

INGENIERÍA DEL SOFTWARE (2º Curso) Cód. 53210 SISTEMAS 54208 GESTIÓN			
	Modelo		
	Junio 2008		Ámbito
			1ª SEMANA

**ESTE EJERCICIO NO ES DE TEST. NO TIENE RETORNO TELEMÁTICO.**

**NECESITO EL EJERCICIO MANUSCRITO POR EL ALUMNO Y LA HOJA DE LECTURA ÓPTICA PARA PODER CALIFICAR.**

**SI ESTE EJERCICIO NO SE HA IMPRESO EN HOJA DE LECTURA ÓPTICA, SOLICITE UNA AL TRIBUNAL.**

Todas las preguntas de este ejercicio son eliminatorias en el sentido de que debe obtener una nota mínima en cada una de ellas. En cada pregunta teórica, que se valora con 2'5 puntos, la nota mínima es 1 punto; en la segunda parte (ejercicio de teoría aplicada que se valora con 5 puntos) la nota mínima que debe obtener es de 2 puntos.

**¡ATENCIÓN! PONGA SUS DATOS EN LA HOJA DE LECTURA ÓPTICA QUE DEBERÁ ENTREGAR JUNTO CON EL RESTO DE LAS RESPUESTAS.**

Conteste a las preguntas teóricas, en cualquier orden, en hojas diferentes a las que utilice para la contestación de la segunda parte. En cada parte, la cantidad MÁXIMA de papel (de examen, timbrado) que puede emplear ESTÁ LIMITADA al equivalente a DOS (2) HOJAS de tamaño A4 (210 x 297 mm)

PRIMERA PARTE. PREGUNTAS TEÓRICAS (2'5 PUNTOS CADA UNA)

1. Explique qué se entiende por “pseudocódigo” y para qué se emplea principalmente.

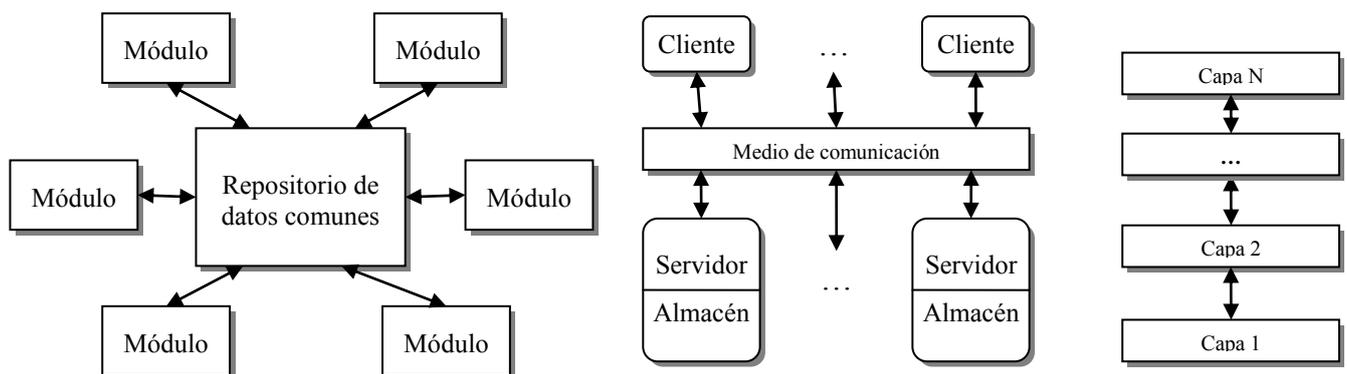
**Solución**

El pseudocódigo es una notación empleada para la especificación de los **requisitos funcionales** de un sistema. Se basa en las estructuras básicas de los lenguajes de programación estructurada (secuencia, selección e iteración), pero sin los aspectos de declaración de tipos, variables, constantes o subprogramas. No se trata de una notación con reglas estrictas sino que cada analista tendrá su propio estilo a la hora de utilizar el pseudocódigo. Esta notación también sirve para realizar el diseño del sistema, en cuyo caso se estará llegando a un mayor nivel de detalle sobre los requisitos funcionales, sin llegar a la codificación, pero sí detallando el algoritmo que se empleará. Existen pseudocódigos para utilizar en la fase de diseño, que se denominan PDLs (lenguajes de descripción de programas).

2. Una de las decisiones importantes que hay que tomar al diseñar la arquitectura de un sistema consiste en definir el «*estilo arquitectónico*». Según Garlan y Shaw, un estilo arquitectónico es un patrón de organización de un sistema que incluye estos tres importantes aspectos del diseño:

- 1) La estructura más adecuada (distribución y organización de los módulos y sus dependencias).
- 2) Cómo se descomponen los subsistemas en sus componentes.
- 3) Cómo se controla la ejecución de los subsistemas.

En relación al primer aspecto, tres estilos organizativos ampliamente utilizados son los siguientes:



a) Modelo de repositorio

b) Modelo Cliente-Servidor

c) Modelo por capas

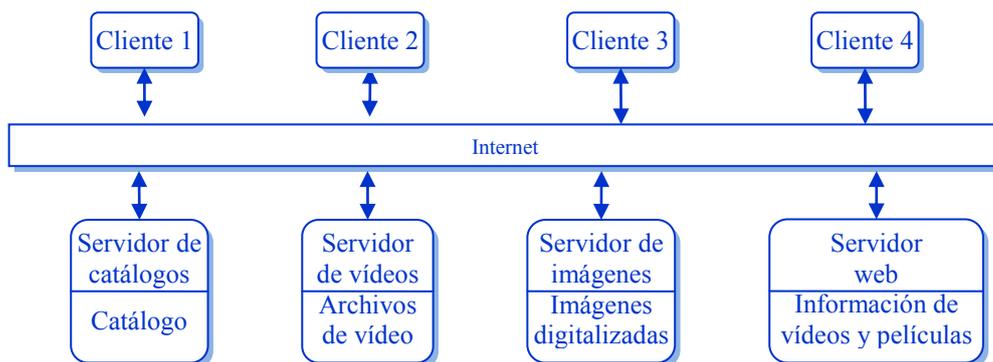
Ahora, estudie este caso con atención:

«Un sistema multiusuario basado en Web, proporciona una biblioteca de películas y fotografías. En él, varios servidores gestionan y visualizan diferentes tipos de dispositivos. Las secuencias de vídeo necesitan ser transmitidas rápidamente y en sincronía, aunque con una resolución relativamente baja. Éstas se pueden comprimir en un almacén para que el servidor de vídeo pueda gestionar la compresión y descompresión de vídeo en diferentes formatos. Sin embargo, las fotografías deben mantenerse con una resolución alta, por lo que es adecuado gestionarlas en un servidor separado. Un gestor de catálogos debe ser capaz de manejar una gran variedad de peticiones y proporcionar enlaces al sistema de información Web, que incluye datos sobre las películas de vídeo y las fotografías, y un sistema de comercio electrónico que soporta la venta de las fotografías y de las películas. Cada programa que usa estos servicios no es más que una interfaz de usuario integrada con estos servicios y construida usando un navegador Web.»

**Seleccione, entre los tres de más arriba, el modelo de estructura (organización) de la arquitectura más adecuado para este sistema. Represente el diagrama seleccionado completando las cajas con los elementos que se enuncian para este sistema. Argumente porqué ha tomado esta decisión y qué ventajas tiene.**

### Solución

La ventaja más importante del modelo cliente-servidor es que es una arquitectura distribuida. Se puede hacer un uso efectivo de los sistemas en red con muchos procesadores distribuidos. Es fácil añadir un nuevo servidor e integrarlo con el resto del sistema o actualizar los servidores de forma transparente sin afectar al resto del sistema. Puede no haber un único modelo de datos compartido entre los servidores y, por tanto, los subsistemas pueden organizar sus datos de formas diferentes. Esto quiere decir que se puede optimizar el rendimiento de cada servidor estableciendo en él un modelo de datos específico.



b) Modelo Cliente-Servidor

Sin embargo, un inconveniente podría ser que, para obtener mayores beneficios de la integración de un nuevo servidor, fuera necesario realizar cambios en los clientes o en los servidores existentes. Si se usara una representación de los datos basada en XML, sería relativamente simple realizar las conversiones de un esquema a otro –en el supuesto de que se hiciera una modificación en algún elemento y el sistema no compartiera un modelo de datos único–. Desgraciadamente, XML es una forma de representar los datos poco eficiente y, su uso, podría acarrear problemas en el rendimiento cuya facilidad de optimización se mencionaba como ventaja.

## SEGUNDA PARTE. PREGUNTA DE TEORÍA APLICADA (MÁXIMO 5 PUNTOS)

3. El sistema Mail Secure sirve para enviar correos electrónicos firmados y/o cifrados. El funcionamiento básico (simplificado) es el siguiente:

Cada usuario tiene dos claves: una privada, solo accesible por él, y otra clave pública que es compartida por el resto de los usuarios.

Veamos con un ejemplo cómo Alicia envía un mensaje firmado y luego cifrado (oculto) a Bernardo, su amigo.

- 1) Alicia pasa su mensaje por una función hash que lo comprime a modo de resumen.
- 2) Este mensaje hash o resumen se codifica con la clave privada de Alicia. El mensaje resultante se conoce como firma digital.
- 3) El mensaje original queda firmado al unirlo con su firma.
- 4) Alicia puede enviar el mensaje firmado (original + firma) a Bernardo.
- 5) Pero si Alicia quisiera además cifrar (codificar) su mensaje antes de enviarlo, utilizaría la clave pública de Bernardo para cifrar el mensaje firmado obteniendo así un mensaje firmado y cifrado (encriptado).
- 6) Ahora Alicia envía a Bernardo su mensaje firmado y cifrado (encriptado) a salvo de cualquier mirada y además llevando la marca de autenticidad (su firma).

Veamos ahora como procede Bernardo para descifrar el mensaje y comprobar su autoría.

- 1) Cuando Bernardo recibe el mensaje cifrado y firmado de Alicia pasa a descifrarlo utilizando su clave privada, obteniendo el mensaje original de Alicia y su firma.
- 2) Para comprobar que es Alicia quien lo ha firmado, descifra la firma con la clave pública de Alicia. Por otra parte, aplica la función hash al mensaje descifrado de Alicia. Obteniendo así dos mensajes resúmenes.
- 3) Si coinciden, se puede garantizar que la firma es de Alicia, si no, se rechaza el mensaje.

