TEMA 1

1. Seleccionar la única respuesta incorrecta a la siguiente cuestión sobre los denominados sistemas software abiertos:

A. Son concurrentes
B. Son orientados a objetos
C. Su funcionalidad se puede extender de forma independiente a otros
D. No pueden definir un estado global externamente visible
E. Han de poder dar de baja/alta a sus componentes de forma dinámica

2. Seleccionar la única respuesta correcta sobre diferencias entre patrones de diseño y marcos de trabajo para componentes (component frameworks) en sistemas software:

A. Los patrones de diseño son menos abstractos que los marcos de trabajo
B. Un marco de trabajo es un conjunto de clases que se puede modificar para ser adaptado a las necesidades concretas de una aplicación
C. Los marcos de trabajo solo consisten en una colección de componentes software
D. Un patrón puede estar compuesto por varios marcos de trabajo.
E. Un marco de trabajo está formado por uno o más objetos y clases, que no pueden pertenecer a más de uno de estos marcos de trabajo.

3. Indicar cuál de las siguientes características no es propia de un componente software:

A. Se compone con otros componentes en la fase más tardía posible de implantación y despliegue del software
B. Está pensado para propiciar la reutilización del software mediante acuerdos a nivel de servicios o contratos con los clientes
C. Ha de poder modificar su propio estado con el uso de mecanismos denominado reflexión.
D. Se trata de una entidad polimórfica que reacciona de forma diferente si se despliega en distintos contextos
E. Se puede reemplazar por otros componentes cuya interfaz cumpla con el contrato

4. Un componente software puede contener objetos que se crean un su interior y son visibles a los clientes del componente ¿No viola esto la propiedad que dice que los componentes no pueden contener un estado que sea observable desde el exterior del componente?

> No, una cosa es que los clientes no puedan ver el estado y en función de eso modificarlo para que el componente se comporte de una forma distinta según ese estado y otra que no se puedan crear objetos dentro del componente accesibles por los clientes. Un componente, a través de la interfaz por contrato que define, puede ofrecer a sus clientes la funcionalidad que sea, entre ellas proporcionar un conjunto de elementos, pensemos en un diccionario.

5. Sobre el concepto de “Independently extensible system”. Indicar la única respuesta incorrecta sobre la definición de un sistema software basado en componentes

A. Se trata de sistemas que pueden manejar la adición tardía de componentes sin que se necesite realizar una comprobación de la integridad global del sistema
B. Un componente software es una unidad de despliegue de software independiente y que carece de estado
C. Estas aplicaciones están formadas por componentes estables, desarrollados por terceras partes, y de scripts que los conectan entre sí

D. Se pueden hacer copias distinguibles de un mismo componente para ser instaladas en diferentes lugares o momentos del sistema
E. Las restricciones de comportamiento (behavioral constraints) de los objetos no pueden representarse solo con la información (invariantes, pre y postcondiciones) en las interfaces de dichos componentes

6. Explicar porque un componente software, que puede tener estructura interna (múltiples clases, objetos, etc…), no puede desplegar sólo parte de su estructura, sino que ha de desplegarse como una sola unidad

Un componente ha de poder desplegarse independientemente y ser dinámicamente extensible, para lo cual ha de estar bien diferenciado de su entorno y de los otros componentes del sistema. Por tanto, un componente encapsula todas las características constitutivas que lo definen y nunca se puede desplegar parcialmente.

