UNIVERSIDAD DE SANTIAGO DE CHILE

FACULTAD DE INGENIERÍA

Departamento de Ingeniería Informática

Informe De Laboratorio: Creación De Un Sistema De Chatbots

Laboratorio 3 Paradigmas de la Programación

Gonzalo Moncada

21321047-5

Fecha:

11 de diciembre 2023

Profesor:

Gonzalo Martínez

TABLA DE CONTENIDO

1. CONTENIDO DEL INFORME

1.1. INTRODUCCIÓN

El informe presentado corresponde a la segunda entrega del laboratorio de la asignatura Paradigmas de Programación, asociado al paradigma orientado a objetos aplicado al lenguaje de programación Java, todo esto será implementado en la solucion del problema sobre un Sistema de Chatbots y su interacción al usuario, esto se realizará con el paradigma presentado y con la herramienta de edición Intellij. Los tópicos para cubrir en el mismo serán: Descripción del problema, descripción del paradigma, análisis del problema, diseño de la solución, aspectos de implementación, instrucciones de uso, resultados y autoevaluación y conclusiones. En este semestre el proyecto asignado es un sistema de chatbots.

1.2. DESCRIPCIÓN DEL PROBLEMA

Un sistema de chatbots permite a un usuario generar interacciones a través de un chat en base a un tópico especifico, es utilizado comúnmente para consultas y agendamientos o reservas. Un sistema de chatbots permite crear, vincular, interactuar y modificar las opciones, flujos y chatbots para realizar asi interacciones entre estos mismos, además de esto permite obtener una síntesis de lo realizado dentro del sistema.

Este proyecto se concentrará en trabajar en la generación de un menú el cual pueda ser controlado por terminal, y permita a un usuario administrador generar todos los vínculos y los objetos que requiera, además de esto poder generar interacciones con estos sistemas mediante la terminal.

1.3. DESCRIPCIÓN DEL PARADIGMA

La base de el Paradigma orientado a objetos asociado a Java es un enfoque de programación que se basa en el uso de Objetos para la distribución de clases en el programa, debido a esto se puede hacer uso del mismo objeto en distintas definiciones y su uso permite al usuario utilizar estos objetos para la resolución de problemas. Java, es un lenguaje de programación del cual se puede utilizar este paradigma, el cual se basa en objetos o clases que contienen

atributos y métodos, estas clases pueden ser de distintos tipos tales como interfaces, clases abstractas o simplemente clases.

Estas clases pueden o no contener atributos, las cuales son un tipo de descripción de la clase en sí, definen lo que contiene o no, estas son de distintos tipos ya sean los nativos de java como int, String, boolean, etc. Lo bueno de trabajar con estos atributos es que incluso pueden ser de otro Objeto dentro del ambiente que se sitúe.

También se tienen los métodos los cuales permite generar funcionalidad en el programa, por lo mismo se pueden modificar ciertos atributos de objetos y trabajar con ellos de la manera que uno quiera, también estos métodos permiten generar ciertas relaciones entre objetos.

Estas relaciones pueden ser del tipo Fuertes (asociación, agregación o composición) las cuales se caracterizan en tener como atributo el objeto en cuestión, mientras que del tipo débil (Dependencias), estas caracterizadas por influir en un método de la clase en cuestión. Por último se tiene que estos atributos y métodos pueden tener distintos tipos de privacidad, tal como públicos, privados o protegidos, estas definen que tanto acceso existe entre cada uno de los objetos.

1.4. ANÁLISIS DEL PROBLEMA

En específico, las funciones a realizar por el programa son: option, flow, flowAddOption, chatbot, chatbotAddFlow, user, system, systemAddChatbot, systemAddUser, systemLogin, systemLogout, systemTalk, systemSynthesis y systemSimulate. Esto en síntesis seria generar métodos que generen cada una de las estructuras demostradas (option, flow, chatbot, user ,system) y sus correspondientes modificadores, además de esto permitir dentro del sistema el almacenamiento de usuarios los cuales puedan ser logeados y deslogeados, ademas se busca la manera de generar una interacción con el sistema para luego esta ser guardada en una parte del mismo, esto para lograr generar un formato de salida para la fácil comprensión de las interacciones hechas dentro del sistema, por último se busca generar una simulación de este con elecciones aleatorias, por lo mismo se tiene este UML de análisis en el anexo 1, el cual representa los principales TDA o Clases pensadas para este proyecto, teniendo en si un menú que genere las opciones al usuario, un System que almacene los datos, estos datos siendo las estructuras mencionadas anteriormente, las cuales tendrían sus propias clases, también se tiene esperado generar una clase ChatHistory la cual genere este guardado de la interacción realizada, por último se tiene el EstadoUsuarioMenu el cual mantendrá el estado del usuario luego de una interacción, cabe aclarar que el diagrama se realizó sin todos los métodos para poder presentar las relaciones que tendrán los objetos entre sí.

