Tarea 1

Entornos de desarrollo

José Carlos Manjón Carrasco

Supuesto práctico:

Una notaría desea crear una aplicación que facilite el trabajo en equipo entre sus empleados para poder compartir toda la información necesaria entre los mismos, pero sin que sea accesible desde fuera de la notaría. El procedimiento ideado es crear grupos de trabajo diferenciados, es decir, un grupo de trabajo para los administrativos, otro para la asesoría legal, otro para los contables/facturación, otro para los copistas y otro para la atención al cliente. Cada uno de estos grupos estarán dirigidos por un responsable, que será el que se comunique con el notario. Cada trabajo será asignado a un miembro del equipo que será el responsable de la documentación y los demás tendrán acceso a todos los archivos correspondientes de esa documentación pudiendo leer, aportar o modificar su contenido, a excepción de las escrituras, sobre las que sólo tendrán acceso de lectura. El sistema deberá tener un control de versiones y evitar los accesos simultáneos, permitiendo en todo momento saber quién tiene el acceso al documento y dando la posibilidad de solicitar el tomar el control previa aceptación por parte de quien lo tenía. El notario deberá conocer el estado de la documentación a través de un informe que elaborará el responsable de la documentación y habrá un control de tiempos para conocer quién ha tenido acceso a los archivos, durante cuánto tiempo y el tipo de acceso.

Esta notaría tiene cierta urgencia para que se le desarrolle su software a medida, especificando que sea una aplicación de escritorio robusta y segura, con un software instalable sólo en los equipos que ellos decidan. Debido a esta urgencia, se le irán entregando partes de la aplicación funcionales conforme se vayan desarrollando, comenzando con la principal que es el trabajo colaborativo. La notaría quiere que haya controles de acceso y que tan solo los miembros de un mismo grupo puedan acceder a sus documentos. En una segunda versión se implementará un sistema de comunicación instantáneo entre los miembros de un mismo grupo y de comunicación entre todos los responsables de cada grupo con el notario. Se deberá velar por la confidencialidad, y deberán establecerse los mecanismos de seguridad necesarios.

Teniendo en cuenta el citado supuesto práctico, responde a las siguientes preguntas:

1. Según la cercanía que esté del lenguaje humano, ¿sería adecuado utilizar un lenguaje de programación de bajo nivel? Justifica tu respuesta. Según la técnica de programación utilizada, ¿sería adecuado utilizar un lenguaje de programación estructurado? Justifica tu respuesta. En ambos casos, ¿cuáles serían las elecciones correctas?

Un lenguaje de bajo nivel es aquel que está más cercano al cómo funciona internamente un ordenador, se podrían algún lenguaje ensamblador o lenguaje máquina, esta opción aunque podría ser válida, no es recomendable, sobre todo por las inconveniencias que plantearía, como podría ser encontrar al autor del código, estos lenguajes no son muy populares, es único para cada procesador, por lo que si hubiera ordenadores con diversos procesadores, se multiplicaría el trabajo, otro inconveniente es que el lenguaje ensamblador en el código se hace referencia a la ubicación física de los archivos, por lo que no se podría reutilizar código, multiplicando el trabajo a realizar y, por tanto, el coste, por lo que en mi opinión se debería usar un lenguaje de alto nivel o visual, ya que los inconvenientes anteriores los tienen eliminados.

2. ¿Sería adecuado utilizar un lenguaje de programación estructurado?

En mi opinión, no sería adecuado, porque en un lenguaje estructurado el código está en un único bloque, complicando la programación y el mantenimiento, como sería el caso de la notaría, al ser un desarrollo con bastantes retos.

La elección correcta, sin entrar al lenguaje concreto, recomendaría un lenguaje de alto nivel o si acaso visual, ya que sería más fácil el desarrollo y mantenimiento, puesto que son tipos de lenguaje más populares, con uno de esos lenguajes evitaríamos el realizar trabajo múltiple y, por tanto, el coste al ser independiente del procesador y poder usar un mismo código para diversos ordenadores, eso sí, debemos tener en cuenta el sistema operativo sobre el que nos vayamos a basar aunque Windows es el más popular, podríamos perfectamente basarnos en Linux o Apple y esto nos puede plantear diversos retos.

3. El software que se va a instalar en los equipos de la notaría, ¿sería código objeto? Justifica tu respuesta. ¿Cuál sería el tipo de código correcto?

