

Índice general

1. Introducción	3
1.1. Introducción	3
1.2. Planificación del proyecto	4
1.3. Estructura del documento	5
2. Ingeniería de Requisitos	9
2.1. Introducción	9
2.2. Captura de requisitos	9
2.3. Modelado de requisitos	10
3. Análisis y documentación	13
3.1. GO.JS	13
3.1.1. ¿Qué es GO.JS?	13
3.1.2. ¿Porqué GO.JS?	13
3.1.3. Características	14
3.1.4. Explicación	14
3.2. Vaadin	14
3.2.1. ¿Qué es GO.JS?	14
3.2.2. Características	15
3.2.3. Aplicación	15
4. Diseño Arquitectónico	17
4.1. Process Component	17
4.2. Luca Process	17
5. Implementación	19
6. Pruebas	21
7. Conclusión	23

Índice de figuras

1.1. Metodología En Cascada	4
3.1. GO.JS Logo	14
3.2. Vaadin Logo	15
3.3. Esquema Cliente-Servidor	15

Índice de cuadros

2.1. Conjunto de Requisitos del Proyecto	10
2.2. Conjunto de Requisitos del Componente Gráfico	10

Agradecimientos

Me gustaría dar agradecimientos a mi familia y facultad, ya que sin ellos esto no habría sido posible nada de estos.

Es importante agradecer también a CIC Consulting Informático por permitirme la oportunidad de realizar el desarrollo del proyecto en su empresa, sin olvidarme de mis compañeros de LUCA, que han sido un gran apoyo durante el mismo..

Para finalizar, me gustaría agradecer a mi mentor Pablo, por guiarme durante el desarrollo del proyecto con eficacia y ayudarme a afrontar este trabajo de fin de grado.

Resumen

Las empresas actuales utilizan ya no un único sistema de información que de soporte a sus procesos de trabajo, sino un ecosistema de sistemas información que dan soporte a diferentes procesos de negocio ejecutados dentro de dicha organización. Como consecuencia de esta nueva situación, cuando un usuario quiere obtener una información concreta cuyos datos residen en varios de estos sistemas, necesita acceder a cada uno de estos sistemas, extraer de cada sistema la información que precisa, filtrarla y unificarla para finalmente obtener los datos requeridos.

Por ejemplo, una tienda de electrodomésticos podría tener sistemas informáticos diferentes para el departamento de atención al cliente, para el departamento técnico de postventa y para el departamento de compras y adquisiciones. Por tanto, para conocer el estado actual de una reparación, podríamos necesitar:

- Acceder al primer sistema para obtener el identificador de la incidencia y en qué fase de su gestión se encuentra.
- Comprobado que la incidencia está actualmente en reparación, recuperaríamos otro sistema el estado detallado de la reparación, comprobando que está a la espera de una pieza.
- Finalmente accederíamos al sistema de compra y adquisiciones para comprobar cuando está prevista la entrega de dicha pieza. Los sistemas de almacenamiento de la información puede ser diversos, incluyendo desde un servicio web, una base de datos relacional, un repositorio de ficheros accesible vía FTP o una base de datos NoSQL.

El objetivo de este proyecto es facilitar dicho proceso de composición al usuario mediante el desarrollo de un mecanismo gráfico para la especificación de estos procesos de composición de consultas.

Palabras clave:

Preface

Keywords:

Capítulo 1

Introducción

Este capítulo del documento se propone a describir, en leves pinceladas, los objetivos principales que el proyecto debe de completar.

1.1. Introducción

En los últimos años, el volumen de datos que una empresa o entidad necesita y/o es capaz de gestionar o manipular, ha aumentado de forma vertiginosa. Además del volumen de datos a manipular, nos damos cuenta de que para acceder a lo diversos datos es necesario muchas veces establecer comunicaciones con los diversos recursos existentes para alcanzar el objetivo.

Con el objeto de facilitar este proceso de recuperación de información almacenada en sistemas y fuentes de datos heterogéneas, dentro de la empresa CIC, se está desarrollando una aplicación denominada LUCA. Para facilitar este proceso de recuperación de información, LUCA proporciona un lenguaje común para todas las fuentes de datos a unificar, permitiendo al usuario abstraerse de los detalles de cada fuente.

Actualmente LUCA proporciona mecanismos o abstracciones para permitir al usuario recuperar de manera uniforme información de diferentes fuentes de datos. No obstante, LUCA actualmente sólo es capaz recuperar información de una única fuente de datos a la vez. Por tanto, cuando es necesario combinar información procedente de distintas fuentes, el propio usuario es el que debe realizar dicho proceso de composición, ejecutando cada consulta a mano, y utilizando las salidas de cada una de ellas como las entradas de las siguientes.

