PROYECTO FIN DE CARRERA

Universidad de Murcia

Facultad de Informática

Desarrollo de un Dashboard en la Plataforma Paula

Alumno: Juan Ramírez Monreal

Directores: Jesualdo Tomás Fernández Breis

Convocatoria: Febrero 2008

INDICE

Índice de ilustraciones

Capitulo 1.Introducción

1.1 Contexto

La empresa Aquiline Computer S.A., con una experiencia de 4 años en el campo de I + D, está desarrollando un sistema ERP[1], que además se le están incorporando una serie de funcionalidades para satisfacer todas las necesidades de una empresa. Esta plataforma, se bautizó con el nombre de Paula. El proyecto surge para cubrir una de estas funcionalidades de control que se le quieren incorporan al sistema.

Ilustración 1:
Logotipo
Aquiline S.A

La plataforma Paula pretende conseguir una herramienta tecnológica que acelere de forma importante la competitividad de la empresa que lo utilice. Su cometido se puede concretar en tres procesos fundamentales:

a) simplificar la gestión interna de todos los procesos empresariales de dirección,

b) la administración de los recursos de la propia empresa y de sus colaboradores,

c) el suministro de información básica necesaria para la toma de decisiones desde la dirección o gerencia, tanto desde el punto de vista financiero como de gestión.

El presente proyecto aporta herramientas que agilizan el flujo de información dentro de la empresa, que impulsan el enriquecimiento del conocimiento y el saber hacer, mediante estructuras de colaboración y comunicación. Además se promueve el acceso a los datos del negocio desde cualquier ubicación dentro de la propia empresa, haciendo especial hincapié en la seguridad, ya que solo desde el conocimiento de los indicadores y de las claves del cuadro de mando, se puede acceder a si interpretación.

En su confección**,** se ha hecho un uso extensivo de proyectos software de código abierto, presentando un entorno de integración adecuado que los unifica en una sola plataforma.

De esta manera, el cuadro de mandos será una parte muy importante dentro de la plataforma, ya que se encargará del control de los puntos críticos para cada tipo de usuario que utilice la plataforma. El usuario podrá tomar decisiones en tiempo real, según el estado de los parámetros

1 os sistemas de **planificación de recursos empresariales** (ERP) son sistemas de información gerenciales que integran y manejan muchos de los negocios asociados con las operaciones de producción y de los aspectos de distribución de una compañía comprometida en la producción de bienes o servicios.

medidos.

Así, se pretende que cada tipo de usuario tenga asignado una serie de medidores que automáticamente podrán enviar un aviso a un usuario en particular al producirse un evento en su sector de responsabilidad.

Para ello se han desarrollado dos mejoras:

La primera para gestionar los usuarios y los indicadores, dentro de la gestión de la plataforma.

La segunda dentro del "escritorio" de esta, que mostrará el valor de los indicadores asignados.

La solución ha sido diseñada para que se puedan medir valores con varios tipos de indicadores que podrán ser elegidos en el momento de la configuración de esta plataforma. en una empresa en particular.

Las aplicaciones han sido implementadas en Java por ser multiplataforma.

1.2 Motivación

El concepto de cuadro de mando deriva del concepto denominado Tableau de bord en Francia, que traducido de manera literal, vendría a significar algo así como tablero de mandos, o cuadro de instrumentos, como en el salpicadero de un coche.

Ilustración 2: Cuadro de mando de un automóvil

La gestión de las empresas requiere un sistema de indicadores que nos faciliten la toma de decisiones y el control e su aplicación y eficacia. Se requiere por tanto, de un **sistema completo de análisis.**

El sistema de indicadores que hayamos elegido para determinada empresa, debe organizarse en un cuadro de mando, que como hemos dicho, recogerá los principales indicadores y los presentará de un modo claro y útil, de manera que nos informe de la evolución de los parámetros fundamentales del negocio en un solo plano visual, lo que facilita su comprensión y la toma de decisiones ulteriores.

Existen infinidad de posibles indicadores que podemos utilizar. Algunos ratios o indicadores son de uso muy general y extendido. Entre los más habituales tenemos por ejemplo, los indicadores de venta, margen de beneficios, rotación del inventario y rentabilidad. Otros indicadores mas concretos, deberán ser elaborados expresamente para analizar una empresa concreta según sus necesidades,y así surgirán diferentes indicadores en los cuadros de mando de las diferentes empresas.

Los cuadros de mando siempre han de presentar sólo aquella información que sea imprescindible, de una forma sencilla y por supuesto, concreta y resumida para que sean efectivos.

1.2 Cuadros de mando

Los cuadros de mando son herramientas de gestión del rendimiento, que se presentan ante los usuarios como una visualización de los indicadores empresariales. Permiten monitorizar, controlar y gestionar los procesos de una organización a través de códigos que establecen alertas con las que disponer de una visión completa del rendimiento de la compañía.

Los cuadros de mando de gestión, con sus capacidades de alerta, proporcionan una gran ventaja y muchos beneficios. Los indicadores de cumplimiento, evaluación, eficiencia y eficacia contenidos en ellos ofrecen una visión completa de la organización y su rendimiento, permitiendo comprobar, por ejemplo, si la actividad diaria está alineada con la estrategia corporativa o interpretar lo que está ocurriendo y saber si debemos tomar medidas de mejora.

***Clasificación de los Cuadros de Mando**:*

Según su función se pueden clasificar en:

- ***1. Cuadro de mando operacional**:* cuadro de mando que nos ayuda en la ejecución de procesos.
- ***2. Cuadro de mando táctico*:** nos muestra información que ayuda a controlar procesos
- ***3. Cuadro de mando estratégico*:** consideramos estratégico al cuadro de mando que intervenga en la gestión del proceso para la consecución final de objetivos.

Se establecen cuatro clasificaciones en función de su contenido:

- ***1. Business Activity Monitoring (BAM)*:** muestra en tiempo real información de carácter operacional y táctico, utiliza los KPI [2] (Key Performance Indicator), están orientados a la monitorización, dan soporte a la toma de decisiones a muy corto plazo y no necesitan de trazabilidad decisional.
- ***2. Dashboarding**:* se conocen así los cuadros de mando que muestran información sin compararla con objetivos propuestos. Es decir, con ellos únicamente se pueden visualizar los KPI (Key Performance Indicator).
- ***3. Scorecarding*:** muestran información estratégica y están orientados a mostrar objetivos, por lo tanto, además de ofrecer los indicadores KPI, permiten almacenar en el sistema los KGI (Key Goal Indicador)
- ***4. Balanced Scorecard:*** constituyen una metodología de gestión estratégica independiente.

2 miden el nivel del desempeño de un proceso, enfocándose en el como e indicando que tan buenos son los procesos, de forma que se pueda alcanzar el objetivo fijado.

Ilustración 3: Ejemplo Balanced Scorecard

1.3 Cuadro de mando integral (Balanced ScoreCard).

Desde la perspectiva financiera, en general, los indicadores están basados en la contabilidad de la Compañía, y muestran el pasado de la misma. El motivo se debe a que la contabilidad no es inmediata (Al emitir un proveedor una factura, la misma no se contabiliza automáticamente), sino que deben efectuarse cierres que aseguren la completitud y consistencia de la información. Debido a estas demoras, algunos autores sostienen que dirigir una Compañía prestando atención solamente a indicadores financieros es como conducir a 100 km/h mirando por el espejo retrovisor.

Para lograr el desempeño financiero que una empresa desea, es fundamental que posea clientes leales y satisfechos, con ese objetivo **y** en esta perspectiva se miden las relaciones con los clientes y las expectativas que los mismos tienen sobre los negocios. Además, en esta perspectiva se toman en cuenta los principales elementos que generan valor para los clientes, para poder así centrarse en los procesos que para ellos son más importantes y que más los satisfacen.

El conocimiento de los clientes y de los procesos que más valor generan es muy importante para lograr que el panorama financiero sea próspero. Sin el estudio de las peculiaridades del mercado al que está enfocada la empresa no podrá existir un desarrollo sostenible en la perspectiva financiera, ya que en gran medida el éxito financiero proviene del aumento de las ventas, situación que es el efecto de clientes que repiten sus compras porque prefieren los productos que la empresa desarrolla teniendo en cuenta sus preferencias

El cuadro de mando integral, analiza la adecuación de los procesos internos de la empresa de cara a la obtención de la satisfacción del cliente y conseguir altos niveles de rendimiento financiero. Para alcanzar este objetivo se propone un análisis de los procesos internos desde una perspectiva de negocio y una predeterminación de los procesos clave a través de la cadena de valor.

Se distinguen cuatro tipos de procesos:

- 1.**Procesos de Operaciones**. Desarrollados a través de los análisis de calidad y reingeniería. Los indicadores son los relativos a costes, calidad, tiempos o flexibilidad de los procesos.
- 2.**Procesos de Gestión de Clientes**. Indicadores: Selección de clientes, captación de clientes, retención y crecimiento de clientes.
- 3.**Procesos de Innovación** (difícil de medir). Ejemplo de indicadores: % de productos nuevos, % productos patentados, introducción de nuevos productos en relación a la competencia...
- 4.**Procesos relacionados con el Medio Ambiente y la Comunidad**. Indicadores típicos de Gestión Ambiental, Seguridad e Higiene y Responsabilidad Social Corporativa.

Básicamente, y de manera resumida, podemos destacar tres características fundamentales de este Cuadro de mandos:

1. La naturaleza de las informaciones recogidas en él, dando cierto privilegio a las secciones operativas, (ventas, etc.) para poder informar a las secciones de carácter financiero, siendo éstas últimas el producto resultante de las demás.

2. La rapidez de ascenso de la información entre los distintos niveles de responsabilidad.

3. La selección de los indicadores necesarios para la toma de decisiones, sobre todo en el menor

número posible.

En definitiva, lo importante es establecer un sistema de señales en forma de Cuadro de mando que nos indique la variación de las magnitudes verdaderamente importantes que debemos vigilar para someter a control la gestión.

1.4 Dashboard en Plataforma PAULA.

El Dashboard de la plataforma Paula está diseñado para proporcionar el máximo impacto visual en un formato optimizado para una rápida comprensión. Combinan tablas, gráficos, indicadores y otros controles gráficos, totalmente configurables, y con un enriquecimiento semántico agregado.

Tenemos que subrayar el hecho de que, aparte de reunir información de similares características que la empleada en las distintas disciplinas de naturaleza contable, es decir, financiera, puede contener información de carácter no financiero.

La personalización del contenido del Dashboards es una característica importante que precisa de una arquitectura de plataforma segura que garantice el acceso a los datos únicamente a usuarios autorizados.

La arquitectura es suficientemente segura para dar soporte a cualquier nivel de acceso de usuarios. El contenido des Dashboard se puede adaptar automáticamente al perfil basado en la función de cada persona, a las restricciones de privacidad, a las políticas de seguridad o a los intereses de cada área.

Hay que destacar la relación mutua que existe entre el Cuadro de mando y el perfil de la persona a quien va destinado. Precisamente, las necesidades de cada usuario, han de marcar la pauta que caracterice y haga idónea a esta herramienta en cada caso y situación, sobre todo con respecto al nivel de mayor responsabilidad de la jerarquía actual de la empresa, debido a que se precisa un esfuerzo mucho mayor de generalidad y síntesis. Así, un usuario normal se le asignaran una serie de indicadores relacionados con su desempeño en la empresa, mientras que los asignados al gerente de la empresa serán mas de un ámbito de control e información.

Un rasgo más del Cuadro de mando es la solución de problemas mediante lanzamientos de workflows[3]. Cuando incorporamos indicadores que necesitamos que lancen una acción al sobrepasar unos límites marcados en el indicador, le asignaremos un id de workflow. Al sobrepasar unos rangos, el workflow se lanzará, y el usuario podrá observar al momento esta incidencia, pudiendo tomar medidas de control.

En relación a los tipos de indicadores tendremos que tener en cuenta las principales variables en la Dirección General, Direcciones Funcionales y Subdirecciones Funcionales. Ya afirmábamos que no existe una única fórmula para todas las empresas, sino que para cada tipo de organización habrá que tomar unas variables determinadas con las que llevar a cabo la medición de la gestión, creando en cada caso, indicadores nuevos. Es importante tener en cuenta que el contenido de cualquier Cuadro de mando, no se reduce tan sólo a cifras o números, ha de ser un contenido muy concreto para cada departamento o para cada responsable. De igual manera, se ha de tener presente que la información que se maneja en un Cuadro de mando determinado puede ser válida para otro, pudiendo así reutilizar indicadores existentes en la plataforma. Los indicadores son los elementos objetivos que describen situaciones específicas, y que tratan de medir de alguna manera las variables propuestas en cada caso. Por eso tenemos diferentes tipos, tanto por como muestran los datos medidos, como en la manera de mostrarlos.

3 es el estudio de los aspectos operacionales de una actividad de trabajo: cómo se estructuran las tareas, cómo se realizan, cuál es su orden correlativo, cómo se sincronizan, cómo fluye la información que soporta las tareas y cómo se le hace seguimiento

La integración del Dashboard en Paula permite que cualquier número de usuarios pueda acceder con seguridad a los informes necesarios en cualquier lugar y momento, y a través de cualquier interface. Tiene escalabilidad de usuarios ilimitada y se pueden publicar y distribuir a decenas y centenares de usuarios de la empresa, internos y externos, de forma eficaz y segura, mediante un navegador web.

1. 5 Referencias históricas

Desde hace muchos años se está utilizando un nuevo sistema de gestión estratégico denominado "Balanced Scorecard", originalmente desarrollado por el profesor Robert Kaplan de la Universidad de Harvard y David Norton de Nolan & Norton. Al inició se desarrollo como un sistema de medición mejorado, pero con el tiempo ha venido evolucionado hasta convertirse en el núcleo o piedra angular del sistema de gestión estratégico de cualquier compañía.

El BSC[4] se ha convertido en el gran aliado de los presidentes y directivos de las más importantes compañías del mundo. Por una sencilla razón, el BSC les garantiza el cumplimiento de la visión de sus compañías, y esta es la actividad más importante que deberían ejecutar para lograr sus objetivos.

En la mayoría de las empresas casi nadie conoce la visión. Un estudio reciente de la firma Business Intelligence realizado en Estados Unidos y Europa demostró que en las compañías no se conoce la visión: el 70% de la Alta gerencia de una compañía conocía la visión, pero solo el 40% de la Gerencia Media y nada más que el 10% de los empleados la conoce.

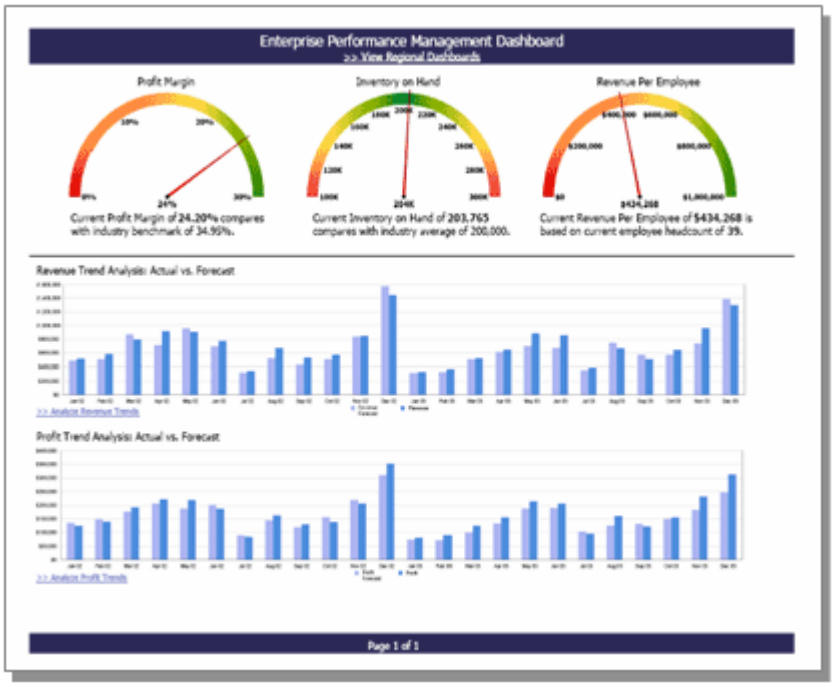


Ilustración 4: Ejemplo Dashboard

El Balanced Scorecard surge como una herramienta excelente para comunicar a toda la organización la visión de la compañía.

Pero conocer la visión no lo es todo. Se ha visto también que la mayoría de empresas al tratar de llevar a la acción la visión no consiguen hacerlo. Cuantos fracasos al implementar herramientas gerenciales como Planeación Estratégica, Calidad Total, Reingeniería, y muchas más. La visión se convierte en algo precioso, que en ocasiones todo el mundo repite de memoria, pero

4Balanced Scorecard

que es etérea, no logramos hacerla realidad en nuestras empresas. No existe un enlace entre las metas individuales y la estrategia y mucho menos entre la estrategia y el presupuesto.

El BSC logra que todos los empleados se comprometan a llevar la organización por medio de acciones concretas las cuales es posible ir monitoreando.

Capitulo 2. Análisis de objetivos y metodología de trabajo

2. 1 Objetivos

Los objetivos marcados para la realización del proyecto son los siguientes:

- Desarrollo del Dashboard, con estado de los recursos, desarrollado con tecnología j2ee[5] orientado a la web. Ha de ser un medio informativo destacable y a la misma vez una herramienta de diagnostico.
- Desarrollo de una aplicación para gestionar los recursos del cuadro de mando y los usuarios.
- Con el objetivo de cumplir los principios de la web 2.0, tendremos que desarrollar el Dashboard sobre una plataforma web. Ofrecer una arquitectura de participación, dando la posibilidad al usuario de agregar nueva información. Introducir una base de datos especializada. Ofrecer el desarrollo en forma de servicio, y no de producto. Apoyarse en modelos de programación ligeros. No limitar el desarrollo al dispositivo PC.
- Documentación e investigación en la red, sobre los actuales Cuadros de mandos y la manera de implementarlos en el cuadro de mando
- Separación de la información de forma que cada persona reciba sólo que le interesa conocer del sistema de información.
- Uso de la tecnología AJAX para actualizar los indicadores, sin necesidad de recargar toda la pagina
- Gestión de indicadores basada en tags. Posibilidad de crear etiquetas (que completarían la semántica del indicador) y desarrollar un buscador de indicadores en base a etiquetas.
- Generar un evento cuando se rebase un límite superior o inferior, lanzando un workflow, donde podemos controlar las tareas relacionadas con ese indicador.

5 plataforma de programación parte de laPlataforma Java para desarrollar y ejecutar software de aplicaciones

2.2 Metodología

La metodología seguida en la realización del proyecto, ha sido una adaptación de la metodología propia que utiliza la empresa Aquiline Computer S.A. en sus proyectos de desarrollo, para sus clientes. El papel del cliente será realizado por el tutor, y el proyecto realizado es el Dashboard. También hay que tener en cuenta que el trabajo ha sido ejecutado por una sola persona, y no por un equipo de trabajo, omitiendo todas las fases de equipo. El proyecto seguirá los siguientes pasos, pudiendo iterar en algunos de sus puntos varias veces:

Estudio Previo:

- Análisis de la información disponible sobre los cuadros de mando y todo los relacionado con ellos
- Desarrollo de una estrategia para ir cumpliendo plazos dentro de lo posible, con un pequeño margen, tenido en cuenta también, a la hora de ir marcando los hitos.
- Estudio y comprensión de todas las herramientas necesarias para la realización del proyecto
- Establecimiento de puntos críticos
- Investigación de dominios disponibles

Diseño de las interfaces y prototipos del website

- Desarrollo del contenido visual de las dos aplicaciones
- Desarrollo del prototipo de un indicador
- Desarrollo de un prototipo de páginas con varios indicadores
- Aprobación del tutor en la empresa del trabajo realizado

Creatividad aplicada al sitio y elaboración del anteproyecto

- Creación de gráficos con librería JFreeChart
- Desarrollo de la navegabilidad del sitio
- Creación del modelo de datos en la BBDD[6] de la plataforma
- Definición de los formularios necesarios para interactividad
- Pruebas Unitarias
- Aprobación del tutor para pasar al proyecto final

Desarrollo del proyecto final

- Introducción de las mejoras solicitadas por el tutor
- Creación de documento que recogerá toda la información sobre las herramientas y trabajo realizado
- Verificación de completitud y exactitud del website
- Verificación en diferentes tipos de navegadores para asegurar compatibilidad
- Aprobación del tutor del trabajo final realizado

6 Abreviatura de Base de datos Postgres

Capitulo 3.Estudio de las tecnologías y herramientas usadas

En esta sección haremos una breve descripción de las herramientas utilizadas en la resolución del problema:

3. 1 Soluciones conceptuales y de gestión del sistema

Web 2.0

El concepto original de la web (en este contexto, llamada Web 1.0) era páginas estáticas HTML que no eran actualizadas frecuentemente. El éxito de las punto-com dependía de webs más dinámicas (a veces llamadas Web 1.5) donde los CMS[7] servían páginas HTML dinámicas creadas al vuelo desde una actualizada base de datos. En ambos sentidos, el conseguir hits (visitas) y la estética visual eran considerados como unos factores muy importantes.

