PONTIFICIA UNIVERSIDAD CATÓLICA DE VALPARAÍSO
FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA INFORMÁTICA

Informe final Proyecto: Gestión de Trabajadores Públicos por Carteras Ministeriales

Integrantes:

Bastián Jorquera García

Matías Altamirano Castro

Catalina Marín Hernández

ÍNDICE

ÍNDICE 1

Resumen 3

Lista de Figuras 4

1. Introducción 5

2. Explicación general del programa 6

3. Clases con las que se trabajará y su explicación 6

4. Definición y organización de las clases y ventanas gráficas 7

4.1 Clase Empleado: 7

4.2 Clase Sucursal 7

4.3 Clase Ministerio 8

4.4 Clase Gobierno 8

4.5 Clase menú 8

4.6 Main 9

4.7 Vista menú empleado 10

4.8 Vista agregar empleado 10

4.9 Vista Buscar empleado 10

4.10 Vista Edita empleado 10

5. Entrada y carga de los datos 10

6. Diseño gráfico UML 11

7. Uso del HashMap 12

8. Navegación dentro del menú de ventanas 13

Recapitulación 13

ANEXOS 15

Supuestos 15

Limitaciones 15

Funcionalidades extra para desarrollar 15

Enlace de GitHub 15

Resumen

En este informe de proyecto vamos a mostrar el funcionamiento de un sistema de gestión de datos capaz de organizar y ordenar a los funcionarios públicos presentes en múltiples cartillas ministeriales dentro de los ministerios del gobierno, mediante una interfaz gráfica interactiva con el usuario. Inicialmente se usarán distintas estructuras informáticas del lenguaje de programación Java para hacer las respectivas gestiones de organización, se considerarán las funciones más básicas como son; eliminar, agregar y mover en los distintos objetos de trabajo, usaremos diversos métodos según las características que se necesiten para los distintos objetivos y sus situaciones correspondientes. Analizaremos qué datos se van a utilizar, qué clases utilizaremos y qué atributos y métodos son los que nos parecieron más acordes a la problemática.

Lista de Figuras

Figura 1 Diagrama de las clases. 6

Figura 2 Esquema del diseño gráfico de clases. 11

Figura 3 Esquema lógico del mapa y sucursales. 12

1. Introducción

La principal problemática del tema elegido es almacenar información relevante de un número indeterminado de empleados del gobierno, que pertenecen a sus propias sucursales, y estas últimas a su vez a sus ministerios correspondientes, la dificultad radica en lo cambiante que es la información de cada área mencionada, lo variable que puede ser su número de existencias y la correcta asignación de los datos.

Este proyecto pretende abordar las problemáticas mencionadas desarrollando una base de datos con un sistema de almacenamiento de información y gestión de esta, automatizado y ordenando la gestión de los trabajadores públicos en diversas reparticiones del gobierno. El sistema nos permitirá crear, eliminar y reubicar a los trabajadores, además de eliminar y agregar sucursales y ministerios, organizado y estructurado bajo una clase principal gobierno. En un comienzo se utilizó un sistema de menús de alternativas numéricas por consola, pero actualmente el sistema evolucionó a implementar una navegación usando la interfaz gráfica de Apache NetBeans, con ventanas emergentes simples para navegar dentro de las distintas pestañas con opciones y funcionalidades de la base de datos.

Este informe estará enfocado en detallar el funcionamiento del código para este proyecto, analizar su estructura conceptual y lógica, y por último detallar las distintas estructuras informáticas usadas y por último como se utiliza la interfaz de navegación. Todo lo anterior desarrollado en el lenguaje de programación Java y utilizando la herramienta grafica Apache NetBeans.

2. Explicación general del programa

El programa desarrollado es una base de datos ordenada en cuatro clases jerárquicas que nos permitirán como objetivo principal ordenar, organizar, agregar o eliminar a los funcionarios dentro de distintas sucursales, ministerios y el gobierno, cada uno de estos objetos está representado por una clase definida con atributos y métodos específicos. La forma de trabajar con esta base de datos es a través ventas desplegables interactivas en tiempo real, donde el usuario debe indicar qué acción desea realizar y siguiendo las instrucciones con los ejemplos indicados.

