|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  | **3** |
|  | TEMA |

Principios que guían la práctica de la ingeniería del *software*

[3.1] Principios que guían el proceso

[3.2] Principios que guían la práctica

[3.3] Principios de comunicación

[3.4] Principios de planificación

[3.5] Principios de modelado

[3.6] Principios de construcción

[3.7] Principios de despliegue

[3.8] Referencias

Esquema



Ideas clave

3.1. Principios que guían el proceso

Los **principios fundamentales** que aplican a todo el **proceso de *software*** son la agilidad, la orientación a la calidad, la adaptación, la eficacia del equipo, la comunicación y coordinación, la gestión del cambio, la evaluación del riesgo y la generación de productos con valor.

3.2. Principios que guían la práctica

Los **principios fundamentales** para práctica de la **ingeniería de *software*** son «divide y vencerás», la abstracción, la coherencia, la transferencia de información, la modularidad, los patrones, la variedad de perspectivas y el mantenimiento del *software*.

3.3. Principios de comunicación



Para logarlo habría que seguir una serie de **principios fundamentales**: escuchar, preparación previa, líder definido, comunicación cara a cara, tomar notas y documentar decisiones, colaboración, estructurar la reunión, dibujar para aclarar, avanzar, negociar con beneficio para todos.

3.4. Principios de planificación

La planificación de un proyecto de *software* ofrece al equipo y al cliente una **guía** para conocer en todo momento dónde y cómo se debería encontrar el proyecto.

Para ello deben aplicarse los siguientes principios: entender el alcance del proyecto, involucrar a todos los participantes, reconocer que es iterativa, estimar en base a lo conocido, tener en cuenta los riesgos, ser realista, establecer un nivel de detalle proporcional, asegurar la calidad, gestionar cambios, realizar seguimiento frecuente y ajustes de la planificación.

3.5. Principios de modelado

Los modelos deben representar la arquitectura, las funcionalidades y la información. Para llevarlos a cabo correctamente el equipo debe tener como objetivo construir *software* (no modelar), crear solo los modelos necesarios, lo más sencillos posibles y susceptibles al cambio, dar un propósito a cada modelo, adaptar los modelos al sistema, construir modelos útiles, utilizar la representación que facilite la comunicación, seguir el instinto de la experiencia y obtener retroalimentación lo antes posible.



* **Modelos de requisitos**, que representan los requisitos del cliente a nivel de información, funcionalidad y comportamiento. Para su correcta realización, debe representarse y entenderse el dominio de información del problema, definirse las funciones que realizará el *software*, representarse el comportamiento del *software*, estructurar los modelos, todo ello, para permitir avanzar de la información a la implementación.
* **Modelos de diseño**, que representan las características que tendrá el *software* para facilitar su futuro desarrollo a nivel de arquitectura, interfaz de usuario y componente. Estos modelos deben permitir enlazar con el modelo de requisitos, tener en cuenta la arquitectura del sistema, prestar atención al diseño de datos e interfaces, independizar funcionalmente los componentes y no acoplarlos en exceso, facilitar la comprensión de los modelos y realizarse de modo iterativo.

3.6. Principios de construcción

La construcción de *software* conlleva actividades de codificación y pruebas que presentan una serie de principios a seguir:



3.7. Principios de despliegue

El **despliegue** implica la entrega, el apoyo y la retroalimentación. El despliegue no ocurre una sola vez durante la ejecución de un proyecto, sino cada vez que se entrega al cliente un incremento de *software*.

En el despliegue se debe tener en cuenta las **expectativas del cliente**, se debe ensamblar y probar el paquete completo, para entregarlo solo cuando libre de errores. Además, se deberá proporcionar información, documentación y material de aprendizaje del *software* entregado.

3.8. Referencias

Pressman, R. (2010). *Ingeniería del software* (7ª ed.). México: McGrawHill.

Lo + recomendado

No dejes de leer…

**El mítico hombre-mes**

En este ensayo de Federick P. Brooks se explica de un modo original los principios para organizar un equipo de desarrollo en base a un equipo quirúrgico.

Accede al ensayo desde el aula virtual o a través de la siguiente dirección web:

Traducción: <http://naldog.blogspot.com.es/2013/08/el-mitico-hombre-mes-mythical-man-month.html>

Libro completo: <https://archive.org/details/MythicalManMonth>

No dejes de ver…

**Principios del éxito personal**

En los principios que guían la práctica del *software* hay que tener muy en cuenta al personal del equipo y trabajar con ellos que hagan suyo el éxito del proyecto. En este vídeo se proyectan siete claves para el éxito personal.



