.680,00 €
Segunda revisión		200,00 €			200,00 €
Retoques finales		880,00 €			880,00 €
Documentación		800,00 €	800,00 €		1.600,00 €
Revisión final			240,00 €		240,00 €
Cierre y presentación del proyecto			160,00 €		160,00 €
Fin					
Total	3.820,00 €	3.480,00 €	1.200,00 €		8.500,00 €

Tabla 10: Costes humanos por tarea

Como se puede ver en la Tabla 11, también se han generado los costes totales por cada mes de cada recurso humano interviniente en el proyecto.

	noviembre	diciembre	enero	febrero	Total
David Sánchez de Castro	3.520,00 €	3.360,00 €	960,00 €		7.840,00 €
Abelardo Pardo	60,00 €				60,00 €
Deric Leony	60,00 €		240,00 €		300,00 €
Luis de la Fuente Valentín	180,00 €	120,00 €			300,00 €
Total	3.820,00 €	3.480,00 €	1.200,00 €		8.500,00 €

Tabla 11: Costes humanos

En resumen, como se puede ver, el coste del personal asciende a:

Coste de personal: **8.500€**

5.4. Costes de material

Debido a que la consecución del proyecto depende nada más de un ordenador personal para poder desarrollarse, se ha contabilizado en 600€ la cantidad de dinero a añadir en concepto relativo al entorno de trabajo. Además, como el coste de escalabilidad de un nuevo módulo en el servidor se puede considerar despreciable, no se contabilizan más gastos de materia.

En resumen, el coste del material asciende a:

Coste del material: **600€**

5.5. Coste Económico total

El presupuesto final para la realización del Proyecto de fin de Carrera, teniendo en cuenta todos los gastos calculados en las dos secciones anteriores, un margen establecido del 10% y la carga impositiva del IVA que en España actualmente es del 21%, es el que aparece detallado en Tabla 12.

Concepto	Importe
Coste personal	8.500€
Coste material	600€
Margen 10%	910€
Base imponible	10.010€
IVA (21%)	2.102€
Total	**12.112€**

Tabla 12: Costes totales

En resumen, el coste total del Proyecto Fin de Carrera (PFC) asciende a:

Coste total: 12.112€

Leganés a 28 de Febrero de 2013

El ingeniero proyectista:

Fdo. David Sánchez de Castro

Capítulo 6

Conclusiones y líneas futuras

En esta sección se describen las conclusiones detectadas a partir del trabajo realizado en este Proyecto de Fin de Carrera y algunas de las posibles líneas futuras que permitirían seguir más allá de este proyecto.

6.1. Conclusiones

Las interfaces de usuario mejoradas como las bibliotecas digitales y bases de datos multimedia han dado lugar a atractivos productos. En contra de un texto complejo, se están utilizando consultas flexibles, sonido, gráficos, imágenes, bases de datos y vídeos. Ahora es posible que, junto a los paquetes estadísticos y la formulación de consultas, se añadan aproximaciones gráficas de manipulación directa.

La visualización de información se está moviendo fuera de los laboratorios de investigación con un número cada vez mayor de productos comerciales, añadidos a los paquetes estadísticos, y al desarrollo de ambientes comerciales y, como en el caso de este proyecto, para la mejora de la educación en e-learning.

Aunque el ordenador contribuye a la explosión de la información, es también potencialmente la lente mágica para buscar, ordenar, filtrar, y presentar los productos en cuestión. La necesidad de búsqueda en documentos estructurados complejos, gráficos,

imágenes y archivos de sonido o video presenta grandes oportunidades para el diseño de interfaces de usuario avanzadas. Potentes motores de búsqueda serán capaces de encontrar las agujas en los pajares y los bosques más allá de los árboles. Los métodos que pueden utilizar las herramientas de exploración de información, tales como consultas dinámicas, diagramas de árbol, interfaces de usuario, interfaces jerárquicas y coordenadas paralelas, son sólo algunos de los inventos que tendrán que ser domesticados y validados por los investigadores de la interfaz de usuario. Es necesaria una mejor integración con la psicología perceptual (comprensión preatencional de los procesos y el impacto de la codificación o variado destacando técnicas) y con la toma de decisiones empresariales (identificando tareas y procedimientos que se dan en situaciones reales), como lo son los fundamentos teóricos y prácticos para la elección de los puntos de referencia entre las diversas técnicas de visualización. La exploración de la información puede resultar abrumadora en interfaces complejas para los nuevos usuarios y pueden ser útiles los nuevos métodos de demostración. Los estudios empíricos ayudarán a resolver la situación específica en la que la visualización es de gran ayuda.

En la actualidad ha surgido el concepto del análisis del aprendizaje para medir, recopilar, analizar y presentar los datos de los alumnos y su contexto. Estas son las bases fundamentales que han hecho pensar en la existencia de una herramienta como GLASS y por lo tanto en el módulo que nos atañe.

El e-learning nos ofrece herramientas muy eficaces para apoyar a los diferentes planes y programas educativos que se llevan a cabo en los centros, así como para crear redes profesionales. No es necesario salir de casa para conseguir dar una clase o al menos complementarla, poder solucionar todas las dudas y hasta hacer exámenes. Para ello sólo es necesario un PC conectado a Internet. Esto ayuda a que fácilmente, mediante el uso de sensores, se obtenga información de lo que la gente hace y mediante el análisis de la información tratar de mejorar la calidad de la enseñanza o de la herramienta.

Hay que tener cuidado en como se muestra esta información para no saturar a un estudiante, hay que saberle dirigir en el aprendizaje, remarcándole aquello que es más importante para que vaya adquiriendo la información de una manera eficaz. Es muy importante no desmotivarlo ni que caiga en la falsa sensación de tener todo controlado. La información tiene que llegar de una manera simple pero a su vez llegar a ser detallada si se quiere profundizar más. Una manera de conseguir todo esto es gracias al diseño adecuado de visualizaciones, siempre se ha dicho que una imagen vale más que mil palabras. Por ello las visualizaciones tienen que ser simples; con un rápido vistazo hay que ser capaz de ver lo relevante pero que gracias a su potencial ofrezca la posibilidad de poder indagar más en los datos. Además tienen que ser fáciles de comprender e intuitivas, y si la información que muestran va cambiando a lo largo del tiempo, es importante que muestre su evolución y que refleje el cambio; una buena visualización tiene que incitar a que la vuelvan a ver, si es interactiva que su atractivo incite a ser reutilizada, que no sea un simple gráfico que una vez visto lo puedas olvidar.

Pero dar con la herramienta perfecta es muy difícil por lo que ofrecer una herramienta escalable, personalizable, reutilizable y amigable garantiza en cierta medida la calidad. Si un módulo no cumple con las necesidades, siempre se puede desarrollar uno nuevo que si las cumpla. Mostrar una rica información de manera simple y con un número poco cargado de filtros ayuda a que el usuario reutilice la herramienta. Siempre es sabido que muchos filtros provocan rechazo debido a que dan la sensación de complejidad y por lo tanto no invitan a ser utilizados. Además, si ya se ofrece una configuración automáticamente personalizada se le quita trabajo al usuario y por lo tanto se garantiza su confianza.

Es necesario integrar todos los módulos de tal forma que no funcionen como herramientas independientes unas de otras. Con ella se consigue una aplicación global en la que no haya redundancia de información. Los procesos tienen que estar lo más automatizados posibles, con la posibilidad de elegir características por defecto para reducir el tiempo de interactuación entre el usuario y la aplicación. La información debe mostrarse de forma concisa y amigable y la interfaz gráfica debe ser lo más intuitiva posible. Sólo de esta manera se consiguen herramientas competentes y útiles que dan valor a la plataforma y permiten crear una herramienta con múltiples funcionalidades, todas ellas aplicables.

Las conclusiones no estarían completas si no se hace una autocrítica de lo que este proyecto ha supuesto al autor. Conociendo su situación, se puede pensar que tras desarrollar una herramienta como investigador del departamento, realizar un módulo para dicha herramienta no es un auténtico reto como el que sería haberlo realizado sin ninguna experiencia previa. Como desarrollador de la herramienta el módulo no ha sido la única tarea realizada. Debido a que GLASS se encuentra en fase de construcción, ha sido necesario realizar algunas modificaciones a sus características y funcionalidades para mejorar la interoperabilidad con los módulos, así como solucionar algunos Bugs o mejorar y definir nuevas pautas para futuros módulos. Además se han mejorado muchas de las funcionalidades que se pueden utilizar con los módulos para interactuar con GLASS, así el trabajo es más complejo de lo que pudiera parecer. Además, no mucho antes de empezar el módulo se hizo la portabilidad total a Mongo por lo que durante todo el desarrollo del módulo ha sido necesario adquirir conocimientos de la base de datos no relacional. El proyecto ha supuesto una mejora notable de las habilidades de programación y una ganancia de experiencia que ha servido de gran ayuda en el trabajo profesional compaginado.

Debido a que se trataba de un proyecto de desarrollo software, los conocimientos adquiridos de asignaturas externas al departamento han sido poco utilizados durante este segundo ciclo. Han servido de gran ayuda los conocimientos en matemática y estadística para solucionar los problemas del análisis del rendimiento. En cambio, del área de telemática si se han utilizado muchos conceptos relacionados con la programación, análisis, planificación, administración de servicios en Unix, utilización de programas de control de versiones como SVN, etc. Además ha sido un gran reto compaginar 8 horas de trabajo en una empresa como Ericsson, desarrollar un PFC como éste en, aproximadamente 3 o 4 meses, y aprobar las últimas asignaturas de la carrera.

6.2. Líneas futuras

Lo primero que puede venir a la mente a la hora de pensar en las posibles líneas del proyecto es, continuando con una de las motivaciones de este proyecto, seguir haciendo uso de la escalabilidad de GLASS y desarrollar un nuevo módulo.

Por un lado se podría hacer este módulo más configurable, para poder evaluar no sólo el rendimiento de los usuarios, sino también el de un grupo de trabajo o el de una clase en el caso que el curso estuviera formado por varias clases. Bajo mi punto de vista, esto debería ser planteado como otro módulo debido a que la línea seguida para éste era el no incluir apenas filtros que complicaran la visualización. Por lo que un nuevo, al

llevar contenido las bases de este, llevaría menos trabajo de desarrollo. Otra opción sería actualizarlo y permitir elegir, a la hora de su instalación el grado de complejidad deseado.

Por otro lado, se podría desarrollar un módulo que rompa totalmente con el concepto y que se base en otras necesidades, o incluso módulos no dedicados a visualizaciones de datos sino a funcionalidades de personalización de GLASS.

Si se quieren mejorar las funcionalidades ya existentes de GLASS hay que tener cuidado ya que esto puede hacer que los módulo ya desarrollados pasen a ser inservibles debido a la compatibilidad. A lo largo del documento, siguiendo esta línea, se han ido proponiendo mejoras. Se puede también mejorar, por ejemplo, el filtro temporal, tanto gráfica como funcionalmente y ofrecer más configuraciones para el desarrollador, para que a partir de algunas variables que se establezcan en su configuración, muestre conceptos totalmente diferentes.

Para terminar, otra línea sería, en la fase de instalación de una nueva base de datos, diseñar algún algoritmo capaz de transformar cualquier base de datos a la estructura que GLASS necesita para operar. Incluso un método que transforme bases de datos relacionales en no relacionales y que no sólo realice el cambio en el momento de la instalación sino periódicamente para que las visualizaciones estén en continuo cambio. Una posible solución sería definiendo, como nuevo campo de la base de datos, la estructura que tiene la base de datos. Así sería entendible a partir de algún algoritmo.

Capítulo 7

Glosario

API

Una API (del inglés Application Programming Interface - Interfaz de Programación de Aplicaciones) es el conjunto de funciones y procedimientos (o métodos si se refiere a programación orientada a objetos) que ofrece cierta librería para ser utilizados por otro software como una capa de abstracción.

Applets

Un applet es un componente de una aplicación que corre en el contexto de otro programa, por ejemplo un navegador Web. El applet debe correr en un contenedor, que proporciona un programa anfitrión, mediante un plugin, o en aplicaciones como teléfonos móviles que soportan el modelo de programación por applets.

Banner

Un banner es un formato publicitario en Internet. Esta forma de publicidad online consiste en incluir una pieza publicitaria dentro de una página web. Prácticamente en la totalidad de los casos, su objetivo es atraer tráfico hacia el sitio web del anunciante que paga por su inclusión.

Browser

Navegador Web.

Bug

Un web bug es una gráfica en una página web o en un mensaje de correo electrónico que se diseña para controlar quién lo lee. Su tamaño es inapreciable, pudiendo ser un píxel en formato GIF y transparente. Se representan como etiquetas HTML <IMG>.

Caché

En informática, una caché es un conjunto de datos duplicados de otros originales, con la particularidad de que los datos originales son costosos de acceder, normalmente en tiempo, respecto a la copia en la caché. Cuando se accede por primera vez a un dato, se hace una copia en la caché; los accesos siguientes se realizan a dicha copia, haciendo que el tiempo de acceso medio al dato sea menor.

Cadena de caracteres

En programación, una cadena de caracteres, palabra, ristra de caracteres o frase (*string* en inglés) es una secuencia ordenada de longitud arbitraria (aunque finita) de elementos que pertenecen a un cierto lenguaje formal o alfabeto, análogas a una frase o a una oración. En general, una cadena de caracteres es una sucesión de caracteres (letras, números u otros signos o símbolos).

Código fuente abierto

Cuando un producto de software se lanza con una licencia de Open Source (o "Código Fuente Abierto"), en principio ésta permite que todo programador que quiera pueda adaptar el software o mejorarla. Por lo tanto, el modelo de Open Source tiene importantes ventajas, tanto para los usuarios como para los fabricantes del software.

Cookie

Conjunto de caracteres que se almacenan en el disco duro o en la memoria temporal del ordenador de un usuario cuando accede a las páginas de determinados sitios web. Se utilizan para que el servidor accedido pueda conocer las preferencias del usuario al volver éste a conectarse. Dado que pueden ser un peligro para la intimidad de los usuarios, éstos deben saber que los navegadores permiten desactivarlos.

Copyright

Derecho que tiene un autor, incluido el autor de un programa informático, sobre todas y cada una de sus obras y que le permite decidir en qué condiciones han ser éstas reproducidas y distribuidas. Aunque este derecho es legalmente irrenunciable puede ser ejercido de forma tan restrictiva o tan generosa como el autor decida. El símbolo de este derecho es ©.

CVS

CVS es un Sistema Concurrente de Versiones. Normalmente se usa como una forma de almacenar el código fuente, ya que mantiene las versiones de todos los archivos de manera que no se pierda nada, y se registra el uso que hacen diferentes personas. También proporciona herramientas para combinar código si hay dos o más personas trabajando en el mismo archivo. Todo el código y todas las versiones se almacenan en un servidor central (en este caso, en Sourceforge).

Dashboard

El Dashboard es el panel o la pizarra donde se insertan todos los Widget. El concepto se utiliza para hacer referencia a un lugar donde aparecen múltiples vistas o *views*.

eLearnning

Se denomina aprendizaje electrónico (conocido también por el anglicismo e-learning) a la educación a distancia completamente virtualizada a través de los nuevos canales electrónicos (las nuevas redes de comunicación, en especial Internet), utilizando

para ello herramientas o aplicaciones de hipertexto (correo electrónico, páginas web, foros de discusión, mensajería instantánea, plataformas de formación que aúnan varios de los anteriores ejemplos de aplicaciones, etc.) como soporte de los procesos de enseñanza-aprendizaje. En un concepto más relacionado con lo semipresencial, también es llamado b-learning (blended learning). El b-learning es una modalidad que combina la educación a distancia y la educación presencial; retomando las ventajas de ambas modalidades y complementando el aprendizaje de los aprendices. También puede definirse como un sistema de comunicación masiva y bidireccional que sustituye la interacción personal en el aula del profesor y alumno, como medio preferente de enseñanza, por la acción sistemática y conjunta de diversos recursos didácticos y el apoyo de una organización tutorial, que proporcionan el aprendizaje autónomo de los estudiantes, además de reforzar la habilidad de la comunicación efectiva con los participantes a través de las plataformas usadas.

Enconding

Codificación, codificar. Es el proceso por el cual la información de una fuente es convertida en símbolos para ser comunicada. El proceso contrario es el decoding.

Iframe

Iframe (inline frame o marco incorporado) es un elemento HTML que permite insertar o incrustar un documento HTML dentro de un documento HTML principal. Insertar un iframe entre una sección o bloque es semejante a insertar un elemento *object*. Esto permite que se pueda insertar un documento HTML dentro de otro, alineado de acuerdo a sus límites.

Flash

Adobe FLASH (hasta 2005 Macromedia FLASH) o FLASH se refiere tanto al programa de edición multimedia como al reproductor de SWF (Shockwave FLASH) Adobe Flash Player, escrito y distribuido por Adobe, que utiliza gráficos vectoriales e imágenes ráster, sonido, código de programa, flujo de vídeo y audio bidireccional (el flujo de subida sólo está disponible si se usa conjuntamente con Macromedia Flash Communication Server). En sentido estricto, Flash es el entorno y Flash Player es el programa de máquina virtual utilizado para ejecutar los archivos generados con Flash.

Login/Logout

En el ámbito de seguridad informática, login o logon (en español ingresar o entrar) es el proceso mediante el cual se controla el acceso individual a un sistema informático mediante la identificación del usuario utilizando credenciales provistas por el usuario. Un usuario se puede login a un sistema para obtener acceso y se puede logout o logoff (en español salir o desconectar) cuando no se precisa mantener el acceso. Logout consiste en cerrar el acceso personal a un sistema informático, al cual anteriormente se había realizado el login. A su vez, existen castellanizaciones de estos términos como los sustantivos logueo/deslogueo y los verbos loguearse y desloguearse.

Módulo

Cada una de las partes en las que se divide una herramienta y que aporta una funcionalidad independiente al mismo. Las actividades de la plataforma son módulos, así como alguna de sus propiedades transversales, como las escalas y la copia de seguridad.

Paths

Una ruta es la forma de referenciar un archivo o directorio en un sistema de archivos y un sistema operativo.

Protocolo

Un conjunto de reglas y estándares que permiten a los ordenadores intercambiar información.

Root

Administrador.

Script

El guion o archivo de procesamiento por lotes es un programa usualmente simple, que generalmente se almacena en un archivo de texto plano. Los guiones son casi siempre interpretados, pero no todo programa interpretado es considerado un guion. El uso habitual de los guiones es realizar diversas tareas como combinar componentes, interactuar con el sistema operativo o con el usuario. Por este uso es frecuente que los shells sean a la vez intérpretes de éste tipo de programas.

Servlets

Los servlets son objetos que corren dentro del contexto de un contenedor de servlets (por ejemplo Tomcat) y extienden su funcionalidad. También podrían correr dentro de un servidor de aplicaciones (por ejemplo OC4J Oracle) que además de contenedor para servlet tendrá contenedor para objetos más avanzados como son los EJB (Tomcat sólo es un contenedor de servlets).

Shell

Intérprete de comandos o shell es un programa informático lector de líneas de texto que un usuario de un ordenador ha predefinido y este programa lo interpreta para un sistema operativo o lenguaje de programación.

Tag

Una etiqueta es una marca con tipo que delimita una región en los lenguajes basados en XML o HTML.

TIC

Acrónimo de Tecnologías de la Información y de la Comunicación.

Vista

Una vista o *view* es una gráfica o un conjunto de ellas del mismo tipo.

Visualización

Conjunto de vistas

Widget

Cuando se hace referencia a un Widget se refiere al resumen o vistas más pequeñas. Es un término que viene importado de la telefonía móvil.

Capítulo 8

Referencias

[1] Edward Segel y Jeffrey Heer, “Narrative visualization: telling stories with data”, Agosto 2010.

[2] Ben Schneiderman y Catherine Plaisant, “Designing the user interface”. 4ª edición. Capítulo 14.5, Information visualization, pág. 580-601.

[3] Real Academía Española, RAE, http://www.rae.es, 2012.

[4] Advertisement: Bus. United Technology. http://www.pewclimate.org/docUploads/UTC_fuel_cell%20ad_0.pdf.

[5] S. Carter and A. Cox. Paths to the Top of the Home Run Charts. The New York Times, 2007. http://www.nytimes.com/ref/sports/20070731_BONDS_GRAPHIC.html.

[6] Acadametrics House Price Index. Financial Times. http://www.acadametrics.co.uk/house_prices_June10.swf.

[7] V. Bevins and R. Birkett. UK Economic Data. Financial Times, 2010. http://www.ft.com/cms/s/0/bd7b628c-2068-11df-bf2d-00144feab49a.html.

[8] T. Laureyssens. Pedestrians Crossing the Street. Unknown, 2005. http://www.visualcomplexity.com/vc/project_details.cfm?id=255&index=7&domain=Pattern%20Recognition.

[9] J. Heer, A. Haeg, and M. Agrawala. Minnesota Employment Explorer. Minnesota Public Radio, 2007. http://minnesota.publicradio.org/projects/2008/07/16_minnesota_slowdown/.

[10] M. Bloch, S. Carter, A. Cox, and J. Ward. All of Inflation's Little Parts. The New York Times, 2008. http://www.nytimes.com/interactive/2008/05/03/business/20080403_SPENDING_GRAPHIC.html.

[11] E. Tufte. Politicians Abuse their Free-mailing Privileges before Elections. Visual Display of Quantitative Information, 2001.

[12] E. Tufte. John Snow's Chart of Deaths from Cholera. Visual Display of Quantitative Information, 2001.

[13] R. Birkett. GDP Moves by Sector. Financial Times, 2010. http://www.ft.com/cms/s/0/14cc2e70-04e6-11df-9a4f-0144feabdc0.html.

[14] P. Allen. Moscow Metro Bombs: Interactive Map. Guardian, 2010. http://www.guardian.co.uk/world/interactive/2010/mar/.

[15] M. Bloch, L. Byron, S. Carter, and A. Cox. The Ebb and Flow of Movies: Box Office Receipts 1986-2008. The New York Times, 2008. http://www.nytimes.com/interactive/2008/02/23/movies/20080223_REVENUE%_GRAPHIC.html.

[16] Advertisement: Helicopter. United Technology. http://farm4.static.flickr.com/3030/2572980233_0339ee260a.jpg.

[17] C. Oliver. The World Economy Turns the Corner. Guardian, 2010. http://www.guardian.co.uk/business/interactive/2010/jan/26/recession-gdp.

[18] S. Carter, A. Cox, and K. Quealy. The Jobless Rate for People Like You. The New York Times, 2009. http://www.nytimes.com/interactive/2009/11/06/business/economy/unemployment-lines.html.

[19] P. Allen. Formula One 2010: Driver's Rankings. Guardian, 2010. http://www.guardian.co.uk/sport/interactive/2010/feb/02/formula1-championship-points-2010.

[20] W. Andrews, A. Cypress, J. Kazil, T. Lindeman, and K. Yourish. The Climate Agenda. The Washington Post. http://www.washingtonpost.com/wp-srv/special/climate-change/global-emissions.html?ad=inw.

[21] C. Wilson. When Did Your County's Jobs Disappear? The Washington Post/Slate, 2009. http://www.slate.com/id/2216238/?ad=ins.

[22] N. V. Kelso, M. Lebling, K. Yourish, R. O'Neil,W. Andrews, J. Kazil, T. Lindeman, L. Shackelford, and P. Volpe. Analyzing Obama's Schedule. The Washington Post, 2010. http://projects.washingtonpost.com/potus-tracker/?ad=inw.

[23] Acadametrics House Price Index. Financial Times. http://www.acadametrics.co.uk/house_prices_June10.swf.

[24] C. Oliver and M. Wainwright. Lighting Up Hadrian's Wall. Guardian, 2010. http://www.guardian.co.uk/culture/interactive/2010/mar/12/hadrians-wall-lights-500-torches.

[25] P. Allen, J. Ridley, and C. Levene. Budget 2010: Reaction from around the UK. Guardian, 2010. http://www.guardian.co.uk/uk/interactive/2010/mar/24/budget-2010-case-studies-map.

