**En negrita están marcadas las soluciones

**Del examen de 2015 solo he puesto las del segundo parcial por que las del primero están repetidas

Examen DSS

- junio 2010 –

1. El objetivo del patrón fachada es…

    a. Protegernos ante posibles variaciones en el sistema y obtener mas acoplamiento

    b. Simplificar el acceso a la interfaz mediante funcionalidades.

    c. Resolver los problemas de compatibilidad entre las interfaces.

    d. Todas son ciertas.

2 . RUP NO ENTRA

3. RUP NO ENTRA

4. El patrón command permite:

    a. Acoplar el emisor y receptor

    b. Manipular comandos como objetos

    c. Añadir nuevos comandos modificando a otras clases.

    d. Combinarse con el patrón bridge para obtener comandos compuestos

5. Según UML, los casos de usos:

    a. Están condicionados por la implementación del sistema

    b. Son independientes de la computación.

    c. Son independientes del sistema.

    d. Ninguna de las anteriores

6. La arquitectura establece:

    a. Division en capas del sistema en base a las necesidades de distribución

    b. Configuración de componentes proporcionando una vista estructural a grandes ragos de la aplicación así como de la ubicación de la funcionalidad.

    c. a y b son verdades.

    d. Ninguna de las anteriores

7. Las entidades de negocio ¿¿?¿?¿?¿?¿?¿?¿

a. Exponen los métodos de mayor nivel que la lógica de negocio expone al cliente

b. Suelen contener la información de una clase de dominio con sus atributos operaciones y restricciones

c. Persisten y recuperan los datos de dominio de la aplicación

d. Se responsabilizan de los métodos CRUD y exponen esta funcionalidad a las capas superiores de la aplicación

13. Si necesitamos proporcionar una interfaz para crear familias de objetos relacionados dependientes, sin especificar sus clases concretas. ¿ Qué patrón debemos aplicar?

a. Patrón State

b. Patrón Composite

c. Patrón Bridge.

d. Patrón Abstract Factory

14. De los siguientes patrones, indica cuál de ellos es Creacional

a. El patrón state

b. El patrón Bridge.

c. El patrón Singleton

d. El patrón Proxy

17. ¿Cuál de las siguientes sentencias describe mejor cuándo utilizar el patrón State?

a. El patrón puede ser utilizado cuando una clase tiene muchos estados.

b. El patrón puede ser utilizado cuando un objeto parece cambiar de clase en tiempo de ejecución

c. El patrón puede ser utilizado cuando cada estado tiene atributos relevantes a un objeto de contexto específico.

d. Ninguna de las anteriores

- junio 2013 –

8. ¿Cuál de las siguientes afirmaciones es falsa?

a. Los patrones son mas abstractos y generales que los frameworks

b. Un patrón puede incorporar un framework y un framework puede hacer uso de varios patrones

c. un Patron es una descripción de cómo se puede resolver un problema, y un framework da soporte a una posible solución

9. ¿ Cual de los siguientes grupos de patrones incluye sólo patrones de comportamiento?

a. Command, Composite, Builder

b. Command, Proxy, Strategy

c. Strategy, Observer, Command

10. Uno de los principales inconvenientes del patrón Builder es:

a. Aumenta excesivamente el número de clases del sistema

b. No permite crear distintas variantes de un producto

c. Existe un alto acoplamiento entre el cliente y el constructor del producto

11. ¿Cuál de los siguientes grupos de patrones incluye solo patrones estructurales?

a. Composite, Builder, Proxy

b. Façade, Composite, Proxy

c. Proxy, Factory Method, Command

12. ¿ Qué patrón permitiría crear colas de prioridad o delegar la ejecución de distintas operaciones?

a. Strategy

b. Command

c. Proxy

13. ¿Qué patrón seria mejor aplicar cuando no sabemos la clase de los objetos que van a crear las subclases de una clase abstracta?

