

Modelo de Diseño

CheckPoint

Laboratorio de Desarrollo de Software

GVR



*El propósito del Modelo de Diseño es empezar a realizar los casos de uso desarrollados durante las etapas anteriores en especial la etapa de captura de requerimientos. Es decir, tomar el Modelo de Casos de Uso y las Especificaciones Suplementarias creadas con anterioridad entre otros insumos y generar un modelo de diseño que pueda ser usado por los desarrolladores durante la etapa de implementación.*

Tabla de contenido

[Introducción 3](#_Toc257623469)

[Propósito 3](#_Toc257623470)

[Alcance 3](#_Toc257623471)

[Definiciones, siglas y abreviaturas. 3](#_Toc257623472)

[Referencias 3](#_Toc257623473)

[Visión general 3](#_Toc257623474)

[Diseño de Casos de Uso 3](#_Toc257623475)

[Diseño del Caso de Uso [nombre del caso de uso 1] 3](#_Toc257623476)

[Diagrama de paquetes 3](#_Toc257623477)

[Diagrama de Interacción 3](#_Toc257623478)

[Diseño de Flujo de eventos 3](#_Toc257623479)

[Requerimientos especiales o de implementación 3](#_Toc257623480)

[Diseño del Caso de Uso [nombre del caso de uso 2] 3](#_Toc257623481)

[Diagrama de paquetes 3](#_Toc257623482)

[Diagrama de Interacción 3](#_Toc257623483)

[Diseño de Flujo de eventos 3](#_Toc257623484)

[Requerimientos especiales o de implementación 3](#_Toc257623485)

[Diseño de Objetos 3](#_Toc257623486)

[[Objeto 1] 3](#_Toc257623487)

[[Objeto 2] 3](#_Toc257623488)

[Diseño de Subsistemas 3](#_Toc257623489)

[Subsistemas Específicos 3](#_Toc257623490)

[[Nombre del Subsistema Específico 1] 3](#_Toc257623491)

[Propósito 3](#_Toc257623492)

[Función 3](#_Toc257623493)

[Subordinados 3](#_Toc257623494)

[Dependencias 3](#_Toc257623495)

[Recursos 3](#_Toc257623496)

[Diagramas 3](#_Toc257623497)

[Diagrama de componentes 3](#_Toc257623498)

[Diagrama de Clases 3](#_Toc257623499)

[Diagrama de Secuencia 3](#_Toc257623500)

[Diagramas de Paquetes 3](#_Toc257623501)

[Diagrama de Colaboración 3](#_Toc257623502)

Modelo de Diseño

Introducción

El Modelo de diseño es un modelo de objetos que describe la realización física de los casos de uso centrándose en como los requisitos funcionales y no funcionales, junto con otras restricciones relacionadas con el entorno de implementación, tiene impacto en el sistema a desarrollar.

Propósito

Los propósitos de este documento son:

* Adquirir una compresión en profundidad de los aspectos relacionados con los requisitos no funcionales y restricciones relacionadas con los lenguajes de programación, componentes reutilizables, sistemas operativos, tecnologías de distribución y concurrencia, tecnologías de interfaz de usuario, tecnologías de gestión de transacciones, etc.
* Crear una entrada apropiada y un punto de partida para actividades de implementación subsiguientes capturando los requisitos o subsistemas individuales, interfaces y clases.
* Ser capaces de descomponer los trabajos de implementación en partes más manejables que puedan ser llevadas a cabo por diferentes equipos de desarrollo, teniendo en cuenta la posible concurrencia. Esto resulta útil en los casos en los que la descomposición no puede ser hecha basándose en los resultados de la captura de requisitos o análisis.

Alcance

[Esta sección es una breve descripción de a que se aplica el documento Modelo de Diseño, que es afectado o influenciado por este documento.]

Definiciones, siglas y abreviaturas.