[26] Human-computer interaction lab. Hierarchical clustering explorer for interactive exploration of multidimensional data. University of Marilan. http://www.cs.umd.edu/hcil/hce/

[27] GeoVISTA Studio. Multiform Bivariate Matrix With Map and Scatterplot. GeoVista Center, PENNSTATE. http://www.geovista.psu.edu/

[28] Rammana Rao and Stuart K. Card. Table Lens: merging Graphical and Symbolic representations in an interactive focus+context visualization for tabular information. Xerox Palo Alto Research Center. www.inxight.com.

[29] InfoZoom 8.0. http://www.infozoom.com/en.html

[30] Human-computer interaction lab. LifeLines for visualizing patient records. University of Marilan. http://www.cs.umd.edu/hcil/hce/

[31] Lecture Notes in Computer Science series, Proceedings for the Symposium on Graph Drawing, GD '98, Montreal, Canada, August 1998, to appear. http://www-graphics.stanford.edu/papers/h3draw/

[32] Wang, M., Peng, J., Cheng, B., Zhou, H., & Liu, J.. (2011). Knowledge Visualization for Self-Regulated Learning. Educational Technology & Society, 14 (3), 28–42.

[33] Bransford, J.D., Brown, A.L., & Cocking, R.R. (2000). How people learn: brain, mind, experience, and school, Washington, D.C.: National Academy Press.

[34] Zimmerman, B.J. (2000). Attaining self-regulation: A social cognitive perspective. In Boekaerts, M., Pintrich, P.R., & Zeidner, M. (Eds.), Handbook of self-regulation, New York: Elsevier, 13-39.

[35] Winne, P.H. (2001). Self-regulated learning viewed from models of information processing. In Zimmerman, B.J. & Schunk, D.H.

[36] (Eds.), Self-regulated learning and academic achievement: Theoretical perspectives, Mahwah, NJ: Lawrence Erlbaum 153-189.

[37] Reigeluth, C.M. (1999). The elaboration theory: Guidance for Scope and Sequences Decisions. In R. M. Reigeluth, (Ed.), Instructional-design theories and models: A new paradigm of instructional theory (Vol. II), Mahwah, NJ: Lawrence Erlbaum, 425-454.

[38] Simon, H.A. (1974). How Big is a Chunk? Science, 183, 482-488.

[39] Jonassen, D.H. (2000). Computers as Mindtools for Schools: Engaging Critical Thinking. Columbus, OH: Prentice-Hall.

[40] Jacobson, M. J., & Archodidou, A. (2000). The design of hypermedia tools for learning: Fostering conceptual change and transfer of complex scientific knowledge. The Journal of the Learning Sciences, 9(2), 149-199.

[41] Paige, J.M., & Simon, H.A. (1966). Cognitive processes in solving algebra and word problems. In B. Kleinmuntz (Ed.), Problem solving: Research, method and theory. New York, NY: Wiley.

[42] Wang, M., Jia, H., Sugumaran, V., Ran, W., & Liao, J. (in press). A Web-Based Learning System for Software Test Professionals. IEEE Transactions on Education. DOI: 10.1109/TE.2010.2051546.

[43] Jonassen, D.H. (2000). Computers as Mindtools for Schools: Engaging Critical Thinking. Columbus, OH: Prentice-Hall.

[44] Mayer, R.E. (2001). Multimedia learning. Cambridge: Cambridge University Press.

[45] Miller, S.M. & Miller K.L. (1999). Using instructional theory to facilitate communication in web-based courses. Educational Technology & Society, 2(3), 106-114.

[46] Newell, A. & Simon, H.A. (1972). Human problem solving. Englewood Cliffs, NJ: Prentice-Hall.

[47] Reynolds, R.E., Sinatra, G.M. & Jetton, T.L. (1996). View of Knowledge Acquisition and Representation: A continuum fromexperience centered to mind centered. Educational Psychologist, 31(2), 93-104.

[48] Jonassen, D.H. (1999). Designing constructivist learning environments. In C. M. Reigeluth (Ed.), Instructional design theories and models: A new paradigm for instructional theory, Mahwah, NH: Lawrence Erlbaum, 215-239.

[49] Bransford, J.D., Brown, A.L., & Cocking, R.R. (2000). How people learn: brain, mind, experience, and school, Washington, D.C.: National Academy Press.

[50] Reigeluth, C.M. (1999). The elaboration theory: Guidance for Scope and Sequences Decisions. In R. M. Reigeluth, (Ed.), Instructional-design theories and models: A new paradigm of instructional theory (Vol. II), Mahwah, NJ: Lawrence Erlbaum, 425-454.

[51] Hyerle, D. (2000). A field guide to using visual tools. Alexandria, VA: Association for Supervision and Curriculum Development.

[52] Costa, A.L. & Garmston, R.J. (2002). Cognitive coaching: a foundation for renaissance schools. Norwood, MA: Christopher-Gordon.

[53] Wikipedia, la enciclopedia libre. http://es.wikipedia.org

[54] George Siemens, ELEARNESPACE. Research Institute at Athabasca University. knoledgehttp://www.elearnspace.org/blog/2011/12/10/open-learning-analytics-a-proposal/

[55] José Manuel Martín, formación online y aprendizaje formal e informal, desde un punto de vista técnico. Learning analytics – Análisis del aprendizaje. http://www.josemanuelmartin.com/2013/01/learning-analytics-anlisis-del-aprendizaje/

[56] Talend, integration at any scale. http://www.talend.com/

[57] Blackboard analytics, improve decision making improve institutional performance. http://www.blackboard.com/platforms/analytics/overview.aspx

[58] Pentaho, powerful analytics made easy. http://community.pentaho.com/

[59] Lorenzo Alberton. Graphs in the database: SQL meets social networks. TECHPORTAL http://techportal.inviqa.com/2009/09/07/graphs-in-the-database-sql-meets-social-networks/

[60] Analytics and Recommendations. Module for Moodle. https://moodle.org/plugins/view.php?plugin=block_analytics_recommendations

[61] MongoDB. Non-relational database. http://www.mongodb.org/

[62] Highcharts JS, Interactive JavaScript charts for your web projects, 08 February 2013. http://www.highcharts.com/

[63] Richard Heyes. RGraph - Free HTML5/JavaScript charts and graphs 2008-2013. http://www.rgraph.net/

[64] The Apache Software Foundation. http://www.apache.org/

[65] PHP. http://php.net/

[66] MySQL. The world's most popular open source database. http://www.mysql.com/

Capítulo 9

Anexos

9.1. Apache

9.1.1. Introducción

Apache [64] es un servidor Web, es decir, un programa que implementa el protocolo HTTP (Hypertext Transfer Protocol, Protocolo de transferencia de Hipertexto) que se encarga de mantenerse a la espera de peticiones HTTP llevadas a cabo por un cliente que se suele conocer como navegador. El navegador realiza una petición al servidor y éste le responde con el contenido que el cliente solicita. Este contenido son todos los elementos multimedia que pueden representarse en una página Web.

Sus creadores son el Apache Group y son los que se dedican a mantenerlo y mejorarlo. Es un proyecto de código abierto, con una gran comunidad de usuarios detrás, de ahí que sea un servidor robusto, fiable y uno de los más utilizados en Internet.

9.1.2. Historia

Cuando comenzó su desarrollo en 1995 se basó inicialmente en el código del popular NCSA_HTTPd 1.3, pero más tarde fue reescrito por completo.

En febrero de 1995, el servidor Web más popular de Internet era un servidor de dominio público desarrollado en el NCSA (National Center for Supercomputing

Applications en la Universidad de Illinois). No obstante, al dejar Rob McCool (el principal desarrollador del servidor) el NCSA en 1994, la evolución de dicho programa había quedado prácticamente abandonada. El desarrollo pasó a manos de administradores que lo utilizaban y no querían que se quedara obsoleto; progresivamente introdujeron diversas mejoras. Un grupo de éstos, que usando el correo electrónico como herramienta básica de coordinación, se puso de acuerdo en poner en común estas mejoras (en forma de "parches" patches).

De ahí el nombre de Apache, pues consistía solamente en un conjunto de parches a aplicar al servidor de NCSA. Era, en inglés, a patchy server (un servidor "parcheado").

Posteriormente publicaron lo que sería la primera versión oficial del servidor Apache (la 0.6.2, de Abril de 1995).

En este momento el desarrollo de Apache siguió dos líneas paralelas. Por un lado, un grupo de los desarrolladores siguió trabajando sobre el Apache 0.6.2 para producir la serie 0.7, incorporando mejoras, etc. Un segundo grupo reescribió por completo el código, creando una nueva arquitectura modular.

Esta arquitectura modular fue la que triunfó, pues facilitaba el desarrollo en grupo y era muy fácil de instalar, de manera que el 1 de diciembre de 1995, apareció Apache 1.0, que incluía documentación y muchas mejoras en forma de módulos agregables.

En 1999 los miembros del Grupo Apache fundaron la Apache Software Foundation, que provee soporte legal y financiero al desarrollo del servidor Apache y al resto de proyectos que han surgido de éste.

Poco después, Apache sobrepasó al servidor de NCSA como el más usado en Internet, posición que ha mantenido hasta nuestros días, donde es el servidor HTTP del 70% de los sitios Web del mundo.

9.1.3. Características

Las características que destacan de Apache son:

- Corre en una multitud de Sistemas Operativos, lo que lo hace prácticamente universal.
- Apache es una tecnología gratuita de código abierto. Esto le proporciona transparencia a este software de manera que se puede ver fácilmente que es lo que se está instalando como servidor.
- Apache es un servidor altamente configurable de diseño modular. Resulta sencillo ampliar sus capacidades. Actualmente existen muchos módulos con funcionalidades para Apache. Otro punto importante es que cualquiera que posea alguna experiencia con la programación de C o Perl puede escribir un módulo para realizar una función determinada.
- Integra Perl. Perl se ha convertido en el lenguaje de programación por excelencia de la programación de scripts en CGI. Y posiblemente uno de los factores que ha influido para que ocurriese esto ha sido Apache. Por medio del módulo mod-Perl podría cargar scripts CGI programados en Perl en la memoria del ordenador y utilizarlos tantas veces como desee. Este proceso elimina los problemas de inicio asociados a los lenguajes de interpretación (entre los que se encuentra Perl).

- Apache es también uno de los primeros servidores que permite trabajar con nombres virtuales.
- Es totalmente configurable, tanto para cuestiones de seguridad como de administración, como puede ser la creación y gestión de logs, de este modo se puede tener un mayor control sobre lo que sucede en tu servidor.

Apache tiene otras muchas características, como la indexación de directorios, uso de sobrenombres con las carpetas, negociación de contenidos, informes configurables sobre los errores HTTP, ejecuciones SctUID o programas CGI, administración de recursos para los procesos emparentados, mapas de imagen para los servidores, reescritura de URL, corrección de URL y manuales online.

Además en la versión 2.0 se han incluido varias mejoras respecto a la 1.3:

- Apache puede ahora funcionar en multiprocesadores híbridos, o en modo multihilo (POSIX). Esto mejora la escalabilidad y el rendimiento para casi todas las configuraciones.
- Apache 2.0 es más rápido y más estable que antes en plataformas que no tengan base UNIX tales como BeOS, OS/2, Windows…
- En los sistemas donde IPv6 es apoyado por la biblioteca Runtime portable subyacente de Apache, Apache consigue los sockets que escuchan IPv6 por defecto
- Los mensajes de respuesta de error al browser ahora se proporcionan en varios idiomas, además pueden ser modificados para requisitos particulares por el administrador.
- Ahora utiliza utf-8 para todas las codificaciones de nombres de ficheros.

9.1.4. Módulos

La arquitectura del servidor Apache es modular. El servidor consta de un núcleo denominado core y el resto de la funcionalidad que podría considerarse básica para un servidor Web es provista por módulos. Algunos de estos son:

- mod_ssl - Comunicaciones Seguras vía TLS.
- mod_rewrite - Reescritura de direcciones servidas (generalmente utilizado para transformar páginas dinámicas como PHP en páginas estáticas HTML para así engañar a los navegantes o a los motores de búsqueda en cuanto a cómo fueron desarrolladas estas páginas).
- mod_dav - Soporte del protocolo WebDAV (RFC 2518).
- mod_deflate - Compresión transparente con el algoritmo deflate del contenido enviado al cliente.
- mod_auth_ldap - Permite autentificar usuarios con un servidor LDAP.
- mod_proxy_ajp - Conector para enlazar con el servidor Jakarta Tomcat de páginas dinámicas en Java (servlets y JSP).

El servidor de base puede ser extendido con la inclusión de módulos externos entre los cuales se encuentran:

- mod_perl - Páginas dinámicas en Perl.
- mod_php - Páginas dinámicas en PHP.
- mod_python - Páginas dinámicas en Python.
- mod_rexx - Páginas dinámicas en REXX y Object REXX.
- mod_ruby - Páginas dinámicas en Ruby.
- mod_aspdotnet - Páginas dinámicas en .NET_de_Microsoft (Módulo retirado).
- mod_security - Filtrado a nivel de aplicación, para seguridad.

9.1.5. Seguridad

La cuestión de la seguridad en un servidor Web es un aspecto muy importante, de ahí que sea un campo muy estudiado actualmente.

El servidor Apache tiene un buen expediente con respecto a la seguridad y una amplia comunidad que se dedica a mejorarla. Pero es inevitable que algunos problemas sean descubiertos después de lanzar una versión. Por esta razón, es crucial realizar siempre las actualizaciones del software. Se puede suscribir a la lista de avisos de Apache donde cualquier administrador puede mantenerse informado de nuevos lanzamientos y actualizaciones de seguridad.

La mayoría de las veces que un servidor está comprometido, no es debido a problemas en el código del servidor, el problema suele residir en el código adicional. El código del CGI, o el sistema operativo subyacente. Por lo tanto el administrador debe permanecer enterado de problemas y de actualizaciones de todo el software de su sistema.

Además de requerir contraseñas para acceder a cierto contenido del servidor, existe un problema aún mayor que es el de proteger el servidor contra ataques. Con un acceso inapropiado, se podría acceder o modificar archivos protegidos, o ejecutar comandos en el servidor. De ahí que este apartado esté centrado, en un principio, en la seguridad respecto a ataques y después se hablará de diferentes métodos de identificación y autenticación.

9.1.5.1. Archivo de configuración

El archivo de configuración del servidor Web Apache es httpd.conf. El archivo httpd.conf está bien comentado y es bastante autoexplicativo. La configuración predeterminada funciona para los ordenadores de la mayoría de los usuarios, así que probablemente no se necesita cambiar ninguna de las directivas en el fichero httpd.conf. Sin embargo, para realizar una configuración segura se debe entrar en otras opciones.

Los ficheros srm.conf, mime.typc y access.conf junto con el fichero httpd.conf se utilizaron anteriormente como ficheros de configuración de Apache, actualmente no se suelen utilizar y se incluyen dentro del httpd.

Si se necesita configurar Apache sólo se tiene que modificar el fichero httpd.conf y después recargar o bien apagar y arrancar el proceso del comando httpd.

Antes de modificar el fichero httpd.conf se debe copiar el fichero original dándole otro nombre cualquiera. Si se comete un error mientras se está modificando el fichero de configuración, no hay problema porque siempre dispone de una copia de ocurrido.

Si se comete un error y el servidor de Web no funciona correctamente, el primer sitio donde acudir es lo que se acaba de modificar en httpd.conf. También se puede consultar el fichero de conexión de error (error_log). Este puede ser difícil de interpretar, todo depende del nivel de experiencia. Las últimas entradas deberían marcar que es lo que ha pasado.

Las siguientes secciones dan una breve descripción de las directivas incluidas en el fichero httpd.conf, ordenadas según se encuentran en él.

9.1.5.2. Directivas

9.1.5.2.1. ServerType

El comando ServerType puede ser tanto inetd como standalone. El servidor tiene como comando predeterminado el ServerType standalone.

Dicho comando, el ServerType standalone, significa que el servidor arranca cuando se han llevado a cabo todas las conexiones. Por otro lado, el comando ServerType inetd quiere decir que arranca una nueva instancia cada vez que se produzca una conexión HTTP. Cada una de las instancias del servidor contiene la conexión y aunque acabe la conexión, sigue existiendo. La utilización del comando inetd, puede ser más segura porque no está siempre escuchando, pero puede no funcionar correctamente según el grupo Apache. Por ello, para evitar estos problemas es aconsejable que el comando ServerType sea standalone.

9.1.5.2.2. ServerRoot

El ServerRoot es el directorio principal donde se encuentran todos los ficheros del servidor.

9.1.5.2.3. LockFile

El comando LockFile configura el path al fichero de bloqueo utilizado por el servidor Apache cuando se compila con USE_FCNTL_SERIALIZED_ACCEPT o USE_FLOCK_SERIALIZED_ACCEPT. No se debe cambiar el valor predeterminado del comando LockFile.

9.1.5.2.4. PidFile

El comando PidFile nombra el archivo en el que el servidor graba su ID de proceso (pid).

9.1.5.2.5. ResourceConfig

La directiva ResourceConfig instruye al servidor a que lea el fichero ResourceConfig para buscar más directivas de configuración. La directiva ResourceConfig está comentada porque el servidor sólo usa httpd.conf para directivas de configuración.

9.1.5.2.6. AccessConfig

La directiva AccessConfig instruye al servidor a leer el fichero AccessConfig para buscar más directivas de configuración, tras haber leído el fichero ResourceConfig. Dicha

directiva está comentada porque el servidor sólo usa httpd.conf para directivas de configuración.

9.1.5.2.7. Timeout

El comando Timeout define, en segundos, el tiempo que el servidor esperará para recibir y enviar peticiones durante la comunicación. Específicamente, el comando Timeout define cuánto esperará el servidor para recibir peticiones GET, cuánto esperará para recibir paquetes TCP en una petición POST o PUT y cuánto esperará entre una ACK y otra respondiendo a paquetes TCP. El comando Timeout está ajustado a 300 segundos, que es el tiempo apropiado para la mayoría de las situaciones.

9.1.5.2.8. KeepAlive

El comando KeepAlive determina si el servidor permitirá varias conexiones a la vez (por ejemplo, más de una petición por conexión). KeepAlive puede usarse para impedir que un cliente consuma muchos recursos del servidor. El comando KeepAlive aparece en on por defecto, lo que significa que se permiten varias conexiones a la vez. Se puede poner en off para desactivarlas.

9.1.5.2.9. MaxKeepAliveRequests

Esta directiva se complementa con la anterior, establece el número máximo de peticiones permitidas por cada conexión que se produzca a la vez. El Grupo Apache recomienda un valor alto, lo que mejora el rendimiento, a costa del ancho de banda. El valor predeterminado del comando MaxKeepAliveRequests es de 100 que es suficiente en la mayoría de los casos.

9.1.5.2.10. KeepAliveTimeout

La directiva KeepAliveTimeout establece el número de segundos que el servidor esperará a la siguiente petición, tras haber dado servicio a una petición, antes de cerrar la conexión. Una vez recibida la petición, aplica la directiva Timeout en su lugar.

9.1.5.2.11. StartServers

StartServers establece cuántos procesos serán creados al arrancar. Ya que el servidor Web crea y elimina dinámicamente servidores según el tráfico, no se necesitará cambiar este parámetro. El servidor está configurado para desplegar ocho procesos al arrancar.

9.1.5.2.12. MaxClients

El comando MaxClients establece un límite al total de los procesos del servidor (es decir, clientes conectados simultáneamente) que se ejecutan a la vez. Se debe mantener el comando MaxClients a un valor alto (el valor por defecto es 150), porque no se permitirán nuevas conexiones una vez que se alcance el número máximo de clientes simultáneamente conectados. El valor del comando MaxClients no puede superar la cifra de 256 sin que se haya recompilado Apache. La principal razón de tener el parámetro MaxClients es evitar que un servidor que falle vuelva inestable al sistema operativo.

9.1.5.2.13. MaxRequestsPerChild

El comando MaxRequestsPerChild establece el número máximo de peticiones que cada proceso hijo procesa antes de morir. La principal razón para tener el comando

MaxRequestsPerChild es evitar que procesos de larga vida ocupen mucha memoria. El valor predeterminado de MaxRequestsPerChild para el servidor es de 100.

9.1.5.2.14. Listen

El comando Listen establece los puertos en los que se aceptan las peticiones entrantes. Apache está configurado para escuchar en el puerto 80 para comunicaciones no seguras y en el puerto 443 (ssl) para comunicaciones seguras.

Para puertos por debajo de 1024, el comando httpd deberá ser ejecutado como root. Para el puerto 1024 y superiores, el comando httpd puede ser ejecutado como si se fuera un usuario cualquiera.

El comando Listen también se puede usar para especificar direcciones IP específicas en las cuales aceptará conexiones el servidor, pudiendo así, por ejemplo, restringir el acceso de un laboratorio de la escuela.

9.1.5.2.15. BindAddress

BindAddress es un modo de especificar en qué direcciones IP escucha el servidor. Debería usarse la directiva Listen en su lugar si se necesita esta funcionalidad. El servidor no usa el comando BindAddress el cual ya aparece comentado en httpd.conf.

9.1.5.2.16. LoadModule

El comando LoadModule se usa para cargar los módulos, Dynamic Shared Object (DSO). El orden de los módulos es importante, es conveniente no modificarlo.

9.1.5.2.17. IfDefine

Las etiquetas <IfDefine> y </IfDefine> rodean a directivas de configuración que son aplicadas si la comprobación aplicada a la etiqueta <IfDefine> resulta verdadera.

Dicha comprobación aplicada a la etiqueta <IfDefine> es un nombre de un parámetro (por ejemplo, HAVE_PERL). Si el parámetro está definido, es decir, si se da como argumento al comando de arranque del servidor, entonces la comprobación es verdadera.

Por defecto, las etiquetas <IfDefine HAVE_SSL> rodean las etiquetas de la máquina virtual del servidor seguro. Las etiquetas <IfDefine HAVE_SSL> también rodean a las directivas LoadModule para ssl_module.

9.1.5.2.18. Port

Normalmente, el comando Port define el puerto en el que escucha el servidor. Apache, sin embargo, escucha en más de un puerto por defecto, ya que la directiva Listen también se usa. Cuando la directiva Listen está en uso, el servidor escucha en todos esos puertos.

El comando Port también se utiliza para especificar el número de puerto usado para crear el nombre canónico para el servidor.

9.1.5.2.19. User

La directiva User establece el userid usado por el servidor para responder a peticiones. El valor de User determina el acceso al servidor. Cualquier fichero al que no pueda acceder este usuario será también inaccesible al visitante de la Web. El comando predeterminado para User es Apache.

User debería tener privilegios de tal manera que sólo pudiera acceder a ficheros que se supone que todo el mundo puede ver. Al comando User no se le debería permitir ejecutar ningún código que no esté pensado para responder peticiones HTTP.

9.1.5.2.20. Group

El comando Group es similar a User. Group establece el grupo en el que el servidor responde a las peticiones. El valor predeterminado del comando Group también es Apache.

9.1.5.2.21. ServerAdmin

ServerAdmin debería ser la dirección de correo del administrador. Esta dirección de correo aparecerá en los mensajes de error generados por el servidor, de tal manera que los usuarios puedan comunicar errores enviando correo al administrador. El comando ServerAdmin ya se encuentra en la dirección admin@localhost.

9.1.5.2.22. ServerName

El comando ServerName puede usarse para establecer el nombre del servidor diferente al nombre real de máquina como por ejemplo, usar www.tudominio.com aunque el nombre real del servidor sea otro. ServerName debe ser un nombre "Domain Name Service" (DNS) válido que se tenga derecho a usar (no basta con inventar uno).

9.1.5.2.23. DocumentRoot

DocumentRoot es el directorio que contiene la mayoría de los archivos que se entregarán en respuesta a peticiones. El directorio predeterminado DocumentRoot es "C:/moodle/moodle".

