EXAMEN SEGUNDO PARCIAL **Concurrencia y Sistemas Distribuidos**
Unidades Didácticas 7, 8, 9 y 11 - Prácticas 3 y 4 **12 de Junio de 2015**

PARTE TEORIA

Sobre los sistemas distribuidos:

1.	La transparencia de acceso oculta las diferencias en la representación de los datos y en cómo se accede a los recursos.	V
2.	Cuando un sistema distribuido es abierto facilita que uno de sus módulos o componentes pueda utilizarse en otro sistema distribuido.	V
3.	Para conseguir escalabilidad de distancia debe asumirse que se está utilizando una red de área local. JUSTIFICACIÓN: *La escalabilidad de distancia permite extender el sistema por redes de área amplia (WAN), por lo que si se usan algoritmos basados en redes de área local, se debe considerar que no se está usando una red local, sino una red WAN, por lo que hay que tener en cuenta los efectos de los retardos en la transmisión de los datos y la menor fiabilidad de las comunicaciones.*	F
4.	Los algoritmos descentralizados facilitan distribuir la carga computacional entre diferentes ordenadores.	V
5.	Se requiere utilizar algoritmos descentralizados para conseguir escalabilidad administrativa, de modo que los cómputos se distribuyen entre diferentes áreas administrativas del sistema. JUSTIFICACIÓN: *Para conseguir escalabilidad administrativa se deben utilizar protocolos y mecanismos estándar de autenticación y autorización; así como implementar mecanismos para proteger a cada organización del resto y del propio sistema.*	F
6.	Para mejorar la escalabilidad de tamaño, los clientes deben delegar en el servidor tantas responsabilidades como sea posible. JUSTIFICACIÓN: *Al contrario, contra menos se centralicen las tareas en los servidores, mayor escalabilidad de tamaño se podrá conseguir.*	F
7.	La *capa de middleware*, que se ubica bajo el nivel de aplicación, puede integrar algunos mecanismos de comunicación que faciliten la programación de aplicaciones distribuidas; por ejemplo: JMS.	V

Sobre el mecanismo de comunicación ROI:

8.	El componente denominado ORB se encarga, entre otras cosas, de identificar y localizar a los objetos remotos.	V
9.	El middleware de un sistema con comunicación basada en ROI emplea generalmente comunicación sincrónica no persistente y direccionamiento directo.	V
10.	Cuando se pasa un objeto por valor, se copia una referencia al mismo. JUSTIFICACIÓN: *Cuando se pasa un objeto por valor, el estado del objeto se empaqueta, mediante un proceso denominado serialización.*	F
11.	El proxy empaqueta los argumentos una vez recibe la contestación del esqueleto. JUSTIFICACIÓN: *El proxy desempaqueta los argumentos al recibir la contestación, para así devolverlos al proceso cliente.*	F

Sobre el mecanismo de comunicación Java RMI:

12.	El proxy correspondiente a un objeto remoto se crea en tiempo de ejecución cuando se accede por primera vez al objeto remoto.	V
13.	Se considera objeto local todo objeto que sólo puede invocarse desde la computadora en que se define (aunque sea desde otras máquinas virtuales Java), y objeto remoto a todo objeto que puede invocarse desde otras computadoras. *JUSTIFICACIÓN: Si un objeto se invoca desde otras máquinas virtuales Java de la misma computadora, también se considera como objeto remoto.*	F
14.	No proporciona transparencia de ubicación, porque la sintaxis de invocación del método es distinta dependiendo de si el objeto es local o remoto. *JUSTIFICACIÓN: La sintaxis de invocación es la misma. En la interfaz del objeto definimos los métodos del objeto (con independencia de que vaya a ser remoto o local). Si el objeto es remoto, la interfaz debe extender java.rmi.Remote.*	F
15.	La clase de un objeto remoto debe implementar un interfaz que extienda java.rmi.Remote	V

Respecto a los servicios Web RESTful:

16.	En un servicio Web RESTFul, los mensajes constan de una cabecera formada por varios campos fijos definidos por el estándar RESTful, un conjunto de propiedades que pueden ser definidas por la aplicación y de un contenido (normalmente, en XML o JSON). *JUSTIFICACIÓN: El estilo arquitectónico REST no especifica ninguna cabecera, ni campos fijos definidos. Por tanto, los mensajes tienen estructura libre.*	F
17.	GET https:/weatherapp.com/zipcodes es un ejemplo de llamada a un servicio Web RESTFul.	V
18.	Un sistema distribuido con comunicación basada en servicios web REST emplea generalmente comunicación sincrónica no persistente y direccionamiento directo.	V
19.	Son atendidos por servidores que mantienen el estado de las peticiones de los clientes. *JUSTIFICACIÓN: Los servicios son "sin estado", de modo que los servicios no mantienen ninguna sesión respecto a las peticiones de los clientes.*	F

Respecto al mecanismo de comunicación Java Message Service:

20.	JMS utiliza direccionamiento directo, pues en la cabecera del mensaje se especifica claramente el destino del mensaje. *JUSTIFICACIÓN: JMS emplea direccionamiento indirecto, ya que el cliente envía el mensaje a un destino (Destination), que puede ser una cola (Queue) o un tema (Topic). Y no tiene por qué conocer quién será el consumidor del mensaje.*	F
21.	Java Message Service ofrece una comunicación fuertemente acoplada, pues tanto el productor como el consumidor de un mensaje se crean utilizando la misma factoría de conexión. *JUSTIFICACIÓN: JMS ofrece una comunicación débilmente acoplada, ya que al emplear destinos, el productor y el consumidor del mensaje no necesitan conocerse. Solamente requieren conocer el destino (y estar de acuerdo en el formato del mensaje).*	F
22.	Existen dos tipos de destinos: Colas (Queues), utilizadas para enviar mensajes a un único cliente; y Temas (Topics), que permiten la publicitación/suscripción.	V
23.	Las factorías de conexiones y los destinos se crean utilizando una herramienta administrativa ofrecida por el Proveedor de JMS.	V

Respecto a la sincronización en sistemas distribuidos:

24.	En un sistema distribuido, si varios nodos desean hacer uso de un recurso compartido, pero dicho recurso no puede ser utilizado a la vez por más de un nodo, debemos utilizar un algoritmo de selección de líder para escoger al nodo que puede hacer uso del recurso. *JUSTIFICACIÓN: Para el acceso en exclusión mutua a un recurso compartido debemos utilizar un algoritmo de exclusión mutua de sistemas distribuidos (por ejemplo, cualquiera de los tres algoritmos vistos en la asignatura, i.e. algoritmo centralizado, algoritmo distribuido o algoritmo para anillos).*	F
25.	El algoritmo de Chandy-Lamport permite establecer un orden total entre los eventos de un sistema distribuido. *JUSTIFICACIÓN: El algoritmo Chandy-Lamport permite obtener el estado global del sistema distribuido. Para establecer un orden total, utilizaremos los relojes lógicos de Lamport y los identificadores de los nodos.*	F
26.	Si los nodos de un sistema distribuido están dispuestos en un anillo lógico, resultaría sencillo obtener el estado global del sistema, pues pueden registrar su estado cuando reciben el token que circula por el anillo. *JUSTIFICACIÓN: El estado global incluye no sólo el estado de cada nodo, sino también los mensajes en tránsito (que han sido enviados y que todavía no han llegado a su destino). Por tanto, haría falta registrar también (de algún modo) cuáles son dichos mensajes, de modo que obtengamos una instantánea consistente.*	F

Sobre los algoritmos vistos en esta asignatura de exclusión mutua y elección de líder para sistemas distribuidos:

27.	El algoritmo distribuido de exclusión mutua requiere que se utilicen relojes lógicos de Lamport e identificadores para decidir la precedencia de las solicitudes. *JUSTIFICACIÓN: Se requiere establecer un orden total de los eventos para así resolver los empates que se puedan producir cuando dos o más nodos desean acceder "a la vez" a la sección crítica.*	V
28.	En el algoritmo Bully, un nodo sabe que será el nuevo coordinador cuando, tras enviar los mensajes de tipo ELECCIÓN, no recibe ninguna respuesta. *JUSTIFICACIÓN: Como envía ELECCIÓN a los nodos con mayor identificador, si no recibe respuesta, entonces él es el nodo activo con mayor identificador. Por tanto, él es el nuevo líder.*	V
29.	En el algoritmo de elección de líder para anillos, el mensaje ELECCIÓN incluye dos campos: uno para registrar el iniciador y otro para informar de cúal era el líder antes de iniciar el algoritmo. *JUSTIFICACIÓN: El mensaje ELECCIÓN incluye dos campos. Uno para registrar el iniciador y el otro para registrar la lista de nodos activos.*	F
30.	En el algoritmo centralizado de exclusión mutua, cuando un nodo solicita entrar en la sección crítica y ésta está ocupada, el coordinador envía un mensaje al nodo que está en la sección crítica para que este último, cuando termine, se lo notifique al nodo que ha hecho la solicitud. *JUSTIFICACIÓN: El coordinador no envía ningún mensaje al nodo en la sección crítica. Cuando éste termina, notifica directamente al coordinador.*	F
31.	En el algoritmo distribuido de exclusión mutua, un nodo entra en la sección crítica cuando recibe un mensaje OK de cada uno de los demás nodos.	V

Respecto a los algoritmos de sincronización de relojes físicos.

32.	En el algoritmo de Cristian, si un cliente C pregunta al servidor su hora en el instante 20.000 (según el reloj de C), recibe la respuesta del servidor en el instante 20.010 (según el reloj de C) y calcula que el nuevo valor para su reloj debe ser 20.024, entonces se puede deducir que la respuesta del servidor habrá sido 14. *JUSTIFICACIÓN: En el algoritmo de Cristian, el servidor envía "su valor de reloj", por ejemplo, 20.014. Pero no envía "su diferencia" respecto al cliente.*	F
33.	En el algoritmo de Berkeley, el nodo que actúa como coordinador puede que tenga que ajustar también su reloj, al igual que los otros nodos que participan en el algoritmo.	V
34.	El algoritmo de Berkeley asume que el envío de un mensaje desde el nodo A al nodo B consume el mismo tiempo que la respuesta desde B hasta A.	V

Respecto a los relojes lógicos de Lamport y vectoriales:

35.	Si el valor en un evento **x** del reloj lógico de Lamport es igual a C(**x**)=1 y el valor en **y** del reloj lógico de Lamport es igual a C(**y**)=6, entonces podemos afirmar que **x→ y**. *JUSTIFICACIÓN: Dados los valores de los relojes lógicos de Lamport, no se puede afirmar si dos eventos son concurrentes o si uno ocurre antes que el otro.*	F
36.	Si el valor en **x** del reloj vectorial es igual a V(x)=[9,0,1] y el valor en **y** del reloj vectorial es igual a [6,2,2], entonces podemos afirmar que **x || y**	V

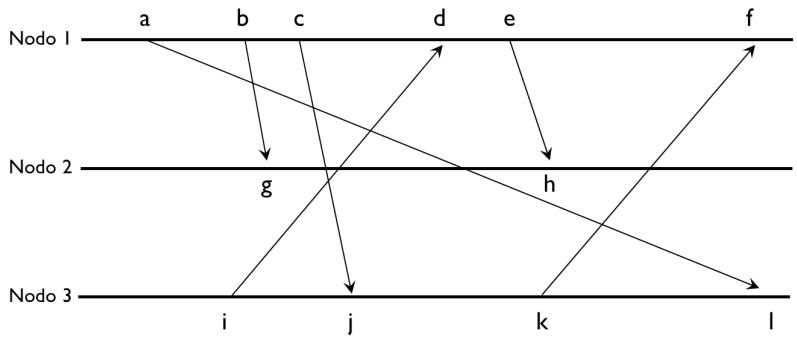


Figura 1: muestra todos los envíos de mensajes entre tres nodos

Respecto a la figura 1:

37.	Se cumple que $\mathbf{c} \rightarrow \mathbf{l}$ y que $\mathbf{e} \| \| \mathbf{l}$	V
38.	El valor en **l** del reloj lógico de Lamport es igual a 6	V
39.	El valor en **d** del reloj vectorial es igual a V(d)=[4,0,1] y el valor en **j** del reloj vectorial es igual a V(j)= [3,1,2] *JUSTIFICACIÓN: V(d)=[4,0,1] , V(j)=[3,0,2]*	F

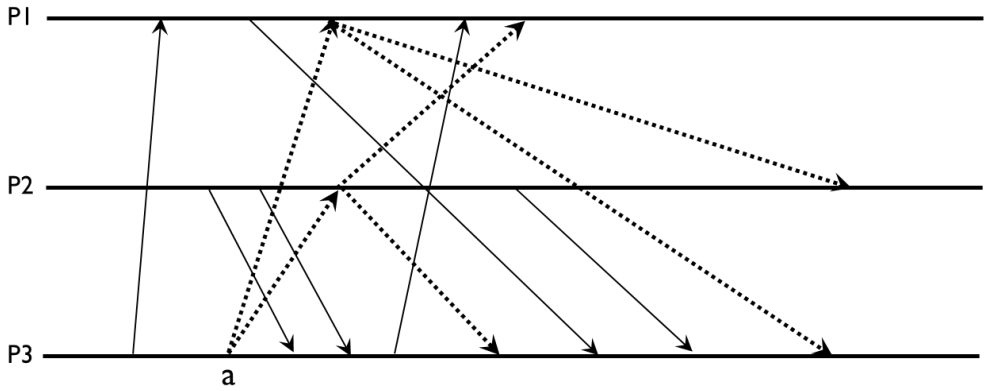


Figura 2

Respecto a la arquitectura de los sistemas distribuidos:

40.	Las arquitecturas de capas, las arquitecturas basadas en objetos y las arquitecturas basadas en eventos son ejemplos de estilos arquitectónicos de arquitecturas software.	V
41.	En una arquitectura centralizada se distinguen dos tipos de roles: servidor, responsable de la gestión de un recurso determinado y que define una serie de servicios (sobre dicho recurso); y cliente, que es un componente que utiliza dichos servicios.	V
42.	Los sistemas replicados y los sistemas peer-to-peer presentan distribución horizontal.	V

Respecto a la figura 2, suponga que P3 inicia el algoritmo de Chandy-Lamport en **a**, siendo las líneas discontinuas los mensajes que se envían como consecuencia de la ejecución de este algoritmo y las líneas continuas los mensajes normales. Cuando finaliza el algoritmo,

43.	en el canal (P3,P1) se habrán registrado 0 mensajes *JUSTIFICACIÓN: Al acabar el algoritmo, la instantánea consistente comprende los puntos donde P1 y P2 recibieron el primer MARCA (pues ahí guardaron su estado), así como el punto "a" (donde P3 guardó su estado). Si unimos esos puntos, podemos ver claramente que el algoritmo habrá registrado 1 mensaje en canal (P1,P3), 2 mensajes en canal (P2, P3), 0 mensajes en canal (P3,P1) y 0 mensajes en canal (P3,P2). Los otros mensajes, o bien pasaron antes de la instantánea (como el primer mensaje del dibujo), o bien pasarán después, en el futuro, como ocurre con el mensaje en canal (P3,P1) y el último mensaje del canal (P2, P3).*	V
44.	en el canal (P2,P3) se habrán registrado 3 mensajes	F
45.	en el canal (P1,P3) se habrá registrado 1 mensaje.	V

Respecto a los sistemas peer-to-peer, los sistemas Grid y los sistemas Cloud:

46.	Aunque los sistemas *peer-to-peer* se basan en distribución horizontal, para el servicio de localización de recursos se han realizado implementaciones con uno o con varios servidores centralizados, a los que dirigir las búsquedas de recursos.	V
47.	En las arquitecturas *peer-to-peer parcialmente centralizadas* el servicio de transferencia de datos se realiza a través de servidores denominados *supernodes* (supernodos). *JUSTIFICACIÓN: Es el servicio de localización de recursos el que se realiza a través de los supernodos.*	F
48.	Los cuatro objetivos de los sistemas distribuidos se observan claramente en los sistemas Grid: permiten el acceso a recursos remotos compartidos; ofrecen transparencia (por ejemplo, de acceso, de localización); son sistemas abiertos y ofrecen escalabilidad (por ejemplo, de distancia).	V

49.	Cuando se utilizan los servicios Cloud, no es necesario sobredimensionar el sistema a priori para prevenir un futuro aumento en la necesidad de recursos en caso de que el sistema necesite crecer. *JUSTIFICACIÓN: Las empresas que proporcionan los servicios Cloud se encargarán de ofrecernos la escalabilidad de tamaño que, como clientes, necesitemos.*	V
50.	El *Cloud Computing de Plataforma como Servicio* (PaaS) permite desarrollar, desplegar y ejecutar aplicaciones web a través de Internet.	V

PARTE PRACTICAS

Respecto a la práctica sobre Active Directory:

1.	Los diferentes dominios de una red corporativa se estructuran de forma jerárquica, formando un solo árbol de dominios.	F
2.	Un usuario de un dominio sólo puede iniciar sesión en ordenadores de su mismo dominio.	F
3.	Para que un usuario pueda modificar un fichero de un recurso compartido, hace falta que tenga permisos para ello tanto sobre el fichero como sobre el recurso compartido.	V
4.	Los administradores de dominios secundarios o subdominios pueden, por defecto, abrir sesión en la maquina controlador del dominio principal.	F
5.	Cada dominio solo puede tener un domain controller (DC, controlador de dominio).	F
6.	Una vez se ha construido el bosque, solo queda un usuario con capacidades administrativas, el administrador del dominio raíz. El resto de administradores locales se desactivan.	F
7.	Para crear un dominio secundario del dominio raíz, es necesario proporcionar las credenciales del administrador del dominio raíz.	V
8.	En esta práctica se pedía crear un único dominio con dos controladores de dominio.	F

Respecto a la práctica del Chat distribuido en Java RMI:

9.	Todos los objetos invocables de forma remota deben estar registrados en el servidor de nombres (rmiregistry).	F
10.	Cada uno de los procesos ChatClient y ChatServer abre un puerto, porque es necesario para utilizar el servidor de nombres (rmiregistry).	F
11.	Los objetos de la clase ChatMessage son creados siempre por el ChatServer.	F
12.	ChatRobot debe utilizar la clase que proporciona la interfaz gráfica (ChatUI) para obtener las indicaciones sobre a qué servidor y canal debe conectarse.	F

Dado el siguiente fragmento de código, que corresponde al código original de la práctica de Chat, donde los puntos suspensivos indican código omitido:

..... (Omitido. Código en el enunciado del examen)

13.	Los objetos de la clase ChatUser pueden ser accedidos remotamente.	V
14.	Los objetos de la clase ChatClient pueden ser accedidos remotamente.	F
15.	El método sendMessage del objeto ChatUser se invoca para que el usuario que representa reciba un mensaje de un canal.	V
16.	En la clase ChatClient, la variable srv es un proxy del servidor de chat (objeto remoto ChatServer).	V

Este examen tiene una duración total de 2 horas y 30 minutos. Consta de dos partes: TEORIA y PRÁCTICAS. Debe escribir su nombre y apellidos en la hoja de respuestas **de cada parte**.

PARTE TEORIA

Esta parte tiene una puntuación máxima de **10 puntos**, que equivalen a **3** puntos de la nota final de la asignatura. Indique, para cada una de las siguientes **50 afirmaciones**, si éstas son verdaderas (**V**) o falsas (**F**).

Cada respuesta vale: correcta= 0.2, errónea= -0.2, vacía=0.

Importante: Los **primeros 3 errores no penalizarán**, de modo que tendrán una valoración equivalente a la de una respuesta vacía. A partir del 4º error (inclusive), sí se aplicará el decremento por respuesta errónea.

Respecto a la sincronización en sistemas distribuidos: V/F

1.	Si los nodos de un sistema distribuido están dispuestos en un anillo lógico, resultaría sencillo obtener el estado global del sistema, pues pueden registrar su estado cuando reciben el token que circula por el anillo. *JUSTIFICACIÓN: El estado global incluye no sólo el estado de cada nodo, sino también los mensajes en tránsito (que han sido enviados y que todavía no han llegado a su destino). Por tanto, haría falta registrar también (de algún modo) cuáles son dichos mensajes, de modo que obtengamos una instantánea consistente.*	F
2.	En un sistema distribuido, si varios nodos desean hacer uso de un recurso compartido, pero dicho recurso no puede ser utilizado a la vez por más de un nodo, debemos utilizar un algoritmo de selección de líder para escoger al nodo que puede hacer uso del recurso. *JUSTIFICACIÓN: Para el acceso en exclusión mutua a un recurso compartido debemos utilizar un algoritmo de exclusión mutua de sistemas distribuidos (por ejemplo, cualquiera de los tres algoritmos vistos en la asignatura, i.e. algoritmo centralizado, algoritmo distribuido o algoritmo para anillos).*	F
3.	El algoritmo de Chandy-Lamport permite establecer un orden total entre los eventos de un sistema distribuido. *JUSTIFICACIÓN: El algoritmo Chandy-Lamport permite obtener el estado global del sistema distribuido. Para establecer un orden total, utilizaremos los relojes lógicos de Lamport y los identificadores de los nodos.*	F

Respecto a los servicios Web RESTful:

4.	Son atendidos por servidores que mantienen el estado de las peticiones de los clientes. *JUSTIFICACIÓN: Los servicios son "sin estado", de modo que los servicios no mantienen ninguna sesión respecto a las peticiones de los clientes.*	F
5.	`GET https:/weatherapp.com/zipcodes` es un ejemplo de llamada a un servicio Web RESTFul.	V
6.	En un servicio Web RESTFul, los mensajes constan de una cabecera formada por varios campos fijos definidos por el estándar RESTful, un conjunto de propiedades que pueden ser definidas por la aplicación y de un contenido (generalmente, en XML o JSON). *JUSTIFICACIÓN: El estilo arquitectónico REST no especifica ninguna cabecera, ni campos fijos definidos. Por tanto, los mensajes tienen estructura libre.*	F
7.	Un sistema distribuido con comunicación basada en servicios web REST emplea generalmente comunicación sincrónica no persistente y direccionamiento directo.	V

Respecto al mecanismo de comunicación Java Message Service:

8.	Existen dos tipos de destinos: Colas (Queues), utilizadas para enviar mensajes a un único cliente; y Temas (Topics), que permiten la publicitación/suscripción.	V
9.	Las factorías de conexiones y los destinos se crean utilizando una herramienta administrativa ofrecida por el Proveedor de JMS.	V
10.	JMS utiliza direccionamiento directo, pues en la cabecera del mensaje se especifica claramente el destino del mensaje. *JUSTIFICACIÓN: JMS emplea direccionamiento indirecto, ya que el cliente envía el mensaje a un destino (Destination), que puede ser una cola (Queue) o un tema (Topic). Y no tiene por qué conocer quién será el consumidor del mensaje.*	F
11.	Java Message Service ofrece una comunicación fuertemente acoplada, pues tanto el productor como el consumidor de un mensaje se crean utilizando la misma factoría de conexión. *JUSTIFICACIÓN: JMS ofrece una comunicación débilmente acoplada, ya que al emplear destinos, el productor y el consumidor del mensaje no necesitan conocerse. Solamente requieren conocer el destino (y estar de acuerdo en el formato del mensaje).*	F

Sobre el mecanismo de comunicación ROI:

12.	Cuando se pasa un objeto por valor, se copia una referencia al mismo. JUSTIFICACIÓN: *Cuando se pasa un objeto por valor, el estado del objeto se empaqueta, mediante un proceso denominado serialización.*	F
13.	El proxy empaqueta los argumentos una vez recibe la contestación del esqueleto. JUSTIFICACIÓN: *El proxy desempaqueta los argumentos al recibir la contestación, para así devolverlos al proceso cliente.*	F
14.	El componente denominado ORB se encarga, entre otras cosas, de identificar y localizar a los objetos remotos.	V
15.	El middleware de un sistema con comunicación basada en ROI emplea generalmente comunicación sincrónica no persistente y direccionamiento directo.	V

Sobre el mecanismo de comunicación Java RMI:

16.	No proporciona transparencia de ubicación, porque la sintaxis de invocación del método es distinta dependiendo de si el objeto es local o remoto. *JUSTIFICACIÓN: La sintaxis de invocación es la misma. En la interfaz del objeto definimos los métodos del objeto (con independencia de que vaya a ser remoto o local). Si el objeto es remoto, la interfaz debe extender java.rmi.Remote.*	F
17.	Se considera objeto local todo objeto que sólo puede invocarse desde la computadora en que se define (aunque sea desde otras máquinas virtuales Java), y objeto remoto a todo objeto que puede invocarse desde otras computadoras. *JUSTIFICACIÓN: Si un objeto se invoca desde otras máquinas virtuales Java de la misma computadora, también se considera como objeto remoto.*	F
18.	El proxy correspondiente a un objeto remoto se crea en tiempo de ejecución cuando se accede por primera vez al objeto remoto.	V
19.	La clase de un objeto remoto debe implementar un interfaz que extienda *java.rmi.Remote*.	V

Sobre los sistemas distribuidos:

20.	Para mejorar la escalabilidad de tamaño, los clientes deben delegar en el servidor tantas responsabilidades como sea posible. JUSTIFICACIÓN: *Al contrario, contra menos se centralicen las tareas en los servidores, mayor escalabilidad de tamaño se podrá conseguir.*	F
21.	Los algoritmos descentralizados facilitan distribuir la carga computacional entre diferentes ordenadores.	V

22.	La transparencia de acceso oculta las diferencias en la representación de los datos y en cómo se accede a los recursos.	V
23.	La *capa de middleware*, que se ubica bajo el nivel de aplicación, puede integrar algunos mecanismos de comunicación que faciliten la programación de aplicaciones distribuidas; por ejemplo: JMS.	V
24.	Cuando un sistema distribuido es abierto facilita que uno de sus módulos o componentes pueda utilizarse en otro sistema distribuido.	V
25.	Para conseguir escalabilidad de distancia debe asumirse que se está utilizando una red de área local. JUSTIFICACIÓN: *La escalabilidad de distancia permite extender el sistema por redes de área amplia (WAN), por lo que si se usan algoritmos basados en redes de área local, se deben considerar los efectos de los retardos en la transmisión de los datos y la menor fiabilidad de las comunicaciones.*	F
26.	Se requiere utilizar algoritmos descentralizados para conseguir escalabilidad administrativa, de modo que los cómputos se distribuyen entre diferentes áreas administrativas del sistema. JUSTIFICACIÓN: *Para conseguir escalabilidad administrativa se deben utilizar protocolos y mecanismos estándar de autenticación y autorización; así como implmentar mecanismos para proteger a cada organización del resto y del propio sistema.*	F

Respecto a la arquitectura de los sistemas distribuidos:

27.	Los sistemas replicados y los sistemas peer-to-peer presentan distribución horizontal.	V
28.	Las arquitecturas de capas, las arquitecturas basadas en objetos y las arquitecturas basadas en eventos son ejemplos de estilos arquitectónicos de arquitecturas software.	V
29.	En una arquitectura centralizada se distinguen dos tipos de roles: servidor, responsable de la gestión de un recurso determinado y que define una serie de servicios (sobre dicho recurso); y cliente, que es un componente que utiliza dichos servicios.	V

Sobre los algoritmos vistos en esta asignatura de exclusión mutua y elección de líder para sistemas distribuidos:

30.	En el algoritmo de elección de líder para anillos, el mensaje ELECCIÓN incluye dos campos: uno para registrar el iniciador y otro para informar de cúal era el líder antes de iniciar el algoritmo. *JUSTIFICACIÓN: El mensaje ELECCIÓN incluye dos campos. Uno para registrar el iniciador y el otro para registrar la lista de nodos activos.*	F
31.	En el algoritmo distribuido de exclusión mutua, un nodo entra en la sección crítica cuando recibe un mensaje OK de cada uno de los demás nodos.	V
32.	El algoritmo distribuido de exclusión mutua requiere que se utilicen relojes lógicos de Lamport e identificadores para decidir la precedencia de las solicitudes *JUSTIFICACIÓN: Se requiere establecer un orden total de los eventos para así resolver los empates que se puedan producir cuando dos o más nodos desean acceder "a la vez" a la sección crítica.*	V
33.	En el algoritmo Bully, un nodo sabe que será el nuevo coordinador cuando, tras enviar los mensajes de tipo ELECCIÓN, no recibe ninguna respuesta. *JUSTIFICACIÓN: Como envía ELECCIÓN a los nodos con mayor identificador, si no recibe respuesta, entonces él es el nodo activo con mayor identificador. Por tanto, él es el nuevo líder.*	V
34.	En el algoritmo centralizado de exclusión mutua, cuando un nodo solicita entrar en la sección crítica y ésta está ocupada, el coordinador envía un mensaje al nodo que está en la sección crítica para que este último, cuando termine, se lo notifique al nodo que ha hecho la solicitud. *JUSTIFICACIÓN: El coordinador no envía ningún mensaje al nodo en la sección crítica. Cuando éste termina, notifica directamente al coordinador.*	F

Respecto a los relojes lógicos de Lamport y vectoriales:

35.	Si el valor en **x** del reloj vectorial es igual a V(x)=[9,0,1] y el valor en **y** del reloj vectorial es igual a [6,2,2], entonces podemos afirmar que **x || y .**	V
36.	Si el valor en un evento **x** del reloj lógico de Lamport es igual a C(**x**)=1 y el valor en **y** del reloj lógico de Lamport es igual a C(**y**)=6, podemos afirmar que **x→ y**. *JUSTIFICACIÓN: Dados los valores de los relojes lógicos de Lamport, no se puede afirmar si dos eventos son concurrentes o si uno ocurre antes que el otro.*	F

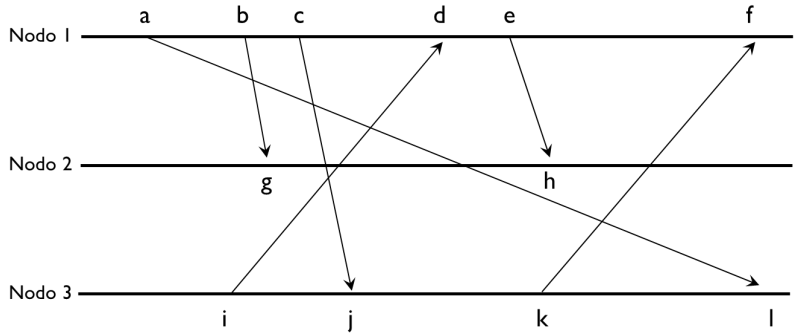


Figura 1: muestra todos los envíos de mensajes entre tres nodos

Respecto a la figura 1:

37.	El valor en **d** del reloj vectorial es igual a V(d)=[4,0,1] y el valor en **j** del reloj vectorial es igual a V(j)= [3,1,2] *JUSTIFICACIÓN: V(d)=[4,0,1] , V(j)=[3,0,2]*	F
38.	Se cumple que **c → l** y que **e || l**	V
39.	El valor en **l** del reloj lógico de Lamport es igual a 6	V

Respecto a los algoritmos de sincronización de relojes físicos.

40.	El algoritmo de Berkeley asume que el envío de un mensaje desde el nodo A al nodo B consume el mismo tiempo que la respuesta desde B hasta A.	V
41.	En el algoritmo de Cristian, si un cliente C pregunta al servidor su hora en el instante 20.000 (según el reloj de C), recibe la respuesta del servidor en el instante 20.010 (según el reloj de C) y calcula que el nuevo valor para su reloj debe ser 20.024, entonces se puede deducir que la respuesta del servidor habrá sido 14. *JUSTIFICACIÓN: En el algoritmo de Cristian, el servidor envía "su valor de reloj", por ejemplo, 20.014. Pero no envía "su diferencia" respecto al cliente.*	F
42.	En el algoritmo de Berkeley, el nodo que actúa como coordinador puede que tenga que ajustar también su reloj, al igual que los otros nodos que participan en el algoritmo.	V

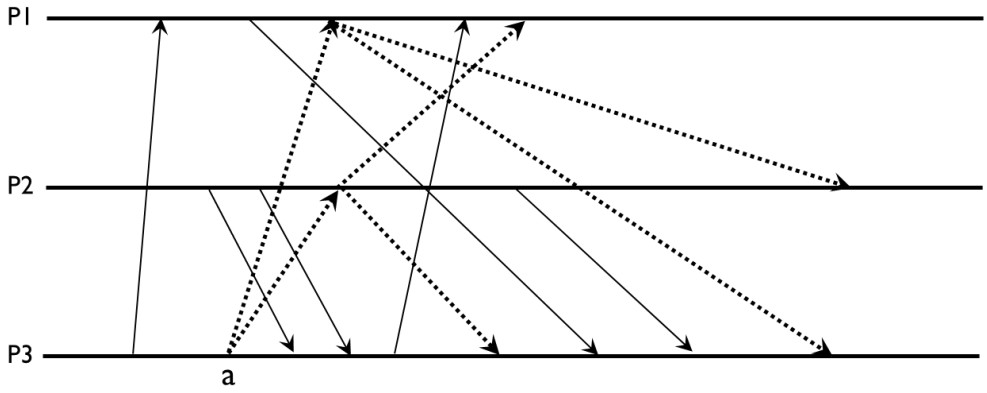


Figura 2

Respecto a la figura 2, suponga que P3 inicia el algoritmo de Chandy-Lamport en **a**, siendo las líneas discontinuas los mensajes que se envían como consecuencia de la ejecución de este algoritmo y las líneas continuas los mensajes normales. Cuando finaliza el algoritmo,

43.	en el canal (P3,P1) se habrán registrado 0 mensajes. *JUSTIFICACIÓN: Al acabar el algoritmo, la instantánea consistente comprende los puntos donde P1 y P2 recibieron el primer MARCA (pues ahí guardaron su estado), así como el punto "a" (donde P3 guardó su estado). Si unimos esos puntos, podemos ver claramente que el algoritmo habrá registrado 1 mensaje en canal (P1,P3), 2 mensajes en canal (P2, P3), 0 mensajes en canal (P3,P1) y 0 mensajes en canal (P3,P2). Los otros mensajes, o bien pasaron antes de la instantánea (como el primer mensaje del dibujo), o bien pasarán después, en el futuro, como ocurre con el mensaje en canal (P3,P1) y el último mensaje del canal (P2, P3).*	V
44.	en el canal (P1,P3) se habrá registrado 1 mensaje.	V
45.	en el canal (P2,P3) se habrán registrado 3 mensajes.	F

Respecto a los sistemas peer-to-peer, los sistemas Grid y los sistemas Cloud:

46.	El *Cloud Computing de Plataforma como Servicio* (PaaS) permite desarrollar, desplegar y ejecutar aplicaciones web a través de Internet.	V
47.	Cuando se utilizan los servicios Cloud, no es necesario sobredimensionar a priori el sistema para prevenir un futuro aumento en la necesidad de recursos en caso de que el sistema necesite crecer. *JUSTIFICACIÓN: Las empresas que proporcionan los servicios Cloud se encargarán de ofrecernos la escalabilidad de tamaño que, como clientes, necesitemos.*	V
48.	En las arquitecturas *peer-to-peer parcialmente centralizadas* el servicio de transferencia de datos se realiza a través de servidores denominados *supernodes* (supernodos). *JUSTIFICACIÓN: Es el servicio de localización de recursos el que se realiza a través de los supernodos.*	F
49.	Aunque los sistemas *peer-to-peer* se basan en distribución horizontal, para el servicio de localización de recursos se han realizado implementaciones con uno o con varios servidores centralizados, a los que dirigir las búsquedas de recursos.	V
50.	Los cuatro objetivos de los sistemas distribuidos se observan claramente en los sistemas Grid: permiten el acceso a recursos remotos compartidos; ofrecen transparencia (por ejemplo, de acceso, de localización); son sistemas abiertos y ofrecen escalabilidad (por ejemplo, de distancia).	V

PARTE PRACTICAS

Esta parte tiene una puntuación máxima de **10 puntos**, que equivalen a **1** punto de la nota final de la asignatura. Indique, para cada una de las siguientes **16 afirmaciones**, si éstas son verdaderas (**V**) o falsas (**F**).

Cada respuesta vale: correcta= 0.625, errónea= -0.625, vacía=0.

Importante: El **primer error no penalizará**, de modo que tendrá una valoración equivalente a la de una respuesta vacía. A partir del 2º error (inclusive), sí se aplicará el decremento por respuesta errónea.

Respecto a la práctica sobre Active Directory:

1.	En esta práctica se pedía crear un único dominio con dos controladores de dominio.	F
2.	Cada dominio solo puede tener un domain controller (DC, controlador de dominio).	F
3.	Los diferentes dominios de una red corporativa se estructuran de forma jerárquica, formando un solo árbol de dominios.	F
4.	Para crear un dominio secundario del dominio raíz, es necesario proporcionar las credenciales del administrador del dominio raíz.	V
5.	Un usuario de un dominio sólo puede iniciar sesión en ordenadores de su mismo dominio.	F
6.	Para que un usuario pueda modificar un fichero de un recurso compartido, hace falta que tenga permisos para ello tanto sobre el fichero como sobre el recurso compartido.	V
7.	Los administradores de dominios secundarios o subdominios pueden, por defecto, abrir sesión en la maquina controlador del dominio principal.	F
8.	Una vez se ha construido el bosque, solo queda un usuario con capacidades administrativas, el administrador del dominio raíz. El resto de administradores locales se desactivan.	F

Respecto a la práctica del Chat distribuido en Java RMI:

9.	Cada uno de los procesos ChatClient y ChatServer abre un puerto, porque es necesario para utilizar el servidor de nombres (rmiregistry).	F
10.	ChatRobot debe utilizar la clase que proporciona la interfaz gráfica (ChatUI) para obtener las indicaciones sobre a qué servidor y canal debe conectarse.	F
11.	Todos los objetos invocables de forma remota deben estar registrados en el servidor de nombres (rmiregistry).	F
12.	Los objetos de la clase ChatMessage son creados siempre por el ChatServer.	F

Dado el siguiente fragmento de código, que corresponde al código original de la práctica de Chat, donde los puntos suspensivos indican código omitido:

```
// Simple ChatUser implementation.
// Notice how this implementation just calls a listener
public class ChatUser extends UnicastRemoteObject implements IChatUser
{
   private String nick;
   private MessageListener lis;

   public ChatUser (...) throws RemoteException {...}
   public String getNick() throws RemoteException {...}
   public void sendMessage (IChatMessage msg) throws RemoteException {
      lis.messageArrived (msg);
   }
}

public class ChatClient implements MessageListener {
   …
   public String [] doConnect (…) throws Exception {
      …
       Registry reg = LocateRegistry.getRegistry (…);
       srv = (IChatServer) reg.lookup (…);
      …
   }
}
```

13.	Los objetos de la clase ChatClient pueden ser accedidos remotamente.	F
14.	En la clase ChatClient, la variable srv es un proxy del servidor de chat (objeto remoto ChatServer).	V
15.	Los objetos de la clase ChatUser pueden ser accedidos remotamente.	V
16.	El método sendMessage del objeto ChatUser se invoca para que el usuario que representa reciba un mensaje de un canal.	V

EXAMEN 2do PARCIAL – Bloque Unidades Didácticas 7 a 11 9 de Junio de 2014
Concurrencia y Sistemas Distribuidos Modelo A

APELLIDOS:				**NOMBRE:**	
DNI:		**GRUPO:**		**FIRMA:**	

Este bloque tiene una puntuación máxima de **10 puntos,** que equivalen a 2.5 puntos de la nota final de la asignatura. Indique, para cada una de las siguientes 50 afirmaciones, si éstas son verdaderas (**V**) o falsas (**F**). **Cada respuesta vale: correcta= 0.2, errónea= -0.2, vacía=0.**

Importante: Los primeros **3 errores** no penalizarán, de modo que tendrán una valoración equivalente a la de una respuesta vacía. A partir del 4º error (inclusive), sí se aplicará el decremento por respuesta errónea.

1. Sobre los sistemas distribuidos:

F	A.- Todos los sistemas de tiempo real son ejemplos de sistemas distribuidos.
V	B.- Algunos tipos de transparencia pueden comprometer la eficiencia del sistema, por ejemplo introduciendo retardos en las interacciones entre componentes de un sistema distribuido.
V	C.- Un sistema distribuido ofrece la imagen de un sistema coherente y único.
F	D.- Los sistemas distribuidos proporcionan diferentes tipos de transparencia. Entre ellos: transparencia en el rendimiento, en la escalabilidad, en su disponibilidad, en la seguridad...
F	E.- Si se desea que un sistema distribuido ofrezca escalabilidad de tamaño, no se debe hacer uso de técnicas de replicación, ya que al actualizar el estado de un componente tendremos que propagar tal actualización a todas las réplicas.

