Modelo 4+1

Laura Mendez Segundo

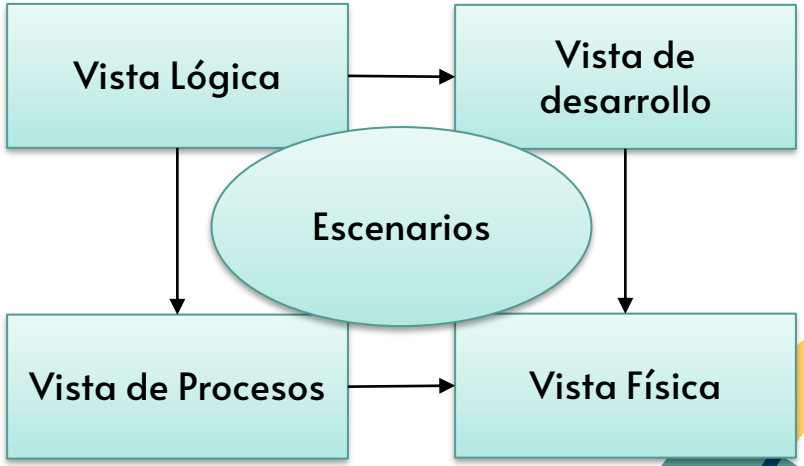
¿Qué es?

El modelo 4+1 describe la arquitectura del software usando cinco vistas concurrentes. Cada vista se refiere a un conjunto de intereses de diferentes stakeholders (usuarios finales, desarrolladores, ingenieros de sistemas, administradores del proyecto, etc.) del sistema.

Diseñado por Philippe Kruchten

Usuarios finales
Funcionalidad

Programadores
Administración del software



Integradores
Performance
Escalabilidad

Ingenieros de sistemas
Topología
Comunicaciones

Características de cada vista

Vista lógica

Describe el modelo de objetos del diseño cuando se usa un método de diseño O.O.

Vista de procesos

Describe lso aspectos de concurrencia y sincronización del diseño.

Vista física

Describe el mapeo del software en el hardware y refleja los aspectos de distribución.

Vista de desarrollo

Describe la organización estética del software en su ambiente de desarrollo.

Los diseñadores de software pueden organizar la descripción de sus decisiones de arquitectura en estas cuatro vistas, y luego ilustrarlas con un conjunto reducido de casos de uso o escenarios, los cuales constituyen la quinta vista..

Arquitectura Lógica

A

a) Apoya los requisitos funcionales de un sistema, es decir lo que el sistema debe hacer.

b) Se descompone el sistema en una serie de objetos o clases de objetos, utilizando mecanismos como encapsulamiento, herencia y abstracción.

c) Aquí se hace uso de diagramas de clases y templates de clases, que nos sirven para definir las características y el comportamiento de cada clase en el sistema.

Arquitectura Lógica

B

d) También se utiliza los diagramas de estado para definir el comportamiento interno de un objeto.

e) Para representarlo con UML se emplea diagrama de clases, diagrama de comunicación (en versión 1 son diagramas de colaboración), diagramas de secuencia..

Arquitectura de Procesos

A

a) Toma en cuenta algunos requisitos no funcionales tales como el performance y la disponibilidad.

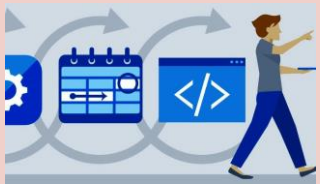
b) Se enfoca en asuntos de concurrencia y distribución, integridad del sistema, de tolerancia a fallas.

Arquitectura de Procesos

B

c) La vista de procesos también especifica en cual hilo de control se ejecuta efectivamente una operación de una clase identificada en la vista lógica.

d) Para representarlo con UML se emplea diagrama de actividades.



Proceso

Es una agrupación de tareas que forman una unidad ejecutable.

Los procesos representan el nivel al que la arquitectura de procesos puede ser controlada tácticamente. Además, los procesos pueden replicarse para aumentar la distribución de la carga de procesamiento, o para mejorar la disponibilidad.

La vista de desarrollo

a)Se centra en la organización real de los módulos de software en el ambiente de desarrollo del software. El software se empaqueta en partes pequeñas que pueden ser desarrollados por uno o un grupo pequeño de desarrolladores. Los subsistemas se organizan en una jerarquía de capas, cada una de las cuales brinda una interfaz estrecha y bien definida hacia las capas superiores.

b) Utiliza el diagrama de componentes para describir los componentes del sistema.

c) Para representarlo con UML emplea el diagrama de paquetes.

La vista física

a) Se centra en los requisitos no funcionales, tales como la disponibilidad del sistema, la fiabilidad (tolerancia a fallos), ejecución y escalabilidad. Y también presenta cómo los procesos, objetos, etc., corresponden a nodos de proceso:

a) Componentes: nodos de proceso.
b) Conectores: LAN, WAN, bus, etc.
c) Contenedores: subsistemas físico

b) Para representarlo con UML se emplea el diagrama de despliegue/ emplazamiento

Vista de escenarios

a) Corresponde con instancias de casos de uso que unifican todas las vistas. Así, desde casos de uso se debiera poder hacer una trazabilidad a todos los componentes del sistema software, viendo, por ejemplo, que máquinas, o clases, o componentes, o .jar, o procesos, son los responsables de que el sistema cubra una cierta funcionalidad.

b) Esta vista es redundante con las otras (y por lo tanto “+1”), pero sirve a dos propósitos principales:

1. Como una guía para descubrir elementos arquitectónicos durante el diseño de arquitectura.
2. Como un rol de validación e ilustración después de completar el diseño de arquitectura, en el papel y como punto de partido de las pruebas de un prototipo de la arquitectura.

Vista de escenarios

c) Para representarlo con UML se emplean casos de uso.

Las vistas que componen la arquitectura 4 + 1 no son completamente independientes unas de otras, se conectan entre sí siguiendo ciertas reglas o heurísticas de diseño. Por ejemplo, cuando se pasa de la vista lógica a la vista de procesos se varias características importantes de las clases de la vista lógica, pues la arquitectura lógica tiene en cuenta solo el aspecto funcional de los requisitos.

Para Bernard Witt y otros autores indican cuatro fases para el diseño de la arquitectura, aunque estas fases pueden no ser suficientes para validar la arquitectura después de terminar con cada una de ellas. Existe también otro enfoque de desarrollo de la arquitectura, basándose en la identificación de la funcionalidad critica del sistema, expresada en forma de escenarios o casos de uso.

El modelo 4+1 vistas, es una propuesta que establece las diferentes perspectivas a través de las cuales se puede representar el diseño y arquitectura de un sistema de software

Bibliografía.

1. García Rubio, Félix Oscar. Medición y estimación del software: Técnicas y Métodos para mejorar la calidad y la productividad. AlfaOmega. México 2008. 332 págs. ISBN 9788478978588.

2. Kan Stephen H., Metrics and Models in Software Quality Engineering, 2 edition. Addison-Wesley Professional

3. Piattini Mario, García Félix. Calidad de Sistemas Informáticos. Alfaomega. México 2007. 388 págs. ISBN 9789701512678

4. Pressman Roger S. Ingeniería del software: Un enfoque Práctico. Mc Graw Hill. México 2005. 900 pags. ISBN 9701054733.

5. Schach Stephen. Análisis y diseño orientado a objetos con UML y el proceso unificado. Mc Graw Hill. España 2005. 458 págs. ISBM 9789701049822

6. Sommerville Ian. Ingeniería de Software. Addison Wesley. España 2008. 712 págs ISBN 9789702602064.