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

Paradigmas de Programación

Laboratorio N°3

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 tercer 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 y al paradigma lógico, los cuales fueron usados en las dos primeras entregas de laboratorio. Además, se presentarán instrucciones de uso de la solución desarrollada.

1.1.1 Descripción del problema

En esta tercera 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", el cual ahora debe ser implementado usando el paradigma orientado a objetos y el lenguaje de programación Java, para lograrlo, 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 para esta segunda entrega de laboratorio.

1.1.2 Descripción del paradigma utilizado

Para esta segunda entrega del laboratorio, se solicita hacer uso del paradigma orientado a objetos en conjunto del lenguaje de programación Java, 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 puede ejecutar, por ejemplo, un objeto auto tiene atributos como color, tamaño, marca y modelo. Y además posee métodos como: encender motor, acelerar, frenar, apagar motor, girar a la izquierda, girar a la derecha, etc. Java permite definir clases, de las cuales se pueden generar instancias, también conocidas como objetos, entonces el ejemplo anterior de auto sería una clase en Java, de la cual se pueden instanciar muchos autos con distintas características. En paradigmas anteriores vistos en la asignatura se podía ver la abstracción que se usa en el paradigma orientado a objetos al definir un objeto en la forma de TDA.
Ya que se hará uso del paradigma orientado a objetos, todas las características a implementar serán hechas como clases, las cuales se relacionarán con otras clases mediante distintas relaciones y métodos para lograr la simulación del foro.

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.
- Menú interactivo por terminal: El menú permite la interacción del usuario con el sistema de foro implementado y sus distintas funcionalidades.
- Register/login/logout: Permite registrar nuevos usuarios en la plataforma, a la ves 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.

Antes de comenzar con la implementación de la solución y tomando en cuenta el primer punto descrito en la sección 2.1.1 de este informe, la implementación del sistema de foro necesita de distintas clases y estructuras para funcionar, por ende, se creó un diagrama de análisis (Ver anexo 1 disponible en el capítulo 3 del informe).
Sin embargo, luego en las clases de catedra de la asignatura se presento el patrón de arquitectura MVC (model-view-controller), el cual permite tener una mejor organización del proyecto y producir una solución con mucha más cohesión y con menor acoplamiento que la solución inicialmente ideada.
Gracias a la estructuración del laboratorio mediante los requisitos funcionales descritos en el punto 2.1.1, la solución se implementó bajo una división de problemas en subproblemas, siguiendo los mismos subproblemas de los requisitos funcionales, esto es: Implementar primero las distintas clases y estructuras de base que necesite el foro, siguiendo la arquitectura MVC, implementar el menú interactivo por consola, para, a continuación, implementar las funcionalidades de register, login, luego ask y así sucesivamente, para finalmente combinar todas esas implementaciones en el menú 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]
A continuación, se presentará la lógica que implementan algunos métodos descritos en la sección 2.1.1 del informe:

- Implementación de Clases y estructuras:
  Para la creación de las distintas clases y estructuras se procuró usar la arquitectura MVC, por ende, en el modelo se crearon las clases: Stack, Usuario, Pregunta, Respuesta y Etiqueta, estas clases son las fundamentales para qué funcione el sistema de foro. En el controlador se creo la clase Controlador, la cual contiene la mayoría de la lógica y métodos del sistema de foros presentados en la sección 2.1.1 y finalmente en la vista se creó la clase Menu, el cual contiene toda la lógica del menú interactivo solicitado en la segunda sección y que además permite la interacción del usuario con las distintas funcionalidades del sistema de foro, para observar los distintos atributos y métodos de cada clase se recomienda ver el diagrama de diseño (disponible en el anexo 2 en el capítulo 3 del informe). Cabe destacar que la clase Stack es la que contiene las estructuras de datos que permiten almacenar los distintos tipos de datos, esto gracias a las ArrayList's.
- Menú interactivo por terminal:
  El menú interactivo se encuentra en la clase Menu, y presenta dos tipos de menus dependiendo del atributo de la clase Stack "conectado" si conectado es false solo se permitirá interactuar con el menú que permite loguearse, registrarse o salir del programa, si conectado es true permitirá la ejecución de todas las demás funcionalidades presentadas en la sección 2.1.1 del informe que involucran a un usuario con sesión activa en el stack. Los menus se encuentran construidos