Las palabras desconocidas o ambiguas que son utilizadas por primera vez en este documento se encuentran definidas en el documento “Glosario del Proyecto”, se debe remitir al mismos para encontrar allí términos importantes de este documento.

Referencias

* Especificación de Requerimientos.
* Modelo de Casos de Uso.

Visión general

Las secciones subsiguientes se componen de:

* **Diseño de los casos de uso** a implementar en esta iteración, con sus respectivos diagramas, de paquetes, de interacción, de flujo de eventos, y los requerimientos especiales o de implementación, todos estos según corresponda.
* **Diseño de objetos**, donde se describe brevemente al objeto, se enumeran sus atributos y métodos.
* **Diseño de subsistemas**,donde se especifican los subsistemas, su propósito, función subordinados, dependencias y recursos.
* **Diagramas**, finalmente la sección de diagramas, presenta diagrama de componentes, de clase, de secuencia, de paquete y de colaboración

Diseño de Casos de Uso

En esta sección se especifica el comportamiento de los casos de uso, mediante subsistemas u objetos de diseño que interactúan, y se determinan las operaciones e interfaces de los distintos subsistemas u objetos de diseño.

Diseño de CU01 - Login

Diagrama de paquetes

[En esta sección se identifican los objetos y subsistemas de diseño que intervienen en el caso de uso y sus relaciones.]

Diagrama de Interacción

[Se sugiere realizar un diagrama de interacción para representar la realización de cada Caso de Uso.]

Diseño de Flujo de eventos

[Descripción escrita en términos de objetos y subsistemas de diseño, que explica y complementa el diagrama de interacción y sus niveles.]

Requerimientos especiales o de implementación

[Descripción que recoge los requerimientos (no funcionales) en la realización de un caso de uso.]

Diseño del CU02 - HabilitarServicio

Diagrama de paquetes

[En esta sección se identifican los objetos y subsistemas de diseño que intervienen en el caso de uso y sus relaciones.]

Diagrama de Interacción

[Se sugiere realizar un diagrama de interacción para representar la realización de cada Caso de Uso.]

Diseño de Flujo de eventos

[Descripción escrita en términos de objetos y subsistemas de diseño, que explica y complementa el diagrama de interacción y sus niveles.]

Requerimientos especiales o de implementación

[Descripción que recoge los requerimientos (no funcionales) en la realización de un caso de uso.]

Diseño del CU03 - AsignarEncargadoAServicio

Diagrama de paquetes

[En esta sección se identifican los objetos y subsistemas de diseño que intervienen en el caso de uso y sus relaciones.]

Diagrama de Interacción

[Se sugiere realizar un diagrama de interacción para representar la realización de cada Caso de Uso.]

Diseño de Flujo de eventos

[Descripción escrita en términos de objetos y subsistemas de diseño, que explica y complementa el diagrama de interacción y sus niveles.]

Requerimientos especiales o de implementación

[Descripción que recoge los requerimientos (no funcionales) en la realización de un caso de uso.]

Diseño del CU04 - EditarServicio

Diagrama de paquetes

[En esta sección se identifican los objetos y subsistemas de diseño que intervienen en el caso de uso y sus relaciones.]

Diagrama de Interacción

[Se sugiere realizar un diagrama de interacción para representar la realización de cada Caso de Uso.]

Diseño de Flujo de eventos

[Descripción escrita en términos de objetos y subsistemas de diseño, que explica y complementa el diagrama de interacción y sus niveles.]

Requerimientos especiales o de implementación

[Descripción que recoge los requerimientos (no funcionales) en la realización de un caso de uso.]

Diseño del CU05 - DeshabilitarServicio

Diagrama de paquetes

[En esta sección se identifican los objetos y subsistemas de diseño que intervienen en el caso de uso y sus relaciones.]

Diagrama de Interacción

[Se sugiere realizar un diagrama de interacción para representar la realización de cada Caso de Uso.]

Diseño de Flujo de eventos

[Descripción escrita en términos de objetos y subsistemas de diseño, que explica y complementa el diagrama de interacción y sus niveles.]