9.1.5.2.24. Directory

Las etiquetas <Directory /path/to/directory> y </Directory> se utilizan para agrupar directivas de configuración que sólo se aplican a ese directorio y sus subdirectorios. Cualquier directiva aplicable a un directorio puede usarse en las etiquetas <Directory>. Las etiquetas <File> pueden aplicarse de la misma forma a un fichero específico.

Por defecto, se aplican parámetros muy restrictivos al directorio raíz, la utilización de las etiquetas Location permite al comando DocumentRoot tener parámetros menos rígidos para que el servidor sirva las peticiones HTTP.

9.1.5.2.25. Options

La directiva Options controla características del servidor que están disponibles en un directorio en particular.

Por defecto, en el directorio DocumentRoot, Options está configurado para incluir los comandos Indexes, Includes y FollowSymLinks. Indexes permite al servidor generar un listado de un directorio si no se especifica el DirectoryIndex (index.php, etc.). Includes implica que se permiten inclusiones en el servidor y el comando FollowSymLinks posibilita al servidor seguir enlaces simbólicos en ese directorio.

9.1.5.2.26. AllowOverride

AllowOverride establece qué directivas Options puede obviar un archivo .htaccess. Por defecto, tanto el directorio raíz como DocumentRoot están configurados para no permitir la prevalencia de .htaccess.

9.1.5.2.27. Order

Order simplemente controla el orden en que allow y deny se evaluan. El servidor está configurado para evaluar Allow antes que deny para el directorio DocumentRoot.

9.1.5.2.28. Allow

Allow especifica qué peticionario puede acceder un directorio dado. El peticionario puede ser all, un nombre de dominio, una dirección IP, una dirección IP parcial, un par red/máscara de red, etc. El directorio DocumentRoot está configurado para permitir peticiones de all (cualquiera).

9.1.5.2.29. Deny

Deny funciona como allow, pero específica a quién se niega el acceso. DocumentRoot no está configurado para rechazar peticiones de nadie.

9.1.5.2.30. DirectoryIndex

DirectoryIndex es la página por defecto que entrega el servidor cuando hay una petición de índice de un directorio especificado con una barra (/) al final del nombre del directorio.

Por ejemplo, cuando un usuario pide la página http://tudominio/estedirectorio/, recibe la página DirectoryIndex si existe, o un listado generado por el servidor. El valor para DirectoryIndex es index.php index.php4 index.php3 index.cgi index.pl index.html index.htm index.shtml index.phtml. El servidor intentará encontrar cualquiera de estos, y entregará el primero que encuentre. Si no encuentra ninguno y si Options Indexes se encuentra en el directorio, el servidor generará un listado, en formato HTML, de los subdirectorios y archivos del directorio.

9.1.5.2.31. AccessFileName

AccessFileName denomina el archivo que el servidor utilizará para controlar el acceso en cada directorio. Por defecto, el servidor utilizará .htaccess, si existe, para controlar el acceso en cada directorio.

Justo tras AccessFileName, el comando Files controla el acceso a cualquier archivo que empiece con .ht. Estas directivas niegan acceso a todo tipo de archivo .htaccess (u otros archivos que empiecen .ht) por razones de seguridad.

9.1.5.2.32. TypesConfig

TypesConfig denomina el fichero que establece la lista predeterminada de mapeado de tipos MIME (extensiones de ficheros a tipos de contenido). El fichero predeterminado TypesConfig es mime.types. En vez de modificar el mime.types, se recomienda añadir mapeados de tipos MIMEs con AddType.

9.1.5.2.33. DefaultType

DefaultType establece el contenido por defecto que el servidor utilizará para documentos cuyos tipos MIME no puedan ser determinados. El servidor predispone el texto para cualquier fichero con un tipo de contenido indeterminado.

9.1.5.2.34. IfModule

<IfModule> y </IfModule> envuelven a directivas que son condicionales. Las directivas contenidas dentro de IfModule son procesadas si se cumple una de las dos condiciones. Las directivas son procesadas si el módulo contenido en la etiqueta <IfModule> está compilado en el servidor Apache, o, si una "!" (exclamación) aparece antes del nombre.

El fichero magic.c está incluido en IfModule. El módulo mod_mime_magic puede compararse al comando UNIX file, que examina los primeros bytes de un fichero, y usa "números mágicos" y otros trucos para decidir el tipo MIME del fichero.

9.1.5.2.35. HostnameLookups

HostnameLookups puede aparecer en on o en off. Si el servidor permite la directiva HostnameLookups (poniéndolo en on), el servidor resolverá automáticamente la dirección IP de cada conexión que pida un documento del servidor. Resolver la dirección IP implica que el servidor hará una o más conexiones al DNS para averiguar qué nombre de máquina se corresponde con una dirección IP.

Generalmente, debería dejarse HostnameLookups en off porque las peticiones de DNS añaden carga al servidor y pueden ralentizarlo. Si el servidor tiene carga, los efectos de HostnameLookups pueden ser considerables.

9.1.5.2.36. ErrorLog

ErrorLog nombra el fichero donde se guardan los errores del servidor.

9.1.5.2.37. ServerSignature

El comando ServerSignature añade una línea que contiene la versión del servidor Apache y el ServerName de la máquina a los documentos generados por el servidor (por ejemplo, mensajes de error devueltos a clientes). ServerSignature suele aparecer en on. Se puede cambiar a off para no añadir nada, lo que quitará pistas a posibles atacantes o se puede cambiar a EMail. EMail añadirá una etiqueta HTML mailto:ServerAdmin a la línea de firma.

9.1.5.2.38. Alias

El comando Alias permite que haya directorios fuera del DocumentRoot a los que puede acceder el servidor. Cualquier URL que termine en un alias será automáticamente traducido por el recorrido del alias.

9.1.5.2.39. Redirect

Cuando se cambia una página de sitio, el comando Redirect se puede usar para pasar del viejo URL al nuevo URL.

9.1.5.2.40. HeaderName

La directiva HeaderName dicta el fichero (si existe dentro del directorio) que se antepondrá al comienzo de los listados de los directorios. Al igual que con ReadmeName, el servidor intentará incluirlo como documento HTML si es posible, o como texto.

9.1.5.2.41. LanguagePriority

La directiva LanguagePriority permite dar la prioridad a ciertos ficheros en distintos idiomas, que entrarán en vigor si el cliente no especifica la preferencia de idioma.

9.1.5.2.42. AddType

Use la directiva AddType para definir parejas de tipos MIME y sus extensiones. Por ejemplo, la siguiente línea AddType permite al servidor reconocer las extensiones .shtml (para la inclusión en el servidor):

AddType text/html .shtml

9.1.5.2.43. AddHandler

La directiva AddHandler mapea y amplia gestores específicos. Por ejemplo, el gestor cgi-script puede usarse para hacer que la extensión .cgi automáticamente sea manejada como un script CGI.

9.1.5.2.44. Action

La directiva Action permite especificar un par de tipos de contenido MIME y un script CGI, de tal forma que cuando se pida un fichero de este tipo, se ejecute un script en particular.

9.1.5.2.45. MetaDir

MetaDir especifica el nombre del directorio donde el servidor debería buscar los ficheros que contengan información meta (cabeceras extra de HTTP) que se deba incluir al entregar los documentos.

9.1.5.2.46. ErrorDocument

Por defecto, en caso de error, el servidor muestra un mensaje de error (generalmente críptico) para el cliente. En vez de usar esta opción ya predeterminada, puede usarse ErrorDocument para devolver un mensaje de error personalizado o redireccionar al cliente a un URL local o remoto. ErrorDocument simplemente asocia un código de respuesta HTTP con un mensaje o un URL que se devolverá al cliente. En Moodle está desactivado.

9.1.5.2.47. Location

Las etiquetas <Location> y </Location> permiten controlar el acceso específico a cada URL.

El primer uso de Location es configurar Options y proporcionar guías extra de configuración para DocumentRoot. Estas directivas de configuración, que se encuentran dentro de las etiquetas <Location "/"> y </Location>, son necesarias para permitir el acceso a documentos en DocumentRoot.

9.1.5.2.48. VirtualHost

<VirtualHost> y </VirtualHost> envuelven directivas de configuración que se aplican a máquinas virtuales. La mayoría de las directivas de configuración pueden usarse en etiquetas de máquina virtual, y sólo se aplicarán a esa máquina virtual.

9.1.5.2.49. SetEnvIf

La directiva de configuración de Apache SetEnvIf se usa para desactivar HTTP keepalive y permitir a SSL cerrar las conexiones sin avisar desde el cliente. Este parámetro es necesario para clientes que no cierran bien la conexión SSL.

9.1.5.2.50. Directivas de configuración SSL

Se han incluído las directivas SSL mediante un:

Include conf/extra/httpd-ssl.conf

Para permitir comunicaciones seguras Web usando las directivas SSL y TLS.

9.1.5.3. Medidas a tomar

Una vez vista la configuración de las directivas más importantes, se exponen unas cuantas medidas fáciles de implementar y que mejorarán mucho la seguridad del servidor:

9.1.5.3.1. Restringir permisos carpetas

Al igual que se restringen los permisos para el caso de comandos o modificaciones peligrosas que se puedan ejecutar en un S.O, se debe tener esta precaución con el servidor Apache. No sólo los mismos archivos de configuración deben estar protegidos contra escritura, también sus directorios y todos los padres de los directorios.

9.1.5.3.2. Desactiva la ejecución de CGI

Si es necesario la ejecución de CGI por algún motivo en concreto hay que desactivarlos, lo cual se consigue con las opciones de directiva dentro de la etiqueta directorio que tiene dos posibles valores none o ExecCGI.

Options -ExecCGI

9.1.5.3.3. Cambiar extensiones de archivo SSI

Permitir SSI para los archivos.html o.htm puede ser peligroso. Plantean los mismos riesgos que se asocian a los CGI en general. Usando el elemento cmd del exec, los archivos SSI-permitidos pueden llegar a ejecutar códigos indeseados. Los archivos SSI-permitidos deben tener una extensión separada, tal como .sh. Esto permite minimizar el riesgo.

9.1.5.3.4. Comprobar los registros

Comprobar los registros diarios, ahí se puede ver si se han realizado operaciones malignas en nuestro sistema.

9.1.5.3.5. 1.5.3.5. Restringir acceso por IP

Si se cuenta con un recurso al que deba solamente tener acceso alguna red, o IP en concreto se puede configurarlo en Apache.

9.1.5.3.6. Ocultar la versión y otra información delicada

Por defecto muchas instalaciones de Apache muestran el número de versión que está funcionando, el sistema operativo y un informe de módulos de Apache que están instalados en el servidor. Los usuarios maliciosos pueden utilizar esta información para atacar el servidor. Hay dos directivas que se necesitan corregir en el archivo httpd.conf:

```
ServerSignature Off
ServerTokens Prod
```

9.1.5.3.7. Apache debe funcionar bajo su propia cuenta y grupo de usuario

Algunas versiones de Apache corren bajo el usuario nobody, esto compromete mucho su seguridad por lo tanto se debe hacer lo siguiente:

```
User apache
Group apache
```

9.1.5.3.8. Deshabilitar cualquier módulo innecesario

Apache viene por defecto instalado con una serie de módulos. Se debe echar un vistazo a la documentación de Apache y ver para que sirven cada uno de ellos, y de esta manera darse cuenta de que hay algunos que no son útiles en el servidor.

Se debe buscar en httpd.conf las líneas que contengan LoadModule. Para deshabilitar el módulo se debe agregar un # al principio de la línea, para que de esta forma pase a ser un comentario.

Aquí están algunos módulos que se instalan por defecto pero a menudo no son necesarios: mod_imap, mod_include, mod_info, mod_userdir, mod_status, mod_autoindex.

9.1.5.3.9. Desactivar las opciones para explorar directorios

Esta desactivación se puede llevar a cabo con las opciones de directiva que dentro de la etiqueta directorio tiene dos posibles valores none o indexes.

Options –Indexes

9.1.5.3.10. No permitir que apache siga enlaces simbólicos

De nuevo se configura con las opciones de directiva dentro de la etiqueta directorio tiene dos posibles valores none o FollowSymLinks.

Options –FollowSymLinks

9.1.5.3.11. Desactivar la ayuda para los archivos .htaccess

Esto está ya hecho con la directiva AllowOverride, se debe cambiar a none.

AllowOverride None

Otra opción interesante sería bloquear la descarga de todos los archivos que comiencen con .ht por ejemplo, se haría de la siguiente manera:

```
AccessFileName .httpdoverride
Order allow,deny
Deny from all
Satisfy All
```

9.1.5.3.12. Limitar el tamaño máximo de peticiones

Apache tiene varias directivas que permiten limitar el tamaño de una petición, lo cual puede ser muy útil.

9.1.5.3.13. No confiar en campos ocultos

Los campos ocultos se denominan así porque no se visualizan en la pantalla del navegador, pero sí se ven si se lista el código fuente en HTML de la página. Por lo tanto, cualquiera puede cargar la página en su disco local y editarla, modificando los valores de los campos ocultos. En consecuencia, nunca se deben utilizar para contener información confidencial ni confiar en ellos para almacenar información sensible (como precio de un producto). El servidor deberá contrastar la información recibida procedente de los campos ocultos.

9.1.5.3.14. Otras medidas, identificación y autenticación

A la hora de aplicar reglas de seguridad en cualquier sistema, las dos preguntas que se le deben hacer a cualquier usuario son:

- ¿Eres realmente quien dices ser?
- ¿Tienes permiso para estar aquí?

Estos pasos reciben el nombre de identificación y autorización. Llevado a un caso de la vida real, sería como una azafata que comprueba la foto del carnet del viajero (identificación) y su billete (autorización) antes de permitirle el acceso al avión.

El fallo de cualquiera de las dos operaciones muestra el mismo tipo de error, que siempre corresponde al código 401 (de no autorización), incluso si el fallo se produjo en la identificación. Esto es así para evitar que posibles atacantes conozcan cuando tienen credenciales válidas.

9.1.5.4. Identificación

La identificación puede ser sencilla (basic) o severa (digest). La primera se basa en que las credenciales (nombre de usuario y contraseña) del usuario sean correctas (que en realidad puede haber tomado del usuario real), mientras que la segunda se basa en atributos sobre los que el usuario tiene escaso control y que son los mismos para todas las solicitudes, tal como la dirección IP del sistema.

El método de identificación Basic es bastante primitivo e inseguro. Se basa en codificar las credenciales del usuario mediante un algoritmo reversible (codificación 64) y transmitir el resultado en forma de texto como parte del encabezamiento de la solicitud.

Cualquiera que intercepte la transmisión puede hacerse con dichas credenciales para un uso ilegítimo, por lo que este método debe emplearse sólo cuando los documentos protegidos no son de contenido demasiado privado o cuando no exista otra alternativa.

Por otro lado, el método de identificación Digest es más seguro y hace más difícil el posible robo de contraseñas. La diferencia principal es que ni el nombre de usuario ni la contraseña viajan por la red en forma de texto.

Para utilizar el método de identificación básico tan sólo hay que almacenar la combinación de nombre de usuario y contraseña e indicar al servidor dónde debe buscarla. La contraseña no tiene que estar encriptada obligatoriamente. Se pueden aplicar las mismas credenciales a cualquier reino del servidor, o copiarse directamente en otro servidor distinto. Pueden almacenarse también en una gran variedad de bases de datos ya que existen muchos módulos para almacenar credenciales básicas en archivos de texto, archivos GDBM, BBDD MySQL, o directorios LDAP entre otros.

La desventaja que tiene este segundo método es que el proceso de configuración es mucho más complejo. Por un lado, no es posible pasar credenciales de un reino a otro, ya que al generarlas, se especifica el reino en el que son válidas. Por otro lado, el único modo de almacenamiento soportado por el paquete de Apache es a través de archivos de texto, por lo que si queremos guardar credenciales en un directorio LDAP o en una base de datos Oracle, tendremos que emplear módulos no estándares de Apache o crearles nosotros mismos.

Además de implicar un proceso de configuración más complejo, el método de identificación Digest no está muy extendido, y, aunque Apache lo soporta, hay muchos buscadores o clientes Web que no. La conclusión es que se debe aplicar el primer método, ya que será el único disponible para todos los usuarios.

9.1.5.4.1. Identificación básica

Una forma para llevar a cabo una política de identificación es usar la información de las cuentas del sistema para identificar a través de la Web. Para ello no tenemos más que introducir las siguientes líneas en el archivo de configuración.

```
<Directory "Aquí el directorio casa">
  AuthType Basic
  AuthName HomeDir
  AuthUserFile Archivo donde se almacenan los usuarios
  Require valid-user
  Satisfy All
</Directory>
```

Este proceso lo llevará a cabo el módulo mod_auth. La sintaxis y el orden de los campos de un registro de credenciales es bastante simple y parecida a la de la mayoría de los sistemas existentes. Emplea un archivo de texto sencillo, generalmente a este archivo se le denomina .htpasswd, donde cada línea comienza con un nombre de usuario y una contraseña, que puede además contener campos adicionales, delimitados por puntos, de la manera siguiente:

```
Alberto:$apr1$GLWef/........$8hOXRasGFDsJY5
```

Esta técnica no funciona si se están utilizando contraseñas ocultas, del tipo "shadow". Las contraseñas se almacenan en un archivo sólo accesible desde la raíz, por lo tanto no será accesible para Apache.

Por último mencionar que no es buena idea utilizar la información de cuenta de sistema como modo de identificación, a menos que el sitio esté asegurado mediante SSL. Por un lado, cualquier intruso que consiga obtener las credenciales de uno de los usuarios podrá acceder no sólo a los archivos protegidos, sino también registrarse dentro del sistema y ocasionar daños considerables.

9.1.5.4.2. Gestionar archivos .htpasswd

Los archivos de contraseñas pueden crearse a mano o con la utilidad htpasswd.

Ejecutando: htpasswd –c user.pass alberto

Se crea un archivo de contraseñas llamado user.pass. Este contendrá el nombre de usuario y la contraseña del cliente alberto, con la sintaxis especificada anteriormente. Al ejecutar este comando el usuario tendrá que introducir su nombre y se le pedirá que escriba la contraseña dos veces para confirmarla.

La marca –c crea un archivo nuevo, incluso si ya existía un archivo con el mismo nombre, por lo que hay que asegurarse de utilizar esta marca sólo la primera vez.

El archivo de contraseñas se creará mediante el algoritmo de encriptación que existe por defecto en cada sistema, en caso de sistemas Windows este es MD5.

Para eliminar usuario se ejecutará el mismo comando, pero con la marca –D o también se podrá hacer editando el archivo y eliminando la línea correspondiente.

Otra opción es utilizar el módulo Perl Apache::Htpasswd, para que se encargue de gestionar el archivo automáticamente. Proporciona una interfaz Perl a los archivos de contraseñas, lo que nos permite modificarlos desde programas CGI o mecanismos similares, utilizando sólo unas sencillas líneas de código Perl, se muestra en este ejemplo.

```
Use Apache::Htpasswd;
$pass = new Apache::Htpasswd("Ruta del fichero de contraseñas") or die "Couldn`t open password file.";
#Añade una entrada
$pass->htpasswd("waldo", "alberto");
#Elimina la entrada
$pass->htDelete("alberto");
```

9.1.5.4.3. Identificación severa

El método Digest está gestionado por el módulo mod_auth_digest, y utiliza también un archivo de contraseñas y el algoritmo MD5 para la encriptación. Para crear un archivo de credenciales ejecutar:

```
Htdigest –c "By invitation only" alberto ("Solo con invitación")
```

La sintaxis del comando es muy similar a la de htpasswd, excepto en que debemos especificar la identificación para la que se va a emplear una contraseña determinada. El archivo resultante contiene una línea para cada usuario y tiene este aspecto:

```
   alberto:By                                    invitation
only:23bndjfgk53poonmgkolsjhhfo5567l
```

La marca –c también crea un nuevo archivo, pero Htdigest no cuenta con ninguna de las otras opciones que tiene htpasswd.

Como se ha explicado anteriormente este método es muy complejo, pero puede ser muy útil en determinados momentos, como por ejemplo para restringir accesos a un determinado archivo sólo a ciertas direcciones IP. Para esto utilizamos una sintaxis mezclada de los dos métodos, gracias a la directiva Satisfy:

```
   <Directory      Directorio      a      restringir      ej:
/www/htdocs/sensitive>
     #Pone en práctica todas las restricciones
     Satisfy All
     #Requiere contraseña
     AuthType Basic
     AuthName Sensitive
     AuthUserFile   ruta   del   archivo   de   usuarios   ej:
/www/passwords/users
     Require valid-user
     #Requiere el acceso desde una red concreta
     Order deny,allow
     Deny from all
     Allow from 192.168.1
   </Directory>
```

Por la manera en la que la autentificación está especificada, el nombre de usuario y contraseña debe ser verificado cada vez que se solicita un documento del servidor. Incluso si está recargando la misma página, y por cada imagen de la página (si vienen de un directorio protegido). Esto retrasa un poco las cosas. El retraso es proporcional al tamaño del archivo de contraseña, porque se tiene que abrir ese archivo, y recorrer la lista de usuarios hasta que encuentre su nombre. Y eso se tiene que hacer cada vez que se cargue la página.

Una consecuencia de esto es que hay un límite práctico de cuántos usuarios se pueden colocar en un archivo de contraseñas. Este límite variará dependiendo del rendimiento del equipo servidor en particular, pero se puede observar una disminución en rendimiento una vez que se insertan unos cientos de entradas, de ahí que se consideren métodos distintos de identificación.

En el caso de nuestro sistema aunque no disponemos de SSL, se puede configurar para que la página de acceso lo utilice, por lo que el método básico es suficiente. En realidad se utilizan los datos almacenados en la base de datos Mysql para proceder a la identificación y autenticación de cada usuario. Este proceso además es más rápido y provee un nivel de seguridad suficiente.

Como última nota sobre el archivo de contraseñas, es recomendable almacenarlo en algún lugar donde no sea accesible a través de la Web. De lo contrario se corre el riesgo

de que alguien pueda descargarlo y aplicarle algún algoritmo para obtener las contraseñas que en él se encuentran.

9.1.6. Comparación y estadísticas

Para finalizar se realiza un breve repaso de los dos servidores más utilizados, ISS y Apache, para mostrar sus características más importantes y justificar la elección de Apache como servidor.

Ventajas:

- Apache
    - Fácil de usar.
    - ASP preparado en la instalación por defecto.
    - .Soporte ODBC integrado.
    - Configuración gráfica y en línea de comandos.
- Apache
    - Código fuente disponible.
    - Existen versiones virtualmente para cualquier sistema operativo.
    - Excelente integración con PHP y MySQL.
    - .Es el servidor Web por excelencia en Internet.

Inconvenientes:

- ISS
    - Multitud de nuevos fallos de seguridad.
    - La mayoría de funcionalidad extra se debe comprar por separado.
    - Sólo funciona en Windows NT/2000.
- Apache
    - No existe aún configuración gráfica oficial.
    - Curva larga de aprendizaje para sacarle el máximo partido.

Elegimos Apache porque es un servidor libre, muy utilizado y, por lo tanto, exhaustivamente probado y porque cumple todos los requisitos a nivel de fiabilidad y seguridad requeridos. Además está perfectamente integrado con PHP y Mysql que son las otras dos tecnologías que se van a utilizar.

9.2. Seguridad PHP

9.2.1. Introducción

PHP [65] es un poderoso lenguaje e intérprete, ya sea incluido como parte de un servidor Web en forma de módulo o ejecutado como un binario CGI separado, es capaz de acceder a archivos, ejecutar comandos y abrir conexiones de red en el servidor. Estas propiedades hacen que cualquier cosa que sea ejecutada en un servidor Web sea insegura por naturaleza. PHP está diseñado específicamente para ser un lenguaje más seguro para escribir programas CGI que Perl o C, y con la selección correcta de opciones de configuración en tiempos de compilación y ejecución, y siguiendo algunas prácticas correctas de programación, PHP le puede dar la combinación precisa de libertad y seguridad que usted necesita.