gracias a un ciclo while para permitir su continua ejecución en conjunto a un interruptor switch case, el cual permite acceder a las distintas funcionalidades del sistema de foro, además se hace uso de try-catch para evitar errores como por ejemplo que el usuario ingrese una palabra en vez de un número, además de ello el menú se encuentra precargado con un stack predefinido, el cual se genera en el mismo constructor de la clase Stack al empezar la ejecución del programa.

- Register/login/logout
  Register fue posible implementarlo instanciando a la clase Usuario, cuyo constructor necesita el nombre de usuario y una contraseña, los cuales otorga el usuario mediante el menú, luego agregando ese objeto usuario recién creado a la ArrayList "usuarios" del Stack, sin embargo antes de todo ello se verifica con el nombre de usuario si ese nombre de usuario ya se encuentra registrado en el foro, para eso se recorre la ArrayList de usuarios del Stack, en caso de estar se omite el proceso antes mencionado y se muestra en pantalla que el usuario ya se encuentra registrado.
  Login funciona de una manera similar a register, recibe un nombre de usuario y una contraseña, luego se recorre la ArrayList de usuarios, buscando el nombre de usuario dentro de los usuarios ya registrados, en caso de encontrar el nombre de usuario ingresado por el usuario, se compara si las contraseñas ingresadas son iguales tanto del usuario ya registrado al ingresado, en caso de serlo se conecta al usuario, setteando en el stack al atributo usuarioConectado como el usuario que acaba de loguearse y al atributo conectado como true.
  Logout aprovecha la implementación de los dos menus en la clase Menu, cuyo despliegue depende del atributo conectado de la clase Stack, entonces logOut cambia el atributo conectado a false, de esta manera no hay usuario conectado y solo se muestra el primer menú que tiene a las funciones que no necesitan a un usuario con sesión iniciada.

- Ask:
  La funcionalidad ask funciona recibiendo el titulo de la pregunta, su contenido y una lista con las etiquetas que tendrá la pregunta crea una nueva pregunta instanciando a la clase Pregunta, cuyo constructor recibe las tres entradas antes mencionadas, luego usando el getter del stack se consigue al usuario conectado que hizo la pregunta y se vincula a la pregunta, la fecha de la pregunta se consigue automáticamente usando un método de la librería estándar de java "date", finalmente se agrega la pregunta a la ArrayList preguntas del Stack, lo que guarda la pregunta en el sistema de foros.

- Reward:
  La funcionalidad reward recibe Una pregunta a la cual se le fijara la recompensa, la cual se elige exteriormente mediante el menú interactivo, en conjunto de a recompensa ofrecida para la pregunta. Dentro del menú en la sección de reward se hace la verificación de que el usuario no ofrezca más recompensa de la que tiene actualmente y dentro de reward se verifica que la pregunta no se encuentre cerrada, en caso de que lo este no se fija la recompensa, en caso de no estar cerrada se le fija la recompensa, aumentando el artibuto de recompensa que tiene la Pregunta la cual se descuenta del puntaje del usuario que ofreció la recompensa.

- Accept:
  La funcionalidad accept recibe un objeto Pregunta y un objeto Respuesta, el objeto pregunta representa a la pregunta que será cerrada, y la respuesta a la respuesta que será aceptada, para cerrar la pregunta y respuesta de cambia el atributo “estado” que ambas poseen por “cerrada” y “aceptada” respectivamente, luego se verifica si la pregunta tiene una recompensa, en caso de tenerla se le otorga la recompensa al usuario cuya respuesta fue aceptada, todo mediante getters y setters.