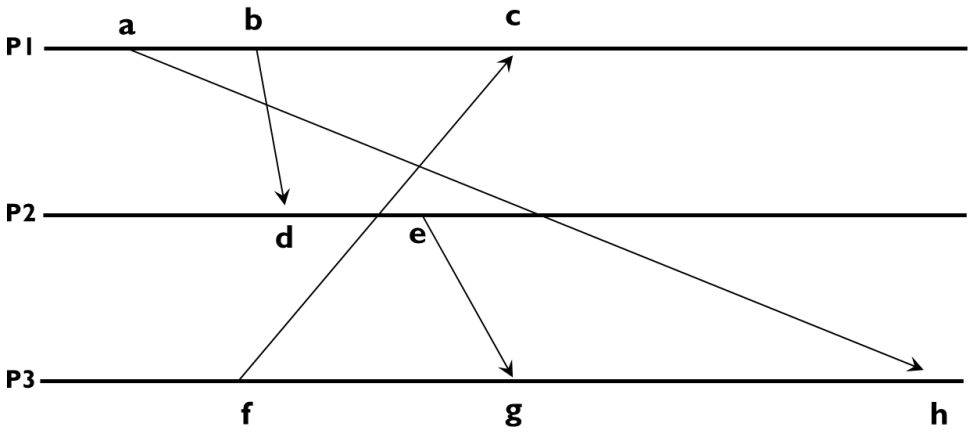
2. Sobre el mecanismo de comunicación RPC:

F	A.- Al igual que ROI, siempre requiere que se utilice un servidor de nombres.
V	B.- El stub cliente sirve para poder invocar procedimientos remotos como si fuesen locales
F	C.- Sólo podemos definir como procedimientos remotos aquellos que no utilicen paso de parámetros por referencia.
F	D.- Se basa en un modelo de invocación/respuesta con asincronía y persistencia
F	E.- El programador debe escribir el código de los stubs cliente y servidor para cada procedimiento que pueda invocarse de forma remota

3. Sobre el mecanismo de comunicación ROI:

V	A.- El cliente obtiene un proxy para invocar al objeto remoto.
V	B.- El proxy incluye una referencia al objeto remoto
F	C.- El mecanismo de ROI permite el paso de objetos como argumentos en las invocaciones, utilizando “paso por valor”. El paso de los objetos por referencia se simula mediante el paso por valor, de forma similar a como se emplea en el mecanismo RPC.
F	D.- En Java RMI, se requiere emplear un lenguaje especial de definición de interfaces, denominado IDL (Interface Definition Language), para describir los interfaces de los objetos remotos
F	E.- El programador debe escribir el código del proxy y esqueleto para cada objeto remoto

4. Dado el siguiente conjunto de eventos en un sistema distribuido, asumiendo que no hay otros eventos previos:



V	A.- Se cumple que f → h
V	B.- Se cumple que a → g
V	C.- Se cumple que d || c
F	D.- Se cumple que a|| e
V	E.- Utilizando solamente el valor de los relojes de Lamport, no podemos determinar si los eventos "a" y "e" son concurrentes entre sí.

5. Respecto a la figura anterior:

F	A.- En "a" el reloj de Lamport C(a) = 0
V	B.- En "e" el reloj de Lamport C(e) = 4
F	C.- En "f" el reloj vectorial V(f) = [1,1,1]
V	D.- En "c" el reloj vectorial V(c) = [3,0,1]
V	E.- En "h" el reloj vectorial V(h) = [2,2,3]

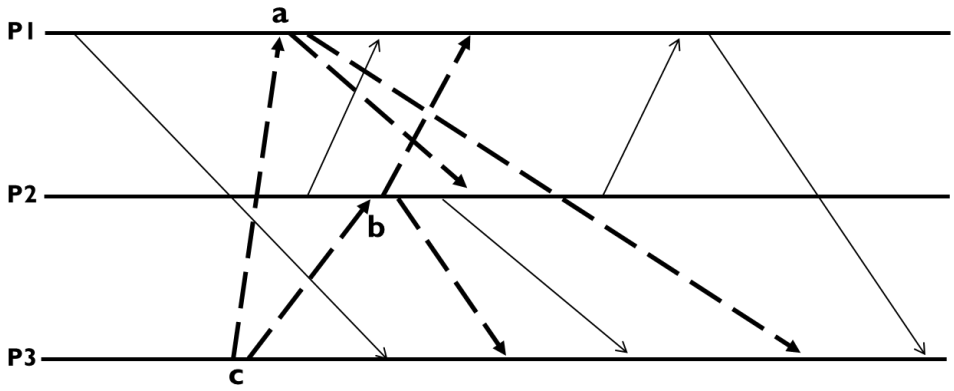
6. Respecto a los algoritmos de sincronización:

V	A.- El algoritmo de Chandy-Lamport requiere que los canales no pierdan ni desordenen mensajes
F	B.- El algoritmo de Bully falla si dos o más nodos inician el algoritmo de forma simultánea (es decir, si hay más de un iniciador)
F	C.- El algoritmo de elección de líder sobre anillos falla si dos o más nodos inician el algoritmo de forma simultánea (es decir, si hay más de un iniciador)
V	D.- El algoritmo centralizado de exclusión mutua visto en clase limita la escalabilidad y la tolerancia a fallos porque el coordinador supone un cuello de botella y un punto de fallo único
V	E.- El algoritmo distribuido de exclusión mutua visto en clase utiliza relojes lógicos para ordenar algunas solicitudes de entrada a la sección crítica.

7. Sobre el espacio de nombres y la resolución de nombres:

V	A.- Para el espacio de nombres suele utilizarse un esquema jerárquico (de tipo árbol).
V	B.- Algunos servicios de nombres permiten asociar atributos a las entidades y buscar por atributos.
V	C.- Por escalabilidad, suelen utilizarse varios servidores de nombres, que se ocupan de directorios distintos.
F	D.- La resolución de nombres iterativa supone que cada servidor debe resolver su parte y pasar el resto al siguiente servidor en la jerarquía.
F	E.- La resolución de nombres recursiva impone mayor responsabilidad al cliente, aligerando la carga sobre los servidores.

8. Respecto a la siguiente figura, suponga que P3 inicia el algoritmo de Chandy-Lamport en "c", siendo las líneas discontinuas los mensajes que se envían como consecuencia de la ejecución de este algoritmo y las líneas continuas los mensajes normales. Cuando finalice el algoritmo:



V	A.- Los estados locales de P1 en "a", P2 en "b" y P3 en "c" formarán parte del estado global
F	B.- Se habrá obtenido en este caso un corte inconsistente, pues alguno de los mensajes registrados como recibidos todavía no ha sido enviado.
V	C.- En el canal (P1,P3) se habrá registrado 1 mensaje
F	D.- En el canal (P2,P3) se habrá registrado 1 mensaje
F	E.- En el canal (P2,P1) se habrán registrado 2 mensajes

9. Sobre el servicio de localización:

V	A.- Dado un identificador, retorna una dirección.
F	B.- El modelo de difusión escala bien en distancia.
F	C.- En el modelo de punteros adelante, el servicio de recolección de residuos (“garbage-collection”) permite acortar la longitud de las cadenas de punteros, y por lo tanto agilizar accesos posteriores
F	D.- En el modelo de difusión, asumiendo que el nodo A contiene la entidad a buscar, el fallo de un nodo distinto de A impide obtener la dirección de dicha entidad.
V	E.- En el modelo de punteros adelante, asumiendo que el nodo A contiene la entidad a buscar, entonces el fallo de un nodo distinto de A que forma parte de la cadena de punteros impide obtener la dirección de dicha entidad.

10. Sobre las arquitecturas de un sistema distribuido:

V	A.- Las arquitecturas de capas y las arquitecturas basadas en objetos son ejemplos de estilos arquitectónicos software.
F	B.- La arquitectura de sistema define cuál es la organización lógica de los componentes, cómo están organizados y cómo deberían interactuar entre sí.
V	C.- Las arquitecturas de sistema se dividen en arquitecturas centralizadas (que emplean distribución vertical), arquitecturas descentralizadas (que emplean distribución horizontal) y arquitecturas híbridas (que combinan distribución vertical y horizontal).
V	D.- Los sistemas peer-to-peer son un ejemplo de arquitectura descentralizada con distribución horizontal.
V	E.- La distribución horizontal se da cuando un cliente o servicio puede estar físicamente dividido en partes equivalentes a nivel lógico, pero cada parte funcionando en un nodo distinto.

EXAMEN 2do PARCIAL – Prácticas 3 y 4 **9 de Junio de 2014**
Concurrencia y Sistemas Distribuidos **Modelo A**

Este bloque tiene una puntuación máxima de **10 puntos** (que aportará 1.5 puntos a la nota final de la asignatura). Indique, para cada una de las siguientes 20 afirmaciones, si éstas son verdaderas (**V**) o falsas (**F**). **Cada respuesta vale: correcta= 0.5, errónea= -0.5, vacía=0.**

Importante: Los primeros **2 errores** no penalizarán, de modo que tendrán una valoración equivalente a la de una respuesta vacía. A partir del 3º error (inclusive), sí se aplicará el decremento por respuesta errónea.

1. Sobre la práctica de Servicios de Dominio de Active Directory AD DS:

F	A.- El administrador de un dominio puede acceder a todas las máquinas de su dominio y subdominios, pero el administrador de un subdominio no puede acceder a ninguna máquina del dominio padre
F	B.- Los recursos compartidos pueden organizarse en unidades administrativas anidadas.
F	C.- Por defecto, cuando se crea una carpeta compartida, tanto el acceso local como el acceso remoto a la misma permiten las mismas operaciones.
F	D.- Sólo se pueden compartir carpetas dentro del mismo subdominio (ej. dentro de *amsterdam*)
F	E.- Los servicios AD DS, gestionados por el controlador de dominio, pueden ser instalados tanto en un sistema Windows Server, como en un sistema Windows 7, en Linux o en Mac OS.

2. Sobre la práctica de Servicios de Dominio de Active Directory AD DS:

V	A.- Se recomienda, pero no se requiere, que exista más de un controlador de dominio en cada dominio.
V	B.- La estructura de dominios de una organización puede estar compuesta por varios árboles.
F	C.- El usuario *eovic\Administrador* no puede eliminar un fichero creado por el usuario *amsterdam\Administrador* con los permisos asignados por defecto.
F	D.- Un usuario del subdominio *amsterdam* no puede a acceder a un recurso compartido de una máquina que no pertenece a dicho subdominio.
V	E.- Un usuario sólo puede pertenecer a una unidad organizativa.

3. Sobre la práctica del Chat distribuido orientado a objetos basado en RMI:

V	A.- El objeto canal mantiene una colección de usuarios y para difundir un mensaje invoca sobre ellos un método de forma remota.
F	B.- Los canales distribuyen los mensajes de forma diferente a los usuarios conectados, dependiendo de si son usuarios de tipo ChatClient o de tipo ChatRobot.
F	C.- Como se indica en el documento de la práctica, cuando un ChatRobot se conecta al servidor obtiene la lista de canales disponibles, a partir de la cual el usuario selecciona de forma interactiva el canal deseado.
F	D.- Un ChatClient se debe lanzar tras el ChatServer. Sin embargo, como el ChatRobot no necesita interactuar con el usuario, el ChatRobot podría lanzarse antes del ChatServer.
F	E.- Cuando se lanzan en distintas máquinas las aplicaciones ChatServer, ChatClient, o ChatRobot, deben tener asignados puertos (*ports*) diferentes.

4. Sobre la práctica del Chat distribuido orientado a objetos basado en RMI:

V	A.- Si lanzamos más de una aplicación ChatClient o ChatRobot en la misma máquina, ambas aplicaciones obligatoriamente deben tener valores *myport* distintos.
F	B.- Todo ChatClient debe recibir como parámetros la ubicación (host y puerto) del ChatServer.
F	C.- ChatClient no abre ningún puerto, puesto que no es un servidor.
F	D.- En la aplicación del Chat distribuido, ChatServer debe registrar primero su objeto principal “ChatServer” en el servidor de nombres, indicando también en dicho servidor el listado de canales que se ofrecen en el chat.
V	E.- La aplicación “ChatServer”, para poder registrar su objeto remoto, localiza primero el servidor de nombres (usando LocateRegistry.getRegistry) y luego registra el servicio (usando el método *bind* o el método *rebind* de la interfaz Registry).

EXAMEN 2do PARCIAL – Bloque Unidades Didácticas 7 a 11 9 de Junio de 2014
Concurrencia y Sistemas Distribuidos Modelo B

APELLIDOS:				**NOMBRE:**	
DNI:		**GRUPO:**		**FIRMA:**	

Este bloque tiene una puntuación máxima de **10 puntos**, que equivalen a 2.5 puntos de la nota final de la asignatura. Indique, para cada una de las siguientes 50 afirmaciones, si éstas son verdaderas (**V**) o falsas (**F**). **Cada respuesta vale: correcta= 0.2, errónea= -0.2, vacía=0.**
Importante: Los primeros **3 errores** no penalizarán, de modo que tendrán una valoración equivalente a la de una respuesta vacía. A partir del 4º error (inclusive), sí se aplicará el decremento por respuesta errónea.

1. Sobre el mecanismo de comunicación ROI:

F	A.- El mecanismo de ROI permite el paso de objetos como argumentos en las invocaciones, utilizando "paso por valor". El paso de los objetos por referencia se simula mediante el paso por valor, de forma similar a como se emplea en el mecanismo RPC.
F	B.- El programador debe escribir el código del proxy y esqueleto para cada objeto remoto.
V	C.- El cliente obtiene un proxy para invocar al objeto remoto.
V	D.- El proxy incluye una referencia al objeto remoto.
F	E.- En Java RMI, se requiere emplear un lenguaje especial de definición de interfaces, denominado IDL (Interface Definition Language), para describir los interfaces de los objetos remotos.

