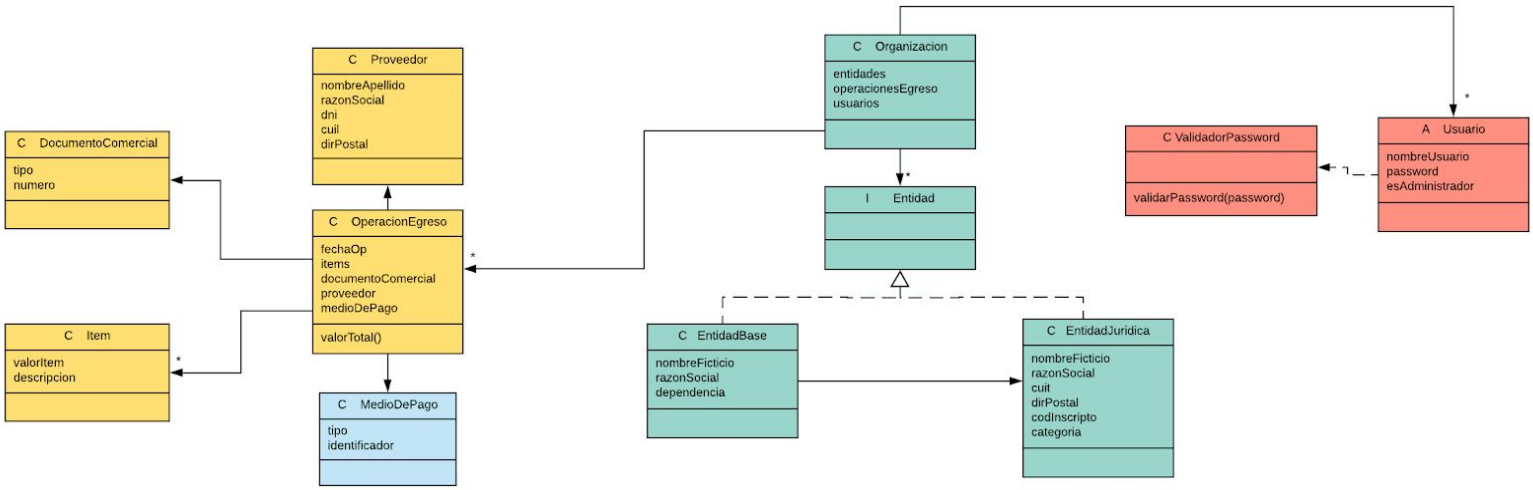
GeSoc DDS

Entrega 1 - Jueves Mañana - Grupo 4



Decisiones tomadas

- En la parte de medio de pago, nos inclinamos por usar composición antes que utilizar herencia, ya que nos pareció lo más óptimo, oportuno, flexible y extensible en relación al actual desarrollo del TP. Flexible por el hecho de que si en un futuro vemos atributos en simultáneo entre los medios de pago, podemos cambiar el empleo de un interfaz por el de una clase abstracta sin mayores inconvenientes. A su vez es extensible porque resulta fácil agregar nuevos tipos de pago, éstos solamente tienen que implementar la interfaz MedioDePago.

- Se podría haber utilizado un dictionary/map/hash en la operación de egreso para los ítems, pero se decide crear la abstracción Item porque nos pareció la opción más clara y que más se ajusta al dominio del negocio.

- Las entidades jurídicas y las base deben poder tratarse de manera indistinta, por lo que decidimos modelarlas empleando una interfaz Entidad. Ésta es implementada por las clases de ambos tipos de entidades. A su vez, EntidadJuridica es una clase abstracta ya que concretamente puede tratarse de una Empresa o de una OSC.

- Como las entidades base son una convención informal, decidimos que sea la entidad base la que conoce la entidad jurídica a la que pertenece, y no al revés. Esto a su vez se asegura que siempre se cumpla el requerimiento de que una entidad base pertenezca a una sola entidad jurídica sin tener que agregar código que verifique esto y evitando así agregar complejidad innecesaria.

- Decidimos usar composición para las distintas categorías de las empresas para poder admitir que una empresa cambie su condición si se recategoriza ante la AFIP; de haber usado herencia, este cambio no sería posible y consideramos que es necesario dar soporte para esto.

- Para los tipos de usuario por el momento hemos decidido implementar una herencia en lugar de usar una composición, ya que no nos interesa que exista la posibilidad que un usuario estandar pase a ser administrador pues consideramos que podría traer problemas de seguridad.
- Optamos por delegar la validación de contraseñas a una nueva clase, dado que no consideramos que esta responsabilidad corresponde al usuario y, además, beneficia la extensibilidad al permitir la adición de nuevas validaciones creando métodos en esta nueva clase.
- Para validar que no hay contraseñas con caracteres repetidos utilizamos un regex. Descartamos utilizar un archivo con caracteres repetidos, dado que podría perjudicar su extensibilidad y dar lugar a posibles errores como la omisión de algún caso particular.