

EXAMEN DISEÑO DE SOFTWARE MAYO 2014

1. Un caso de uso es:
 - La especificación formal de las pruebas de caja negra a las que se debe someter un sistema
 - Una interacción típica entre el sistema y los actores que constituyen su entorno
 - Cada una de las partes en las que se puede descomponer un escenario dado
 - Un elemento de la interfaz gráfica de usuario de un sistema informático
2. ¿Qué se entiende por ingeniería inversa?
 - Ciclo de vida que antepone el diseño de las pruebas a las tareas de diseño y codificación
 - Técnica de modelado para deducir clases a partir de los objetos previamente diseñados
 - Reconstrucción de un modelo a partir de código
 - Método de construcción de software basado en el paradigma orientado a objetos
3. ¿Cuál de las siguientes afirmaciones sobre clases y objetos es correcta?
 - Un objeto es cada uno de los representantes de una clase
 - Un objeto realiza las interfaces declaradas por una clase
 - Un objeto es una descripción de un conjunto de clases que comparten atributos, operaciones, relaciones y semántica
 - Clases y objetos son exactamente lo mismo
4. ¿Qué vínculo existe entre las responsabilidades de una clase y sus atributos y operaciones?
 - Las responsabilidades conectan a los atributos con operaciones
 - Atributos y operaciones son los medios de los que dispone para asumir sus responsabilidades
 - Las responsabilidades se deducen de los atributos y operaciones
 - El término responsabilidad es sinónimo de operación
5. Los diagramas de casos de uso recogen propiamente el siguiente tipo de requisitos:
 - Funcionales
 - De evolución
 - De interfaz
 - De calidad
6. Un actor deberá asociarse a un caso de uso:
 - Si se encuentra dentro del mismo paquete
 - Sólo si desencadena la ejecución del escenario principal
 - Sólo si es el receptor de la información producida por ese caso de uso
 - Si obtiene algún beneficio de la ejecución de ese caso de uso
7. ¿Qué tipos de mensajes pueden aparecer en un diagrama de interacción?
 - Eventos de disparo y acciones
 - Llamadas y señales
 - Llamadas, excepciones e interrupciones
 - Llamadas, señales, retornos, creaciones y destrucciones
8. En un diagrama de secuencia es obligatorio especificar:
 - Los focos de control
 - La numeración de los mensajes
 - Los retornos
 - Las líneas de vida

9. Lo que diferencia una agregación de una composición es que...
- La agregación es una asociación
 - La composición es una relación entre un todo y sus partes
 - En una composición las partes no se pueden compartir entre varios todos
 - En una agregación la vida de las partes está fuertemente ligada a la del todo
10. ¿Es admisible la relación entre los casos de uso *Devolver libro* y *Notificar disponibilidad* que figura en el Diagrama 1?
- Sí, representa que el primero provoca la ejecución del segundo
 - No, ya que la asociación entre casos de uso no está contemplada
 - Sí, indica que la información generada por el primero es de utilidad para la ejecución del segundo
 - No, porque no se admite el envío de señales entre casos de uso
11. Puede el actor *Profesor* del Diagrama 1 tomar parte en el caso de uso *Consultar préstamos*?
- Sí, por existir una relación indirecta a través del caso de uso *Renovar libro*
 - No, pero *Usuario* sí que puede participar en los casos de uso a los que se asocia *Profesor*
 - Sí, en atención al principio de sustitución
 - No, ya que no está asociado a él
12. ¿Cuál de las siguientes afirmaciones acerca de los patrones de diseño es falsa?
- Mejoran la documentación de los sistemas
 - Reducen significativamente la complejidad del sistema
 - Hacen a los sistemas más fáciles de ampliar y modificar
 - Habitualmente sirven para desacoplar diferentes partes de un modelo de forma que puedan evolucionar independientemente
13. ¿Cuál de las siguientes afirmaciones acerca de los patrones de diseño es falsa?
- Ayudan a identificar los objetos necesarios y a determinar su granularidad
 - Fomentan la programación para interfaces y el uso equilibrado de mecanismos de reutilización
 - Permiten detectar los cuellos de botella que impiden la optimización del rendimiento del sistema
 - Contribuyen a lograr un diseño flexible preparado para futuros cambios
14. Para que el paquete A pueda acceder al contenido del paquete B...
- B tendrá que importar a A
 - Necesariamente habrá que anidar ambos paquetes
 - A importará a B y éste exportará elementos estableciendo sus visibilidades adecuadamente
 - Bastará con especificar una dependencia <<import>> desde A hasta B
15. En el Diagrama 2, ¿son necesarios en *Aeronave* los atributos *piloto* y *posiciones*?
- Sí, si lo que se pretende es hacerla responsable de su piloto y las posiciones por las que navega
 - Sí, pero claramente deberían estar en el elemento que las generaliza
 - No, porque probablemente se estaría introduciendo redundancia en el modelo
 - Sí, en caso de que se haya previsto hacer ingeniería directa
16. ¿Puede *Recurso* (ver Diagrama 2) proporcionar un método para la operación *sondearPosicion()*?
- No sólo puede sino que debe
 - No
 - Sí, por tratarse de una operación final
 - No, ya que esta clase es abstracta