Los propulsores de la aproximación a la Web 2.0 creen que el uso de la web está orientado a la interacción y redes sociales, que pueden servir contenido que explota los efectos de las redes creando o no webs interactivas y visuales. Es decir, los sitios Web 2.0 actúan más como puntos de encuentro, o webs dependientes de usuarios, que como webs tradicionales.

En 2005, Tim O'Reilly resumió los principios clave que creen que caracterizan a las aplicaciones web 2.0: la web como plataforma; datos como el "Intel Inside"; efectos de red conducidos por una "arquitectura de participación"; innovación y desarrolladores independientes; pequeños modelos de negocio capaces de sindicar servicios y contenidos; el perpetuo beta; software por encima de un solo aparato.

En general, cuando mencionamos el término Web 2.0 nos referimos a una serie de aplicaciones y páginas de Internet que utilizan la inteligencia colectiva para proporcionar servicios interactivos en red dando al usuario el control de sus datos.

Así, podemos entender como 2.0 "todas aquellas utilidades y servicios de Internet que se sustentan en una base de datos, la cual puede ser modificada por los usuarios del servicio, ya sea en su contenido (añadiendo, cambiando o borrando información o asociando metadatos a la información existente), bien en la forma de presentarlos, o en contenido y forma simultáneamente."

7 **Sistema de gestión de contenidos** (***Content Management System***, en inglés, abreviado **CMS**) permite la creación y administración de contenidos principalmente en páginas web.

Ilustración 1: esquema Web 2.0

La infraestructura de la Web 2.0 es compleja y va evolucionando, pero incluye el software de servidor, sindicación de contenidos, protocolos de mensajes, navegadores basados en estándares, y varias aplicaciones para clientes.

Una web se puede decir que está construida usando tecnología de la Web 2.0 si se caracteriza por las siguientes técnicas:

- CSS, marcado XHTML válido semánticamente y Microformatos
- Técnicas de aplicaciones ricas no intrusivas (como AJAX)
- Java Web Start
- XUL [8]
- Sindicación/Agregación de datos en RSS[9] /ATOM[10]
- URLs sencillas y con significado
- Soporte para postear en un blog
- JCC[11] y APIs REST o XML
- Algunos aspectos de redes sociales
- Mashup (aplicación web híbrida)
- En general, el sitio no debe actuar como un "jardín cerrado": la información debe poderse introducir y extraer fácilmente.
- Los usuarios deberían controlar su propia información
- Basada exclusivamente en la Web: los sitios Web 2.0 con más éxito pueden ser utilizados enteramente desde un navegador

8 acrónimo de ***X**ML-based **U**ser-interface **L**anguage*, lenguaje basado en XML para la interfaz de usuario
9 **RSS** es un sencillo formato de datos que es utilizado para redifundir contenidos a suscriptores de un sitio web
10 **Atom** es un formato XML similar a RSS
11 **JavaScript Client Communication** hace referencia a las técnicas de programación que, utilizando objetos JSI en el navegador facilitan la integración en la misma página Web de aplicaciones a priori independientes

En cuanto a la redifusión, la primera y más importante evolución de la Web 2.0 se refiere a la redifusión del contenido de una Web, usando protocolos estandarizados que permitan a los usuarios finales usar el contenido de la web en otro contexto, ya sea en otra web, en un conector de navegador o en una aplicación de escritorio. Entre los protocolos que permiten sindicar se encuentran RSS, RDF (conocido también como RSS 1.1), y Atom, todos ellos variedades de XML. Los protocolos específicos como FOAF y XFN (ambos para redes sociales) amplían la funcionalidad de los sitios y permiten a los usuarios interactuar sin contar con sitios Web centralizados.

Los protocolos de mensajes bidireccionales, en los servicios web, son uno de los elementos clave de la infraestructura de la Web 2.0. Los dos tipos más importantes son los métodos RESTful y SOAP. REST indican un tipo de llamada a un servicio web donde el cliente transfiere el estado de todas las transacciones. SOAP y otros métodos similares dependen del servidor para retener la información de estado. En ambos casos, el servicio es llamado desde un API. A veces este API está personalizado en función de las necesidades específicas del sitio web, pero los APIs de los servicios web estándares (como por ejemplo escribir en un blog) están también muy extendidos. Generalmente el lenguaje común de estos servicios web es el XML, si bien puede haber excepciones.

Recientemente, una forma híbrida conocida como Ajax ha evolucionado para mejorar la experiencia del usuario en las aplicaciones web basadas en el navegador. Esto puede ser usado en webs propietarias (como en Google Maps) o en formas abiertas utilizando un API de servicios web, una semilla de sindicación.

La funcionalidad de Web 2.0 se basa en la arquitectura existente de servidor web pero con un énfasis mayor en el software dorsal. La sindicación sólo se diferencia nominalmente de los métodos de publicación de la gestión dinámica de contenido, pero los servicios Web requieren normalmente un soporte de bases de datos y flujo de trabajo mucho más robusto y llegan a parecerse mucho a la funcionalidad de intranet tradicional de un servidor de aplicaciones. El enfoque empleado hasta ahora por los fabricantes suele ser bien un enfoque de servidor universal, el cual agrupa la mayor parte de la funcionalidad necesaria en una única plataforma de servidor, o bien un enfoque plugin de servidor Web con herramientas de publicación tradicionales mejoradas con interfaces API y otras herramientas. Independientemente del enfoque elegido, no se espera que el camino evolutivo hacia la Web 2.0 se vea alterado de forma importante por estas opciones.

En ocasiones se ha utilizado el término Web 2.0 como análogo a Web semántica. Sin embargo ambos conceptos, aun siendo afines, no son iguales, sino más bien complementarios. La combinación de sistemas de redes sociales, como FOAF y XFN, con el desarrollo de etiquetas (o tags), que en su uso social derivan en folcsonomías, así como el plasmado de todas estas tendencias a través de blogs y wikis, confieren a la Web 2.0 un aire semántico. Sin embargo, en el sentido más estricto de Web semántica se requiere el uso de ontologías y no de folcsonomías. De momento, el uso de ontologías como mecanismo de estructurar la información en los programas de blogs es anecdótico y sólo se aprecia de manera incipiente en algunos wikis.

Bien podría hablarse de la Web 3.0 para la Web semántica. Pero una diferencia fundamental entre ambas versiones de web (2.0 y semántica) es el tipo de participante. La 2.0 tiene como principal protagonista al usuario humano que escribe artículos en su blog o colabora en un wiki. El requisito es que además de publicar en HTML emita parte de sus aportaciones en XML/RDF (RSS, ATOM, etc.). La web semántica, sin embargo, está orientada hacia el protagonismo de procesadores

mecánicos que entiendan de lógica descriptiva en OWL[12] y concebida para que las máquinas hagan el trabajo de las personas a la hora de procesar la avalancha de información publicada en la Web.

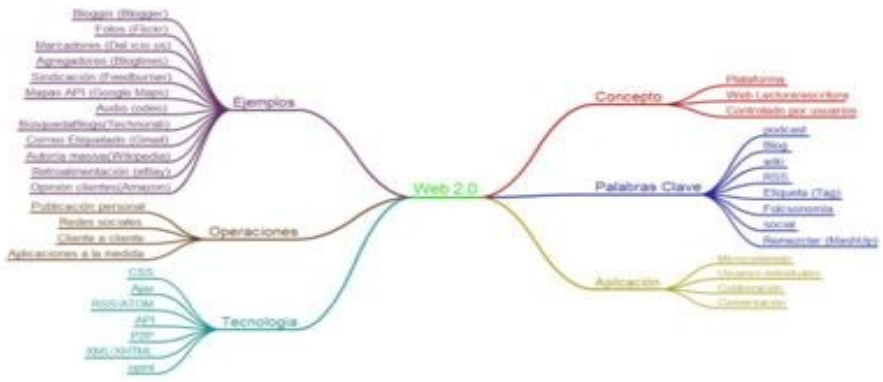


Ilustración 2: Mapa conceptual Web 2.0

12 **Ontology Web Language**, un lenguaje de marcado para publicar y compartir datos usando ontologías en la WWW.

Federación de Identidad

La identidad federada es una de las soluciones para abordar la gestión de identidad en los sistemas de información. El valor añadido adicional respecto a otras soluciones es la gestión de identidad interdependiente entre compañías, lo que se denomina Federated Identity Management.

Como cualquier solución de IdM (Identity Management o Gestión de Identidad en castellano), su objetivo es obtener una gestión de usuarios eficiente, la sincronización de los datos identificativos, gestión de acceso, servicios de agrupación, servicios de directorio, auditoría e informes,...

Mediante soluciones de Identidad Federada los individuos pueden emplear la misma identificación personal (típicamente usuario y contraseña) para identificarse en redes de diferentes departamentos o incluso empresas. De este modo las empresas comparten información sin compartir tecnologías de directorio, seguridad y autenticación, como requieren otras soluciones (metadirectorio, Single Sign On[13], etc.). Para su funcionamiento es necesaria la utilización de estándares que definan mecanismos que permiten a las empresas compartir información entre dominios. El modelo es aplicable a un grupo de empresas o a una gran empresa con numerosas delegaciones y se basa en el "círculo de confianza" de estas, un concepto que identifica que un determinado usuario es conocido en una comunidad determinada y tiene acceso a servicios específicos.

La plataforma Paula utiliza esta solución para el autenticada de usuarios. Un usuario tiene una login y contraseña, y con ellas puede acceder a los diferentes módulos de la plataforma, conectarse al correo o a la mensajería instantánea.

13 procedimiento de autenticación que habilita al usuario para acceder a varios sistemas con una sola instancia de identificación.

3. 2 Entorno trabajo: Linux, Eclipse, apache tomcat y Maven

Linux es la denominación de un sistema operativo tipo Unix (también conocido como GNU/Linux) y el nombre de un núcleo. Es uno de los ejemplos más prominentes del software libre y del desarrollo del código abierto, cuyo código fuente está disponible públicamente, para que cualquier persona pueda libremente usarlo, estudiarlo, redistribuirlo, comercializarlo y, con los conocimientos informáticos adecuados, modificarlo.

Las variantes de los sistemas Linux de denominan "distribuciones" y su objetivo es ofrecer una edición que cumpla con las necesidades de determinado grupo de usuarios, de estar forma existen distribuciones para hogares, empresas y servidores. Algunas son gratuitas y otras de pago, algunas insertan software no libre y otras contienen solo software libre.

Los primeros sistemas Linux se originaron en 1992, al combinar utilidades de sistema y bibliotecas del proyecto GNU con el núcleo Linux, creado por Linus Torvalds, con la colaboración de cientos de codesarrolladores y que se distribuye bajo la Licencia Pública General de GNU. Desde finales del 2000 se ha incrementado el apoyo y respaldo de parte de fabricantes de hardware como IBM, Sun Microsystems, Hewlett-Packard, y Novell. Algunos de ellos lo incluyen pre-instalado en algunos modelos de computadoras portátiles y de escritorio para el usuario final. El respaldo de las compañías de software también está presente, ya que -entre otras aplicaciones- Nero, Java, Google Earth, Google Desktop, Adobe Reader, Adobe Flash, RealPlayer, Silverlight y Yahoo! Messenger están disponibles para Linux.

Linux puede instalarse en computadores de escritorio (PCs x86 y x86-64 así como Macintosh y PowerPC), computadores de bolsillo, teléfonos celulares, portátiles, dispositivos empotrados, videoconsolas (Xbox, PlayStation 3, PlayStation Portable, Dreamcast, GP2X...) y otros, sin embargo su mayor desarrollo se ha llevado a cabo en el los servidores y supercomputadores.

Todo el desarrollo del proyecto ha sido realizado en una maquina con sistema Linux Ubuntu.

Ilustración 3: Tux

Eclipse es una plataforma de software de código abierto independiente de una plataforma para desarrollar lo que el proyecto llama "Aplicaciones de Cliente Enriquecido", opuesto a las aplicaciones "Cliente-liviano" basadas en navegadores. Esta plataforma, típicamente ha sido usada para desarrollar entornos de desarrollo integrados (del inglés IDE), como el IDE de Java llamado Java Development Toolkit (JDT) y el compilador (ECJ) que se entrega como parte de Eclipse (y que son usados también para desarrollar el mismo Eclipse). Sin embargo, también se puede usar para

otros tipos de aplicaciones cliente, como BitTorrent Azureus.

Eclipse es también una comunidad de usuarios, extendiendo constantemente las áreas de aplicación cubiertas. Un ejemplo es el recientemente creado Eclipse Modeling Project, cubriendo casi todas las áreas de Model Driven Engineering.

Eclipse fue desarrollado originalmente por IBM como el sucesor de su familia de herramientas para VisualAge. Eclipse es ahora desarrollado por la Fundación Eclipse, una organización independiente sin ánimo de lucro que fomenta una comunidad de código abierto y un conjunto de productos complementarios, capacidades y servicios.

Toda la parte de programación ha sido realizada dentro de esta plataforma.

Ilustración 4: Logo plataforma eclipse

Tomcat es un servidor web con soporte de servlets y JSPs. Incluye el compilador Jasper, que compila JSPs convirtiéndolas en servlets. El motor de servlets de Tomcat a menudo se presenta en combinación con el servidor web Apache.

Tomcat puede funcionar como servidor web por sí mismo. En sus inicios existió la percepción de que el uso de Tomcat de forma autónoma era sólo recomendable para entornos de desarrollo y entornos con requisitos mínimos de velocidad y gestión de transacciones. Hoy en día ya no existe esa percepción y Tomcat es usado como servidor web autónomo en entornos con alto nivel de tráfico y alta disponibilidad.

Dado que Tomcat fue escrito en Java, funciona en cualquier sistema operativo que disponga de la máquina virtual Java.

El servidor, donde se despliegue la plataforma, deberá tener instalado este servidor web. Para realizar las pruebas, también se ha instalado en la maquina de desarrollo del proyecto esta aplicación.

Ilustración 5: Logo Apache tomcat

Maven es una herramienta software para la gestión y comprensión de proyectos Java. Estaba integrado dentro del proyecto Jakarta pero ahora ya es un proyecto de nivel superior de la Apache Software Foundation.

En la versión 1, basándose en un fichero de configuración en XML (project.xml) y una serie de plugins, esta herramienta puede compilar el proyecto Java, ejecutar los tests unitarios, generar

paquetes (jars, wars, ears o distribuciones en zip) y generar una serie de reports. La versión 2 usa también un fichero de configuración en XML llamado pom.xml. Su funcionalidad es parecida a Apache Ant de manera que permite compilar, ejecutar test o realizar distribuciones pero con la diferencia que trata de forma automática las dependencias del proyecto. Una de las más importantes características es su actualización en línea mediante servidores repositorios. Maven es capaz de descargar nuevas actualizaciones de las bibliotecas de las que depende el proyecto y de igual manera subir una nueva distribución a un repositorio de versiones, dejándola al acceso de todos los usuarios. Las compilaciones del código desarrollado ha sido realizado por medio de esta herramienta.

3. 3 Herramientas principales de desarrollo y ejecución del software

J2EE

Java Platform, Enterprise Edition o Java EE (anteriormente conocido como Java 2 Platform, Enterprise Edition o J2EE hasta la versión 1.4), es una plataforma de programación—parte de la Plataforma Java—para desarrollar y ejecutar software de aplicaciones en Lenguaje de programación Java con arquitectura de n niveles distribuida, basándose ampliamente en componentes de software modulares ejecutándose sobre un servidor de aplicaciones. La plataforma Java EE está definida por una especificación. Similar a otras especificaciones del Java Community Process, Java EE es también considerada informalmente como un estándar debido a que los suministradores deben cumplir ciertos requisitos de conformidad para declarar que sus productos son conformes a Java EE; no obstante sin un estándar de ISO[14] o ECMA[15].

Java EE incluye varias especificaciones de API, tales como JDBC, RMI, e-mail, JMS, Servicios Web, XML, etc y define cómo coordinarlos. Java EE también configura algunas especificaciones únicas para Java EE para componentes. Estas incluyen Enterprise JavaBeans, servlets, portlets (siguiendo la especificación de Portlets Java), JavaServer Pages y varias tecnologías de servicios web. Esto permite al desarrollador crear una Aplicación de Empresa portable entre plataformas y escalable, a la vez que integrable con tecnologías anteriores, que es uno de los objetivos de este proyecto. Otros beneficios añadidos son, por ejemplo, que el servidor de aplicaciones puede manejar transacciones, la seguridad, escalabilidad, concurrencia y gestión de los componentes desplegados, significando que los desarrolladores pueden concentrarse más en la lógica de negocio de los componentes en lugar de en tareas de mantenimiento de bajo nivel.

Hibernate

Hibernate es una herramienta de Mapeo objeto-relacional para la plataforma Java (y disponible también para .Net con el nombre de NHibernate) que facilita el mapeo de atributos entre una base de datos relacional tradicional y el modelo de objetos de una aplicación, mediante archivos declarativos (XML) que permiten establecer estas relaciones.

Como todas las herramientas de su tipo, Hibernate busca solucionar el problema de la diferencia entre los dos modelos usados hoy en día para organizar y manipular datos: El usado en la memoria de la computadora (orientación a objetos) y el usado en las bases de datos (modelo relacional). Para lograr esto permite al desarrollador detallar cómo es su modelo de datos, qué relaciones existen y qué forma tienen. Con esta información Hibernate le permite a la aplicación manipular los datos de la base operando sobre objetos, con todas las características de la POO[16]. Hibernate convertirá los datos entre los tipos utilizados por Java y los definidos por SQL. Hibernate genera las sentencias SQL y libera al desarrollador del manejo manual de los datos que resultan de

14 **Organización Internacional para la Estandarización**
15 organización internacional basada en membresías de estándares para la comunicación y la información
16 Programación orientada a objetos

la ejecución de dichas sentencias, manteniendo la portabilidad entre todas las bases de datos con un ligero incremento en el tiempo de ejecución.

Hibernate está diseñado para ser flexible en cuanto al esquema de tablas utilizado, para poder adaptarse a su uso sobre una base de datos ya existente. También tiene la funcionalidad de crear la base de datos a partir de la información disponible.

Hibernate ofrece también un lenguaje de consulta de datos llamado HQL (Hibernate Query Language), al mismo tiempo que una API para construir las consultas programáticamente (conocida como "criteria").

Hibernate para Java puede ser utilizado en aplicaciones Java independientes o en aplicaciones Java EE, mediante el componente Hibernate Annotations que implementa el estándar JPA, que es parte de esta plataforma.

Con la ayuda de este componente hemos facilitado mucho las transacciones con la base de datos y hemos podido realizar el trabajo mas rápidamente.

Ilustración 6: Logo Hibernate

BBDD postgres

Hemos utilizado la base de datos *Postgres*, desarrollada por la Universidad de Berkeley. Algunas de sus principales características son:

- Alta concurrencia:Mediante un sistema denominado MVCC (Acceso concurrente multiversión, por sus siglas en inglés) PostgreSQL permite que mientras un proceso escribe en una tabla, otros accedan a la misma tabla sin necesidad de bloqueos. Cada usuario obtiene una visión consistente de lo último a lo que se le hizo *commit*. Esta estrategia es superior al uso de bloqueos por tabla o por filas común en otras bases, eliminando la necesidad del uso de bloqueos explícitos.
- Amplia variedad de tipos nativos : PostgreSQL provee nativamente soporte para:
  - Números de precisión arbitraria.
  - Texto de largo ilimitado.
  - Figuras geométricas (con una variedad de funciones asociadas)
  - Direcciones IPs (IPv4 e IPv6).
  - Bloques de direcciones estilo CIDR[17].
  - Direcciones MAC.

17 Encaminamiento Inter-Dominios sin Clases

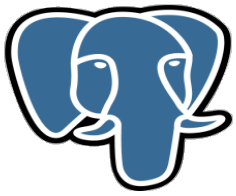


Ilustración 7: Logo PGSQL

ArgoUML

Herramienta de licencia libre, muy útil para el diseño de diagramas UML. Antes de realizar la solución, y para una buena organización, construimos los diferentes diagramas, para apoyarnos en ellos a la hora de realizar un buen desarrollo de la solución.