Ya que hay muchas maneras de utilizar PHP, existen varias opciones de configuración diseñadas para controlar su comportamiento. Un amplio rango de opciones le garantizan que pueda usar PHP para muchos propósitos distintos, pero también quiere decir que hay combinaciones de estas opciones y configuraciones de servidor que pueden resultar en un entorno inseguro.

El nivel de flexibilidad en la configuración de PHP se compara quizás sólo con su flexibilidad de desarrollo. PHP puede ser usado para escribir aplicaciones completas de servidor, con todo el poder de un usuario de un intérprete de comandos, o puede ser usado para inclusiones simples del lado del servidor con muy poco riesgo en un entorno minuciosamente controlado. Cómo construir ese entorno y su nivel de seguridad depende básicamente del desarrollador PHP.

Este segundo capítulo comienza con algunas recomendaciones generales de seguridad, explica las diferentes combinaciones de opciones de configuración y las situaciones en las que pueden ser usadas con seguridad, y describe diferentes consideraciones a la hora de programar para diferentes niveles de seguridad.

9.2.2. Consideraciones generales

Un sistema completamente seguro es prácticamente un imposible, de modo que el enfoque usado con mayor frecuencia en la profesión de seguridad es uno que busque el balance adecuado entre riesgo y funcionalidad. Si cada variable que enviemos requiriere de dos formas de validación biométrica (como rastreo de retinas y análisis dactilar), contaríamos con un nivel extremadamente alto de confiabilidad. También implicaría que llenar los datos de un formulario razonablemente complejo podría tomar media hora, cosa que podría incentivar a los usuarios a buscar métodos para esquivar los mecanismos de seguridad.

La mejor seguridad, con frecuencia, es lo suficientemente razonable como para suplir los requerimientos dados sin prevenir que el usuario realice su labor de forma natural, y sin sobrecargar al autor del código con una complejidad excesiva. De hecho, algunos ataques de seguridad son simples recursos que aprovechan las vulnerabilidades de este tipo de seguridad sobrecargada, que tiende a erosionarse con el tiempo.

Una frase que vale la pena recordar: Un sistema es apenas tan bueno como el eslabón más débil de una cadena. Si todas las transacciones son registradas copiosamente basándose en la fecha/hora, ubicación, tipo de transacción, etc. pero la verificación del usuario se realiza únicamente mediante una cookie sencilla, la validez de atar a los usuarios al registro de transacciones es mermada severamente.

Al realizar pruebas, hay que tener en mente que no seremos capaces de probar todas las diferentes posibilidades, incluso para las páginas más simples. Los datos de entrada que se pueden esperar en las aplicaciones no necesariamente tendrán relación alguna con el tipo de información que podría ingresar un empleado disgustado, un cracker con meses de tiempo entre sus manos, o un gato doméstico caminando sobre el teclado. Es por esto que es mejor observar el código desde una perspectiva lógica, para determinar en dónde podrían introducirse datos inesperados, y luego hacer un seguimiento de cómo esta información es modificada, reducida o amplificada.

Internet está repleto de personas que tratan de crearse fama al romper la seguridad de su código, bloquear su sitio, publicar contenido inapropiado, y haciendo que sus días sean más interesantes. No importa si se administra un sitio pequeño o grande, somos un objetivo por el simple hecho de estar en línea, por tener un servidor al cual es posible conectarse. Muchas aplicaciones de cracking no hacen distinciones por tamaños, simplemente recorren bloques masivos de direcciones IP en busca de víctimas. El fin de este documento es no convertirnos en una.

9.2.3. Instalación como un binario CGI

9.2.3.1. Posibles ataques

El uso de PHP como un binario CGI es una opción para el tipo de situaciones en las que por alguna razón no se desea integrar PHP como módulo de algún software de servidor web (como Apache), o en donde se espera usar PHP con diferentes tipos de capas que envuelven el entorno CGI para crear ambientes chroot y setuid seguros para la ejecución de scripts. Esta configuración usualmente involucra la instalación de un binario ejecutable del intérprete PHP en el directorio cgi-bin del servidor Web. El aviso de seguridad de CERT CA-96.11 recomienda que se evite la colocación de cualquier intérprete bajo cgi-bin. Incluso si el binario PHP puede ser usado como un intérprete independiente, PHP está diseñado para prevenir el tipo de ataques que esta configuración hace posible:

Acceso a archivos del sistema: http://mi.servidor/cgi-bin/php?/etc/passwd. La información del query en una URL, la cual viene después del signo de interrogación (?), es pasada como argumentos de línea de comandos al intérprete por la interfaz CGI. Usualmente los intérpretes abren y ejecutan el archivo especificado como primer argumento de la línea de comandos. Cuando es invocado como un binario CGI, PHP se rehúsa a interpretar los argumentos de la línea de comandos.

Acceso a cualquier documento web en el servidor: http://mi.servidor/cgi-bin/php/zona_secreta/doc.html. El segmento de la URL que sigue al nombre del binario de PHP, que contiene la información sobre la ruta /zona_secreta/doc.html es usada convencionalmente para especificar el nombre de un archivo que ha de ser abierto e interpretado por el programa CGI. Usualmente, algunas directivas de configuración del servidor Web (Apache: Action) son usadas para redireccionar peticiones de documentos

como http://mi.servidor/zona_secreta/script.php al intérprete de PHP. Bajo este modelo, el servidor Web revisa primero los permisos de acceso al directorio /zona_secreta, y después de eso crea la petición de redireccionamiento a http://mi.servidor/cgi-bin/php/zona_secreta/script.php. Desafortunadamente, si la petición se hace originalmente en esta forma, no se realizan chequeos de acceso por parte del servidor Web para el archivo /zona_secreta/script.php, únicamente para el archivo /cgi-bin/php. De este modo, cualquier usuario capaz de acceder a /cgi-bin/php es capaz también de acceder a cualquier documento protegido en el servidor Web. En PHP, la configuración de tiempo de compilación --enable-force-cgi-redirect y las directivas de configuración en tiempo de ejecución doc_root y user_dir pueden ser usadas para prevenir este tipo de ataques, si el árbol de documentos del servidor llegara a tener directorio alguno con restricciones de acceso. A continuación se explica detalladamente las diferentes combinaciones:

9.2.3.1.1. Caso 1: sólo se sirven archivos públicos

Si el servidor no tiene contenido alguno que no esté restringido por contraseñas o control de acceso basado en direcciones IP, no hay ninguna necesidad de recurrir a estas opciones de configuración. Si el servidor Web no le permite hacer redireccionamientos, o el servidor no tiene una forma de comunicarle al binario PHP que la petición de redireccionamiento es segura, puede especificar la opción --enable-force-cgi-redirect en el script de configuración. Aun así debe asegurarse de que sus scripts PHP no dependan de alguna forma especial de hacer llamados al script, ya sea directamente mediante http://mi.servidor/cgi-bin/php/dir/script.php ni por la redirección http://mi.servidor/dir/script.php.

Los redireccionamientos pueden ser configurados en Apache mediante el uso de directivas AddHandler y Action.

9.2.3.1.2. Caso 2: uso de --enable-force-cgi-redirect

Esta opción en tiempo de compilación previene que cualquier persona haga llamadas a PHP directamente mediante una URL como http://mi.servidor/cgi-bin/php/directorio_secreto/script.php. En lugar de esto, PHP analizará documentos de esta forma únicamente si han pasado por una regla de redirección del servidor Web.

Por lo general, el redireccionamiento en la configuración de Apache es realizado con alguna de las siguientes directivas:

```
Action php-script /cgi-bin/php
AddHandler php-script .php
```

Esta opción ha sido probada únicamente con el servidor Web Apache, y depende de que Apache defina la variable de entorno no-estándar REDIRECT_STATUS a la hora de gestionar peticiones redirigidas. Si un servidor Web no dispone de modo alguno de comunicar si la petición es directa o redirigida, no se puede usar esta opción y se debe recurrir a alguna de las otras formas documentadas aquí de ejecutar la versión CGI.

9.2.3.1.3. Caso 3: configuración de doc_root o user_dir

Incluir contenido activo en los directorios de documentos del servidor Web, como scripts y ejecutables, es considerada en ocasiones una práctica insegura. Si, por algún fallo de configuración, los scripts no llegaran a ser ejecutados sino desplegados como documentos HTML normales, esto podría resultar en la revelación de información crítica

como trabajos cubiertos por normas de propiedad intelectual o datos de seguridad como contraseñas. Por lo tanto muchos administradores de sistemas preferirán la configuración de otra estructura de directorios para los scripts que sean asequibles únicamente a través del CGI PHP, y por lo tanto deben ser interpretados siempre y no desplegados directamente.

Así mismo, si el método para asegurarse de que las peticiones no son redireccionadas, tal y como describimos en la sección anterior, no está disponible, es necesario entonces configurar un directorio raíz (doc_root) de scripts que sea diferente al directorio raíz de documentos Web.

Se puede definir el directorio raíz para scripts de PHP mediante la directiva de configuración doc_root en el archivo de configuración, o podemos darle un valor a la variable de entorno PHP_DOCUMENT_ROOT. Si ésta está definida, la versión CGI de PHP construirá siempre el nombre del archivo a abrir con este doc_root y la información de la ruta dada en la petición, de modo que puede estar seguro de que ningún script será ejecutado por fuera de este directorio (excepto por aquellos indicados en user_dir, como veremos a continuación).

Otra opción que puede ser usada en este caso es user_dir. Cuando user_dir no está definida, lo único que controla la apertura de archivos es doc_root. Abrir una URL como http://mi.servidor/~usuario/doc.php no resulta en la apertura de un archivo bajo el directorio personal del usuario, sino de un archivo llamado ~usuario/doc.php bajo la ruta doc_root.

Si user_dir está definido como, por ejemplo, public_php, una petición como http://mi.servidor/~usuario/doc.php abrirá un archivo llamado doc.php bajo el directorio con el nombre public_php ubicado en el directorio personal del usuario. Si el directorio personal del usuario es /home/usuario, el archivo ejecutado es /home/usuario/public_php/doc.php.

La expansión del valor de user_dir ocurre independientemente del parámetro doc_root, de modo que es posible controlar el directorio raíz de los documentos y el acceso a los directorios de los usuarios en forma separada.

9.2.3.1.4. Caso 4: intérprete PHP por fuera del árbol Web

Una opción bastante segura es colocar el intérprete binario de PHP en alguna parte por fuera del árbol de archivos Web. En /usr/local/bin, por ejemplo. El único inconveniente real con esta alternativa es que ahora se tendrá que colocar la siguiente línea:

```
#!/usr/local/bin/php
```

Al comienzo de cualquier archivo que contenga etiquetas PHP. También se tiene que hacer cada archivo ejecutable. Esto quiere decir que se debe tratar exactamente igual a como trataría cualquier otro script CGI escrito en Perl o sh o cualquier otro lenguaje de scripting común que usara el mecanismo de escape-shell #! para el lanzamiento del intérprete.

Para lograr que PHP gestione correctamente la información de PATH_INFO y PATH_TRANSLATED con este tipo de configuración, el intérprete PHP debe haber sido compilado con la opción de configuración --enable-discard-path.

9.2.4. Instalación como módulo de Apache

Cuando PHP es utilizado como un módulo de Apache, hereda los permisos del usuario de Apache (generalmente los del usuario "nobody"). Este hecho representa varios impactos sobre la seguridad y las autorizaciones. Por ejemplo, si se está haciendo uso de PHP para acceder a una base de datos, a menos que tal base de datos disponga de un control de acceso propio, se tendrá que hacer que la base de datos sea asequible por el usuario "nobody". Esto quiere decir que un script malicioso podría tener acceso y modificar la base de datos, incluso sin un nombre de usuario y contraseña. Es completamente posible que un archivador automatizado de documentos Web pudiera toparse con la página Web de administración de una base de datos, y eliminar todas sus bases de datos. Podemos protegernos de este tipo de situaciones mediante mecanismos de autorización de Apache, o podemos diseñar un modelo propio de acceso usando LDAP, archivos .htaccess, etc. e incluir ese código como parte de sus scripts PHP.

Con frecuencia, una vez la seguridad se ha establecido en un punto en donde el usuario de PHP (en este caso, el usuario de apache) tiene asociada una muy leve capacidad de riesgo, se descubre que PHP se encuentra ahora imposibilitado de escribir archivos en los directorios de los usuarios. O quizás se le haya desprovisto de la capacidad de acceder o modificar bases de datos. Se ha prevenido exitosamente que pudiera escribir tanto archivos buenos como malos, o que pudiera realizar transacciones buenas o malas en la base de datos.

Un error de seguridad cometido con frecuencia en este punto es darle permisos de administrador (root) a apache, o incrementar las habilidades del usuario de apache de alguna otra forma.

Escalar los permisos del usuario de Apache hasta el nivel de administrador es extremadamente peligroso y puede comprometer al sistema entero, así que el uso de entornos sudo, chroot, o cualquier otro mecanismo que sea ejecutado como root no debería ser una opción viable para aquellos que no son profesionales en seguridad.

Existen otras soluciones más simples. Mediante el uso de open_basedir se pueden controlar y restringir aquellos directorios que podrían ser usados por PHP. También se puede definir áreas sólo-Apache, para restringir todas las actividades basadas en Web a archivos que no son de usuarios, o del sistema.

9.2.5. Seguridad del sistema de archivos

PHP está sujeto a la seguridad misma de la mayoría de sistemas de servidores en lo que a permisos sobre archivos y directorios se refiere. Esto le permite controlar cuáles archivos en el sistema de archivos pueden ser leídos. Debe tenerse cuidado con aquellos archivos que tengan permisos de lectura globales, para asegurarse de que su contenido es seguro y no represente peligro el que pueda ser leído por todos los usuarios con acceso al sistema de archivos.

Ya que PHP fue diseñado para permitir acceso al nivel de usuarios al sistema de archivos, es completamente posible escribir un script PHP que le permita leer archivos del sistema como /etc/passwd, modificar sus conexiones tipo ethernet, enviar trabajos de impresión masivos, etc. Esto tiene algunas implicaciones obvias, en el sentido en que

tiene que asegurarse que los archivos desde los que lee y hacia los que escribe datos, sean los correctos.

Si se considera el siguiente script, en donde un usuario indica que quisiera eliminar un archivo ubicado en su directorio personal. Este caso asume que se tratará de una situación en donde se usa normalmente una interfaz Web que se vale de PHP para la gestión de archivos, así que el usuario de Apache tiene permitido eliminar archivos en los directorios personales de los usuarios.

```
<?php
// eliminar un archivo del directorio personal del usuario
$nombre_usuario = $_POST['nombre_enviado_por_el_usuario'];
$directorio    = "/home/$nombre_usuario";
$archivo_a_eliminar = "$archivo_de_usuario";
unlink ("$directorio/$archivo_de_usuario");
echo "&iexcl;El archivo $archivo_a_eliminar ha sido eliminado!";
?>
```

Ya que el nombre de usuario es enviado desde un formulario de usuario, cualquiera puede enviar un nombre de usuario y archivo propiedad de otra persona, y eliminar archivos. En este caso, se opta por la utilización de otro método de autenticación. Consideraremos lo que sucede si las variables enviadas son "../etc/" y "passwd". El código entonces se ejecutaría efectivamente como:

```
<?php
// elimina un archivo de cualquier parte del disco duro al que el
// usuario de PHP tiene acceso. Si PHP tiene acceso de root:
$nombre_usuario = "../etc/";
$directorio    = "/home/../etc/";
$archivo_a_eliminar = "passwd";
unlink ("/home/../etc/passwd");
echo "&iexcl;El archivo /home/../etc/passwd ha sido eliminado!";
?>
```

Hay dos importantes medidas que se deben tener en cuenta para prevenir estas situaciones:

- Otorgarle únicamente permisos limitados al usuario Web del binario PHP.
- Chequear todas las variables que son enviadas por usuarios.

Aquí hay una versión mejorada del script:

```
<?php
// elimina un archivo de cualquier parte del disco duro al que el
// usuario de PHP tiene acceso.

$nombre_usuario = $_SERVER['REMOTE_USER']; // uso de un
                                           //mecanismo de
                                          // autenticación
$directorio = "/home/$nombre_usuario";
$archivo_a_eliminar = basename("$archivo_de_usuario"); // remover rutas
unlink ($directorio/$archivo_a_eliminar);
//registrar el proceso
$fp = fopen("/home/registros/eliminacion.log","+a");
$cadena_de_registro = "$nombre_usuario $directorio $archivo_a_eliminar";
fwrite ($fp, $cadena_de_registro);
fclose($fp);
echo "&iexcl;El archivo $archivo_a_eliminar ha sido eliminado!";
?>
```

Sin embargo, incluso este caso no está libre de problemas. Si nuestro sistema de autenticación nos ha permitido la creación de nuestros propios nombres en el sistema, y un usuario elige "../etc/", el sistema se encuentra nuevamente expuesto. Por esta razón, se prefiere escribir un chequeo más personalizado:

```
<?php
$nombre_usuario = $_SERVER['REMOTE_USER'];
// uso de un mecanismo de autenticacion
$directorio = "/home/$nombre_usuario";
if (!ereg('^[^./][^/]*$', $archivo_de_usuario))
     die('nombre de archivo inv&aacute;lido');//finaliza
                        // no ejecutar el proceso

if (!ereg('^[^./][^/]*$', $nombre_usuario))
     die('nombre de archivo inv&aacute;lido');//finaliza
        // no ejecutar el proceso etc...
?>
```

Dependiendo de su sistema operativo, existe una amplia variedad de archivos sobre los que deberemos estar atentos, incluyendo las entradas de dispositivos (/dev/ o COM1), archivos de configuración (archivos /etc/ y los archivos .ini), áreas conocidas de almacenamiento de datos (/home/, Mis Documentos), etc. Por esta razón, usualmente es más sencillo crear una política en donde se prohíba toda transacción excepto por aquellas que permitamos explícitamente.

9.2.6. Seguridad de bases de datos

Hoy en día, las bases de datos son componentes cardinales de cualquier aplicación basada en Web, permitiendo que los sitios Web provean contenido dinámico. Debido a que información considerablemente sensible o secreta puede ser almacenada en una base de datos, se debe considerar seriamente la protección de sus bases de datos.

Para recuperar o almacenar cualquier información se necesita conectarse a la base de datos, enviar una consulta válida, recoger el resultado y cerrar la conexión. Hoy en día, el lenguaje de consultas usado comúnmente en estas interacciones es el Lenguaje de Consultas Estructurado (SQL). Se puede apreciar cómo un atacante puede intentar acometidas con una consulta SQL.

Como es fácil suponer, PHP no puede proteger su base de datos por sí solo. A continuación se muestran conceptos básicos de cómo acceder y manipular bases de datos desde scripts PHP.

Hay que mantener en mente esta simple regla: protección en profundidad. Cuantas más acciones tomadas para incrementar la protección de su base de datos, menor será la probabilidad de que un atacante tenga éxito exponiendo o abusando de cualquier información almacenada. Un buen diseño del esquema de la base de datos y de la aplicación basta para lidiar con sus mayores temores.

9.2.6.1. Diseño de Bases de Datos

El primer paso siempre es crear la base de datos, a menos que se desee usar una creada por alguien más. Cuando una base de datos es creada, ésta es asignada a un dueño, quien ejecutó la sentencia de creación. Usualmente, únicamente el dueño (o un super-usuario) puede hacer cualquier tarea con los objetos de esa base de datos, y para que otros usuarios puedan utilizarla, deben otorgarse privilegios.

Las aplicaciones nunca deberían conectarse a la base de datos bajo el usuario correspondiente a su dueño, o como un super-usuario, ya que estos usuarios pueden, por ejemplo, ejecutar cualquier consulta a su antojo, modificando el esquema (por ejemplo, eliminando tablas) o borrando su contenido completo.

Se pueden crear diferentes usuarios de la base de datos para cada aspecto de su aplicación con derechos muy limitados sobre los objetos de la base de datos. Tan sólo deben otorgarse los privilegios estrictamente necesarios, y evitar que el mismo usuario pueda interactuar con la base de datos en diferentes casos de uso. Esto quiere decir que si un intruso gana acceso a su base de datos usando las credenciales de sus aplicaciones, él sólo puede efectuar tantos cambios como su aplicación se lo permita.

Es buena idea que no implemente toda la lógica del asunto en la aplicación Web (es decir, en su script); en su lugar, se hará en el esquema de la base de datos usando vistas,

disparadores o reglas. Si el sistema evoluciona, se espera que nuevos puertos sean abiertos a la aplicación, y tendrá que re-implementar la lógica para cada cliente de la base de datos. Pero sobre todo, los disparadores pueden ser usados para gestionar de forma transparente todos los campos automáticamente, lo cual con frecuencia provee información útil cuando se depuren problemas de nuestra aplicación, o se realicen rastreos sobre transacciones particulares.

9.2.6.2. Conexión con la Base de Datos

Puede que se deseen establecer las conexiones sobre SSL para encriptar las comunicaciones cliente/servidor, incrementando el nivel de seguridad, se puede hacer uso de SSH para encriptar la conexión de red entre los clientes y el servidor de la base de datos. Si se utiliza cualquiera de estos recursos, entonces monitorear el tráfico y adquirir información sobre la base de datos resultará difícil para un atacante potencial.

9.2.6.3. Modelo de Almacenamiento encriptado

SSL/SSH protege los datos que viajan desde el cliente al servidor, SSL/SSH no protege los datos persistentes almacenados en la base de datos. SSL es un protocolo sobre-el-cable.

Una vez el atacante adquiere acceso directo a su base de datos (evitando el paso por el servidor Web), los datos críticos almacenados pueden estar expuestos o mal utilizados, a menos que la información esté protegida en la base de datos misma. La encriptación de datos es una buena forma de mitigar esta amenaza, pero muy pocas bases de datos ofrecen este tipo de mecanismo de encriptación de datos.

La forma más sencilla de evitar este problema es crear primero su propio paquete de encriptación, y luego utilizarlo desde sus scripts de PHP. PHP puede ayudarle en este sentido con varias extensiones, como Mcrypt y Mhash, las cuales cubren una amplia variedad de algoritmos de encriptación. El script encripta los datos antes de insertarlos en la base de datos, y los descripta cuando los recupera.

En el caso de datos realmente escondidos, si nuestra representación original no se necesita (es decir, no debe ser desplegada), los resúmenes criptográficos pueden llegar a considerarse también. El ejemplo clásico de gestión de resúmenes criptográficos es el almacenamiento de secuencias MD5 de una contraseña en una base de datos, en lugar de la contraseña misma. Otra función para encriptado es la función crypt(). A continuación mostramos un ejemplo de la utilización de la función md5().

```
<?php


// almacenamiento de resumen criptografico de la
contrasenya
$consulta  = sprintf("INSERT INTO usuarios(nombre,contr)
VALUES('%s','%s');",
                        addslashes($nombre_usuario),
md5($contrasenya));
$resultado = pg_query($conexion, $consulta);
```

```
   // consulta de verificacion de la contrasenya enviada
   $consulta    =  sprintf("SELECT  1  FROM  usuarios  WHERE
nombre='%s' AND contr='%s';",
                        addslashes($nombre_usuario),
md5($contrasenya));
   $resultado = pg_query($conexion, $consulta);

   if (pg_num_rows($resultado) > 0) {
      echo '&iexcl;Bienvenido, $nombre_usuario!';
   }
   else {
      echo 'No pudo autenticarse a $nombre_usuario.';
   }

   ?>
```

9.2.6.4. Inyección de SQL

Muchos desarrolladores Web no son conscientes de cómo pueden manipularse las consultas SQL, y asumen que una consulta SQL es un comando confiable. Esto representa que las consultas SQL pueden burlar los controles de acceso, y de este modo evitar los chequeos estándares de autenticación y autorización, y a veces las consultas SQL pueden incluso permitir acceso a comandos al nivel del sistema operativo de la máquina huésped.

