-Diagrama de clases UML para un sistema de gestión de empleados-

Antecedentes

El presente informe describe el diagrama de clases UML para un sistema de gestión de empleados. El diagrama cubre las funcionalidades del sistema, que son las siguientes:

- Registro de empleados con sus datos personales y laborales.
- Actualización de información de empleados (datos personales y laborales).
- Consulta de datos de empleado(todos).
- Los empleados tendrán una categoría profesional.
- Consulta de categorías profesionales existentes.
- Registros de departamentos a los que puede pertenecer el empleado.

Para la creación del Diagrama de Clases UML he utilizado la aplicación web https://app.diagrams.net lo he usado bastante para el módulo de Base de Datos y es muy completo, fácil e intuitivo.

El primer paso es identificar las clases; en el enunciado de la actividad se indican empleado, categoría profesional y departamento, he añadido una clase extra “persona” que será padre de empleado y posibilita una futura escalabilidad del programa en caso de tener que gestionar más cargos distintos a empleados, como pudieran ser socios o colaboradores externos. La clase persona es padre de la clase empleado.

Clases

El diagrama de clases define las siguientes clases:

- Persona: Representa a los tipos de personas que componen la empresa.
- Empleado: Representa a un empleado de la empresa.
- CategoríaProfesional: Representa una categoría profesional de la empresa.
- Departamento: Representa un departamento de la empresa.

Relaciones

Las clases del diagrama de clases están relacionadas de la siguiente manera:

Asociación:

- Un empleado pertenece a una categoría profesional.
- Clases relacionadas: **Empleado, CategoríaProfesional**
- Tipo de asociación: 1..*
- Cardinalidad: Un empleado puede tener una sola categoría profesional, pero una categoría profesional puede tener varios empleados.
- Un empleado puede pertenecer a varios departamentos.
- Clases relacionadas: **Empleado, Departamento**
- Tipo de asociación: *..*
- Cardinalidad: Un empleado puede pertenecer a varios departamentos, pero un departamento puede tener varios empleados.

Herencia:

La clase Empleado hereda de la clase Persona

- Tipo de asociación: 1..*

- Cardinalidad: una persona puede tener varios empleados, pero un empleado es solo una persona.

Atributos y métodos heredados:

- DNI
- Nombre
- Apellidos
- Teléfono
- Correo electrónico
- Edad

Explicación de las relaciones:

La relación de asociación entre las clases Empleado y CategoríaProfesional indica que un empleado puede pertenecer a una o más categorías profesionales. La cardinalidad 1..* indica que un empleado puede tener una sola categoría profesional, pero una categoría profesional puede tener varios empleados.

La relación de asociación entre las clases Empleado y Departamento indica que un empleado puede pertenecer a uno o más departamentos. La cardinalidad *..* indica que un empleado puede pertenecer a varios departamentos, pero un departamento puede tener varios empleados.

La relación de herencia entre las clases Empleado y Persona indica que la clase Empleado hereda los atributos y métodos de la clase Persona. Esto permite reutilizar código y reducir la complejidad del sistema.

No encontramos relación de agregación o composición en las relaciones entre las clases.

Funcionalidad de los métodos:

Los métodos definidos en las clases del diagrama de clases tienen la siguiente funcionalidad:

- Persona.updateInfoPers(): Actualiza información personal de la persona
- Persona.updateInfoLab(): Actualiza información laboral de la persona
- Empleado.registrarEmpleado(): Registra un nuevo empleado en el sistema.
- Empleado.actualizarEmpleado(): Actualiza la información de un empleado existente.
- Empleado.consultarEmpleado(): Consulta la información de un empleado.
- CategoríaProfesional.consultarCategoríaProfesional(): Consulta una categoría profesional existente.
- Departamento.registrarDepartamento(): Registra un nuevo departamento en el sistema.
- Departamento.consultarDepartamento(): Consulta un departamento existente.

El diagrama de clases propuesto puede cubrir todas las funcionalidades del sistema especificado. Las relaciones de asociación permiten que las clases interactúen entre sí. La herencia permite reutilizar código y reducir la complejidad del sistema.