Esta herramienta tiene las siguientes características:

- Esta desarrollada completamente en JAVA, con lo que teniendo la maquina virtual instalada, podremos lanzarla desde cualquier sistema.
- Puede realizar los siguientes diagramas UML:De Clases, de Estado, de Actividad, de Casos de Uso, de Colaboración, Deployment (combinación de diagramas de Objetos / Componentes / Depliegue)
- Permite la navegación por el modelo
- Permite la exportación de diagramas
- Contiene diferentes métricas y soporta XMI
- Apoyo a OCL y control de errores (checklist) automático

Ilustración 8: Logo ArgoUML

3. 4 Tecnologías de apoyo al desarrollo software

CSS: Separación de Diseño y Contenido

Las hojas de estilo en cascada (Cascading Style Sheets, CSS) son un lenguaje formal usado para definir la presentación de un documento estructurado escrito en HTML o XML (y por extensión en XHTML). El W3C (World Wide Web Consortium) es el encargado de formular la especificación de las hojas de estilo que servirá de estándar para los agentes de usuario o navegadores.

La idea que se encuentra detrás del desarrollo de CSS es separar la estructura de un documento de su presentación.

Por ejemplo, el elemento de HTML <H1> indica que un bloque de texto es un encabezamiento y que es más importante que un bloque etiquetado como <H2>. Versiones más antiguas de HTML permitían atributos extra dentro de la etiqueta abierta para darle formato (como el color o el tamaño de fuente). No obstante, cada etiqueta <H1> debía disponer de la información si se deseaba un diseño consistente para una página, y además, una persona que lea esa página con un navegador pierde totalmente el control sobre la visualización del texto.

Cuando se utiliza CSS, la etiqueta <H1> no debería proporcionar información sobre cómo va a ser visualizado, solamente marca la estructura del documento. La información de estilo separada en una hoja de estilo, específica cómo se ha de mostrar <H1> : color, fuente, alineación del texto, tamaño, y otras características no visuales como definir el volumen de un sintetizador de voz, por ejemplo.

La información de estilo puede ser adjuntada tanto como un documento separado o en el mismo documento HTML. En este último podrían definirse estilos generales en la cabecera del documento o en cada etiqueta particular mediante el atributo "style".

Las ventajas de utilizar CSS (u otro lenguaje de estilo) son:

- Control centralizado de la presentación de un sitio web completo con lo que se agiliza de forma considerable la actualización del mismo.
- Los Navegadores permiten a los usuarios especificar su propia hoja de estilo local que será aplicada a un sitio web, con lo que aumenta considerablemente la accesibilidad. Por ejemplo, personas con deficiencias visuales pueden configurar su propia hoja de estilo para aumentar el tamaño del texto o remarcar más los enlaces.
- Una página puede disponer de diferentes hojas de estilo según el dispositivo que la muestre o incluso a elección del usuario. Por ejemplo, para ser impresa, mostrada en un dispositivo móvil, o ser "leída" por un sintetizador de voz.
- El documento HTML en sí mismo es más claro de entender y se consigue reducir considerablemente su tamaño.

Todas las paginas web desarrolladas en este proyecto tienen un CSS aplicado para la presentación atractiva de los datos al usuario.

JAVASCRIPT

JavaScript es un lenguaje de programación interpretado, es decir, que no requiere compilación, utilizado principalmente en páginas web, con una sintaxis semejante a la del lenguaje Java y el lenguaje C.

Al contrario que Java, JavaScript no es un lenguaje orientado a objetos propiamente dicho, ya que no dispone de Herencia, es más bien un lenguaje basado en prototipos, ya que las nuevas clases se generan clonando las clases base (prototipos) y extendiendo su funcionalidad.

Todos los navegadores interpretan el código JavaScript integrado dentro de las páginas web. Para interactuar con una página web se provee al lenguaje JavaScript de una implementación del DOM.

El lenguaje fue inventado por Brendan Eich en la empresa Netscape Communications, que es la que desarrolló los primeros navegadores web comerciales. Apareció por primera vez en el producto de Netscape llamado Netscape Navigator 2.0.

Tradicionalmente, se venía utilizando en páginas web HTML, para realizar tareas y operaciones en el marco de la aplicación únicamente cliente, sin acceso a funciones del servidor. JavaScript se ejecuta en el agente de usuario al mismo tiempo que las sentencias van descargándose junto con el código HTML.

En 1997 los autores propusieron JavaScript para que fuera adoptado como estándar de la European Computer Manufacturers' Association ECMA, que a pesar de su nombre 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 lo fue como un estándar ISO.

JScript es la implementación de ECMAScript de Microsoft, muy similar al JavaScript de Netscape, pero con ciertas diferencias en el modelo de objetos del navegador que hacen a ambas versiones con frecuencia incompatibles.

Para evitar estas incompatibilidades, el World Wide Web Consortium diseñó el estándar Document Object Model (DOM, ó Modelo de Objetos del Documento en castellano), que incorporan Konqueror, las versiones 6 de Internet Explorer y Netscape Navigator, Opera versión 7, y Mozilla desde su primera versión.

Para realizar los menús desplegables de la plataforma, por ejemplo, hemos utilizado esta tecnología

AJAX

Ajax, acrónimo de Asynchronous JavaScript And XML (JavaScript asíncrono y XML), es una técnica de desarrollo web para crear aplicaciones interactivas o RIA (Rich Internet Applications). Éstas se ejecutan en el cliente, es decir, en el navegador de los usuarios y mantiene comunicación asíncrona con el servidor en segundo plano. De esta forma es posible realizar cambios sobre la misma página sin necesidad de recargarla. Esto significa aumentar la interactividad, velocidad y reusabilidad en la misma.

AJAX es una combinación de tres 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 ECMAScript como JavaScript y JScript, para mostrar e interactuar dinámicamente con la información presentada.
- El objeto XMLHttpRequest para intercambiar datos asincrónicamente 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.

XML es el formato usado comúnmente para la transferencia de vuelta al servidor, aunque cualquier formato puede funcionar, incluyendo HTML preformateado, texto plano, JSON y hasta EBML.

Como el 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.

3. 5 Librerías Utilizadas

Jfreechart

JFreeChart es una libre 100% Java biblioteca gráfica que se facilita a los desarrolladores para mostrar gráficos de calidad profesional en sus aplicaciones. Sus características principales son:

- API coherente y bien documentada, que contiene una amplia gama de tipos de gráficos;
- Un flexible diseño que es fácil de extender, tanto en aplicaciones distribuidas, como en aplicaciones cliente
- Muchos tipos de devolver la salida de la imagen, incluyendo los componentes Swing, archivos de imagen (incluyendo PNG y JPEG), y los formatos de archivos de gráficos vectoriales (incluyendo PDF, EPS y SVG)
- JFreeChart es "código abierto" o, más concretamente, es software libre. Se distribuye bajo los términos de la GNU Lesser General Public Licence (LGPL), que permite su uso en aplicaciones propietarias.

Jcrontab

Jcrontab es un planificador de tareas totalmente escrito en Java, diseñado con el propósito de brindar al usuario un sistema de planificación de tareas completo y funcional, además de específico para proyectos Java.

Nos brinda un sistema que permite planificar la ejecución de clases, threads de control, clientes EJB, servlets, programas nativos y mucho más en un tiempo dado, ofreciendo un "parser" (analizador sintáctico) compatible con los ficheros cron UNIX-POSIX; las clases empotradas pueden usarse en cualquier proyecto.

Además de incluir un servlet que se integra con los servidores de aplicaciones como Tomcat, resin, Jetty, Jboss y muchos más, puede funcionar de forma independiente como sustituto del cron. Ofrece un entorno Web y, concienzudo, nos envía un mensaje de correo electrónico una vez los resultados obtenidos.

Sus características principales las podemos resumir a continuación:

- Puede ejecutarse embebido en otras aplicaciones.
- Puede ser instanciado dentro de un servidor web o de aplicaciones y participar en transacciones distribuidas.
- Puede ejecutarse como aplicación independiente.
- Puede usarse como un clúster de programadores de tareas.
- Las tareas pueden ser ejecutadas: en un cierto momento (precisión de milisegundos), ciertos días de la semana, ciertos días del mes, ciertos días del año, todos los días menos unos determinados, repetirse una serie de veces, repetirse hasta una fecha, etc

- Posibilidad de introducir listeners para que se ejecuten otras tareas simultáneamente.
- Incluye la posibilidad de guardar trabajos y eventos en base de datos.

Capitulo 4.Diseño del Sistema

4.1 Casos de Uso

Los casos de uso, mostrados a continuación, son los pertenecientes a las mejoras hechas en la plataforma Paula, con la realización de este proyecto. Se omiten todos los demás casos de uso de la Plataforma:

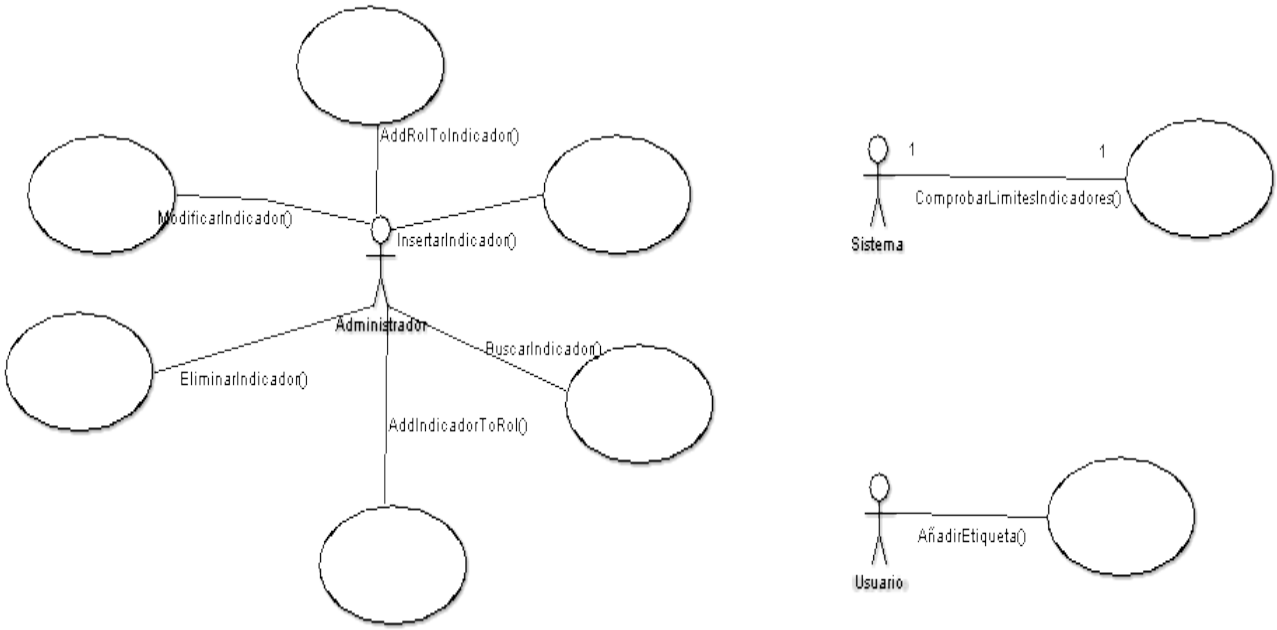


Ilustración 1: Casos de uso del sistema

Caso De Uso 1: InsertaIndicador

Objetivo:

Registrar un indicador dentro de la aplicación Plataforma Paula .

Actor principal:

Administrador

Precondiciones:

El administrador se ha autenticado en el sistema.

Escenario principal:

1. El administrador crea un nuevo indicador.
2. El administrador introduce todas las propiedades necesarias para dar de alta al indicador.
3. El sistema asocia un identificador interno al indicador recién creado.
4. El sistema añade el indicador al catálogo de indicadores.
5. El sistema muestra la información del indicador recién añadido a la lista de indicadores junto a los demás indicadores del sistema

Diagrama de Colaboración

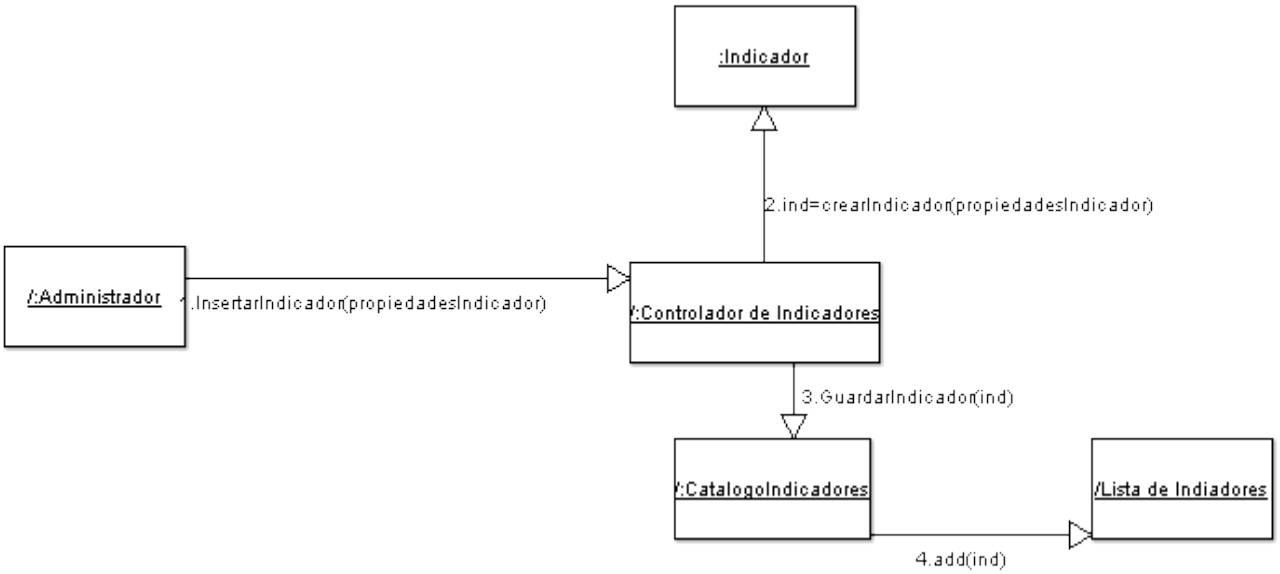


Ilustración 2: Diagrama colaboración InsertarIndicador

Caso De Uso 2: ModificarIndicador

Objetivo:

Modificar un indicador dentro de la aplicación Plataforma Paula .

Actor principal:

Administrador

Precondiciones:
El indicador a modificar ha sido creado en el sistema.

Escenario principal:

1. El administrador selecciona un indicador de la lista de indicadores.

2. El administrador introduce la opción de Editar un Indicador
3. El sistema recupera el Indicador asociado a ese indicador.
4. El sistema muestra las propiedades del indicador al Administrador
5. El administrador modifica uno o varios campos del Indicador recuperado.
6. El sistema guarda el indicador en el catalogo de indicadores
7. El sistema muestra la información del indicador recién añadido a la lista de indicadores junto a los demás indicadores del sistema

Diagrama de Colaboración

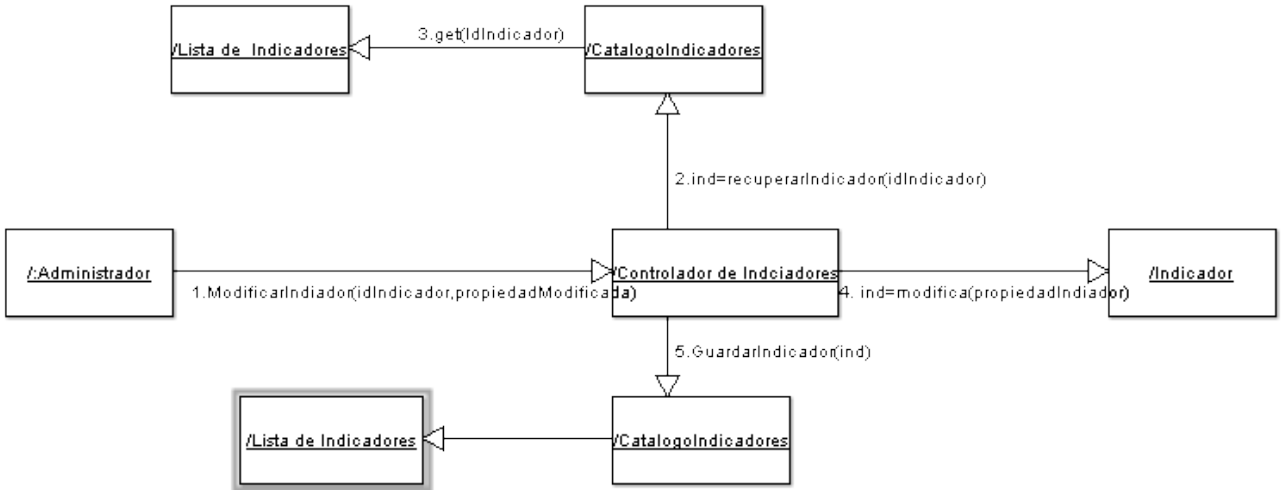


Ilustración 3: Diagrama de Colaboración ModificarIndicador

Caso De Uso 3: EliminarIndicador

Objetivo:

Eliminar un indicador dentro de la aplicación Plataforma Paula .

Actor principal:

Administrador

Precondiciones:

El administrador se ha autenticado en el sistema.
El indicador debe existir en el sistema.

Escenario principal:

1. El administrador selecciona un indicador de la lista de indicadores.
2. El administrador introduce la opción de Eliminar un Indicador
3. El sistema elimina el Indicador asociado a ese indicador.
4. El sistema muestra la lista de indicadores del sistema

Diagrama de Colaboración

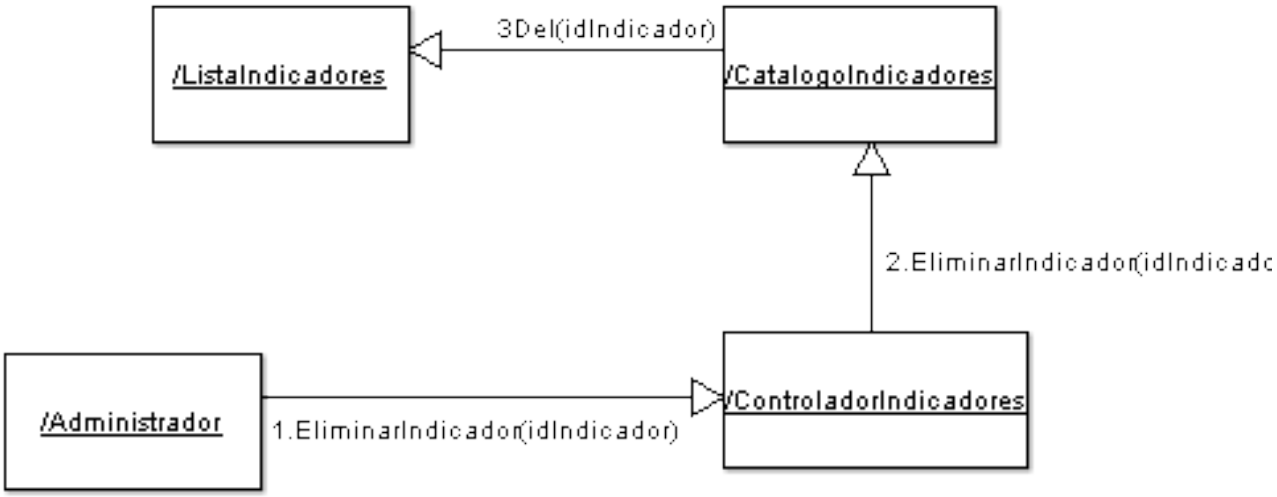


Ilustración 4: Diagrama de Colaboración de EliminarIndicador

Caso De Uso 4: BuscarIndicador

Objetivo:

Buscar un indicador dentro de la aplicación Plataforma Paula .

Actor principal:

Administrador

Precondiciones:

El administrador se ha autenticado en el sistema.