La Inyección Directa de Comandos SQL es una técnica en la cual un atacante crea o altera comandos SQL existentes para exponer datos escondidos, o sobrescribir datos críticos, o incluso ejecutar comandos del sistema peligrosos en la máquina en donde se encuentra la base de datos. Esto se consigue cuando la aplicación toma información de entrada del usuario y la combina con parámetros estáticos para construir una consulta SQL.

Debido a la falta de validación de la información de entrada y el establecimiento de conexiones con la base de datos desde un super-usuario o aquel que puede crear usuarios, el atacante podría crear un super-usuario en su base de datos.

En el siguiente ejemplo se muestra una paginación del conjunto de resultados y creación de super-usuarios:

```
   <?php
   $offset  = $argv[0]; //atencion, no se valida la entrada
   $consulta = "SELECT id, nombre FROM productos ORDER BY
nombre LIMIT 20 " .
               "OFFSET $offset;";
```

```
$resultado = pg_query($conexion, $consulta);
?>
```

Los usuarios normales pulsan sobre los enlaces 'siguiente' y 'anterior', en donde el desplazamiento ($offset) se encuentra codificado en la URL. El script espera que el valor entrante $offset sea un número decimal. Sin embargo, ¿qué sucede si alguien intenta un ataque añadiendo una forma codificada (urlencode()) de lo siguiente en la URL?:

```
0;
insert                                                    into
pg_shadow(usename,usesysid,usesuper,usecatupd,passwd)
      select 'crack', usesysid, 't','t','crack'
      from pg_shadow where usename='postgres';
--
```

Si esto ocurriera, entonces el script le presentaría un acceso de superusuario al atacante. Note que 0; es usado para ofrecer un desplazamiento válido a la consulta original y finalizarla.

Nota: Es una técnica común obligar al analizador sintáctico de SQL a que ignore el resto de la consulta escrita por el desarrollador mediante --, el cual es el signo de comentarios en SQL.

Una forma viable de adquirir contraseñas es jugar con las páginas de resultados de búsquedas. Lo único que necesita el atacante es ver si existen variables enviadas por el usuario que sean usadas en sentencias SQL, y que no sean tratadas apropiadamente. Estos filtros pueden ubicarse por lo general previos a cláusulas WHERE, ORDER BY, LIMIT y OFFSET en sentencias SELECT para personalizar la instrucción. Si nuestra base de datos soporta la construcción UNION, el atacante puede intentar añadir una consulta completa a la consulta original para generar una lista de contraseñas desde una tabla cualquiera. El uso de campos encriptados de contraseñas es altamente recomendable.

```
<?php
$consulta   = "SELECT id, nombre, insertado, tam FROM
productos WHERE tam = '$tam' ORDER BY $orden LIMIT $limite,
$offset;";
$resultado = odbc_exec($conexion, $consulta);
?>
La parte estática de la consulta puede combinarse con
otra sentencia SELECT la cual revela todas las contraseñas:
'
union select '1', concat(uname||'-'||passwd) as name,
'1971-01-01', '0' from usertable;
--
```

Si esta consulta (la cual juega con ' y --) fuera asignada a una de las variables usadas en $consulta, la bestia de la consulta habrá despertado.

Las sentencias UPDATE de SQL son también susceptibles a ataque. Estas consultas también se encuentran amenazadas por un posible acotamiento y adición de una consulta completamente nueva. Pero en este caso el atacante puede amañar la información de una cláusula SET. En este caso se requiere contar con cierta información sobre el esquema de la base de datos para poder manipular la consulta satisfactoriamente. Esta información puede ser adquirida mediante el estudio de los nombres de variables de los formularios, o simplemente por fuerza bruta. No existen demasiadas convenciones para nombrar campos de contraseñas o nombres de usuario.

```
<?php
$consulta = "UPDATE usertable SET pwd='$pwd' WHERE uid='$uid';";
?>
```

Pero un usuario malicioso envía el valor ' or uid like'%admin%'; -- como $uid para cambiar la contraseña del administrador, o simplemente establece $pwd a "hehehe', admin='yes', trusted=100 " (con un espacio al inicio) para adquirir más privilegios. En tal caso, la consulta sería manipulada:

```
<?php
// $uid == ' or uid like'%admin%'; --
$consulta = "UPDATE usertable SET pwd='...' WHERE uid='' or uid like '%admin%'; --";


// $pwd == "hehehe', admin='yes', trusted=100 "
$consulta = "UPDATE usertable SET pwd='hehehe', admin='yes', trusted=100 WHERE ...;"
?>
```

Un horrible ejemplo de cómo puede accederse a comandos del nivel del sistema operativo en algunas máquinas anfitrionas de bases de datos.

```
<?php
$consulta = "SELECT * FROM productos WHERE id LIKE '%$prod%'";
$resultado = mssql_query($consulta);
?>
```

Si el atacante envía el valor a%' exec master..xp_cmdshell 'net user test testpass /ADD' -- a $prod, entones la $consulta será:

```
<?php
$consulta  = "SELECT * FROM productos
                    WHERE id LIKE '%a%'
                    exec master..xp_cmdshell 'net user test testpass /ADD'--";
$resultado = mssql_query($consulta);
```

```
?>
```

MSSQL Server ejecuta sentencias SQL en el lote, incluyendo un comando para agregar un nuevo usuario a la base de datos de cuentas locales. Si esta aplicación estuviera corriendo como sa y el servicio MSSQLSERVER está corriendo con los privilegios suficientes, el atacante tendría ahora una cuenta con la que puede acceder a esta máquina.

Nota: Algunos de los ejemplos anteriores están atados a un servidor de base de datos específico. Esto no quiere decir que un ataque similar sea imposible con otros productos. Su base de datos puede ser vulnerable de forma semejante, en alguna otra manera.

9.2.6.4.1. Técnicas de protección

Se puede argumentar con justa razón que el atacante debe poseer cierta cantidad de información sobre el esquema de la base de datos en la mayoría de ejemplos que se han detallado. Ciertamente, nunca se sabrá cuándo y cómo puede filtrarse esta información, y si ocurre, su base de datos estará expuesta. Si se está usando un paquete de gestión de bases de datos de código abierto, o cuyo código fuente está disponible públicamente, el cual puede pertenecer a algún sistema de administración de contenido o foro, los intrusos pueden producir fácilmente una copia de un trozo de su código. También puede ser un riesgo de seguridad si es un segmento de código pobremente diseñado.

Estos ataques se basan principalmente en la explotación del código que no ha sido escrito pensando en la seguridad. Nunca se debe confiar en ningún tipo de información de entrada, especialmente aquella que proviene del lado del cliente, aun si lo hace desde una caja de selección, un campo de entrada hidden o una cookie. El primer ejemplo le muestra que una consulta así de descuidada puede causar desastres.

- Nunca se conecte a la base de datos como un super-usuario o como el dueño de la base de datos. Use siempre usuarios personalizados con privilegios muy limitados.
- Revise si la entrada recibida es del tipo apropiado. PHP posee un amplio rango de funciones de validación de datos, desde los más simples encontrados en Funciones sobre variables y en Funciones de tipo de carácter (por ejemplo, is_numeric(), ctype_digit() respectivamente) hasta el soporte para Expresiones Regulares compatibles con Perl.
- Si la aplicación espera alguna entrada numérica, considere la verificación de información con is_numeric(), o modifique silenciosamente su tipo usando settype(), o utilice su representación numérica, dada por sprintf().

```
<?php
settype($offset, 'integer');
$consulta = "SELECT id, nombre FROM productos ORDER BY
nombre " .
            "LIMIT 20 OFFSET $offset;";

// note el simbolo %d en la cadena de formato, usar %s no
tendria sentido
```

```
   $consulta = sprintf("SELECT id, nombre FROM productos
ORDER BY nombre" .
                       "LIMIT 20 OFFSET %d;", $offset);
   ?>
```

- Hay que ubicar cada valor no-numérico que entrega el usuario y que sea pasado a la base de datos entre comillas con la función de escape de cadenas específica a su base de datos (p.ej. mysql_escape_string(), sql_escape_string(), etc.). Si no hay disponible un mecanismo de escape de cadenas específico a la BD, las funciones addslashes() y str_replace() pueden ser útiles (dependiendo del tipo de base de datos).
- No se debe imprimir ninguna información específica sobre la base de datos, especialmente sobre su esquema, ya sea por razones justas o por equivocaciones.
- Se puede hacer uso de procedimientos almacenados y cursores previamente definidos para abstraer el acceso a las bases de datos, de modo que los usuarios no tengan acceso directo a las tablas o vistas, aunque esta solución tiene otros impactos.

Además de estas acciones, se puede producir un beneficio con el registro explícito de las consultas realizadas, ya sea desde su script o por la base de datos misma, si ésta soporta la gestión de registros. Por supuesto, el registro de acciones no puede prevenir cualquier intento peligroso, pero puede ser útil para rastrear qué aplicaciones han sido usadas para violar la seguridad. El registro en sí no es útil; lo es la información que contiene. Por lo general, es mejor contar con más detalles que con menos.

9.2.7. Reporte de errores

Hablando de la seguridad en PHP, hay dos caras en lo que se concierne al reporte de errores. Una es benéfica al incremento de la seguridad, la otra va en dirección de su detrimento.

Una táctica de ataque típica involucra la acumulación de un perfil de datos del sistema, alimentándolo con datos inapropiados, y luego chequeando los tipos de errores que son devueltos, y sus contextos. Esto permite que el cracker del sistema pueda adquirir información del servidor, para así determinar posibles debilidades. Por ejemplo, si un atacante ha recogido información sobre una página creada a partir de los datos de un formulario, él podría intentar sobrescribir las variables, o modificarlas:

```
   <form                                    method="post"
action="destino_del_ataque?username=badfoo&amp;password=badf
oo">
   <input type="hidden" name="username" value="badfoo" />
   <input type="hidden" name="password" value="badfoo" />
   </form>
```

Los errores de PHP que son devueltos normalmente pueden ser bastante útiles para un desarrollador que esté tratando de depurar un script, indicando cosas como la función

o archivo que falló, el archivo PHP y el número de línea en donde ocurren los fallos. Toda esta es información de la que puede sacarse provecho. No es extraño que un desarrollador php use show_source(), highlight_string(), o highlight_file() como medida de depuración, pero en un sitio en producción, esta acción puede exponer variables ocultas, sintaxis sin chequear, y otra información peligrosa. Algo especialmente peligroso es ejecutar código que proviene de fuentes bien conocidas con gestores de depuración incorporados, o que usan técnicas de depuración comunes. Si el atacante puede determinar qué técnica general está usando, puede intentar un ataque de fuerza bruta sobre una página, enviando varias cadenas comunes de depuración:

```
<form method="post"
action="destino_del_ataque?errors=Y&amp;showerrors=1&amp;deb
ug=1">
<input type="hidden" name="errors" value="Y" />
<input type="hidden" name="showerrors" value="1" />
<input type="hidden" name="debug" value="1" />
</form>
```

Independientemente del método de gestión de errores, la capacidad de conseguir que un sistema revele sus posibles estados de error representa un camino para darle información al atacante.

Por ejemplo, el estilo mismo de un error de PHP genérico indica que el sistema está ejecutando PHP. Si el atacante estuviera viendo una página .html, y quisiera consultar qué está siendo usado para la generación de ella por detrás (en busca de debilidades conocidas en el sistema), podría determinar que el sistema fue creado usando PHP alimentándolo con información equivocada.

Un error de función puede indicar si el sistema está ejecutando un tipo particular de motor de base de datos, o dar pistas sobre cómo fue programada o diseñada una página Web. Esto facilita posteriores investigaciones en determinados puertos abiertos de bases de datos, o en busca de fallos específicos o debilidades en una página Web. Al entregar diferentes trozos de datos inválidos al sistema, por ejemplo, un atacante puede determinar el orden de autenticación en un script, (a partir de los números de línea de los errores) así como averiguar sobre vulnerabilidades que pueden aprovecharse en diferentes puntos del script.

Un error del sistema de archivos o en general de PHP puede indicar qué permisos tiene el servidor Web, así como la estructura y organización de los archivos en el servidor Web. Algún código de gestión de errores escrito por el desarrollador puede agravar este problema, llevando a la fácil explotación de información hasta entonces "escondida".

Existen tres soluciones principales a este problema. La primera es revisar cuidadosamente todas las funciones, y tratar de compensar por la mayoría de errores encontrados. La segunda es deshabilitar el reporte de errores completamente del código que está siendo ejecutado. La tercera es usar las funciones de gestión de errores personalizables de PHP para crear su propio gestor de errores. Dependiendo de su política de seguridad, puede encontrar que todas ellas pueden ser aplicables a su situación.

Una forma de detectar este problema por adelantado es hacer uso del reporte de errores propio de PHP (error_reporting()), para ayudarnos a asegurar nuestro código y

encontrar uso de variables que pueda ser peligroso. Al probar nuestro código, previamente a su entrega final, con E_ALL, se pueden encontrar rápidamente áreas en donde nuestras variables pueden estar abiertas a la manipulación y explotación en distintas formas. Una vez esté listo para liberar nuestro código, es buena idea que deshabilitemos el reporte de errores por completo definiendo error_reporting() a 0, o desactivar la impresión de errores usando la opción de php.ini display_errors, de modo que se puede aislar nuestro código de ataques potenciales. Si se opta por la opción de la última opción, debemos definir también la ruta a nuestros archivos de registro usando la directiva ini error_log, y habilitar log_errors.

```
<?php
if ($nombre_usuario) {   // Variable no inicializada o chequeada antes de su uso
    $login_correcto = 1;
}
if ($login_correcto == 1) { // Si la condicion anterior falla, esta variable
                            //     no     se     encuentra inicializada ni validada
                            // antes de su uso
    readfile ("/informacion/altamente/confidencial/index.html");
}
?>
```

9.2.8. Uso de register globals

Quizás el cambio más controvertido en la historia de PHP se ha dado cuando la directiva register_globals pasó de tener como valor por defecto ON al valor OFF en PHP 4.2.0. La dependencia sobre esta directiva era bastante común y muchas personas ni siquiera estaban enteradas de que existía y asumían que ese era el modo en que PHP trabajaba. A continuación se explica cómo puede llegar a escribirse código inseguro con esta directiva pero hay que tener en cuenta, que no es la directiva misma la que es insegura sino el uso inapropiado de ella.

Cuando se encuentra activa, la directiva register_globals se inyecta en los scripts con todo tipo de variables, como variables de peticiones provenientes de formularios HTML. Esto junto con el hecho de que PHP no requiere la inicialización de variables significa que es muy fácil escribir código inseguro. Fue una decisión difícil, pero la comunidad de PHP decidió deshabilitar esta directiva por defecto. Cuando está habilitada, se podrían usar variables sin saber con seguridad de dónde provienen y sólo queda asumir. Las variables internas que son definidas en el script mismo son mezcladas con los datos enviados por los usuarios y al deshabilitar register_globals se modifica este comportamiento. A continuación, se demuestra este caso con un ejemplo del uso incorrecto register_globals:

```
<?php
```

```
   // definir $autorizado = true solo si el usuario ha sido
autenticado


   if (usuario_autenticado()) {
      $autorizado = true;
   }
   // Ya que no inicializamos $autorizado como false, esta
podria estar
   // definida a traves de register_globals, como en el caso
de GET
   // auth.php?autorizado=1
   //  De modo que cualquier persona podria verse como
autenticada!
   if ($autorizado) {
      include "/datos/muy/importantes.php";
   }
   ?>
```

Cuando register_globals = on, nuestra lógica anterior podría verse comprometida. Cuando la directiva está deshabilitada, $autorizado no puede definirse a través de peticiones, así que no habrá ningún problema, aunque es cierto que siempre es una buena práctica de programación inicializar las variables primero. Por ejemplo, en nuestro ejemplo anterior pudimos haber realizado primero algo como $authorized = false. Hacer esto representa que el código anterior podría funcionar con register_globals establecido a on u off ya que los usuarios no serían autorizados por defecto.

Otro ejemplo es aquel de las sesiones. Cuando register_globals = on, podríamos usar también $nombre_usuario en nuestro siguiente ejemplo, pero nuevamente usted debe notar que $nombre_usuario puede provenir de otros medios, como GET (a través de la URL).

```
   <?php
   // No sabriamos de donde proviene $nombre_usuario, pero
sabemos que
   // $_SESSION es para datos de sesion
   if (isset($_SESSION['nombre_usuario'])) {
      echo "Hola <b>{$_SESSION['nombre_usuario']}</b>";
   } else {
      echo "Hola <b>Invitado</b><br />";
      echo "&iquest;Quisiera iniciar su sesi&oacute;n?";
   }
   ?>
```

Incluso es posible tomar medidas preventivas para advertir cuando se intente falsificar la información. Si se sabe previamente con exactitud el lugar de donde debería provenir una variable, se podrá chequear si los datos enviados provienen de una fuente inadecuada. Aunque esto no garantiza que la información no haya sido falsificada, esto requiere que un atacante adivine el medio apropiado para falsificar la información. Si no nos importa de dónde proviene la información, podemos usar $_REQUEST ya que allí se incluye una mezcla de variables que provienen de datos GET, POST y COOKIE.

```
<?php
if (isset($_COOKIE['COOKIE_MAGICA'])) {
   // COOKIE_MAGICA proviene de una cookie.
   // Asegurese de validar los datos de la cookie!
} elseif (isset($_GET['COOKIE_MAGICA']) || isset($_POST['COOKIE_MAGICA'])) {
   mail("admin@example.com", "Posible intento de intromision",
       $_SERVER['REMOTE_ADDR']);
   echo "Violaci&oacute;n de seguridad, el administrador ha sido alertado.";
   exit;
} else {
   // COOKIE_MAGICA no fue definida en este REQUEST
}
?>
```

Por supuesto, deshabilitar register_globals no quiere decir que su código vaya a ser seguro. Por cada trozo de datos que sea enviado por el usuario, éste debe ser chequeado en otras formas. ¡Siempre hay que validar los datos de los usuarios e inicializar sus variables! Para chequear por variables no inicializadas, podemos usar error_reporting() para mostrar errores del nivel E_NOTICE.

9.2.9. Datos enviados por el usuario

Las mayores debilidades de muchos programas PHP no son inherentes al lenguaje mismo, sino simplemente un problema generado cuando se escribe código sin pensar en la seguridad. Por esta razón, deberíamos tomarnos siempre un tiempo para considerar las implicaciones de cada pedazo de código, para averiguar el posible peligro involucrado cuando una variable inesperada es enviada.

```
<?php
// eliminar un archivo del directorio personal del usuario .. o
// quizas de alguien mas?
unlink ($variable_malvada);
```

```
// Imprimir el registro del acceso... o quizas una entrada de /etc/passwd?
fwrite ($desc_archivo, $variable_malvada);
// Ejecutar algo trivial.. o rm -rf *?
system ($variable_malvada);
exec ($variable_malvada);
?>
```

Se debe examinar siempre, y cuidadosamente nuestro código para asegurarnos de que cualquier variable siendo enviada desde un navegador Web sea chequeada apropiadamente, y preguntarse a sí mismo:

- ¿Este script afectará únicamente los archivos que se pretende?
- ¿Puede tomarse acción sobre datos inusuales o indeseados?
- ¿Puede ser usado este script en formas malintencionadas?
- ¿Puede ser usado en conjunto con otros scripts en forma negativa?
- ¿Serán adecuadamente registradas las transacciones?

Al preguntarse adecuadamente estas preguntas mientras se escribe nuestro script, en lugar de hacerlo posteriormente, se previene una desafortunada re-implementación del programa cuando se desee incrementar el nivel de seguridad. Al comenzar con esta mentalidad, no se garantiza la seguridad de su sistema, pero se puede ayudar a mejorarla.

Puede que también deseemos considerar la deshabilitación de register_globals, magic_quotes, u otros parámetros convenientes que pueden causar confusión sobre la validez, fuente o valor de una determinada variable. Trabajar con PHP en modo error_reporting(E_ALL) también puede ayudarnos a advertir variables que están siendo usadas antes de ser chequeadas o inicializadas (de modo que se pueda prevenir que datos inusuales produzcan operaciones inadvertidas).

9.2.10. Comillas mágicas

Las comillas mágicas (o "Magic Quotes") se refieren a un proceso que automáticamente escapa datos de entrada en los scripts de PHP. Es recomendable escribir código con las comillas mágicas deshabilitadas, y en su lugar escapar los datos en tiempo de ejecución, a medida que se necesite.

9.2.10.1. Qué son lasas comillas mágicas

Cuando se habilitan, todos los caracteres ' (comilla sencilla), " (comilla doble), \ (barra invertida) y NULL se escapan con una barra invertida de forma automática. Esto es idéntico a lo que hace addslashes().

Existen tres directivas de comillas mágicas:

- magic_quotes_gpc. Afecta los datos de peticiones HTTP (GET, POST y COOKIE). No puede definirse en tiempo de ejecución, y su valor predeterminado es on en PHP.
- magic_quotes_runtime. Si se habilita, la mayoría de funciones que devuelven datos de una fuente externa, incluyendo bases de datos y archivos de texto, escaparán las comillas con una barra invertida. Puede definirse en tiempo de ejecución, y su valor predeterminado en PHP es off.
- magic_quotes_sybase. Si se habilita, una comilla sencilla se escapa con una comilla sencilla en lugar de una barra invertida. Asimismo, sobreescribe completamente magic_quotes_gpc. Habilitar ambas directivas quiere decir que sólo las comillas sencillas se escapan como ". Las comillas dobles, las barras invertidas y los NULL permanecerán intactos y sin escapar.

9.2.10.2. Por qué usar comillas mágicas

- Útil para principiantes
- Las comillas mágicas son implementadas en PHP para ayudar a que el código escrito por principiantes no resulte peligroso. Aunque la Inyección SQL es posible aun con las comillas mágicas habilitadas, el riesgo es reducido.
- Conveniencia
- Para insertar datos en una base de datos, las comillas mágicas básicamente usan addslashes() sobre todos los datos Get, Post, y Cookie, y lo hace automáticamente.

9.2.10.3. Porque no usar comillas mágicas

- Portabilidad. Asumir que estén habilitadas, o deshabilitadas, afecta la portabilidad. Use get_magic_quotes_gpc() para verificar su valor, y escriba su código de forma acorde.
- Rendimiento. Ya que no todo fragmento de datos escapados es insertado en una base de datos, existe una pérdida en rendimiento al escapar todos esos datos. Simplemente llamar las funciones de escape (como addslashes()) en tiempo de ejecución es más eficiente. Aunque php.ini-dist habilita estas directivas por defecto, php.ini-recommended las deshabilita. Esta recomendación se hace principalmente por razones de rendimiento.
- Inconveniencia. Ya que no todos los datos necesitan ser escapados, con frecuencia resulta molesto ver datos escapados cuando no debieran estarlo. Por ejemplo, enviar un correo electrónico desde un formulario y ver un montón de \' en el mensaje. Arreglar esto puede requerir un uso exagerado de stripslashes().

9.2.10.4. Desactivación de comillas mágicas

Puede que la directiva magic_quotes_gpc solo pueda ser desactivada en el nivel de sistema, y no en tiempo de ejecución. En otras palabras, usar ini_set() no es posible.

Un ejemplo que define el valor de estas directivas a Off en php.ini:

```
; Comillas magicas
```

```
;
; Comillas magicas para datos GET/POST/Cookie de entrada.
magic_quotes_gpc = Off
; Comillas magicas para datos generados en tiempo de
ejecucion,
; p.ej. desde SQL, exec(), etc
magic_quotes_runtime = Off
; Usar comillas magicas tipo Sybase (escapar ' con '' en
lugar de \').
magic_quotes_sybase = Off
```

Si el acceso a la configuración del servidor no se encuentra disponible, el uso de .htaccess es también una opción. Por ejemplo:

```
php_flag magic_quotes_gpc Off
```

Con el propósito de escribir código portable, por ejemplo si la configuración en el nivel del servidor no es posible, he aquí un ejemplo de cómo deshabilitar magic_quotes_gpc en tiempo de ejecución. Este método es ineficiente así que es preferible definir las directivas apropiadas en algún otro lugar.

```
<?php
if (get_magic_quotes_gpc()) {
    function stripslashes_profundo($valor)
    {
        $valor = is_array($valor) ?
                    array_map('stripslashes_profundo',
$valor) :
                    stripslashes($valor);
        return $valor;
    }
    $_POST = array_map('stripslashes_profundo', $_POST);
    $_GET = array_map('stripslashes_profundo', $_GET);
    $_COOKIE     =     array_map('stripslashes_profundo',
$_COOKIE);
}
?>
```

9.2.11. Ocultando PHP

En general, la seguridad por oscuridad es una de las formas más débiles de seguridad. Pero, en algunos casos, cada pequeño elemento extra de seguridad es deseable.