El objetivo de este proyecto es facilitar dicho proceso de composición al usuario mediante el desarrollo de un mecanismo gráfico para la especificación de estos procesos de composición de consultas.

1.2. Planificación del proyecto

En este apartado se describe la metodología de desarrollo que ha sido utilizada para la cumplimentación del proyecto.

En el proyecto se optó por utilizar una metodología en cascada, en la que de forma iterativa se va puliendo el producto hasta completar los requisitos y objetivos establecidos.

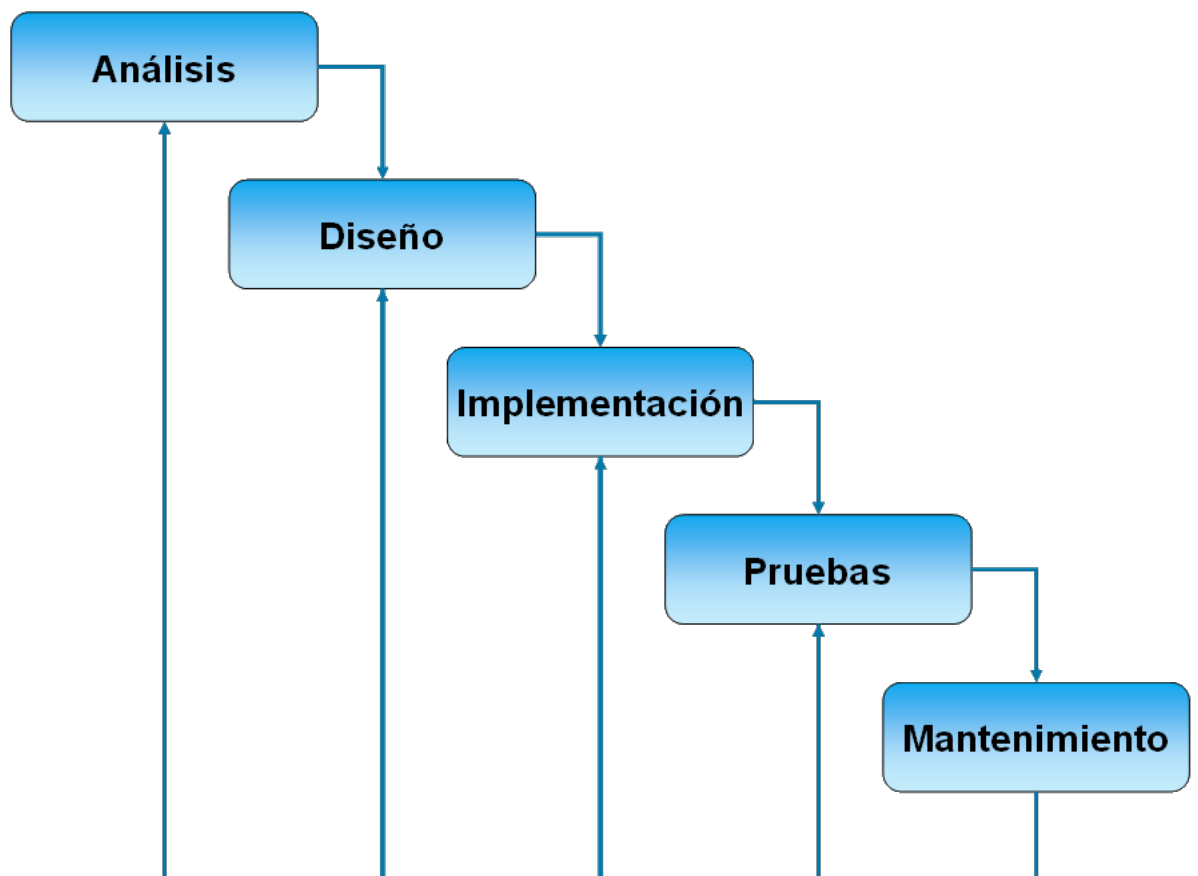


Figura 1.1: Metodología En Cascada

Acorde a la imagen mostrada previamente, la primera etapa de la metodología que se llevo a cabo para la cumplimentación del proyecto fue el análisis de los requisitos y objetivos del proyecto. El objetivo de esta etapa es describir de forma detallada el conjunto del requisitos y funcionalidades necesarias para la satisfacción del stakeholder y la completitud del sistema.

Una vez en posesión de la especificación de requisitos, el siguiente paso fue definir el diseño arquitectónico del proyecto en bloques independientes que pudiesen ser elaborados y probados de forma aislada. Otro requisito importante que se tuvo que tener en cuenta fue, que el producto debe de estar diseñado para poder hacer frente a futuros incrementos.