Accede al vídeo desde el aula virtual o a través de la siguiente dirección web:

<http://www.youtube.com/watch?v=uRFjWLI-m4o>

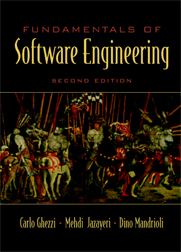
+ Información

A fondo

**Ingeniería de *software***

Ghezzi, C., Jazayeri, M. & Mandrioli, D. (2002). *Fundamentals of Software Engineering*. Ed. Prentice Hall

Los fundamentos de la ingeniería de *software* deben ser conocidos entendidos y aplicados por todo el equipo implicado en los proyectos de *software*. Este libro trata los fundamentos de la ingeniería de *software* como son el diseño, la especificación, la verificación, la producción, la gestión y las herramientas. También incorpora casos de estudio y ejercicios prácticos para facilitar la comprensión.



The Power of Process

Steve McConnell en su artículo en IEEE Computer presenta como la aplicación de los principios de proceso de *software* repercute en el trabajo productivo del equipo.

Accede al artículo desde el aula virtual o a través de la siguiente dirección web:

<http://www.stevemcconnell.com/articles/art09.htm>

Enlaces relacionados

Conceptos, principios y métodos

Roger Pressman & Associates ofrece una web donde recopilan información sobre las distintas áreas de la ingeniería de *software*. El enlace que se presenta se concentra en conceptos, principios y métodos de la ingeniería de *software*



Accede a la página desde el aula virtual o a través de la siguiente dirección web:

[http://www.rspa.com/spi/practice.html#concepts](#concepts)

Test

**1.** Todo proceso *software* debe tener en cuenta:

A. La agilidad es un principio a tener en cuenta solo en metodologías ágiles.

B. La comunicación y coordinación del equipo es una tarea exclusiva del jefe de proyecto.

C. Hay que evitar los cambios y no deben aprobarse durante el ciclo de vida de *software* en el que se esté.

D. Hay que establecer planes de contingencia por si algo sale mal.

**2.** En la práctica de la ingeniería de *software*:

A. Nunca se debe aplicar el principio «divide y vencerás».

B. Los problemas hay que presentarlos desde una única perspectiva para evitar ambigüedades.

C. El uso de patrones facilita la resolución de problemas recurrentes.

D. El mantenimiento de *software* no hay que plantearlo hasta el final del proyecto.

**3.** Para lograr la comunicación efectiva:

A. Es preferible asistir a las reuniones sin hacer investigaciones previas sobre el problema a resolver, para evitar prejuicios iniciales.

B. Cuando algo no está claro, es bueno dibujar para clarificarlo.

C. La comunicación por videoconferencia es preferible a la presencial.

D. Cuando no haya un acuerdo en algún tema, hay que centrarse en ese tema hasta que se consensue.

**4.** En la planificación de un proyecto:

A. Debe seguirse a rajatabla desde el inicio al final del proyecto.

B. Los hitos de los entregables deben marcarlos unilateralmente el cliente.

C. Hay que ser realista y saber que las personas no trabajan al 100% todos los días.

D. Debe ser muy detallada desde el inicio para ofrecer confianza al cliente.

**5.** En referencia al modelado de *software*

A. Es el principal objetivo del equipo de *software.*

B. Deben estar perfectamente diseñados y seguir la notación estándar.

C. No debe presentarse al equipo de *software* hasta que esté finalizado.

D. No se deben crear más modelos que los necesarios.

**6.** Durante la construcción de *software*:

A. La programación debe realizarse individualmente para lograr una mayor productividad.

B. Hay que evitar los entornos de programación que proporcionen herramientas.

C. Hay que crear un conjunto de pruebas unitarias antes de comenzar a codificar.

D. Se debe poner nombres cortos de variables, de una o dos letras máximo.

**7.** En las pruebas de *software*:

A. No es posible realizar pruebas exhaustivas.

B. Una prueba exitosa es aquella que no encuentra problemas en el *software.*

C. Las pruebas de *software* están orientadas a verificar la calidad del *software*, pero no los requisitos del cliente que se verificaron con el análisis y el diseño.

D. Las pruebas deben realizarse comenzando con todo el aplicativo y gradualmente ir probando cada vez módulos más pequeños.

**8.** En la fase de despliegue del *software*:

A. Hay que evitar que el cliente tenga grandes expectativas.

B. El *software* debe entregarse en plazo, aunque contenga pequeños errores.

C. Una vez entregado el *software* deberá tenerse reuniones con el cliente para acordar el modelo de apoyo para la resolución de problemas.

D. Se debe obtener retroalimentación del cliente del *software* entregado.