Unas cuantas técnicas simples pueden ayudarle a esconder PHP, posiblemente retrasando a un atacante que esté intentando descubrir debilidades en su sistema. Al establecer expose_php = off en su archivo php.ini, se reduce la cantidad de información disponible a posibles atacantes.

Otra táctica consiste en configurar los servidores Web como Apache para que procesen diferentes tipos de archivos como scripts de PHP, ya sea con una directiva .htaccess, o en el archivo de configuración de apache mismo. En ese caso puede usar extensiones de archivo que produzcan confusión:

```
# Hacer que el codigo PHP parezca como otro tipo de codigo
AddType application/x-httpd-php .asp .py .pl
```

Y ocultarlo completamente:

```
# Hacer que el codigo PHP parezca como de tipos desconocidos
AddType application/x-httpd-php .bop .foo .133t
```

O escondiéndolo como código HTML, lo que tiene un pequeño impacto de rendimiento ya que todos los documentos HTML serán procesados por el motor de PHP:

```
# Hacer que todo el codigo PHP luzca como HTML
AddType application/x-httpd-php .htm .html
```

Para que esto funcione de manera efectiva, se debe renombrar nuestros archivos PHP con las extensiones anteriores. Aunque es una forma de seguridad por oscuridad, representa una medida preventiva menor con pocos inconvenientes.

9.2.12. Mantenerse al Día

PHP, como cualquier otro sistema de tamaño considerable, está bajo constante escrutinio y remodelación. Cada nueva versión incluye con frecuencia cambios mayores y menores para mejorar la seguridad y reparar cualquier fallo, problemas de configuración, y otros asuntos que puedan afectar la seguridad y estabilidad global de su sistema.

Como cualquier lenguaje y programa de scripting del nivel del sistema, el mejor enfoque es el de actualizar con frecuencia, y mantenerse alerta sobre las últimas versiones y sus cambios.

9.3. JavaScript

JavaScript [53] es un lenguaje de programación interpretado que proviene del estándar ECMAScript. Se define como orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico.

Se utiliza principalmente en su forma del lado del cliente (*client-side*), implementado como parte de un navegador Web, permitiendo mejoras en la interfaz de usuario y

páginas Web dinámicas en bases de datos locales al navegador, aunque existe una forma de JavaScript del lado del servidor (Server-side JavaScript o SSJS). Se utiliza en aplicaciones externas a la Web, por ejemplo, en documentos PDF o aplicaciones de escritorio (mayoritariamente widgets).

JavaScript se diseñó con una sintaxis similar al lenguaje de programación C, aunque adopta nombres y convenciones de Java. Sin embargo, Java y JavaScript no están relacionados y tienen semánticas y propósitos diferentes.

Todos los navegadores modernos interpretan el código JavaScript integrado en las páginas Web. Para interactuar con una página Web, se provee al lenguaje JavaScript de una implementación del Document Object Model (DOM).

Tradicionalmente se venía utilizando en páginas Web HTML para realizar operaciones y, únicamente en el marco de la aplicación cliente, sin acceso a funciones del servidor. JavaScript se interpreta en el agente de usuario, al mismo tiempo que las sentencias van descargándose junto con el código HTML.

9.3.1. Denominación e historia

JavaScript fue desarrollado originalmente por Brendan Eich de Netscape con el nombre de Mocha, el cual fue renombrado posteriormente a LiveScript, para finalmente quedar como JavaScript. El cambio de nombre coincidió aproximadamente con el momento en que Netscape agregó soporte para la tecnología Java en su navegador Web Netscape Navigator, en la versión 2.002 en diciembre de 1995. La denominación produjo confusión, dando la impresión de que el lenguaje es una prolongación de Java, y se ha caracterizado por muchos como una estrategia de mercadotecnia de Netscape para obtener prestigio e innovar en lo que eran los nuevos lenguajes de programación Web.

JavaScript es una marca registrada de Oracle Corporation. Se utiliza con licencia por los productos creados por Netscape Communications y entidades actuales como la Fundación Mozilla.

Microsoft dio como nombre a su dialecto JavaScript (JScript), para evitar problemas relacionados con la marca. JScript fue adoptado en la versión 3.0 de Internet Explorer, liberado en agosto de 1996, e incluyó compatibilidad con el Efecto 2000 con las funciones de fecha. Los lenguajes pueden parecer tan similares que los términos JavaScript y JScript a menudo se utilizan indistintamente, pero la especificación de JScript es incompatible con la de ECMA en muchos aspectos.

Para evitar estas incompatibilidades, el World Wide Web Consortium diseñó el estándar Document Object Model (DOM, o Modelo de Objetos del Documento), que incorporan Konqueror, las versiones 6 de Internet Explorer y Netscape Navigator, la versión 7 de Opera, Mozilla Application Suite y Mozilla Firefox desde su primera versión.

En 1997, los autores propusieron que JavaScript fuera adoptado como estándar de la European Computer Manufacturers Association ECMA, que aunque no lo parezca, no es europeo sino internacional, con sede en Ginebra. En junio de 1997, fue adoptado como un estándar ECMA, con el nombre de ECMAScript. Poco después también como un estándar ISO.

9.3.2. Diferencias entre Java y JavaScript

Realmente JavaScript se llamó así porque Netscape, que estaba aliado con los creadores de Java en la época, quiso aprovechar el conocimiento y la percepción que las personas tenían del popular lenguaje. Con todo, se creó un producto que tenía ciertas similitudes como la sintaxis del lenguaje o el nombre. Se hizo entender que era un hermano pequeño orientado específicamente para hacer cosas en las páginas Web, pero también se hizo caer a muchas personas en el error de pensar que eran lo mismo.

Actualmente son productos totalmente distintos y no guardan entre si más relación que la sintaxis idéntica y alguna otra similitud. Algunas diferencias entre estos dos lenguajes son las siguientes:

- **Compilador**. Para programar en Java se necesita un Kit de desarrollo y un compilador. Sin embargo, JavaScript no es un lenguaje que necesite que sus programas se compilen, sino que éstos se interpretan por parte del navegador cuando éste lee la página.
- **Orientado a objetos**. Java es un lenguaje de programación orientado a objetos. JavaScript no es orientado a objetos, esto quiere decir que se puede programar sin necesidad de crear clases, tal como se realiza en los lenguajes de programación estructurada como C o Pascal.
- **Propósito**. Java es mucho más potente que JavaScript, esto se debe a que Java es un lenguaje de propósito general, con el que se pueden hacer aplicaciones de lo más variado, sin embargo, con JavaScript sólo se puede escribir programas para que se ejecuten en páginas Web.
- **Estructuras fuertes**. Java es un lenguaje de programación fuertemente tipado, esto quiere decir que al declarar una variable hay que indicar su tipo y no podrá cambiar de un tipo a otro automáticamente. Por su parte, JavaScript no tiene esta característica, y se puede meter en una variable la información deseada, independientemente del tipo de ésta. Además, se puede cambiar el tipo de información de una variable en el momento que se quiera.
- **Otras características**. Como se puede ver, Java es mucho más complejo, aunque también más potente, robusto y seguro. Tiene más funcionalidades que JavaScript y las diferencias que los separan son lo suficientemente importantes como para distinguirlos fácilmente.

9.4. Ajax

AJAX [53], acrónimo de ***A**synchronous **J**avaScript **A**nd **X**ML* (JavaScript asíncrono y XML), es una técnica de desarrollo Web para crear aplicaciones interactivas o RIA (*Rich Internet Applications*). Estas aplicaciones se ejecutan en el cliente, es decir, en el navegador de los usuarios mientras se mantiene la comunicación asíncrona con el servidor en segundo plano. De esta forma es posible realizar cambios sobre las páginas sin necesidad de recargarlas, lo que significa aumentar la interactividad, velocidad y usabilidad en las aplicaciones.

Ajax es una tecnología asíncrona, en el sentido de que los datos adicionales se solicitan al servidor y se cargan en segundo plano sin interferir con la visualización ni el comportamiento de la página. JavaScript es el lenguaje interpretado (scripting language) en el que normalmente se efectúan las funciones de llamada de Ajax mientras que el acceso a los datos se realiza mediante *XMLHttpRequest*, objeto disponible en los navegadores actuales. En cualquier caso, no es necesario que el contenido asíncrono esté formateado en XML.

Ajax es una técnica válida para múltiples plataformas y utilizable en muchos sistemas operativos y navegadores debido a que está basado en estándares abiertos como JavaScript y Document Objec Model (DOM).

9.4.1. Tecnologías incluidas en Ajax

Ajax es una combinación de cuatro tecnologías ya existentes:

- XHTML (o HTML) y hojas de estilos en cascada (CSS) para el diseño que acompaña a la información.
- Document Object Model (DOM) accedido con un lenguaje de scripting por parte del usuario, especialmente implementaciones ECMAScrip como JavaScript y JScript, para mostrar e interactuar dinámicamente con la información presentada.
- El objeto XMLHttpRequest para intercambiar datos de forma asíncrona con el servidor Web. En algunos frameworks y en algunas situaciones concretas, se usa un objeto `iframe` en lugar del XMLHttpRequest para realizar dichos intercambios. PHP es un lenguaje de programación de uso general de script del lado del servidor originalmente diseñado para el desarrollo Web de contenido dinámico también utilizado en el método Ajax.
- XML es el formato utilizado generalmente para la transferencia de datos solicitados al servidor, aunque cualquier formato puede funcionar, incluyendo HTML preformateado, texto plano, JSON y hasta EBML.

Como DHTML, LAMP o SPA, Ajax no constituye una tecnología en sí, sino que es un término que engloba a un grupo de éstas que trabajan conjuntamente.

9.4.2. Antecedentes de Ajax

A pesar de que el término Ajax fue creado en 2005 por Jesse James Garrett, la historia de las tecnologías que permiten Ajax se remonta a una década antes con la iniciativa de Microsoft en el desarrollo de Scripting Remoto. Sin embargo, las técnicas para la carga asíncrona de contenidos en una página existente sin requerir recarga completa, se remontan al tiempo del elemento `iframe` (introducido en Internet Explorer 3 en 1996) y al tiempo de elemento `layer` (introducido en Netscape 4 en 1997, abandonado durante las primeras etapas de desarrollo de Mozilla). Ambos tipos de elementos tenían el atributo `src` que podía tomar cualquier dirección URL externa, y cargar una página que contenga JavaScript que manipule la página paterna. De este modo se logran efectos parecidos al Ajax.

El Microsoft's Remote Scripting (o MSRS, introducido en 1998) resultó un sustituto más elegante para estas técnicas, con envío de datos a través de un applet Java, el cual se podía comunicar con el cliente usando JavaScript. Esta técnica funcionó en ambos navegadores, Internet Explorer versión 4 y Netscape Navigator versión 4. Microsoft la utilizó en el Outlook Web Access provisto con la versión 2000 de Microsoft Exchange Server.

La comunidad de desarrolladores Web, primero, colaborando por medio del grupo de noticias *microsoft.public.scripting.remote* y después usando blogs, desarrollaron una gama de técnicas de scripting remoto para conseguir los mismos resultados en diferentes navegadores. Los primeros ejemplos incluyen la biblioteca JSRS en el año 2000, la introducción a la técnica imagen/cookie en el mismo año y la técnica JavaScript bajo demanda (*JavaScript on Demand*) en 2002. En ese año, se realizó una modificación por parte de la comunidad de usuarios al *Microsoft's Remote Scripting* para reemplazar el applet Java por XMLHttpRequest.

Frameworks de Scripting Remoto como el ARSCIF aparecieron en 2003 poco antes de que Microsoft introdujera Callbacks en ASP .NET.

Desde que XMLHttpRequest está implementado en la mayoría de los navegadores, raramente se usan técnicas alternativas. Sin embargo, todavía se utilizan, donde se requiere una mayor compatibilidad, una implementación reducida o acceso cruzado entre sitios Web. Una alternativa, el Terminal SVG (basado en SVG), emplea una conexión persistente para el intercambio continuo entre el navegador y el servidor.

9.4.3. Problemas e Inconvenientes

Las páginas creadas dinámicamente mediante peticiones sucesivas AJAX, no son registradas de forma automática en el historial del navegador, así que haciendo clic en el botón de "volver" del navegador, el usuario no será devuelto a un estado anterior de la página, en cambio puede volver a la última página que visitó. Soluciones incluyen el uso de IFrames invisibles para desencadenar cambios en el historial del navegador y el cambio de la porción de anclaje de la dirección (después de un #).

Los motores de búsqueda no analizan JavaScript. La información en la página dinámica no se almacena en los registros del buscador.

Existen problemas usando Ajax entre nombres de dominios, a esto se le conoce como *Same Origin Policy* o *Política del Mismo Origen*, el cual es una medida de seguridad, que puede ser solucionada con *Cross-Origin Resource Sharing* (CORS).

Dependiendo de cómo se desarrolle el sitio web, se puede mejorar o empeorar la carga en el servidor. Ajax ayuda al servidor a evitar la fase de renderización de HTML, dejándole ese trabajo al cliente, pero también puede sobrecargar al servidor si se hacen varias llamadas.

Es posible que páginas con Ajax no puedan funcionar en teléfonos móviles, PDA’s u otros dispositivos. Ajax no es compatible con todo el software destinado a ciegos u personas con otras discapacidades.

9.4.4. Navegadores que permiten Ajax

- Navegadores basados en Gecko como Mozilla, Mozilla Firefox, SeaMonkey, Camino, K-Meleon, IceWeasel, Flock, Epiphany, Galeon y Netscape versión 7.1 y superiores.
- Navegadores basados en WebKit como Google Chrome de Google o Safari de Apple.
- Microsoft Internet Explorer para Windows versión 5.0 y superiores, y los navegadores basados en él.
- Navegadores con el API KHTML versión 3.2 y superiores, incluyendo Konqueror versión 3.2 y superiores y el Web Browser for S60 de Nokia tercera generación y posteriores.
- Opera versión 8.0 y superiores, incluyendo Opera Mobile Browser versión 8.0 y superiores.

9.4.5. Navegadores que no permiten Ajax

- Opera 7 y anteriores.
- Microsoft Internet Explorer para Windows versión 4.0 y anteriores.
- Anteriores a Safari 1.2.
- Dillo.
- Navegadores basados en texto como Lynx y Links.
- Navegadores destinados a personas con discapacidad visual (Braille).
- Algunos navegadores de teléfonos móviles.
- Navegador de la PSP.

9.5. HTML5

HTML5 (***H**yper**T**ext **M**arkup **L**anguage*, versión 5) es la quinta revisión importante del lenguaje básico de la World Wide Web, HTML. HTML5 [53] especifica dos variantes de sintaxis para HTML: un clásica (`text/html`), la variante conocida como *HTML5* y una variante, XHTML, conocida como sintaxis *XHTML5* que deberá ser servida como XML (XHTML) (application/xhtml+xml). Esta es la primera vez que HTML y XHTML se han desarrollado en paralelo.

Aunque todavía se encuentra en modo experimental, lo cual indica la misma W3C, ya es utilizado por múltiples desarrolladores Web por sus avances, mejoras y ventajas.

Al no ser reconocido en viejas versiones de navegadores por sus nuevas etiquetas, se le recomienda al usuario común actualizar a la versión más nueva, para poder disfrutar de todo el potencial que provee HTML5.

El desarrollo de este lenguaje de marcado es regulado por el Consorcio W3C.

9.5.1. Nuevos elementos

HTML5 establece una serie de nuevos elementos y atributos que reflejan el uso típico de los sitios web modernos. Algunos de ellos son técnicamente similares a las etiquetas `<div>` y `<span>`, pero tienen un significado semántico, como por ejemplo `<nav>` (bloque de navegación del sitio web) y `<footer>`. Otros elementos proporcionan nuevas funcionalidades a través de una interfaz estandarizada, como los elementos `<audio>` y `<video>`.

Se incorporan mejoras en el elemento `<canvas>`, capaz de renderizar en los navegadores más importantes (Mozilla, Chrome, Opera, Safari e IE) elementos 3D.

Algunos elementos de HTML 4.01 se han quedado obsoletos, incluyendo elementos puramente de presentación, como `<font>` y `<center>`, cuyos efectos son manejados por el CSS. También se ha renovado el énfasis en la importancia del scripting DOM para el comportamiento de la Web.

9.6. JSON

JSON [53], acrónimo de *JavaScript Object Notation*, es un formato ligero para el intercambio de datos. JSON es un subconjunto de la notación literal de objetos de JavaScript que no requiere el uso de XML.

La simplicidad de JSON ha dado lugar a la generalización de su uso, especialmente como alternativa a XML en AJAX. Una de las supuestas ventajas de JSON sobre XML como formato de intercambio de datos es que es mucho más sencillo escribir un analizador sintáctico (parser) de JSON. En JavaScript, un texto JSON se puede analizar fácilmente usando la función `eval()`, lo cual ha sido fundamental para que JSON haya sido aceptado por parte de la comunidad de desarrolladores AJAX, debido a la ubicuidad de JavaScript en casi cualquier navegador Web.

En la práctica, los argumentos a favor de la facilidad de desarrollo de analizadores o del rendimiento de los mismos son poco relevantes, debido a las cuestiones de seguridad que plantea el uso de `eval()` y el auge del procesamiento nativo de XML incorporado en los navegadores modernos. Por esa razón, JSON se emplea habitualmente en entornos donde el tamaño del flujo de datos entre cliente y servidor es de vital importancia (de aquí su uso por Yahoo, Google, etc, que atienden a millones de usuarios) cuando la fuente de datos es explícitamente de fiar y donde no es importante el no disponer de procesamiento XSLT para manipular los datos en el cliente.

Si bien es frecuente ver JSON posicionado frente a XML, también es frecuente el uso de JSON y XML en la misma aplicación. Por ejemplo, una aplicación de cliente que integra datos de Google Maps con datos meteorológicos en SOAP hace necesario soportar ambos formatos.

Cada vez hay más soporte de JSON mediante el uso de paquetes escritos por terceras partes. Entre ellos están ActionScript, C, C++, C#, ColdFusion, Common Lisp, Delphi,

Eiffel, Java, JavaScript, ML, Objective-C, Objective CAML, Perl, PHP, Python, Rebol, Ruby, Lua y Visual FoxPro.

En diciembre de 2005, Yahoo comenzó a dar soporte opcional de JSON en algunos de sus servicios Web.

El término JSON está altamente difundido en los medios de programación, sin embargo, es un término mal descrito ya que en realidad es solo una parte de la definición del estándar ECMA-262 en que está basado Javascript. De ahí que ni Yahoo, ni Google empleen JSON, sino LJS. Una de las cualidades intrínsecas de Javascript, denominada LJS (Literal Javascript), facilita el flujo de datos e incluso de funciones, para la cual, si son datos lo que ese trasfiere, no requiere la función `eval()`, como en el caso de XML. Todo lo referente a transferencia de datos en todos sus tipos, incluyendo arrays, booleans, integers, etc. no requieren de la función `eval()`, y es precisamente en eso en donde supera con creces JavaScript a XML, si se utiliza el LJS y no la incorrecta definición de JSON.

9.6.1. Uso de JSON

En teoría, es trivial analizar JSON en JavaScript usando la función `eval()` incorporada en el lenguaje. Por ejemplo:

```
miObjeto = eval('(' + json_datos + ')');
```

En la práctica, las consideraciones de seguridad por lo general recomiendan no usar eval sobre datos crudos y debería usarse un analizador JavaScript distinto para garantizar la seguridad. El analizador proporcionado por JSON.org usa `eval()` en su función de análisis, protegiéndola con una expresión regular de forma que la función sólo ve expresiones seguras.

Un ejemplo de acceso a datos JSON usando XMLHttpRequest es:

```
var http_request = new XMLHttpRequest();
var url = "http://example.net/jsondata.php"; // Esta URL debería devolver datos JSON

// Descarga los datos JSON del servidor.
http_request.onreadystatechange = handle_json;
http_request.open("GET", url, true);
http_request.send(null);

function handle_json() {
  if (http_request.readyState == 4) {
    if (http_request.status == 200) {
      var json_data = http_request.responseText;
      var the_object = eval("(" + json_data + ")");
    } else {
      alert("Ocurrio un problema con la URL.");
    }
    http_request = null;
  }
}
```

Obsérvese que el uso de XMLHttpRequest en este ejemplo no es compatible con todos los navegadores, aunque existen variaciones sintácticas para Internet Explorer, Opera, Safari, y navegadores basados en Mozilla.

También es posible usar elementos `<iframe>` ocultos para solicitar los datos de manera asíncrona, o usar peticiones `<form target="url_to_cgi_script" />`. Estos métodos eran los más habituales antes del comienzo del uso generalizado de XMLHttpRequest.

Hay una biblioteca2 para el framework .NET que exporta clases .NET con la sintaxis de JSON para la comunicación entre cliente y servidor, en ambos sentidos.

9.6.2. Ejemplo de JSON

A continuación se muestra un ejemplo simple de la definición de barra de menús usando JSON y XML.

JSON:

```
 {"menu": {
   "id": "file",
   "value": "File",
   "popup": {
     "menuitem": [
       {"value": "New", "onclick": "CreateNewDoc()"},
       {"value": "Open", "onclick": "OpenDoc()"},
       {"value": "Close", "onclick": "CloseDoc()"}
     ]
   }
 }
}
```

XML:

```
<menu id="file" value="File">
  <popup>
    <menuitem value="New" onclick="CreateNewDoc()" />
    <menuitem value="Open" onclick="OpenDoc()" />
    <menuitem value="Close" onclick="CloseDoc()" />
  </popup>
</menu>
```

Si bien los defensores de JSON a menudo recalcan que éste es más abreviado que XML, obsérvese que los dos ejemplos tienen unos 190 caracteres cuando se eliminan los espacios en blanco. Además, el uso de compresión GZIP para enviar los datos al navegador puede reducir la diferencia de tamaños entre ambos formatos. De hecho, cuando se usa GZIP sobre los ejemplos anteriores, el ejemplo en XML es más pequeño por 6 bytes. Si bien esto no es concluyente, muestra que es necesario experimentar con el conjunto de datos a tratar para determinar qué formato será más eficiente en términos de tamaño. JSON no es siempre más pequeño que XML.

El beneficio de JSON, entonces, no es que sea más pequeño a la hora de transmitir, sino que representa mejor la estructura de los datos y requiere menos codificación y procesamiento.

9.6.3. Comparación con XML y otros lenguajes de marcado

XML goza de mayor soporte y ofrece muchas más herramientas de desarrollo (tanto en el lado del cliente como en el lado del servidor). Hay muchos analizadores JSON en el lado del servidor, existiendo al menos un analizador para la mayoría de los entornos. En algunos lenguajes, como JAVA o PHP, hay diferentes implementaciones donde escoger. En JavaScript, el análisis es posible de manera nativa con la función `eval()`. Ambos formatos carecen de un mecanismo para representar grandes objetos binarios.

Con independencia de la comparación con XML, JSON puede ser muy compacto y eficiente si se usa de manera efectiva. Recibe los listados de directorio como JSON desde el servidor. Esta aplicación de búsqueda está permanentemente consultando al servidor por nuevos directorios, y es notablemente rápida, incluso sobre una conexión lenta.

Los entornos en el servidor normalmente requieren que se incorpore una función u objeto analizador de JSON. Algunos programadores, especialmente los familiarizados con el lenguaje C, encuentran JSON más natural que XML, pero otros desarrolladores encuentran su escueta notación algo confusa, especialmente cuando se trata de datos fuertemente jerarquizados o anidados muy profundamente.

