UNIVERSIDAD DE SANTIAGO DE CHILE
FACULTAD DE INGENIERÍA
DEPARTAMENTO DE INGENIERÍA INFORMÁTICA

Paradigmas de Programación

Laboratorio N°4

Alumno: Israel Arias Panez.
Sección: B-2.
Profesor: Víctor Flores Sánchez.

Santiago - Chile
2-2020

TABLA DE CONTENIDOS

CAPÍTULO 1. INTRODUCCIÓN

En el presente informe se describirá el desarrollo de la solución al cuarto laboratorio de la asignatura de Paradigmas de programación. Se efectuará una revisión del problema, se describirá el paradigma utilizado para el desarrollo de la solución además de las herramientas utilizadas, también se hará un contraste de resultados conseguidos con este paradigma en comparación al paradigma funcional, al paradigma lógico y al paradigma orientado a objetos, los cuales fueron usados en las tres primeras entregas de laboratorio. Además, se presentarán instrucciones de uso de la solución desarrollada.

1.1.1 Descripción del problema

En esta cuarta entrega de laboratorio se continua con la misma tarea de las entregas anteriores, el desarrollo de una simulación de un software de sistema de foros, tomando como referencia el conocido foro web “StackOverflow”, sin embargo, para esta entrega debe ser implementado usando el paradigma orientado a objetos y el paradigma dirigido por eventos en el lenguaje de programación Java o C#, además debe contar con una interfaz gráfica. Para esta ocasión se decidió por hacer la implementación en C# con WPF (Windows Presentation Foundation) como biblioteca de entorno gráfico, para lograr simular el sistema de foro, se debe implementar características que tiene un foro, como, por ejemplo: registrar usuarios, crear preguntas, responder preguntas, además cada usuario tiene características como username, contraseña, reputación, todas las preguntas que ha hecho, etc. Las preguntas tienen cantidad de votos, respuestas a la pregunta, fecha de publicación, recompensa por la pregunta, entre otros. En la sección 2.1.1 del informe se presentarán los requisitos específicos de implementación para esta entrega de laboratorio.

1.1.2 Descripción de los paradigmas utilizados

Para esta cuarta entrega del laboratorio, se solicita hacer uso del paradigma orientado a objetos en conjunto del paradigma dirigido por eventos, la principal característica del paradigma orientado a objetos es la de, como su nombre indica “programar objetos”, un objeto tiene sus atributos que lo caracterizan y además tiene sus propias acciones o métodos que pueden ejecutar, por ejemplo, en la implementación realizada en esta entrega una pregunta tiene características como título, contenido, fecha, autor, estado, recompensa, etc. Por otra parte el paradigma dirigido por eventos tiene como principal característica la de que los eventos “dirigen” el flujo y estructura del programa, o sea al contrario de lo que ocurre por ejemplo en la programación imperativa donde el desarrollador fija el orden de ejecución de los distintos algoritmos como si de una receta se tratase, acá el flujo es controlado por el usuario a través de los eventos que el mismo provoca, un evento es una acción que puede ser detectada por un programa, como por ejemplo el cliquear el botón con una X que se encuentra en la esquina de un programa provocara que se cierre el programa, ya que al cliquear el botón (acción) provoco la ejecución del algoritmo de cierre del programa, en la solución implementada para esta entrega se pueden ver distintos eventos que ocurren en un foro, los cuales son desencadenados al tocar botones, seleccionar elementos de listas, doble-clickear un elemento, etc.