1.5. DISEÑO DE LA SOLUCIÓN

El enfoque de solución emplea el uso constante de modificadores dentro de un TDA principal que será el de system. este TDA posee los atributos de name (string), Usuarios (ArrayList), Usuario Logeado (String), ChatHistory (ArrayList), CurrentState(EstadoActualUsuario), ActualFlowCodeLink (int) y chatbots (ArrayList), todos estos atributos será explicados a lo largo de esta sección, por lo cual se podría explicar con el flujo que cada sistema tiene un chatbot que tiene un flow que contiene opciones, por lo cual se utilizó distintos TDA para definir cada estructura con sus propios constructores y modificadores, debido a que en el enunciado se presentaron los dominios de cada una de las estructuras presentadas se definieron estos predicados como listas dinámicas de cada atributo del dominio presentado, debido a que se presentaba que cada objeto de estas listas tendrá su única id, se generó un método para cada estructura que definiera una lista limpia en base a la id de cada objeto dentro de la lista. Luego de esto se logró realizar todos los métodos solicitados hasta el de añadir usuario, el cual explica por qué se vio la necesidad de agregar el Usuarios presentado en el inicio de esta sección, por lo cual se generó un objeto el cual tuviera su username, debido a que se debe implementar los menús y la diferencia entre el admin y usuario normal también se tiene de atributo un booleano admin, y por ultimo para que se tenga la primera interacción registrada se agregó otro booleano que fuera firstInteraction. Para verificar que cada usuario fuera único en el sistema se implementó un método el cual dada una lista de usuarios revisara si el usuario a ingresar en cuestión este dentro de la lista entregada, esto en base a su username, esto se realizó obteniendo una lista con solo los username de los usuarios y que estos revisen si el username a ingresar este contenido o no en esta lista. Por último la función systemTalkRec fue la más importante en desarrollar, por lo cual se optó por agregar como atributo de System el EstadoUsuarioMenu, este objeto tendría de atributos ActualChatbotCodeLink (int) y ActualFlowCodeLink (int) serian valores en los cuales se basaría la búsqueda de las opciones posibles o las keywords, además de esto estos se irían actualizando para que la siguiente interacción fuera en base a estos, por último en el momento de un logout estos podrán resetearse para generar interacciones con otro usuario, en lo demás se usó de base este predicado para generar un formato de string y que este se imprima, y generar una simulación con valores aleatorios que fuera desde un numero sacando el resto de este y colocándolo como elección del usuario, debido a que la creación de cada objeto tendría que asignarse a otro por la agregación que tienen, se optó por generar la clase Data, la cual guarda cada objeto creado para que un administrador lograra elegir en el menú el objeto a ingresar. además de esto debido a que las acciones se hacen mediante un menú, se optó en generar esta clase que realizara todas las acciones descritas, además de que debido a que están los constructores del sistema,

se optó por tener una lista de sistemas dentro del menú para que se lograra cambiar el sistema actual si es que el usuario lo desea, además de que debido a que se debe tener un sistema base con datos, se generó la clase CargaDeDatos la cual permite generar un sistema base y este dejarlo como el sistema inicial, el UML con las relaciones está en el anexo .

1.6. ASPECTOS DE IMPLEMENTACIÓN

El compilador utilizado en este proyecto fue de Java junto con el JDK en su versión 11. El código se estructura por TDA o Clases, para facilitar la edición a la hora de agregar contenido a alguno de estos, utilizando llamados para importar y exportar información de estos a otros archivos y asi hacer utilización de estos. Para este laboratorio no se utilizo bibliotecas externas, solo funciones nativas de java.