Requerimientos especiales o de implementación

[Descripción que recoge los requerimientos (no funcionales) en la realización de un caso de uso.]

Diseño de Objetos

Servicio

* Descripción: Se trata de un objeto que representa un servicio prestado.
* Objetivo: Almacenar los atributos del servicio creado.
* Atributos:
* id
* nombre
* email\_valoraciones
* habilitado
* encargado
* icono
* Métodos

GETTER’s

* getId
* getNombre
* getEmailValoraciones
* getHabilitado
* getEncargado
* getIcono

SETTER’s

* setId
* setNombre
* setEmailValoraciones
* setHabilitado
* setEncargado
* setIcono
* Interacciones:
* Diagrama de estado
* Interfaces que implementa

GestorServicio

* Descripción: Se trata de un objeto que maneja objetos de tipo servicio.
* Objetivo: Almacenar los objetos servicio en la base de datos.
* Atributos:
* Servicio
* Métodos
* agregarServicio (CONSULTAR PARA AGREGARLO TMB EN CU)
* habilitarServicio
* deshabilitarServicio
* editarServicio
* Interacciones:
* Diagrama de estado
* Interfaces que implementa

Valoracion

* Descripción: Se trata de un objeto que representa una valoración asociada a un servicio.
* Objetivo:
* Atributos:
* id
* nombre
* tipo
* notificar
* foto
* descripción
* email
* habilitado
* vencimiento
* servicio
* Métodos

GETTER’s

SETTER’s

* Interacciones:
* Diagrama de estado
* Interfaces que implementa

GestorValoracion

* Descripción: Se trata de un objeto que maneja objetos de tipo valoracion.
* Objetivo:
* Atributos:
* valoracion
* Métodos
* altaValoracion (CONSULTAR)
* habilitarValoracion
* deshabilitarValoracion
* editarValoracion
* Interacciones:
* Diagrama de estado
* Interfaces que implementa

Ubicacion

[Se especifican los parámetros, reglas, condiciones usando la misma sintaxis del lenguaje y el código y métodos son especificados en pseudocódigo.]

* Descripción:
* Objetivo:
* Atributos:
* Métodos
* Interacciones:
* Diagrama de estado
* Interfaces que implementa

Usuario

[Se especifican los parámetros, reglas, condiciones usando la misma sintaxis del lenguaje y el código y métodos son especificados en pseudocódigo.]

* Descripción:
* Objetivo:
* Atributos:
* Métodos
* Interacciones:
* Diagrama de estado
* Interfaces que implementa

ValoracionHecha

[Se especifican los parámetros, reglas, condiciones usando la misma sintaxis del lenguaje y el código y métodos son especificados en pseudocódigo.]

* Descripción:
* Objetivo:
* Atributos:
* Métodos
* Interacciones:
* Diagrama de estado
* Interfaces que implementa

Diseño de Subsistemas

Subsistemas Específicos

[Nombre del Subsistema Específico 1]

Propósito

[Descripción de por qué el subsistema existe. Este atributo debe dar la razón de la creación del subsistema. Como ser la funcionalidad específica y los requerimientos de performance por los cuales fue creado. También describe requerimientos especiales que se deben lograr con él que no están incluidos en la especificación de requerimientos del software.]

Función

[Expresa qué realiza el subsistema. Establece la transformación aplicada a las entradas del subsistema para producir la salida deseada.]

Subordinados

[Se identifican los objetos de diseño y subsistemas de diseño que componen el subsistema que se describe. Se propone representar esta información con un diagrama de paquetes.]

Dependencias

[Descripción de la relación de este subsistema con otros subsistemas de diseño. Describir la naturaleza de cada interacción incluyendo características como tiempo y condiciones de la interacción. Estas, pueden involucrar la iniciación, orden de ejecución, datos compartidos, creación, duplicación, uso o almacenamiento.