CAPÍTULO 2. SOLUCIÓN DEL PROBLEMA

2.1.1 Análisis del problema respecto a sus requisitos específicos.

Para el desarrollo de la solución se solicitan los siguientes requisitos funcionales específicos:

- Implementación de distintas clases y estructuras para cumplir con todos los siguientes requisitos funcionales que se listaran a continuación.
- Interacciones a través de interfaz gráfica: La interacción del usuario con con el sistema de foro implementado debe suceder gracias a una interfaz gráfica que desencadena los distintos eventos.
- Register/login/logout: Permite registrar nuevos usuarios en la plataforma, a la vez de permitir el ingreso y salida de la plataforma.
- Ask: Permite a un usuario con su sesión iniciada realizar una pregunta.
- Answer: Permite a un usuario el añadir una respuesta a una pregunta que exista dentro del sistema de foro.
- Reward: Permite a un usuario el ofrecer puntos de recompensa a una pregunta, la cual se descuenta de sus puntos de reputación y serán otorgados al usuario cuya respuesta a esa pregunta sea aceptada.
- Accept: Permite a un usuario aceptar una respuesta en alguna de sus preguntas, cerrando de esta manera la pregunta, otorgando la recompensa, si correspondiese, al usuario cuya respuesta fue aceptada y además efectúa la actualización de puntajes correspondiente.

También se solicita implementar una funcionalidad complementaria, para este laboratorio se eligió implementar la función vote:

- Vote: permite a un usuario con sesión iniciada, votar positiva o negativamente una pregunta o respuesta que se encuentre en el sistema de foro, al hacerlo se registrara el voto en la pregunta o respuesta y se sumara o restara reputación a los usuarios según corresponda.

Tomando en consideración los requisitos funcionales quedan claras las funcionalidades que deben implementarse en esta entrega de laboratorio, además es posible darse cuenta que es conveniente ir realizando la implementación de los predicados bajo el mismo orden que se ha presentado arriba, debido a que por ejemplo las funcionalidades ask, answer, reward, accept y vote necesitan de la funcionalidad login para funcionar, por lo cual para esta entrega, se implementaran los requerimientos funcionales siguiendo el orden entregado.

2.1.2 Diseño de la solución.

El diseño de la solución comenzó tomando como base el diagrama de diseño de la entrega anterior de laboratorio, por lo que corresponde al diagrama de análisis para el diseño de esta solución (Ver anexo 1 disponible en el capítulo 3 del informe).
Para poder desacoplar de una manera ordenada la lógica del programa se decidió que la solución seguiría el patrón de arquitectura MVC (modelo-vista-controlador) lo que permite tener el programa dividido en tres capas, modelo: el cual se encarga de definir toda la estructura y clases del proyecto. Controlador: que se encarga de ocupar elementos del modelo para hacer acciones y luego comunicárselo a la vista. y finalmente la vista: la cual contiene toda la representación gráfica que ve el usuario y se encuentra comunicada con el controlador.
Una vez definida la arquitectura a utilizar, la solución se implementó bajo una división de problemas en subproblemas, en primer lugar, se hizo la creación de todas las clases que serían ocupadas en el modelo: Etiqueta, Pregunta, Respuesta, Stack y Usuario. Luego de definir todas las clases se hizo un bosquejo de cómo sería la vista del usuario, se definió la cantidad de ventanas existentes y donde iría ubicado cada elemento y como se presentaría cada elemento del foro, una vez decidido el diseño de la vista que llevara la solución se procedió a ir creando cada ventana y cada método del controlador que es usado por la ventana, bajo este proceso y gracias a la estructuración del laboratorio mediante los requisitos funcionales descritos en el punto 2.1.1, se implementó cada requisito funcional en orden esto es: implementar las funcionalidades de register, login, luego ask y así sucesivamente, siempre creando su vista respectiva que permitiera el ingreso de datos para todas ellas, para finalmente combinar todas esas y así lograr la simulación del foro, lo que también es conocido como divide y vencerás, “El esquema de dividir y vencer(también conocido como divide y vencerás) es una técnica para resolver problemas que consiste en dividir el problema original en subproblemas (de menor tamaño), resolver los subproblemas y finalmente combinar las soluciones de los subproblemas para dar una solución al problema original”.[1]
La solución se implementó en el IDE Visual Studio en el lenguaje C# con la biblioteca gráfica WPF (Windows Presentation Foundation) biblioteca que separaba la parte grafica de la vista y la lógica de la vista en dos archivos distintos, la parte gráfica siguiendo el lenguaje declarativo XAML y la lógica de la vista siguiendo el lenguaje de C#, cabe destacar que el uso de las herramientas de Visual Studio permitió crear la vista sin la necesidad de interiorizarse demasiado en el aprendizaje del lenguaje XAML, insertar imágenes, crear botones, etc. Otra característica importante que cabe destacar que fue usada en la implementación de esta simulación de foro es el “Data Binding” disponible en el lenguaje XAML, que permitió anclar datos por ejemplo el username para que fuera desplegado por