Escenario principal:

1. El administrador introduce un termino en el buscador para realizar la búsqueda
2. El sistema recupera la lista de indicadores
3. El sistema realiza un filtro sobre la lista obteniendo otra lista con los indicadores que cumplen la condición.
4. El sistema muestra la información de la lista de indicadores del sistema

Diagrama de Colaboración

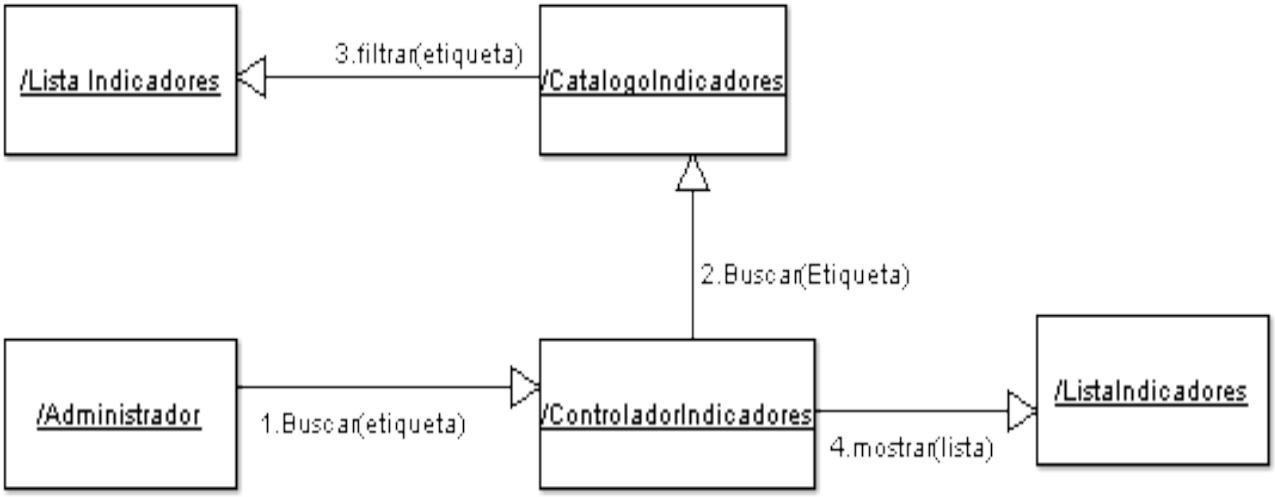


Ilustración 5: Diagrama de Colaboración Buscar

Caso De Uso 5: AddRolToIndicador

Objetivo:

Añadir un permiso a un indicador dentro de la aplicación Plataforma Paula .

Actor principal:

Administrador

Precondiciones:

El administrador se ha autenticado en el sistema.
El indicador y el rol existen en el sistema

Escenario principal:

1. El administrador selecciona un indicador.
2. El administrador selecciona un rol para asociar al indicador
3. El sistema asocia un identificador interno a la asociación.
4. El sistema guarda la asociación IndicadorRol en el sistema
5. El sistema muestra la información del indicador al cual se le acaba de añadir el permiso en el sistema

Diagrama de Colaboración

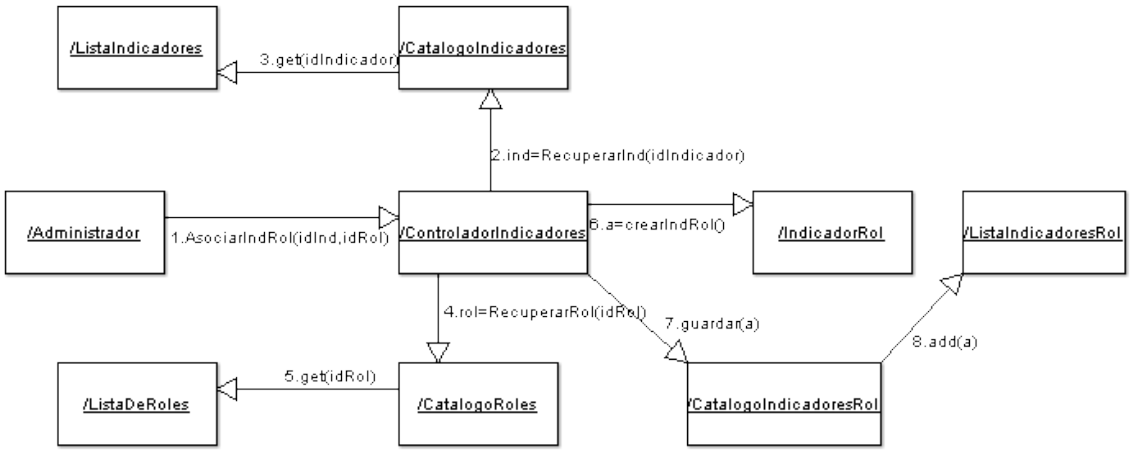


Ilustración 6: Diagrama de colaboración AddRolToIndicador

Caso De Uso 6: AddIndicadorToRol

Objetivo:

Añadir un indicador a un rol dentro de la aplicación Plataforma Paula .

Actor principal:

Administrador

Precondiciones:

El administrador se ha autenticado en el sistema.
El Rol y el Indicador existen dentro del sistema

Escenario principal:

1. El administrador selecciona un rol.
2. El administrador selecciona un indicador para asociar al indicador
3. El sistema asocia un identificador interno a la asociación.
4. El sistema guarda la asociación IndicadorRol en el sistema
5. El sistema muestra la información del indicador al cual se le acaba de añadir el permiso en el sistema

Diagrama de Colaboración

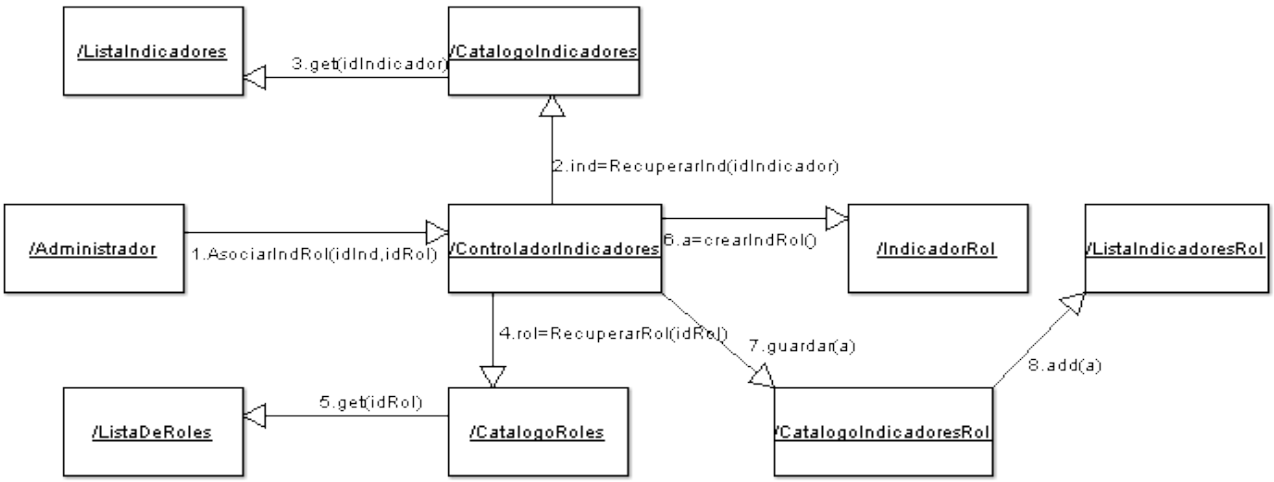


Ilustración 7: AddIndicadorToRol

Caso De Uso 7: ComprobarLimitesIndicadores

Objetivo:

Comprobar que el valor del indicador no sobrepasa los limites de de los indicadores dentro de la aplicación Plataforma Paula .

Actor principal:

Sistema

Precondiciones:

La plataforma se ha iniciado con éxito

Escenario principal:

1. El sistema recupera la lista de indicadores.
2. El sistema comprueba el valor del indicador recuperado
3. Si el limite inferior o superior es rebasado el sistema lanza un flujo de trabajo asociado al indicador
4. El sistema añade flujo de trabajo a la consola de nodos.

Diagrama de Colaboración

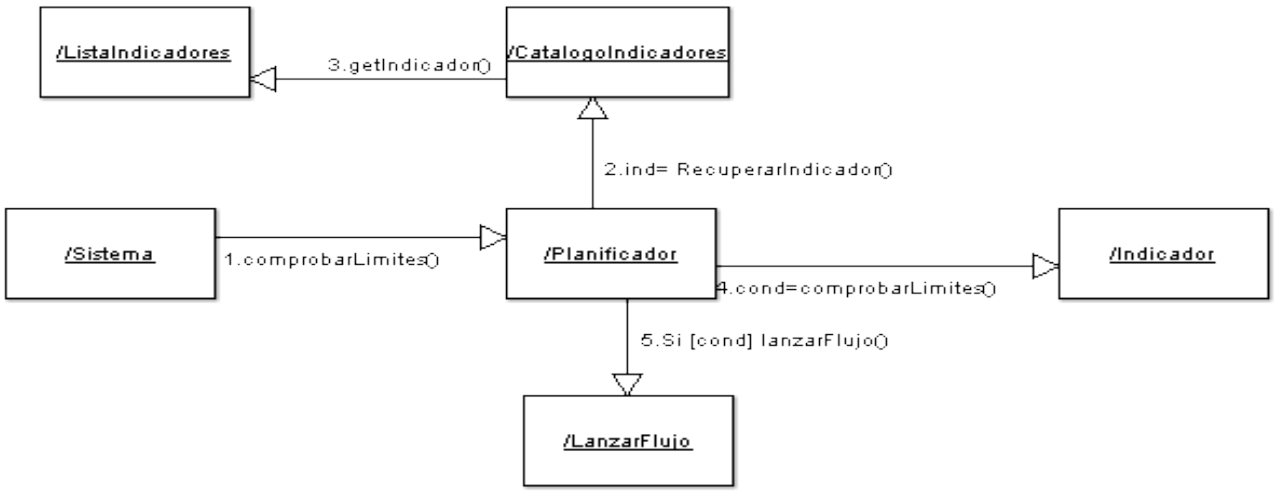


Ilustración 8: ComprobarLimitesIndicador

Caso De Uso 8: añadirEtiqueta

Objetivo:

Asociar una etiqueta a un indicador dentro de la aplicación Plataforma Paula .

Actor principal:

Usuario

Precondiciones:

El usuario se ha autenticado en el sistema.
El indicador existe en el sistema

Escenario principal:

1. El usuario introduce una etiqueta a un indicador
2. El sistema recupera el indicador asociado al id de este componente.
3. El sistema asocia la etiqueta al indicador creando una instancia de EtiquetaIndicadoresAsociada.
4. El sistema añade la asociación al catálogo de EtiquetasIndicadores.
5. El sistema muestra el escritorio al usuario

Diagrama de Colaboración

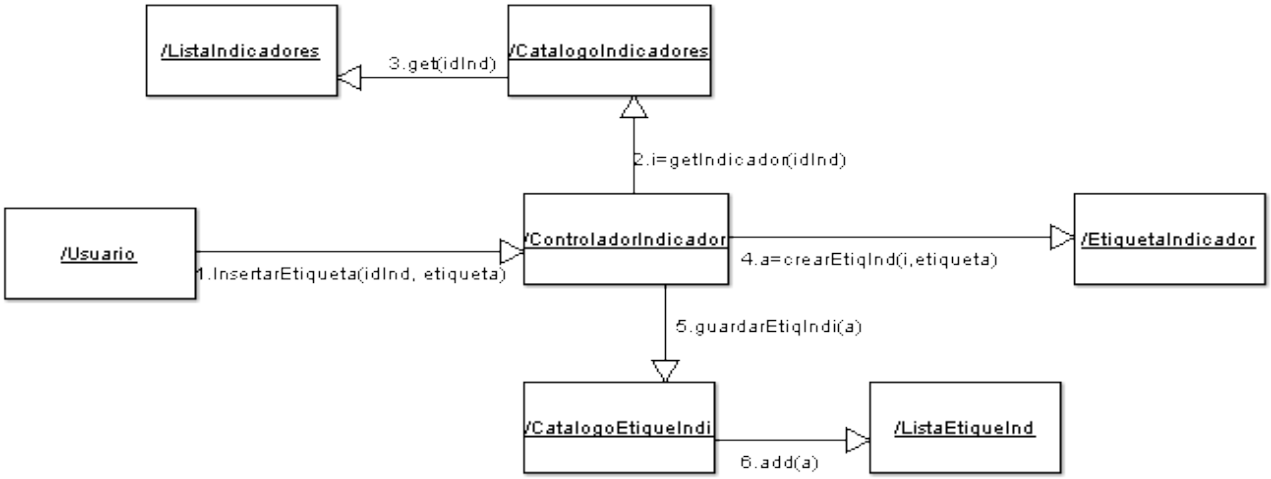


Ilustración 9: AñadirEtiqueta

4.2 Diagrama de clases

Mostramos a continuación el diagrama de clases del desarrollo realizado. Hemos omitido algunos métodos y atributos para la mejor comprensión del diagrama:

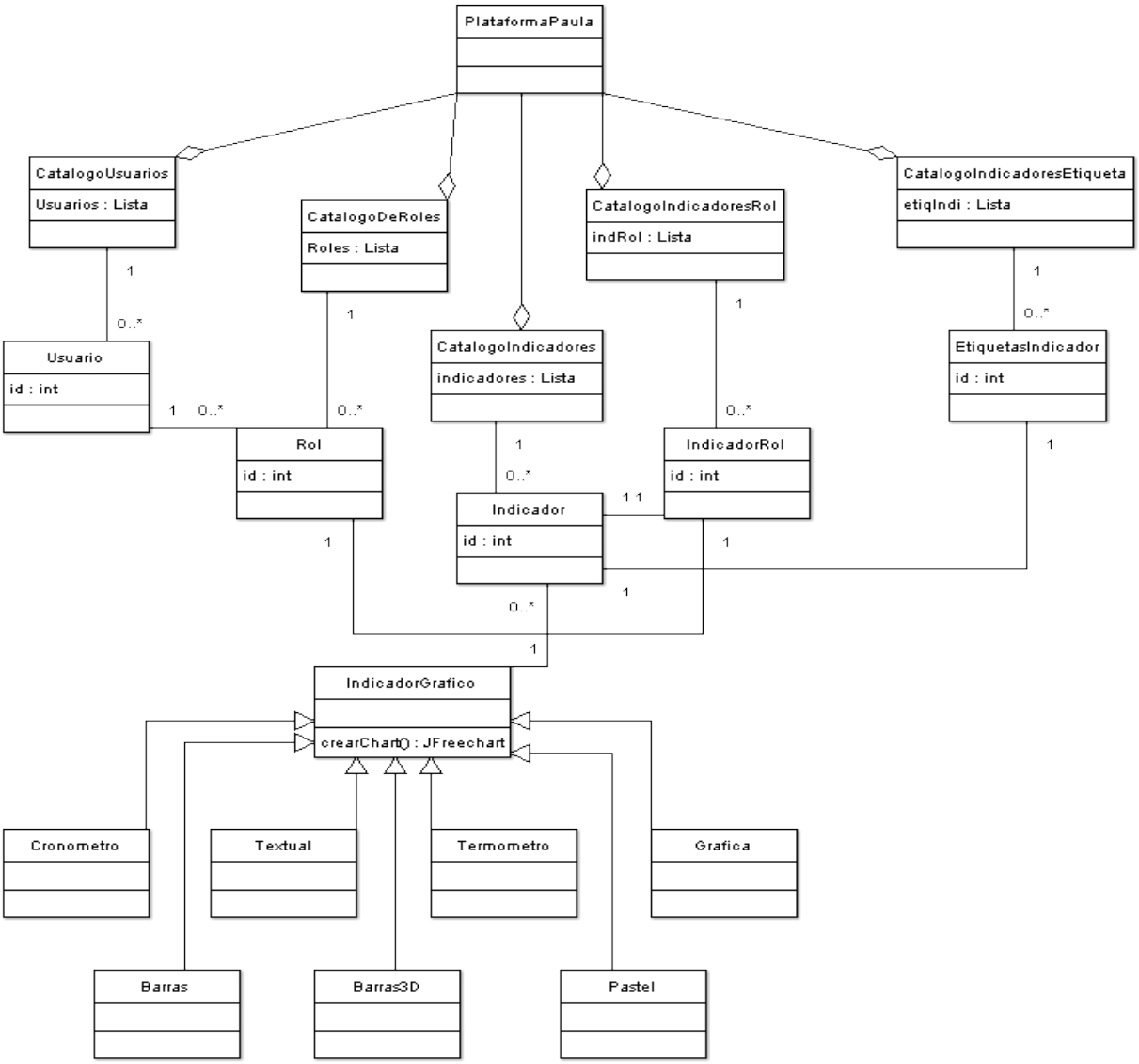


Ilustración 10: Diagrama de clases del sistema

Capitulo 5.Resolución del Problema e implementación de la solución

El sistema que queremos desarrollar estará dividido en dos contextos diferentes. El primero estará orientado a la gestión de indicadores y roles de usuario, por parte del Administrador. El segundo estará orientado a la presentación del Dashboard en el módulo del Escritorio de la plataforma Paula. Se quiere conseguir un buen diseño de indicadores y que además estos sean capaces de alertar al usuario al recibir un evento de sistema fuera de un rango esperado.

5.1 Problema planteado

Se propone desarrollar un Dashboard para un conjunto de aplicaciones empresariales ERP de la Plataforma PAULA de Aquiline Computer SL.

El objetivo es tener un panel web en que se muestren la informa instantánea del estado de los recursos empresarias. Para ello empezaremos distinguiendo cuales son los elementos gráficos que son susceptibles de utilizar así como su comportamiento dependiendo del valor de un determinado indicador. Es necesario hacer un análisis de los perfiles que pueden tener acceso al sistema y que tendrán distintos indicadores según sus necesidades de trabajo. Por último se diseñará y desarrollará toda la interfaz entre el panel y el sistema de planificación de recursos empresariales correspondiente a la plataforma P.A.U.L.A. Todo el desarrollo se hará teniendo en cuenta que el entorno de ejecución en un navegador web, pudiendo hacer uso de las tecnologías más novedosas en AJAX y Web 2.0.

5.2 Descripción general de la arquitectura de la solución

Debido al uso de la tecnología J2EE, podemos dividir la arquitectura del trabajo desarrollado en 4 niveles.

En el primer nivel, se encuentra todo lo relacionado con la interfaz gráfica del usuario, la vista de la aplicación. Son las páginas que le aparecen al cliente en su navegador, son las paginas JSPs desarrolladas.

En un segundo nivel, encontramos los servlets que realizan la misión de redirigir los datos y eventos producidos en la capa web, a las clases que deben tratar esos datos o eventos. Los servlets hacen la misión de controlador del sistema. Así tendremos, por ejemplo, un controlador para controlar los indicadores.

En el tercer nivel, tendremos el modelo de clases que conforma el sistema, dentro del contenedor de servicios.

En el cuarto nivel tendremos el modelo de datos, de la base de datos Postgres.

Un ejemplo de esta arquitectura podría ser la mostrado a continuación:

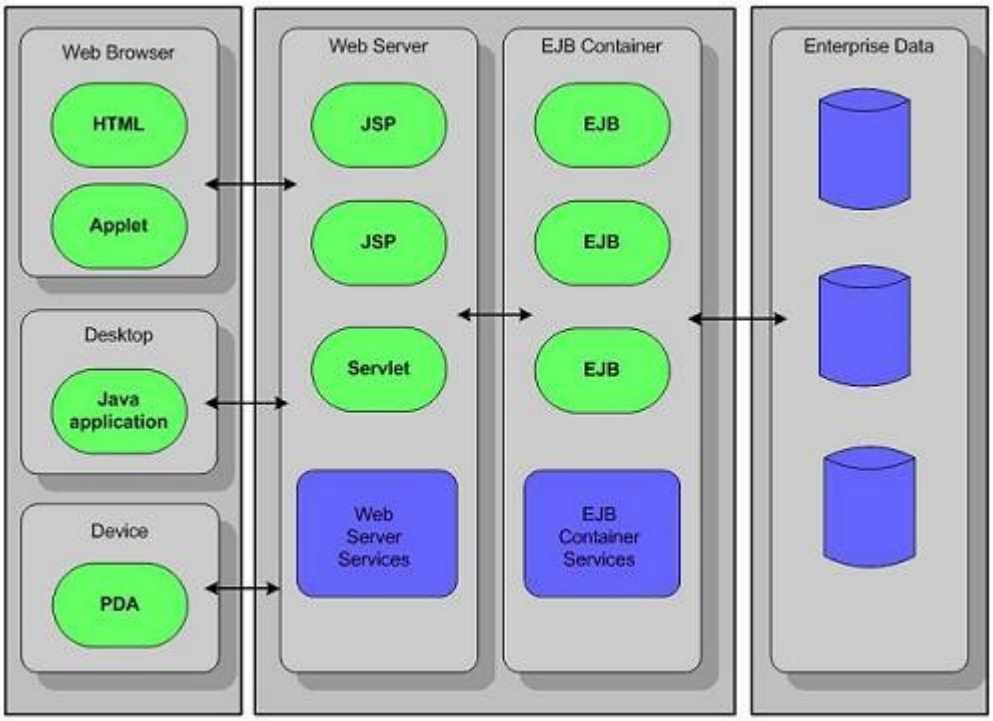


Ilustración 1: Ejemplo arquitectura J2EE

Con este tipo de arquitectura conseguiremos:

- Modificar y actualizar la información frecuentemente
- Facilita la integración con los sistemas existentes
- Favorece la escalabilidad de la aplicación y la seguridad de la aplicación

Así, podremos separar el modelo del sistema, que encapsula la lógica de negocio y el modelo de datos, de la vista. Todo ello controlado por medio de los diferentes controladores.