7. Indicar la única respuesta incorrecta sobre la definición de las siguientes entidades-software:

A. Para construir una clase, utilizando lenguajes de POO, podemos heredar la implementación de otras clases, pero no se deben instanciar separadamente una clase base y sus clases hijas
B. La diferencia entre módulo y paquete (package de Java) consiste en el primero puede contener elementos no orientados a objetos y un paquete sólo puede tener clases
C. Un recurso es una colección de entidades (clases, colecciones, variables globales, etc…) que no se pueden modificar y poseen tipo
D. Los subsistemas en UML son subclases de Classifier y Package, por tanto, poseen el comportamiento propio y diferenciado de una parte del sistema
E. Los objetos son entidades de los lenguajes de programación orientados a objetos que se instancia de una sola vez (su estado interno no puede ser fraccionado)

8. Indicar la única respuesta incorrecta sobre componentes y objetos:

A. Un marco de trabajo basado en componentes es una caja negra (black-box) que acepta que se le enchufen (plug-in) componentes software
B. Un componente puede contener varios elementos del lenguaje: clases, objetos-prototipo y otros recursos asociados (variables, constantes, imágenes, etc…), no todos estos elementos pertenecen a un lenguaje con orientación a objetos
C. También se puede ver como arquitectura software casi completa que contiene algoritmos de uso común dentro de un dominio de aplicaciones
D. Las superclases de una clase no tienen por qué pertenecer al mismo componente que esta
E. El estado de los objetos y de los componentes se obtiene a través de sus referencias

9. Explica por qué decimos que los lenguajes de programación orientados a objetos no definen una unidad de composición independiente de elementos para las aplicaciones

La POO no permite expresar con claridad la distinción entre aspectos computacionales y meramente composicionales de una aplicación. No define una unidad concreta de composición independiente de las aplicaciones (los objetos no lo son) y suele definir interfaces de un nivel tan bajo que suelen servir para establecer contratos entre las diferentes partes que desean utilizar los objetos de una determinada aplicación.

10. Cual es correcto de los siguientes ítems sobre Programación Orientada a Componentes (POC):

A. Los objetos de la Programación Orientada a Objetos (POO) son elementos pensados para conseguir la reutilización de los campos de una clase
B. Los componentes nunca pueden modificar los recursos que contienen
C. La reutilizaron de un componente software significa que puede ser utilizado más de una vez en la implementación de un sistema o de aplicaciones software
D. Los componentes software admiten una forma de polimorfismo que no se resuelve hasta el momento de la ejecución. -> Verdadero, el polimorfismo paramétrico (similar a los templates en C++) se resuelve en los componentes en tiempo de ejecución
E. Un componente no admite ni la redefinición (overriding) ni sobrecarga (overloading) de los lenguajes POO

11. Explica cuales son y en qué consisten las posibilidades de polimorfismo que nos ofrece la POC

La POC ofrece 3 tipos de polimorfismo:

- Reemplazabilidad: Capacidad de un componente de reemplazar a otro en una app sin romper los contratos con sus clientes. También puede ser denominado como polimorfismo de subtipos o de inclusión.
- Paramétrico: No se resuelve hasta tiempo de ejecución y se refiere a la implementación genérica de un componente.
- Polimorfismo acotado: Se da cuando se combinan polimorfismos paramétricos y de inclusión.

12. Explicar qué problemas presentaría la utilización del polimorfismo que se utilizar en la POO si intentamos utilizarlo en la POC.

En POO el polimorfismo se basa en sobre-escribir métodos de los objetos en tiempo de ejecución o sobrecargar operadores en tiempo de compilación. En cualquier caso, en mi opinión este tipo de polimorfismo no puede darse sobre un componente pues estaríamos modificando el comportamiento o el estado interno del componente.

13. Indicar la única respuesta incorrecta sobre la especificación de interfaces de los componentes:

A. Un componente software podría disponer de múltiples interfaces
B. Solo con la información de la interfaz, los usuarios de un componente no podrían conocer cómo se comportaría exactamente si se utilizase dentro de un marco de trabajo
C. La especificación concreta de las dependencias de un componente software dependerá del modelo de componentes que se utilice en un determinado marco de trabajo
D. Para que un componente sea reutilizable no puede tener ninguna dependencia con su contexto de uso (se exige su máxima robustez)
E. La implantación de estándares de componentes software en el futuro tendrá como consecuencia un mercado de componentes menos robustos, pero más reutilizables. Verdadero