**Modele el sistema anterior mediante dos DFD's: uno para la parte emisora (Alicia) y otro para la parte receptora (Bernardo) para el caso que se quiera enviar un mensaje cifrado y firmado.**

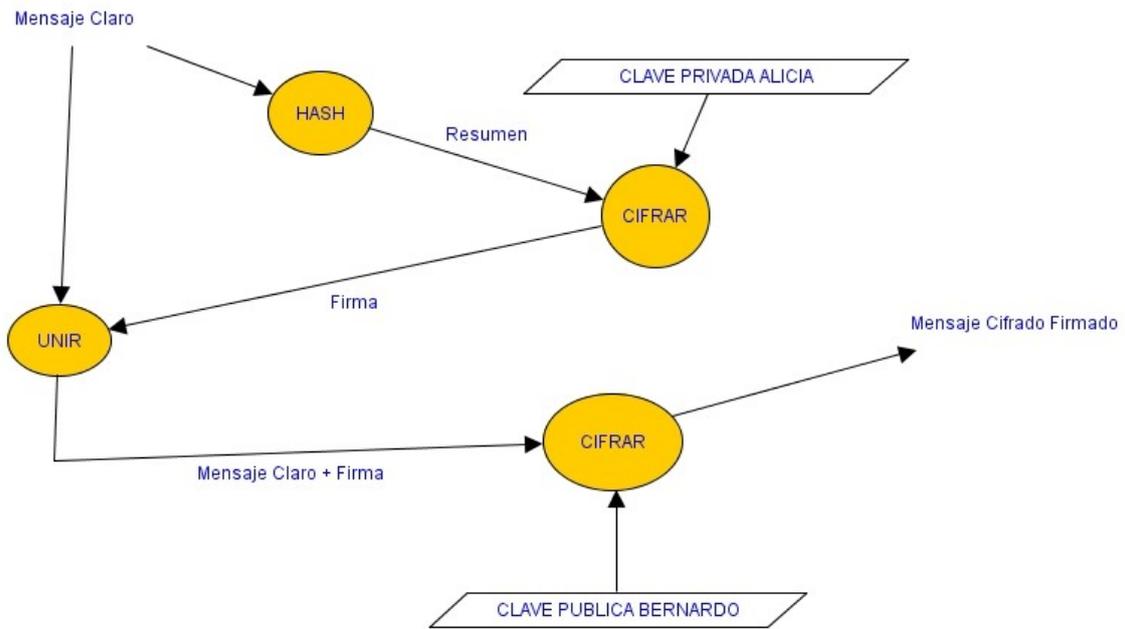
Nota. Codificar, cifrar o encriptar es aplicar una transformación a un texto utilizando un algoritmo que necesita un parámetro llamado clave. La finalidad es ocultar el texto original obteniendo otro texto ilegible. Descifrar o desencriptar es la operación inversa.

### Solución

Emisor (Alicia):



DFD de contexto. Nivel 0

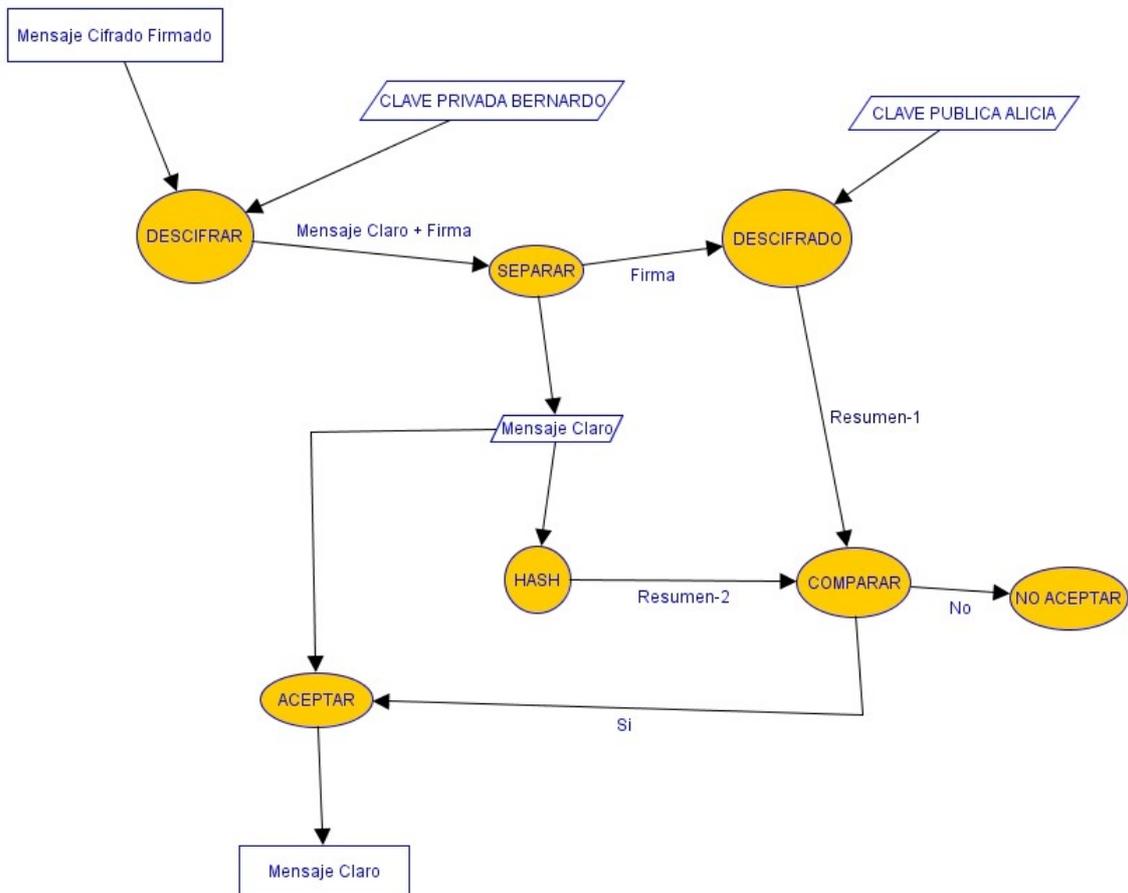


DFD de nivel 1.

Receptor (Bernardo):



DFD de contexto. Nivel 0



DFD de nivel 1.

INGENIERÍA DEL SOFTWARE (2º Curso) Cód. 53210 SISTEMAS 54208 GESTIÓN			
	Modelo		
	Junio 2008		Ámbito
			UE 2ª SEMANA

**ESTE EJERCICIO NO ES DE TEST. NO TIENE RETORNO TELEMÁTICO.**

**NECESITO EL EJERCICIO MANUSCRITO POR EL ALUMNO Y LA HOJA DE LECTURA ÓPTICA PARA PODER CALIFICAR.**

**SI ESTE EJERCICIO NO SE HA IMPRESO EN HOJA DE LECTURA ÓPTICA, SOLICITE UNA AL TRIBUNAL.**

Todas las preguntas de este ejercicio son eliminatorias en el sentido de que debe obtener una nota mínima en cada una de ellas. En cada pregunta teórica, que se valora con 2'5 puntos, la nota mínima es 1 punto; en la segunda parte (ejercicio de teoría aplicada que se valora con 5 puntos) la nota mínima que debe obtener es de 2 puntos.

**¡ATENCIÓN! PONGA SUS DATOS EN LA HOJA DE LECTURA ÓPTICA QUE DEBERÁ ENTREGAR JUNTO CON EL RESTO DE LAS RESPUESTAS.**

Conteste a las preguntas teóricas, en cualquier orden, en hojas diferentes a las que utilice para la contestación de la segunda parte. En cada parte, la cantidad MÁXIMA de papel (de examen, timbrado) que puede emplear ESTÁ LIMITADA al equivalente a DOS (2) HOJAS de tamaño A4 (210 x 297 mm)

PRIMERA PARTE. PREGUNTAS TEÓRICAS (2'5 PUNTOS CADA UNA)

1. ¿En qué consiste el “análisis del dominio” del sistema software que se desarrolla y qué ventajas tiene su realización?

**Solución**

Se trata del estudio del entorno en el que se enmarcará el sistema que se desarrolla, en el cual existirá una manera de hacer las cosas y una terminología que deberá ser tenida en cuenta. Esta es una actividad que se debe contemplar en la fase de análisis del proceso de desarrollo. Al estudiar el contexto del sistema, el analista podrá alcanzar una **comprensión más clara y precisa** de lo que se necesita desarrollar; fundamental para que las actividades tengan éxito. Además, permite tener una **comunicación fluida** con aquellos interlocutores con los que deberá interactuar para llevar a cabo la especificación del sistema. Si se ignora este contexto, la solución estará excesivamente particularizada al caso concreto que pide un cliente, por lo que la solución desarrollada no servirá para otras empresas. Por tanto, el conocimiento de la problemática del entorno servirá al analista para situar la aplicación en un contexto más global, lo que facilitará la **reutilización** posterior del software desarrollado.

2. Explique cómo se lleva a término, y con qué tipo de pruebas, la fase de validación del sistema.

**Solución**

Validar el sistema significa comprobar que el sistema desarrollado cumple con los requisitos previstos desde el principio, especificados por el cliente y recogidos en el documento de análisis. Para ello se aplican estrategias de caja negra a todo el sistema una vez integrado.

Por último se emplean las pruebas alfa y beta para ver la utilidad que tiene la aplicación a sus usuarios finales.

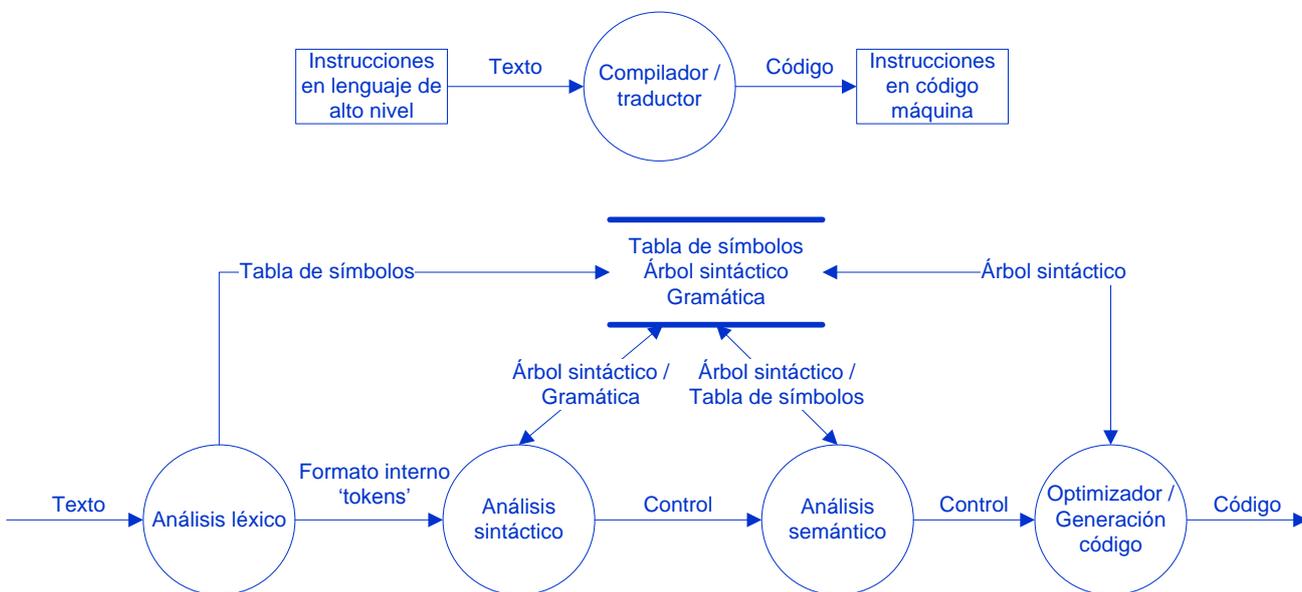
## SEGUNDA PARTE. PREGUNTA DE TEORÍA APLICADA (MÁXIMO 5 PUNTOS)