5.3 Modelo de datos:

El diagrama Entidad Relación construido, esta formador por un esquema sencillo, pero que resuelve eficientemente las relaciones que necesitamos tener entre las tablas. Se han englobado en un atributo entidadPropiedades los campos menos significativos de la entidad. Un pequeño esquema se muestra a continuación:

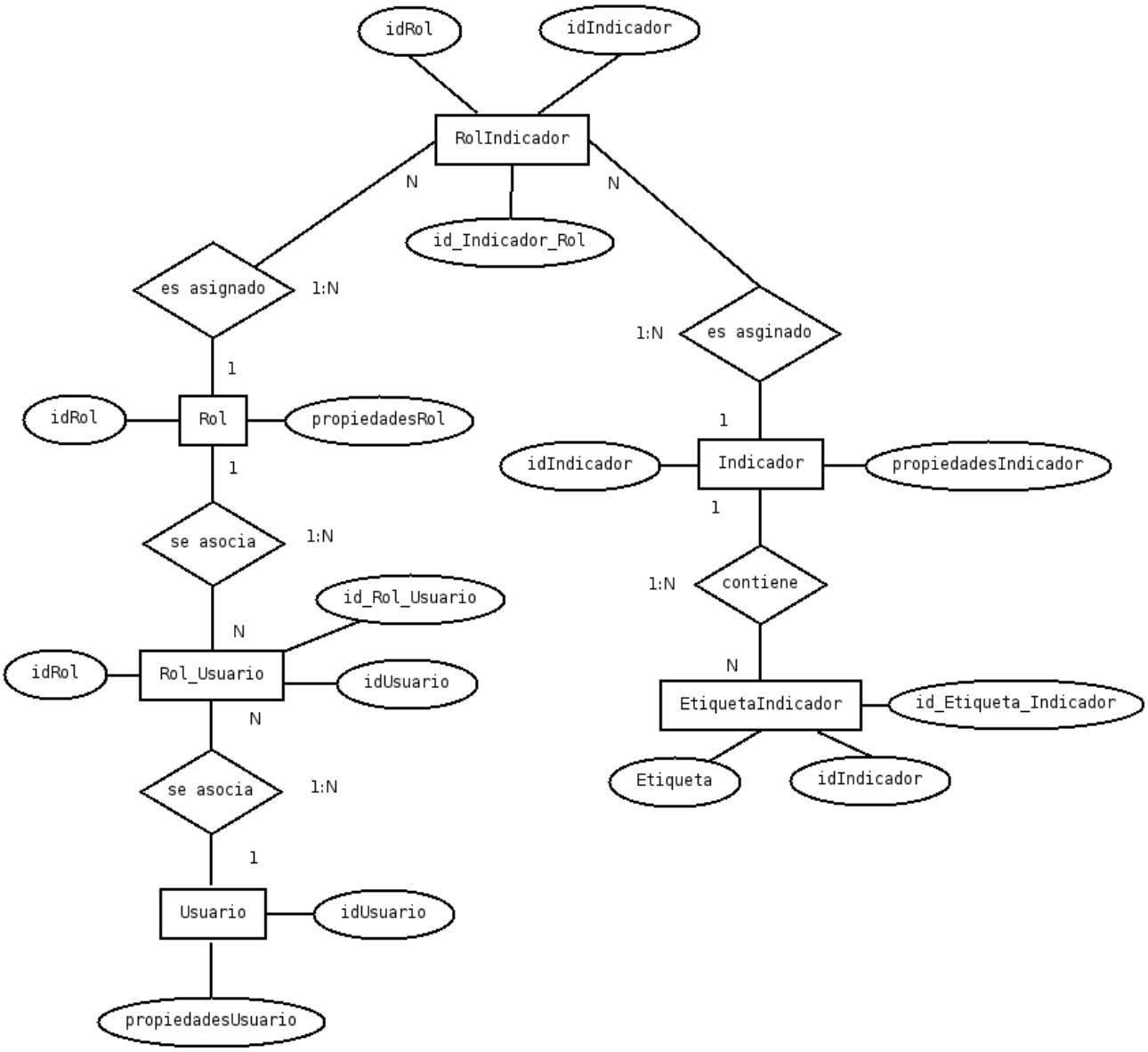


Ilustración 2: Diagrama E-R

5.4 Indicadores

Concepto Teórico de Indicador

Para la configuración del indicador, tendremos que conocer bien, cada una de sus propiedades, representadas internamente por la clase Indicador. Las mostramos a continuación:

- **Nombre:** El título del indicador que queremos introducir en el sistema.
- **Descripción:** Un breve resumen del uso del indicador, y cualquier cosa relacionada con este que se quiera destacar. No se muestra al usuario. Solo puede verla el administrador encargado de asignar los indicadores a los usuarios.
- **url:** Damos la posibilidad de introducir una url, que al pinchar encima del indicador nos redirija el sistema a esa página. Con esto queremos conseguir que un indicador que esté relacionado por ejemplo con un flujo de trabajo, al pinchar encima de este, nos lleve directamente al flujo relacionado.
- **Sql:** Propiedad muy importante. Según lo completa y efectiva que sea la consulta sql introducida, el valor del indicador será más óptimo.
- **Tipo:** Con esta propiedad indicaremos que componente gráfico mostrará el indicador.
- **Orden:** Será la disposición con la que se irán pintando en el cuadro de mandos. Los mostraremos de menor a mayor.
- **Rango superior:** marca el dominio superior del indicador
- **Rango inferior:** marca el dominio inferior del indicador
- **Límite superior:** marca superior que al sobrepasarla accionará el lanzamiento de una orden.
- **Límite inferior:** marca inferior que al sobrepasarla accionará el lanzamiento de una orden.
- **Tiempo entre avisos:** tiempo en segundos que se volverá a ejecutar una acción según un evento recibido. Esto es, el tiempo que transcurrirá, desde el lanzamiento del primer flujo de trabajo, hasta el lanzamiento de un nuevo flujo de trabajo, si la incidencia sigue sin ser resuelta.
- **Acción superior:** acción que se realizará al sobrepasar el límite superior del indicador.
- **Acción inferior:** acción que se realizará al sobrepasar el límite inferior del indicador.
- **Fecha inicial:** cuando empezará a funcionar
- **Fecha final:** cuando terminará de funcionar.
- **Activo:** Si está marcado, el indicador será apto para ser pintado en el cuadro de mandos, en caso de que no esté marcado el sistema lo tomará como un indicador no apto.

Vamos a identificar cada campo en un indicador para mejor comprensión del lector:

Ilustración 3: Cronometro

En la Imagen anterior:

- La propiedad *Nombre* se corresponderá con el título “Facturas de compras”
- La *descripción* nos resumirá en breves palabra que este tipo de indicador mide el número de facturas de compra que la empresa ha pagado, del total de facturas que debe pagar.
- La *url* nos llevará al Flujo de trabajo de facturas de compra.
- La ejecución de la sentencia *sql* nos muestra el resultado del cuadro, el valor 26.
- El *tipo* será el etiquetado con la letra 'A', ya que lo muestra el tipo indicador Cronometro. Mas adelante explicamos la relación del tipo de visualización y la letra asignada.
- El campo *Orden*, suponiendo que apareciera junto a otro indicador, y este se cargará primero en la página, será de un valor menor, que el del otro indicador.
- El *rango superior*, tendrá un valor de 50.
- El *rango inferior*, tendrá un valor de 0.
- El *límite superior,* que delimita el color verde y amarillo, tendrá un valor de 40.
- El *límite inferior,* que delimita el color rojo y amarillo, tendrá un valor de 20.
- La *orden superior,* en este caso no hará nada, debido a que es el estado ideal para la empresa.
- La *orden inferior,* lanzará el proceso de pago de facturas de compra, avisando al usuario, de que debe realizar esta acción con urgencia.
- El *tiempo entre avisos* es el tiempo que tardará el indicador en volver a avisar al usuario, si se sigue manteniendo el número de facturas de compras en la zona roja.
- El campo *Activo* habrá sido pinchado por el administrador, ya que el usuario puede ver el indicador.

Diferentes visualizaciones de los Indicadores

Una de las características más importantes del cuadro de mando, es que es totalmente

configurable. Según el valor que pongamos en el campo Tipo, explicado anteriormente, tendremos una representación de la información diferente. Así, cuando configuremos un indicador podremos elegir la representación de este.

Podemos clasificar los distintos componentes en dos grupos. Por una parte tenemos los que muestran un solo valor, y por otro los que muestran un conjunto de estos. De esta manera, tendremos unos tipos de componente gráfico convenientes para los indicadores que necesiten mostrar un conjunto de valores, como podría ser una gráfica, y otros tipos que serán más provechosos cuando el indicador introducido solo necesite publicar un solo valor.

A continuación mostramos los diferentes componentes:

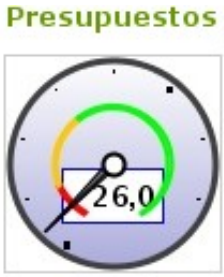


Ilustración 4: Cronometro

Este tipo de indicador, emula a un cuenta kilómetros de coche, marcando el valor devuelto por la consulta de la BBDD. Tiene una parte marcada en rojo, que significara que el valor mostrado está en una situación crítica. Si esta en naranja, el valor está en una situación delicada, y si esta en verde, quiere decir que la situación es ideal. Está etiquetado con la letra A.

Ilustración 5: Termómetro

Muy parecido al anterior con la diferencia de que las zonas marcadas como critica, delicada o ideal, están colocadas de manera inversa. Así, al contrario del anterior, el valor mostrado, cuanto menor sea, mejor será, y según vaya subiendo este, nos acercaremos a la zona critica. Está etiquetado con la letra C.

Ilustración 6: Quesito

La ilustración 28,es recomendado, cuando el valor que se va a mostrar , es un tanto por ciento, mostrándonos el componente, el tanto por ciento que cumple la condición. Está etiquetado con la letra D.

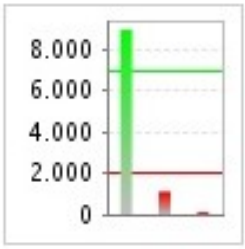


Ilustración 7: Barras 2D

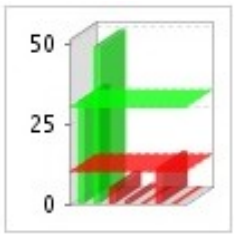


Ilustración 8: Barras 3D

Estos componentes, realmente los podemos tomas como el mismo, ya que solo varía el diseño del componente al mostrarse. Uno muestra los valores de las barras en 2 dimensiones y el otro en 3 dimensiones. Podemos agruparlos dentro de los componentes que muestran un conjunto de datos. Como podremos observar también muestran las líneas que diferencian los valores que están en un nivel normal, y los que están fuera de esta normalidad. Si sobrepasan el límite superior o inferior, representadas por una línea verde y otra roja, el valor recogido de la bbdd estará fuera de la normalidad. El primero está etiquetado con la letra E y el de la derecha con la F

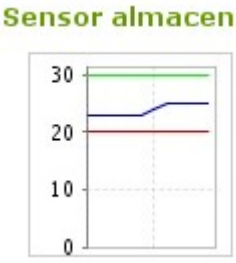


Ilustración 9: Gráfica

Con este tipo de gráfica, sacaremos los valores con una línea continua a diferencia de los anteriores que lo hacía con barras. Según el tipo de indicador a configurar, elegiremos el que más nos agrade

visualmente. Está etiquetado con la letra G.

Ilustración 10: Indicador Texto

También cabe la posibilidad de mostrar el resultado en modo texto, pintándose el valor en rojo, si el valor mostrado esta en un estado crítico, naranja, si esta en un estado de atención, y verde, si el valor esta en un estado ideal. Está etiquetado con la letra B.

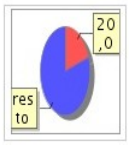
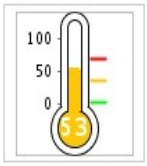
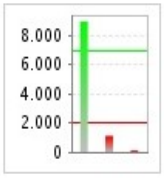
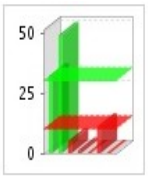
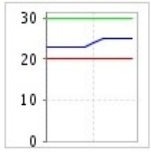
En el apéndice se muestran un conjunto de indicadores teóricos, para completar la comprensión de los diferentes tipos de indicadores que podríamos usar, para cada tipo de componente gráfico.

El desarrollo de los indicadores esta basado en la librería JFreeChart. Esta librería contiene una gran cantidad de gráficos para poder usar libremente por los desarrolladores.

Hemos construido una jerarquía de clases, con una clase Indicador Gráfico genérico, que contendrá todas las propiedades y métodos comunes de cada indicador. De esta clase heredarán lo diferentes tipo de indicadores gráficos, implementando cada uno un método abstracto de la clase padre, llamado construirChart(). Dependiendo del tipo de indicador que valla a dibujar,en cada clase y con la ayuda de la librería, construimos en este método el indicador correspondiente.

Es importante destacar que las propiedades del indicador son recuperadas de la base de datos, dependiendo del id que le pasemos. Este id esta asociado con el indicador que hemos dado de alta desde la gestión de Paula. Tanto cuando recuperamos un objeto indicador de la base de datos como cuando lo guardamos, utilizamos el componente hibernate, haciendo uso de sus métodos de recuperación y inserción de la base de datos. También para ejecutar las diferentes sentencias sql que contienen los indicadores.

A continuación mostramos un cuadro comparativo, recomendando el uso de cada visualización para una situación concreta:

Presupuestos 	Recomendado para situaciones en que a mayor valor del indicador (verde), mejor será la acción medida por este. Por ejemplo el numero de productos vendidos en un periodo de tiempo.
% hr formacion 	Cuando necesitamos expresar un valor en tanto por ciento, el uso de esta visualización es ideal. Por ejemplo el % de horas de formación de la empresa.
Facturas de ventas 	El termómetro sera en antagonista de cronometro. A mayor valor del indicador, el resultado de la acción medida será peor. Por ejemplo el nº de empleados que están de baja.
	En situación que necesitamos medir un conjunto de datos, y nos interesa saber sobre todo el valor cuantitativo de cada una de las medidas devueltas. Por ejemplo el estado de stock de 3 almacenes diferentes.
	Igual que el anterior pero con una visualización en 3 dimensiones.
Sensor almacen 2 	Para el seguimiento progresivo de el estado de la entidad que estamos midiendo. Para su uso necesitamos un conjunto de datos. Como por ejemplo la temperatura de un frigorífico industrial.
TEXTO	Cuando solo queremos mostrar el valor con el color del estado de la medición.

5.5 Gestión de los indicador

Para la gestión de los indicadores y los permisos, hemos introducido una aplicación que se integra con todo el resto de la plataforma. Desde esta aplicación podremos ser capaces de dar de alta un nuevo indicador, o de asignarle permisos. También podremos relacionar una etiqueta a un indicador en particular, para enriquecerlo semánticamente, de tal manera que luego podremos hacer búsquedas de indicadores según el valor de estas etiquetas. Así si hacemos una búsqueda, según el término que introduzcamos, nos mostrara todos los indicadores que tengan asignados una etiqueta con ese término.

Una vez creado el indicador, podremos asignarle un permiso existente en el sistema. Según el rol que elijamos, se asociará un indicador a un usuario en concreto. Por ejemplo si le asignamos a un indicador financiero un permiso de Gerente, todos los usuarios con el rol gerente, al conectarse al sistema, les mostrara en su cuadro de mandos este indicador.

Ilustración 11: Indicadores asociados a un Rol

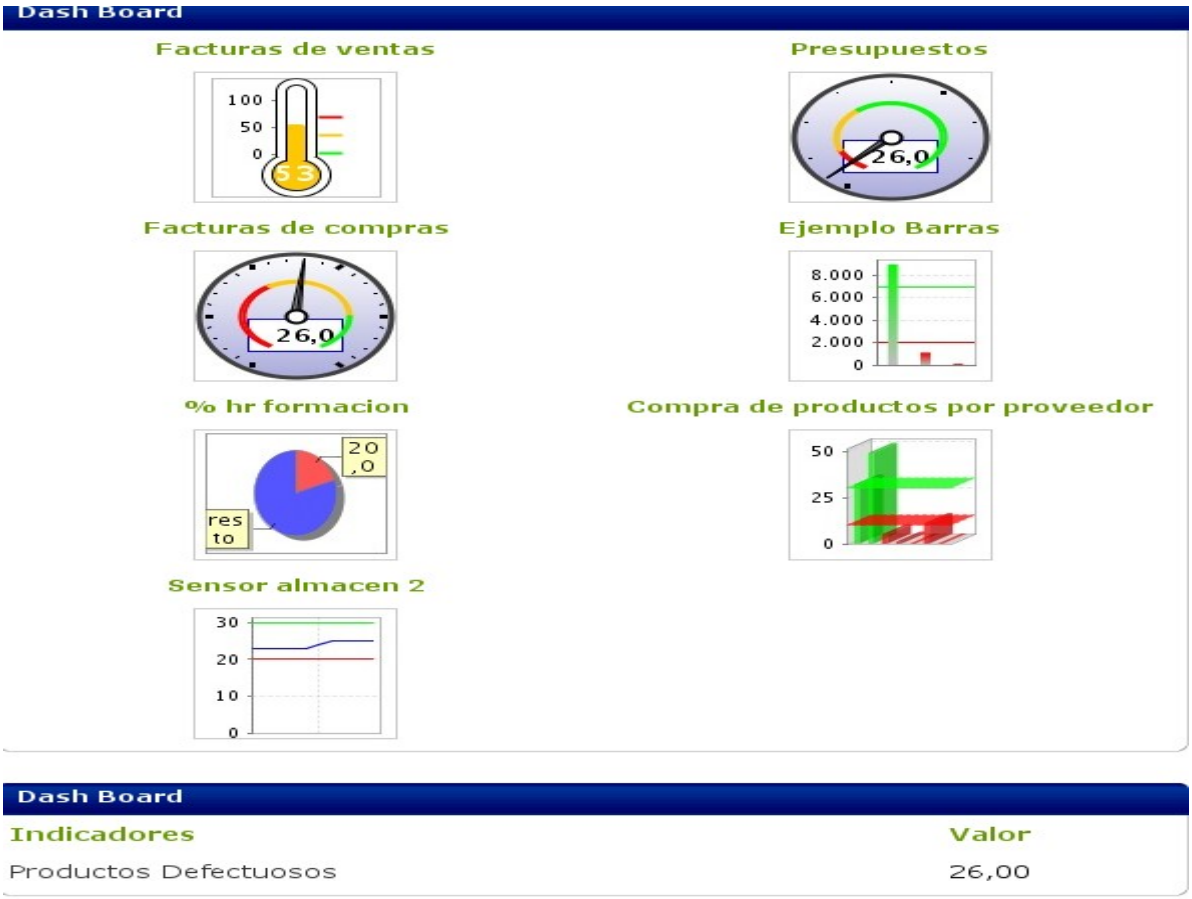


Ilustración 12: Indicadores mostrados al usuario actual

En cuanto a las etiquetas o tags, un usuario puede estar usando un tipo de indicador, por ejemplo de gestión, y asignarle una etiqueta con el término "gestión". Así, cuando queramos asignarle a un usuario todos los indicadores relacionados con "gestión", haremos una búsqueda por gestión, apareciendo todos los indicadores que contienen esta etiqueta.

Ilustración 13: Filtrado por etiquetas

Ilustración 14: Indicadores mostrados tras un filtrado

Para contener todas estas propiedades tendremos un clases indicador, que contendrá todas las diferentes características anteriores declaradas dentro de la clase, junto a los diferentes métodos para trabajar con ellos. Hay que distinguir, entre esta clase y la jerarquía de clases indicador gráfica, ya que la segunda hace uso de esta primera para formar el indicador gráfico.

La tabla rol_indicador, mantendrá la relación entre los indicadores y los permisos asignados a estos. Cuando un usuario se conecta al sistema, solo se cargaran en el escritorio los indicadores que tengan relación con el rol del usuario que se conecta.

En el caso de la asignación de etiquetas a un indicador particular, también hemos creado, una tabla que relaciona un indicador con una etiqueta concreta. De esta forma, cuando asignamos un tag, el sistema lo introduce relacionándolo con el identificador del indicador.

Destacar que hemos desarrollado una función en JAVASCRIPT que al pulsar encima de la etiqueta del sistema la colocará directamente en el textbox de introducir etiqueta.