17. ¿Qué mecanismo es el apropiado para *factorizar* el comportamiento común a varios casos de uso?
- Realización
 - Extensión
 - Generalización
 - **Inclusión**
18. Dado que los patrones de diseño describen una estructura y un comportamiento, en UML se representarán como...
- Escenarios
 - **Colaboraciones**
 - Interacciones
 - Secuencias
19. Un estereotipo en UML es un:
- Mecanismo de extensibilidad para añadir nuevas propiedades a un elemento $p(\text{¿...?})$
 - **Mecanismo de extensibilidad que permite crear nuevos bloques de construcción(¿...?)**
 - Patrón de diseño que ilustra cómo crear instancias de una clase a partir de $u(\text{¿...?})$
 - Mecanismo de extensibilidad que permite añadir nueva semántica o $m(\text{¿...?})$
20. ¿Qué es un patrón de arquitectura?
- Un diagrama de clases de alto nivel que muestra la línea base de la arquitectura de un sistema
 - **Un paquete de patrones de diseño**
 - Lo mismo que un patrón de diseño
 - El diagrama de despliegue que muestra la topología de cualquier sistema software
21. En una máquina de estados, para cada transición es opcional especificar:
- La condición de guarda
 - La condición de guarda y la acción desencadenada
 - El evento de disparo, si el estado de origen no es ocioso
 - **El evento de disparo, la guarda y la acción**
22. ¿Cuál de los siguientes no es un uso previsto para los diagramas de clases?
- Modelado de vocabulario de un dominio de aplicación
 - **Modelado de un flujo de trabajo ("workflow")**
 - Modelado de la parte estructural de una colaboración
 - Modelado del esquema conceptual de una base de datos
23. En UML es admisible que una clase...
- **Especialice a varias clases y realice varias interfaces**
 - Especialice a varias clases y realice una sola interfaz
 - Especialice a una sola clase y realice varias interfaces
 - Especialice a una sola clase y realice una sola interfaz
24. ¿Qué relación existe entre los diagramas de secuencia y los diagramas de colaboración?
- Los diagramas de secuencia muestran el comportamiento de una estructura de objetos especificada por medio de un diagrama de colaboración
 - Ambos representan la interacción entre una comunidad de objetos, pero los diagramas de secuencia hacen explícitos los enlaces entre objetos
 - Son idénticos salvo que los diagramas de colaboración representan instancias de componentes
 - **Ambos representan la interacción entre una comunidad de objetos, pero los diagramas de secuencia explicitan el transcurso del tiempo**