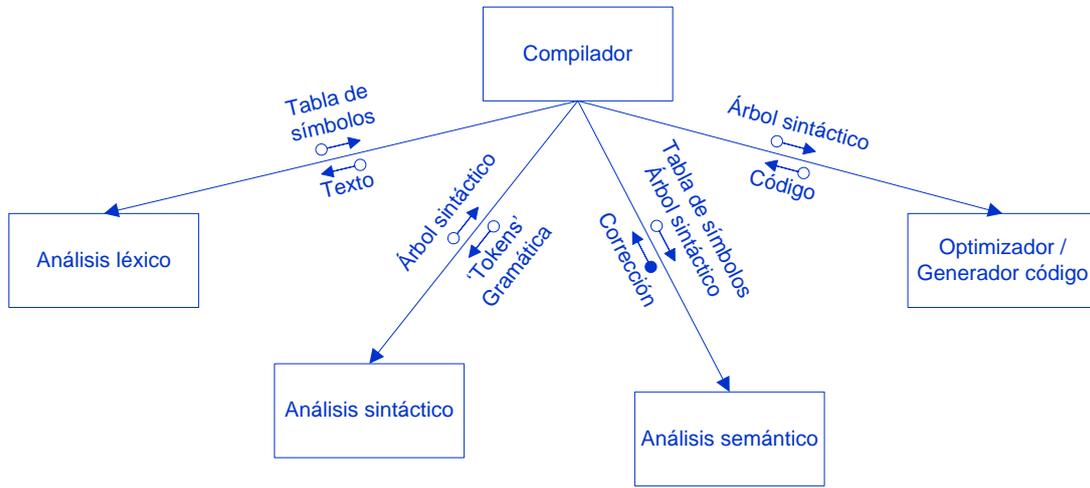
3. Los sistemas de procesamiento de lenguaje aceptan como entrada sentencias en algún lenguaje y generan como salida alguna otra representación del lenguaje de entrada. Un caso particular son los compiladores, que traducen un lenguaje de programación artificial de alto nivel a código máquina (abstracta o de un procesador real). Los principales componentes de un traductor (sea para un compilador u otro procesador de lenguaje) son:

- 1) Un analizador léxico, que toma como entrada los elementos del lenguaje, identifica sus símbolos y los convierte a un formato interno.
- 2) Una tabla de símbolos, que almacena información sobre los nombres de las entidades (variables, nombres de clases, de objetos, etc.) usadas en el texto que se está traduciendo.
- 3) Un analizador sintáctico, que comprueba la sintaxis del lenguaje que se está traduciendo. Utiliza una gramática definida y construye un árbol sintáctico.
- 4) Un árbol sintáctico, que es una estructura interna que representa el texto que se está traduciendo o el programa que se está compilando.
- 5) Un analizador semántico, que utiliza la información del árbol sintáctico y de la tabla de símbolos para comprobar la corrección semántica del texto en el lenguaje de entrada.
- 6) Un optimizador, que transforma el árbol sintáctico para mejorar la eficiencia y eliminar redundancias en el código máquina que se genere.
- 7) Un generador de código, que «recorre» el árbol sintáctico y genera código máquina (traducido).

a) Represente la arquitectura genérica de un compilador mediante un modelo de flujo de datos (haga el DFD, el correspondiente análisis de flujo de datos y represente la arquitectura con un diagrama de estructura). (2 puntos)

### Solución



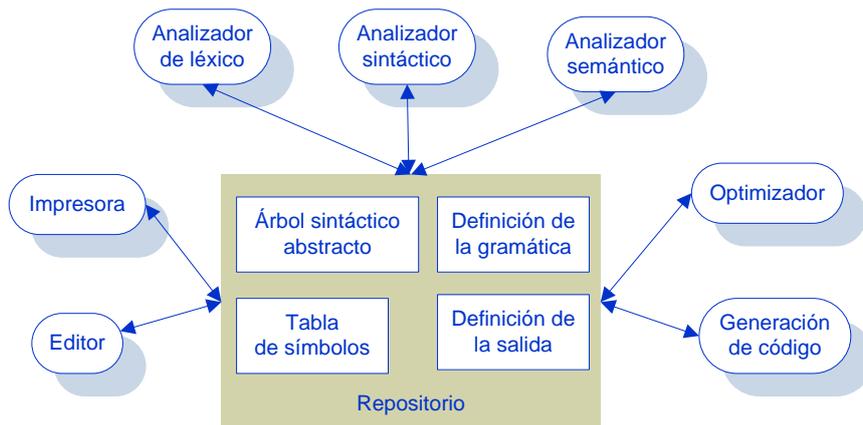


Una decisión fundamental para el diseño de la arquitectura es la estrategia para estructurar el sistema. Se denomina «modelo de repositorio» a la estructura en la que los subsistemas se agrupan en torno a una base de datos compartida, almacén o repositorio. Suponga ahora que el compilador anterior forma parte de un conjunto integrado de herramientas de soporte para la programación. Tales herramientas son el compilador, un sistema de edición estructurado, un depurador interactivo y un programa de impresión. La gramática del lenguaje y el formato de salida del programa son elementos generalmente embebidos en las herramientas pero, por ser comunes, se incluyen en el repositorio junto con los datos comunes del compilador. El repositorio no sólo sirve de almacén, sino también de vía de comunicación entre las herramientas.

b) **Transforme la arquitectura definida anteriormente para el compilador a un modelo de repositorio al integrarlo con las herramientas mencionadas en un entorno de programación con un editor dirigido por la sintaxis (según se escribe, verifica si la sintaxis es correcta e informa al usuario).** (2 puntos)

Sugerencia: para que se vea más claro y sencillo, no use un diagrama de estructura sino un diagrama de bloques o uno parecido a un DFD.

**Solución**



c) *¿Qué ventajas tiene la última estructura arquitectónica respecto a, por ejemplo, el editor dirigido por la sintaxis, el depurador interactivo o el sistema de impresión? (1 punto)*

### **Solución**

Como la definición de la gramática está compartida en el repositorio –en lugar de estar embebida en el editor, que supervisa la sintaxis–, se puede comprobar la corrección sintáctica del programa fuente al mismo tiempo que se está escribiendo.

Un depurador interactivo utiliza el programa compilado –lenguaje máquina– y suele permitir hacer un seguimiento en la ejecución, ejecución paso a paso y visualización de los valores de las variables, posiciones de memoria, registros, etc. Esta funcionalidad del depurador es más efectiva cuando el compilador está integrado junto con las otras herramientas –arquitectura en repositorio común–. No lo es tanto si el compilador está aislado de las otras herramientas; en cuyo caso, está más orientado al procesamiento por lotes, en el que los programas son compilados y ejecutados sin la interacción del usuario.

De igual forma, al estar embebido también el formato de salida del programa, el sistema de impresión puede generar, en cualquier momento, listados adecuados que son fáciles de leer.

INGENIERÍA DEL SOFTWARE (2º Curso) Cód. 53210 SISTEMAS 54208 GESTIÓN			
	Modelo		
	Junio 2008		
			Ámbito
			CP -- ORIGINAL

**ESTE EJERCICIO NO ES DE TEST. NO TIENE RETORNO TELEMÁTICO.**

**NECESITO EL EJERCICIO MANUSCRITO POR EL ALUMNO Y LA HOJA DE LECTURA ÓPTICA PARA PODER CALIFICAR.**

**SI ESTE EJERCICIO NO SE HA IMPRESO EN HOJA DE LECTURA ÓPTICA, SOLICITE UNA AL TRIBUNAL.**

Todas las preguntas de este ejercicio son eliminatorias en el sentido de que debe obtener una nota mínima en cada una de ellas. En cada pregunta teórica, que se valora con 2'5 puntos, la nota mínima es 1 punto; en la segunda parte (ejercicio de teoría aplicada que se valora con 5 puntos) la nota mínima que debe obtener es de 2 puntos.

**¡ATENCIÓN! PONGA SUS DATOS EN LA HOJA DE LECTURA ÓPTICA QUE DEBERÁ ENTREGAR JUNTO CON EL RESTO DE LAS RESPUESTAS.**

Conteste a las preguntas teóricas, en cualquier orden, en hojas diferentes a las que utilice para la contestación de la segunda parte. En cada parte, la cantidad MÁXIMA de papel (de examen, timbrado) que puede emplear ESTÁ LIMITADA al equivalente a DOS (2) HOJAS de tamaño A4 (210 x 297 mm)

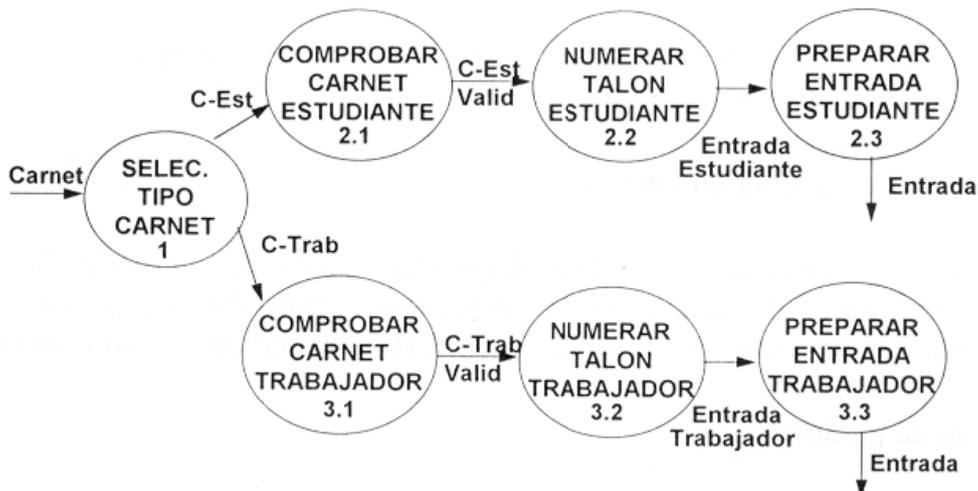
PRIMERA PARTE. PREGUNTAS TEÓRICAS (2'5 PUNTOS CADA UNA)

1. Explique el concepto de abstracción aplicado al diseño de software y sus formas fundamentales de uso.

**Solución**

Por medio de la abstracción prescindimos de los elementos y utilidades concretas de un sistema software determinado para considerarlos más allá de ese sistema. De esta manera, los elementos que diseñemos serán fácilmente **mantenibles** (se podrán sustituir por otros con mejores prestaciones) y **reutilizables** en otros proyectos similares. Hay tres formas fundamentales de abstracción: abstracciones funcionales, tipos abstractos de datos y máquinas abstractas.

