Unidad 1

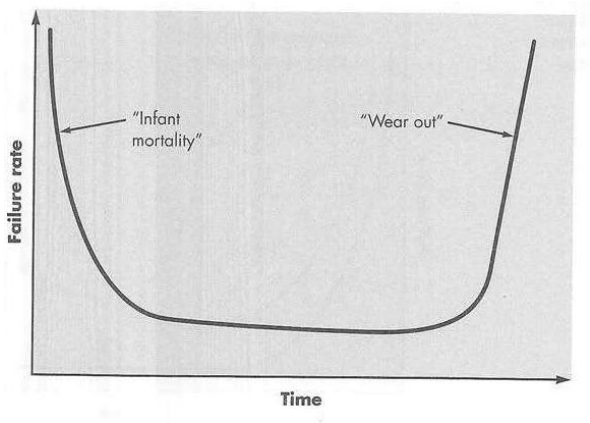
Guía Práctica N° 1 - Guía de preguntas de desarrollo teórico-práctico

Consignas:

- Conformar grupos en clase para trabajar sobre el cuestionario.
- Responder a las preguntas en forma grupal. Tomar apuntes de los puntos discutidos.
- Puesta en común: un integrante de cada grupo estará a cargo de presentar la respuesta del grupo.

Ejercicios:

1. El *software* es la característica que diferencia a muchos productos y sistemas informáticos. Dé ejemplos de dos o tres productos y de, al menos, un sistema en el que el software, no el hardware, sea el elemento diferenciador.
2. Analice y explique por qué el software se *desarrolla*, pero no se *manufactura*.
3. Explique porque el siguiente gráfico no aplica al software:



4. Muchos autores han tratado el impacto de la «era de la información». Dé varios ejemplos (positivos y negativos) que indiquen el impacto del software en nuestra sociedad.
5. De dos ejemplos de software para cada una de las categorías de 'Tipos de Software' vistos en clase (Pressman – Sommerville), e indique cuáles de ellos pueden clasificar para varias categorías al mismo tiempo.
6. Dada la clasificación de Dominios de aplicación propuesta por Pressman, compárela con los tipos de aplicación propuestas por Sommerville.
7. Un apartado especial lo tiene el denominado "Software Heredado" ("Legacy Software").
    a. ¿Qué es el software heredado?,
    b. ¿Porqué merece especial atención?,
    c. ¿Son frecuentes o escasos los casos en que se presentan en las organizaciones?

d. De ejemplos de Software Heredado.

8. A medida que el software se difunde más, los riesgos para el público general (debido a programas defectuosos) se convierten en una preocupación cada vez más significativa. Desarrolle un escenario realista del juicio final (distinto a 'Y2K'[1]) en donde el fallo de computadoras podría hacer un gran daño (económico o humano).
9. Analice las siguientes afirmaciones e indique si son ciertas o falsas. Justifique su respuesta:

   a) Si el cronograma va atrasado es posible contratar más programadores para terminar a tiempo.

   b) Cuanto más pronto se comience a escribir código, más se tardará en terminarlo.

10. Los *mitos del software* destacados en clase se están viniendo abajo lentamente a medida que pasan los años. Intente indicar qué elementos de la 'Ingeniería de Software' favorecen esta caída. ¿O estos permanecen totalmente vigentes?
11. Haciendo referencia a la distribución de costos del software**,** explique por qué es apropiado considerar que el software es más que programas que son ejecutados por los usuarios finales de un sistema.
12. ¿Cuáles son las diferencias entre el desarrollo de un producto de software genérico y el desarrollo de un software personalizado?
13. ¿Por qué razón Pressman considera que las Aplicaciones Web (Webapp) son diferentes al resto de aplicaciones?
14. Los métodos de la Ingeniería de software se empezaron a utilizar cuando la tecnología CASE estuvo disponible para apoyarlos. Mencione cinco tipos herramientas CASE que ayuden a aplicar alguno de estos métodos, indique además cuál es el método aplicado.
15. Según diversos estudios, sólo un promedio del 30% de los proyectos de desarrollo de software se completan en el tiempo y con el presupuesto planificado, con todas las funciones y características especificadas originalmente. ¿Cuáles son las cuestiones que hacen que la mayor parte de los proyectos no sean exitosos? Y ¿cuáles los que favorecen el éxito?
16. Pressman establece lo que considera la "Esencia de la Práctica de la Ingeniería de Software". ¿En qué consiste esta?. Analice el significado de cada aspecto considerado desde el punto de vista del software.
17. Explique, con sus palabras, el significado de los 'Principios Centrales' establecidos por Pressman:

