**Sistema Quick Order**

**Documento de Arquitectura**

**Versión 1.0**

**Revisiones y aprobaciones**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Versión** | **Fecha Revisión** | **Realizado Por** | **Estado** | **Sección Cambiada** | **Descripción Cambio** |
| 1.0 | 02-04-2013 | LUIS ANDRES OLARTE ZABALA | CREACION |  | Versión inicial. |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

1. **INTRODUCCION**

Este documento ofrece al usuario una visión global de la arquitectura del sistema llamado “Quick Order” el cual se espera sea un medio de comunicación entre todos los involucrados y el área de desarrollo.

* 1. **PROPOSITO**

Este documento ofrece una visión general de la estructura del sistema, las partes que lo componen y las funciones de estas. Adicionalmente se brinda una descripción general de las definiciones a las cuales se llego para la construcción del sistema.

* 1. **ACRÓNIMOS, ABREVIATURAS Y TÉRMINOS**

|  |  |
| --- | --- |
| Palabra | Definición |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

* 1. **VISIÓN GENERAL DEL DOCUMENTO**

En este apartado se listan las vistas que se utilizaran para representar la arquitectura del sistema de Quick Order

* Vista de Casos de uso

En esta vista se enumeran los casos de uso especificados durante la etapa de elaboración.

En la sesión de la vista de casos de uso se listan los casos de uso identificados como de mayor impacto en la arquitectura planteada.

* **Vista de procesos:**

Esta vista es usada para definir los requerimientos funcionales y la visión global que los usuarios del negocio tienen de la aplicación y describir el modelo de negocio que la arquitectura debe cubrir.

* **Vista lógica**

Esta vista muestra los componentes principales de diseño y sus relaciones de forma independiente de los detalles técnicos y cómo la funcionalidad será implementada en la plataforma de ejecución.

* **Vista de Implementación:**

Esta vista describe la partición del sistema en las diferentes capas y sus subsistemas correspondientes, discute las reglas que gobiernan la asignación de componentes a las varias capas y subsistemas. Además describe la descomposición del sistema en término de los diferentes procesos que lo componen.

* **Vista Física o despliegue:**

La principal responsabilidad de esta vista es relacionar la vista anterior con los nodos físicos dentro de la infraestructura. Esta vista describe la distribución físicas del sistema.

*Para un mayor detalle ver Philippe Kruchten 4+1 View Model.*

* 1. **REPRESENTACIÓN DE LA ARQUITECTURA**

La arquitectura a utilizar será Cliente-Servidor. Se desarrollará una sola aplicación integrada, en la que solo se permitirá el acceso a todo usuario en modalidad consulta los demás roles requieren autenticación en el sistema las tareas a las cuales tengan acceso autorizado por rol. Se empleará un solo servidor centralizado pero permitiendo escalabilidad.

La arquitectura se muestra como una serie de vistas, utilizando la herramienta ***Enterprise Architect.***

# METAS Y RESTRICCIONES DE LA ARQUITECTURA

* 1. **METAS DE LA ARQUITECTURA**
* La arquitectura debe tener presente el futuro crecimiento de la aplicación por lo tanto debe estar preparada para ser dividida en servidores sin perder su calidad.
* Definir una arquitectura consistente con los objetivos de negocio y que han sido especificados en casos de uso y requerimientos no funcionales.
* La arquitectura debe ser simple y consistente para evitar complicaciones en el momento del desarrollo debido a la brevedad del tiempo de desarrollo.
* El diseño de la solución debe considerar la utilización de capas garantizando una separación adecuada entre la presentación, la lógica y reglas de negocio y la manipulación de datos.
* Los componentes de la solución deben de ser claros, definidos claramente las responsabilidades de cada subsistema y paquete. Obteniendo así una partición consistente de la solución
* La arquitectura debe considerar un manejo de excepciones adecuado que permite un seguimiento ágil de los errores que genere el sistema, este manejo de errores no debe afectar los indicadores de desempeño.