2. La Comunidad Autónoma ha decidido hacerse cargo de la gestión de la piscina «El Remojón». Para ello sólo dejará acceder, de forma gratuita, a dichas instalaciones a aquellos usuarios que sean estudiantes o trabajadores locales. Dependiendo del tipo de usuario (estudiante o trabajador), que se acreditará con un carnet, se realizarán diferentes tratamientos para imprimir la entrada. Suponga que este es el DFD de nivel 2 simplificado para un diseño estructurado:



Se pide:

- a) Aislar el centro de transformación o el de transacciones. Defina estos conceptos y muestre un diagrama de estructura con cada uno de estos casos. (1 punto)
- b) Represente el diagrama de estructura correspondiente al diseño estructurado de este ejemplo. (1'5 puntos)

### Solución

- a) Un *centro de transformación* es una operación, acción, proceso o agrupación de ellos, que define la transformación fundamental de un sistema en el que se ha identificado un flujo de información global desde los elementos de entrada hacia los de salida.

Un *centro de transacción* es un proceso que define una bifurcación, a partir de la cual, el flujo de datos global se descompone en líneas separadas. En el enunciado, la selección del carnet es el centro de transacción que activa una u otra línea (transacción) en función del tipo de dato de entrada –el tipo de carnet–.

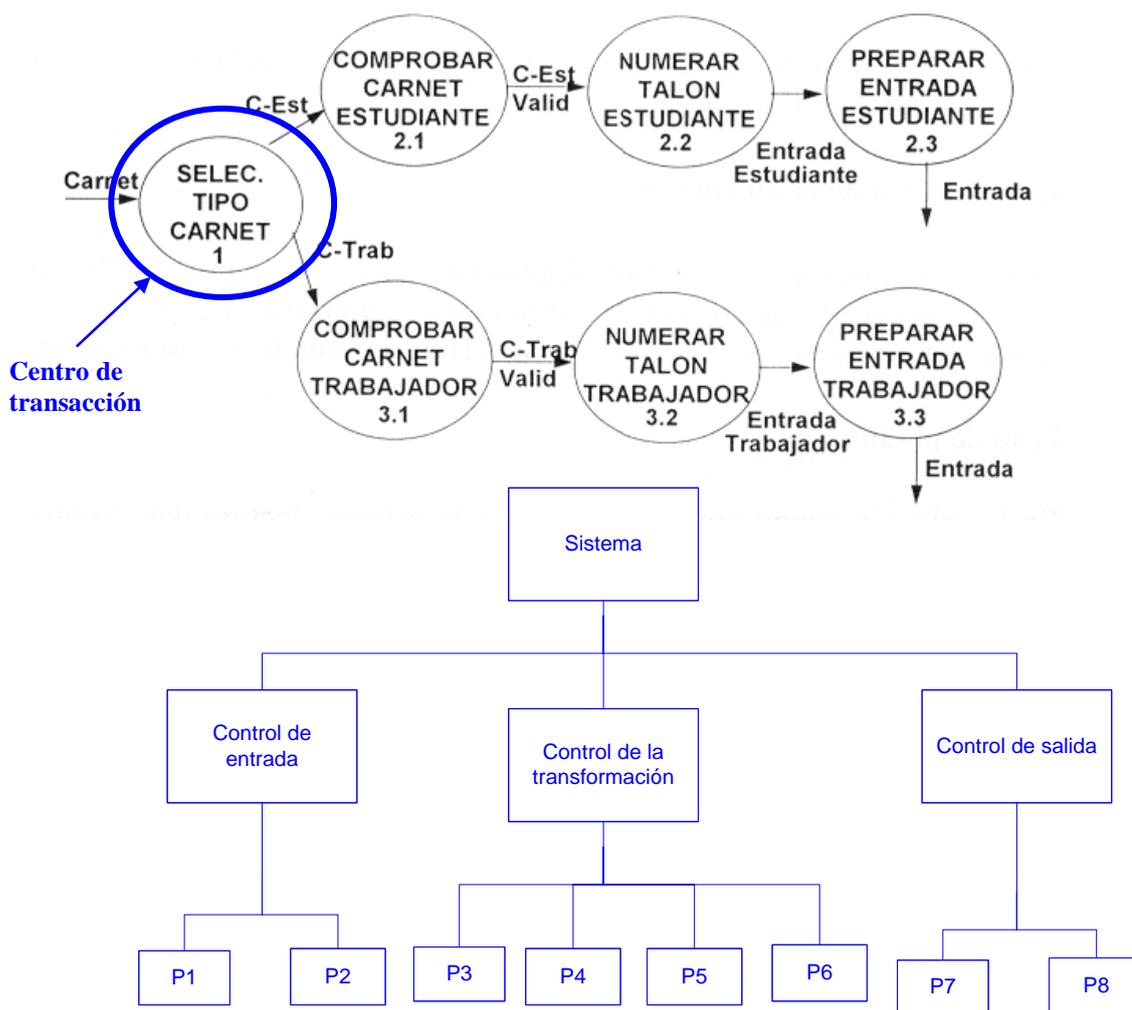


Diagrama de estructura con flujo de transformación.

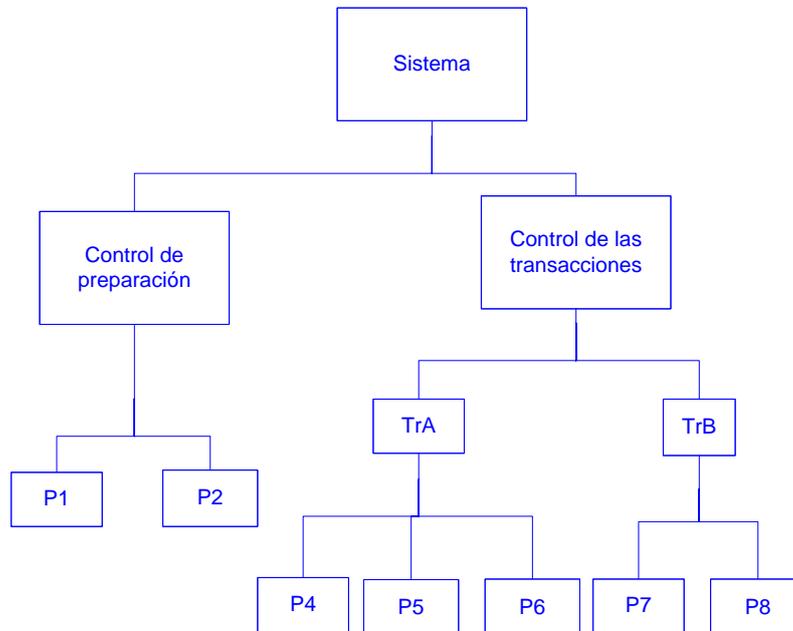


Diagrama de estructura con flujo de transacciones.

b)

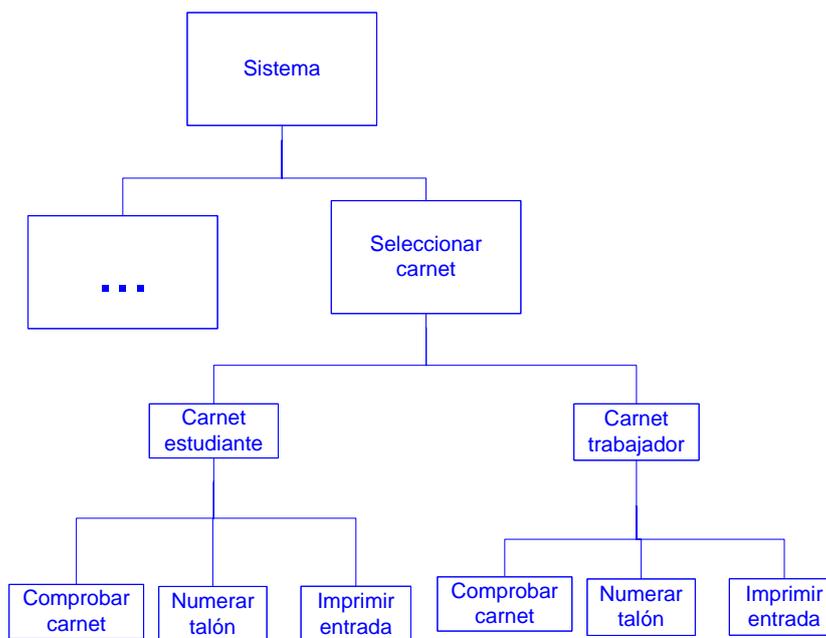


Diagrama de estructura para el control de la entrada a la piscina.

SEGUNDA PARTE. PREGUNTA DE TEORÍA APLICADA (MÁXIMO 5 PUNTOS)

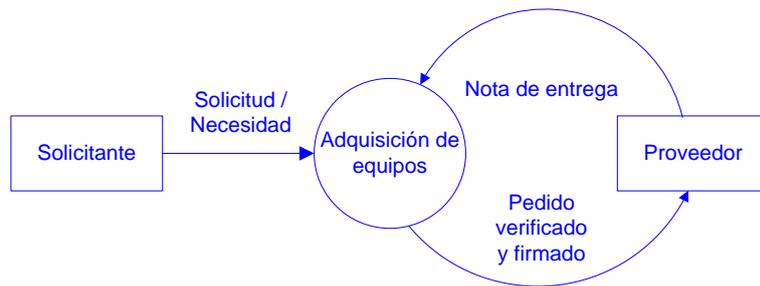
- Se le pide que defina el proceso de adquisición de equipos en una empresa. Esto implica especificar el equipo deseado, buscar y elegir a los proveedores teniendo en cuenta, también, el presupuesto disponible y su aceptación; pedir el equipo, recibirlo a su llegada y, a continuación, verificarlo. La petición del equipo se hará mediante formularios en los que no sólo se reflejarán los detalles del equipo y del proveedor, sino que deberán ir visados para garantizar que hay disponibilidad de fondos y se ha aprobado el gasto. Al emitir el pedido, este se registra para un posterior seguimiento. La recepción de un equipo requiere la verificación de la entrega, instalación, comprobación del funcionamiento especificado, aceptación y registro en la base de datos de inventario.