El software que se instale en los equipos, no sería código objeto, ya que este código es un código intermedio, obtenido de traducir código fuente (el código que hace el desarrollador/a) a un código equivalente formado por unos y ceros que aún no puede ser ejecutado directamente por los equipos, dicho de otra forma, es el código resultante de la compilación del código fuente que consiste en un código binario que

está distribuido en varios archivos, cada uno de los cuales corresponde a cada programa fuente compilado.

El tipo de código correcto sería un código ejecutable, que definimos como el código que se obtiene una vez que se une el código objeto con las librerías necesarias a través de un compilador y enlazador (enlazad el código objeto con las librerías necesarias) para poder ejecutar el programa, este código ya si lo entiende el ordenador, siendo el sistema operativo será el encargado de cargar el código ejecutable en memoria RAM y proceder a ejecutarlo.

4. ¿Qué tipo de lenguaje deberíamos utilizar si necesitáramos independencia de nuestro software con el hardware concreto del equipo? Justifica tu respuesta. Pon un ejemplo concreto del lenguaje de programación.

Esa independencia de la que habla el enunciado, lo que nos quiere decir es que se pueda ejecutar en cualquier ordenador, independientemente del hardware que tenga, se trata de una característica muy interesante, para ello necesitaríamos usar una capa intermedia entre el código objeto y el hardware del equipo donde se está ejecutando, esa capa intermedia se llama “Máquina virtual”, el software que desarrollásemos se instalaría en la “Máquina virtual”, por lo que no se ejecutaría con el hardware físico del equipo real, conseguiríamos entre otras cosas que el software sea portable, no ocupar memoria si no es necesario, más seguro pues no se ejecuta en el equipo directamente, se verifica todo el código antes de ejecutarlo, la “Máquina virtual” protege direcciones de memoria, la “Máquina virtual” gestiona el control del hardware físico necesario para ejecutar el software.

Todos los lenguajes no son susceptibles de ejecutarse mediante máquina virtual, sólo son válidos lenguajes compilados (C, C++, Java entre otros) e interpretados (JavaScript, Python, Basic).

5. Previa instalación de nuestra aplicación en los equipos de la notaría, ¿qué requisitos hardware deben cumplir estos equipos? ¿qué software deberemos tener instalado previamente?

El propósito de la aplicación , además de ejecutarse correctamente, debe ser lo más universal posible, es decir, que no sólo se pueda ejecutarse en un ordenador con un hardware y software concreto, sino todo lo contrario, por lo que antes de analizar el software y hardware de los ordenadores de la notaría, debemos analizar nuestra aplicación y determinar qué requisitos y características necesita la aplicación para poder ejecutarse correctamente, y en base a esa información comparar con los equipos informáticos donde se instalaría y determinar si es posible o no, ya que según se haya desarrollador es posible que se necesiten librerías, plataformas , sistemas operativos … , que el ordenador de destino no pueda o sea tan caro la actualización , que se necesite comprar un ordenador nuevo, un ejemplo claro, podría ser la

instalación de un videojuego en los que antes de realizar la instalación y poder ejecutarlo se hace una comprobación del ordenador de destino, y en el ordenador de destino se analiza el hardware:

- Sistema operativo: Windows, Mac, Linux…
- Procesador: fabricante, velocidad, normalmente se establece el mínimo con el que funciona correctamente, en este caso también puede afectar el fabricante ya que cada fabricante realiza una arquitectura interna que puede impedir funcione, como por ejemplo actualizar a Windows 11 con procesadores AMD
- Memoria Ram, lo habitual es establecer la memoria mínima, con la que funciona correctamente
- Espacio mínimo de instalación, ya sea en disco duro, tarjeta de almacenamiento, memoria USB o en lugar de destino
- Tarjeta gráfica, lo habitual es establecer la memoria mínima, con la que funciona correctamente, en este caso también puede afectar el fabricante
- Resolución mínima recomendada del monitor, pues el diseño gráfico se basa en una resolución a partir de la cual se ve todo correctamente, si se varía de forma extrema, la visualización y experiencia de usuario puede verse muy afectada

En cuanto al software:

- Plataforma del software, sistema operativo: Windows, Mac, Linux…
- Otros paquetes software adicionales necesarios
- Actualizaciones de seguridad, antivirus
- Navegador, si fuera necesario, por ejemplo, en todo lo referente con la administración pública se debe usar Internet explore o Edge, ya que da menos problemas

6. Para el proceso de traducción de código fuente a código objeto, ¿usaremos un compilador o un intérprete? Justifica tu respuesta.