1.7. INSTRUCCIONES DE USO

Debido a que es este proyecto es trabajado por un menú en la terminal, la generación de cada estructura presentada será realizada por el medio de pasos hasta seleccionar todo, en el anexo 3 se presenta cada estructura creada, cabe aclarar que, como posible error si se ingresa un objeto que ya está repetido (por ejemplo en un modificador de Flow agregar una opción que contenga la misma id de alguna ya presente), este no será agregado, aun asi seguirá la ejecución del programa por su ingreso en el registro, por último se espera que en el uso de la clase data el usuario pueda ingresar opciones independientes de que si están asignadas o no.

1.8. RESULTADOS Y AUTOEVALUACIÓN

En el amplio aspecto, el proyecto fue exitoso, pues las funciones requeridas se pudieron realizar en su totalidad. Debido a que este es un lenguaje orientado en Objetos, se pudo observar como la manipulación de ciertos Objetos podía cambiar dependiendo del tipo de atributo que se colocaba, un ejemplo fue el cambio de list a ArrayList, debido a que una era estática mientras que la otra era dinámica, esto permitiendo el manejo correcto de las listas dentro del laboratorio. además de esto se tiene todos los menús completos, teniendo todas las implementación para generar cualquier objeto definido anteriormente.
La autoevaluación radica entre valores de 0 (no implementado) hasta 1 (implementado y sin errores) aumentando en una escala de 0.25 y se puede observar en el anexo 3. Se les asigna a

todas las funciones 1 dado que al momento de realizar diversas pruebas estas trabajan correctamente y sin errores.

1.9. CONCLUSIONES

La definición de clases y su manejo es algo completamente distinto a como se trabajaron los anteriores laboratorios, debido a que además de cómo es el paradigma, java tiene distintas funciones nativas que permiten un mejor manejo de los datos, permitiendo obtener listas limpias de una manera en la cual se aprovecha mejor el lenguaje utilizado.

Las primeras interacciones con el lenguaje fueron lo más complejo, debido a que se debía colocar ciertas definiciones para desde ahí empezar a realizar un método, pero debido a que se empezó a hacer repetitivo se logró avanzar de la mejor manera. Se podría decir que la mayor limitación fue la implementación de interfaces, debido a que frente a la generación de una nueva clase se presenta este impedimento a tener que implementarla sobre una interfaz, siendo que algunas no necesitaban esta definición.

Finalmente, se puede decir que este paradigma pudo ser abordado de la mejor manera, debido a que se logró generar un proceso bastante lineal con ciertas dificultades en funciones específicas pero que no terminaron de detener el progreso del propio proyecto.