Ilustración 15: Ventana asignar etiqueta

Ilustración 16: Etiquetas asignadas a un indicador

Cuando realizamos una búsqueda, internamente construimos una sentencia sql con la condición que el campo etiqueta coincida con el término introducido por el usuario. Esto se traduce con el lanzamiento de la Query por medio del componente hibernate sobre la tabla etiquetas_indicador.

5.6 Escritorio: Carga de los Indicadores

Para la carga de indicadores en el sistema, usaremos dos listas, una que contendrá los indicadores de tipo gráfico, y otros que contendrá los indicadores de tipo texto. Esto es así debido a que en el cuadro de mando primeramente pintaremos los indicadores gráficos, y a continuación los indicadores de texto.

Es muy importante el modo en que cargamos los indicadores, ya que tendremos que ir filtrando según el rol del usuario que acabe de conectar al sistema, y que indicadores están asignados a este. También deberemos comprobar si esta Activo el indicador asignado y por ultimo si está bien configurado.

Hemos utilizado hibernate para guardar en una variable de sesión los indicadores correspondientes al usuario en cuestión. Esta variable es recuperada luego en la JSP, e invocando al servlet que controla los indicadores que debe ser pintados en el cuadro de mando. El servlet pasa el id a la clase java correspondiente, según el tipo. Entonces la clase devolverá el chart asociado a ese id dentro de un stream, que el browser sabrá identificar como imagen, gracias a las cabeceras. El resultado final será el cuadro de mandos del usuario conteniendo los indicadores pertenecientes al usuario conectado.

Para una mejor comprensión, mostramos a continuación, en pseudocódigo[18], la carga de los indicadores del usuario conectado:

```
ListaIndicadoresGraficos = CargarIndicadoresGraficosDeLaBaseDeDatos()

ListaIndicadoresNoGraficos = CargarIndicadoresNoGraficosDeLaBaseDeDatos()

Para cada elemento "a" de la  ListaIndicadoresGraficos
    Si(Esta asociado al rol del usuario conectado)
        pintarEnDashboard(a)

Para cada elemento "b" de la  ListaIndicadoresNoGraficos
    Si(Esta asociado al rol del usuario conectado)
        pintarEnDashboard(b)
```

18 es una serie de normas léxicas y gramaticales parecidas a la mayoría de los lenguajes de programación, pero sin llegar a la rigidez de sintaxis de estos ni a la fluidez del lenguaje coloquial

5.7 Lanzamiento de workflows en segundo plano

Como hemos dicho anteriormente un indicador, podría tener relacionadas unas acciones. Estas acciones serán lanzamientos de flujos de trabajo. Un flujo de trabajo o workflow, teóricamente es el estudio de los aspectos operacionales de una actividad de trabajo: cómo se estructuran las tareas, cómo se realizan, cuál es su orden correlativo, cómo se sincronizan, cómo fluye la información que soporta las tareas y cómo se le hace seguimiento al cumplimiento de las tareas.

En nuestro sistema, un workflow automatiza la secuencia de acciones, actividades o tareas utilizadas para la ejecución del proceso, incluyendo el seguimiento del estado de cada una de sus etapas y la aportación de las herramientas necesarias para gestionarlo.

Así si por ejemplo tenemos un indicador que nos muestra el nivel de stock de un almacén interno de la empresa y el valor de este es inferior al límite mínimo que debe tener el almacén, el sistema lanzara el workflow asociado, que por ejemplo podría ser el de “pedidos de proveedores”.

Realmente, todo este trabajo esta coordinado con la ayuda de la herramienta Jcrontab. Con ayuda de sus librerías, lo único que tenemos que hacer es seguir una serie de pasos, para que luego, Jcrontab, ejecute la tarea programada.

Primeramente debemos crear una clase que realice la tarea que queremos programar. Para ello, creamos una clase que carga todos los indicadores, del usuario de esa sesión. Luego, irá recuperando cada uno de estos y comprobando si el valor que marca, ha superado el valor superior o el valor inferior. Cabe destacar, que haremos dos tipos de comprobaciones. La primera, sera para los indicadores que devuelven un solo valor, comprobando si se ha superado el limite inferior o superior. El segundo caso es mas complejo, ya que tendremos que ir comprobando en cada uno de los datos si se ha superado el nivel superior o inferior, teniendo un orden de complejidad mas alto.

También necesitaremos una librería externa, para poder lanzar un flujo desde el escritorio. Para ello hacemos hacemos uso de la librería workflow.jar, facilitada en el repositorio de la plataforma. Solo necesitamos saber, que para lanzar un flujo debemos instanciar un objeto de la clase workflow e invocar al método lanzarFlujo, pasándole el id del flujo en cuestión a lanzar. En el caso que un indicador supere el limite superior o inferior, recuperaremos el id asociado a su campo acción, y lanzaremos el flujo.

Debemos añadir todas las librerías necesarias para el buen funcionamiento de Jcrontab, y configurar en el fichero de tareas de este, el método de la clase creada, explicada anteriormente. Cambien debemos especificar, cada cuanto tiempo queremos que se compruebe la tarea. En nuestro caso, lo hemos configurado para que lo haga cada 2 minutos.

Jcrontab se lanza en segundo plano, por lo que el usuario no percibe en ningún momento que se estén realizando estas comprobaciones.

Una interfaz gráfica que se asemeja a las tareas programadas que podemos configurar en el fichero de configuración de la librería, con la salvedad, que en vez de introducir como tarea un shell de Linux, usaríamos una clase java, es la siguiente:

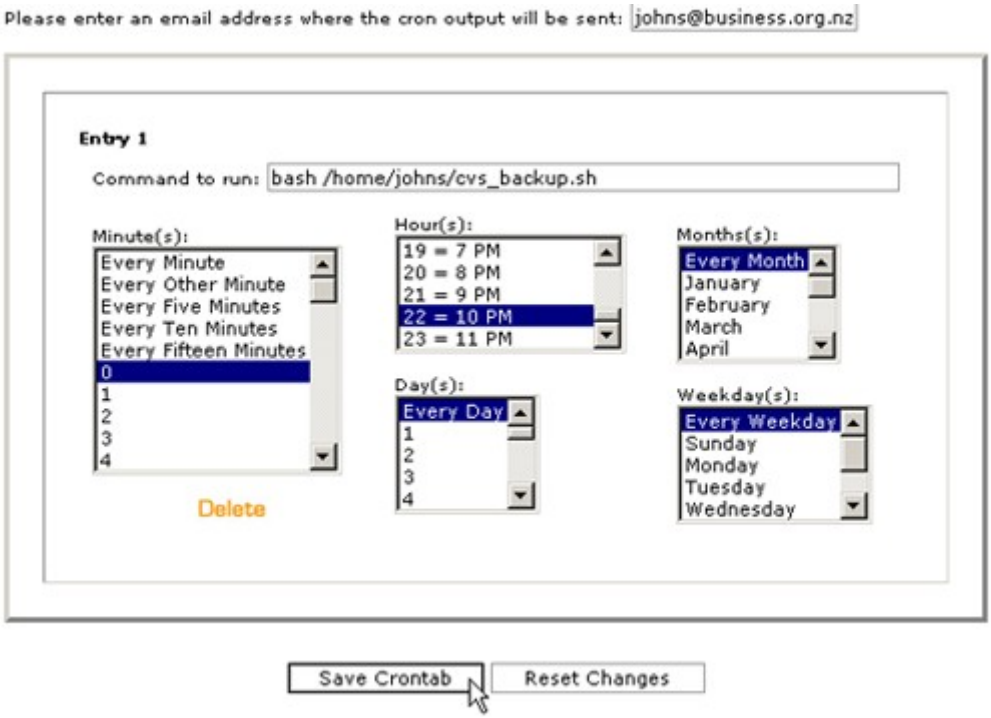


Ilustración 17: Ejemplo de un planificador

Un ejemplo de lanzamiento de workflow sería el mostrado a continuación:

-Un indicador sobrepasa uno de sus limites:

Ilustración 18: Límite inferior rebasado

-El flujo de trabajo es lanzado dentro de la consola de nodos en el escritorio del usuario:

Error	Error	Error
Inicio	CREACION FACTURA COMPRAS	28/12/2007 - 12:32:02

Ilustración 19: Flujo de trabajo lanzado

5.8 Enriquecimiento semántico por medio de tags

Cada indicador del Dashboard lleva asociado una colección de etiquetas o tags, que lo enriquece semánticamente.

Una etiqueta es una marca que identifica al indicador. La etiqueta contiene un término que relaciona al indicador con un tema concreto, para el cual el indicador podría ser interesante usarlo. Por ejemplo, a un indicador que nos midiese la temperatura de un almacén, podría ser interesante asignarle una etiqueta con el término "mediciones", ya que realiza la comprobación de la temperatura, o el término "almacén", debido a que la medición en concreto lo hace de un almacén. De esta manera, un usuario que quisiera tener asignado todos los indicadores con mediciones de algún tipo, filtraría con el término" mediciones" y el sistema le ofrecería todos los indicadores etiquetados con este tag. De igual modo, si un usuario quisiera tener todos los indicadores relacionados con el almacén, accedería al buscador, insertando almacén en el textbox, y el sistema al filtraría, devolviendo todos los indicadores relacionados con el.

Cuando un usuario introduce una etiqueta en la base de datos, y otro usuario se conecta al tiempo y quiere asignarle una etiqueta a otro indicador, el sistema le mostrara todas las etiquetas que se han introducido en el sistema hasta ese momento. Con esto conseguimos que se agrupen los indicadores con mayor eficiencia, y las búsquedas que luego se realicen, sean lo mas óptimas posibles.

Ilustración 20: Indicador con sus etiquetas asignadas

5.9 Refresco del indicador por medio de Ajax

En realidad, Ajax no existe. Es una nueva forma de trabajo que engloba a diferentes tecnologías Web que existen desde hace varios años, como el lenguaje XHTML -sucesor del actual HTML- las hojas de estilo en cascada (CSS) y JavaScript, junto a otros artilugios más técnicos como son XMLHttpRequest o XSLT. De hecho, el propio término es la abreviatura de 'Asynchronous Javascript + XML'.

Para el uso de esta tecnología en la plataforma la idea es la siguiente: en el modelo actual, la Web es un lugar muy hostil la actualización de los indicadores. Cada vez que queremos saber el valor actual exacto del indicador -un clic en actualizar pagina- el navegador debe solicitar datos a otra computadora (el servidor) a través de Internet, para luego regenerar la página que el usuario está viendo. De esta forma, la interacción se transforma en un pimpón de datos que hacen imposible cualquier actividad continua con una aplicación interactiva como esta.

Para sortear este inconveniente, varias empresas desarrollaron plataformas que se cargan dentro del navegador, como Macromedia Flash o Java, previa descarga de los plugin respectivos y previo pago de las licencias para el desarrollo. Pero, ¿qué pasaría si tuviéramos estas herramientas ya disponibles en los computadores de cada usuario y en forma nativa?

El modelo de AJAX propone cargar y renderizar una página, luego sirviéndose de las tecnologías antes mencionadas y presentes en la gran mayoría de los navegadores. De esta manera el usuario podría mantenerse en el escritorio, mientras scripts y rutinas van al servidor buscando, en background, los datos que si han cambiado, refrescaran únicamente esta parte del escritorio, manteniendo siempre los datos actualizados en el escritorio, casi en tiempo real. De esta forma, los usuarios pueden acceder de inmediato al contenido de los indicadores sin tener que estar refrescando toda la página web.

Para realizar todo este proceso, podemos diferenciar 4 etapas:

1. Creación del objeto Ajax, para ello usa el objeto XMLHttpRequest para hacer peticiones asíncronas sin que el navegador tenga que enviar la página al servidor. El objeto se ha implementado en los navegadores más utilizados (Internet Explorer, Firefox, etc.), así que basta con crear una instancia de este objeto para poder realizar peticiones HTTP al servidor.
2. Creamos una función que recibirá como parámetros el objeto Ajax, el id del indicador, la url del servlet que recuperará el valor actual del indicador, y el valor que tiene en el cuadro de mandos del usuario en su navegador. Se irá ejecutando cada 20 segundos para cada indicador. Estas peticiones son enviadas al servidor gracias al objeto a XMLHttpRequest.
3. El servlet anteriormente nombrado, cargará cada indicador directamente de la base de datos. El objeto Ajax devolverá un xml que indicará si ha cambiado el valor actual, con el cargado anteriormente en el cuadro de mandos.
4. Si detectamos que el valor de algún indicador ha cambiado, refrescamos el cuadro de mandos del usuario, sin tener que recargar el escritorio completo en el navegador.

No hemos utilizado ninguna librería externa, ya que hemos creado nuestro propio manejo del objeto XMLHttpRequest.

Mostramos a continuación el cuadro de mandos antes y después de ser refrescado gracias a la tecnología ajax. El indicador que provoca el refresco del frame es el componente Facturas de compras :

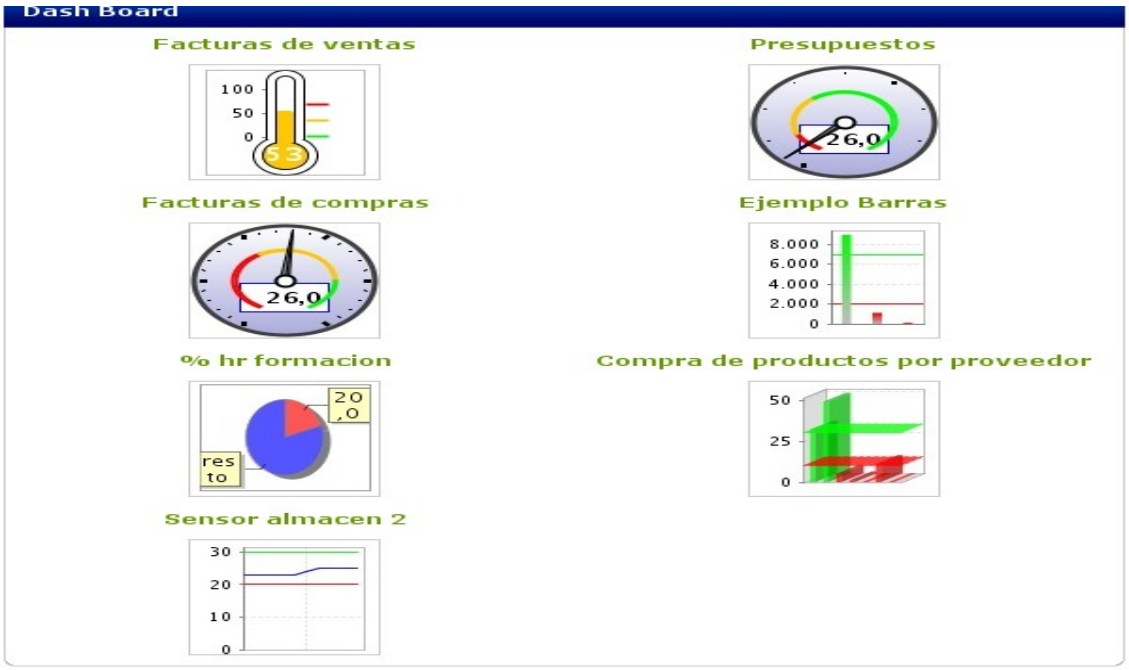


Ilustración 21: Dashboard antes de ser refrescado. Valor 26 en Facturas de Compra

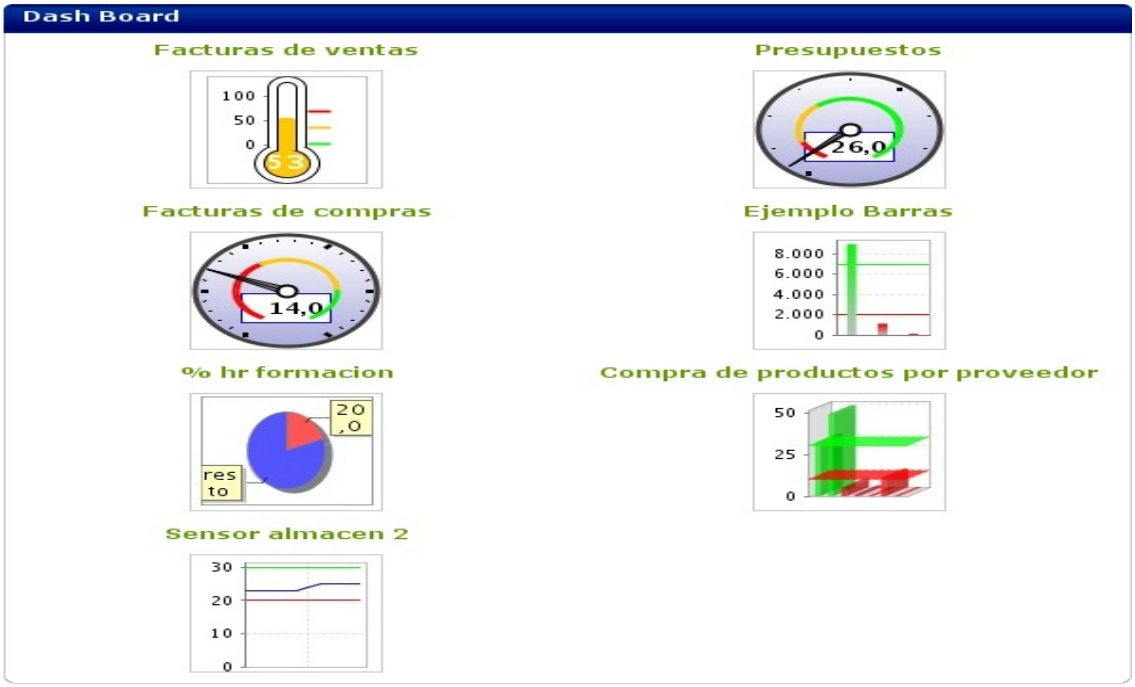


Ilustración 22: Dashboard después de ser refrescado. Valor 14 en Facturas de Compra

5.10 Funcionamiento global del sistema

Podemos acceder a la plataforma desde cualquier sistema operativo, siempre y cuando tengamos conexión a internet y un navegador instalado.

Una vez tecleado la url de la plataforma, el navegador nos redireccionara al portal de la plataforma. Este portal es una herramienta de libre distribución llamada Joomla. Con ella podremos configurar el portal de cualquier pagina web, personalizándola a nuestra conveniencia. Esta desarrollada completamente en el lenguaje PHP[19].

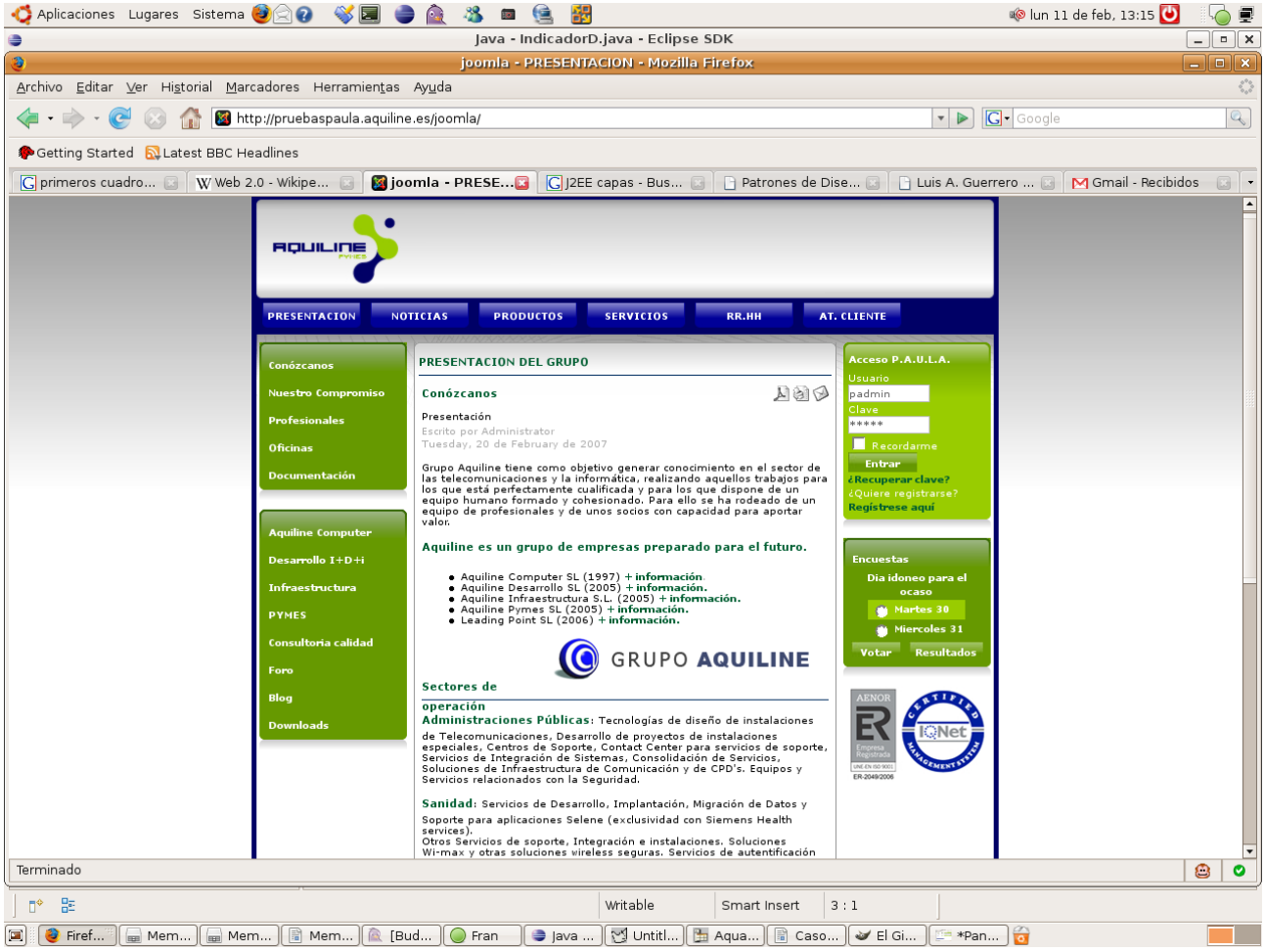


Ilustración 23: Portal de la plataforma

Una vez introducido el nombre y la contraseña de usuario, el navegador nos lleva al Escritorio, modulo que contiene el Dashboard. Desde aquí, podremos acceder a la gestión de ellos, en el modulo de gestión de Paula.