2. Respecto a los algoritmos de sincronización:

F	A.- El algoritmo de Bully falla si dos o más nodos inician el algoritmo de forma simultánea (es decir, si hay más de un iniciador).
V	B.- El algoritmo de Chandy-Lamport requiere que los canales no pierdan ni desordenen mensajes.
V	C.- El algoritmo distribuido de exclusión mutua visto en clase utiliza relojes lógicos para ordenar algunas solicitudes de entrada a la sección crítica.
F	D.- El algoritmo de elección de líder sobre anillos falla si dos o más nodos inician el algoritmo de forma simultánea (es decir, si hay más de un iniciador).
V	E.- El algoritmo centralizado de exclusión mutua visto en clase limita la escalabilidad y la tolerancia a fallos porque el coordinador supone un cuello de botella y un punto de fallo único.

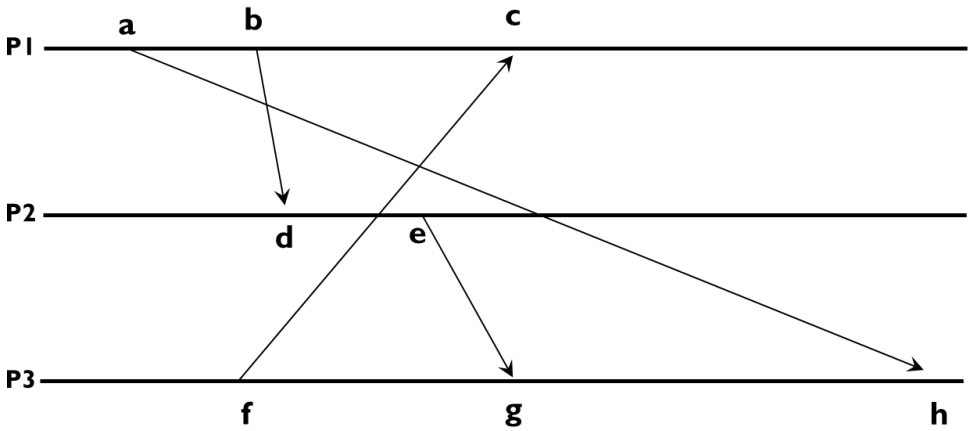
3. Sobre los sistemas distribuidos:

V	A.- Un sistema distribuido ofrece la imagen de un sistema coherente y único.
F	B.- Todos los sistemas de tiempo real son ejemplos de sistemas distribuidos.
F	C.- Si se desea que un sistema distribuido ofrezca escalabilidad de tamaño, no se debe hacer uso de técnicas de replicación, ya que al actualizar el estado de un componente tendremos que propagar tal actualización a todas las réplicas.
V	D.- Algunos tipos de transparencia pueden comprometer la eficiencia del sistema, por ejemplo introduciendo retardos en las interacciones entre componentes de un sistema distribuido.
F	E.- Los sistemas distribuidos proporcionan diferentes tipos de transparencia. Entre ellos: transparencia en el rendimiento, en la escalabilidad, en su disponibilidad, en la seguridad...

4. Sobre el mecanismo de comunicación RPC:

F	A.- Se basa en un modelo de invocación/respuesta con asincronía y persistencia.
F	B.- Al igual que ROI, siempre requiere que se utilice un servidor de nombres.
F	C.- El programador debe escribir el código de los stubs cliente y servidor para cada procedimiento que pueda invocarse de forma remota.
V	D.- El stub cliente sirve para poder invocar procedimientos remotos como si fuesen locales.
F	E.- Sólo podemos definir como procedimientos remotos aquellos que no utilicen paso de parámetros por referencia.

5. Dado el siguiente conjunto de eventos en un sistema distribuido, asumiendo que no hay otros eventos previos:



V	A.- En el evento “e” el reloj de Lamport es C(e) = 4
F	B.- En el evento “a” el reloj de Lamport es C(a) = 0
V	C.- En el evento “c” el reloj vectorial es V(c) = [3,0,1]
F	D.- En el evento “f” el reloj vectorial es V(f) = [1,1,1]
V	E.- En el evento “h” el reloj vectorial es V(h) = [2,2,3]

6. Respecto a la figura anterior:

F	A.- Se cumple que a|| e
V	B.- Se cumple que d || c
V	C.- Se cumple que f → h
V	D.- Se cumple que a → g
V	E.- Utilizando solamente el valor de los relojes de Lamport, no podemos determinar si los eventos “a” y “e” son concurrentes entre sí.

7. Sobre las arquitecturas de un sistema distribuido:

V	A.- La distribución horizontal se da cuando un cliente o servicio puede estar físicamente dividido en partes equivalentes a nivel lógico, pero cada parte funcionando en un nodo distinto.
V	B.- Las arquitecturas de capas y las arquitecturas basadas en objetos son ejemplos de estilos arquitectónicos software.
V	C.- Las arquitecturas de sistema se dividen en arquitecturas centralizadas (que emplean distribución vertical), arquitecturas descentralizadas (que emplean distribución horizontal) y arquitecturas híbridas (que combinan distribución vertical y horizontal).
F	D.- La arquitectura de sistema define cuál es la organización lógica de los componentes, cómo están organizados y cómo deberían interactuar entre sí.
V	E.- Los sistemas peer-to-peer son un ejemplo de arquitectura descentralizada con distribución horizontal.

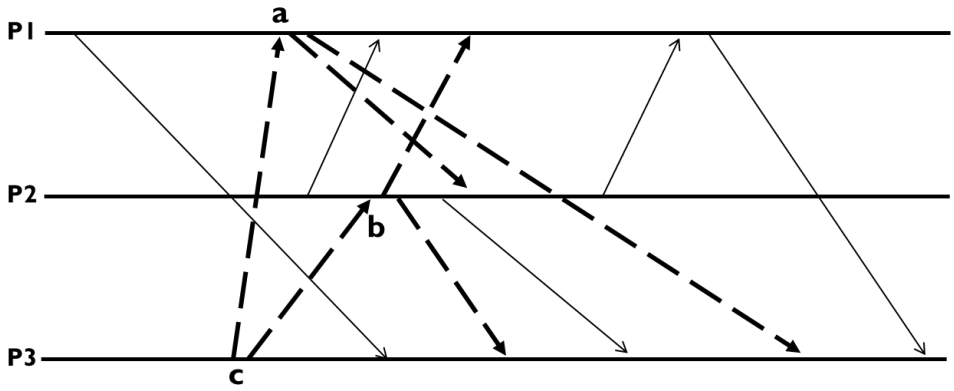
8. Sobre el servicio de localización:

F	A.- En el modelo de punteros adelante, el servicio de recolección de residuos (“garbage-collection”) permite acortar la longitud de las cadenas de punteros, y por lo tanto agilizar accesos posteriores
V	B.- En el modelo de punteros adelante, asumiendo que el nodo A contiene la entidad a buscar, entonces el fallo de un nodo distinto de A que forma parte de la cadena de punteros impide obtener la dirección de dicha entidad.
V	C.- Dado un identificador, retorna una dirección.
F	D.- En el modelo de difusión, asumiendo que el nodo A contiene la entidad a buscar, el fallo de un nodo distinto de A impide obtener la dirección de dicha entidad.
F	E.- El modelo de difusión escala bien en distancia.

9. Sobre el espacio de nombres y la resolución de nombres:

V	A.- Algunos servicios de nombres permiten asociar atributos a las entidades y buscar por atributos.
F	B.- La resolución de nombres iterativa supone que cada servidor debe resolver su parte y pasar el resto al siguiente servidor en la jerarquía.
V	C.- Para el espacio de nombres suele utilizarse un esquema jerárquico (de tipo árbol).
V	D.- Por escalabilidad, suelen utilizarse varios servidores de nombres, que se ocupan de directorios distintos.
F	E.- La resolución de nombres recursiva impone mayor responsabilidad al cliente, aligerando la carga sobre los servidores.

10. Respecto a la siguiente figura, suponga que P3 inicia el algoritmo de Chandy-Lamport en **c**, siendo las líneas discontinuas los mensajes que se envían como consecuencia de la ejecución de este algoritmo y las líneas continuas los mensajes normales. Cuando finalice el algoritmo:



F	A.- En el canal (P2,P3) se habrá registrado 1 mensaje.
V	B.- En el canal (P1,P3) se habrá registrado 1 mensaje.
F	C.- En el canal (P2,P1) se habrán registrado 2 mensajes.
F	D.- Se habrá obtenido en este caso un corte inconsistente, pues alguno de los mensajes registrados como recibidos todavía no ha sido enviado.
V	E.- Los estados locales de P1 en “a”, P2 en “b” y P3 en “c” formarán parte del estado global.

EXAMEN 2do PARCIAL – Prácticas 3 y 4 **9 de Junio de 2014**
Concurrencia y Sistemas Distribuidos **Modelo B**

Este bloque tiene una puntuación máxima de **10 puntos** (que aportará 1.5 puntos a la nota final de la asignatura). Indique, para cada una de las siguientes 20 afirmaciones, si éstas son verdaderas (**V**) o falsas (**F**). **Cada respuesta vale: correcta= 0.5, errónea= -0.5, vacía=0.**

Importante: Los primeros **2 errores** no penalizarán, de modo que tendrán una valoración equivalente a la de una respuesta vacía. A partir del 3º error (inclusive), sí se aplicará el decremento por respuesta errónea.

1. Sobre la práctica de Servicios de Dominio de Active Directory AD DS:

V	A.- Se recomienda, pero no se requiere, que exista más de un controlador de dominio en cada dominio.
F	B.- El administrador de un dominio puede acceder a todas las máquinas de su dominio y subdominios, pero el administrador de un subdominio no puede acceder a ninguna máquina del dominio padre.
F	C.- El usuario *eovic\Administrador* no puede eliminar un fichero creado por el usuario *amsterdam\Administrador* con los permisos asignados por defecto.
F	D.- Un usuario del subdominio *amsterdam* no puede a acceder a un recurso compartido de una máquina que no pertenece a dicho subdominio.
F	E.- Sólo se pueden compartir carpetas dentro del mismo subdominio (ej. dentro de *amsterdam*).

2. Sobre la práctica de Servicios de Dominio de Active Directory AD DS:

F	A.- Los recursos compartidos pueden organizarse en unidades administrativas anidadas.
F	B.- Por defecto, cuando se crea una carpeta compartida, tanto el acceso local como el acceso remoto a la misma permiten las mismas operaciones.
V	C.- La estructura de dominios de una organización puede estar compuesta por varios árboles.
V	D.- Un usuario sólo puede pertenecer a una unidad organizativa.
F	E.- Los servicios AD DS, gestionados por el controlador de dominio, pueden ser instalados tanto en un sistema Windows Server, como en un sistema Windows 7, en Linux o en Mac OS.

3. Sobre la práctica del Chat distribuido orientado a objetos basado en RMI:

F	A.- Cuando se lanzan en distintas máquinas las aplicaciones ChatServer, ChatClient, o ChatRobot, deben tener asignados puertos (*ports*) diferentes.
V	B.- El objeto canal mantiene una colección de usuarios y para difundir un mensaje invoca sobre ellos un método de forma remota.
F	C.- En la aplicación del Chat distribuido, ChatServer debe registrar primero su objeto principal "ChatServer" en el servidor de nombres, indicando también en dicho servidor el listado de canales que se ofrecen en el chat.
F	D.- Como se indica en el documento de la práctica, cuando un ChatRobot se conecta al servidor obtiene la lista de canales disponibles, a partir de la cual el usuario selecciona de forma interactiva el canal deseado.
V	E.- La aplicación "ChatServer", para poder registrar su objeto remoto, localiza primero el servidor de nombres (usando LocateRegistry.getRegistry) y luego registra el servicio (usando el método *bind* o el método *rebind* de la interfaz Registry).

4. Sobre la práctica del Chat distribuido orientado a objetos basado en RMI:

V	A.- Si lanzamos más de una aplicación ChatClient o ChatRobot en la misma máquina, ambas aplicaciones obligatoriamente deben tener valores *myport* distintos.
F	B.- ChatClient no abre ningún puerto, puesto que no es un servidor.
F	C.- Los canales distribuyen los mensajes de forma diferente a los usuarios conectados, dependiendo de si son usuarios de tipo ChatClient o de tipo ChatRobot.
F	D.- Un ChatClient se debe lanzar tras el ChatServer. Sin embargo, como el ChatRobot no necesita interactuar con el usuario, el ChatRobot podría lanzarse antes del ChatServer.
F	E.- Todo ChatClient debe recibir como parámetros la ubicación (host y puerto) del ChatServer.

EXAMEN PARCIAL – Bloque Unidades Didácticas 7 a 11 (04/06/2013)

APELLIDOS:		NOMBRE:	
DNI:		FIRMA:	

Este bloque tiene una puntuación máxima de **10 puntos** (que aportará 2.5 puntos a la nota global).

Indique, para cada una de las siguientes 50 afirmaciones, si éstas son verdaderas (V) o falsas (F). **Cada respuesta vale: correcta= 0.2, errónea= -0.2, vacía=0.** Al final de cada pregunta dispone de un espacio reservado para justificar mejor su respuesta, en caso de considerarlo necesario.

1. Un sistema distribuido:

V	Es una clase de sistema concurrente; aunque no todos los sistemas concurrentes son distribuidos.
V	Ofrece la imagen de un sistema coherente y único.
F	Proporcionará diferentes tipos de transparencia. Entre ellos: transparencia en el rendimiento, en la escalabilidad, en su disponibilidad, en la seguridad...
F	Es un sistema de tiempo real que necesitará un análisis de planificabilidad.

JUSTIFICACIÓN:

2. La comunicación basada en mensajes:

F	Es un tipo particular de memoria compartida y requiere acceso en exclusión mutua.
F	Será sincrónica cuando el canal pueda mantener los mensajes durante un intervalo indefinido.
V	Se utiliza en su variante asincrónica para implantar el servicio de correo electrónico.
V	Se utiliza en su variante sincrónica no persistente para implantar las llamadas a procedimiento remoto.