- Vote:
  La funcionalidad de vote en un inicio, en el menú despliega todas las preguntas y respuestas del sistema de foro, para luego preguntar si se desea votar por una pregunta o por una respuesta, esto debido a que vote tiene dos definiciones dentro de la clase Controlador, gracias a la sobrecarga que permite el paradigma orientado a objetos, el cual permite tener dos métodos vote, que reciben distintos parámetros, uno para preguntas y otro para respuestas. Para ambos casos se solicitara el elegir la pregunta/respuesta por su id, sin embargo se hará la verificación de que el usuario no pueda elegir una de sus preguntas u respuestas para votar por ella, luego se preguntara si desea votar positiva o negativamente por la pregunta/respuesta, para finalmente registrar el voto, el cual se registra modificando el atributo votos que poseen tanto pregunta como respuestas, sumándole uno en caso de que sea un voto positivo y restándole uno en caso que el voto sea negativo, además se hace la distribución de puntajes correspondientes entre el usuario votante y el usuario cuya pregunta/respuesta fue votada, todo eso gracias a getters y setters.

2.1.3 Aspectos de implementación.

La implementación de este programa se realizó en el lenguaje Java, usando el IDE IntelliJ IDE, se hacen uso de distintas funciones, todas pertenecientes a la librería estándar de Java, por ejemplo java.util.Date, la cual permite capturar la fecha actual del sistema.

La estructura del proyecto es posible visualizarla de mejor manera en el diagrama de diseño disponible en el anexo número 2 del capitulo 3 de este informe.

Para compilar se prefirió la creación de un script de compilación .bat, el cual facilitará la labor de ejecutar el sistema de foros.

2.1.4 Instrucciones de uso.

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

- En primer lugar, abrir la carpeta del código fuente, dentro de ella se encuentra el archivo “scriptCompilacion.bat”
- Ejecutar el script de compilación anteriormente mencionado tocándolo dos veces, pasados unos segundos se habrá compilado y ejecutado automáticamente el proyecto.
- Puede interactuar con las diversas opciones del sistema de foros mediante el menú interactivo que se le presenta en pantalla, el cual también le retroalimentara las distintas opciones que usted use.
- Se recomienda crear un nuevo usuario para poder probar cada funcionalidad que posee el programa, sin embargo, si desea ocupar una cuenta que ya tiene preguntas y respuestas registras puede ocupar la cuenta de usuario “israel” con contraseña “qwerty”, a la cual puede acceder mediante la opción 1 del menú interactivo “Conectarse a su cuenta”.

  Nota: no se asegura que el script de compilación funcione en un sistema fuera de Windows, debido a que solo se pidió un script de compilación manual Batch en Windows o bash en Linux, se implemento el batch de Windows presente en este proyecto.

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:

La implementación debe ser en el lenguaje de programación Java, usando uno de los tres Ide's presentados. Además debe existir un script de compilación o proyecto en gradle	Logrado: el lenguaje usado en el programa es Java, utilizando OpenJDK en su versión 11 y usando el Ide IntelliJ IDEA Comunnity V.2020.3. Además se incluye un script de compilación Batch.
Interraciones con el programa:	Logrado: Todas las interacciones se hacen a través del menú interactivo, sin recurrir a ninguna librería externa que no sea la biblioteca estándar de Java.
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.
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 14 commits, el primero hecho el 27 de diciembre de 2020, hasta el último hecho el 10 de enero de 2021.

Requerimientos Funcionales:

Todos los requerimientos funcionales funcionan sin problemas, cumpliendo con las restricciones solicitadas para cada una de ellas, fueron probadas ingresándoles distintas entradas a cada función y verificando que la salida fuera la esperada de acuerdo con la entrada.

