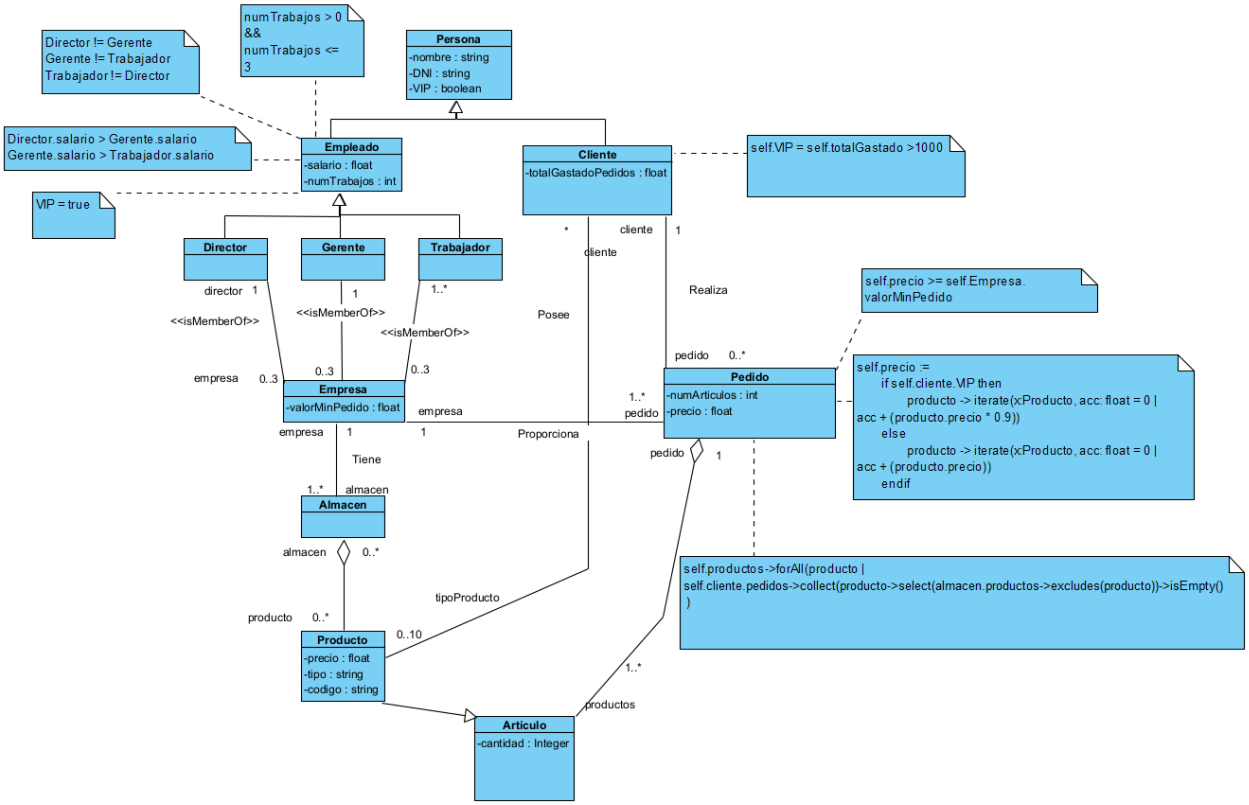