a. Factory Method

b. Abstract Factory

c. Builder

14. El patrón Command hace uso, entre otros, de los siguientes patrones GRASP

a. Polimorfismo, protección de variaciones y creador

b. Indirección, frabricación pura y polimorfismo

c. Creador, Experto y delegación

15. Cuando necesitamos trabajar con distintos tipos de recursos que dependen de la localización debemos pensar en la conveniencia de usar el patrón

a. Proxy

b. Abstract Factory

c. Composite

16. ¿Cuál es el principal inconveniente de la fabricación pura?

a. Si se abusa de ello, aparecen clases con un único método(clases “functoides”) dando lugar a un diseño centrado en funciones que se implementan sobre un diseño orientado a objetos?¿?¿?¿?¿?¿

b. El problema es que las clases ficticias (es decir, clases que no existen en el análisis original) añadidas en la fabricación pura solo pueden contener un método.

c. Mejora la cohesión pero empeora el acoplamiento

17. NADA

18. Hemos implementado un sistema para la agencia F de noticias de manera que la agencia puede informar inmediatamente, cuando ocurre cualquier evento, a cualquiera de sus clientes por Email, SMS, Twitter o cualquier otro canal que puedan requerir en un futuro ¿ Con qué patrón lo hemos implementado?

a. Strategy

b. Obrserver

c. Command

30. Un componente de entidad de negocio:

a. Representa a las entidades de dominio localizadas en la capa lógica de negocio

b. Representa a las entidades de dominio que pueden almacenar y transitar por las diferentes capas

c. Representa a las entidades de negocio pero conteniendo solamente su estado

- junio 2014 -

1. Según el patrón GRASP Creador, la clase A debe ser la encargada de crear una instancia de la clase B si:
   a. **A contiene o agrega instancias de B**
   b. B usa instancias de A
   c. Las dos son ciertas

2. En una arquitectura de 3 capas, ¿Cómo se relacionan las capas de dominio y presentación?
   a. **La capa de dominio nunca debe depender de la capa de presentación**
   b. La capa de presentación nunca debe depender de la capa de dominio
   c. Nunca deben estar relacionadas

3. ¿Qué patrón GOF permite crear estructuras jerárquicas de objetos, de forma que el cliente pueda manejar objetos simples o compuestos indistintamente?
   a. Builder
   b. Strategy
   c. **Composite**

4. Cuando se usa el patrón Active Record, ¿Dónde se sitúa el código de acceso a la base de datos?
   a. En el controlador del sistema
   b. En clases especializadas de acceso a datos
   c. **En los objetos de dominio**

5. ¿Cuál es la diferencia entre los patrones de diseño y los frameworks?
   a. **Los frameworks se basan en uno o varios patrones para solucionar problemas concretos**
   b. Los patrones usan frameworks para ofrecer soluciones reutilizables
   c. Ninguna, los frameworks se usan para el diseño detallado de un sistema

6. En una arquitectura de capas, ¿Qué capas se pueden ver afectadas por una nueva funcionalidad?
    a. Solamente la capa de dominio
    b. Solamente la capa de presentación
    c. Puede haber varias capas afectadas

7. El patrón GOF Abstract Factory:
    a. Provee un interfaz para crear familias de productos de forma consistente
    b. Permite controlar el número de instancias de los objetos que se crearan
    c. Disminuye notablemente el número de clases del sistema

8. ¿Cuál de las siguientes afirmaciones es CIERTA? Para que un diseño sea fácil de mantener y usar:
    a. Debemos mantener un bajo acoplamiento y una baja cohesión
    b. Debemos mantener un alto acoplamiento y una baja cohesión
    c. Debemos mantener un bajo acoplamiento y una alta cohesión

9. ¿Cuál de las siguientes afirmaciones sobre el patrón Remote Façade es FALSA?
    a. Las fachadas remotas no deben contener lógica de dominio
    b. Las fachadas remotas no deben tener estado
    c. Las fachadas remotas deben ofrecer un único método para acceder a todas las propiedades de un objeto

