INGENIERÍA DE SOFTWARE ORIENTADA A ASPECTOS.

La ingeniería de software orientada a aspectos (AOSE, por las siglas de Aspect-Oriented Software Engineering) es un enfoque al desarrollo de software que está destinado a enfrentar este problema y así elaborar.

LA SEPARACIÓN DE INTERESES.

La separación de competencias o intereses (concerns): es un principio clave del diseño e implementación de software.

Tipos de competencias o intereses para el participante:

1. Competencias funcionales, que se relacionan con la funcionalidad específica a incluir en un sistema.

2. Competencias de calidad del servicio, las cuales se relacionan con el comportamiento no funcional de un sistema.

3. Competencias de política, que se relacionan con políticas generales que rigen el uso de un sistema.

4. Competencias de sistema, las cuales se relacionan con atributos del sistema como un todo, tales como su mantenibilidad o configurabilidad.

5. Competencias organizacionales, que se relacionan con las metas y prioridades de la organización.

ASPECTOS, PUNTOS DE ENLACE Y PUNTOS DE CORTE.

Tres enfoques diferentes al tejido de aspectos:

1. Preprocesamiento de código fuente, en el que un tejedor toma entrada de código fuente y genera nuevo código fuente en un lenguaje como Java o C++, que luego pueden compilarse usando el compilador de lenguaje estándar.

2. Tejido de tiempo de vinculación, en el que el compilador se modifica para incluir un tejedor de aspectos.

3. Tejido dinámico en tiempo de ejecución. En este caso, se monitorizan los puntos de enlace y, cuando ocurre un evento referenciado en un punto de corte, se integra el consejo correspondiente con el programa en ejecución.

INGENIERÍA DE SOFTWARE CON ASPECTOS.

Tipos de extensiones que se derivan de los distintos tipos de competencias:

1. Extensiones funcionales secundarias: Agregan capacidades adicionales a la funcionalidad que ofrece el sistema central.

2. Extensiones de política: Agregan capacidades funcionales para soportar políticas de la organización.

3. Extensiones QoS: Agregan capacidades funcionales para ayudar a alcanzar los requerimientos de calidad del servicio que se especificaron para el sistema.

4. Extensiones de infraestructura: Estas extensiones agregan capacidades funcionales para soportar la implementación de un sistema en alguna plataforma de implementación específica.

INGENIERÍA DE REQUERIMIENTOS ORIENTADOS A COMPETENCIAS.

Las competencias reflejan los requerimientos de las partes interesadas. Dichas competencias pueden reflejar la funcionalidad requerida por un participante, la calidad de servicio del sistema, políticas o conflictos de la organización que se relacionan con los atributos del sistema como un todo.

DISEÑO Y PROGRAMACIÓN ORIENTADA A ASPECTOS.

Es el proceso de diseñar un sistema que utilice los aspectos para implementar las competencias transversales y extensiones que se identificaron durante el proceso de ingeniería de requerimientos.

En un proceso de diseño orientado a aspectos debe incluirse las siguientes actividades:

1. Diseño del sistema central: En esta etapa se diseña la arquitectura del sistema para soportar la funcionalidad central del sistema.

2. Identificación y diseño de aspectos: A partir de las extensiones identificadas en los requerimientos del sistema, habrá que efectuar un análisis para ver si son aspectos en sí mismos o si deben descomponerse en varios aspectos.

3. Diseño de composición: En esta etapa se analiza el sistema central y los diseños de aspectos para descubrir dónde deben combinarse los aspectos con el sistema central.

4. Análisis y resolución de conflictos: Un problema con los aspectos es que pueden interferir entre sí cuando se combinan con el sistema central.

5. Diseño de nombre: Ésta es una importante actividad de diseño que define estándares para nombrar las entidades del programa.

VERIFICACIÓN Y VALIDACIÓN.

La verificación y la validación: implican demostrar que un programa cumple su especificación (verificación) y satisface las necesidades reales de las partes interesadas (validación).

En un sistema orientado a aspectos, existen dos problemas con este enfoque:

1. ¿Cómo puede usarse el conocimiento del programa para derivar sistemáticamente pruebas del programa?

2. ¿Qué significa exactamente cobertura de prueba?

Problemas en las pruebas de los programas orientados a aspectos:

1. ¿Cómo deben especificarse los aspectos de manera que puedan derivarse pruebas para dichos aspectos?

2. ¿Cómo pueden probarse los aspectos independientemente del sistema base con el que deben tejerse?

3. ¿Cómo puede someterse a prueba la interferencia de aspectos? Como se estudió, la interferencia de aspectos ocurre cuando dos o más aspectos usan la misma especificación de punto de corte.

EJEMPLO INGENIERÍA DE SOFTWARE ORIENTADA A ASPECTOS.

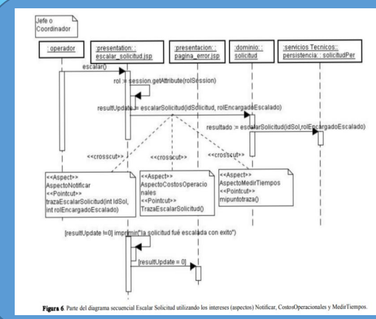

Figura 6. Parte del diagrama secuencial Escalar Solicitud utilizando los intereses (aspectos) Notificar, CostosOperacionales y MedirTiempos.

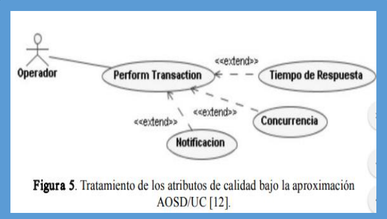

Figura 5. Tratamiento de los atributos de calidad bajo la aproximación AOSD/UC [12].

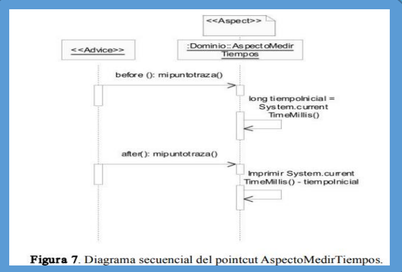

Figura 7. Diagrama secuencial del pointcut AspectoMedirTiempos.