1.10. BIBLIOGRAFÍA

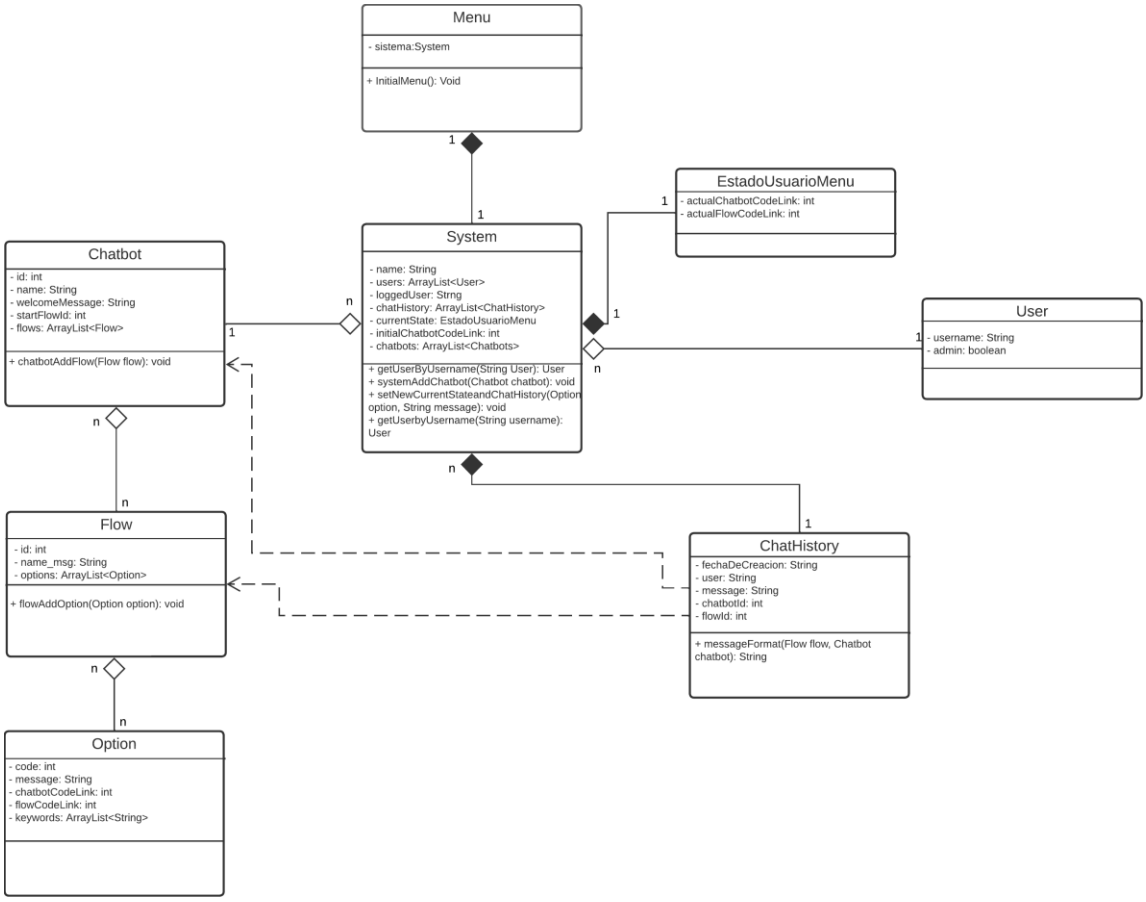
Bailón, J. y Baltazar, J. M. (2021). Características de la POO. En Algoritmos y codificación. Portal Académico del CCH, UNAM. https://portalacademico.cch.unam.mx/cibernetica1/algoritmos-y-codificacion/caracteristicas-POO

IBM documentation. (2021, November 22). https://www.ibm.com/docs/es/spss-modeler/saas?topic=language-object-oriented-programming

Qué es la programacion orientada a objetos. (n.d.). DesarrolloWeb.com. https://desarrolloweb.com/articulos/499.php

1.11. ANEXO

- Anexo 1: UML de analisis.



- Anexo 2: UML final.