10. El uso de interfaces de grano fino mejora el rendimiento de los sistemas distribuidos:
    a. Verdadero
    b. Solo si el diseño es orientado a objetos
    c. Falso

11. El patrón GRASP Indirección implica:
    a. Relacionar dos clases mediante una clase intermedia
    b. Favorecer las relaciones directas entre clases, evitando caminos más largos
    c. Asignar menos responsabilidades a las clases que hay en la frontera con otras capas

12. ¿Cuál de las siguientes responsabilidades NO corresponde a la capa de presentación?
    a. **Comunicarse con las capas superiores para ofrecer funcionalidades complejas**
    b. Manejar la interacción y mostrar información al usuario
    c. Comunicarse con la capa de lógica de dominio para transmitir los cambios realizados por el usuario

13. Los objetos Data Transfer Object:
    a. Solamente pueden contener datos de un objeto de dominio para garantizar la consistencia
    b. **Pueden contener datos de varios objetos de dominio para mejorar el rendimiento**
    c. No deben contener datos de los objetos de dominio para aumentar la cohesión

14. ¿Qué mide el rendimiento (performance) de un sistema?
    a. **El tiempo de respuesta**
    b. El tiempo que el sistema está funcionando
    c. El tiempo que el sistema funciona sin fallos

15. Cuando se desea simplificar el acceso a un subsistema o capa ¿Qué patrón GOF es el más indicado?
    a. Strategy
    b. Proxy
    c. **Façade**

16. ¿Con que patrón podemos mantener la consistencia cuando se modifican dos instancias de un mismo objeto desde distintas partes del sistema?
    a. **Unit of Work**
    b. Identity Map
    c. Lazy Load

17. ¿A que nos referimos normalmente cuando hablamos de la arquitectura de un sistema?
    a. A la infraestructura de hardware que dará soporte al sistema
    b. **A los principales componentes del sistema y sus relaciones**
    c. A la estructura de bajo nivel de cada componente del sistema

18. ¿con que patrón de lógica de dominio se combina normalmente el patrón Row Data Gateway?
    a. Table Data Gateway
    b. Domain Model
    c. Transaction Script

19. ¿En qué capa se deben situar las operaciones complejas de la aplicación?
    a. En la capa de dominio
    b. En la capa de servicio
    c. En la capa de presentación

20. El acoplamiento entre clases es una medida de:
    a. El grado de dependencia entre las clases del sistema
    b. Cuántas funcionalidades distintas se asigna a cada clase
    c. Cuántas clases intermedias hay que recorrer para llegar de una clase A a otra B

21. ¿Cuándo es conveniente usar el patrón Data Mapper?
    a. Cuando el modelo de dominio es sencillo
    b. Cuando el modelo de dominio es complejo
    c. Cuando se usa una base de datos orientada a objetos

22. ¿Cuál de los siguientes patrones no se aplica a la capa de dominio?
    a. Transaction Script
    b. Table Module
    c. Table Data Gateway

23. ¿Cuál es la diferencia entre los patrones Table Module y Domain Model?
    a. Table Module pertenece a la capa de acceso a datos, y Domain Model a la capa de lógica de dominio
    b. Normalmente, con Table Module hay un objeto por cada tabla, mientras que con Domain Model hay un objeto por cada fila de la tabla
    c. Las dos son ciertas

24. Las clases de diseño que surgen como resultado del patrón Fabricación Pura suelen aparecer
    a. Para representar objetos de dominio
    b. Por descomposición funcional, para dividir responsabilidades
    c. Por la aparición de jerarquías de herencia

25. ¿Cuál es la principal ventaja de usar patrones GOF a la hora de diseñar un sistema?
    a. Permite disminuir el número de clases del sistema
    b. Hace que el sistema sea más fácil de comprender, a cambio de disminuir las posibilidades de reutilización de código
    c. Aumenta la flexibilidad del sistema frente a futuros cambios