- a) Resuelva las faltas y ambigüedades que estime. Anótelas e indique las decisiones que ha tomado. (1 punto)
- b) Represente la funcionalidad del proceso mediante diagramas DFD (nivel 0, 1 y 2). (3 puntos)
- c) En el nivel 1, enmarque con una línea discontinua qué parte del proceso decide usted automatizar informáticamente. Explique y argumente su decisión. (1 punto)

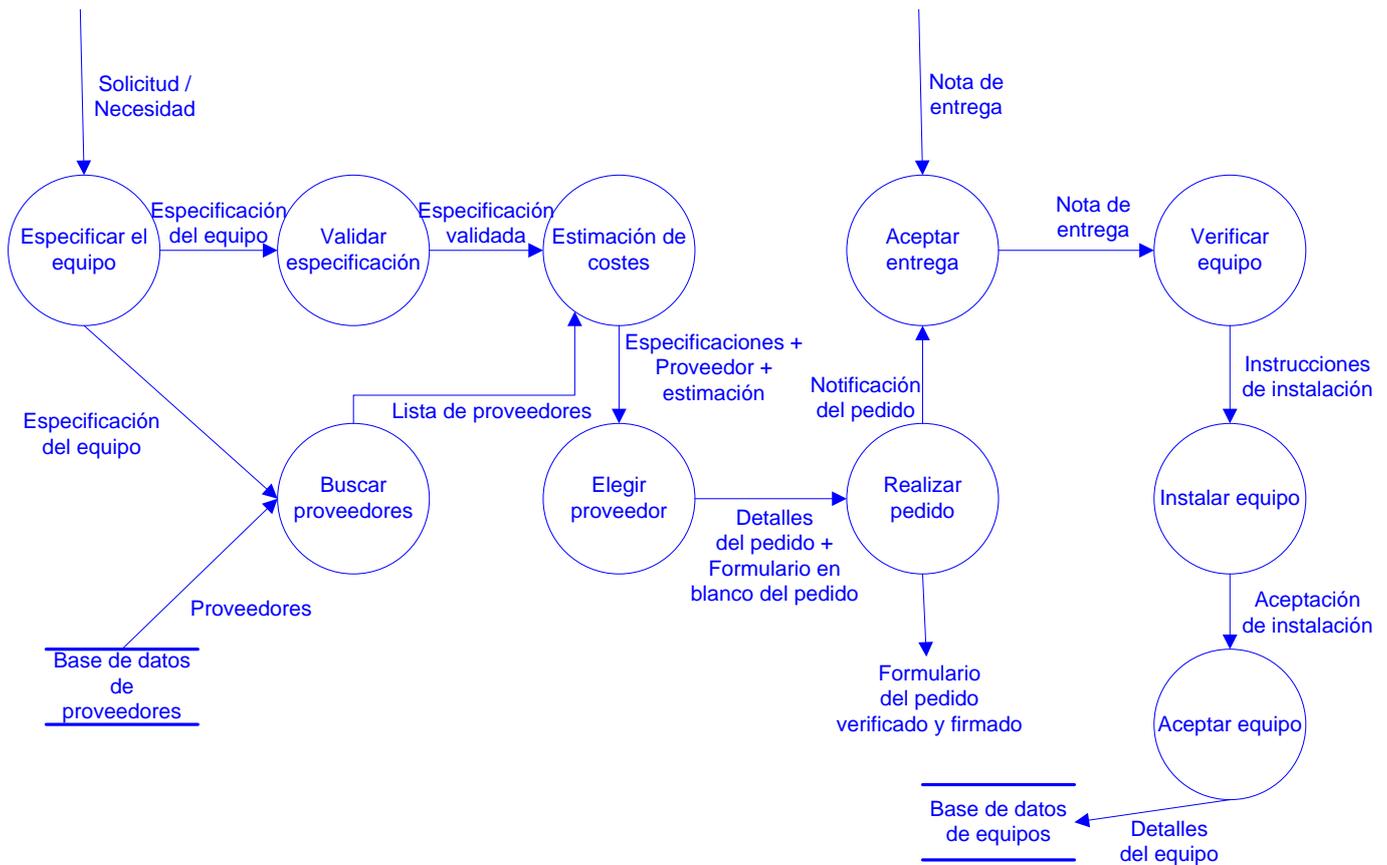
### Solución

a) No queda claro cómo se tiene en cuenta el presupuesto disponible y cuál es el procedimiento de aceptación del pedido y visado de éste. Tampoco queda claro si la evaluación del pedido, su aprobación, registro de éste y el del inventario del equipo –una vez recibido y verificado– se realiza en el propio sistema o bien es una entidad externa como, por ejemplo, ‘Gerencia. Gestión económica e inventario’ la que lo hace.

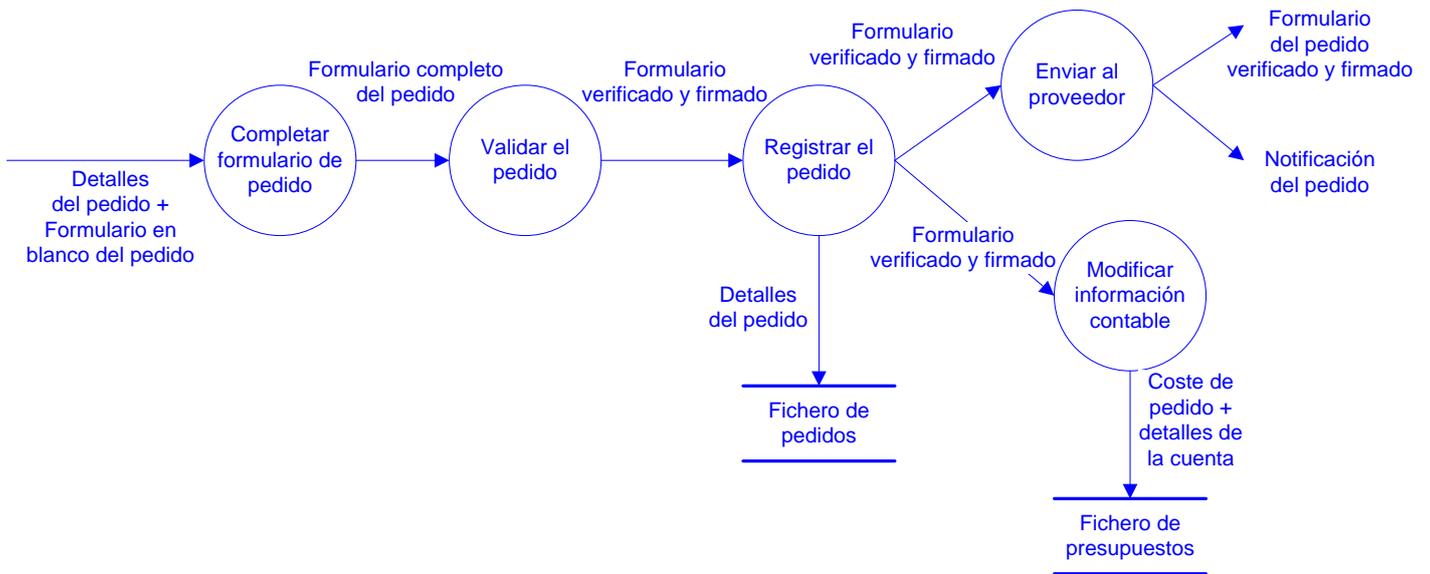
b)



DFD de contexto

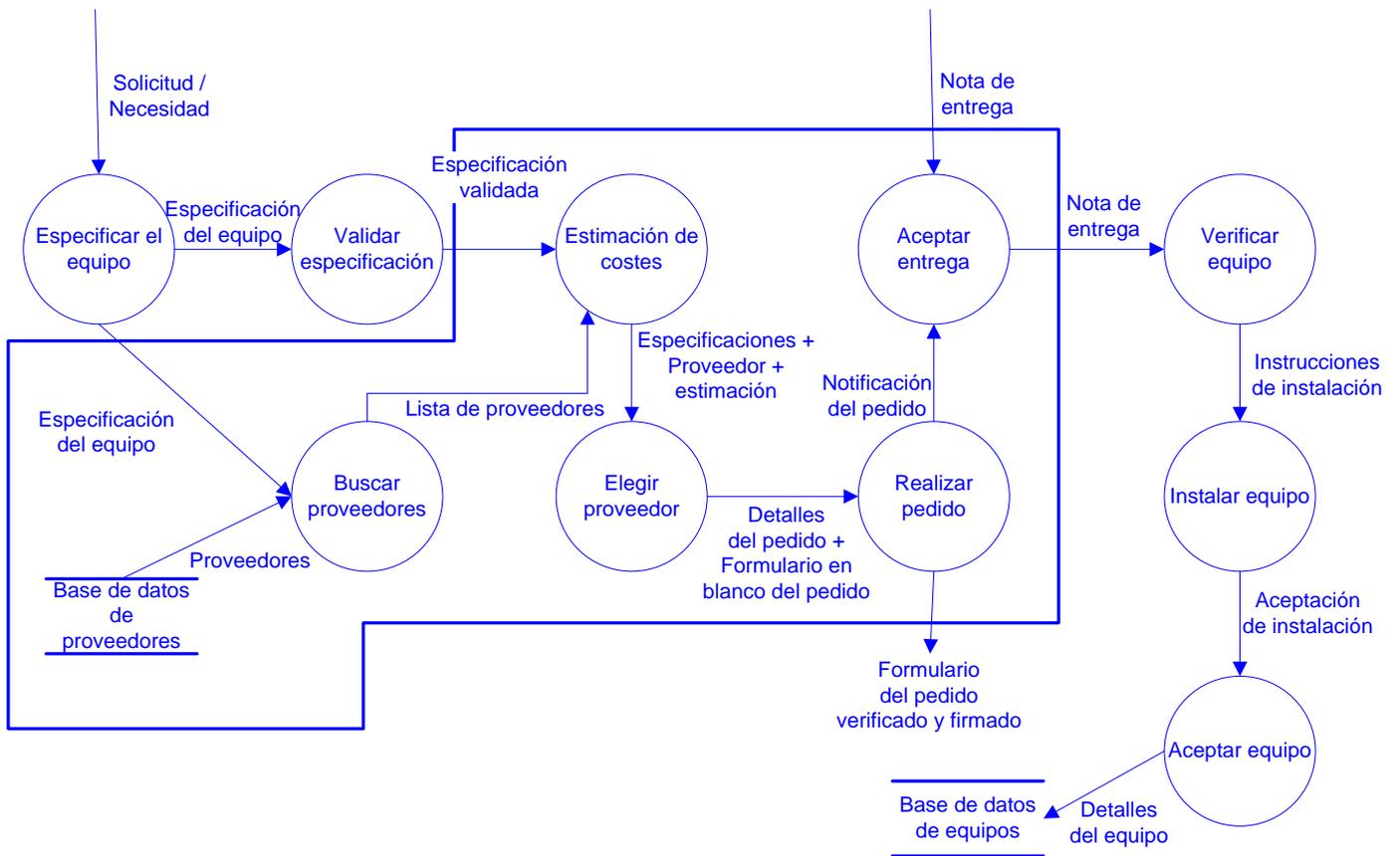


DFD de nivel 1



DFD nivel 2. Explosión de 'Realizar pedido'

c) Se sugiere la parte que se automatiza con un recuadro de línea continua más gruesa.