19 lenguaje de programación interpretado usado normalmente para la creación de páginas web dinámicas

5.11 Visión general de la solución

Para una solución eficiente y que cubra todos los objetivos, hemos dividido el desarrollo en dos partes. La parte de la gestión y la parte de la visualización.

Dentro de la parte de gestión de la plataforma Paula hemos introducido las funcionalidades para la creación y administración de los indicadores del Dashboard:

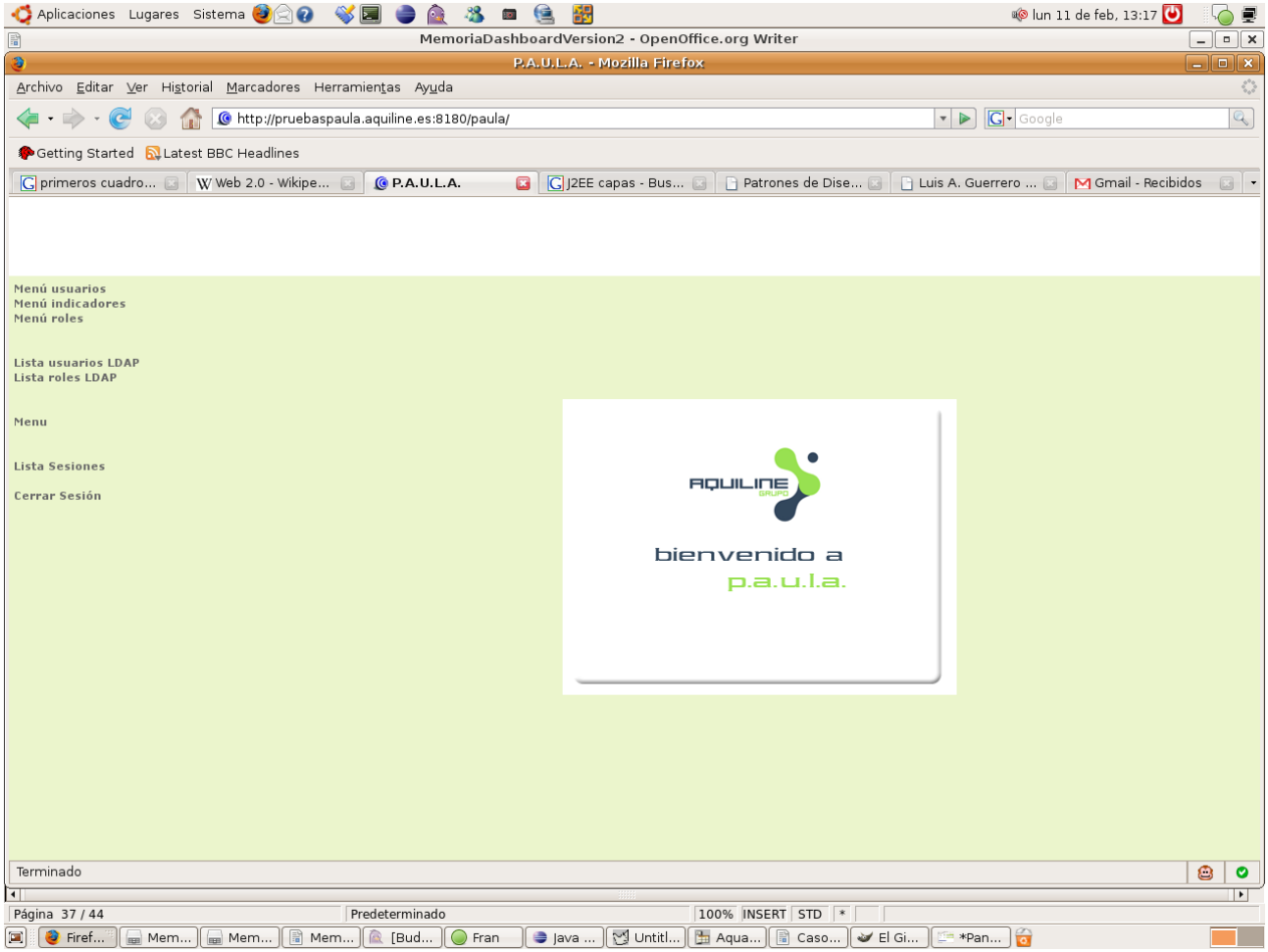


Ilustración 24: Gestión de la Plataforma

En lo que se refiere a la parte de Visualización, el cuadro de mandos lo hemos desarrollado dentro del escritorio de la plataforma Paula. Es el sitio más indicado para ello, ya que en él, se centraliza todo el control dentro de la plataforma:

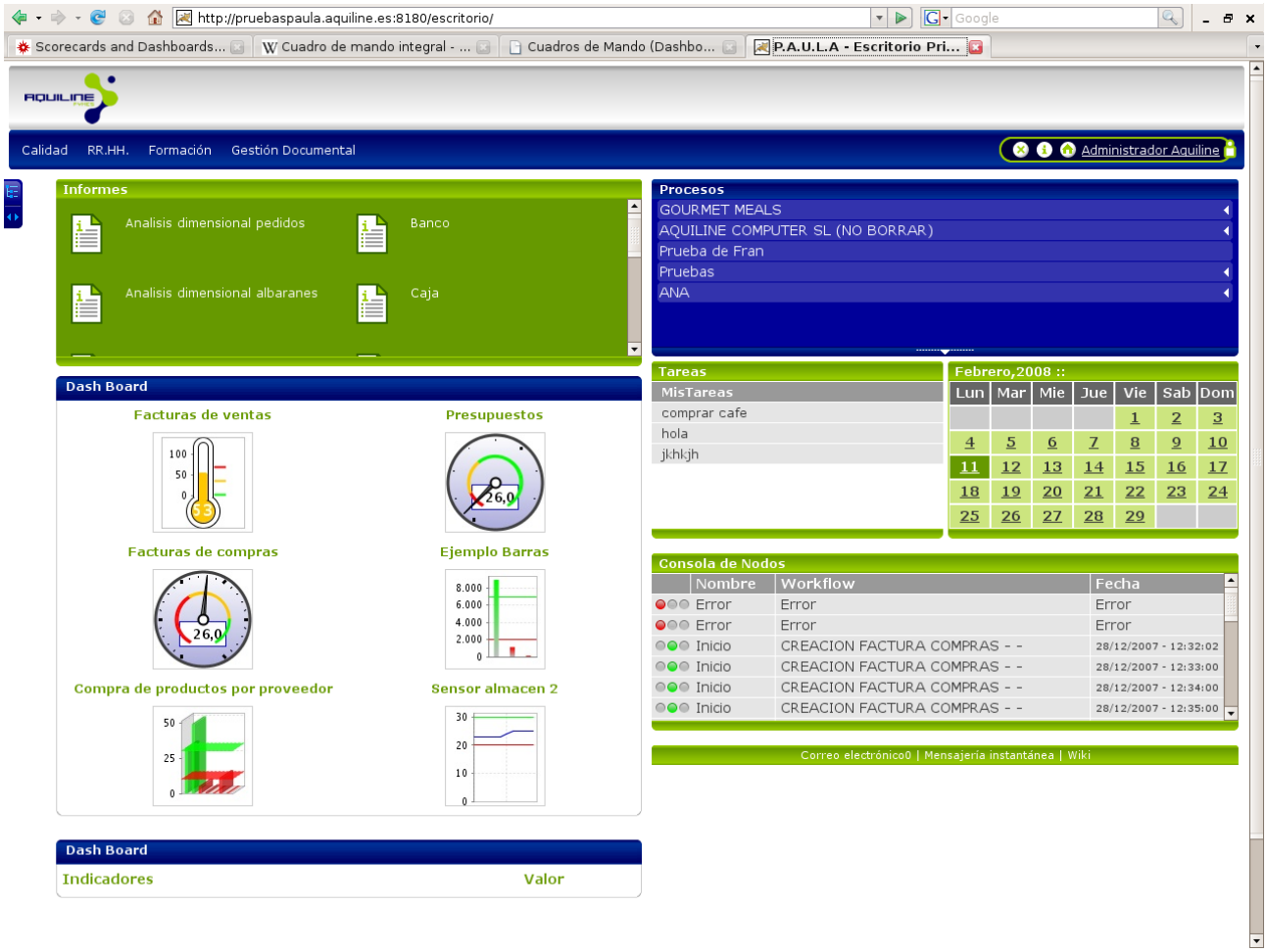


Ilustración 25: Escritorio de la plataforma

5.12 Gestión de Paula

La aplicación que gestiona la plataforma Paula es la encargada de manejar los indicadores. Pero no solo gestiona los componentes del Dashboard, sino también otras entidades de la plataforma que también son configurables. Es la responsable de administrar los usuarios y roles del sistema, el LDAP [20], los menús y las sesiones abiertas en la plataforma.

Ilustración 26: Barra diferentes secciones de gestión

El enlace "menú indicadores" nos lleva a un listado con todos los indicadores introducidos en el sistema. La lista de indicadores tiene implementada la funcionalidad de paginación, de modo que cuando llegue a 20 indicadores introducidos en el sistema, el siguiente que introduzcamos será mostrado en la siguiente página.

Los campos mostrados en esta lista son los siguientes:

- Nombre del indicador
- Descripción del indicador
- Grupo al que pertenece el indicador
- Padre, campo que en futuras mejoras podrá ser implementado como indicador padre lleva asociados una serie de indicadores hijos.
- Modulo al que pertenece el indicador. Esto es el contexto que se concatenara con la url que insertemos en el indicador.
- Activo: si el indicador es activo o no.
- Acciones que podemos realizar con los indicadores, en este caso, se pueden eliminar o

20 protocolo a nivel de aplicación que permite el acceso a un servicio de directorio ordenado y distribuido para buscar diversa información en un entorno de red

editar.

Destacar también, que en la barra superior aparecerán dos enlaces a Filtrar por etiquetas, o a crear un nuevo indicador.

Indicadores — Nuevo — Filtrar

Listado

29 registros encontrados, mostrando 1 a 20. (Pï¿½gina 1 de 2)
|< << 1 | 2 >> >|

Nombre	Creado	CreadoPor	Modificado	ModificadoPor	Modulo	Activo	Acciones
% de incidencias con proveedores	23/11/2007	1	15/01/2008	1	paula	☐	Editar Eliminar
% de respuestas enviados	21/11/2007	1	15/01/2008	1	paula	☑	Editar Eliminar
% hr formacion	19/11/2007	1	15/01/2008	1	paula	☑	Editar Eliminar
% medio de proyectos retrasados	21/11/2007	1	15/01/2008	1	paula	☑	Editar Eliminar
% Propuestas cerradas	21/11/2007	1	15/01/2008	1	paula	☑	Editar Eliminar
Clientes repetir llamadas para modelos	21/11/2007	1	15/01/2008	1	paula	☑	Editar Eliminar
Compra de productos por proveedor	21/11/2007	1	15/01/2008	1	paula	☑	Editar Eliminar
Dias aceptacion oferta y cobro	21/11/2007	1	15/01/2008	1	paula	☐	Editar Eliminar
Ejemplo Barras	13/11/2007	1	15/01/2008	1	paula	☑	Editar Eliminar
Ejemplo Barras3d	15/11/2007	1	15/01/2008	1	paula	☑	Editar Eliminar
Facturas de compras	17/10/2007	1	24/01/2008	1	paula	☑	Editar Eliminar
Facturas de ventas	17/10/2007	1	02/01/2008	1	paula	☑	Editar Eliminar
Horas de formacion por empleado	20/11/2007	1	02/01/2008	1	paula	☐	Editar Eliminar
incidencias con proveedores	23/11/2007	1	15/01/2008	1	paula	☐	Editar Eliminar
NÃ§Ã§Ã§Ã§Ã§Ã§Ã§Ãº de no conformidades	28/11/2007	1	15/01/2008	1	paula	☑	Editar Eliminar
NÃ§Ã§Ãº de dias de respuesta al cliente	21/11/2007	1	15/01/2008	1	paula	☐	Editar Eliminar
No de empleados	22/10/2007	1	15/01/2008	1	paula	☑	Editar Eliminar
No de proveedores	22/10/2007	1	15/01/2008	1	paula	☑	Editar Eliminar
Numero de clientes	20/11/2007	1	15/01/2008	1	paula	☑	Editar Eliminar

Ilustración 27: Listado indicadores

Para construir la lista de indicadores, recuperamos la variable de sesión de hibernate, que contiene todos los indicadores que están en la base de datos. Una vez recuperado, vamos cargando en una tabla cada uno de los indicadores con sus campos principales. Gracias a la aplicación de un CSS, tienen una apariencia agradable para el usuario. También creamos el acceso a crear indicador, editar o eliminar.

Todos estos accesos, y los que hemos realizado en el desarrollo, son paginas mapeadas en el web.xml de la aplicación, que al pulsar encima, no solo te llevan a la pagina web adecuada si no que también se realizan acciones a mas bajo nivel, dependiendo del contexto de la plataforma en el que estemos.

Si pulsamos en el acceso a crear un nuevo indicador, nos parecerá un formulario, para poder introducir todas las propiedades del indicador en cuestión a crear.

Así nos mostraran cada una de las propiedades del indicador en un objeto HTML, dependiendo de la necesidad de este:

- el nombre

- la descripción
- la url
- la sentencia sql
- tipo
- el orden
- el tiempo entre avisos
- la acción superior
- la acción inferior
- el rango menor
- el rango superior
- el límite inferior
- el límite superior
- la fecha inicial
- la fecha final

También tendremos un box para activar si un Indicador es activo o no, y un desplegable para indicar el modulo al que pertenece.

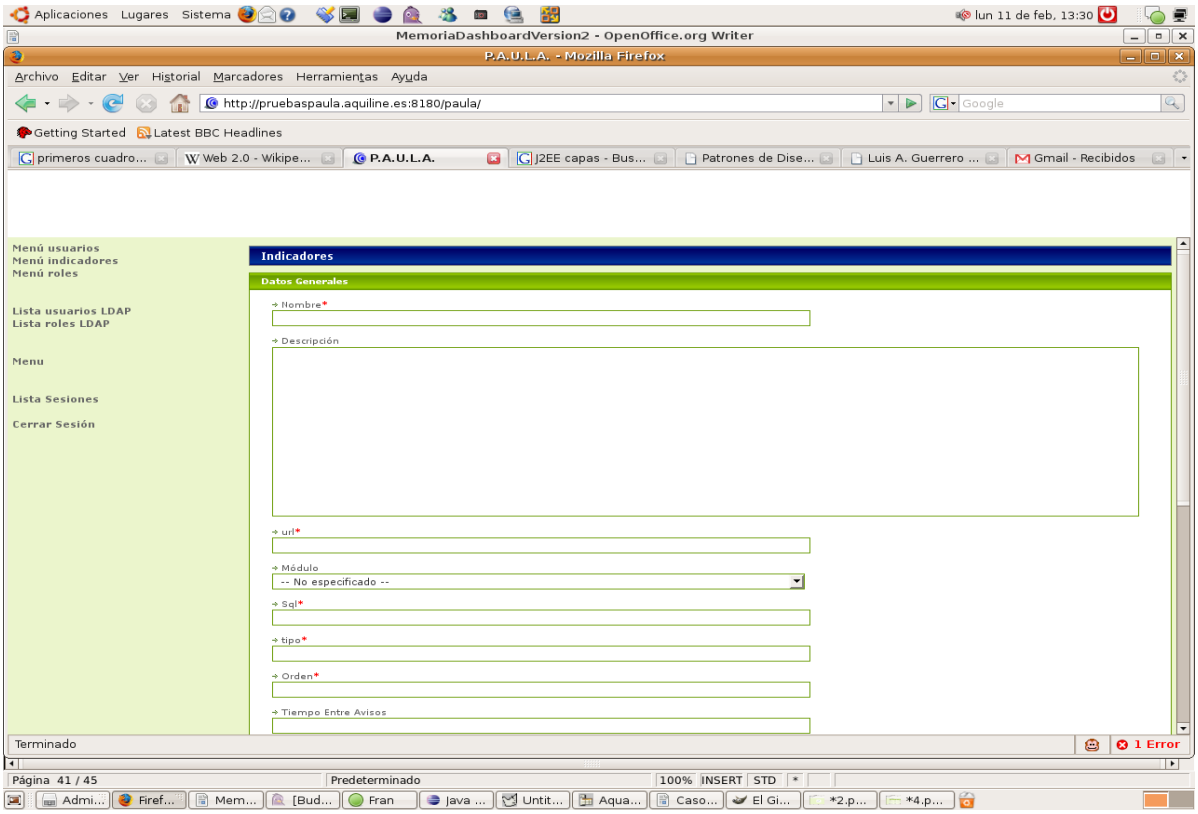


Ilustración 28: Formulario Nuevo Indicador

Una vez introducidos los datos del indicador, recuperaremos estos y crearemos un objeto indicador, con los parámetros pasados. Cabe la posibilidad de asignarle uno o varios permisos a un indicador.

Esto es, simplemente guardar la asociación rol_indicador en la tabla que se encarga de mantener estas asociaciones. Finalmente guardaremos el objeto en la base de datos, todo ello por medio de hibernate.

En el caso de que pulsemos en filtrar nos aparecerá un textbox para introducir la etiqueta por la cual queremos filtrar y una lista de etiquetas creadas ya en el sistema, que son las que están dadas de alta en el sistema.

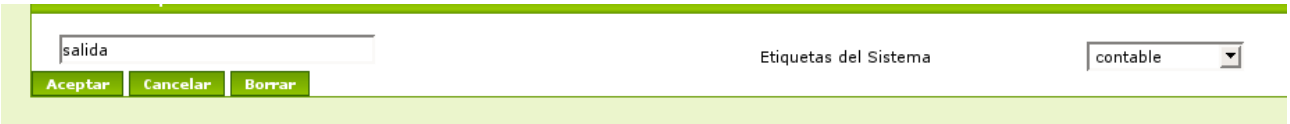


Ilustración 29: Filtro Etiquetas

Por último destacar que también el menú de roles, podemos asignar directamente indicadores a los diferentes roles. Esto esta pensando, para que en el momento que en el sistema se cree, un nuevo rol, podamos asignarle los indicadores que creamos importante asociarle. Para ello, haremos el proceso inverso realizado antes, es decir, en vez de crear un indicador y asociarle permisos, crearemos un rol, y le asociaremos indicadores. Realmente la asociación creada es la misma, con la diferencia , que ahora creamos una instancia de nuevo rol y le asignaremos después los indicadores. LA lista de indicadores asociadas dentro del rol, se mostrarán como una lista que mostramos a continuación:

Ilustración 30: Indicadores asociados a un rol

5.13 Escritorio

Debido a que Paula es una plataforma unificada que integra una serie de aplicaciones (Open bravo, Qualis, Joomla, Frida, etc), tiene la necesitada de desarrollar una aplicación tipo escritorio que pueda centralizar y dar acceso a todas ellas. Surgida esta necesidad, se crea el Escritorio de Paula, desde el cual vamos a poder interactuar con todas las aplicaciones, desde el mismo lugar.