3. Clases con las que se trabajará y su explicación

Se tendrá la clase principal de la que se ramifican el resto de las clases, siendo esta el Gobierno, contará con un mapa de los Ministerios, también se encargará de la organización, la creación y la eliminación de estos y sus métodos asociados. En cuanto a los ministerios lógicamente cumplirá labores muy parecidas, pero asociará los departamentos que lo componen en una lista, y tendrá los métodos asociados a la organización de esta. En cuanto a los departamentos, contarán también con una lista, de los trabajadores pertenecientes a dicho departamento, sus funciones de organización y si existe un superávit o déficit de estos. Por último, tenemos a la clase funcionario, esta representa a un trabajador único con sus respectivas características, como son el nombre, id, cargo, etc.

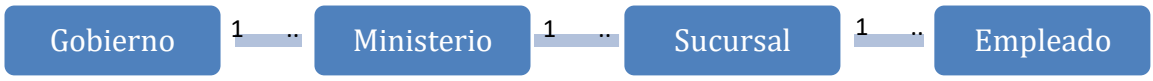

Figura 1 Diagrama de las clases.

Las clases usadas están ordenadas de forma jerárquica (Figura 1.1, mayor jerarquía a la izquierda, menor jerarquía a la derecha) y nos parecieron la forma más ordenada y lógica para representar y trabajar en la organización de los datos del presente proyecto.

4. Definición y organización de las clases y ventanas gráficas

Luego, se mostrará una lista de las clases usadas para la organización del proyecto, un apartado en cada una con sus atributos con un paréntesis que indica el tipo de dato que es, seguido de un segundo apartado que muestra sus métodos.

4.1 Clase Empleado:

Como unidad fundamental de la organización representando a un funcionario único, esta clase contará con los siguientes atributos:

- El nombre del funcionario (String).
- Un identificador único como es el Rut (String).
- Su fecha de nacimiento (LocalDate).
- Fecha de inicio de contrato (LocalDate).
- El cargo que posee (String).
- Su salario (double).
- El departamento en el cual trabaja (String).

En cuanto a los métodos que tendrá:

- Getters y Setters de los atributos.
- Métodos auxiliares y principales que se necesiten.

4.2 Clase Sucursal

Representa una sucursal dentro de un ministerio, y contiene a los funcionarios de cada ministerio, posee los siguientes atributos:

- El nombre de la sucursal (String).
- Un identificador único (String).
- Comuna (String).
- Ciudad (String).
- Región (String).
- A qué ministerio pertenece (Ministerio).

- Una lista con los funcionarios de la sucursal actual (List).

En cuanto a los métodos de la clase:

- Métodos getter y setter.
- Métodos auxiliares y principales que se necesiten.

4.3 Clase Ministerio

Esta clase representa al ministerio del gobierno central y posee una lista de departamentos que lo componen, y posee los siguientes atributos:

- Nombre del ministerio (String).
- Un identificador único (String).
- El año de su creación (LocalDate).
- Una lista con las sucursales que lo componen (ArrayList).

En los métodos que se usarán, tenemos:

- Métodos getter y setter.
- Métodos auxiliares y principales que se necesiten.

4.4 Clase Gobierno

Esta clase tiene a los diversos ministerios y sus funciones para su organización, los atributos que posee son:

- Nombre del país (String).
- Nombre del presidente (String).
- Mapa con todos los Ministerios (HashMap).

En los métodos que se usarán, tenemos:

- Métodos getter y setter.
- Métodos auxiliares y principales que se necesiten.