14. Indicar cuál de los siguientes aspectos no es necesario para conseguir la interoperabilidad de los componentes:

A. Cómo se localizan y proporcionan los servicios que ofrecen
B. Cómo se gestiona la evolución y mantenimiento de los componentes
C. Conocer la forma de especificar correctamente las interfaces entre componentes

D. Ha de existir compatibilidad binaria en el paso de argumentos durante la llamada al servicio del componente
E. Manejar correctamente las referencias entre objetos remotos (cuando el espacio de direcciones se extiende más allá de los límites de un proceso)

15. Indicar la opción incorrecta sobre los denominados modelos de componentes:

A. Definen la forma y los mecanismos de interacción entre las interfaces de los componentes
B. Todo el marco de trabajo para componentes (component framework) ha de estar basado en un modelo de componentes concreto
C. Los marcos de trabajo actuales pueden incluir componentes dispersos geográficamente (distribución total de componentes)
D. La implementación de un sistema software consiste en la instanciación de un marco de trabajo mediante un conjunto de componentes (plugins) y una arquitectura específica que resuelven un determinado problema
E. Cada modelo de componentes (por ejemplo, DCOM, JavaBeans, CORBA) es específico de una sola plataforma de componentes distribuidos (ActiveX/OLE, Enterprise Beans, Orbix)

TEMA 2

1. Seleccionar la única respuesta correcta a la siguiente cuestión sobre concepto de patrón en el desarrollo software:

A. Un patrón es básicamente un esquema algorítmico (como divide y vencerás, programación dinámica, etc..) pero expresado con clases
B. Cada patrón es una estructura de clases fija que se adapta al desarrollo de una determinada aplicación o sistema-software
C. Las clases de un patrón se pueden modificar para adaptarse a las condiciones particulares de un problema que se repite en un contexto
D. Un patrón propone una solución totalmente general a un problema-tipo para cualquier plataforma de ejecución

2. Seleccionar cuál de los elementos siguientes no se puede considerar una característica propia de un SOA:

A. Su utilización para el desarrollo de aplicaciones y sistemas software propicia el bajo acoplamiento entre servicios
B. Han de permitir la composición siempre que se respete el contrato del servicio.
C. Son dependientes del tipo de modelo de negocio para el que se desarrollará un sistema de información
D. Los servicios que incluyen siempre han de poder ser descubiertos por otras aplicaciones
E. Pueden utilizar varios patrones de diseño arquitectónico
F. Nunca exponen el estado interno de un servicio a su entorno

3. Seleccionar la única respuesta correcta a la siguiente cuestión sobre las diferencias entre patrones y marcos de trabajo (frameworks):

A. Los marcos de trabajo son más abstractos que los patrones
B. Un marco de trabajo es el esqueleto de una sola aplicación que ha de ser adaptado a las necesidades concretas por el programador de la aplicación
C. Un patrón arquitectónico puede estar compuesto por varios marcos de trabajo
D. Los marcos de trabajo son un conjunto de clases que se han de utilizarse sin modificación

4. De las siguientes afirmaciones relacionadas sobre la descripción de un servicio con WSDL, indicar cuál de ellas no es correcta:

A. PortType proporciona la ordenación de los mensajes que un servicio puede procesar
B. La parte de ligadura en una descripción WSDL indica los requisitos que se han de cumplir para poder establecer las conexiones físicas
C. Operation define cual es el conjunto de mensajes de entrada / salida del servicio que se está describiendo
D. Las secciones Mensajes SOAP sirven para caracterizar a cada mensaje que se va a intercambiar por el servicio
E. Port representa la dirección física desde la cual se puede acceder al servicio