26. Al usar el patrón Transaction Script:
    a. Cada procedimiento creado es independiente del resto de capas del sistema
    b. Surgen dos tipos de procedimientos: los que acceden a la base de datos y los que se comunican con el resto de capas del sistema
    c. Cada procedimiento representa una acción que el usuario puede ejecutar

27. En el patrón Model-View-Controller, cuando hay una serie de operaciones comunes que se deben realizar para cada petición del usuario, es conveniente usar el patrón:
    a. Page Controller
    b. Front Controller
    c. Application Controller

28. ¿Cuáles son las capas típicas de un sistema de 3 capas?
    a. Acceso a datos, servicio y lógica de dominio
    b. Servicio, lógica de dominio y presentación
    c. Acceso a datos, lógica de dominio y presentación

29. ¿Qué problema pretende evitar el patrón Lazy Load?
    a. Problemas de integridad referencial al cargar objetos relacionados
    b. Problemas de rendimiento al cargar objetos relacionados
    c. Problemas por el uso excesivo de herencia en la lógica dominio

30. En el patrón GOF Observer, ¿Cuál es la clase encargada de notificar un suceso en el sistema?
    a. La clase que desempeña el rol Observer
    b. La clase que desempeña el rol Concrete Observer
    c. La clase que desempeña el rol Subject

- Julio 2015 –

16. Para que un diseño de clases sea más fácil de entender y usar, la cohesión debe ser
    a. Baja
    b. Alta
    c. La cohesión no influye

17. ¿Qué patrón está pensado para poder intercambiar distintos comportamientos en tiempo de ejecución?
    a. Strategy
    b. Proxy
    c. Builder

18. ¿Cuál de los siguientes patrones NO es de comportamiento?
    a. Observer
    b. Proxy
    c. Command

19. El uso del patrón GRASP Creador implica
    a. Disminuye el acoplamiento entre clases
    b. Aumenta el acoplamiento entre clases
    c. No afecta al acoplamiento entre clases

20. ¿Qué inconveniente tiene el uso del patrón Composite?
    a. El cliente no distingue entre clases simples y compuestas
    b. No permite añadir nuevas clases simples
    c. Resulta complicado imponer restricciones sobre la estructura de los objetos compuestos

21. ¿Qué inconveniente tiene el uso del patrón Abstract Factory?
    a. Los productos de distintas familias comparten el mismo interfaz
    b. No es fácil añadir nuevos productos
    c. Para el cliente no es fácil seleccionar la familia de productos a crear

22. ¿Con que patrón podemos reducir el acoplamiento entre dos partes de un sistema, cuando una parte usa un conjunto de clases de la otra?
    a. Composite
    b. Façade
    c. Proxy

23. ¿Qué ventaja tiene el patrón Builder?
    a. El director no necesita conocer los pasos del proceso en construcción
    b. Cada constructor tiene un interfaz distinto
    c. Permite crear distintos productos siguiendo el mismo proceso

24. ¿Cuál es la principal motivación de los patrones GRASP?
    a. Crear modelos que no cambiaran a lo largo del proyecto
    b. Proteger al sistema frente a posibles variaciones
    c. Identificar las clases del modelo de dominio que se comunican con el resto del sistema

25. En el patrón Command, ¿Qué información necesitan los comandos para ejecutarse?
    a. Qué clase ha instanciado la acción
    b. Qué clase ha invocado la acción
    c. Qué clase es la receptora de la acción

26. ¿Qué patrón sería más adecuado para llevar un recuento del número de veces que se llama a cada método de un objeto?
    a. Strategy
    b. Observer
    c. Proxy

27. En el patrón Observer, ¿Qué rol es el encargado de llevar un registro de los objetos que deben ser notificados cuando hay un cambio?
    a. Subject
    b. Observer
    c. Client

28. ¿Cuál de los siguientes tipos de asociación implica un menor grado de acoplamiento?
    a. Uso
    b. Herencia
    c. Implementación de interfaces?¿?¿?¿?