Para poder realizar el escritorio hubo que integrar todos los diferentes programas, para no solo poder acceder a ellos desde el mismo lugar, si no para que pudieran comunicarse entre ellos.

Podemos dividir el escritorio en 5 zonas:

- **Consola de nodos**: Muestra los diferentes flujos de trabajo lanzados, donde el campo *nombre* es el nodo del flujo por el que se encuentra la ejecución de este. El campo *workflow* es nombre del flujo de trabajo. Por último el campo *fecha* es el momento en el tiempo cuando se lanzo el flujo.
- **Tareas:**
- **Procesos:**
- **Informes:**
- **Dashboard**

También destacar que tenemos un acceso al cliente de correo Kolab, a la mensajería instantánea de la empresa, y al wiki de la entidad. Todos estos accesos también están integrados en la plataforma.

Es muy importante destacar el hecho de que el escritorio se refresca daca 2 minutos, por lo que las actualizaciones sobre el escritorio son casi inmediatas, dando la posibilidad de refrescar el Dashboard, sin tener que refrescar toda la pagina.

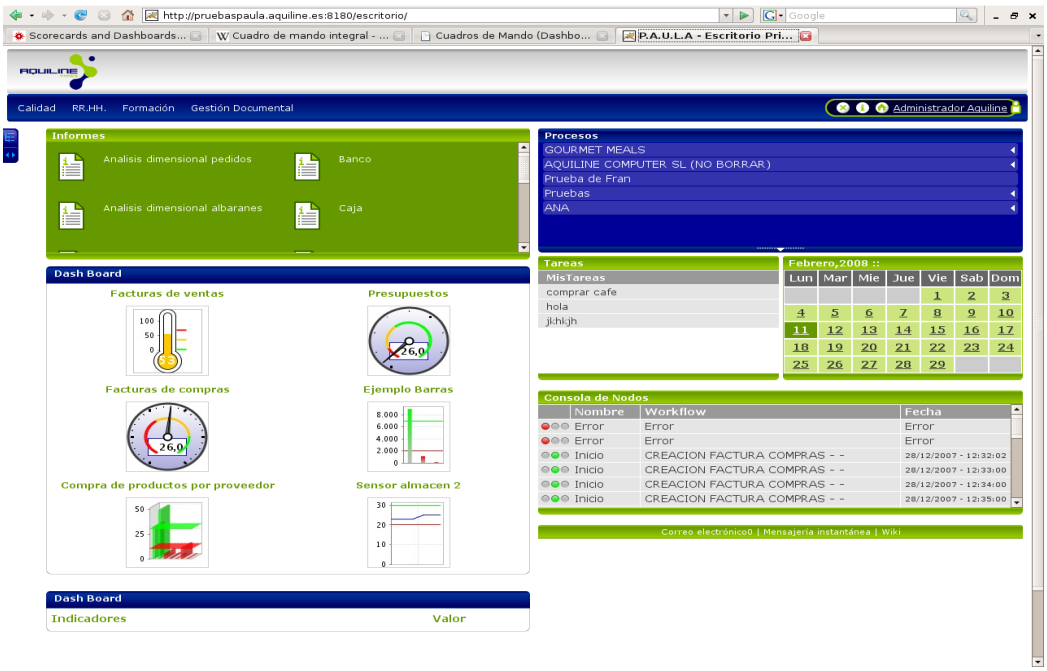


Ilustración 31: Escritorio

El escritorio, al ser el núcleo o eje centrar de la plataforma, fue el lugar elegido para colocar el cuadro de mando. Realmente, el Dashboard, no es una aplicación independiente que se integro en el escritorio, si no una mejora para el escritorio.

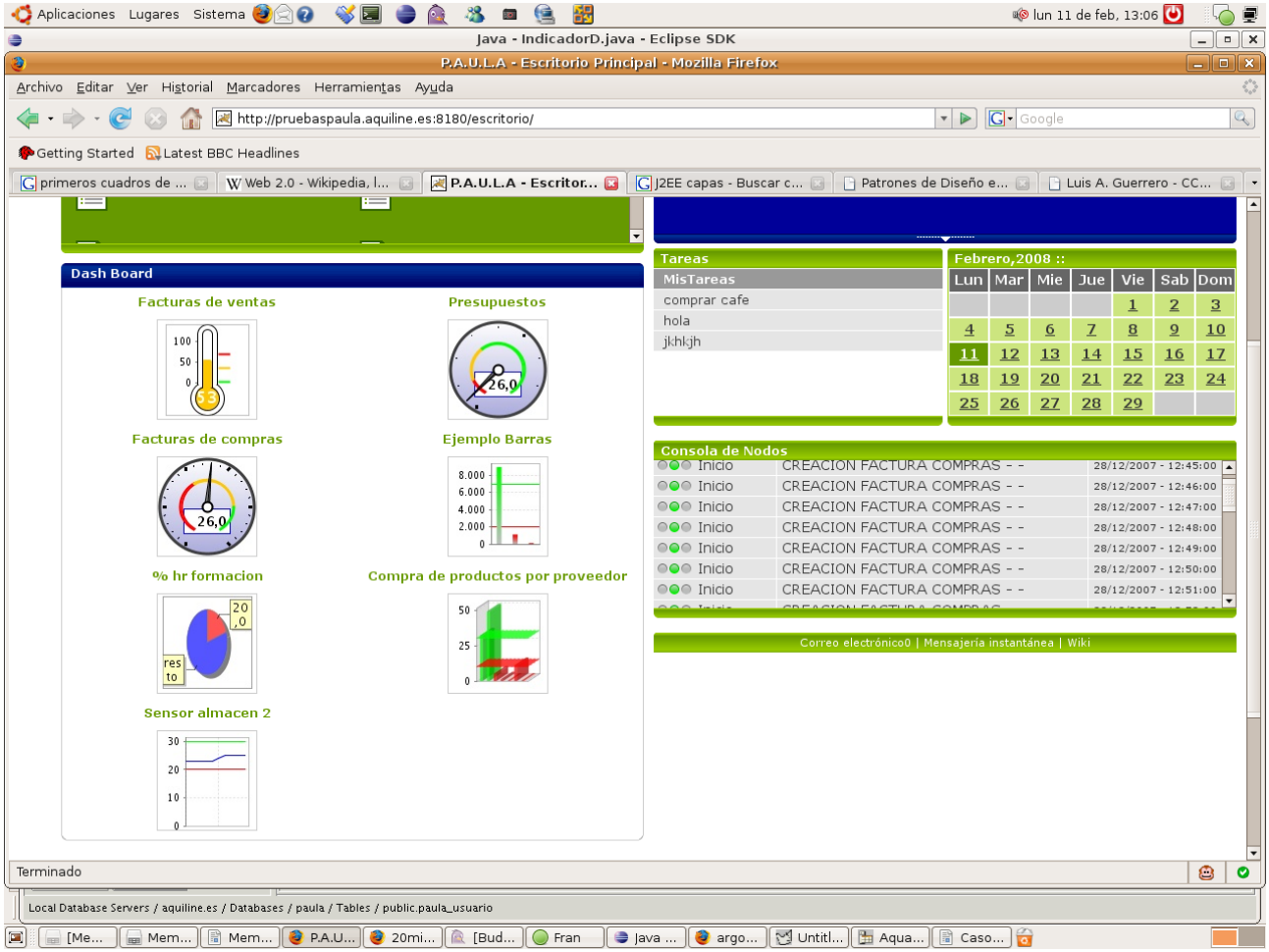


Ilustración 32: Escritorio

El contenedor que mantiene los indicadores, es un JSP, que aplicándole un CSS apropiado, presenta al usuario todos los indicadores asociados a su rol.

Uno de los claros ejemplos en que el escritorio interactúa con otra aplicación, es la operación de introducir una etiqueta a un indicador, ya que esto realmente esta desarrollado en la parte de gestión de la plataforma. El usuario indica que quiere introducir una etiqueta pulsando sobre el hiper enlace tag , el cual abre una ventana para introducir la etiqueta asociada a ese indicador. Una vez introducida se almacena el la BBDD.

Facturas de ventas [tag]

Ilustración 33: enlace insertar Etiqueta desde el escritorio

Ilustración 34: Ventana insertar etiqueta en el escritorio

Esta ventana, es simplemente una llamada al servlet que mapeado en el web.xml, redirecciona el contenido de la ventana al jsp, de la gestión de Paula. Este recoge los datos para introducirlos en la base de datos, del mismo modo que se hacia en la gestión, ya que realmente se esta haciendo desde allí.

Otra funcionalidad importante, que esta relacionada con el escritorio y el Dashboard, es la de mostrar al usuario los flujos de trabajos lanzados por un indicador. Cuando uno de los límites superiores o inferiores de un indicador es rebasado, el flujo de trabajo asociado se dispara. Entonces llega un evento sobre la consola de nodos, mostrando el flujo de trabajo que se acaba de lanzar por medio del indicador.

Consola de Nodos

Nombre	Workflow	Fecha
Error	Error	Error
Error	Error	Error
Inicio	CREACION FACTURA COMPRAS	28/12/2007 - 12:32:02
Inicio	CREACION FACTURA COMPRAS	28/12/2007 - 12:33:00

Ilustración 35: Consola de nodos

Como ya explicamos anteriormente, todo este proceso es responsabilidad de la herramienta Jcrontab, y la clase que realiza la tarea de comprobar los valores de los indicadores.

5.14 Conclusiones

Tras el desarrollo de este proyecto, se ha conseguido un sistema automático de información y gestión de un cuadro de mandos, con funcionalidad completa.

Este estudio, demuestra la forma de desarrollar una aplicación vía web usando diferentes tecnologías e intentando llegar a una aproximación a la Web 2.0. Se han usado la mayoría de tecnologías y técnicas que deben utilizarse para crear este tipo de sistemas. Además se han aprovechado otras herramientas que han hecho posible toda la resolución.

La aplicación de las etiquetas sobre los indicadores por parte de los usuarios, añade un carácter semántico agregado al indicador y permiten a los usuarios interactuar con la aplicación.

Otro objetivo que hemos conseguido, gracias a la asociación de los indicadores con cada rol de la plataforma, es que cada usuario reciba solo los indicadores que le interesa conocer del sistema de información.

También ha sido importante el ejercicio de recopilación de información sobre los cuadros de mando, para entender, y así poder implementar, un cuadro de mandos.

A lo largo del proyecto se han ido añadiendo mejoras, que han motivado una mayor investigación, sobre todo en la aplicación de la tecnología AJAX. Fue difícil entender su funcionamiento y su aplicación a los servlets e imágenes dinámicas. Inicialmente se pensó en refrescar únicamente un indicador, en vez de todo el cuadro de mandos, como se hizo finalmente. Esta decisión final, se tomo al no poder llegar a conseguir un buen funcionamiento de lo que se ideó primeramente.

Finalmente se decidido que sería importante, sobre todo desde el punto de vista de la empresa, que se genere un evento cuando se rebasen unos límites marcados.

El Dashboard está listo para ser utilizado junto al conjunto de aplicaciones empresariales ERP de la Plataforma PAULA de Aquiline Computer SL.

Bibliografía y enlaces

1. *Http://ajaxwidgets.com/AllControlsSamples/DynamicImage.aspx*
2. Http://bi.abast.es/dashboards.shtml
3. Http://cuadrodemando.unizar.es/inicio2.html
4. *Http://docs.jquery.com/Ajax/load#urldatacallback*
5. *Http://dolf.trieschnigg.nl/jfreechart/*
6. *Http://en.wikipedia.org/*
7. *http://es.wikipedia.org/*
8. Http://es.wikipedia.org/wiki/Cuadro_de_mando
9. Http://es.wikipedia.org/wiki/BSC
10. Http://java.sun.com/javaee/5/docs/tutorial/doc/
11. Http://www.jcrontab.org/index.shtml
12. *Http://mojo.codehaus.org/tomcat-maven-plugin/*
13. *Http://p2p.wrox.com/topic.asp?TOPIC_ID=50276*
14. *Http://spanish.osstrans.net/software/svgchart.html*
15. *Http://sourceforge.net/projects/jfreechart*
16. *Http://www.**ajax**.nl/*
17. Http://www.arearh.com/rrhh/balanced_scorecard.htm
18. *Http://www.cristalab.com/tutoriales/162/tutorial-de-ajax*
19. *Http://www.google.es*.
20. Http://www.microstrategy.es/Solutions/5Styles/scorecards_dashboards.asp
21. *Http://www.slideshare.net/vicente.ordonez/ajax-atlas-219010/*

Apéndice

Documentos Complementarios

A1. Creación de las tablas en la base de datos

Tabla de Indicadores:

```
CREATE TABLE public.paula_indicador (
    indicador_id        numeric(11,0) NOT NULL,
    nombre              varchar(80) NOT NULL,
    descripcion         varchar(255) NOT NULL,
    modulo              varchar(80) NOT NULL,
    url             varchar(170) NOT NULL,
    sql             varchar(2000) NOT NULL,
    tipo            char(1) NOT NULL DEFAULT 'G'::character varying,
    rangomen            float8 NOT NULL,
    rangomay            float8 NOT NULL,
    cortemin            float8 NOT NULL,
    cortemax            float8 NOT NULL,
    fechaini            timestamp NOT NULL DEFAULT now(),
    fechafin            timestamp NOT NULL DEFAULT now(),
    activo              char(1) NOT NULL DEFAULT 'Y'::character varying,
    modificado          timestamp NOT NULL DEFAULT now(),
    creado              timestamp NOT NULL DEFAULT now(),
    modificadopor       numeric(11,0) NOT NULL,
    creadopor           numeric(11,0) NOT NULL,
    orden               numeric NULL,
    tiempoentreavisos   numeric(15,5) NULL,
    accionsup           numeric(15,5) NULL,
    accioninf           numeric(15,5) NULL,
    fechaaccionsup      timestamp NULL,
    fechaaccioninf      timestamp NULL,
    PRIMARY KEY(indicador_id)
```

```
)

ALTER TABLE public.paula_indicador
   ADD CONSTRAINT indicador_activo_values
         CHECK (((activo)::text = 'Y'::text) OR ((activo)::text = 'N'::text))
```

Tabla de IndicadoresEtiquetas:

```
CREATE TABLE public.paula_indicadores_etiquetas (
   indicador_etiqueta_id       numeric NOT NULL DEFAULT
nextval(('"paula_indicadores_etiquetas_sec"'::text)::regclass),
   indicador_id          numeric(11,0) NOT NULL,
   etiqueta              varchar(25) NOT NULL,
   activo                char(1) NOT NULL DEFAULT 'Y'::character varying,
   modificado            timestamp NOT NULL DEFAULT now(),
   creado                timestamp NOT NULL DEFAULT now(),
   modificadopor               numeric(11,0) NOT NULL,
   creadopor             numeric(11,0) NOT NULL
   )
```

Tabla de IndicadoresRol:

```
CREATE TABLE public.paula_indicadores_rol (
   indicador_id          numeric(11,0) NOT NULL,
   rol_id         numeric(11,0) NOT NULL,
   activo                char(1) NOT NULL DEFAULT 'Y'::character varying,
   modificado            timestamp NOT NULL DEFAULT now(),
   creado                timestamp NOT NULL DEFAULT now(),
   modificadopor         numeric(11,0) NOT NULL,
   creadopor             numeric(11,0) NOT NULL,
   indicador_rol_id      numeric(11,0) NOT NULL DEFAULT
nextval(('"paula_indicadores_rol_sec"'::text)::regclass),
   PRIMARY KEY(indicador_id,rol_id)
```

```
)

ALTER TABLE public.paula_indicadores_rol
   ADD CONSTRAINT indicador_rol_id_unique
        UNIQUE (indicador_rol_id)

ALTER TABLE public.paula_indicadores_rol
   ADD CONSTRAINT indicador_rol_activo_values
        CHECK (((activo)::text = 'Y'::text) OR ((activo)::text = 'N'::text))

ALTER TABLE public.paula_indicadores_rol
   ADD CONSTRAINT indicador_rol_fk
        FOREIGN KEY(rol_id)
        REFERENCES public.paula_rol(rol_id)
```

Tabla de Roles:

```
CREATE TABLE public.paula_rol (
   rol_id        numeric(11,0) NOT NULL,
   nombre        varchar(80) NOT NULL,
   descripcion           varchar(256) NULL,
   mail          varchar(100) NULL,
   activo        char(1) NOT NULL DEFAULT 'Y'::character varying,
   modificado            timestamp NOT NULL DEFAULT now(),
   creado        timestamp NOT NULL DEFAULT now(),
   modificadopor         numeric(11,0) NOT NULL,
   creadopor             numeric(11,0) NOT NULL,
   PRIMARY KEY(rol_id)
)

ALTER TABLE public.paula_rol
   ADD CONSTRAINT rol_activo_values
        CHECK (((activo)::text = 'Y'::text) OR ((activo)::text = 'N'::text))
```

Tabla de usuarios:

```
CREATE TABLE public.paula_usuario (
   usuario_id            numeric(11,0) NOT NULL,
   login         varchar(80) NOT NULL,
   password              varchar(40) NOT NULL,
   nombre                varchar(80) NOT NULL,
   apellidos             varchar(80) NOT NULL,
   nombre_completo  varchar(170) NOT NULL,
   mail          varchar(100) NOT NULL,
   activo        char(1) NOT NULL DEFAULT 'Y'::character varying,
   modificado            timestamp NOT NULL DEFAULT now(),
   creado                timestamp NOT NULL DEFAULT now(),
   modificadopor         numeric(11,0) NOT NULL,
   creadopor             numeric(11,0) NOT NULL,
   defrol        numeric(11,0) NULL,
   defentidad            numeric(11,0) NULL,
   deforganizacion       numeric(11,0) NULL,
   defalmacen            numeric(11,0) NULL,
   PRIMARY KEY(usuario_id)
)
ALTER TABLE public.paula_usuario
   ADD CONSTRAINT usuario_nombre_completo_unique
        UNIQUE (nombre_completo)
GO
ALTER TABLE public.paula_usuario
   ADD CONSTRAINT usuario_mail_unique
        UNIQUE (mail)
GO
ALTER TABLE public.paula_usuario
```

```
   ADD CONSTRAINT usuario_login_unique
         UNIQUE (login)
GO
ALTER TABLE public.paula_usuario
   ADD CONSTRAINT usuario_activo_values
         CHECK (((activo)::text = 'Y'::text) OR ((activo)::text = 'N'::text))
GO
```

Tablar rol_usuario

```
CREATE TABLE public.paula_rol_usuario (
   rol_id         numeric(11,0) NOT NULL,
   usuario_id   numeric(11,0) NOT NULL
   )
GO
ALTER TABLE public.paula_rol_usuario
   ADD CONSTRAINT rol_usuario_usuario_fk
         FOREIGN KEY(usuario_id)
         REFERENCES public.paula_usuario(usuario_id)
GO
ALTER TABLE public.paula_rol_usuario
   ADD CONSTRAINT rol_usuario_rol_fk
         FOREIGN KEY(rol_id)
         REFERENCES public.paula_rol(rol_id)
GO
```

A2. Ejemplos de varios indicadores financieros reales

Los indicadores que más se suelen utilizar a nivel empresarial para el control de calidad son los mostrados a continuación . Van por colores en función de si se trata de indicadores de formación, prestación del servicio, valoración de satisfacción de cliente, compras o no conformidades (por este orden):

1. % de horas de formación por empleado
2. % de horas destinadas a formación en el despacho
3. Relación de aceptación de propuestas de servicio (propuestas aceptadas/propuestas emitidas en total)
4. % tipo de clientes sobre el total de clientes
5. % de clientes que hay que repetir llamadas para realizar modelos
6. Nº de días de respuesta al cliente (desde toma de datos hasta entrega de oferta).
7. Nº de días desde aceptación de oferta y realización del primer cobro.
8. % medio de desfase en proyectos (horas)
9. Valoración global de satisfacción de clientes
10. % de respuestas de cuestionarios de satisfacción enviados
11. % de compra de productos por proveedor/total de compras
12. % de incidencias con proveedores que han afectado a producción.
13. % de pedidos con cantidades superiores a X cantidad/total
14. % de pedidos con un plazo de entrega inferior a X días/total de pedidos.
15. Tiempo de respuesta para cerrar una no conformidad.
16. nº de no conformidades resueltas antes de X horas
17. % de acciones correctoras eficaces / total lanzadas
18. % de acciones correctoras ejectuadas con retraso / total lanzadas
19. % cantidad de productos no conformes / cantidad total fabricada
20. % de sugerencias de mejora
21. % de reclamaciones de calidad fundadas / total reclamaciones
22. Tiempo de lanzamiento de nuevos productos
23. Índices de defectos de los productos recibidos