INGENIERÍA DEL SOFTWARE (2º Curso) Cód. 53210 SISTEMAS 54208 GESTIÓN			
	Modelo		
	Septiembre 2008		
		Ámbito NACIONAL – UE ORIGINAL	

**ESTE EJERCICIO NO ES DE TEST. NO TIENE RETORNO TELEMÁTICO.**

**NECESITO EL EJERCICIO MANUSCRITO POR EL ALUMNO Y LA HOJA DE LECTURA ÓPTICA PARA PODER CALIFICAR.**

**SI ESTE EJERCICIO NO SE HA IMPRESO EN HOJA DE LECTURA ÓPTICA, SOLICITE UNA AL TRIBUNAL.**

Todas las preguntas de este ejercicio son eliminatorias en el sentido de que debe obtener una nota mínima en cada una de ellas. En cada pregunta teórica, que se valora con 2'5 puntos, la nota mínima es 1 punto; en la segunda parte (ejercicio de teoría aplicada que se valora con 5 puntos) la nota mínima que debe obtener es de 2 puntos.

**¡ATENCIÓN! PONGA SUS DATOS EN LA HOJA DE LECTURA ÓPTICA QUE DEBERÁ ENTREGAR JUNTO CON EL RESTO DE LAS RESPUESTAS.**

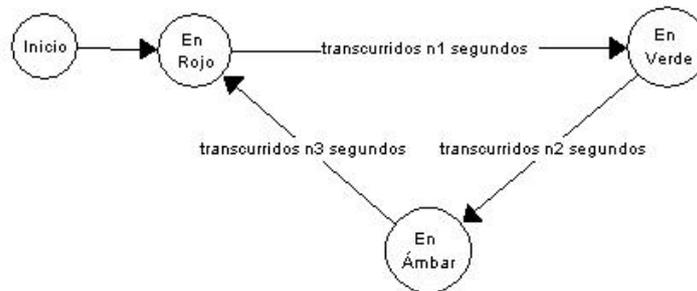
Conteste a las preguntas teóricas, en cualquier orden, en hojas diferentes a las que utilice para la contestación de la segunda parte. En cada parte, la cantidad MÁXIMA de papel (de examen, timbrado) que puede emplear ESTÁ LIMITADA al equivalente a DOS (2) HOJAS de tamaño A4 (210 x 297 mm)

PRIMERA PARTE. PREGUNTAS TEÓRICAS (2'5 PUNTOS CADA UNA)

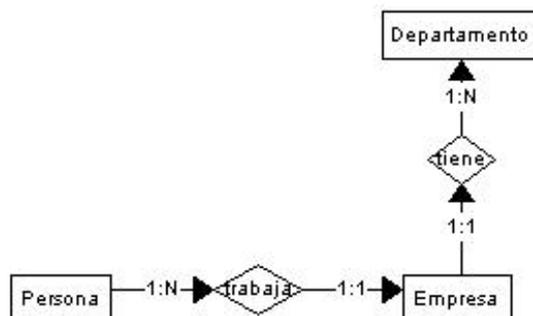
1. Los Diagramas de Transición de Estados (DTE) y los Diagramas Entidad Relación (DER) son notaciones para describir distintos aspectos de un sistema software. ¿Cuáles son estos aspectos? Razone la contestación proponiendo un ejemplo sencillo de aplicación de cada notación.

Solución

Los DTEs sirven para modelar el comportamiento temporal de un sistema software. Por ejemplo, el siguiente diagrama representa la secuencia de colores de un semáforo.



Los DERs permiten modelar la estructura de los datos que maneja una aplicación informática. Por ejemplo, el siguiente diagrama muestra el tipo de elementos que manipula un sistema de nóminas.



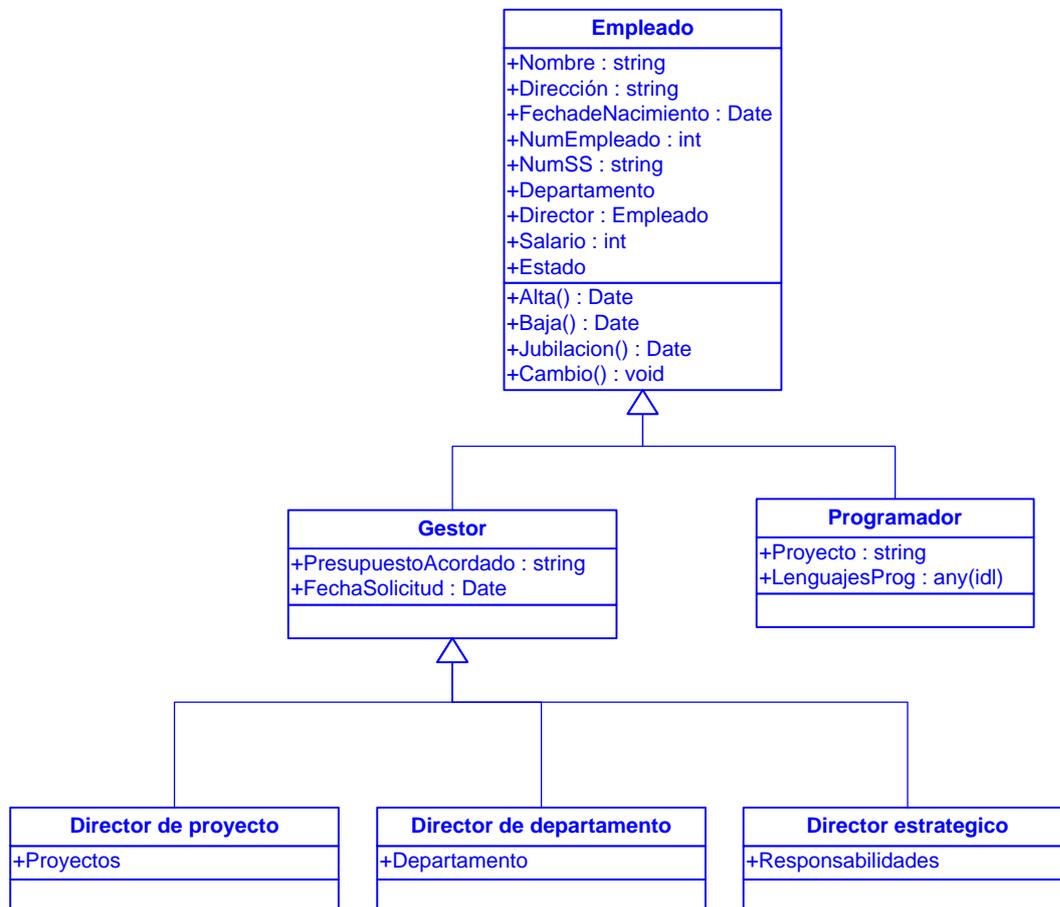
**OBSERVACIÓN:** un ejercicio interesante para entender los puntos fuertes y las limitaciones de cada notación sería tratar de modelar cada ejemplo intercambiando la notación propuesta en esta solución.

2. A partir de la siguiente descripción:

«Los empleados de una organización de desarrollo de software pueden ser gestores o programadores. Los programadores se caracterizan por la lista de lenguajes de programación para los que están capacitados para hacer desarrollos y por el proyecto en el que están trabajando. Los gestores, por el contrario, se diferencian por el presupuesto de los trabajos que gestionan y por la fecha de solicitud de dichos trabajos. Los gestores pueden ser, a su vez, directores de proyecto –caracterizados por el proyecto que gestionan–, directores de departamento o directores estratégicos –caracterizados por determinadas responsabilidades en las áreas de negocio de la organización o de la unidad a la que pertenecen–».

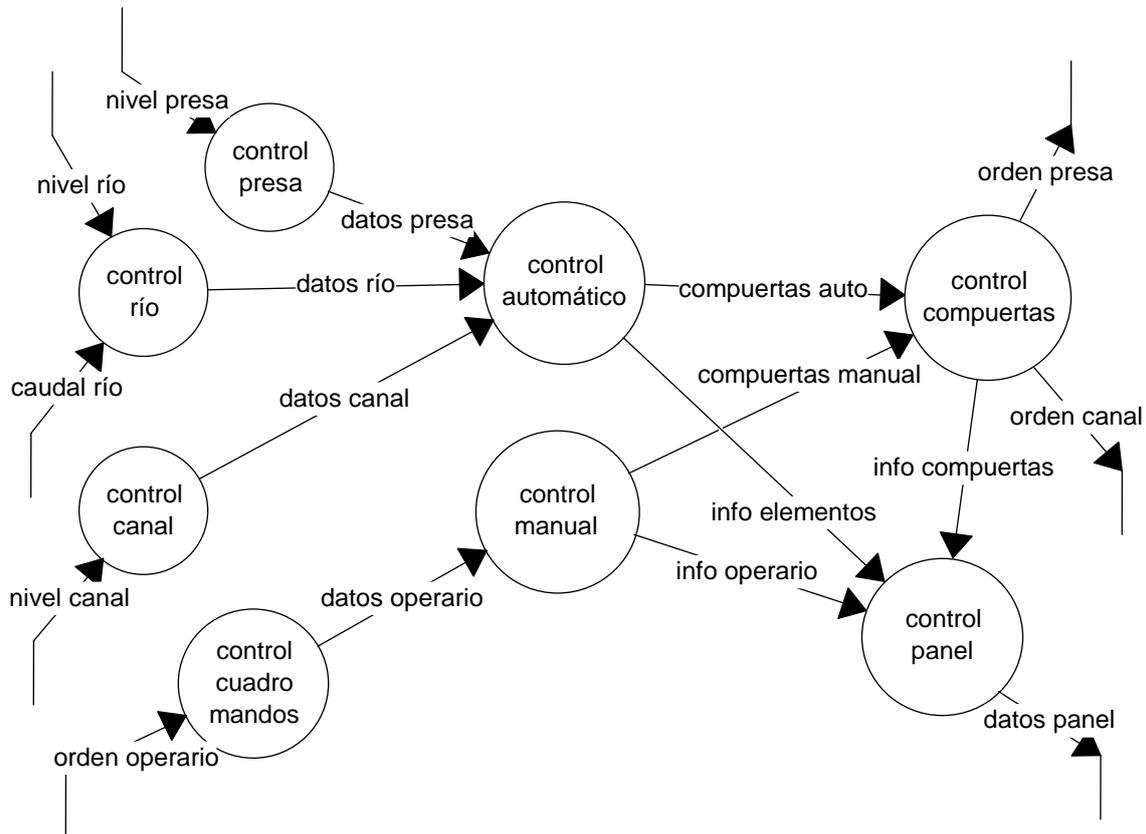
**Construya un modelo de objetos solamente de los empleados.**

### Solución



SEGUNDA PARTE. PREGUNTA DE TEORÍA APLICADA (MÁXIMO 5 PUNTOS)

3. La figura que se muestra a continuación corresponde al diagrama de flujo de datos de un **sistema de control de desbordamiento de una presa**. Realiza un control automático coordinado de 2 compuertas: la de la presa y la de un canal de riego abastecido por el río al que vierte agua el embalse. Además, muestra en un panel de control un diagrama que representa todos los elementos del sistema y su estado instantáneo. El control de las compuertas se basa en datos proporcionados por sensores de nivel y de caudal situados en el embalse, el río y el canal de riego. También se puede ejercer un control manual de las compuertas mediante un cuadro de mandos de operario.