una caja de texto de manera mucho más simple que si fuera hecho usando código y métodos de C# y todo siendo definido en la misma ventana gráfica escrita en XAML.
Finalmente, la solución se puede ver gráficamente mediante el diagrama de diseño (Ver anexo 2 disponible en el capítulo 3 del informe).

2.1.3 Aspectos de implementación.

La implementación de este programa se realizó en el lenguaje C#, usando el IDE Visual Studio y la biblioteca grafica WPF, la elección de este conjunto de opciones y no el uso de Java fue debido a que .NET es un framework más actualizado y debido al interés de aprender un nuevo lenguaje. Cabe destacar que por esta elección la solución creada solo es posible ser ejecutada en el sistema operativo Windows.

La estructura del proyecto sigue la arquitectura MVC (modelo-vista-controlador), la cual puede ser observada de una mejor manera en el diagrama de diseño disponible en el anexo número 2 del capítulo 3 de este informe.

2.1.4 Instrucciones de uso.

A diferencia de otras entregas de laboratorio anteriores, este laboratorio gracias a su característica de poseer una interfaz gráfica, su uso se hace muy intuitivo sin necesidad de otorgar instrucciones detalladas de uso.

A continuación, se explicará las instrucciones para poder usar el programa de una manera adecuada:

- En primer lugar, abrir el ejecutable vista.exe
- Ahora se muestra la ventana inicial y es posible el uso del programa, la solución está programada para cargar información inicial apenas inicie la ejecución, de manera de poder ver mucha de sus funciones sin tener que crear todo desde cero.

Se adjuntan imágenes del programa y distintas opciones en el anexo disponible en el capítulo 3 de este informe.

2.2 Resultados y autoevaluación.

En esta sección se hará una revisión de los resultados, para ello se verificarán los requisitos no funcionales y funcionales solicitados

Requerimientos no funcionales:

Lenguaje y herramientas de trabajo:	Logrado: el lenguaje usado en el programa es C# en su versión 8.0 con el IDE Visual Studio y WPF como biblioteca de entorno gráfico.
Interacciones con el programa:	Logrado: Todas las interacciones se hacen a través de una interfaz gráfica, creada gracias a la biblioteca de entorno grafico WPF.
Uso del paradigma**:**	Logrado: Todas la implementación y diseño sigue los lineamientos del paradigma orientado a objetos, presentando características del paradigma como: relaciones, métodos, objetos, sobrecargas, entre otros. Además, se sigue el lineamiento del paradigma dirigido por eventos, el programa reacciona a las distintas acciones que realiza el usuario y es el usuario el que define el flujo del programa.
Separación modelo – vista:	Logrado: No existe acoplamiento alguno entre los objetos del modelo y la vista gracias a la implementación siguiendo la arquitectura MVC.
Prerrequisitos:	Logrado: Cada funcionalidad tiene implementada su prerrequisito.
Documentación:	Logrado: El código se encuentra documentado siguiendo los comentarios tipo XML
Organización del código:	Logrado: Al estar implementado siguiendo la arquitectura MVC, el programa goza de una alta cohesión y un bajo acoplamiento.
Diagrama de análisis:	Logrado: Existe el diagrama creado antes de cualquier implementación, es posible visualizarlo en el anexo al final del informe.
Diagrama de diseño:	Logrado: Existe el diagrama, el cual fue creado post implementación de la solución, es posible visualizarlo en el anexo al final del informe
Uso de git:	Logrado: existen 16 commits, el primero hecho el 31 de enero de 2021, hasta el último hecho el 27 de febrero de 2021.