El proceso de traducción de código fuente, el que realiza el desarrollador/a, a código objeto, el que entiende el ordenador, depende del tipo de lenguaje elegido, pues

Compilación: El proceso de traducción se realiza sobre la totalidad del código fuente, en un solo paso. Se crea código objeto que habrá que enlazar. El software responsable se llama compilador.

Interpretación: El proceso de traducción del código fuente se realiza línea a línea y se ejecuta simultáneamente. No existe código objeto intermedio. El software responsable se llama intérprete. El proceso de traducción es más lento que en el caso de la compilación, pero es recomendable cuando el programador es inexperto, ya que da la detección de errores es más detallada.

7. Realiza en ejemplo de documento ERS, según las especificaciones dadas por la notaría. ¿En qué fase del desarrollo se encuentra este documento?

Estaríamos en la primera y más vital fase del desarrollo, en la Análisis de requisitos, pues es en esta fase donde se establecen los requisitos funcionales y no funcionales que el software debe cumplir, en esta fase, se habla con el cliente, en el caso del supuesto práctico con el responsable la notaría, el notario, para saber qué quiere el notario que el software haga y también el cómo, se deben realizar las reuniones que sean necesarias, pues una buena toma de requisitos evitará problemas a futuro, y hará que el desarrollo sea más sencillo para equipo desarrollo y también más barato para el cliente.
A continuación, se encuentra el ejemplo de documento ERS, he realizado la parte posible con los conocimientos actuales.

Nombre del Software: Gestión de Notaría 1.0

Autor: José Carlos Manjón Carrasco

Nombre del Proyecto: Gestión Notaría

1. Introducción

Este documento es la Especificación de Requisitos Software (ERS) para el software Gestión de Notaría 1.0.

1.1. Alcance

El alcance se limita a la notaría que ha encargado el desarrollo del software ad hoc.

1.2. Objetivos

Gestión de una notaría, entendiendo gestión como el control de la información, crear roles y definición de estos, control de versiones de la documentación, control de tiempos, seguridad y protección de datos y sistemas de comunicación interno.

2. Análisis de requerimientos de Software

2.1. Descripción general y definición

Gestión de Notaría 1.0, ha de ser un software portable, es decir, se pueda ejecutar con independencia de software y hardware del equipo en el que se esté ejecutando, siendo su objetivo la digitalización de la gestión de la notaría, teniendo en cuenta una alta seguridad debido a la naturaleza de la documentación y trabajos a realizar.

2.2. Requerimientos específicos

2.2.1. Requerimientos funcionales

- Debe permitir compartir información/ documentación
- Creación de roles para los usuarios:
    - Administrativos
    - Asesoría legal
    - Contables /facturación
    - Copistas
    - Atención al cliente
    - Responsable de grupo
    - Notario
- Un responsable por cada rol, Responsable de grupo, salvo el rol Notario, el cual podrá comunicarse con el notario
- El responsable de grupo tendrá la capacidad de realizar un informe que sólo compartirá con Notario, un informe de la documentación en el que se detalle
    - El documento en cuestión
    - Control de tiempos
        - Quien ha tenido acceso a qué archivo
        - Durante cuanto tiempo
        - Tipo de acceso

- Creación de grupos de usuarios, cada grupo podrá compartir documentación entre el mismo grupo y no fuera
- Control de versiones
- Control de accesos para evitar accesos múltiples para un usuario
    - Creación de usuarios, que tendrá un determinado rol
- Control de la documentación se debe conocer quien tiene el documento abierto
- Posibilidad de pedir control del documento si está abierto por otro usuario,
    - Darlo si el primero lo concede
    - Rechazarlo si el primero no lo concede
- Tras la asignación de un trabajo, cada usuario, será el responsable que podrá gestionar la documentación, el resto de los usuarios siempre que sean del mismo grupo podrán leer, aportar y modificar el contenido a excepción de las escrituras que sólo podrán leer
- Creación de un sistema de comunicación interno, chat, entre los usuarios de un mismo rol o grupo de trabajo, y otro para los Responsables de grupo y Notario
- Control de excepciones y errores

2.2.2. Requerimientos no funcionales

- Creación de una intranet, para evitar posibles ataques externos y mejorar la seguridad
- Usabilidad, es decir, funcional y de fácil transición
- El mantenimiento debe ser sencillo y requerir el menor tiempo de baja posible
- Cumplir con todo lo referente a la ley de protección de datos
- Alto rendimiento, debe ser lo más rápido posible
- Posibilidad de admitir múltiples formatos y tamaños de documentación
- El software debe poder instalarse en determinados equipos sin valorar inicialmente, el software ni hardware
- Realización diaria de copias de seguridad, de toda la información almacenada

8. ¿Qué ventajas e inconvenientes tendría la utilización de un framework para desarrollar el proyecto solicitado? ¿Qué ventajas e inconvenientes tendría la utilización de un entorno de desarrollo para desarrollar el proyecto solicitado? ¿Cuál de las anteriores opciones utilizarías? Justifica tu respuesta.