4.5 Clase menú

Esta clase gestiona la interacción del usuario en las ventanas con el programa y los métodos que requieren estos datos ingresados para los principales métodos encargados de modificar, buscar, eliminar, listar empleados y más funciones auxiliares, además en menú están las principales gestiones para los métodos en las ventanas interactivas. Además, posee los métodos

utilizados para inicializar la base de datos. Algunos métodos que poseen son:

- Scanner (static)
- Objeto clase Sucursal

En los métodos que se usarán:

- Inicializar Sistema
- Mostrar menú
- Obtener opción (mediante scanner)
- Procesar opción
- Ingresar datos de empleado
- Buscar empleado (por Rut o nombre y departamento)
- Procesar opción dos (procesar las opciones de4 búsqueda de empleados
- Modificar datos empleado
- Eliminar empleado
- Obtener dato
- Obtener fecha
- Obtener salario

4.6 Main

El main en nuestro código es realmente simple, y solo se encarga a llamar a los métodos que inicializan el programa, las mayores responsabilidades las lleva la clase menú.

4.7 Vista menú empleado

Esta ventana es la vista principal del menú, es la primera que aparece y nos permitirá navegar dentro de las opciones disponibles, posee 3 botones; Agregar Empleado, Buscar Empleado y salir

4.8 Vista agregar empleado

Esta ventana nos permitirá desplegar los campos para ingresar los datos del empleado y agregarlo a la base de datos.

4.9 Vista Buscar empleado

Esta ventana nos permitirá ver la información referente a todos los empleados, y nos permitirá filtrar por empleados específicos según su rut. También posee el acceso para abrir la vista editar y el otro para eliminar a un empleado.

4.10 Vista Edita empleado

Desde esta ventana podremos modificar los datos de un empleado.

5. Entrada y carga de los datos

La creación de los de los distintos objetos como se les atribuye a los distintos constructores para cada clase, estos datos vienen por parte de métodos en su salida y desde la entrada de datos de los usuarios desde las ventanas emergentes según lo que se pida ingresar en cada caso.

Respecto a la carga de los datos, en este código si se almacenan los datos en un archivo no volátil, en el cual podremos trabajar y modificar desde los distintos menús.

6. Diseño gráfico UML

Las clases tienen un ordenamiento lógico y jerárquico, en la parte superior de la jerarquía tenemos a la clase Gobierno, de la cual se despliega el resto, (como se puede observar en la Figura 1.2 y Figura 1.1), luego le seguiría la clase Ministerio, Sucursal y por último la clase Empleado. La clase Gobierno posee una estructura de ordenamiento lógico de mapa, de esta se ramifica la lista de elementos de tipo clase Ministerios, los que a su vez tienen la lista de elementos de tipo Sucursales, y por último esta posee una lista de elementos de tipo clase Empleados con los empleados de la sucursal actual. En el grafico UML que se presenta podemos ver los distintos atributos pertenecientes a cada clase y su relación con el resto de las clases del código.