Requerimientos Funcionales:

Todos los requerimientos funcionales funcionan sin problemas, cumpliendo con las restricciones solicitadas para cada una de ellas, fueron probadas probándolas de distintas maneras y verificando su funcionamiento.

2.3 Conclusión.

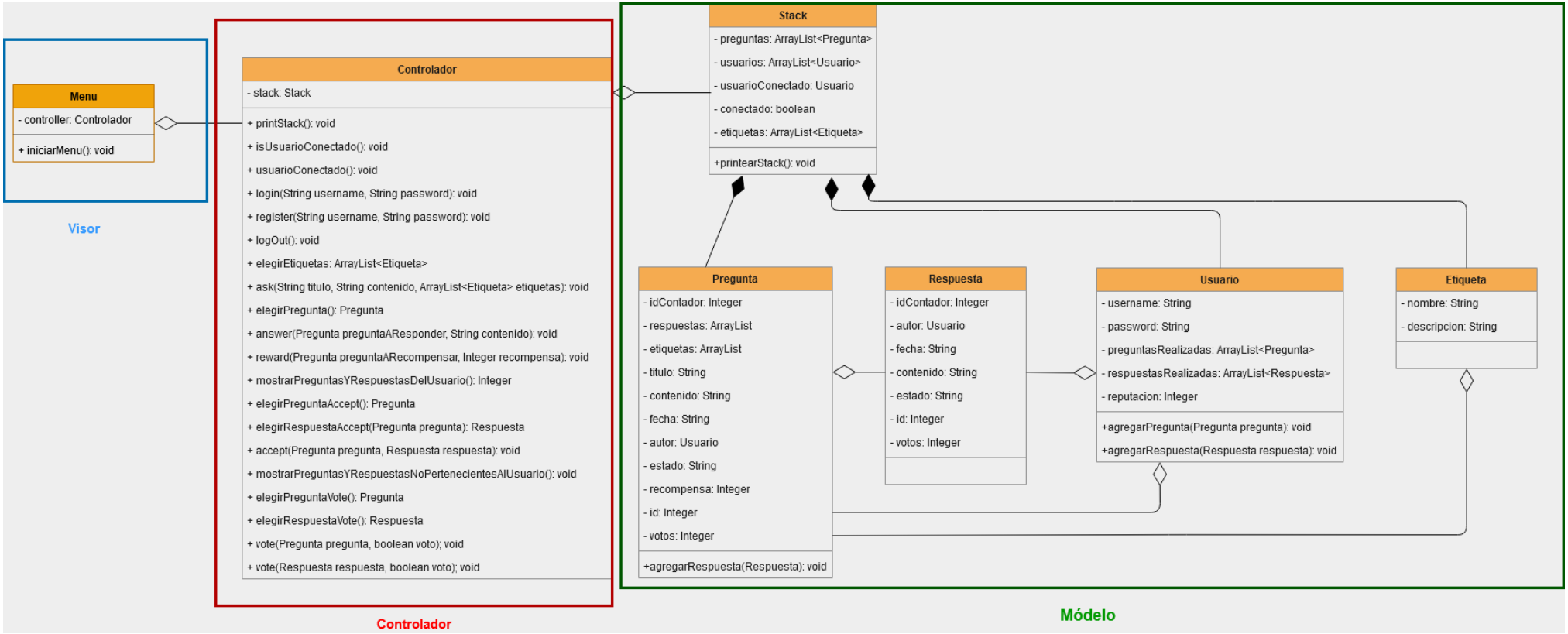
Tomando en consideración los resultados listados en la sección 2.2 de este informe, es posible concluir que se ha logrado cumplir con todos los objetivos requeridos para esta cuarta entrega de laboratorio, tanto funcionales como no funcionales.