Hay más comparaciones entre JSON y XML en *JSON.org*

YAML es un superconjunto de JSON que trata de superar algunas de las limitaciones de éste. Aunque es significativamente más complejo, aún puede considerarse como ligero. El lenguaje de programación Ruby utiliza YAML como el formato de serialización por defecto. Así pues, es posible manejar JSON con bastante sencillez.

9.7. SQL

9.7.1. Introducción

El Lenguaje de Consulta Estructurado (Structured Query Language [53]) es un lenguaje declarativo de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones sobre las mismas. Aúna características del álgebra y el cálculo relacional permitiendo lanzar consultas con el fin de recuperar información de interés de una base de datos, de una forma sencilla. Es un lenguaje de cuarta generación (4GL).

9.7.1.1. Orígenes y evolución

Los orígenes del SQL están ligados a los de las bases de datos relacionales. En 1970 Codd propone el modelo relacional y asociado a este un sublenguaje de acceso a los datos basado en el cálculo de predicados. Basándose en estas ideas los laboratorios de IBM definen el lenguaje SEQUEL (Structured English QUEry Language) que más tarde sería ampliamente implementado por el SGBD experimental System R, desarrollado en 1977

también por IBM. Sin embargo, fue Oracle quien lo introdujo por primera vez en 1979 en un programa comercial.

El SEQUEL terminaría siendo el predecesor de SQL, siendo éste una versión evolucionada del primero. El SQL pasa a ser el lenguaje por excelencia de los diversos SGBD relacionales surgidos en los años siguientes y es por fin estandarizado en 1986 por el ANSI, dando lugar a la primera versión estándar de este lenguaje, el SQL-86 o SQL1. Al año siguiente este estándar es también adoptado por la ISO.

Sin embargo este primer estándar no cubre todas las necesidades de los desarrolladores e incluye funcionalidades de definición de almacenamiento que se consideraron suprimir. Así que en 1992 se lanza un nuevo estándar ampliado y revisado del SQL llamado SQL-92 o SQL2.

En la actualidad el SQL es el estándar de facto de la inmensa mayoría de los SGBD comerciales. Y, aunque la diversidad de añadidos particulares que incluyen las distintas implementaciones comerciales del lenguaje es amplia, el soporte al estándar SQL-92 es general y muy amplio.

9.7.2. Structured Query Language

9.7.2.1. Características generales

El SQL es un lenguaje de acceso a bases de datos que explota la flexibilidad y potencia de los sistemas relacionales permitiendo gran variedad de operaciones sobre los mismos. Es un lenguaje declarativo de alto nivel o de no procedimiento, que gracias a su fuerte base teórica y su orientación al manejo de conjuntos de registros, y no a registros individuales, permite una alta productividad en codificación. De esta forma una sola sentencia puede equivaler a uno o más programas que utilizasen un lenguaje de bajo nivel orientado a registro.

9.7.2.2. Funcionalidad

El SQL proporciona funcionalidad más allá de la simple consulta (o recuperación) de datos. Asume el papel de lenguaje de definición de datos (LDD), lenguaje de definición de vistas (LDV) y lenguaje de manipulación de datos (LMD). Además permite la concesión y denegación de permisos, la implementación de restricciones de integridad y controles de transacción, y la alteración de esquemas. Las primeras versiones del SQL incluían funciones propias de lenguaje de definición de almacenamiento (LDA) pero fueron suprimidas en los estándares más recientes con el fin de mantener el lenguaje sólo a nivel conceptual y externo.

9.7.2.3. Modos de uso

El SQL permite fundamentalmente dos modos de uso:

- Un uso interactivo, destinado principalmente a los usuarios finales avanzados u ocasionales, en el que las diversas sentencias SQL se escriben y ejecutan en línea de comandos, o un entorno semejante.

- Un uso integrado, destinado al uso por parte de los programadores dentro de programas escritos en cualquier lenguaje de programación anfitrión. En este caso el SQL asume el papel de sublenguaje de datos.

En el caso de hacer un uso embebido del lenguaje se pueden utilizar dos técnicas alternativas de programación. En una de ellas, en la que el lenguaje se denomina SQL estático, las sentencias utilizadas no cambian durante la ejecución del programa. En la otra, donde el lenguaje recibe el nombre de SQL dinámico, se produce una modificación total o parcial de las sentencias en el transcurso de la ejecución del programa. La utilización de SQL dinámico permite mayor flexibilidad y mayor complejidad en las sentencias, pero como contrapunto se obtiene una eficiencia menor y el uso de técnicas de programación más complejas en el manejo de memoria y variables.

9.7.2.4. Optimización

Como ya se ha comentado con anterioridad, y como suele ser común en los lenguajes de acceso a bases de datos de alto nivel, el SQL es un lenguaje declarativo. Es decir, que especifica qué es lo que se quiere y no cómo conseguirlo, por lo que una sentencia no establece explícitamente un orden de ejecución. Este orden de ejecución interno de una sentencia puede afectar gravemente a la eficiencia del SGBD, por lo que se hace necesario que éste lleve a cabo una optimización antes de la ejecución de la misma. Muchas veces, el uso de índices acelera una instrucción de consulta, pero ralentiza la actualización de los datos, dependiendo del uso de la aplicación, se priorizará el acceso indexado o una rápida actualización de la información. La optimización difiere sensiblemente en cada motor de base de datos y depende de muchos factores. Existe una ampliación de SQL conocida como FSQL (Fuzzy SQL, SQL difuso) que permite el acceso a bases de datos difusas, usando la lógica difusa. Este lenguaje ha sido implementado a nivel experimental y está evolucionando rápidamente.

9.7.2.5. Lenguaje de Definición de datos

El lenguaje de Definición de datos, en inglés Data Definition Language (DDL), es el que se encarga de la modificación de la estructura de los objetos de la base de datos. Existen cuatro operaciones básicas: CREATE, ALTER, DROP y TRUNCATE.

9.7.2.5.1. CREATE

Este comando crea un objeto dentro de la base de datos. Puede ser una tabla, vista, índice, trigger, función, procedimiento o cualquier otro objeto que el motor de la base de datos soporte.

Ejemplo 1 (creación de una tabla):

```
CREATE TABLE TABLA_NOMBRE (
 my_field1   INT           UNSIGNED,
 my_field2   VARCHAR (50),
 my_field3   DATE          NOT NULL,
 PRIMARY KEY (my_field1, my_field2)
```

9.7.2.5.2. ALTER

Este comando permite modificar la estructura de un objeto. Se pueden agregar/quitar campos a una tabla, modificar el tipo de un campo, agregar/quitar índices a una tabla, modificar un trigger, etc.

Ejemplo 1 (agregar columna a una tabla):

```
ALTER TABLE TABLA_NOMBRE (
 ADD NUEVO_CAMPO INT UNSIGNED
)
```

9.7.2.5.3. DROP

Este comando elimina un objeto de la base de datos. Puede ser una tabla, vista, índice, trigger, función, procedimiento o cualquier otro objeto que el motor de la base de datos soporte. Se puede combinar con la sentencia ALTER.

Ejemplo 1:

```
DROP TABLE TABLA_NOMBRE
```

Ejemplo 2:

```
ALTER TABLE TABLA_NOMBRE
(
   DROP COLUMN CAMPO_NOMBRE1
)
```

9.7.2.5.4. TRUNCATE

Este comando trunca todo el contenido de una tabla. La ventaja sobre el comando DELETE es que si se desea borrar todo el contenido de la tabla, resulta mucho más rápido, especialmente si la tabla es muy grande, pero la desventaja es que TRUNCATE sólo sirve cuando se quieren eliminar absolutamente todos los registros, ya que no se permite la cláusula WHERE. Si bien, en un principio, esta sentencia parecería ser DML (Lenguaje de Manipulación de Datos), es en realidad una DDL, ya que internamente, el comando truncate borra la tabla y la vuelve a crear y no ejecuta ninguna transacción.

Ejemplo 1:

```
TRUNCATE TABLE TABLA_NOMBRE
```

9.7.3. MySQL

MySQL [66] es un sistema de gestión de base de datos, multihilo y multiusuario con más de seis millones de instalaciones. MySQL AB desarrolla MySQL como software libre en un esquema de licenciamiento dual. Por un lado lo ofrece bajo la GNU GPL, pero, empresas que quieran incorporarlo en productos privativos pueden comprar a la empresa una licencia que les permita ese uso.

Está desarrollado en su mayor parte en ANSI C.

Al contrario de proyectos como el Apache, donde el software es desarrollado por una comunidad pública, y el copyright del código está en poder del autor individual, MySQL está poseído y patrocinado por una empresa privada, que posee el copyright de la mayor parte del código. Esto es lo que posibilita el esquema de licenciamiento anteriormente mencionado. Además de la venta de licencias privativas, la compañía ofrece soporte y servicios. Para sus operaciones contratan trabajadores alrededor del mundo que colaboran vía Internet. MySQL AB fue fundado por David Axmark, Allan Larsson, y Michael Widenius.

9.7.3.1. Historia del proyecto

SQL (Lenguaje de Consulta Estructurado) fue comercializado por primera vez en 1981 por IBM, el cual fue presentado a ANSI y desde ese entonces ha sido considerado como un estándar para las bases de datos relacionales. Desde 1986, el estándar SQL ha aparecido en diferentes versiones como por ejemplo: SQL:92, SQL:99, SQL:2003. MySQL es una idea originaria de la empresa opensource MySQL AB establecida inicialmente en Suecia en 1995 y cuyos fundadores son David Axmark, Allan Larsson, y Michael "Monty" Widenius. El objetivo que persigue esta empresa consiste en que MySQL cumpla el estándar SQL, pero sin sacrificar velocidad, fiabilidad o usabilidad.

Michael Widenius en la década de los 90 trató de usar mSQL para conectar las tablas usando rutinas de bajo nivel ISAM, sin embargo, mSQL no era rápido y flexible para sus necesidades. Esto lo conllevó a crear una API SQL denominada MySQL para bases de datos muy similar a la de mSQL pero más portable.

La procedencia del nombre de MySQL no es clara. Por más de 10 años, las herramientas han mantenido el prefijo My. También, se cree que tiene relación con el nombre de la hija del cofundador Monty Widenius quien se llama My.

Por otro lado, el nombre del delfín de MySQL es Sakila y fue seleccionado por los fundadores de MySQL AB en el concurso “Name the Dolphin”. Este nombre fue enviado por Ambrose Twebaze, un desarrollador de Opensource Africano, derivado del idioma SiSwate, el idioma local de Swaziland y corresponde al nombre de una ciudad en Arusha, Tanzania, cerca de Uganda la ciudad origen de Ambrose.

9.7.3.2. Lenguajes de programación

Existen varias APIs que permiten, a aplicaciones escritas en diversos lenguajes de programación, acceder a las bases de datos MySQL, incluyendo C, C++, C#, Pascal, Delphi (via dbExpress), Eiffel, Smalltalk, Java (con una implementación nativa del driver de Java), Lisp, Perl, PHP, Python, Ruby, REALbasic (Mac), FreeBASIC, y Tcl; cada uno de estos utiliza una API específica. También existe un interfaz ODBC, llamado MyODBC que permite a cualquier lenguaje de programación que soporte ODBC comunicarse con las bases de datos MySQL.

9.7.3.3. Aplicaciones

MySQL es muy utilizado en aplicaciones web como MediaWiki o Drupal, en plataformas (Linux/Windows-Apache-MySQL-PHP/Perl/Python), y por herramientas de seguimiento de errores como Bugzilla. Su popularidad como aplicación web está muy ligada a PHP, que a menudo aparece en combinación con MySQL. MySQL es una base

de datos muy rápida en la lectura cuando utiliza el motor no transaccional MyISAM, pero puede provocar problemas de integridad en entornos de alta concurrencia en la modificación. En aplicaciones Web hay baja concurrencia en la modificación de datos y en cambio el entorno es intensivo en lectura de datos, lo que hace a MySQL ideal para este tipo de aplicaciones.

9.7.3.4. Especificaciones

9.7.3.4.1. Plataformas

MySQL funciona sobre múltiples plataformas, incluyendo AIX, BSD, FreeBSD, HP-UX, GNU/Linux, Mac OS X, NetBSD, Novell Netware, OpenBSD, OS/2 Warp, QNX, SGI IRIX, Solaris, SunOS, SCO OpenServer, SCO UnixWare, Tru64, Windows 95, Windows 98, Windows NT, Windows 2000, Windows XP, Windows Vista y otras versiones de Windows. También existe MySQL para OpenVMS en http://www.pi-net.dyndns.org/anonymous/kits/.

9.7.3.4.2. Características de la versión 5.0.22

- Un amplio subconjunto de ANSI SQL 99, y varias extensiones.
- Soporte a multiplataforma
- Procedimientos almacenados
- Triggers
- Cursors
- Vistas actualizables
- Soporte a VARCHAR
- INFORMATION_SCHEMA
- Modo Strict
- Soporte X/Open XA de transacciones distribuidas; transacción en dos fases como parte de esto, utilizando el motor InnoDB de Oracle
- Motores de almacenamiento independientes (MyISAM para lecturas rápidas, InnoDB para transacciones e integridad referencial)
- Transacciones con los motores de almacenamiento InnoDB, BDB Y Cluster; puntos de recuperación(savepoints) con InnoDB
- Soporte para SSL
- Query caching
- Sub-SELECTs (o SELECTs anidados)
- Replication with one master per slave, many slaves per master, no automatic support for multiple masters per slave.
- indexing y buscando campos de texto completos usando el motor de almacenamiento MyISAM
- Embedded database library

- Soporte completo para Unicode
- Conforme a las reglas ACID usando los motores InnoDB, BDB y Cluster
- Shared-nothing clustering through MySQL Cluster

9.7.3.4.3. Características adicionales

- Usa GNU Automake, Autoconf, y Libtool para portabilidad
- Uso de multihilos mediante hilos del kernel.
- Usa tablas en disco b-tree para búsquedas rápidas con compresión de índice
- Tablas hash en memoria temporales
- El código MySQL se prueba con Purify (un detector de memoria perdida comercial) así como con Valgrind, una herramienta GPL
- Completo soporte para operadores y funciones en cláusulas select y where.
- Completo soporte para cláusulas group by y order by, soporte de funciones de agrupación
- Seguridad: ofrece un sistema de contraseñas y privilegios seguro mediante verificación basada en el host y el tráfico de contraseñas está encriptado al conectarse a un servidor.
- Soporta gran cantidad de datos. MySQL Server tiene bases de datos de hasta 50 millones de registros.
- Se permiten hasta 64 índices por tabla (32 antes de MySQL 4.1.2). Cada índice puede consistir desde 1 hasta 16 columnas o partes de columnas. El máximo ancho de límite son 1000 bytes (500 antes de MySQL 4.1.2).
- Los clientes se conectan al servidor MySQL usando sockets TCP/IP en cualquier plataforma. En sistemas Windows se pueden conectar usando named pipes y en sistemas Unix usando ficheros socket Unix.
- En MySQL 5.0, los clientes y servidores Windows se pueden conectar usando memoria compartida.
- MySQL contiene su propio paquete de pruebas de rendimiento proporcionado con el código fuente de la distribución de MySQL

3.3.4.4. Características (versión 4.0)

Inicialmente, MySQL carecía de elementos considerados esenciales en las bases de datos relacionales, tales como integridad referencial y transacciones. A pesar de ello, atrajo a los desarrolladores de páginas Web con contenido dinámico, justamente por su simplicidad; aquellos elementos faltantes fueron llenados por la vía de las aplicaciones que la utilizan.

Poco a poco los elementos de los que carecía MySQL están siendo incorporados tanto por desarrollos internos, como por desarrolladores de software libre. Entre las características disponibles en las últimas versiones se puede destacar:

- Amplio subconjunto del lenguaje SQL. Algunas extensiones son incluidas igualmente.
- Disponibilidad en gran cantidad de plataformas y sistemas.

- Diferentes opciones de almacenamiento según si se desea velocidad en las operaciones o el mayor número de operaciones disponibles.
- Transacciones y claves foráneas.
- Conectividad segura.
- Replicación.
- Búsqueda e indexación de campos de texto.

MySQL es un sistema de administración de bases de datos. Una base de datos es una colección estructurada de tablas que contienen datos. Esta puede ser desde una simple lista de compras a una galería de pinturas o el vasto volumen de información en un red corporativa. Para agregar, acceder a y procesar datos guardados en un computador, usted necesita un administrador como MySQL Server. Dado que los computadores son muy buenos manejando grandes cantidades de información, los administradores de bases de datos juegan un papel central en computación, como aplicaciones independientes o como parte de otras aplicaciones.

MySQL es un sistema de administración relacional de bases de datos. Una base de datos relacional archiva datos en tablas separadas en vez de colocar todos los datos en un gran archivo. Esto permite velocidad y flexibilidad. Las tablas están conectadas por relaciones definidas que hacen posible combinar datos de diferentes tablas sobre pedido.

MySQL es software de fuente abierta. Fuente abierta significa que es posible para cualquier persona usarlo y modificarlo. Cualquier persona puede bajar el código fuente de MySQL y usarlo sin pagar. Cualquier interesado puede estudiar el código fuente y ajustarlo a sus necesidades. MySQL usa el GPL (GNU General Public License) para definir qué puede hacer y que no puede hacer con el software en diferentes situaciones. Si usted no se ajusta al GLP o requiere introducir código MySQL en aplicaciones comerciales, usted puede comprar una versión comercial licenciada.

9.7.3.5. Mejoras actuales

El mapa de ruta de MySQL 5.1 indica soporte para:

- Particionado de la base de datos
- Backup en línea para todos los motores de almacenamiento
- Replicación segura
- Restricciones a nivel de columna
- Planificación de eventos
- Funciones XML

9.7.3.6. Características distintivas

Las siguientes características son implementadas únicamente por MySQL:

- Múltiples motores de almacenamiento (MyISAM, Merge, InnoDB, BDB, Memory/heap, MySQL Cluster, Federated, Archive, CSV, Blackhole y Example en 5.x), permitiendo al usuario escoger la que sea más adecuada para cada tabla de la base de datos.

- Agrupación de transacciones, reuniendo múltiples transacciones de varias conexiones para incrementar el número de transacciones por segundo.

9.7.3.7. Tipos de compilación del servidor

- Hay tres tipos de compilación del servidor MySQL:
- Estándar: Los binarios estándar de MySQL son los recomendados para la mayoría de los usuarios, e incluyen el motor de almacenamiento InnoDB.
- Max (No se trata de MaxDB, que es una cooperación con SAP): Los binarios incluyen características adicionales que no han sido lo bastante probadas o que normalmente no son necesarias.
- MySQL-Debug: Son binarios que han sido compilados con información de depuración extra. No debe ser usada en sistemas en producción porqué el código de depuración puede reducir el rendimiento.

9.7.3.8. Especificaciones del código fuente

MySQL está escrito en una mezcla de C y C++. Hay un documento que describe algunas de sus estructuras internas en http://dev.mysql.com/doc/internals/en/ (en inglés).

9.7.3.9. Desarrollo del proyecto

El desarrollo de MySQL se fundamenta en el trabajo de los desarrolladores contratados por la empresa MySQL AB quienes se encargan de dar soporte a los socios comerciales y usuarios de la comunidad MySQL y dar solución a los problemas encontrados por los usuarios. Los usuarios o miembros de la comunidad MySQL pueden reportar bugs revisando el manual en línea que contiene las soluciones a problemas encontrados; el historial de cambios; la base de datos bugs que contiene bugs reportados y solucionados y en las listas de correo MySQL.

A través de las listas de correo los usuarios pueden enviar preguntas y éstas serán contestadas por desarrolladores brindándoles soporte. Otras funcionalidades de las listas de correo son:

- Anuncios: informan sobre nuevas versiones de MySQL y programas relacionados.
- MySQL: lista principal de discusión de MySQL.
- Bugs: permite a la gente buscar y arreglar bugs.
- Temas internos: para gente que trabaja con el código de MySQL. Es el fórum para discutir sobre el desarrollo de MySQL.
- MySQLdoc: para las personas que trabaja en documentación.
- Pruebas de rendimiento: para gente interesada en temas de rendimiento no solo de MySQL, sino de otros motores de bases de datos.
- Empaquetadores: para discusiones sobre empaquetamiento y distribución de MySQL.
- Java: discusiones sobre MySQL Server y Java.

- Otras listas de correo son: MyODBC, Herramientas GUI, Cluster, Dotnet, PlusPlus y Perl.

Adicional a las listas de correo, se encuentra el soporte de IRC de la comunidad MySQL. Además, hay soporte a través de foros agrupados en categorías tales como: Migración, Uso de MySQL, Conectores MySQL, Tecnología MySQL y Negocios

9.7.3.10. Estructuras organizativas/asociativas o de decisión

La dirección y el patrocinio de los proyectos MySQL están a cargo de la empresa MySQL AB quien posee el copyrigth del código fuente MySQL, su logo y marca registrada. MySQL, Inc. y MySQL GmbH son ejemplos de empresas subsidiarias de MySQL AB. Están establecidas en los Estados Unidos y Alemania respectivamente. MySQL AB, cuenta con más de 200 empleados en más de 20 países y funcionan bajo la estrategia de teletrabajo.

9.7.3.11. Industria relacionada

La industria radica en la venta de productos software y de algunos servicios relacionados a numerosas empresas que utilizan estos productos.

- MySQL AB clasifica los productos así:
- MySQL Enterprise: incluye MySQL Enterprise Server , Monitoreo de la red MySQL, servicios de consulta y soporte de producción MySQL
- MYSQL Cluster
- MySQL Embedded Database
- MySQL Drivers: para JDBC, ODBC y .Net
- MySQL Tools: MySQL Administrator, MySQL Query Browser, and the MySQL Migration Toolkit
- MaxDB: MaxDB es una base de datos open source certificada para SAP/R3

Los ingresos de esta empresa por la venta de licencias privativas de sus productos y los servicios suma los U$12 millones.

9.7.3.12. MySQL en cifras

Según las cifras del fabricante, existirían más de seis millones de copias de MySQL funcionando en la actualidad, lo que supera la base instalada de cualquier otra herramienta de bases de datos.

El tráfico del sitio web de MySQL AB superó en 2004 al del sitio de IBM.

9.7.3.13. Qué licencia utilizar

La licencia GNU GPL de MySQL obliga a distribuir cualquier producto derivado (aplicación) bajo esa misma licencia. Si un desarrollador desea incorporar MySQL en su producto pero no desea distribuirlo bajo licencia GNU GPL, puede adquirir la licencia comercial de MySQL que le permite hacer justamente eso.

9.7.3.14. Estado actual

La serie en desarrollo de MySQL Server actualmente, es la 5.1 al cual se añaden nuevas características en relación a la serie 5.0. La serie de producción actual de MySQL es 5.0, cuya penúltima versión estable es la 5.0.26 lanzada en octubre de 2006. Actualmente, se puede descargar la serie 5.0.27. La serie de producción anterior fue la 4.1, cuya versión estable es 4.1.7 lanzada en octubre de 2004. A estas versiones de producción sólo se arreglan problemas, es decir, ya no se añaden nuevas características. Y a las versiones anteriores solamente se les corrigen bugs críticos.

9.7.3.15. Usuarios destacados

Amazon.com

- Cox Communications - La cuarta televisión por cable más importante de EEUU, tienen más de 3.600 tablas y aproximadamente dos millones de inserciones cada hora.
- Craigslist.
- Digg - Sitio de noticias.
- Google - Para el motor de búsqueda de la aplicación AdWords.
- LiveJournal - Cerca de 300 millones de páginas servidas cada día.
- NASA.
- Omniture.
- RightNow.
- Sabre, y su sistema de reserva de viajes Travelocity.
- Slashdot - con cerca de 50 millones de páginas servidas cada día.
- Yahoo! - para muchas aplicaciones críticas.
- Nokia, usa un cluster MySQL para mantener información en tiempo real sobre usuarios de redes de móviles.
- Flickr, usa MySQL para gestionar millones de fotos y usuarios.
- NetQOS, usa MySQL para la gestión de algunas de las redes más grandes del mundo como las de Chevron, American Express y Boeing.
- Universidad de Piura | Campus Lima, para su sistema académico denominado SIAD.
- CNET Networks.
- Friendster, sirve más de 85 millones de páginas dinámicas cada día.
- Wikipedia, sirve más de 200 millones de consultas y 1,2 millones de actualizaciones cada día, con picos de 11.000 consultas por segundo.