Utilizando el diseño arquitectónico ya definido, se focalizó el esfuerzo en el diseño de la aplicación. En esta se tuvo en cuenta que se pretenden realizar dos elaboraciones o componentes software:

- En una primera instancia se desarrollara un componente genérico cuya funcionalidad es de índole gráfica, por lo que no precisa de una desarrollo en capas común, ya que consta unicamente de un modelo de datos y una implementación web a través del Framework GO.JS, así la integración con Vaadin [1].
- En una segunda instancia se desarrollará una aplicación incremento del producto actual ya elaborado llamado LUCA. El cuál sigue un patrón MVP [2], y esta elaborado en Spring [4].

Para aplicación desarrollado con Spring, se creó una base de datos relacional asociada, la cuál servirá de fuente de persistencia de los datos almacenados.

Tras la especificación de la base de datos comenzó un proceso iterativo de implementación, verificación y pruebas en el que en función de los diversos requisitos se fue completando el proyecto.

1.3. Estructura del documento

Esta sección se propone describir la ingeniería de requisitos llevada a cabo, así como, profundizar en el diseño arquitectónico. También como introducción

inicial, se relatará de forma breve y concisa la librería javascript GO.JS y el framework Vaadin.

En primera instancia explicará el proceso de captura y especificación de requisitos llevado a cabo, así como, la toma de decisiones seguidas. También se centrará en detallar y explicar las decisiones para la especificación de la base de datos, que deberán de estar de acuerdo con las implementación actuales de las bases de datos de LUCA, y con la especificacion basada en Spring. Por último se explicará el proceso de desarrollo de las capas de repositorio, servicios y presentación.

Capítulo 2

Ingeniería de Requisitos

Este capítulo describe el proceso de captura de requisitos, así como, el modelado de los requisitos tanto funcionales como no funcionales.

Contents

2.1. Introducción	9
2.2. Captura de requisitos	9
2.3. Modelado de requisitos	10

2.1. Introducción

El capítulo actual se propone explicar de forma detallada la fase de ingeniería de requisitos, en la que se recoge toda la información necesaria relativa a los requisitos para poder presentar al stakeholder los objetivos o requerimientos que debe de tener la aplicación para poder considerarse como completada.

Por tanto, teniendo en cuenta lo citado anteriormente, este capítulo tratará primero el proceso de captura de requisitos llevado a cabo, para posteriormente, mostrar y detallar el modelado de requisitos subyacente de la etapa anterior.

2.2. Captura de requisitos

En la presente sección se describe el proceso de identificación de requisitos que nuestra aplicación debe de completar o satisfacer.

Dado que la aplicación a construir es un incremento de un producto ya existente, no hace falta identificar la fuente de los requisitos ya que es el propio gerente o impulsor de dicho incremento el stakeholder, junto con el jefe del proyecto, los objetivos conjuntos de la captura de información y de los propios requisitos.

A raíz de una demostración del producto actual LUCA, y de la reunión para la captura de requisitos con el gerente y de la empresa CIC, así como del jefe de proyecto bajo el que estuvo supervisado el proyecto actual, se muestra el resultado del proceso de la captura de requisitos llevada a cabo.

Cabe explicar también, que este proyecto se funda en la implementación o desarrollo de dos componentes. Uno representa una herramienta focalizada en el ámbito gráfico para la creación, borrado y actualización, en términos generales, manipulación gráfica de los elementos constituyentes del proyecto. El otro representa un proyecto incremento del actual producto LUCA, centrado en la gestión de los procesos y elementos hijos.

Referencia	Requisito
REF-01	Los usuarios podrán consultar los datos referentes a un proceso.
REF-02	Los usuarios podrán actualizar o modificar los datos de los procesos.
REF-03	Los usuarios podrán persistir los nuevos procesos creados.
REF-04	Los usuarios podrán crear, a partir de las consultas almacenadas en el sistema

Cuadro 2.1: Conjunto de Requisitos del Proyecto

Referencia	Requisito
------------	-----------

Cuadro 2.2: Conjunto de Requisitos del Componente Gráfico

2.3. Modelado de requisitos

Capítulo 3

Análisis y documentación

Este capítulo describe...

Contents

3.1. GO.JS	13
3.1.1. ¿Qué es GO.JS?	13
3.1.2. ¿Porqué GO.JS?	13
3.1.3. Características	14
3.1.4. Explicación	14
3.2. Vaadin	14
3.2.1. ¿Qué es GO.JS?	14
3.2.2. Características	15
3.2.3. Aplicación	15

3.1. GO.JS

3.1.1. ¿Qué es GO.JS?