2.3 Conclusión.

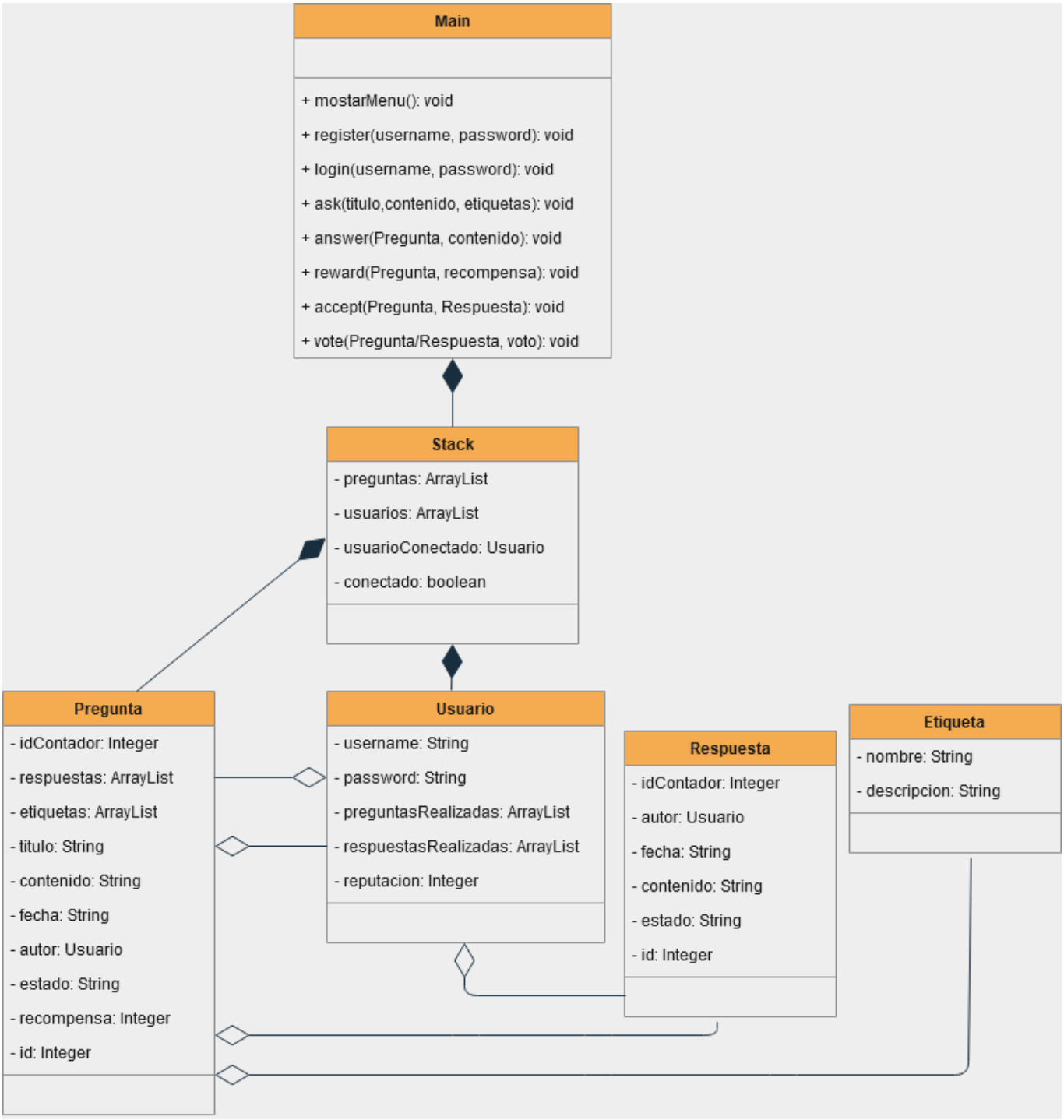
Tomando en consideración los resultados listados en la sección 2.2 de este informe, es posible concluir que se ha logrado cumplir con todos los objetivos requeridos para esta tercera entrega de laboratorio, tanto funcionales como no funcionales.

