



TALLER 2

PARCIAL No. 1

**METODOLOGIAS DE DESARROLLO EN
SOFTWARE**

METODOLOGIAS DE DESARROLLO EN SOFTWARE
ÁREA DE COMPUTACION
DEPARTAMENTO DE ITIN

Taller -C1.

Nombres del estudiante:	Cristian Jesus Becerra Loaiza Jhon Kevin Castillo Quishpe Ismael Alejandro Silva Flores
Nivel:	Tercer Nivel
NRC:	29022
Asignatura:	Met. de Desarrollo en Software
Nombre del profesor:	Ing. Jenny Alexandra Ruiz Robalino

METODOLOGIAS DE DESARROLLO EN SOFTWARE
ÁREA DE COMPUTACION
DEPARTAMENTO DE ITIN

Contenido

Autoevaluación del Capítulo 1:	2
1. La persona que acuñó por primera vez el término “ingeniería del software” fue:	2
2. Los elementos que componen el software son:	2
3. Oficialmente, el término ingeniería del software se acuñó en:	2
4. La definición de tipo de software correcto es:	3
5. ¿Cuáles son los atributos de un buen software?	3
6. Las características del software son:	3
7. La crisis del software se refiere a los problemas que desde sus inicios ha ido experimentado este. Muchas veces los problemas de gran magnitud se generan debido a la mínima eficacia que presenta una gran cantidad de empresas al momento de realizar un software.	4
8. A partir del siguiente gráfico, los nombres de las fases del modelo en Cascada (Waterfall) son:	5
9. El modelo de proceso de software en espiral propuesto por Bohem conjuga la naturaleza iterativa de la construcción de prototipos con los aspectos controlados y sistemáticos del modelo lineal secuencial. La etapa que no pertenece al modelo es:	6
10. Se construye un buen sistema de información considerando que el punto de partida es:	6

Autoevaluación del Capítulo 1:

1. La persona que acuñó por primera vez el término “ingeniería del software” fue:

- a) Margaret Hamilton.
- b) Margaret Sanger.
- c) Margaret Atwood.

Justificación:

La primera persona a quien se le atribuye el uso del término por primera vez es Margaret Hamilton, quien después de haber trabajado en el programa SAGE (Semi-automatic Ground Environment), se convirtió en la desarrolladora principal de Skylab y Apollo, mientras trabajaba en el “Draper Lab” (Piattini, 2016).

2. Los elementos que componen el software son:

- a) Personal, proceso y producto.
- b) Programas, procedimientos, documentación y datos relacionados.
- c) Programas o instrucciones, partes y piezas y datos.

Justificación:

Ingeniería del Software es la aplicación práctica del conocimiento científico al diseño y construcción de programas de computadora y a la documentación asociada requerida para desarrollar, operar (funcionar) y mantenerlos. Se conoce también como desarrollo de software o producción de software (García, s.f.).

3. Oficialmente, el término ingeniería del software se acuñó en:

- a) La Conferencia de la OTAN de 1968.
- b) La Conferencia de la CEPAL de 1963.
- c) La Conferencia de la OTAN de 1986.

Justificación:

Más de uno indica que este término aparece formalmente en la Conferencia de la OTAN de 1968 sobre ingeniería del software y fue referenciado por Friedrich Bauer (Hinojosa, 2019).

4. La definición de tipo de software correcto es:

- a) Programas que resuelven necesidades específicas de las organizaciones (software de sistemas).
- b) Conjunto de programas que han sido escritos para servir a otros programas (software de gestión o aplicación).
- c) Software que hace uso de algoritmos no numéricos para resolver problemas complejos para los que no son adecuados el cálculo o el análisis directo (software de inteligencia artificial).

Justificación:

Las otras dos opciones están incorrectamente definidas. El "software de sistemas" sirve al propio ordenador (ej: sistemas operativos), no a necesidades específicas de las organizaciones. El "software de aplicación" es el que sirve al usuario final para tareas específicas, no a otros programas.

5. ¿Cuáles son los atributos de un buen software?

- a) Funcionalidad y el rendimiento requerido por el usuario.
- b) Hacer que se malgasten los recursos del sistema.
- c) Mantenable, confiable y fácil de utilizar.

Justificación:

La ingeniería de software es el establecimiento de principios fundamentales de la ingeniería con objeto de desarrollar, en forma económica, software que sea confiable y que trabaje con eficiencia en máquinas reales” (Bauer, 1968).

6. Las características del software son:

- a) El software usa componentes estándar con funciones e interfaces bien definidas.

b)El software se desarrolla o modifica con intelecto, no se fabrica en el sentido clásico.

c)El software se desgasta con el transcurso del tiempo.