-Anexo 3: Ejemplos de cada estructura.

```
### Sistema de Chatbots - Usuario Administrador ###
Bienvenido Ghox usted es administrador.
1. Crear un Chatbot
2. Modificar un Chatbot
3. Interactuar con el Sistema
4. Obtener una Síntesis del Historial de un Usuario
5. Visualizar todos los chatbots con sus flujos y opciones creadas
6. Visualizar todos los chatbots con sus flujos y opciones creadas dentro del sistema Actual
7. Crear un Sistema
8. Añadir Chatbot a un Sistema
9. Cambiar Sistema actual
10. Ejecutar una simulación del sistema de chatbot actual
11. Salir
INTRODUZCA SU OPCION: 1
### Sistema de Chatbots - Usuario Administrador ###
1. Crear Opciones
2. Crear un Flow
3. Modificar un Flow
4. Definir un Chatbot
5. Volver
INTRODUZCA SU OPCION: 1
### Sistema de Chatbots - Usuario Administrador ###
INTRODUZCA LA ID ASIGNADA DE SU OPCION: 1
INTRODUZCA EL MENSAJE DE SU OPCION: ejemplo
INTRODUZCA EL LINK AL CHATBOT DE SU OPCION: 1
INTRODUZCA EL LINK AL FLOW DE SU OPCION: 1
INGRESA LAS PALABRAS CLAVE DE SU OPCION, CUANDO SEA LA ULTIMA INGRESA UN -1:
palabra1
palabra2
-1
La opcion se ingreso al registro correctamente
```

```
### Sistema de Chatbots - Usuario Administrador ###
1. Crear Opciones
2. Crear un Flow
3. Modificar un Flow
4. Definir un Chatbot
5. Volver
INTRODUZCA SU OPCION: 2
### Sistema de Chatbots - Usuario Administrador ###
INTRODUZCA LA ID ASIGNADA DE SU FLOW: 1
INTRODUZCA EL NOMBRE Y MENSAJE DE SU FLOW(debe ser separado con un \n ej: name \n message): ejemplo\nmensaje
INTRODUZCA SECUENCIALMENTE LAS OPCIONES A INCLUIR, TERMINAR CON UN -1:
1. Option{1, '1) Viajar', 1, 1, [viajar, turistear, conocer]}
2. Option{2, '2) Estudiar', 2, 1, [estudiar, aprender, perfeccionarme]}
3. Option{1, '1) New York, USA', 1, 2, [USA, Estados Unidos, New York]}
4. Option{2, '2) París, Francia', 1, 1, [Paris, Eiffel]}
5. Option{3, '3) Torres del Paine, Chile', 1, 1, [Chile, Torres, Paine, Torres Paine, Torres del Paine]}
6. Option{4, '4) Volver', 0, 1, [Regresar, Salir, Volver]}
7. Option{1, '1) Central Park', 1, 2, [Central, Park, Central Park]}
8. Option{2, '2) París, Francia', 1, 1, [Museo, Antiguedades]}
9. Option{3, '3) Ningún otro atractivo', 1, 3, [Ninguno]}
10. Option{4, '4) Cambiar destino', 1, 1, [Cambiar, Volver, Salir]}
11. Option{1, '1) Solo', 1, 3, [Solo]}
12. Option{2, '2) En pareja', 1, 3, [Pareja]}
13. Option{3, '3) En familia', 1, 3, [Familia]}
14. Option{4, '4) Agregar más atractivos', 1, 2, [Volver, Atractivos]}
15. Option{5, '5) En realidad quiero otro destino', 1, 1, [Cambiar destino]}
16. Option{1, '1) Carrera Técnica', 2, 1, [Técnica]}
17. Option{2, '2) Postgrado', 2, 1, [Doctorado, Magister, Postgrado]}
18. Option{3, '3) Volver', 0, 1, [Volver, Salir, Regresar]}
19. Option{1, 'ejemplo', 1, 1, [palabra1, palabra2]}

Ingrese Opciones Aqui.
19
2
-1
El flujo se ingreso al registro correctamente
```

```
### Sistema de Chatbots - Usuario Administrador ###
1. Crear Opciones
2. Crear un Flow
3. Modificar un Flow
4. Definir un Chatbot
5. Volver
INTRODUZCA SU OPCION: 4
### Sistema de Chatbots - Usuario Administrador ###
INTRODUZCA LA ID ASIGNADA DE SU CHATBOT: 1
INTRODUZCA EL NOMBRE DE SU CHATBOT: ejemplo
INTRODUZCA EL MENSAJE DE BIENVENIDA DE SU CHATBOT: mensaje
INTRODUZCA EL LINK DE INICIO DE FLOW DE SU CHATBOT: 1
INTRODUZCA SECUENCIALMENTE LOS FLUJOS A INCLUIR, TERMINAR CON UN -1:
1. Flow{1, 'Flujo Principal Chatbot 1
Bienvenido
¿Qué te gustaría hacer?', [Option{1, '1) Viajar', 1, 1, [viajar, turistear, conocer]}, Option{2, '2) Estudiar', 2, 1, [estudiar, aprender, perfeccionarme]}