En un principio fue todo un desafío el entender Java, en especial todos los distintos tipos de relaciones que podían tener las clases y cual era la ventaja de cada una de ellas, una vez entendida el funcionamiento de las distintas relaciones la implementación del foro "StackOverflow" se hizo mucho más fácil incluso que en la entrega anterior hecha en el paradigma lógico, esto gracias a las distintas herramientas disponibles en la librería estándar de Java.

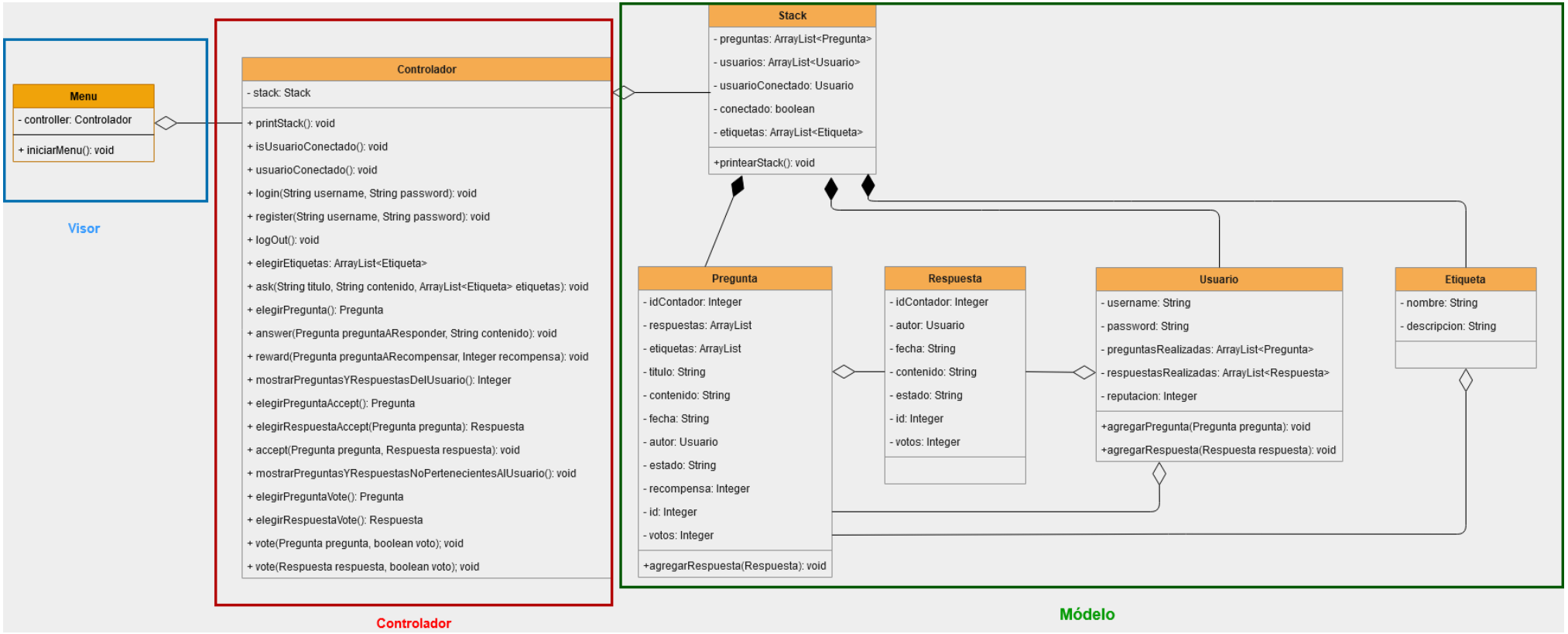
2.4 Referencias

[1] Duch A. (2006). *Esquema de Dividir y Vencer.* (Recuperado 09/01/2020). https://www.cs.upc.edu/~duch/home/duch/dyd.pdf

CAPÍTULO 3. ANEXO



Anexo 1: Diagrama de análisis.



Anexo 2: Diagrama de diseño