Justificación:

Para los años cincuenta, se aplica al desarrollo de software el mismo proceso de desarrollo de hardware, es decir, sobre la base de la experiencia de quienes lo realizan. El software en esta década es considerado como un producto añadido y la programación de computadores es todo un arte, para el cual no existen métodos sistemáticos. El desarrollo de software se realizaba sin ninguna planificación, una sola persona lo escribía, lo ejecutaba y, si fallaba, lo depuraba (Hinojosa, 2019).

El diseño en sí mismo era un proceso implícito que se realizaba en la mente del programador y en el cual la documentación simplemente no existía (Piattini, 2016).

7. La crisis del software se refiere a los problemas que desde sus inicios ha ido experimentado este. Muchas veces los problemas de gran magnitud se generan debido a la mínima eficacia que presenta una gran cantidad de empresas al momento de realizar un software.

a)Verdadero.

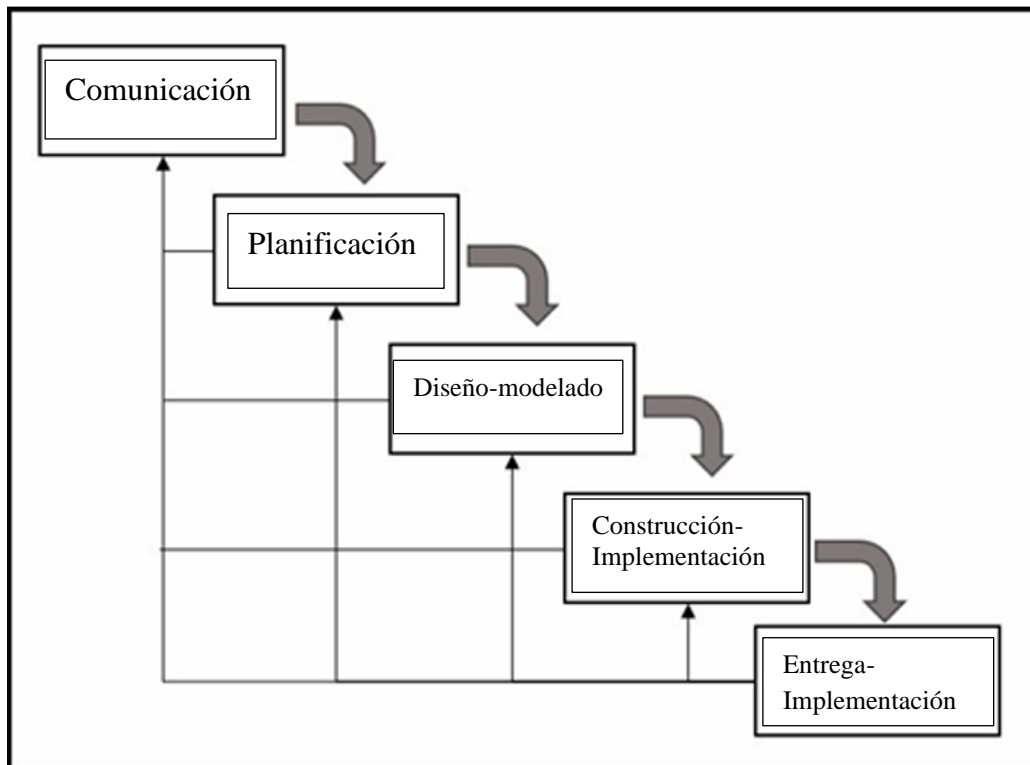
b)Falso.

c)Ninguna de las opciones

Justificación:

El ingeniero se enfrenta a menudo con problemas mal definidos y con soluciones parciales, y tiene que apoyarse en métodos empíricos para evaluar soluciones. En la literatura se encuentra una larga historia de proyectos de software que fallaron (Figura 1). Ante esta realidad, la ingeniería de software se enfoca precisamente en la solución de problemas propios de la naturaleza cambiante y compleja del software (Charette, s.f.).

8. A partir del siguiente gráfico, los nombres de las fases del modelo en Cascada (Waterfall) son:



a) Gestión de proyecto, comunicación, planificación, diseño-modelado, construcción-implementación.

b) Comunicación, planificación, diseño-modelado, construcción-implementación, entrega-implantación.

c) Gestión de la configuración, comunicación, planificación, diseño-modelado, entrega-implantación.

Justificación:

“La ingeniería de software es la aplicación de un enfoque sistemático, disciplinado y cuantificable al desarrollo, operación y mantenimiento de software; es decir, la aplicación de la ingeniería al software” (Potts, 1993).

9. El modelo de proceso de software en espiral propuesto por Bohem
conjuga la naturaleza iterativa de la construcción de prototipos con los
aspectos controlados y sistemáticos del modelo lineal secuencial. La
etapa que no pertenece al modelo es:

- a) Evaluación del cliente.
- b) Comunicación con el cliente.
- c) Definición de un paradigma de desarrollo.