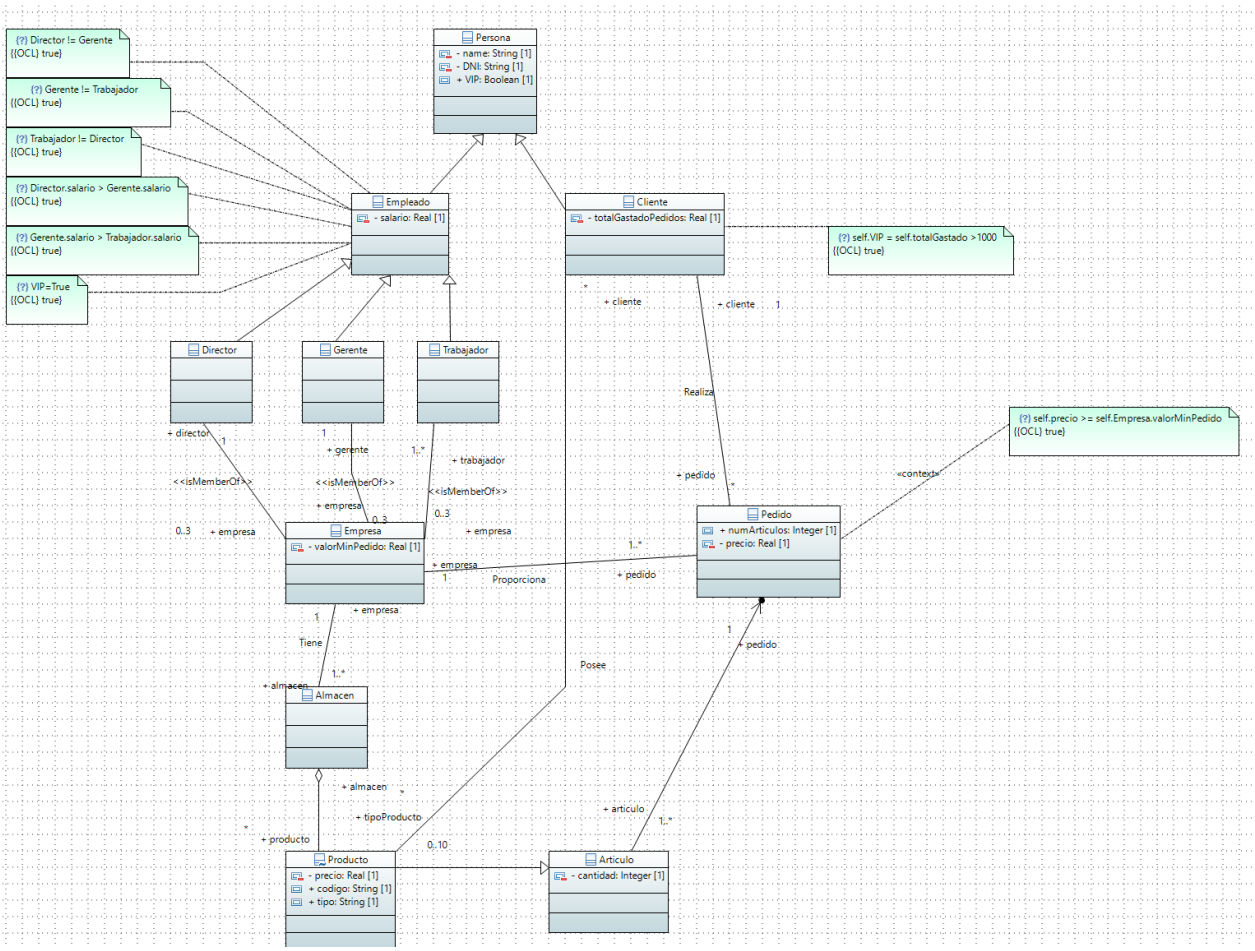
MEMORIA PRÁCTICA 1 GRUPO C4:

Cristian Ruiz Martín
Eduardo García Rivas
Pablo Rubia Arias
Ignacy Borzestowski
Mikołaj Żabski

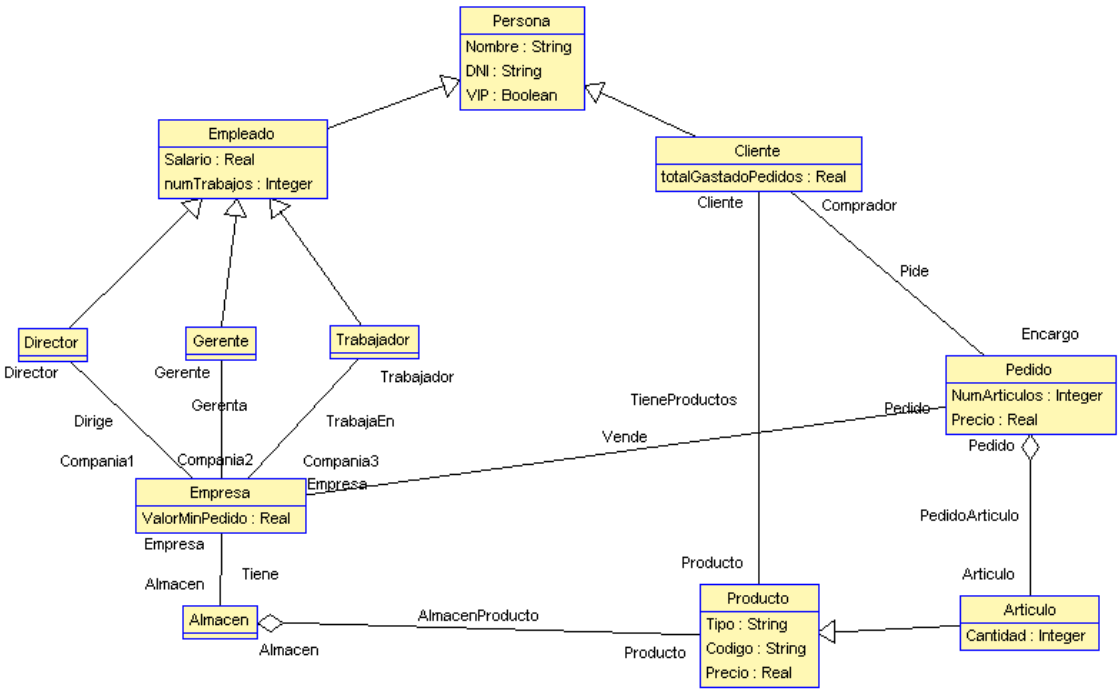
VISUAL PARADIGM



PAPYRUS


{?} Director != Gerente
{{OCL} true}
{?} Gerente != Trabajador
{{OCL} true}
{?} Trabajador != Director
{{OCL} true}
{?} Director.salario > Gerente.salario
{{OCL} true}
{?} Gerente.salario > Trabajador.salario
{{OCL} true}
{?} VIP=True
{{OCL} true}
Persona
- name: String [1]
- DNI: String [1]
+ VIP: Boolean [1]
Empleado
- salario: Real [1]
Cliente
- totalGastadoPedidos: Real [1]
{?} self.VIP = self.totalGastado >1000
{{OCL} true}
Director
Gerente
Trabajador
+ cliente
+ cliente
1
Realiza
+ pedido
{?} self.precio >= self.Empresa.valorMinPedido
{{OCL} true}
«context»
+ director
1
+ gerente
1..*
+ trabajador
<<isMemberOf>>
<<isMemberOf>>
<<isMemberOf>>
+ empresa
0..3
0..3
0..3
+ empresa
Empresa
- valorMinPedido: Real [1]
+ empresa
+ empresa
1
Proporciona
1..*
+ pedido
Pedido
+ numArticulos: Integer [1]
- precio: Real [1]
+ empresa
1
Tiene
1..*
+ almacen
Almacen
Posee
1
+ pedido
+ almacen
+ tipoProducto
+ producto
0..10
+ articulo
1..*
Producto
- precio: Real [1]
+ codigo: String [1]
+ tipo: String [1]
Articulo
- cantidad: Integer [1]

USE


Persona
Nombre : String
DNI : String
VIP : Boolean
Empleado
Salario : Real
numTrabajos : Integer
Cliente
totalGastadoPedidos : Real
Cliente
Comprador
Pide
Encargo
Director
Director
Gerente
Gerente
Trabajador
Trabajador
Gerenta
Dirige
TrabajaEn
TieneProductos
Pedido
NumArticulos : Integer
Precio : Real
Pedido
Vende
Compania1
Compania2
Compania3
Empresa
Empresa
ValorMinPedido : Real
Empresa
PedidoArticulo
Producto
Articulo
Almacen
Tiene
Almacen
AlmacenProducto
Producto
Tipo : String
Codigo : String
Precio : Real
Articulo
Cantidad : Integer
Almacen
Producto

CLASES:

```
model P1

class Empresa
attributes
ValorMinPedido : Real
end

class Persona
attributes
Nombre : String
DNI : String
VIP : Boolean
end

class Empleado < Persona
attributes
Salario : Real
numTrabajos : Integer
end

class Director < Empleado
end

class Gerente < Empleado
end

class Trabajador < Empleado
end

class Cliente < Persona
attributes
totalGastadoPedidos : Real
end

class Pedido
attributes
NumArticulos : Integer
Precio : Real
end

class Producto
attributes
Tipo : String
Codigo : String
Precio : Real
end

class Almacen
end

class Articulo < Producto
attributes
Cantidad : Integer
end
```

RELACIONES:

```
association Pide between
Cliente [1] role Cliente
Pedido [*] role Encargo
end

association Dirige between
Director [1] role Director
Empresa [0..3] role Compania1
end

association Gerenta between
Gerente [1] role Gerente
Empresa [0..3] role Compania2
end

association TrabajaEn between
Trabajador [1..*] role Trabajador
Empresa [0..3] role Compania3
end

association Tiene between
Empresa [*] role Empresa
Almacen [*] role Almacen
end

aggregation AlmacenProducto between
Almacen [*] role Almacen
Producto [*] role Producto
end

aggregation PedidoArticulo between
Pedido [1] role Pedido
Articulo [1..*] role Articulo
end

association TieneProductos between
Cliente [*] role Cliente
Producto [0..10] role Producto
end

association Vende between
Empresa [1] role Empresa
Pedido [1..*] role Pedido
end
```

Cada clase que involucra un puesto de trabajo es una relación de 1 a 0..3, ya que el empleado puede trabajar para 3 empresas distintas, en diferentes puestos.