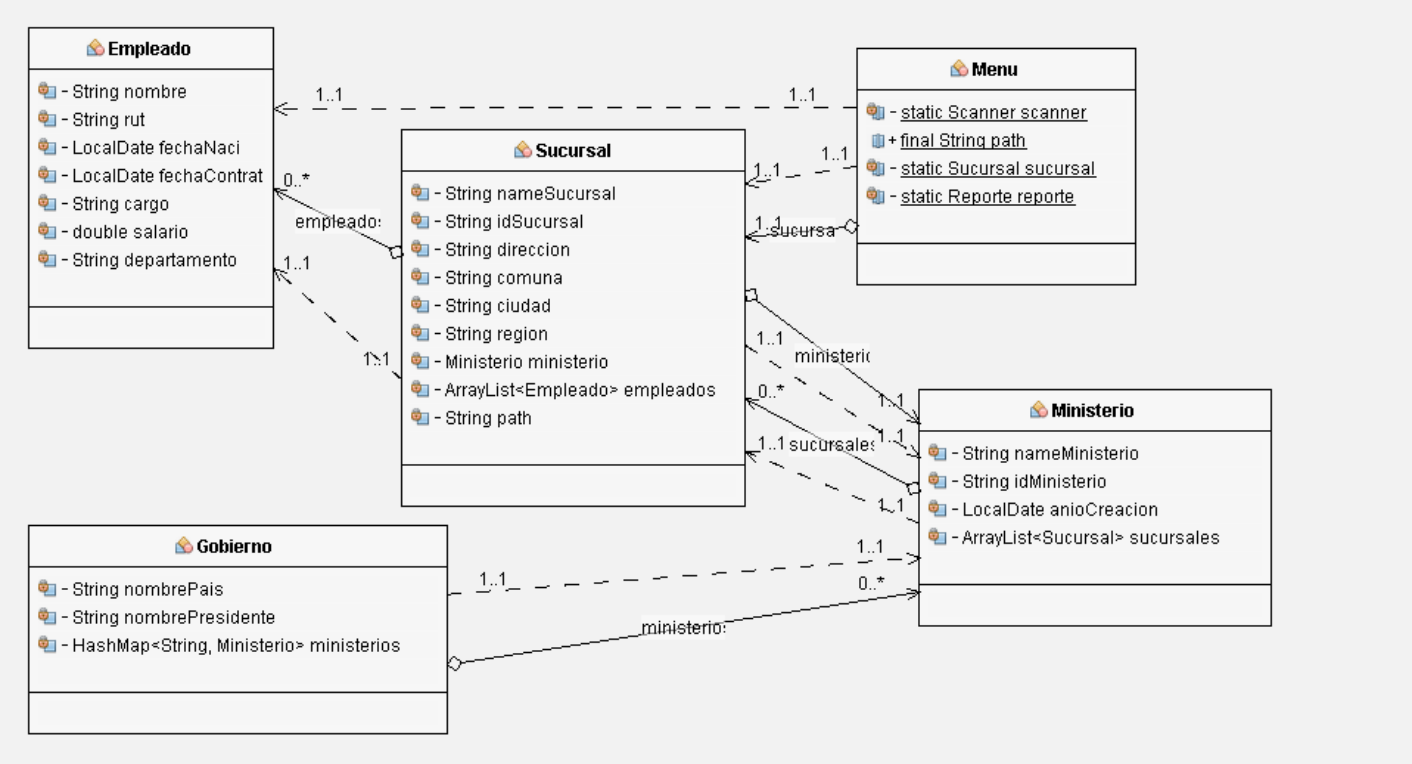

Figura 2 Esquema del diseño gráfico de clases.

7. Uso del HashMap

Anteriormente ya se vieron las relaciones lógicas entre las distintas clases, ahora nos enfocaremos en la clase Gobierno, específicamente en el mapa definido en esta. Este hashmap nos guarda referencias a los distintos ministerios ya definidos y nos permite buscarlos, eliminarlos o agregar nuevos de manera independiente a su cantidad gracias a que no hay que recorrer ningún tipo de arreglo o lista para buscarlos, además garantiza que no estén repetidos y manteniendo a los identificadores únicos como debe ser, es a través de este identificador único de cada ministerio que es como accederemos a él a través del mapa, el gobierno posee el mapa con múltiples clases ministerio, y este último solo posee un gobierno al cual pertenece.

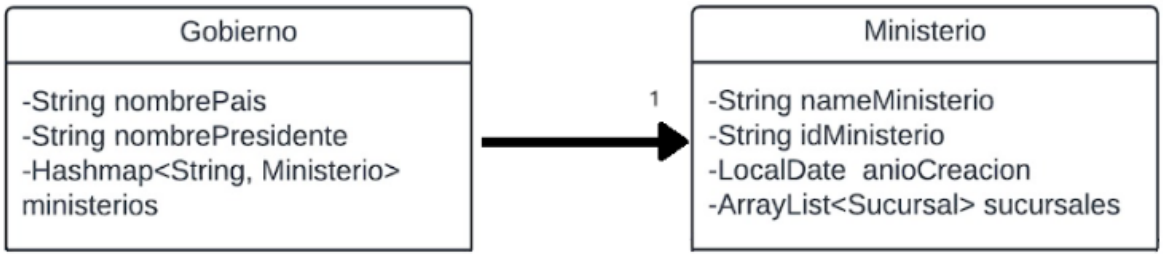

Figura 3 Esquema lógico del mapa y sucursales.

8. Navegación dentro del menú de ventanas

Al hacer la compilación del código lo primero que se notará es el despliegue de una venta con distintas opciones, a continuación, se explicará cómo es la navegación dentro de las opciones de control en la ventana emergente.