29. El patrón GRASP Indirección implica
    a. Relacionar dos clases mediante otra intermedia
    b. Favorecer las relaciones directas entre clases
    c. Asignar menos responsabilidades a las clases que hay en la frontera con otras capas

30. Para disminuir la dependencia entre clases y favorecer la reutilización de código el acoplamiento debe de ser
    a. Lo más bajo posible
    b. Lo más alto posible
    c. El acoplamiento no influye

-Junio 2015-

1. GRASP es el acrónimo de 'patrones generales de software para la asignación de responsabilidades'. Con respecto a dichas responsabilidades, ¿cuál de las siguientes afirmaciones es falsa?
    a. Los métodos se implementas para cubrir las responsabilidades.
    b. Cada responsabilidad se traduce en un método que se debe asignar a alguna clase software
    c. Los métodos pueden colaborar con otros métodos u objetos para cubrir una determinada responsabilidad

2. ¿Cuál de las siguientes afirmaciones relacionadas con la herencia es verdadera?
    a. La herencia escala bien cuando aumenta el número de variaciones del sistema
    b. Siempre que tenemos una jerarquía de herencia, es posible sustituirla por una agregación o composición(delegación), lo cual aumenta la flexibilidad del sistema
    c. La herencia mejora el acoplamiento del sistema

3. ¿cuál de los siguientes efectos se puede deber al alto acoplamiento entre clases?
    a. Mayor cohesión de las clases fuertemente acopladas
    b. Mayor facilidad de reúso
    c. Alta probabilidad de propagación inadvertida de errores ante modificaciones en el código

4. ¿Qué beneficio obtenemos con el patrón GRASP creador?
    a. Reducir cohesión
    b. Reducir acoplamiento
    c. Evitar la necesidad de inicializar la clase creada

5. ¿Cuál es la principal motivación de los patrones GRASP?
    a. Proteger el sistema de posibles variaciones
    b. Elaborar modelos que no cambien a lo largo del proyecto
    c. Identificar las clases del modelo de dominio que se comunican con el resto del sistema

6. ¿Con qué patrón GOF se relaciona el patrón GRASP controlador?
    a. Fachada
    b. Fabricación pura
    c. Factory method

7. ¿Cuáles de estas clases es mas probable que requieran la aplicación de un patrón SINGLETON?
    a. Los comandos de un patrón COMMAND
    b. Los componentes de un patrón COMPOSITE
    c. Las factorías concretas de un patrón ABSTRACT FACTORY

8. Hemos diseñado un sistema de venta de vehículos y para la representación de las empresas cliente (que pueden tener filiales, subfiliales, etc) hemos aplicado el patrón COMPOSITE. De este modo hemos permitido que el resto del sistema interaccione con las empresas sin tener que preocuparse de si es una empresa simple o si tiene filiales que hay que tener en cuenta para operaciones como realizar ofertas, calcular costes, etc. ¿Cuál de las siguientes afirmaciones NO es un inconveniente de la solución aportada?
    a. Gestión de jerarquías ortogonales de herencia y agregación.
    b. Gestión de restricción de los tipos de hijos en las composiciones.
    c. Gestión de los tipos de operación permitidos en los objetos simples.

9. Una empresa que ofrece servicios remotos (de momento 3: S1, S2, S3) nos ha encargado implementar una pequeña aplicación que permita gestionar el cobro de servicios de manera que puedan cobrar a los clientes en función de las llamadas que realicen a dichos servicios. El precio de cada servicio es fijo (S1 vale 100, S2 vale 200 y S3 vale 300). La solución debe incluir una clase llamada ServicioCobro. ¿Cuál sería el patrón más adecuado para resolver este problema?
    a. Command
    b. Façade
    c. Strategy

10. ¿Cuál sería el rol del patrón aplicado en la pregunta anterior que ostentaría la clase ServicioCobro?
    a. Façade
    b. Strategy (abstract)
    c. Invoker