Los objetivos que debe cumplir el sistema son los siguientes, **de mayor a menor prioridad**:

- atender las peticiones del operario
- evitar desbordamientos de la presa
- evitar desbordamientos del río
- evitar desbordamientos del canal de riego
- garantizar agua en el canal de riego

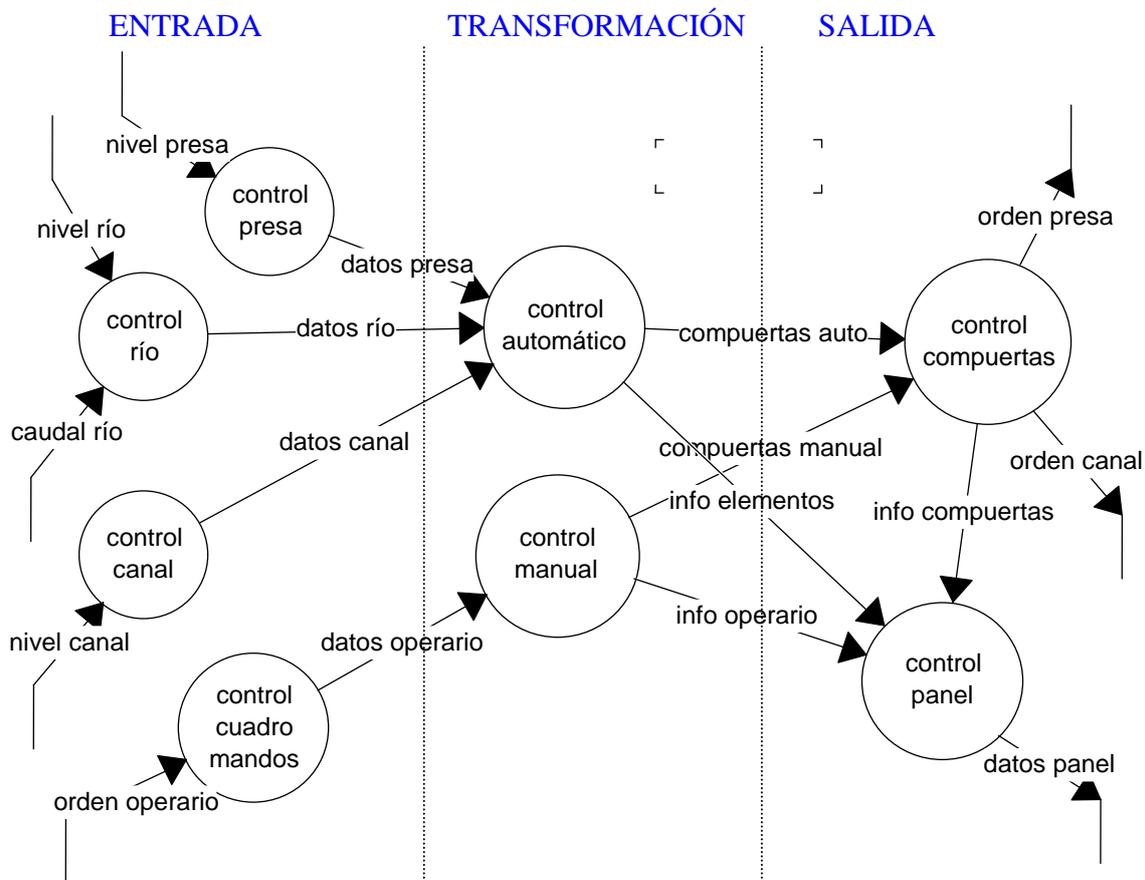
Se pide:

- Utilizando la técnica de diseño estructurado, realice el diagrama de estructura del sistema a partir del anterior DFD.
- Proponga un algoritmo, utilizando pseudocódigo, para el diseño procedimental del módulo “control de transformación” resultante del paso 1. Haga lo mismo con los módulos que cuelgan de él. Tenga para ello en cuenta los objetivos del sistema y su prioridad.

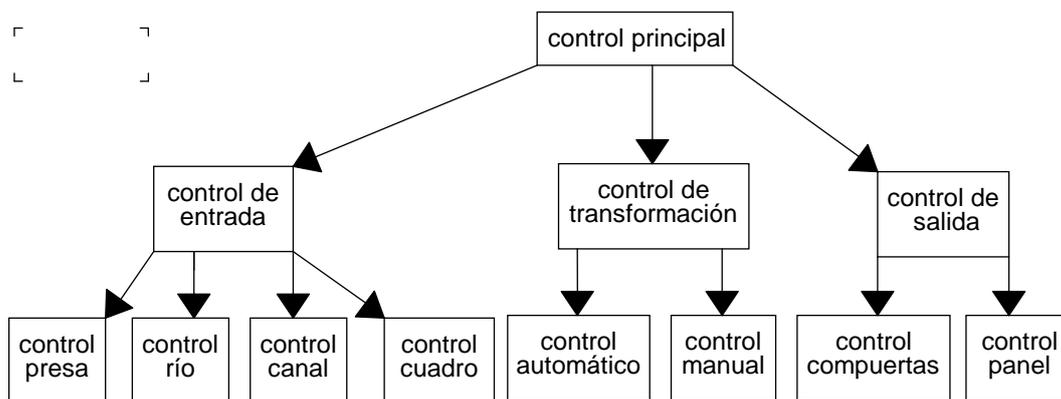
### Solución

- El diagrama de estructura añade al DFD una jerarquía de control. La técnica de diseño estructurado recomienda realizar análisis de flujo de transformación o de flujo de transacción para establecer

dicha jerarquía. En nuestro caso el análisis de flujo apropiado es el de transformación, siendo fácilmente identificables los flujos de información de entrada, transformación y salida del sistema:



El diagrama de estructura será:



b) “Control de transformación”:

```

REPETIR
    MIENTRAS no lleguen "datos operario"
        Realizar "control automático"
    FIN-MIENTRAS
    Realizar "control manual"
HASTA apagado del sistema
    
```

“control automático”:

```
SI "datos presa" = alerta desbordamiento presa ENTONCES
  abrir compuerta presa y abrir compuerta canal
  SI NO, SI "datos rio"= alerta desbordamiento río ENTONCES
    cerrar compuerta presa y abrir compuerta canal
    SI NO, SI "datos canal"=alerta desbordamiento canal ENTONCES
      cerrar compuerta canal
      SI NO abrir compuerta canal
    FIN-SI
  FIN-SI
FIN-SI
```

“control manual”:

```
CASO "dato operario"
  SI-ES abrir compuerta presa: ENTONCES abrir compuerta presa
  SI-ES abrir compuerta canal: ENTONCES abrir compuerta canal
  SI-ES cerrar compuerta presa: ENTONCES cerrar compuerta presa
  SI-ES cerrar compuerta canal: ENTONCES cerrar compuerta canal
FIN-CASO
```

INGENIERÍA DEL SOFTWARE (2º Curso) Cód. 53210 SISTEMAS 54208 GESTIÓN			
	Modelo		
	Septiembre 2008		
		Ámbito NACIONAL – UE RESERVA	

**ESTE EJERCICIO NO ES DE TEST. NO TIENE RETORNO TELEMÁTICO.**

**NECESITO EL EJERCICIO MANUSCRITO POR EL ALUMNO Y LA HOJA DE LECTURA ÓPTICA PARA PODER CALIFICAR.**

**SI ESTE EJERCICIO NO SE HA IMPRESO EN HOJA DE LECTURA ÓPTICA, SOLICITE UNA AL TRIBUNAL.**

Todas las preguntas de este ejercicio son eliminatorias en el sentido de que debe obtener una nota mínima en cada una de ellas. En cada pregunta teórica, que se valora con 2'5 puntos, la nota mínima es 1 punto; en la segunda parte (ejercicio de teoría aplicada que se valora con 5 puntos) la nota mínima que debe obtener es de 2 puntos.

**¡ATENCIÓN! PONGA SUS DATOS EN LA HOJA DE LECTURA ÓPTICA QUE DEBERÁ ENTREGAR JUNTO CON EL RESTO DE LAS RESPUESTAS.**

Conteste a las preguntas teóricas, en cualquier orden, en hojas diferentes a las que utilice para la contestación de la segunda parte. En cada parte, la cantidad MÁXIMA de papel (de examen, timbrado) que puede emplear ESTÁ LIMITADA al equivalente a DOS (2) HOJAS de tamaño A4 (210 x 297 mm)

PRIMERA PARTE. PREGUNTAS TEÓRICAS (2'5 PUNTOS CADA UNA)

- Según Ian Sommerville, el objetivo final de un desarrollo basado en prototipo rápido es «trabajar con el cliente en la construcción de un producto final a partir de una especificación simplificada. Se debería partir de una buena comprensión del sistema inicial y sus requisitos e ir añadiendo nuevas prestaciones según sean propuestos por el cliente.» Desde el punto de vista de la gestión y de la ingeniería, se producen estos inconvenientes:
  - Si una ventaja es la pronta entrega de versiones al cliente, aunque con funcionalidad limitada, esto puede producir que no resulte rentable generar la documentación necesaria para un conocimiento adecuado de las versiones intermedias; con la consiguiente falta de visibilidad del progreso real del desarrollo.
  - El añadir nuevos requisitos y funcionalidad en las sucesivas versiones, facilita que la estructura de la aplicación se degrade o se corrompa. El resultado final puede tener una estructura deficiente; es decir, el funcionamiento definido para la descomposición modular inicial puede no ser coherente con el funcionamiento de los componentes finales –sobre todo si no se refactoriza (reingeniería)–, aunque las versiones y el resultado final se hayan validado correctamente.
  - En el desarrollo completo de sistemas grandes –o de ciclo de vida muy largo–, en el que lo habitual es que el trabajo se distribuya entre equipos independientes, el enfoque evolutivo obliga a mantener una vinculación y comunicación muy estrecha entre los desarrolladores y con el cliente; lo cual es difícil realizar. Por consiguiente, el modelo evolutivo no es aconsejable para este tipo de sistemas.