    a. Principios que Guían el Proceso
    b. Principios que Guían la Práctica

18. De los ocho principios que guían el proceso, ¿cuál considera que es el más importante? Justifique.
19. De los ocho principios que guían la práctica, ¿cuál considera que es el más importante? Justifique.
20. ¿Por qué son importantes los modelos en el trabajo de ingeniería de software? ¿Son estos siempre necesarios?
21. Dado los casos de estudio adjuntos analice los éxitos y fracasos de la aplicación de la Ingeniería de Software en cada uno de estos:

    a. Ariane 5 (1996) (TP1-Caso1.doc)

    b. El Virtual Case File del FBI (2005) (TP1-Caso2.doc)

[1] http://histinf.blogs.upv.es/2012/12/18/el-efecto-2000/

c. Crisis de pasaportes en el Reino Unido (1999) (TP1-Caso3.doc)
d. El proyecto Júpiter (2001) (TP1-Caso4.doc)
e. Mariner 1 (1962) (TP1-Caso5.doc)
f. Explosión del gasoducto soviético (1982) (TP1-Caso6.doc)
g. Aeropuerto de Denver (1989) (TP1-Caso7.pdf)
h. London Ambulance Service (1990) (TP1-Caso8.pdf)
i. Misiles Patriot (1991) (TP1-Caso9.pdf)
j. MARS CLIMATE ORBITER (1999) (TP1-Caso10.pdf)
k. Error del milenio (2000) (TP1-Caso11.pdf)

Discutir:

a) ¿Cuál era el objetivo del proyecto de software?
b) Realice una reseña de los pasos que llevaron a que el proyecto fracasara.
c) ¿A quién o quiénes se les atribuye la culpa del fracaso del proyecto?
d) Según su análisis: ¿Cuáles fueron las principales causas del fracaso? ¿Se podrían haber evitado? ¿De qué manera?

22. Investigue casos de fallas de software importantes que se hayan producido en los últimos días, semanas, meses, o años. ¿Cuáles fueron sus efectos? ¿Cuáles sus causas?

Mitos del Software

Mito: *Tenemos un libro lleno de estándares y procedimientos para elaborar software. ¿No le dará a mi personal todo lo que necesita saber?*

Mito: *Si nos atrasamos, podemos agregar más programadores y ponernos al corriente (en ocasiones, a esto se le llama "concepto de la horda de mongoles").*

Mito: *Si decido subcontratar el proyecto de software a un tercero, puedo descansar y dejar que esa compañía lo elabore.*

Mito: *Para comenzar a escribir programas, es suficiente el enunciado general de los objetivos —podremos entrar en detalles más adelante.*

Mito: *Los requerimientos del software cambian continuamente, pero el cambio se asimila con facilidad debido a que el software es flexible.*

Mito: *Una vez que escribimos el programa y hacemos que funcione, nuestro trabajo ha terminado.*

Mito: *Hasta que no se haga "correr" el programa, no hay manera de evaluar su calidad.*

Mito: *El único producto del trabajo que se entrega en un proyecto exitoso es el programa que funciona.*

Mito: *La ingeniería de software hará que generemos documentación voluminosa e innecesaria, e invariablemente nos retrasará.*