Se propone representar esta información con una tabla de dependencias.]

|  |  |  |
| --- | --- | --- |
| Subsistema del que depende | Naturaleza de interacción | Características |
| [Identificación del subsistema del que depende] | [Condiciones para que se realice la interacción] | [Características de la interacción, como ser, pasaje de parámetros, mensajes, datos compartidos, etc.] |

Recursos

[Se deben Identificar y describir todos los recursos externos al diseño que necesita el subsistema para realizar su función. Especificar las reglas de interacción y métodos para usar el recurso.

Este atributo brinda información sobre elementos como dispositivos físicos (impresoras, particiones de disco, bancos de memoria), servicios de software (librerías, servicios del sistema operativo), y recursos de procesamiento (ciclos de CPU, ubicación de memoria, buffers).

Se deben describir características de uso como el tiempo de proceso al cual se debe adquirir el recurso e incluir la cantidad de tiempo de uso. Debe incluir también la identificación de capacidad potencial y facilidades de manejo del recurso.]

# Diagramas

Diagrama de componentes

[Lo que distingue el diagrama de componentes de otro tipo de diagramas es sin duda su contenido. Normalmente contiene componentes, interfaces y relaciones entre ellos.

Los componentes pertenecen a un mundo físico, es decir, representan a un bloque de construcción al modelar aspectos físicos de un sistema.

Cada componente debe tener un nombre que lo distinga de los demás. Al igual que las clases los componentes pueden enriquecerse con compartimientos adicionales que muestran sus detalles. ]



Diagrama de Clases

[En UML el diagrama de clases es uno de los tipos de diagramas o símbolo estático y tiene como fin describir la estructura de un sistema mostrando sus clases, atributos y relaciones entre ellos. Estos diagramas son utilizados durante el proceso de análisis y diseño de los sistemas informáticos, en donde se intentan conformar el diagrama conceptual de la información que se manejará en el sistema. Como ya sabemos UML es un modelado de sistema Orientados a Objetos, por ende los conceptos de este paradigma se incorporan a este lenguaje de modelado.

Los diagramas de clases tienen las siguientes características:

* Las clases define el ámbito de definición de un conjunto de objetos.
* Cada objeto pertenece a una clase.
* Los objetos se crean por instanciación de las clases.

En su representación gráfica contamos con:

* Nombre de la Clase.
* Atributos de la Clase.
* Operaciones con las Clases. ]



Diagrama de Secuencia

[En el diagrama de secuencia se muestra la interacción de los objetos que componen un sistema de forma temporal.

Un diagrama de secuencia muestra la interacción de un conjunto de objetos en una aplicación a través del tiempo y se modela para cada método de la clase]



Diagramas de Paquetes

[Los diagramas de Paquetes se usan para reflejar la organización de paquetes y sus elementos. Los usos más comunes de para los diagrama de paquete son para organizar diagramas de casos de uso y diagramas de clases, estos paquetes son como grandes contenedores de clases.

Los elementos contenidos en un paquete comparten el mismo espacio de nombres, esto significa que los elementos contenidos en un mismo espacio de nombres específico deben tener nombres únicos. Como otra característica de estos diagramas, cada paquete se debe identificar con un nombre único y opcionalmente mostrar todos los elementos dentro del mismo. ]



Diagrama de Colaboración

[Un diagrama de colaboración, se puede decir que es una forma alternativa al diagrama de secuencias a la hora de mostrar un escenario.

Este tipo de diagrama muestra las interacciones que ocurren entre los objetos que participan en una situación determinada. A diferencia del diagrama de secuencia, el diagrama de colaboración se enfoca en la relación entre los objetos y su topología de comunicación. En estos diagramas los mensajes enviados de un objeto a otro se representa mediante flechas, acompañado del nombre del mensaje, los parámetros y la secuencia del mensaje.

Estos diagramas están indicados para mostrar una situación o flujo de programa específico y son considerados uno de los mejores diagramas para mostrar o explicar rápidamente un proceso dentro de la lógica del programa. ]