25. En el contexto de los diagramas de secuencia, ¿en qué se distingue una llamada del envío de una señal?
- Las llamadas son síncronas, mientras que las señales son asíncronas
 - Las llamadas no tienen valores de retorno
 - Las llamadas son asíncronas, mientras que las señales son síncronas
 - Las señales no admiten el paso de parámetros
26. ¿Para qué sirven las barras de sincronización en los diagramas de actividades?
- Para visualizar comportamiento opcional
 - Para capturar eventos procedentes del exterior
 - Para representar la repetición de acciones que forman parte de una actividad
 - Para mostrar ejecución concurrente
27. ¿Qué condiciones debe reunir un componente para sustituir a otro?
- El nuevo tiene que estereotiparse igual que el original
 - Tiene que incluir los mismos métodos que el componente original
 - Necesita definir atributos adecuados para recoger los mismos cambios de estado
 - Tiene que dar soporte a las mismas interfaces
28. ¿Cómo se refleja en un diagrama de clases que una clase sea responsable de crear instancias de otra?
- Usando una dependencia
 - No se puede hacer, habrá que usar un diagrama de interacción
 - Se puede mostrar en una nota con pseudocódigo
 - Mediante una dependencia estereotipada <<instantiate>>
29. ¿Qué patrón de creación está pensado para crear familias de objetos compatibles entre sí?
- Abstract Factory
 - Singleton
 - DAO
 - Factory Method
30. Dado el supuesto: cada estudiante de la ETSE ha de ser tutelado por un profesor: algunos profesores son tutores de varios estudiantes y otros no lo son de ninguno. ¿Qué diagrama representa esta relación con mayor fidelidad?
- El Diagrama 3a
 - El Diagrama 3b
 - El Diagrama 3c
 - El Diagrama 3d
31. Un extremo navegable en una asociación entre dos clases indica...
- Que la responsabilidad de mantener referencias a los objetos de la clase en ese extremo recaerá sobre las instancias de la clase ubicada en el extremo opuesto
 - El sentido en el que fluye la información entre objetos en tiempo de ejecución
 - Las multiplicidades con las que participa cada una de las dos clases
 - El sentido de lectura de la etiqueta de la asociación
32. Dado el Diagrama 4, ¿puede en el Diagrama 5 enviar o1 el estímulo *setPabellon* al objeto *Partido*?
- No, porque corresponde a *:Competicion* iniciar la interacción en calidad de primer participante
 - Sí, utilizando el enlace que la asociación entre las dos clases implicadas posibilita
 - No, porque la operación no forma parte de la interfaz *IObservador*
 - No, porque la clase *Partido* no declara esa operación

33. En el Diagrama 5, ¿tiene sentido que la instancia de competición envíe los dos estímulos finales?
- No, si atendemos a las visibilidades especificadas para los roles en el Diagrama 4
 - Sí, puesto que los objetos o1 y o2 tienen alcance global
 - No, porque la comunicación tendría que tener lugar a través de la interfaz *IObservador*
 - Sí, quién asuma esa responsabilidad es totalmente irrelevante
34. El siguiente patrón de diseño enseña cómo manejar jerarquías de objetos tratando uniformemente a objetos simples y sus agrupaciones:
- Facade
 - Proxy
 - Composite
 - Decorator
35. Dadas las clases y los objetos respectivamente representados en los diagramas 6a y 6b, la secuencia que refleja lo que sucede en caso de que la instancia de la clase *GestorVentanas* le envíe un mensaje *dibujar()* al objeto al que se encuentra enlazada es la mostrada en el...
- Diagrama 6c
 - Diagrama 6d
 - Diagrama 6e
 - Diagrama 6f
36. Dada la similitud entre Adapter y Bridge, la diferencia esencial entre ambos patrones radica en que:
- Adapter es un patrón de comportamiento
 - Bridge hace que las cosas funcionen antes de ser diseñadas y Adapter lo hace después
 - Adapter hace que las cosas funcionen antes de ser diseñadas y Bridge lo hace después
 - Bridge es un patrón estructural
37. ¿Qué representa un nodo en un diagrama de despliegue?
- Un recurso físico dotado habitualmente de capacidad de cómputo y de almacenamiento
 - Un mecanismo de propósito general para agrupación lógica de elementos de modelado
 - Una parte reemplazable de un sistema que se ajusta a y proporciona la realización de un conjunto de interfaces
 - Una colección de operaciones que define un determinado servicio
38. ¿Qué patrón ayuda a implementar un mecanismo de suscripción/notificación?
- Chain of Responsibility
 - Mediator
 - Observer
 - Strategy
39. Dadas las clases y los objetos de los Diagramas 7a y 7b, la interacción que posiblemente se desencadene si la instancia de *Cliente* envía la petición *retirar(1500, cc)* al objeto c1 figura en el...
- Diagrama 7c
 - Diagrama 7d
 - Diagrama 7e
 - Diagrama 7f
40. El patrón de comportamiento que encapsula algoritmos en objetos es...
- Strategy
 - Command
 - State
 - Template Method