Go.JS [3] es una biblioteca de JavaScript con múltiples funciones para implementar diagramas interactivos personalizados y visualizaciones complejas en navegadores y plataformas web modernos.

3.1.2. ¿Porqué GO.JS?

GoJS facilita la construcción de diagramas de JavaScript de nodos, enlaces y grupos complejos con plantillas y diseños personalizables.



Figura 3.1: GO.JS Logo

3.1.3. Características

GoJS ofrece muchas características avanzadas para la interactividad del usuario, tales como arrastrar y soltar, copiar y pegar, edición de texto en el lugar, información sobre herramientas, menús contextuales, diseños automáticos, plantillas, vinculación y modelos de datos, administración de estado transaccional y deshacer, paletas , vistas generales, controladores de eventos, comandos y un sistema de herramientas extensible para operaciones personalizadas.

3.1.4. Explicación

GoJS ofrece muchas características avanzadas para la interactividad del usuario, tales como arrastrar y soltar, copiar y pegar, edición de texto en el lugar, información sobre herramientas, menús contextuales, diseños automáticos, plantillas, vinculación y modelos de datos, administración de estado transaccional y deshacer, paletas , vistas generales, controladores de eventos, comandos y un sistema de herramientas extensible para operaciones personalizadas.

3.2. Vaadin

3.2.1. ¿Qué es GO.JS?

Vaadin [1] es un framework de desarrollo de SPA que permite escribir el código de dichas aplicaciones en Java o en cualquier otro lenguaje soportado por la JVM 1.6+. Esto permite la programación de la interfaz gráfica en lenguajes como Java 8, Scala o Groovy, por ejemplo.



Figura 3.2: Vaadin Logo

3.2.2. Características

Uno de las características diferenciadores de Vaadin es que, contrario a las librerías y frameworks de JavaScript típicas, presenta una arquitectura centrada en el servidor, lo que implica que la mayoría de la lógica es ejecutada en los servidores remotos. Del lado del cliente, Vaadin está construido encima de Google Web Toolkit, con el que puede extenderse.

3.2.3. Aplicación

En este proyecto, Vaadin se encargara de realizar la comunicación entre el cliente y el servidor. De esta forma, será capaz de enviar y recibir datos, eventos y peticiones entre el componente Javascript (cliente) y el servidor.

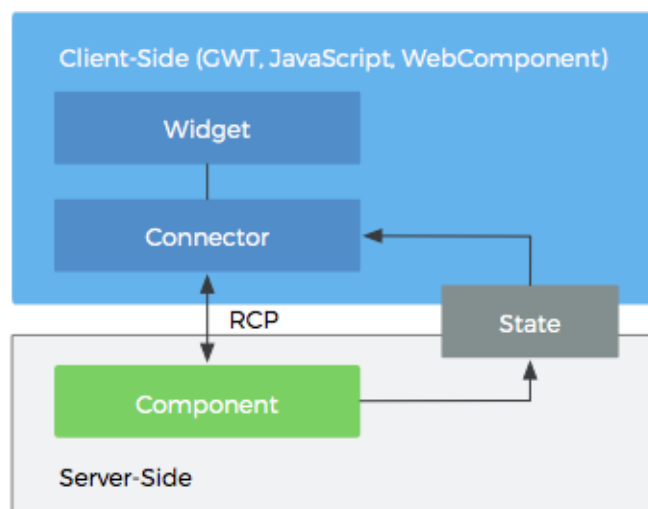


Figura 3.3: Esquema Cliente-Servidor

Capítulo 4

Diseño Arquitectónico

El diseño arquitectónico va a describir tanto la estructura arquitectónica del componente de procesos implementado con GO.js, tanto su integración en el producto LUCA, el cual, requiere de un diseño paralelo que implementar.

Contents

4.1. Process Component	17
4.2. Luca Process	17

Para poder diferenciar estos dos diseños, el componente que implementa GO.JS se nombrará como 'Process Component' y el diseño del producto LUCA que integra el Process Component se llamara 'LUCA Process'.

4.1. Process Component

4.2. Luca Process

Capítulo 5

Implementación

Este capítulo describe...

Capítulo 6

Pruebas

Este capítulo describe...

Capítulo 7

Conclusión

Este capítulo describe...

Bibliografía

- [1] ARI HANDLER GAMBOA. www.adictosaltrabajo.com/tutoriales/introduccion-a-vaadin/.
- [2] EDUARDO BARRIO. mhp-net.es/spring-vaadin-hibernate-tutorial-8-el-patron-mvp/.
- [3] NORTHWOODS SOFTWARE. www.gojs.net/latest/index.html.
- [4] PIVOTAL SOFTWARE. spring.io.