11. Queremos diseñar un sistema de manejo de documentos para una empresa que se dedica a la licitación de proyectos públicos. Este sistema maneja documentos de tipo texto, Word, diagramas de Visio y documentos legales. Aunque son documentos diferentes, todos ellos tienen un titulo, un tipo, un tamaño, una localización, un numero de paginas, etc. Esta similitudes nos sugieren diseñar el sistema de manera que se puedan tratar todos estos documentos de la misma manera (a través de una interfaz común IDcoumento). ¿Qué patrón GOF aplicarías para solucionar este problema?
    a. Factory Method
    b. Abstract Factory
    c. Builder

12. Hemos implementado un sistema al que queremos añadir ahora la posibilidad de leer una series de datos de configuración desde distintas fuentes, como p.ej. ficheros XML, bases de datos, etc. ¿Qué patrón GOF se adapta mejor a este problema?
    a. Command
    b. Strategy
    c. Façade

13. En el patrón OBSERVER…
    a. La lista de observadores concretos puede incluir objetos de distinto tipo
    b. El sujeto puede ser que envíe la información necesaria al observador mediante el método update()
    c. Todas son ciertas

14. Nos han pedido diseñar un sistema de aprendizaje. En el, vamos a tener distintos objetos de aprendizaje que se deben instanciar en función del nivel de experiencia del alumno (de momento tres: K12, Adulto, Profesional). ¿Qué patrón aplicarías para solucionar este problema de manera flexible y extensible a nuevos niveles de experiencia (e.g. alumnos universitarios)?
    a. Builder
    b. Abstract Factory
    c. Proxy

15. En el sistema de aprendizaje uno de los requisitos es la construcción de guías docentes. Estas guías están compuestas de un contexto, una serie de objetivos, contenidos, una descripción de la metodología docente, un cronograma y unos criterios de evaluación. ¿Qué patrón GOF es el mas adecuado para diseñar esta construcción?
    a. Builder
    b. Factory Method
    c. Abstract Factory

16. La clase fachada del patrón fachada…
    a. Se limita a transmitir invocaciones desde el exterior hacia los objetos del sistema que oculta, preservando la interfaz de estos últimos.
    b. Realiza una adaptación entre su interfaz y la interfaz de los objetos del sistema que ésta oculta.
    c. Es una clase abstracta de la que heredan las fachadas concretas de cada componente interno del sistema.

17. El patrón BUILDER se utiliza para…
    a. Permitir la variación de la representación interna de un producto.
    b. Simplificar el código cliente que crea objetos complejos.
    c. Todas son ciertas

18. En el patrón Observer…
    a. La clase Subject debe tener acceso de algún modo a los datos del Observer.
    b. La clase Observer debe tener acceso de algún modelo a los datos del Subject.
    c. No existe acoplamiento entre las clases Subject y Observer.

19. Una tienda de cosméticos nos ha pedido que le realicemos una web responsive para verder sus productos. Con el fin de dar un servicio añadido al cliente, cada producto tiene asociada tanto una descripción y una foto como un pequeño vídeo donde se muestra cómo se debe usar y algún consejo de belleza. Sin embargo, y debido a que muchos de sus clientes se conectan desde el móvil, quieren que el video se reproduzca sólo si el cliente hace click sobre la fotografía del producto. Para ello, hemos decidido aplicar el patrón PROXY ¿De que tipo de proxy estamos hablando?
    a. Proxy virtual

b. Proxy de protección
c. Proxy remoto

20. El patrón Abstract Factory contribuye a preservar el principio OPEN-CLOSED de Bertrand Meyer (diseños abiertos a la extensión y cerrados al cambio) cuando:
    a. El punto más probable de variación supone la adición de nuevas factorías concretas.
    b. El punto más probable de variación supone la adición de nuevas familias de producto.
    c. Los puntos más probables de variación suponen (a) la adición de nuevas familias de producto y/o (b) la adición de nuevas factorías concretas.