JUSTIFICACIÓN:

3. El algoritmo de Cristian:

F	Proporciona la base necesaria para implantar cualquier algoritmo descentralizado.
V	Permite sincronizar el reloj local de un nodo cliente con el mantenido por un nodo servidor.
F	Es uno de los algoritmos de elección de líder más eficientes.
F	Permite identificar los mensajes en tránsito por cada uno de los canales de comunicación de un sistema distribuido.

JUSTIFICACIÓN:

4. Sobre los relojes lógicos de Lamport:

F	Son necesarios para deshacer los empates en el algoritmo de Chandy y Lamport.
V	Ordenan de manera parcial los eventos que hayan ocurrido en un sistema distribuido.
F	Permiten determinar en todos los casos si dos eventos de una traza han sido concurrentes o no.
V	Si C(a)=C(b), entonces a||b.

JUSTIFICACIÓN:

5. Sobre los servicios de nombres en un sistema distribuido:

F	Se necesitan para asegurar la exclusión mutua.
V	Suelen utilizar una estructura jerárquica para facilitar su escalabilidad.
V	Pueden retornar identificadores para facilitar la gestión de entidades móviles.
V	LDAP es un ejemplo de servicio de directorio basado en atributos.

JUSTIFICACIÓN:

6. Un sistema “peer-to-peer”:

F	Es un ejemplo de arquitectura software en niveles.
V	Emplea una arquitectura de sistema basada en distribución horizontal.
F	Es un ejemplo de arquitectura de sistema centralizada.
V	Es un ejemplo de sistema distribuido con buenas escalabilidades de tamaño y distancia.

JUSTIFICACIÓN:

7. Sobre tiempo lógico:

V	Si “a ---> b”, entonces C(a) < C(b).
V	Si “a ---> b”, entonces la ejecución del evento “a” siempre sucederá antes que la ejecución del evento “b”.
F	Si “a||b”, entonces la ejecución de los eventos “a” y “b” siempre sucedará en el mismo instante de tiempo.
V	Sean VT(a)=[3,4,4] y VT(b)=[5,4,8] los relojes vectoriales de dos eventos “a” y “b” en un sistema formado por tres procesos P1, P2 y P3. Podemos afirmar que “b” no ha sido ejecutado por el proceso P2.

JUSTIFICACIÓN:

8. Sobre el mecanismo de invocación a objeto remoto (ROI):

V	Es el utilizado en Java RMI.
V	Proporciona transparencia de ubicación.
V	Admite paso de parámetros por referencia en sus invocaciones.
F	Proporciona transparencia de fallos.

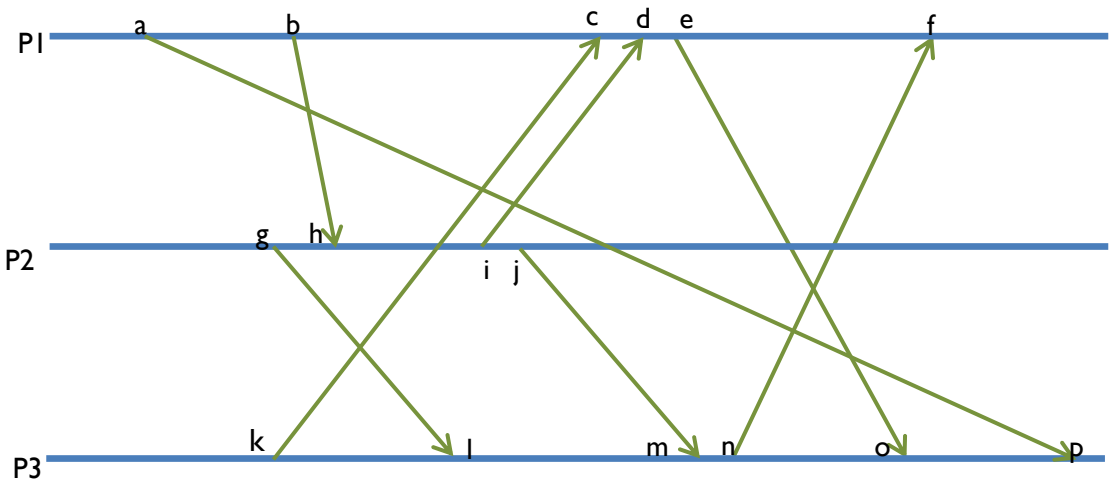
JUSTIFICACIÓN:

9. Sobre los algoritmos descritos en el tema 9 (“Sincronización en sistemas distribuidos”):

F	Se describieron dos algoritmos basados en anillos: uno de exclusión mutua y otro de recolección del estado global.
V	El algoritmo de Chandy y Lamport recoge el estado global del sistema.
F	El algoritmo Bully resuelve el problema de la exclusión mutua en un sistema distribuido.
V	Algunos algoritmos de exclusión mutua pueden utilizar relojes lógicos e identificadores para deshacer los empates que surjan.

JUSTIFICACIÓN:

10. Dado el siguiente conjunto de eventos en un sistema distribuido, asumiendo que no hay otros eventos previos:



V	El reloj vectorial de “p” es VT(p)=[5,4,6] y el de “f” es VT(f)=[6,4,4].
V	Los eventos “c” y “m” son concurrentes.
V	El reloj de Lamport de “d” es C(d)=5 y el de “m” es C(m)=6.
V	A partir de la figura podemos afirmar que “c ---> o” (“c ocurre antes que o”), pues existe un camino dirigido que va desde “c” hasta “o”.
V	Si el reloj vectorial de un evento “x” fuera VT(x)=[5,4,1] y el de “f” fuese VT(f)=[6,4,4], entonces podríamos afirmar que “x ---> f”.

JUSTIFICACIÓN:

11. Sobre la gestión de recursos:

V	Cuando una entidad pasa a no tener ninguna referencia en el sistema, se convierte en un residuo y debería ser localizada y eliminada.
V	El nombre de un punto de entrada es una dirección.
F	Las direcciones no pueden reutilizarse.
F	Un subservicio de localización guarda las correspondencias entre nombres e identificadores.

JUSTIFICACIÓN:

12. Los sistemas “cloud”:

V	Soportan el paradigma de “pago por uso”.
V	Son un ejemplo de arquitectura de sistema descentralizada.
F	Son un ejemplo de arquitectura software orientada a objetos.
F	Son un ejemplo de arquitectura software híbrida.
V	Permiten instalar y/o utilizar aplicaciones distribuidas sin necesidad de realizar una gran inversión en *hardware*.

JUSTIFICACIÓN:

APELLIDOS:		NOMBRE:	
DNI:		FIRMA:	

Este bloque tiene una puntuación máxima de **10 puntos** (que aportará 0.5 puntos a la nota global)**.**

Indique, para cada una de las siguientes 10 afirmaciones, si éstas son verdaderas (V) o falsas (F). **Cada respuesta vale: correcta= 1, errónea= -1, vacía=0.** Al final de cada pregunta dispone de un espacio reservado para justificar mejor su respuesta, en caso de considerarlo necesario.

1. Sobre la práctica de los filósofos comensales:

F	Se ha implementado una solución al problema, basada en la asimetría entre los filósofos que resuelve el problema rompiendo la condición de exclusión mutua.
V	Se ha implementado una solución al problema, basada en la asimetría entre los filósofos que resuelve el problema rompiendo la condición de espera circular.
F	La solución basada en la clase PhiloBothOrNone resuelve el problema de interbloqueos rompiendo la condición de no expropiación.
V	La solución basada en la clase PhiloBothOrNone resuelve el problema de interbloqueos rompiendo la condición de retención y espera.
V	Se ha desarrollado una solución al problema, basada en la limitación del número de comensales que resuelve el problema de interbloqueos rompiendo la condición de espera circular.

JUSTIFICACIÓN:

2. Sobre la práctica 4:

V	Los procesos ChatServer y `rmiregistry` pueden arrancar en el mismo ordenador.
F	El primer paso de todo cliente es conectarse al servidor de chat, tras lo cual contacta con el servidor de nombres.
V	Podemos lanzar varios clientes en una misma máquina o en máquinas diferentes.
V	Los ChatClient deben obtener la lista de canales invocando algún método del ChatServer.
V	El proceso ChatClient no se registra en `rmiregistry`.

JUSTIFICACIÓN:

MODELO A

EXAMEN DE RECUPERACIÓN – SEGUNDO BLOQUE **Concurrencia y Sistemas Distribuidos**

Unidades Didácticas 7,8,9 y 11 - Prácticas 3 y 4 **23 de Junio de 2015**

Este examen tiene una duración total de 2 horas.

Este examen tiene una puntuación máxima de **10 puntos**, que equivalen a **4** puntos de la nota final de la asignatura. Indique, para cada una de las siguientes **60 afirmaciones**, si éstas son verdaderas (**V**) o falsas (**F**).

Cada respuesta vale: correcta= 1/6, errónea= -1/6, vacía=0.

Importante: Los **primeros 3 errores** **no penalizarán**, de modo que tendrán una valoración equivalente a la de una respuesta vacía. A partir del 4º error (inclusive), sí se aplicará el decremento por respuesta errónea.

Sobre los sistemas distribuidos:

1.	La disponibilidad, una de las características de los sistemas distribuidos, sirve para ocultar las diferencias de los mecanismos de comunicación. *JUSTIFICACIÓN: La disponibilidad implica que los servicios deben estar siempre disponibles (está directamente relacionada con el fallo del sistema, no con la transparencia de acceso).*	F
2.	La transparencia de replicación oculta la coordinación entre las actividades que gestionan un conjunto de recursos para mantener su consistencia. *JUSTIFICACIÓN: La definición dada se corresponde con la transparencia de transacción.*	F
3.	Los sistemas distribuidos basados en Active Directory proporcionan escalabilidad administrativa.	V
4.	La capa de middleware ayuda a conseguir los cuatro objetivos principales de los sistemas distribuidos: descentralización, replicación, transparencia de acceso y escalabilidad. *JUSTIFICACIÓN: Los objetivos indicados son incorrectos. Los cuatro objetivos son: facilitar acceso a los recursos remotos, transparencia de distribución, sistema abierto y escalabilidad.*	F

Respecto a los algoritmos descentralizados:

5.	Si hay un nodo que mantiene toda la información relevante de un algoritmo, pero este nodo toma decisiones basadas en su conocimiento local, entonces el algoritmo es descentralizado. *JUSTIFICACIÓN: Para que sea descentralizado, ningún nodo debe mantener toda la información completa que necesite el algoritmo.*	F
6.	La utilización de algoritmos descentralizados permite ofrecer transparencia de acceso y de ubicación, al estar distribuida la carga del algoritmo entre diferentes nodos. *JUSTIFICACIÓN: Esta distribución de la carga del algoritmo por sí misma no ofrece los tipos de transparencia indicados.*	F
7.	La utilización de algoritmos descentralizados permite mejorar la escalabilidad de distancia.	V
8.	Para mejorar la escalabilidad de tamaño se suele utilizar distribución de responsabilidades, replicación, caching y algoritmos descentralizados.	V

Sobre el mecanismo de comunicación ROI:

9.	Se pueden pasar objetos remotos por referencia.	V
10.	Cuando un cliente referencia por vez primera a un objeto remoto, obtiene el esqueleto para dicho objeto. *JUSTIFICACIÓN: Obtiene el proxy.*	F
11.	El esqueleto ofrece la misma interfaz que el objeto remoto. *JUSTIFICACIÓN: El proxy es quien ofrece la misma interfaz que el objeto remoto.*	F

Sobre el mecanismo de comunicación Java RMI:

12.	Si un objeto que se pasa como argumento implementa la interfaz *Remote*, entonces dicho objeto se pasa por referencia.	V
13.	Para implementar un objeto remoto en Java RMI, la clase de los objetos remotos debe implementar la interfaz remota y extender *java.rmi.server.UnicastRemoteObject*, para así poder registrar los objetos en el ORB de Java.	V
14.	El servidor de nombres de Java RMI guarda, para cada objeto remoto, su nombre simbólico y la dirección (host, puerto) de su proxy. *JUSTIFICACIÓN: El servidor de nombre guarda el nombre simbólico y la referencia.*	F
15.	El servidor de nombres de Java RMI puede residir en cualquier nodo (incluso en el nodo cliente) y es accedido usando la interfaz Registry.	V
16.	Java RMI es un ejemplo de middleware de mensajería, donde cliente y servidor deben suscribirse al middleware para así poder enviarse mensajes entre sí. *JUSTIFICACIÓN: Cliente y servidor no requieren subscripción al middleware. Esto se realiza en Java JMS.*	F

Respecto a los servicios Web RESTful y al mecanismo de comunicación Java Message Service:

17.	Resulta interesante utilizar servicios Web RESTful cuando no es necesario que todos los componentes de la aplicación estén simultáneamente en ejecución. *JUSTIFICACIÓN: La comunicación no es persistente, por lo que los componentes deben estar simultáneamente en ejecución.*	F
18.	Los servicios Web RESTful utilizan métodos del protocolo http para indicar el tipo de operación.	V
19.	En JMS, los mensajes no son estructurados y se envían en texto plano en XML. *JUSTIFICACIÓN: Sí son estructurados, con cabecera, contenido y campos predefinidos.*	F
20.	JMS generalmente utiliza comunicación indirecta.	V
21.	Los objetos que implementan la interfaz JMSContext se crean a partir de una factoría de conexiones.	V

Sobre los algoritmos vistos en esta asignatura de exclusión mutua y elección de líder para sistemas distribuidos:

22.	El tiempo de propagación de los mensajes debe ser conocido en el algoritmo Bully para decidir hasta cuándo se esperará que el token regrese al nodo iniciador. *JUSTIFICACIÓN: En el algoritmo Bully no se utiliza ningún token.*	F
23.	En el algoritmo centralizado de exclusión mutua un nodo accede a la sección crítica cuando han dado su permiso todos los demás nodos. *JUSTIFICACIÓN: La explicación se corresponde con el algoritmo distribuido de exclusión mutua.*	F
24.	En el algoritmo de exclusión mutua para anillos no se admite que haya simultáneamente más de un token para una misma sección crítica.	V
25.	Los algoritmos de elección de líder gestionan correctamente la situación en la que los relojes físicos de los nodos que participan en el algoritmo no están sincronizados. *JUSTIFICACIÓN: En estos algoritmos solamente se requiere conocer el identificador de los nodos. No se utilizan relojes de ningún tipo.*	V

26.	Alguno de los algoritmos de exclusión mutua vistos en la asignatura requiere que el sistema utilice relojes vectoriales para ordenar sus eventos. *JUSTIFICACIÓN: Se requiere el uso de relojes lógicos de Lamport y de los identificadores de los nodos para poder establecer un orden total de los eventos.*	F
27.	En el algoritmo de exclusión mutua para anillos, cuando un nodo quiere entrar en la sección crítica, envía un mensaje TRY al resto de nodos para que el propietario actual del token se lo envíe una vez haya terminado su sección crítica. *JUSTIFICACIÓN: El token se va pasando por el anillo y quien tiene el token puede entrar en la sección crítica.*	F

Sobre los algoritmos vistos en esta asignatura de sincronización de relojes físicos:

28.	El algoritmo de Cristian requiere un nodo con un reloj más exacto que el resto.	V
29.	Una vez finalizado el algoritmo de Cristian, se ajusta la hora del nodo que actúa como servidor. JUSTIFICACIÓN: Esto ocurre en el algoritmo de Berkeley.	F
30.	Resulta adecuado utilizar el algoritmo de Berkeley para conseguir que todos los relojes de los nodos de un sistema distribuido se sincronicen entre sí en un momento determinado.	V

Respecto a los relojes lógicos:

31.	Supongamos dos eventos `a,b` en un mismo nodo, etiquetados respectivamente con valores lógicos `C(a)` y `C(b)`: si `C(a) < C(b)` podemos afirmar que `a->b`	V
32.	Supongamos dos eventos `a,b` en nodos distintos, etiquetados respectivamente con valores lógicos `C(a)` y `C(b)`: si `C(a) < C(b)` podemos afirmar que `a->b`	F
33.	Supongamos dos eventos `a,b` en nodos distintos, etiquetados respectivamente con valores lógicos `C(a)` y `C(b)`: si `a->b` podemos afirmar que `C(a) < C(b)`	V

Dados 3 eventos `a,b,c` etiquetados respectivamente con relojes vectoriales con valores `Va=[4,1,2], Vb=[3,0,5], Vc=[3,1,6]:`

34.	Podemos afirmar que `a || c` *JUSTIFICACIÓN: No se cumple que Va<Vc ni tampoco que Vc<Va, por lo que ambos eventos son concurrentes.*	V
35.	Aunque los eventos `b` y `c` ocurran en nodos distintos, podemos afirmar que `b->c` *JUSTIFICACIÓN: Se cumple que Vb<Vc por lo que b->c*	V
36.	A diferencia de los relojes lógicos de Lamport, no se cumple la transitividad (si `m->n` y `n->p`, no implica `m->p`).	F

Respecto al algoritmo de Chandy-Lamport para la obtención del estado global del sistema:

37.	Un corte se considera inconsistente si incluye mensajes enviados y todavía no recibidos. *JUSTIFICACIÓN: La explicación dada se corresponde con un corte consistente.*	F
38.	Se requiere conectividad total: entre cada par de nodos a,b deben existir canales unidireccionales en sentido a-b y b-a.	V
39.	Únicamente envía mensajes MARCA el nodo iniciador. *JUSTIFICACIÓN: Cuando un nodo recibe un mensaje MARCA y no ha guardado aún su estado local, entonces envía MARCA al resto de nodos.*	F
40.	Cuando un proceso recibe un mensaje MARCA, debe guardar su estado local sólo si es la primera vez que recibe dicho mensaje.	V

Respecto a la arquitectura de los sistemas distribuidos:

41.	A la hora de diseñar un sistema distribuido, debemos considerar cuál debe ser su arquitectura software, ya que ésta define qué tipo de componentes necesitamos y cómo se relacionan entre sí dichos componentes.	V
42.	Las arquitecturas de sistema se dividen en arquitecturas centralizadas, con distribución horizontal, y arquitecturas descentralizadas, con distribución vertical. *JUSTIFICACIÓN: Es al revés. Se dividen en: arquitecturas centralizadas, con distribución vertical, y arquitecturas descentralizadas, con distribución horizontal.*	F

Respecto a los sistemas peer-to-peer, los sistemas Grid y los sistemas Cloud:

43.	En la localización de ficheros, las arquitecturas *peer-to-peer parcialmente centralizadas* ofrecen peores tiempos de búsqueda de ficheros que las *puramente descentralizadas*, al poder propagarse la búsqueda por todos los nodos del sistema. *JUSTIFICACIÓN: Al revés, las puramente descentralizadas ofrecen peores tiempos de búsqueda, por el motivo que se indica.*	F
44.	Los sistemas peer-to-peer, los sistemas Grid y los sistemas Cloud ofrecen escalabilidad de distancia y de tamaño.	V
45.	Los sistemas Grid permiten ejecutar aplicaciones con elevados requisitos en cuanto a capacidad computacional o volumen de información a procesar.	V
46.	Los sistemas Cloud permiten proporcionar distintos tipos de servicio: software como servicio (SaaS), plataforma como servicio (PaaS), infraestructura como servicio (IaaS) y almacenamiento como servicio.	V

Respecto a la práctica sobre Active Directory:

47.	En cada dominio tiene que haber una o más máquinas, llamadas administradores de dominio, que proporcionan los servicios de dominio. *JUSTIFICACIÓN: Las máquinas se llaman "controladores de dominio"*	F
48.	Para unir una máquina a un dominio hay que utilizar las credenciales de un administrador de dicho dominio.	V
49.	El administrador de una máquina de un dominio tiene permisos de control total sobre cualquier recurso compartido del propio dominio.	F
50.	Los objetos presentes (usuarios, equipos y grupos) en cada dominio se pueden organizar utilizando contenedores y unidades organizativas	V
51.	El controlador del dominio principal de la práctica debía ser puesto en marcha justo a continuación del router virtual.	V
52.	En esta práctica, los usuarios del dominio secundario no podían acceder a los recursos compartidos del dominio principal.	F
53.	Un usuario puede pertenecer a varias unidades organizativas.	F

Respecto a la práctica del Chat distribuido en Java RMI:

54.	Los procesos *ChatClient* registran su objeto *ChatUser* mediante el servidor de nombres (*rmiregistry*). De esa manera el proceso *ChatServer* puede obtener referencias de los usuarios conectados	F
55.	En la aplicación de chat hay invocaciones remotas en las que los dos procesos involucrados son procesos *ChatClient*.	V
56.	Cuando un usuario se une a un canal, el canal lo notifica a todos los usuarios conectados al canal, invocando sobre ellos un método (invocación remota).	V
57.	*rmiregistry* es un componente middleware de java RMI que permite en esta práctica encontrar a todos los usuarios de un canal	F
58.	Los proxy de los objetos los podemos obtener haciendo una operación *lookup* en un objeto de tipo *Registry*.	V
59.	En la práctica, desde el lado del cliente se pueden crear nuevos canales con la interfaz del usuario.	F
60.	Para localizar el objeto *ChatServer* hay que hacer una búsqueda en la que se suministra la dirección IP y el puerto del ordenador donde se ejecuta el *ChatServer*.	F

MODELO B

EXAMEN DE RECUPERACIÓN – SEGUNDO BLOQUE **Concurrencia y Sistemas Distribuidos**

Unidades Didácticas 7,8,9 y 11 - Prácticas 3 y 4 **23 de Junio de 2015**

Este examen tiene una duración total de 2 horas.

Este examen tiene una puntuación máxima de **10 puntos**, que equivalen a **4** puntos de la nota final de la asignatura. Indique, para cada una de las siguientes **60 afirmaciones**, si éstas son verdaderas (**V**) o falsas (**F**).

Cada respuesta vale: correcta= 1/6, errónea= -1/6, vacía=0.

Importante: Los **primeros 3 errores** **no penalizarán**, de modo que tendrán una valoración equivalente a la de una respuesta vacía. A partir del 4º error (inclusive), sí se aplicará el decremento por respuesta errónea.

Sobre los algoritmos vistos en esta asignatura de sincronización de relojes físicos:

1.	Resulta adecuado utilizar el algoritmo de Berkeley para conseguir que todos los relojes de los nodos de un sistema distribuido se sincronicen entre sí en un momento determinado.	V
2.	El algoritmo de Cristian requiere un nodo con un reloj más exacto que el resto.	V
3.	Una vez finalizado el algoritmo de Cristian, se ajusta la hora del nodo que actúa como servidor. JUSTIFICACIÓN: Esto ocurre en el algoritmo de Berkeley.	F

Respecto a los relojes lógicos:

4.	Supongamos dos eventos `a,b` en nodos distintos, etiquetados respectivamente con valores lógicos `C(a)` y `C(b)`: si `a->b` podemos afirmar que `C(a) < C(b)`	V
5.	Supongamos dos eventos `a,b` en nodos distintos, etiquetados respectivamente con valores lógicos `C(a)` y `C(b)`: si `C(a) < C(b)` podemos afirmar que `a->b`	F
6.	Supongamos dos eventos `a,b` en un mismo nodo, etiquetados respectivamente con valores lógicos `C(a)` y `C(b)`: si `C(a) < C(b)` podemos afirmar que `a->b`	V

Dados 3 eventos `a,b,c` etiquetados respectivamente con relojes vectoriales con valores `Va=[4,1,2], Vb=[3,0,5], Vc=[3,1,6]`:

7.	A diferencia de los relojes lógicos de Lamport, no se cumple la transitividad (si `m->n` y `n->p`, no implica `m->p`).	F
8.	Podemos afirmar que `a || c` *JUSTIFICACIÓN: No se cumple que Va<Vc ni tampoco que Vc<Va, por lo que ambos eventos son concurrentes.*	V
9.	Aunque los eventos `b` y `c` ocurran en nodos distintos, podemos afirmar que `b->c` *JUSTIFICACIÓN: Se cumple que Vb<Vc por lo que b->c*	V

Sobre los algoritmos vistos en esta asignatura de exclusión mutua y elección de líder para sistemas distribuidos:

10.	Alguno de los algoritmos de exclusión mutua vistos en la asignatura requiere que el sistema utilice relojes vectoriales para ordenar sus eventos. *JUSTIFICACIÓN: Se requiere el uso de relojes lógicos de Lamport y de los identificadores de los nodos para poder establecer un orden total de los eventos.*	F
11.	En el algoritmo de exclusión mutua para anillos, cuando un nodo quiere entrar en la sección crítica, envía un mensaje TRY al resto de nodos para que el propietario actual del token se lo envíe una vez haya terminado su sección crítica. *JUSTIFICACIÓN: El token se va pasando por el anillo y quien tiene el token puede entrar en la sección crítica.*	F

12.	Los algoritmos de elección de líder gestionan correctamente la situación en la que los relojes físicos de los nodos que participan en el algoritmo no están sincronizados. *JUSTIFICACIÓN: En estos algoritmos solamente se requiere conocer el identificador de los nodos. No se utilizan relojes de ningún tipo.*	V
13.	El tiempo de propagación de los mensajes debe ser conocido en el algoritmo Bully para decidir hasta cuándo se esperará que el token regrese al nodo iniciador. *JUSTIFICACIÓN: En el algoritmo Bully no se utiliza ningún token.*	F
14.	En el algoritmo centralizado de exclusión mutua un nodo accede a la sección crítica cuando han dado su permiso todos los demás nodos. *JUSTIFICACIÓN: La explicación se corresponde con el algoritmo distribuido de exclusión mutua.*	F
15.	En el algoritmo de exclusión mutua para anillos no se admite que haya simultáneamente más de un token para una misma sección crítica.	V

Respecto a los algoritmos descentralizados:

16.	Para mejorar la escalabilidad de tamaño se suele utilizar distribución de responsabilidades, replicación, caching y algoritmos descentralizados.	V
17.	La utilización de algoritmos descentralizados permite mejorar la escalabilidad de distancia.	V
18.	Si hay un nodo que mantiene toda la información relevante de un algoritmo, pero este nodo toma decisiones basadas en su conocimiento local, entonces el algoritmo es descentralizado. *JUSTIFICACIÓN: Para que sea descentralizado, ningún nodo debe mantener toda la información completa que necesite el algoritmo.*	F
19.	La utilización de algoritmos descentralizados permite ofrecer transparencia de acceso y de ubicación, al estar distribuida la carga del algoritmo entre diferentes nodos. *JUSTIFICACIÓN: Esta distribución de la carga del algoritmo por sí misma no ofrece los tipos de transparencia indicados.*	F

Sobre los sistemas distribuidos:

20	La capa de middleware ayuda a conseguir los cuatro objetivos principales de los sistemas distribuidos: descentralización, replicación, transparencia de acceso y escalabilidad. *JUSTIFICACIÓN: Los objetivos indicados son incorrectos. Los cuatro objetivos son: facilitar acceso a los recursos remotos, transparencia de distribución, sistema abierto y escalabilidad.*	F
21	La disponibilidad, una de las características de los sistemas distribuidos, sirve para ocultar las diferencias de los mecanismos de comunicación. *JUSTIFICACIÓN: La disponibilidad implica que los servicios deben estar siempre disponibles (está directamente relacionada con el fallo del sistema, no con la transparencia de acceso).*	F
22	La transparencia de replicación oculta la coordinación entre las actividades que gestionan un conjunto de recursos para mantener su consistencia. *JUSTIFICACIÓN: La definición dada se corresponde con la transparencia de transacción.*	F
23	Los sistemas distribuidos basados en Active Directory proporcionan escalabilidad administrativa.	V