5. Seleccionar la única alternativa correcta sobre composición de servicios Web (SW):

A. La composición de SW solo consiste en conseguir unir virtualmente varios de estos servicios en uno solo

B. Un caso permite la interoperabilidad de los SW participantes pero dirigidos por un motor central de orquestación
C. Con la orquestación de SW la lógica de proceso de negocio está centralizada, pero mantiene su capacidad de composición con otros servicios y componentes
D. Una orquestación de SW establecerá un protocolo de comunicaciones que define cómo pueden interactuar los servicios participantes
E. El mecanismo de orquestación controla la lógica de negocio de una sola compañía a diferencia del de coreografía que no es propiedad de ninguna entidad

TEMA 4

1. Seleccionar cuál de los elementos siguientes no son características de un SOA:

A. Su utilización para el desarrollo de aplicaciones y sistemas de software propicia el bajo acoplamiento entre los servicios
B. Han de permitir la composición siempre que se respete el contrato de servicio
C. Son dependientes del tipo de modelo de negocio para el que se desarrollará un sistema de información
D. Los servicios que incluyen siempre han de poder ser descubiertos por otras aplicaciones.
E. Pueden utilizar varios patrones de diseño arquitectónico
F. Nunca exponen el estado interno de un servicio a su entorno

2. De las siguientes afirmaciones relacionadas sobre la descripción de un servicio WSDL, indicar cuál de ellas no es la correcta:

A. PortType y proporciona la ordenación de los mensajes que un servicio puede procesar
B. La parte de ligadura indica los requisitos que se han de cumplir para poder establecer las conexiones físicas
C. Operation define cuales el conjunto de mensajes de entrada/salida del servicio
D. Mensajes SOAP que caracteriza a cada mensaje que se va a intercambiar
E. Port representa la dirección física desde la cual se puede acceder al servicio

3. Seleccionar la única alternativa correcta sobre la composición de servicios Web:

A. La composición de servicios Web consiste en conseguir virtualmente varios de estos servicios en uno solo
B. Un caso permite la interoperabilidad de los servicios Web participantes pero dirigidos por un motor central de orquestación
C. Con la orquestación de servicios Web la lógica de proceso de negocio está centralizada, pero mantiene su capacidad de composición con otros servicios y componentes
D. Una orquestación de servicios Web establecerá un protocolo de comunicaciones que define cómo pueden interactuar los servicios participantes
E. El mecanismo de orquestación controla la lógica de negocio de una compañía a diferencia del de coreografía que no es propiedad de ninguna entidad

TEMA 5

1. De las siguientes afirmaciones relacionadas sobre el modelado de sistemas ubicuos, indicar cuál de ellas es la correcta:

A. IoT (Internet of Things) es un concepto para referirse a todo lo que está publicado en la Web
B. Los actuales modelos M2M se refieren a la sustitución que se ha realizado de las antiguas arquitectura cliente-servidor, tales como DCOM de Microsoft, por sistemas desubicados.
C. Gracias a la computación ubicua actualmente se puede hacer realidad por tecnologías IoT y M2M
D. Los temas relacionados con la seguridad y privacidad en sistemas ubicuos actualmente siguen un estándar publicados por W3C
E. Los servicios de los sistemas ubicuos solo se pueden componer si pertenecen al mismo dominio de la aplicación

2. De las siguientes afirmaciones relacionadas sobre servicios ubicuos, indicar cuál de ellas no es la correcta:

A. Para conseguir que los servicios se compongan como un único servicio de utilidad en aplicaciones complejas se necesita utilizar la tecnología SOA para desarrollarlos
B. La computación ubicua se refiere a dispositivos interconectados e inteligentes, que son capaces de funcionar autónomamente y colaborar entre sí para realizar tareas complejas
C. Funcionamiento transparente de dispositivos en el espacio ubicuo se refiere a que los dispositivos se conectan y realizan trabajo colaborativo sin intervención del usuario
D. Los dispositivos se combinan entre ellos para proporcionar una funcionalidad a sus usuarios muy superior a la que poseen individualmente considerados
E. Los términos: computación ubicua, computación pervasiva, inteligencia ambiental y sensibilidad al contexto son todos ellos equivalentes