*A la vista de lo anterior, argumente cómo se ve afectado el mantenimiento del producto final.*

**Solución**

En el caso de la falta de documentación de las versiones, la repercusión para el mantenimiento es pequeña; a no ser que dicha documentación pueda ser de ayuda para una refactorización de la versión final y la generación de la documentación definitiva.

En cuanto a la degradación de la estructura, parece evidente que si la adición de funcionalidad se hace «*parcheando*» el diseño inicial y, por motivos de plazos o presupuesto, no se rediseña el resultado final, éste puede validarse correctamente, pero el mantenimiento puede ser difícil por defectos en el diseño –dependencia modular, poco comprensible o adaptabilidad escasa–.

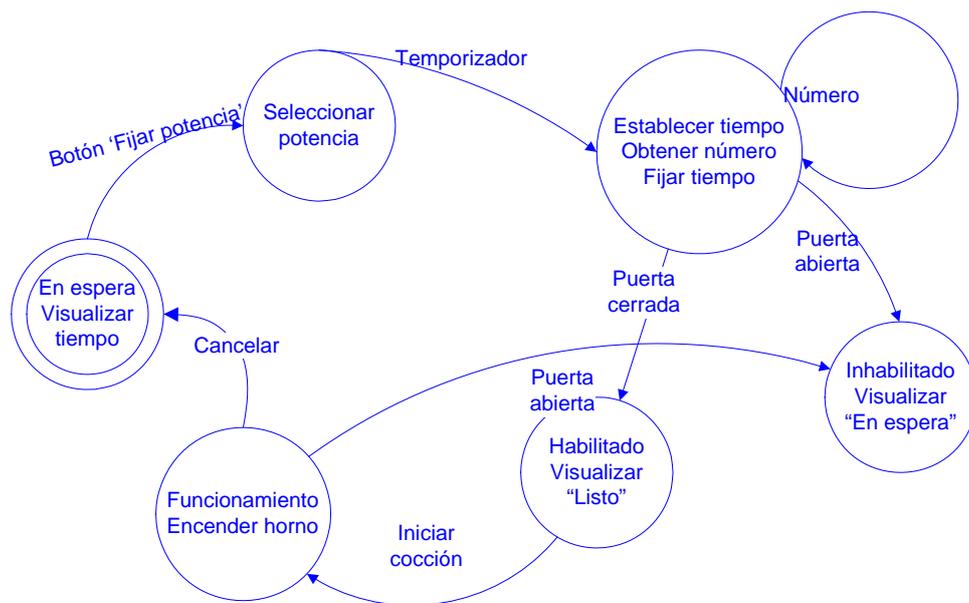
Por último, en el caso de sistemas de gran tamaño –proclives a tener un funcionamiento complejo–, su mantenimiento suele ser más difícil que el de sistemas más simples. Además, las exigencias de este tipo de sistemas respecto a la estabilidad de los requisitos y la coordinación entre equipos de desarrollo –ambas contrarias a la naturaleza del enfoque evolutivo– hacen que, si no se cumplen dichas exigencias, el mantenimiento sea aún más difícil.

2. Suponga que el modelo simplificado de uso de un horno microondas implica estas acciones:
1. Seleccionar el nivel de potencia (media o máxima).
  2. Introducir el tiempo de cocción.
  3. Pulsar el botón de inicio y la comida se cocina durante el tiempo establecido.

El aparato tiene botones para fijar la potencia, el temporizador e iniciar el sistema. Por seguridad, el horno no debería funcionar con la puerta abierta. Cuando se completa la cocción, suena un timbre. El aparato tiene una pantalla alfanumérica para representar información de selección, progreso, alerta y precaución.

**Construya un Diagrama de Transición de Estados para representar este funcionamiento.**

**Solución**



**SEGUNDA PARTE. PREGUNTA DE TEORÍA APLICADA (MÁXIMO 5 PUNTOS)**

3. **Modele con un diagrama de objetos lo que se indica en el siguiente enunciado:**

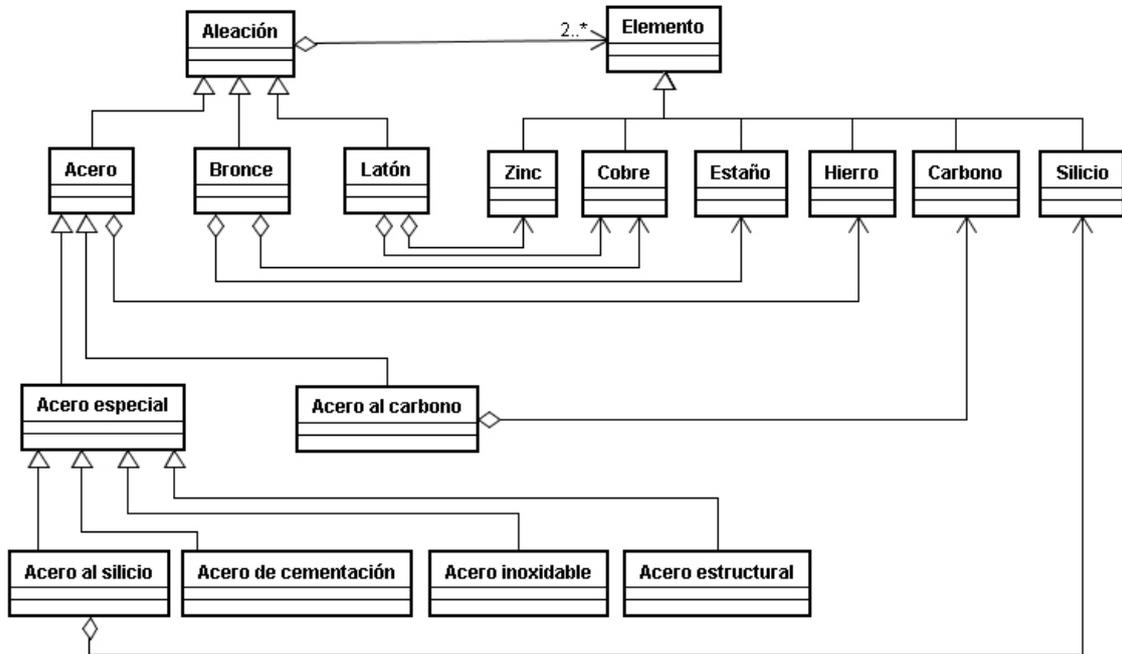
*Por aleación se entiende la unión homogénea de dos o más elementos. Generalmente, se dice que el acero es una aleación de hierro y carbono. Sin embargo, esta definición se restringe a los “aceros al carbono”, ya que existen otros tipos de acero con composiciones muy diversas que reciben denominaciones específicas en virtud:*

- *de los elementos que, además del hierro, predominan en su composición (por ejemplo, aceros al silicio).*
- *de su susceptibilidad a ciertos tratamientos (por ejemplo, aceros de cementación)*

- de alguna característica potenciada (por ejemplo, aceros inoxidables)
- en función de su uso (por ejemplo, aceros estructurales).

Usualmente, estas otras aleaciones de hierro se engloban bajo la denominación genérica de “aceros especiales”. Además del acero, otras aleaciones muy comunes son el bronce, formado por cobre y estaño, y el latón, compuesto por cobre y zinc.

### Solución



INGENIERÍA DEL SOFTWARE (2º Curso) Cód. 53210 SISTEMAS 54208 GESTIÓN			
	Modelo		
	Septiembre 2008		Ámbito
			CENTROS PENITENCIARIOS ORIGINAL

**ESTE EJERCICIO NO ES DE TEST. NO TIENE RETORNO TELEMÁTICO.**

**NECESITO EL EJERCICIO MANUSCRITO POR EL ALUMNO Y LA HOJA DE LECTURA ÓPTICA PARA PODER CALIFICAR.**

**SI ESTE EJERCICIO NO SE HA IMPRESO EN HOJA DE LECTURA ÓPTICA, SOLICITE UNA AL TRIBUNAL.**

Todas las preguntas de este ejercicio son eliminatorias en el sentido de que debe obtener una nota mínima en cada una de ellas. En cada pregunta teórica, que se valora con 2'5 puntos, la nota mínima es 1 punto; en la segunda parte (ejercicio de teoría aplicada que se valora con 5 puntos) la nota mínima que debe obtener es de 2 puntos.

**¡ATENCIÓN! PONGA SUS DATOS EN LA HOJA DE LECTURA ÓPTICA QUE DEBERÁ ENTREGAR JUNTO CON EL RESTO DE LAS RESPUESTAS.**

Conteste a las preguntas teóricas, en cualquier orden, en hojas diferentes a las que utilice para la contestación de la segunda parte. En cada parte, la cantidad MÁXIMA de papel (de examen, timbrado) que puede emplear ESTÁ LIMITADA al equivalente a DOS (2) HOJAS de tamaño A4 (210 x 297 mm)

PRIMERA PARTE. PREGUNTAS TEÓRICAS (2'5 PUNTOS CADA UNA)

1. La metodología llamada 'programación estructurada', ¿qué técnica utiliza en la fase de diseño?

**Solución**

La metodología de programación conocida como 'programación estructurada' se empezó a utilizar a primeros de los años 70, siendo los trabajos clásicos de Dijkstra, Dahl y Wirth los que le dieron popularidad. En la fase de diseño de esta metodología se utiliza una técnica (muy intuitiva) llamada 'desarrollo por refinamiento progresivo' y consiste en descomponer el sistema desde un punto de vista funcional y ver el sistema entero como una operación (función) global y, a partir de ahí, ir descomponiéndolo siempre en funciones más sencillas.

Véase sección 4.2.1 del libro de texto.

2. Razone qué ventajas puede tener la técnica de integración llamada 'sándwich'.

**Solución**

La técnica 'mixta' o 'sándwich', al integrar tanto descendente -con los módulos de nivel alto- y ascendente -con los de nivel bajo-, nos ofrece las ventajas de ambas técnicas al mismo tiempo. Permite ver desde un principio las posibilidades de la aplicación al ir integrando de forma descendente y, al mismo tiempo, se puede trabajar en paralelo con otros desarrolladores que estarán integrando de forma ascendente; empezando con los módulos de nivel bajo.

SEGUNDA PARTE. PREGUNTA DE TEORÍA APLICADA (MÁXIMO 5 PUNTOS)

3. *Modele, con un diagrama de objetos, la composición de un ejército griego según el siguiente enunciado:*

*En la antigua Grecia, la principal unidad del ejército era la infantería, que estaba compuesta por tres tipos de soldados: 1) los hoplitas, pesadamente equipados con un escudo, una lanza, y una espada; 2) los*

peltastas, que portaban un equipamiento más ligero compuesto por varias jabalinas; y 3) los gimnetas, que llevaban una honda.

La principal organización táctica para la guerra era la falange, que luego fue imitada por varias civilizaciones mediterráneas. La falange estaba formada por una profunda fila de hoplitas.

### Solución