* 1. **RESTRICCIONES DE LA ARQUITECTURA**

#### 

# VISTA Y REALIZACIONES DE CASOS DE USO

#### PAQUETE DE USUARIOS

**Diagrama 1. Paquete de Usuarios**

**CU\_PU001***.*

#### PAQUETE DE DOCUMENTOS

**Diagrama 2. Paquete de Documentos**

**CU\_PD001***.*

#### PAQUETE DE DETECCIÓN

**Diagrama 3. Paquete de Detección**

**CU\_PDET001***.*

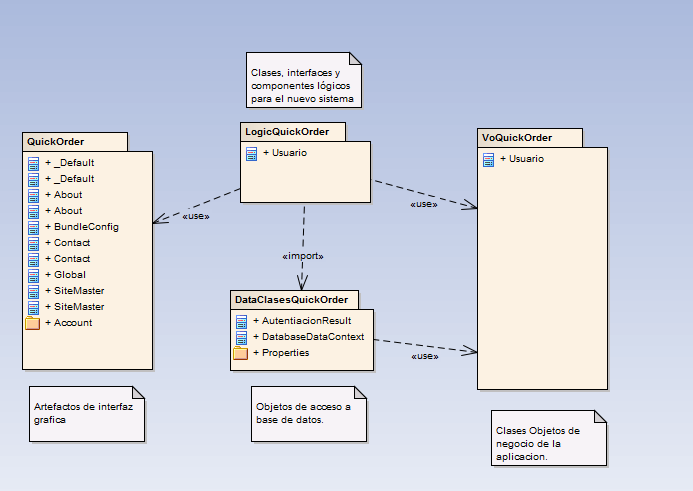
# VISTA LOGICA

* 1. **DESCRIPCIÓN**

El sistema está compuesto por 4 capas las cuales se describen a continuación.

* Capa de acceso a datos: **DataClasesQuickOrder** Esta clase se encarga del acceso a la base de datos se utilizara **LinqToSQL** para la conexión esta capa debe ser actualizada luego de cualquier cambio en los objetos de Base de Datos.
* Capa de lógica de negocio: **LogicQuickOrder** esta capa contiene cálculos y validaciones del sistema, adicionalmente hace el llamado a la capa de datos.
* Capa de objetos de negocio: **VoQuickOrder e**sta capa contiene los objetos de persistencia de datos de la aplicación cada uno de estos es una representación de los objetos en la base de datos.
* Capa de presentación: **QuickOrder** en esta capa se encuentran los objetos de interfaz de usuario.

A continuación un esquema de la distribución de los paquetes en la aplicación.

****

# VISTA DE PROCESO

## DIAGRAMA DE CLASES

# VISTA DE DESPLIEGUE

* 1. **DIAGRAMA DE COMPONENTES**

# Vista de Implementación

## Descripción

## Niveles o Capas

### Capa GUI

El componente GUI maneja las clases que permiten la interacción del usuario con el sistema.

### Capa Lógica

Este componente permite manejar la lógica del negocio de todas las áreas involucradas.

* Administración del Sistema
* Usuarios
* Privilegios
* Comparaciones
* Registro de documentos

### Capa Base de Datos

Este componente contiene toda la información que maneja el Sistema.

//CAMBIAR GRAFICO

Figura 8.1: Diagrama de Implementación del Sistema.

# Diagramas de Secuencia

## Paquete de Usuarios

//Meter diagramas

## Paquete de Documentos

//Meter diagramas

## Paquete de Detección

//Meter diagramas

# Tamaño y performance

La arquitectura de software elegida apoya los requisitos dominantes de capacidad en disco y latencia, según lo estipulado en el Documento de Especificación de Requisitos de Software de Antiplagium y el Documento de Visión del Proyecto del Sistema Antiplagium.

1. **Tiempo de respuesta en el acceso a la Base de Datos**

El sistema proveerá accesos a la base de datos con un tiempo de respuesta no mayor a los 5 segundos.

1. **Tiempo de respuesta de transacciones**

El programa no demorará más de 15 minutos en generar reporte de comparación de documentos haciendo uso del algoritmo elegido y guardándolo en base de datos.

1. **Espacio en disco para el cliente**

El espacio en disco necesario para la parte del cliente deberá tener como mínimo 500 MB de espacio libre para su correcto funcionamiento, esto incluye tanto el tamaño del software como el JRE 1.6.

1. **Espacio en disco para el servidor de Base de datos**

El espacio en disco necesario para la parte del Servidor de Base de datos deberá tener como mínimo 1 GB libres en disco para su correcto funcionamiento.

La arquitectura seleccionada apoya los requisitos de latencia y capacidad en disco en la puesta en práctica de una arquitectura cliente servidor. La porción del cliente solo se pone en ejecución en las PC locales de los distintos ambientes dentro de la empresa. Los componentes se han diseñado para asegurarse de obtener requisitos mínimos de disco y memoria en el lado de las PC del cliente.

# Calidad

Para un mejor aprovechamiento de la arquitectura de software se dan los siguientes requerimientos de calidad:

## Usabilidad

El sistema permitirá un manejo intuitivo por parte de los usuarios.

## Eficiencia

El programa no demorará más de 15 minutos en generar un reporte de comparación de documentos.

## Seguridad

El sistema permitirá el acceso a funcionalidades dependiendo del perfil del usuario que ingresa al sistema, validando su ingreso a través de una clave.

## Confiabilidad

El sistema tendrá en cuenta que la información ingresada en él sea válida, para lo cual mostrará mensajes que expliquen al usuario acerca de los errores que éste pudiera cometer y de aquellos que pueda cometer el mismo sistema.

## Mantenimiento

El sistema será flexible, facilitando su mantenimiento futuro.

## Estándares:

Se usará un estándar para todas las ventanas e interfaces con el usuario que tenga el Sistema Antiplagium, así como los nombres, códigos, etc.