En un inicio fue un desafío el entender el funcionamiento de la biblioteca grafica WPF y cómo funcionaba el paradigma dirigido por eventos en este entorno, también se dificulto el hecho de lograr vincular todos los eventos con todos los métodos creados y actualizar la vista de acuerdo con los cambios, fue la principal dificultad en esta entrega de laboratorio.

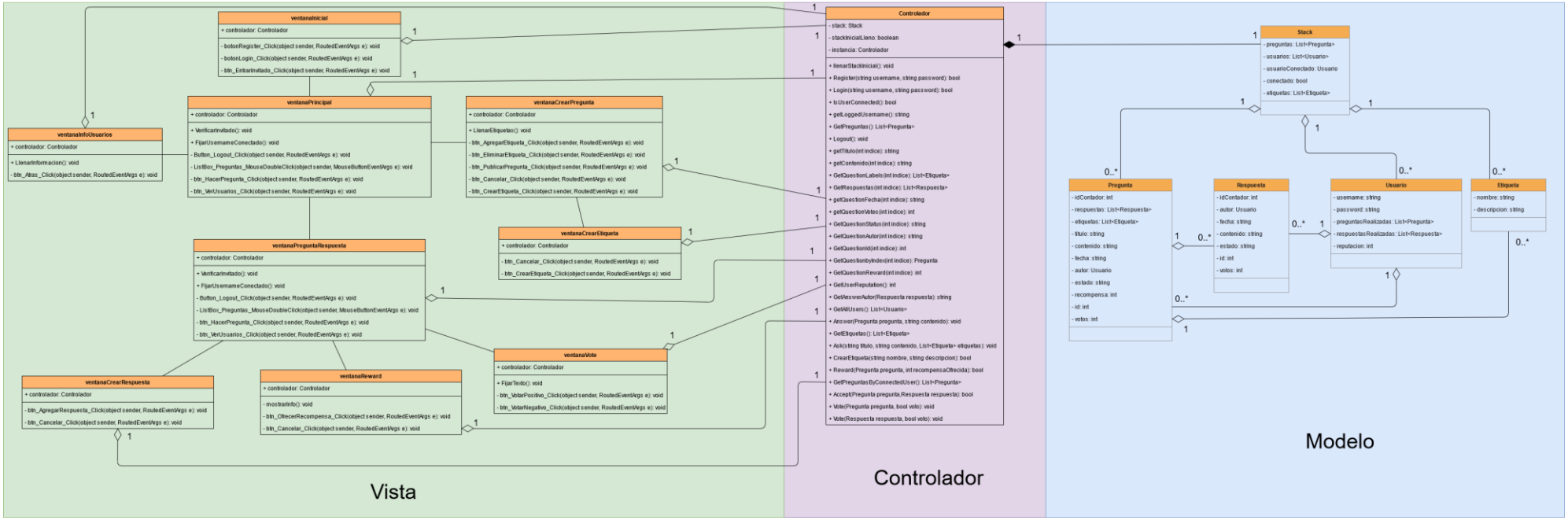
2.4 Referencias

[1] Duch A. (2006). *Esquema de Dividir y Vencer*. (Recuperado 01/03/2020). https://www.cs.upc.edu/~duch/home/duch/dyd.pdf