Ventana menú principal: Nada más entrar y ejecutar el código, se desplegará una ventana con él un texto que dice menú de opciones, contará con tres botones, de arriba hacia abajo tenemos; Agregar empleado, Buscar empleado y salir. El primer botón nos desplegará la ventana Agregar empleado, en el cual tendremos que agregar los datos para agregar un empleado a la base de datos. Respecto al botón Buscar empleado, este nos servirá para desplegar una ventana en la cual podremos ver todos los empleados, modificar sus datos o eliminar algún empleado. Y respecto a el botón salir, este como su nombre indica saldrá del menú terminando la ejecución del programa.

Ventana agregar empleado: Luego de seleccionar esta ventana desde el menú principal tendremos que llenar los campos indicados para agregar un empleado: Nombre de empleado, Rut, Fecha nacimiento, a que ministerio pertenece seleccionando con el clic de una lista de ministerios, seleccionar su cargo también de una lista de opciones, y por ultimo llenar su salario, luego para agregar solo hay que presionar el botón agregar, si el empleado se agregó de forma correcta aparecerá un mensaje indicándolo, si no aparecerá un mensaje de que el empleado ya está en la base, así que no se puede agregar o aparecerá error si por alguna razón no se pudo concretar el proceso, en este caso intentar nuevamente. El botón volver, permitirá volver al menú principal.

Ventana buscar empleado: Luego de seleccionar esta ventana en el menú principal se podrá observar una lista con todos los empleados, si desea buscar una especifico debe escribir su rut con formato adecuado (ej: 12.345.678.9), y presionar el botón buscar, luego con este empleado puede usted modificar sus datos desde el botón “editar” lo que abrirá la ventana editar., o podrá eliminar el empleado seleccionado el botón eliminar. Puede presionar el botón volver para llegar al menú principal o cerrar la ejecución del programa con el botón cerrar.

Ventana modificar empleado: Luego de seleccionar esta ventana desde la ventana buscar empleado, este apartado nos permitirá modificar los datos de un empleado ya existente llenando los campos en pantalla de forma similar a la ventana de agregar empleado, luego de estar conforme con los datos ingresados seleccionar el botón guardar para guardar los cambios.

Recapitulación

Utilizando el esquema de clases y su jerarquía se puede observar fácilmente en qué consiste la organización del proyecto. Para empezar, se utilizó la clase gobierno que cuenta con un mapa con referencia a los ministerios, cada ministerio tiene una referencia a un arreglo con referencia a las sucursales, que a su vez tienen una lista con referencia a los empleados de estas. Se pudo revisar su estructura lógica y los métodos utilizados por clase y sus características, y cómo permiten ordenar las características relevantes para cada clase. En cuanto a los métodos que se utilizaron en este avance fueron los básicos que se pueden esperar en una estructura de organización; como son insertar, eliminar y listar. Asimismo, a modo de organización que permite al usuario gestionar las opciones nombradas anteriormente se despliega en consola un menú con el cual se trabaja con los empleados (segundo nivel de anidación de colecciones) y sus respectivos datos (nombre, rut, fecha de contrato, entre otros).

Lo importante fue crear cimientos lo más sólidos posibles en este avance de proyecto, con el objetivo de realizar una base de datos robusta en el futuro, con nuevas funcionalidades y afinar las ya existentes para hacerlas más precisas en un futuro.

ANEXOS

Supuestos

1. Se asume que los datos ingresados por usuario en todas las instancias son datos válidos y siguen el formato correcto.

Limitaciones

1. No se valida que el RUT tenga el formato adecuado.
2. No existe un control de acceso de distintos usuarios o permisos de estos.

Funcionalidades extra para desarrollar

1. Implementar una base de datos que nos permita guardar información de manera no volátil.
2. Validación de los datos de entrada de forma adecuada y segura.
3. Agregar funciones avanzadas de las ya presentes e implementar nuevas.
4. Agregar un sistema de reportes sobre las distintas clases para mejorar el entendimiento e interpretación de los datos.
5. Carga de datos desde bases de datos externas.

Enlace de GitHub

https://github.com/BastianJorquera/AvanceProgramacionavanzada.git