9.8. Sistemas NoSQL

En informática, NoSQL [53] (a veces llamado "no sólo SQL") es una amplia clase de sistemas de gestión de bases de datos que difieren del modelo clásico del sistema de gestión de bases de datos relacionales (RDBMS) en aspectos importantes, el más destacado que no usan SQL como el principal lenguaje de consultas. Los datos almacenados no requieren estructuras fijas como tablas, normalmente no soportan operaciones JOIN, ni garantizan completamente ACID (atomicidad, coherencia, aislamiento y durabilidad), y habitualmente escalan bien horizontalmente.

Por lo general, los investigadores académicos se refieren a este tipo de bases de datos como almacenamiento estructurado, término que abarca también las bases de datos relacionales clásicas. A menudo, las bases de datos NoSQL se clasifican según su forma de almacenar los datos, y comprenden categorías como clave-valor, las implementaciones de BigTable, bases de datos documentales, y Bases de datos orientadas a grafos.

Los sistemas de bases de datos NoSQL crecieron con las principales compañías de Internet, como Google, Amazon, Twitter y Facebook. Estas tenían que enfrentarse a desafíos con el tratamiento de datos que las tradicionales RDBMS no solucionaban. Con el crecimiento de la web en tiempo real existía una necesidad de proporcionar información procesada a partir de grandes volúmenes de datos que tenían unas estructuras horizontales más o menos similares. Estas compañías se dieron cuenta que el rendimiento y sus propiedades de tiempo real eran más importantes que la coherencia, en la que las bases de datos relacionales tradicionales dedicaban una gran cantidad de tiempo de proceso.

En ese sentido, a menudo, las bases de datos NoSQL están altamente optimizadas para las operaciones recuperar y agregar, y normalmente no ofrecen mucho más que la funcionalidad de almacenar los registros (p.ej. almacenamiento clave-valor). La pérdida de flexibilidad en tiempo de ejecución, comparado con los sistemas SQL clásicos, se ve compensada por ganancias significativas en escalabilidad y rendimiento cuando se trata con ciertos modelos de datos.

9.8.1. Historia del término

Carlo Strozzi usó el término NoSQL en 1998 para referirse a su base de datos. Era una base de datos open-source, ligera, que no ofrecía un interface SQL, pero sí seguía el modelo relacional (Strozzi sugiere que, ya que el actual movimiento NoSQL "Se sale completamente del modelo relacional, debería, por tanto, haberse llamado 'NoREL', o algo así.")

Eric Evans, un empleado de Rackspace, reintrodujo el término NoSQL cuando Johan Oskarsson de Last.fm quiso organizar un evento para discutir bases de datos distribuidas de código abierto. El nombre intentaba recoger el número creciente de bases de datos no relacionales y distribuidas que no garantizaban ACID, atributo clave en las RDBMS clásicas.

9.8.2. Arquitectura

Típicamente las bases de datos relacionales modernas han mostrado poca eficiencia en determinadas aplicaciones que usan los datos de forma intensiva, incluyendo el indexado de un gran número de documentos, la presentación de páginas en sitios que tienen gran tráfico, y en sitios de streaming audiovisual. Las implementaciones típicas de RDBMS se han afinado o bien para una cantidad pequeña pero frecuente de lecturas y escrituras o para un gran conjunto de transacciones que tiene pocos accesos de escritura. Por otro lado NoSQL puede servir gran cantidad de carga de lecturas y escrituras.

Implementaciones de NoSQL usadas en el mundo real incluyen los 3TB de los marcadores verdes de Digg (indicados para señalar las historias votadas por otros en la red social; aunque duró menos de 3 meses y fue abandonado). Los 6 TB de la base de datos del "ENSEMBLE" de la Comisión Europea usado en los modelos de comparación y calidad del aire, y en los 50 TB de la búsqueda de la bandeja de entrada de Facebook.

Las arquitecturas NoSQL frecuentemente aportan escasas garantías de consistencia, tales como consistencia de eventos o transaccional restringida a ítems únicos de datos. Algunos sistemas, sin embargo, aportan todas las garantías de los sistemas ACID en algunas instancias añadiendo una capa intermedia (como por ejemplo, AppScale o CloudTPS). Hay dos sistemas que han sido desplegados y que aportan aislamiento snapshot para almacenamientos de columna: El sistema Percolator de Google (basado en el sistema BigTable) y el sistema transaccional de Hbase desarrollado por la universidad de Waterloo. Estos sistemas, desarrollados de forma independiente, usan conceptos similares para conseguir transacciones ACID distribuídas de múltiples filas con garantías de aislamiento snapshot para el sistema subyacente de almacenamiento en esa columna, sin sobrecarga extra en la gestión de los datos, despliegue en el sistema de middleware, ni mantenimiento introducido por la capa de middleware.

Bastantes sistemas NoSQL emplean una arquitectura distribuida, manteniendo los datos de forma redundante en varios servidores, usando frecuentemente una tabla hash distribuida. De esta forma, el sistema puede realmente escalar añadiendo más servidores, y el fallo en un servidor puede ser tolerado.

Algunos defensores de NoSQL promueven interfaces simples tales como los arrays asociativos o los pares clave-valor. Otros sistemas, tales como las bases de datos nativas en XML, promueven el soporte del estándar Xquery. Los sistemas más novedosos tales como CloudTPS también soportan unión de queries.

9.8.3. Ventajas

Se detectan las siguientes ventajas:

- Estos sistemas responden a las necesidades de escalabilidad horizontal que tienen cada vez más empresas.
- Pueden manejar enormes cantidades de datos.
- No generan cuellos de botella.
- Escalamiento sencillo.
- Diferentes bases de datos NoSQL para diferentes proyectos.

- Se ejecutan en clústeres de máquinas baratas.

9.9. Mongo

Una vez comentado de NoSQL y sus atractivas ventajas para algunos proyectos llega el turno de hablar propiamente de uno de ellos, Mongo [61]. A lo largo de este apartado se ofrece una pequeña descripción de lo que aporta este sistema de almacenamiento de datos.

9.9.1. Descripción y licencia

MongoDB es un sistema de base de datos multiplataforma orientado a documentos, de esquema libre. Como ya os expliqué, esto significa que cada entrada o registro puede tener un esquema de datos diferente, con atributos o "columnas" que no tienen por qué repetirse de un registro a otro. Está escrito en C++, lo que le confiere cierta cercanía a los recursos de hardware de la máquina, de modo que es bastante rápido a la hora de ejecutar sus tareas. Además, está licenciado como GNUAGPL 3.0, de modo que se trata de un software de licencia libre. Funciona en sistemas operativos Windows, Linux, OS X y Solaris.

Las características que más destacaría de MongoDB son su velocidad y su rico pero sencillo sistema de consulta de los contenidos de la base de datos. Se podría decir que alcanza un balance perfecto entre rendimiento y funcionalidad, incorporando muchos de los tipos de consulta que utilizaríamos en nuestro sistema relacional preferido, pero sin sacrificar en rendimiento.

9.9.2. Terminología básica en MongoDB

En MongoDB, cada registro o conjunto de datos se denomina documento. Los documentos se pueden agrupar en colecciones, las cuales se podría decir que son el equivalente a las tablas en una base de datos relacional (sólo que las colecciones pueden almacenar documentos con muy diferentes formatos, en lugar de estar sometidos a un esquema fijo). Se pueden crear índices para algunos atributos de los documentos, de modo que MongoDB mantendrá una estructura interna eficiente para el acceso a la información por los contenidos de estos atributos.

9.9.3. Formato de los documentos e ideas para la organización de datos

Los distintos documentos se almacenan en formato BSON, o Binary JSON, versión modificada de JSON que permite búsquedas rápidas de datos. Para hacerse una idea, BSON guarda de forma explícita las longitudes de los campos, los índices de los arrays, y demás información útil para el escaneo de datos. Es por esto que, en algunos

casos, el mismo documento en BSON ocupa un poco más de espacio de lo que ocuparía de estar almacenado directamente en formato JSON. Pero una de las ideas claves en los sistemas NoSQL es que el almacenamiento es barato, y es mejor aprovecharlo si así se introduce un considerable incremento en la velocidad de localización de información dentro de un documento.

Sin embargo, en la práctica, nunca se ve el formato en que verdaderamente se almacenan los datos, y se trabaja siempre sobre un documento en JSON tanto al almacenar como al consultar información. Un ejemplo de un documento en MongoDB podría ser perfectamente:

```
{
    "_id"          : "4da2c0e2e999fb56bf000002"
    "title"        : "Una introducción a MongoDB",
    "body"         : "Lorem ipsum dolor sit amet…",
    "published_at" : "2011-05-09T18:17:07-07:00",
    "author_info"  : {
                       "_id"  : "4dc8919331c0c00001000002"
                       "name" : "Carlos Paramio"
                     },
    "tags"         : ["MongoDB", "NoSQL", "Bases de datos"]
    "comments"     : [
                       {
                         "author_info" : { "name" : "Jorge Rubira",
"email" : "email1@example.com" },
                         "body"        : "Test",
                         "created_at"  : "2011-05-10T10:14:01-07:00"
                       },
                       {
                         "author_info" : { "name" : "Txema Rodríguez",
"email" : "email2@example.com" },
                         "body"        : "Otro test",
                         "created_at"  : "2011-05-10T10:14:09-07:00"
                       }
                     ]
    "liked_by"     : ["4d7cf768e999fb67c0000001",
"4da34c62ba875a19d4000001"]
}
```

Como podemos adivinar, se representa la manera en que podrían almacenarse los datos correspondientes a un post de un blog. Los atributos "_id" (o clave principal) pueden tener el formato que se desee, aunque MongoDB utiliza un valor parecido a un UUID en hexadecimal por defecto si no se ha especificado ninguno. A pesar de parecer un valor completamente aleatorio (aunque ya sabemos que la aleatoriedad real no existe en informática), utilizan como base una semilla basada en la MAC de la interfaz de red de la máquina (y otros detalles de la misma) para evitar que dos máquinas diferentes puedan generar el mismo valor para la clave de un documento. Y los primeros bytes corresponden a una marca de tiempo, de modo que las claves se ordenan de forma natural por orden de creación (o casi, pues está claro que las distintas máquinas corriendo MongoDB deben tener la fecha y hora sincronizadas) sin tener que mirar cuál fue el último valor usado. Una solución inteligente, bastante más eficiente que un campo autonumérico, en especial para evitar que una máquina bloquee la inserción de registros en una colección sólo para asegurarse que no se dan condiciones de carrera al intentar dos máquinas diferentes escribir un documento con el mismo valor para "_id". Este atributo es el único obligatorio para un documento.

Las etiquetas y los comentarios están en el propio documento que representa al post, en lugar de guardarlos en colecciones separadas y utilizar claves foráneas para referenciar a los mismos. Sin embargo, en el atributo "liked_by" sí que se guarda una relación de claves, que corresponden a los usuarios que han marcado el post como que les ha gustado. Utilizar una forma u otra dependerá de las necesidades de acceso a estos datos En este caso, por ejemplo, se sabe que no se va a pintar información sobre los usuarios que han marcado un post con un "me gusta", pero sí se quiere ver cuántos lo han marcado así, o si el usuario actual ya lo ha marcado o no, con lo que almacenar únicamente las claves de esos usuarios y guardar su información personal detallada en otra colección es lo más conveniente.

Por supuesto, no es necesario pedir a MongoDB que nos devuelva todo el documento cada vez que lo consultamos. Si por ejemplo vamos a pintar únicamente un listado de posts recientes, seguramente sea suficiente obtener el atributo "title", con los documentos ordenados por "published_at". Así, se ahorra ancho de banda entre el motor de base de datos y la aplicación, al mismo tiempo que no hay que instanciar todo el documento. Además, si tenemos muchos miles de visitantes, el atributo "liked_by" podría llegar a crecer bastante. Aunque el tamaño de un documento de MongoDB puede llegar hasta los 16 Megabytes, con lo que se puede almacenar bastante información dentro de un único documento sin necesidad de utilizar referencias, si así lo necesitamos. En caso de que se tuviera que almacenar mucho más, habría que optar por utilizar otro esquema.

A veces, toca desnormalizar para poder tener a mano la información necesaria a la hora de pintar un post. Es por eso que en el atributo "author_info" se utiliza una versión intermedia: Si bien se tiene la clave principal del usuario que ha escrito este post, como es habitual printar el nombre del autor, se almacena también dicho nombre en el documento que representa al post, para que no sea necesario realizar una segunda consulta a la colección "usuarios". Estas desnormalizaciones dependen nuevamente del uso que se den a los datos. En este caso, el nombre de un usuario no va a cambiar demasiado, así que recorrer todos los posts para cambiar este valor en caso de que el usuario realice esta operación, si bien es una modificación que podría llevar un tiempo considerable para ejecutarse, no es una operación habitual frente a la consulta del nombre del autor, y por tanto compensa. Incluso se podría llegar a tratar el post como algo más permanente, de modo que aunque un usuario cambiara su nombre a posteriori, el nombre utilizado para firmar los posts anteriores no varíe, o sencillamente los posts puedan firmarse con diferentes pseudónimos.

Como se puede ver, el modelado del esquema de datos con MongoDB depende más de la forma en que se consulta o se actualizan los datos que de las limitaciones del propio sistema.

9.9.4. Cómo consultar los datos

Sin entrar demasiado en detalles acerca de todas las posibilidades que MongoDB ofrece para consultar los datos almacenados, si se quisiera nombrar algunas de ellas, para hacer notar que no se está frente a un sistema simple de almacenamiento de pares clave-valor.

En primer lugar, MongoDB permite utilizar funciones Map y Reduce escritas en JavaScript para seleccionar los atributos que nos interesan de los datos, y agregarlos (unificarlos, simplificarlos) en la manera deseada, respectivamente. Esto es algo habitual

en muchos sistemas NoSQL, y en algunos casos es incluso la única forma posible de consultar datos. Claro está que muchas veces necesitamos algo bastante más sencillo que esto.

En MongoDB se pueden utilizar consultas al valor de un atributo específico. Por ejemplo, podemos capturar el post que tiene un determinado título:

```
Db.posts.find({ 'title' : 'Una introducción a MongoDB' })
```

El valor a consultar puede estar anidado en un tipo de datos más completo en el atributo del documento (por ejemplo, como valor de un hash asociado al atributo, o como el valor de uno de los ítems de un array). Se utiliza un punto como separador de los nombres de las claves de los diferentes hashes que hay que recorrer hasta llegar al valor deseado. Por ejemplo, la siguiente consulta devolvería todos los posts escritos por un determinado autor:

```
Db.posts.find({ 'autohor_info._id' : '4da2c0e2e999gb56bf000002' })
```

Y esta otra los posts etiquetados con MongoDB:

```
Db.posts.find({ 'tags : 'MongoDB' })
```

El hash utilizado como conjunto de condiciones que deben cumplir los documentos a devolver puede incluir operadores de muy diversos tipos, no sólo comparadores del valor absoluto buscado. Algunos de ellos son:

- $all: Para indicar que el array almacenado como valor del atributo debe tener los mismos elementos que el proporcionado en la condición.
- $exists: Para comprobar que el atributo existe en el documento.
- $mod: Para comprobar el resto de una división del valor del atributo por un número.
- $ne: Para indicar que el valor no puede ser el proporcionado.
- $in: Para indicar que el valor debe estar entre alguno de los proporcionados.
- $nin: Contrario de $in.
- $or: Para indicar que se debe cumplir al menos una condición de entre un grupo de condiciones.
- $nor: Contrario de $or.
- $size: Para indicar el número de elementos que debe haber en el array almacenado como valor.
- $type: Para comprobar el tipo del valor almacenado (número, cadena…)
- Expresiones regulares: El valor debe concordar con la expresión regular indicada.

Y muchos, muchos más. Los resultados se pueden agrupar, ordenar, contar, paginar, y otras tantas operaciones comunes sin necesidad de recurrir al farragoso Map / Reduce. Y siempre que los atributos consultados tengan definidos un índice, la velocidad de las consultas es espectacular.

9.10. Mongo y PHP

9.10.1. Instalación

El controlador MongoDB para PHP debe funcionar en cualquier sistema: Windows, Mac OS X, Unix, y Linux; pequeñas y grandes máquinas; y plataformas de 32- y 64-bits; PHP 5.1, 5.2, 5.3 y 5.4 [65].

9.10.1.1. Instalación manual

Para los desarrolladores de controladores y gente interesada en las últimas correcciones, puede compilar el controlador desde las últimas versiones en Github. Hay que ir a Github y presionar en el botón "download". Ejecutar:

```
$ tar zxvf mongodb-mongodb-php-driver-<commit_id>.tar.gz
$ cd mongodb-mongodb-php-driver-<commit_id>
$ phpize
$ ./configure
$ make all
$ sudo make install
```

Realizar los siguientes cambios en php.ini:

- Asegúrese de que la variable *extension_dir* este apuntando al directorio que contiene *mongo.so*. El *build* se mostrará donde se encuentre instalado el controlador de PHP, la salida será algo similar a esto:

```
Installing '/usr/lib/php/extensions/no-debug-non-zts-
20060613/mongo.so'
```

- Asegúrese de que este en el mismo directorio en donde PHP se encuentre, ejecute:

```
$ php -i | grep extension_dir
  extension_dir => /usr/lib/php/extensions/no-debug-non-zts-20060613 =>
                   /usr/lib/php/extensions/no-debug-non-zts-20060613
```

- Si no es así, cambia *extension_dir* en php.ini o mueva *mongo.so.*
- Para cargar la extensión en el arranque de PHP, agregar una línea:

```
extension=mongo.so
```

9.10.1.2. Instalación en *NIX

Ejecutar:

```
$ sudo pecl install mongo
```

Si se está usando CentOS o Redhat, se puede instalar desde un RPM.

Agregar la siguiente línea en el fichero *php.ini*:

```
$ sudo pecl install mongo
```

Si *pecl* se quedara sin memoria al instalar, asegúrese de que *memory_limit* en *php.ini* sea de al menos de 32MB.

9.10.1.3. Instalación en Windows

Los binarios precompilados para cada versión están disponibles en Github para una gran variedad de combinaciones de versiones, seguridad en hilos, y bibliotecas VC. Descomprima el fichero y copie *php_mongo.dll* en el directorio de extensiones de PHP ("ext" por omisión).

Agregar la siguiente línea al fichero php.ini:

```
extension=php mongo.dll
```

9.10.1.4. OS X

Si su sistema no puede encontrar *autoconf*, necesita instalar Xcode (disponible en el DVD de instalación o como descarga gratuita desde el sitio Web de Apple).

Si usa XAMPP, debe compilar el driver con el siguiente comando:

```
sudo /Applications/XAMPP/xamppfiles/bin/pecl install mongo
```

Si usa MAMP (o XAMPP y el comando anterior no funciona), los binarios precompilados están disponibles en Github (descargue la última versión con "osx" junto con el nombre que corresponda a la versión de PHP). Extraer *mongo.so* desde el archivo y añádalo al directorio de extensiones de MAMP o XAMPP. Agregar:

```
extension=mongo.so
```

al *fichero php.ini* y reinicie el servidor.

9.10.1.5. Gentoo

Gentoo tiene un paquete para el driver de PHP que se llama *dev-php5/mongo* que puede ser instalado con:

```
$ sudo emerge -va dev-php5/mongo
```

Si se utiliza PECL, quizá obtiene un error de versión incorrecta en *libtool*. Compile desde las fuentes que necesite y ejecute *aclocal* y *autoconf*.

```
$ phpize
$ aclocal
$ autoconf
$ ./configure
$ make
$ sudo make install
```

9.10.1.6. Red Hat

Incluye Fedora y CentOS

En estos sistemas, la configuración por omisión de Apache no permite a las peticiones establecer conexiones de red, haciendo que el driver genere errores de "Permiso denegado" cuando se intenta conectar a la base de datos. Si este fuera el caso, pruebe a ejecutar:

```
$ /usr/sbin/setsebool -P httpd can network connect 1
```

Y finalmente reinicie Apache. (Este comportamiento también se da con SELinux.)

9.10.2. Seguridad

9.10.2.1. Ataques de Inyección de Petición

Al pasar parámetros *$_GET* a una consulta, debemos asegurarnos de que se han convertido en strings. Un usuario puede insertar un array asociativo en una petición GET, provocando consultas $ no deseadas.

Un ejemplo aparentemente inofensivo: supongamos que estamos buscando información de un usuario con la petición *http://www.example.com?username=bob*. La aplicación realiza la consulta*$collection->find(array("username" => $_GET['username']))*.

Alguien podría alterarlo realizando una consulta a *http://www.example.com?username[$ne]=foo*, con lo que PHP lo convertirá automáticamente a un array asociativo, creando la consulta*$collection->find(array("username" => array('$ne' => "foo")))*, que devolverá todos los usuarios con nombre distinto de "foo" (probablemente, todos).

Es sencillo defenderse de un ataque como éste: hay que asegurarse de que los parámetros $_GET son del tipo esperado antes de enviarlos a la base de datos (en este caso, convertirlos a string).

Tenga en cuenta que este tipo de ataque se puede usar con cualquier interacción con una base de datos que localice documentos, incluyendo actualizaciones, búsquedas con modificación, y eliminaciones.

Para más información sobre ataques tipo inyección SQL con MongoDB revise » la documentación principal.

9.10.2.2. Ataques de Inyección de Código

Si se está usando JavaScript, debemos asegurarnos que cualquier variable que cruce los límites PHP-JavaScript se pasa en el campo scope de MongoCode, y no interpolado en el código JavaScript. Esto sucede al llamar a *MongoDB::execute(),* consultas *$where*, MapReduces, agrupaciones, y cualquier otra situación en que se proporcione código JavaScript a la base de datos.

Nota:

MapReduce ignora el campo *scope* de *MongoCode,* pero hay una opción *scope* disponible en el comando que puede utilizarse en su lugar.

Por ejemplo, supongamos que tenemos un código JavaScript para saludar a los usuarios en los registros de la base de datos. Podríamos:

```
<?php
// ¡no haga esto!
$username = $_POST['username'];
$db->execute("print('Hola, $username!');");
?>
```

Pero, ¿qué ocurriría si un usuario malicioso metiera código JavaScript?

```
<?php
// ¡no haga esto!
// $username tiene como valor "'); db.users.drop(); print('"
$db->execute("print('Hola, $username!');");
?>
```

Ahora MongoDB ejecuta el código JavaScript *"print('Hola, '); db.users.drop(); print('!');"*. Este ataque es fácil de evitar: utilice *scope* al pasar variables de PHP a JavaScript:

```
<?php
$scope = array("user" => $username);
$db->execute(new MongoCode("print('Hello, '+user+'!');", $scope));
?>
```

Esto añade la variable *user* al ámbito de JavaScript. Si ahora alguien quisiera añadir código malicioso, MongoDB imprimiría, sin causar daños, *Hello, '); db.dropDatabase(); print('!*.

El uso de *scope* ayuda a prevenir que se ejecute ene en la base de datos entradas maliciosas. Sin embargo, debe asegurarse de que su código no cambie y ejecute los datos de entrada. Por ejemplo, nunca utilice la función *eval* de JavaScript con los datos de entrada de un usuario:

```
<?php
// ¡no haga esto!
// $jsShellInput es "db.users.drop();"
$scope = array("input" => $jsShellInput);
$db->execute(new MongoCode("eval(input);", $scope));
?>
```

Use siempre *scope* y nunca permita que la base de datos ejecute como código los datos de entrada del usuario.