CAPÍTULO 3. ANEXO

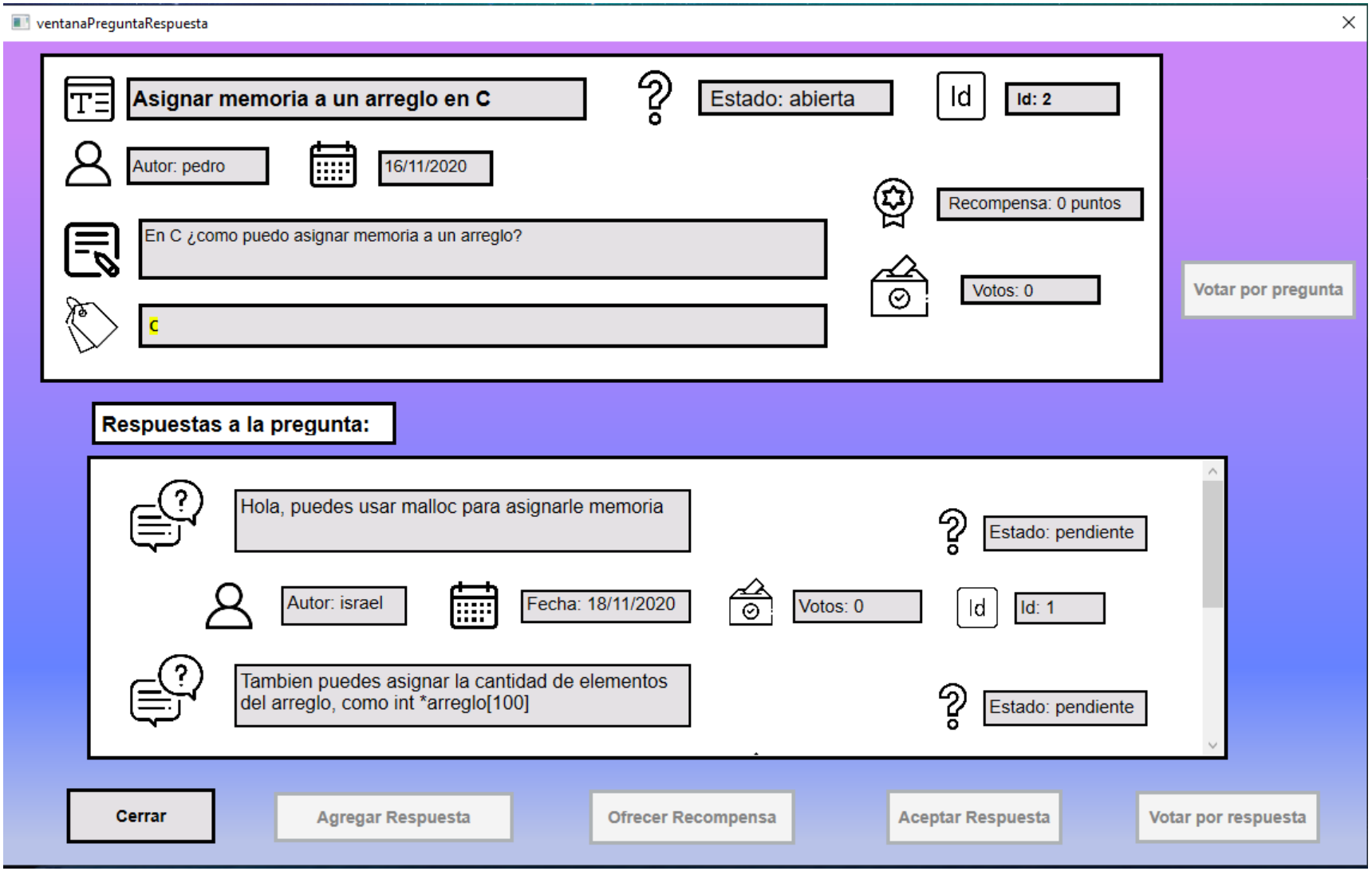


Anexo 1: Diagrama de análisis.



Anexo 2: Diagrama de diseño.

Anexo 3: Ventana inicial del programa.



Anexo 4: Ventana vista de pregunta del programa.