Antes de hablar sobre ventajas en inconvenientes de un framework y de un entorno de desarrollo, debemos tener un poco de conocimientos.
Un framework, es una plataforma de ayuda al programador/a, de ayuda porque permite desarrollar proyectos sin tener que empezar desde 0, es decir, se trata de una agrupación de librerías, que te dan la posibilidad de reusarlas y reducir el código que debemos crear para un requerimiento dado.
Ventajas de utilizar un framework:

- Desarrollo rápido de software, el programador ahorra tiempo ya que lo básico para empezar a codificar viene dado,
- Reutilización de partes de código para otras aplicaciones, al poder analizar el código de proyectos similares, se podría usar ese mismo código para acelerar el desarrollo
- Diseño uniforme del software, sería como hablar todos el mismo lenguaje, de no ser así, se crea mucha dependencia del programador/a y en caso de necesitar que otra persona a la que inició el proyecto lo termine, no podría hacerlo, puesto que tendría que hacer un ejercicio de ingeniería inversa tan elevado, que sería mejor empezar desde 0
- Portabilidad de aplicaciones de un ordenador a otro, ya que los bytecodes que se generan a partir del lenguaje fuente podrán ser ejecutados sobre cualquier máquina virtual, no ciñéndonos a un ordenador concreto

Inconvenientes:

- Gran dependencia del código respecto al framework utilizado (sin cambiamos de framework, habrá que reescribir gran parte de la aplicación), cada framework tiene características que lo hacen único
- La instalación e implementación del framework en nuestro equipo consume bastantes recursos del sistema
- Curva de aprendizaje, cada framework es único, por lo que no se puede cuantificar el tiempo que le requerirá a una persona conocer a fondo cada software
- Si el software es pequeño, puede no merecer la pena el uso de un framework, ya que cada framework sigue una forma de trabajo, y si es pequeño el software, puede requerir más tiempo y esfuerzo el poner todo a punto que luego el desarrollo en sí
- Cada framework impone una forma de trabajar, por lo que deja poco a la elección de quien haga el software
- Incremento de tamaño de programa, que puede no ser necesario, cada framework lleva consigo una serie de estructuras y funcionalidades para que

funcione correctamente que puede no ser necesario, y se aumente el tamaño y complejidad del software sin necesidad alguna

Un entorno de desarrollo, (IDE), es un tipo de software compuesto por un conjunto de herramientas de programación, cuyo objetivo es la construcción de otro software, se compone de: editor de código de programación, compilador, intérprete, depurador y de constructor de interfaz gráfico, es decir, se trata de un software donde escribes tu código, el cual te dará la posibilidad de desarrollar, y depurar más rápidamente.

Ventajas:

- Proporcionan al programador una serie de componentes con la misma interfaz gráfica
- Todo lo necesario está en un único lugar, como, por ejemplo
    - Editor de código: podemos picar nuestro código directamente
    - Depurador: el depurador, permite localizar y solucionar errores de una manera mucho más rápida y fácil
    - Un compilador: no necesitamos usar comandos, el entorno lo hace directamente
- Aumento de eficiencia y reducción de tiempo de codificación
- Los actuales soportan diversos lenguajes
- Hay entornos libres, por lo que no es necesario comprar la licencia para poder usar el entorno de desarrollo
- Normalmente fácil de usar

Desventajas:

- El entorno de desarrollo más interesante, puede ser propietario y no libre, por lo que se puede incurrir en un gasto de compra de licencia antes de escribir una línea de código
- Puede no existir para todos los sistemas operativos
- Algunos entornos de desarrollo no permiten interactuar con la base de datos directamente

Teniendo en cuenta las definiciones, las ventajas y desventajas, además las características del software que debemos realizar, me parece que el entorno de desarrollo es más interesante, ya que nos permite el desarrollo y todo lo necesario, mientras que el framework, es más bien una plataforma de ayuda, por lo que no serviría completamente para un desarrollo tan específico.