|  |  |
| --- | --- |
| http://www.neosistemassrl.com/neosistemas_15/images/stories/implementacion-de-software.pngMODELO DE IMPLEMENTACIONUNIDAD 5 | Descripción breveLos diagramas de componentes describen la descomposición físicos de los elementos de un sistema (modulo, base de datos, programa ejecutable, etc.) y sus relaciones. Muestran las opciones de realización incluyendo código fuente, BINARIO y ejecutable, pueden ser simples archivos, paquetes, bibliotecas cargadas dinámicamente, etc.TAPIA HERNANDEZ ARELIFUNDAMENTOS DE INGENIERIA DE SOFTWARE |

**INDICE**

**UNIDAD 5 MODELO DE IMPLEMENTACION**

**5.1 Diagrama de componentes………………………………………………….PAG.2**

**5.2 Diagrama de despliegue………......................................................……..PAG.3**

**5.3 Modelos de prueba……………………………………………………..…….PAG.4**

 **Bibliografía……………………………………….…………………….…..…PAG.6**

**5.1 Diagrama de componentes**

Los diagramas de componentes describen la descomposición físicos de los elementos de un sistema (modulo, base de datos, programa ejecutable, etc.) y sus relaciones. Muestran las opciones de realización incluyendo código fuente, BINARIO y ejecutable, pueden ser simples archivos, paquetes, bibliotecas cargadas dinámicamente, etc.

Elementos

› Normalmente los DC contienen los siguientes elementos:

› Componentes

› Interfaces

› Relaciones de dependencia, generalización, asociación y realización.

› Paquetes o subsistemas.

 Relaciones de dependencia de los DC.

› Se pueden agrupar en paquetes así como los objetos de clases, además pueden tener entre ellos relaciones, tales como:

› Generalización

› Asociación

› Agregación

› Realización

 Dependencia

Estereotipos de los componentes.

UML define cinco estereotipos estándar que se aplican a los componentes:

• Executable: Especifica un componente que se puede ejecutar en un nodo.

• Library: Especifica una biblioteca de objetos estática o dinámica.

• Table: Especifica un componente que representa una tabla de una base de datos.

• File: Especifica un componente que representa un documento que contiene código fuente o datos.

• Document: Especifica un componente que representa un documento.

Dependencias entre componentes.

Se utilizan en los DC para indicar que un componente se refiere a los servicios ofrecidos por otro componente.

 Subsistemas:

• Los distintos componentes pueden agruparse en paquetes según un criterio lógico y con vistas a simplificar la implementación.

• Son paquetes estereotipados en <<subsistemas>>.

Funcionalidad de los subsistemas.

• Los subsistemas organizan la vista de realización de un sistema.

• Cada subsistema puede contener componentes y otros subsistemas.

• La descomposición en subsistemas no es necesariamente una descomposición funcional.

• La relación entre paquetes y clases en el nivel lógico es el que existe entre subsistemas y componentes en el nivel físico.

• Paquetes (Categorias) y clases en el nivel lógico. Paquetes (Subsistemas) y componentes en el nivel físico.

Pasos para elaborar un diagrama de componentes:

1. Previamente al diagrama de componentes debemos de tener hecho el diagrama de clases.

2. Se debe identificar a todos las clases que participaran en el sistema o subsistema a desarrollar.

3. Una vez identificado las clases, se procede a identificar sus métodos.

4. Estos métodos pasaran a ser módulos con líneas de código independientes.

5. Estos módulos serán los componentes de nuestro diagrama.

6. Estos componentes se relacionan entre si por medio de sus interfaces.

**5.2 Diagrama de despliegue**

Un diagrama de despliegue muestra las relaciones físicas entre los componentes hardwarey softwareen el sistema final, es decir, la configuración de los elementos de procesamiento en tiempo de ejecución y los componentes software(procesos y objetos que se ejecutan en ellos).

 Los diagramas de despliegue representan a los nodos y sus relaciones.

 Los nodos son conectados por asociaciones de comunicación tales como enlaces de red, conexiones TCP/IP, microondas, etc.

Los diagramas de despliegue son los complementos de los diagramas de componentes que unidos, proveen la vista de implementación del sistema.

 Características

Describen la arquitectura física del sistema durante la ejecución, en términos de:

–procesadores

–dispositivos

–componentes de software

-Describen la topología del sistema: la estructura de los elementos de hardware y el software que ejecuta cada uno de ellos.

Componentes de diagrama de despliegue

 Nodo: son objetos físicos que existen en tiempo de ejecución, y que representan algún tipo de recurso computacional (capacidad de memoria y procesamiento):

* Instancia de nodo.
* Estereotipo de nodo.

 Artefactos: es un producto del proceso de desarrollo de software, que puede incluir los modelos del proceso Ej: archivos fuente, ejecutables, documentos de diseño, reportes de prueba, prototipos, manuales de usuario y más.

Asociación: Una asociación representa una ruta de comunicación entre los nodos. Donde esta asociación va incluida con misma dependencia del diagrama de componentes.

 Usos

• Sistemas cliente-servidor

• Sistemas completamente distribuidos

**5.3 Modelos de prueba**

La fase de pruebas del sistema tiene como objetivo verificar el sistema software para comprobar si este cumple sus requisitos.

Dentro de esta fase pueden desarrollarse varios tipos distintos de pruebas en función de los objetivos de las mismas. Algunos tipos son pruebas funcionales, pruebas de usabilidad, pruebas de rendimiento, pruebas de seguridad, etc.

Este trabajo se centra en pruebas funcionales de aplicaciones con interfaces gráficas. Estas pruebas verifican que el sistema software ofrece a los actores humanos la funcionalidad recogida en su especificación.

Una de las técnicas más empleadas para la especificación funcional de sistemas software sonlos casos de uso. Las principales ventajas de los casos de uso son que ocultan los detalles internos del sistema, son rápidos de construir, fáciles de modificar y entender por los clientes y futuros usuarios del sistema y pueden aplicarse a distintos tipos de sistemas y.

Actualmente, existe un amplio número de propuestas que describen cómo generar pruebas del sistema a partir de los casos de uso. Aunque la generación de pruebas se adapta a la filosofía propuesta por MDA.

Este trabajo se centra en la definición de un conjunto de modelos que den soporte al proceso de generación de pruebas del sistema desde una perspectiva MDA.

Las pruebas deben de hacerse en todas las fases del desarrollo.

Tipos de Pruebas:

* Pruebas de Aceptación à Casos de Uso
* Pruebas de Integración/Sistema à Diagramas de Secuencia/Escenarios
* Pruebas Unitarias à Clases/Módulos
* Pruebas de Regresión
* Pruebas de Facilidad de Uso
* Pruebas de Cobertura
* Pruebas de Rendimiento (profile)

 Formato Plan de Pruebas

****

**BIBLIOGRAFIA**

[http://adilennefuentesgutierrez.blogspot.mx/2014/12/53-modelos-de-prueba.htm**l**](http://adilennefuentesgutierrez.blogspot.mx/2014/12/53-modelos-de-prueba.html)