Justificación:

El modelo espiral es un "meta-modelo" que puede albergar dentro de sus iteraciones a otros paradigmas (como el cascada o el desarrollo iterativo). No define un paradigma único. Las otras dos opciones, "Evaluación del cliente" y "Comunicación con el cliente", son actividades centrales en cada ciclo de la espiral (Piattini, 2016).

10. Se construye un buen sistema de información considerando que el
punto de partida es:

- a) La definición de requisitos claros es parte del proceso, pero no es del todo importante.
- b) Utilizar un proceso definido con fases claras, donde cada una de estas genera un producto final.
- c) Utilizar herramientas de desarrollo como medio para alcanzar un producto de calidad

Justificación:

“La ingeniería de software es la aplicación de un enfoque sistemático, disciplinado y cuantificable al desarrollo, operación y mantenimiento de software; es decir, la aplicación de la ingeniería al software” (Potts, 1993).

Conclusiones

Origen y Formalización:

La Ingeniería de Software es una disciplina formal acuñada tras la Conferencia de la OTAN de 1968 para abordar los problemas de desarrollo. El término fue atribuido inicialmente a pioneras como Margaret Hamilton.

Definición del Software:

El software no es solo código; está compuesto por programas, procedimientos, documentación y datos relacionados. Su característica fundamental es que se desarrolla con intelecto y no se fabrica en el sentido clásico, y no se "desgasta" físicamente, sino que se deteriora debido a los cambios y el mantenimiento.

Calidad y Proceso:

Un buen software debe ser mantenible, confiable y fácil de utilizar. Para lograr estos atributos, el desarrollo debe ser sistemático y requiere utilizar un proceso definido con fases claras (como el modelo en Cascada o Espiral) para planificar, diseñar, construir y entregar productos de calidad.

Clasificación:

El software se clasifica en tipos según su propósito, distinguiéndose el software de sistemas (que sirve al propio ordenador) y el software de aplicación (que resuelve necesidades específicas del usuario o negocio, como las de gestión).

Recomendaciones:

Adoptar un Proceso de Ingeniería Formal: Utilizar un enfoque sistemático y cuantificable en todo el ciclo de vida del software, aplicando modelos como Cascada, Espiral o Iterativo para lograr productos confiables, eficientes y económicos.

Definición Rigurosa de Requisitos: Establecer con precisión las necesidades del cliente desde la fase inicial, evitando errores por requisitos mal definidos o cambiantes y garantizando una base sólida para el desarrollo.

Priorizar la Calidad del Software: Desarrollar software mantenible, confiable y fácil de usar mediante código claro, buena documentación e interfaces intuitivas, asegurando calidad y sostenibilidad a largo plazo.

Diferenciar los Tipos de Software: Reconocer las diferencias entre software de sistemas, de aplicación y de inteligencia artificial para aplicar metodologías y herramientas adecuadas según el propósito del proyecto.v

Referencias

Bauer, F. L. (1968). *Software Engineering. Germany: Nato Science Committee. Software Engineering. Germany: Nato Science Committee. Software Engineering. Germany: Nato Science Committee. Software Engineering. Germany: Nato Science Committee.*

Charette, R. (. (s.f.). *Why Software Fails [figura]. IEEE Spectrum for the Technology Insider:.* Why Software Fails [figura]. IEEE Spectrum for the Technology Insider:.

García, F. (s.f.). *Ingeniería del software, en Proyecto Docente e Investigador. GRIAL . Ingeniería del software, en Proyecto Docente e Investigador. GRIAL : . Ingeniería del software, en Proyecto Docente e Investigador. GRIAL . Ingeniería del software, en Proyecto Docente e Investigador. GRIAL : : <https://repositorio.grial.eu/items/81c58c5d-94d3-410c-b516-2a272c7c313a>*

Hinojosa, C. (2019). *Introducción a la ingeniería del software. Scribd. Introducción a la ingeniería del software. Scribd: . Introducción a la ingeniería del software. Scribd. Introducción a la ingeniería del software. Scribd: : <https://es.scribd.com/document/462507105/Introduccion-Ingenieria-de-Software>*

Piattini, M. (2016). *Evolución de la ingeniería del software y la formación de profesionales. Revista Institucional de la Facultad de Informática UNLP. Evolución de la ingeniería del software y la formación de profesionales. Revista Institucional de la Facultad de Informática. Evolución de la ingeniería del software y la formación de profesionales. Revista Institucional de la Facultad de Informática UNLP. Evolución de la ingeniería del software y la formación de profesionales. Revista Institucional de la Facultad de Informática:*

https://sedici.unlp.edu.ar/bitstream/handle/10915/57358/Documento_completo.pdf-PDFA.pdf?sequence=1&isAllowed=y

Potts, C. (1993). *Software-Engineering Research Revisited. IEEE Software*:. *Software-Engineering Research Revisited. IEEE Software*::. Software-Engineering Research Revisited. IEEE Software:. Software- Engineering Research Revisited. IEEE Software:: <https://ieeexplore.ieee.org/abstract/document/232392>