2. Flow{1, 'Flujo 1 Chatbot1
¿Dónde te Gustaria ir?', [Option{1, '1) New York, USA', 1, 2, [USA, Estados Unidos, New York]}, Option{2, '2) París, Francia', 1, 1, [Paris, Eiffel]}, Opti
3. Flow{2, 'Flujo 2 Chatbot1
¿Qué atractivos te gustaria visitar?', [Option{1, '1) Central Park', 1, 2, [Central, Park, Central Park]}, Option{2, '2) París, Francia', 1, 1, [Museo, Ant
4. Flow{3, 'Flujo 3 Chatbot1
¿Vas solo o acompañado?', [Option{1, '1) Solo', 1, 3, [Solo]}, Option{2, '2) En pareja', 1, 3, [Pareja]}, Option{3, '3) En familia', 1, 3, [Familia]}, Opti
5. Flow{1, 'Flujo 1 Chatbot2
¿Qué te gustaría estudiar?', [Option{1, '1) Carrera Técnica', 2, 1, [Técnica]}, Option{2, '2) Postgrado', 2, 1, [Doctorado, Magister, Postgrado]}, Option{3

Ingrese Flujos Aqui.
1
3
-1
DESEA INGRESAR ESTE CHATBOT AL SISTEMA?:
1. Si
2. No
INTRODUZCA SU OPCION: 2
El chatbot se ingreso correctamente
```

```
### Sistema de Chatbots - Usuario Administrador ###
Bienvenido Ghox usted es administrador.
1. Crear un Chatbot
2. Modificar un Chatbot
3. Interactuar con el Sistema
4. Obtener una Síntesis del Historial de un Usuario
5. Visualizar todos los chatbots con sus flujos y opciones creadas
6. Visualizar todos los chatbots con sus flujos y opciones creadas dentro del sistema Actual
7. Crear un Sistema
8. Añadir Chatbot a un Sistema
9. Cambiar Sistema actual
10. Ejecutar una simulación del sistema de chatbot actual
11. Salir
INTRODUZCA SU OPCION: 7
### Sistema de Chatbots - Usuario Administrador ###
INTRODUZCA EL NOMBRE DEL SISTEMA: ejemplo
INTRODUZCA EL ID DEL CHATBOT INICIAL DEL SISTEMA: 1
INTRODUZCA SECUENCIALMENTE LOS CHATBOTS A INCLUIR, TERMINAR CON UN -1:
1. Chatbot{0, 'Inicial', Bienvenido
¿Qué te gustaría hacer?', 1, [Flow{1, 'Flujo Principal Chatbot 1
Bienvenido
¿Qué te gustaría hacer?', [Option{1, '1) Viajar', 1, 1, [viajar, turistear, conocer]}, Option{2, '2) Estudiar', 2, 1, [estudiar, aprender, perfeccionarme
2. Chatbot{1, 'Agencia Viajes', Bienvenido
¿Dónde quieres viajar?', 1, [Flow{1, 'Flujo 1 Chatbot1
¿Dónde te Gustaría ir?', [Option{1, '1) New York, USA', 1, 2, [USA, Estados Unidos, New York]}, Option{2, '2) París, Francia', 1, 1, [Paris, Eiffel]}, Op
¿Qué atractivos te gustaría visitar?', [Option{1, '1) Central Park', 1, 2, [Central, Park, Central Park]}, Option{2, '2) París, Francia', 1, 1, [Museo, A
¿Vas solo o acompañado?', [Option{1, '1) Solo', 1, 3, [Solo]}, Option{2, '2) En pareja', 1, 3, [Pareja]}, Option{3, '3) En familia', 1, 3, [Familia]}, Op
3. Chatbot{2, 'Orientador Académico', Bienvenido
¿Qué te gustaría estudiar?', 1, [Flow{1, 'Flujo 1 Chatbot2
¿Qué te gustaría estudiar?', [Option{1, '1) Carrera Técnica', 2, 1, [Técnica]}, Option{2, '2) Postgrado', 2, 1, [Doctorado, Magister, Postgrado]}, Option

Ingrese Chatbots Aqui.
1
2
-1
El Sistema se ingreso correctamente
```

```
### Sistema de Chatbots - Inicio ###
1. Login de Usuario
2. Registro de Usuario
3. Salir
INTRODUZCA SU OPCION: 2
### Sistema de Chatbots - Registro ###
1. Registrar usuario normal
2. Registrar usuario administrador
INTRODUZCA SU OPCION: 1
### Sistema de Chatbots - Registro Usuario Normal ###
INTRODUZCA NOMBRE DEL USUARIO NORMAL: user
El usuario se ingreso al Sistema correctamente
### Sistema de Chatbots - Inicio ###
1. Login de Usuario
2. Registro de Usuario
3. Salir
INTRODUZCA SU OPCION:
```

- Anexo 4: Autoevaluación.

Autoevaluación	Puntaje
option	1
flow	1
flowAddOption	1
chatbot	1
chatbotAddFlow	1
user	1
system	1
systemAddChatbot	1
systemAddUser	1
systemLogin	1
systemLogout	1
systemTalk	1
systemSynthesis	1
systemSimulate	1