Sobre el mecanismo de comunicación ROI:

24.	El esqueleto ofrece la misma interfaz que el objeto remoto. *JUSTIFICACIÓN: El proxy es quien ofrece la misma interfaz que el objeto remoto.*	F
25.	Se pueden pasar objetos remotos por referencia.	V
26.	Cuando un cliente referencia por vez primera a un objeto remoto, obtiene el esqueleto para dicho objeto. *JUSTIFICACIÓN: Obtiene el proxy.*	F

Sobre el mecanismo de comunicación Java RMI:

27.	Java RMI es un ejemplo de middleware de mensajería, donde cliente y servidor deben suscribirse al middleware para así poder enviarse mensajes entre sí. *JUSTIFICACIÓN: Cliente y servidor no requieren subscripción al middleware. Esto se realiza en Java JMS.*	F
28.	El servidor de nombres de Java RMI puede residir en cualquier nodo (incluso en el nodo cliente) y es accedido usando la interfaz Registry.	V
29.	Si un objeto que se pasa como argumento implementa la interfaz *Remote*, entonces dicho objeto se pasa por referencia.	V
30.	Para implementar un objeto remoto en Java RMI, la clase de los objetos remotos debe implementar la interfaz remota y extender *java.rmi.server.UnicastRemoteObject*, para así poder registrar los objetos en el ORB de Java.	V
31.	El servidor de nombres de Java RMI guarda, para cada objeto remoto, su nombre simbólico y la dirección (host, puerto) de su proxy. *JUSTIFICACIÓN: El servidor de nombre guarda el nombre simbólico y la referencia.*	F

Respecto a los sistemas peer-to-peer, los sistemas Grid y los sistemas Cloud:

32.	Los sistemas peer-to-peer, los sistemas Grid y los sistemas Cloud ofrecen escalabilidad de distancia y de tamaño.	V
33.	Los sistemas Cloud permiten proporcionar distintos tipos de servicio: software como servicio (SaaS), plataforma como servicio (PaaS), infraestructura como servicio (IaaS) y almacenamiento como servicio.	V
34.	En la localización de ficheros, las arquitecturas *peer-to-peer parcialmente centralizadas* ofrecen peores tiempos de búsqueda de ficheros que las *puramente descentralizadas*, al poder propagarse la búsqueda por todos los nodos del sistema. *JUSTIFICACIÓN: Al revés, las puramente descentralizadas ofrecen peores tiempos de búsqueda, por el motivo que se indica.*	F
35.	Los sistemas Grid permiten ejecutar aplicaciones con elevados requisitos en cuanto a capacidad computacional o volumen de información a procesar.	V

Respecto al algoritmo de Chandy-Lamport para la obtención del estado global del sistema:

36.	Únicamente envía mensajes MARCA el nodo iniciador. *JUSTIFICACIÓN: Cuando un nodo recibe un mensaje MARCA y no ha guardado aún su estado local, entonces envía MARCA al resto de nodos.*	F
37.	Cuando un proceso recibe un mensaje MARCA, debe guardar su estado local sólo si es la primera vez que recibe dicho mensaje.	V
38.	Un corte se considera inconsistente si incluye mensajes enviados y todavía no recibidos. *JUSTIFICACIÓN: La explicación dada se corresponde con un corte consistente.*	F

39.	Se requiere conectividad total: entre cada par de nodos a,b deben existir canales unidireccionales en sentido a-b y b-a.	V

Respecto a los servicios Web RESTful y al mecanismo de comunicación Java Message Service:

40.	Resulta interesante utilizar servicios Web RESTful cuando no es necesario que todos los componentes de la aplicación estén simultáneamente en ejecución. *JUSTIFICACIÓN: La comunicación no es persistente, por lo que los componentes deben estar simultáneamente en ejecución.*	F
41.	Los objetos que implementan la interfaz JMSContext se crean a partir de una factoría de conexiones.	V
42.	Los servicios Web RESTful utilizan métodos del protocolo http para indicar el tipo de operación.	V
43.	En JMS, los mensajes no son estructurados y se envían en texto plano en XML. *JUSTIFICACIÓN: Sí son estructurados, con cabecera, contenido y campos predefinidos.*	F
44.	JMS generalmente utiliza comunicación indirecta.	V

Respecto a la arquitectura de los sistemas distribuidos:

45.	Las arquitecturas de sistema se dividen en arquitecturas centralizadas, con distribución horizontal, y arquitecturas descentralizadas, con distribución vertical. *JUSTIFICACIÓN: Es al revés. Se dividen en: arquitecturas centralizadas, con distribución vertical, y arquitecturas descentralizadas, con distribución horizontal.*	F
46.	A la hora de diseñar un sistema distribuido, debemos considerar cuál debe ser su arquitectura software, ya que ésta define qué tipo de componentes necesitamos y cómo se relacionan entre sí dichos componentes.	V

Respecto a la práctica del Chat distribuido en Java RMI:

47.	Cuando un usuario se une a un canal, el canal lo notifica a todos los usuarios conectados al canal, invocando sobre ellos un método (invocación remota).	V
48.	En la práctica, desde el lado del cliente se pueden crear nuevos canales con la interfaz del usuario.	F
49.	*rmiregistry* es un componente middleware de java RMI que permite en esta práctica encontrar a todos los usuarios de un canal	F
50.	Para localizar el objeto *ChatServer* hay que hacer una búsqueda en la que se suministra la dirección IP y el puerto del ordenador donde se ejecuta el *ChatServer*.	F
51.	Los procesos *ChatClient* registran su objeto *ChatUser* mediante el servidor de nombres (*rmiregistry*). De esa manera el proceso *ChatServer* puede obtener referencias de los usuarios conectados	F
52.	En la aplicación de chat hay invocaciones remotas en las que los dos procesos involucrados son procesos *ChatClient*.	V
53.	Los proxy de los objetos los podemos obtener haciendo una operación *lookup* en un objeto de tipo *Registry*.	V

Respecto a la práctica sobre Active Directory:

54.	Un usuario puede pertenecer a varias unidades organizativas.	F
55.	Los objetos presentes (usuarios, equipos y grupos) en cada dominio se pueden organizar utilizando contenedores y unidades organizativas	V
56.	El controlador del dominio principal de la práctica debía ser puesto en marcha justo a continuación del router virtual.	V
57.	En cada dominio tiene que haber una o más máquinas, llamadas administradores de dominio, que proporcionan los servicios de dominio. *JUSTIFICACIÓN: Las máquinas se llaman "controladores de dominio"*	F
58.	Para unir una máquina a un dominio hay que utilizar las credenciales de un administrador de dicho dominio.	V
59.	El administrador de una máquina de un dominio tiene permisos de control total sobre cualquier recurso compartido del propio dominio.	F
60.	En esta práctica, los usuarios del dominio secundario no podían acceder a los recursos compartidos del dominio principal.	F

APELLIDOS:		NOMBRE:	
DNI:		FIRMA:	

EXAMEN FINAL (18 junio 2013) – Bloque segundo parcial teoría

Este bloque tiene una puntuación máxima de **10 puntos.**
Indique, para cada una de las siguientes 50 afirmaciones, si éstas son verdaderas (V) o falsas (F). **Cada respuesta vale: correcta= 0.2, errónea= -0.2, vacía=0.**

1. Un sistema distribuido:

F	Requiere que los relojes de todos sus ordenadores estén sincronizados.
F	Deberá tener siempre una actividad líder, que podrá ser seleccionada empleando algún algoritmo de elección de líder.
F	Requiere que todos sus eventos estén ordenados, empleándose generalmente los relojes lógicos de Lamport para este fin.
V	Será escalable si es capaz de mantener su capacidad de servicio cuando crezca su número de componentes (nodos, procesos, clientes, etc.).

2. Las siguientes son técnicas para aumentar la escalabilidad en sistemas distribuidos:

V	Utilizar replicación.
V	Utilizar algoritmos descentralizados.
V	Delegar parte del procesamiento a los clientes.
V	Repartir tanto las tareas como los datos entre múltiples nodos servidores.

3. El algoritmo de Berkeley:

F	Es un ejemplo de algoritmo descentralizado.
V	Sincroniza los relojes de los nodos de un determinado sistema distribuido, sin importarle la divergencia que pueda haber entre estos relojes y la hora "oficial".
F	Es uno de los algoritmos de exclusión mutua más eficientes.
F	Es el utilizado para actualizar los relojes vectoriales.

4. Los relojes vectoriales:

F	Son necesarios para implantar la transparencia de concurrencia.
F	Son los resultantes del algoritmo de Cristian.
V	Permiten determinar en todos los casos si dos eventos de una ejecución son concurrentes o no.
V	No siempre pueden ordenarse entre sí.

5. Sobre los servicios de nombres en un sistema distribuido:

V	Se necesitan para obtener las direcciones o identificadores de ciertas entidades, cuando se conozca su nombre.
V	Pueden implantar la resolución de nombres de forma recursiva o de forma iterativa.
V	Suelen proporcionar tres operaciones: inserción (o registro), resolución y borrado.
V	LDAP es un ejemplo de servicio de directorio basado en atributos.

6. Un sistema "peer-to-peer":

V	Es un ejemplo de arquitectura de sistema descentralizada.
V	Según el grado de centralización de su arquitectura se distinguen tres variantes: puramente descentralizada, parcialmente centralizada y descentralizada híbrida.
V	La variante descentralizada híbrida centraliza el servicio de localización, pero distribuye otras acciones.
F	Es un tipo particular de sistema Grid.

7. Sobre el mecanismo de llamada a procedimiento remoto (RPC):

F	Proporciona transparencia de persistencia.
F	Utiliza stubs clientes para gestionar la recepción de mensajes de petición y el envío de los mensajes de respuesta.
V	Proporciona transparencia de ubicación.
V	En su variante asincrónica no puede retornar resultados ni argumentos de salida.

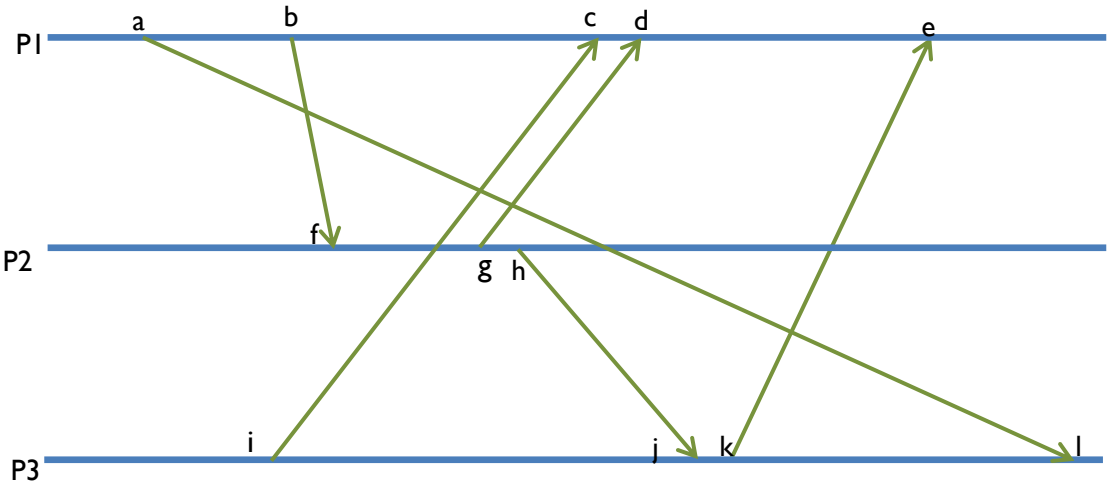
8. Sobre el mecanismo de invocación a objeto remoto (ROI):

V	Es el utilizado en Java RMI.
V	Oculta el envío y recepción de mensajes entre el nodo cliente y el nodo servidor.
F	Siempre utiliza paso de parámetros por valor.
V	Utiliza proxies y esqueletos.

9. Sobre los algoritmos descritos en el tema 9 ("Sincronización en sistemas distribuidos"):

F	El algoritmo de Cristian se utiliza para gestionar relojes lógicos.
F	El algoritmo distribuido de exclusión mutua no necesita realizar ninguna acción en su protocolo de salida.
V	El algoritmo Bully requiere comunicación fiable.
V	El algoritmo de Chandy y Lamport requiere que los canales de comunicación respeten un orden FIFO.

10. Dado el siguiente conjunto de eventos en un sistema distribuido, asumiendo que no hay otros eventos previos:



F	El reloj vectorial de "e" es VT(e)=[5,3,3] y el de "h" es VT(h)=[2,3,1].
F	Los eventos "e" y "f" son concurrentes.
F	El reloj de Lamport de "d" es C(d)=4 y el de "k" es C(k)=7.
F	El reloj de Lamport de "g" es C(g)=4 y el reloj vectorial de ese mismo evento es VT(g)=[2,2,1].
F	Los eventos "b" y "l" son concurrentes.

11. Sobre la gestión de recursos:

F	Cuando una entidad solo tenga una referencia en el sistema (en un servidor de nombres), se convierte en un residuo y debería ser eliminada.
V	El mecanismo de "punteros adelante" puede dejar a un recurso inaccesible cuando falle un ordenador y se "rompa" la cadena de punteros.
V	Los identificadores no podrán reutilizarse nunca.
V	El mecanismo de "punteros adelante" se puede emplear como parte del soporte necesario para reubicar entidades.

12. Los sistemas "cloud":

F	Son un tipo particular de sistemas "peer-to-peer".
V	Permiten implantar aplicaciones distribuidas escalables.
V	Google Drive es un ejemplo de servicio SaaS.
V	Microsof Azure es un ejemplo de servicio PaaS.
V	Amazon EC2 es un ejemplo de servicio IaaS.