Por un objeto de la clase artículo entendemos un número de productos del mismo tipo, y como en un pedido podemos añadir productos de diferente tipo, la relación es de 1 a 1..*.

También, como el cliente solo puede tener 10 tipos distintos de producto, hemos especificado que la multiplicidad es de * a 0..10.

RESTRICCIONES:

```
constraints
context Empresa inv gerenteSalaryHigherThanWorker:
Trabajador.allInstances()->forAll(w | w.Salario < self.Gerente.Salario)

context Empresa inv directorSalaryHigherThanManager:
self.Director.Salario > self.Gerente.Salario

context Empresa inv directorDifferentFromGerente:
self.Director.DNI <> self.Gerente.DNI

context Empresa inv directorDifferentFromTrabajador:
Trabajador.allInstances()->forAll(w | w.DNI <> self.Director.DNI)

context Empresa inv gerentedifferentFromTrabajador:
Trabajador.allInstances()->forAll(w | w.DNI <> self.Gerente.DNI)

context Empleado inv minNumTrabajos:
self.numTrabajos > 0

context Empleado inv maxNumTrabajos:
self.numTrabajos <= 3

context Empleado inv siempreVIP:
self.VIP = true

context Cliente inv esVIP:
self.VIP = self.totalGastadoPedidos > 1000

context Pedido inv precioMinPedido:
self.Precio >= self.Empresa.ValorMinPedido
```

Hemos añadido restricciones para comprobar que una persona no puede tener más de un rol en la misma empresa (restricciones _DifferentFrom_).

También hemos establecido que el salario del director es mayor que el del mánager que a su vez es mayor que el de un trabajador común.

Hemos limitado el número de empleos posibles de un empleado a 3. También hemos establecido que sea mayor que 0, ya que es un empleado.

Comprobamos que todos los empleados son clientes VIP, y que los clientes que han gastado más de 1000 también lo sean.

Finalmente, también comprobamos que el precio de cada pedido sea mayor que el mínimo requerido por la empresa.

COMPROBACIÓN RESTRICCIONES:

```
Empresa.soil> check
checking structure...
checked structure in 1ms.
checking invariants...
checking invariant (1) `Cliente::esVIP': OK.
checking invariant (2) `Empleado::maxNumTrabajos': OK.
checking invariant (3) `Empleado::minNumTrabajos': OK.
checking invariant (4) `Empleado::siempreVIP': OK.
checking invariant (5) `Empresa::directorDifferentFromGerente': OK.
checking invariant (6) `Empresa::directorDifferentFromTrabajador': OK.
checking invariant (7) `Empresa::directorSalaryHigherThanManager': OK.
checking invariant (8) `Empresa::gerenteSalaryHigherThanWorker': OK.
checking invariant (9) `Empresa::gerentedifferentFromTrabajador': OK.
checking invariant (10) `Pedido::precioMinPedido': OK.
checked 10 invariants in 0.013s, 0 failures.
```

EMPRESA.SOIL

```
reset

!-- Personas creation
!new Director
!Director1.Nombre := 'Dir John'
!Director1.DNI := 'D123'
!Director1.VIP := true
!Director1.Salario := 3000
!Director1.numTrabajos := 1

!new Gerente
!Gerente1.Nombre := 'Ger John'
!Gerente1.DNI := 'G123'
!Gerente1.VIP := true
!Gerente1.Salario := 2000
!Gerente1.numTrabajos := 1

!new Trabajador
!Trabajador1.Nombre := 'Trab John'
!Trabajador1.DNI := 'T123'
!Trabajador1.VIP := true
!Trabajador1.Salario := 1500
!Trabajador1.numTrabajos := 1

!new Cliente
!Cliente1.Nombre := 'Cli John'
!Cliente1.DNI := 'C123'
!Cliente1.totalGastadoPedidos := 1200
!Cliente1.VIP := true

!new Pedido
!Pedido1.NumArticulos := 8
!Pedido1.Precio := 800

!new Producto
!Producto1.Tipo := 'Martillo'
!Producto1.Codigo := '1'
!Producto1.Precio := 100

!new Articulo
!Articulo1.Tipo := 'Martillo'
!Articulo1.Codigo := '1'
!Articulo1.Precio := 100
!Articulo1.Cantidad := 8

!-- Empresa creation
!new Empresa
!Empresa1.ValorMinPedido := 100

!new Almacen

!-- Relations
!insert (Director1, Empresa1) into Dirige
!insert (Gerente1, Empresa1) into Gerenta
!insert (Trabajador1, Empresa1) into TrabajaEn
!insert (Empresa1, Almacen1) into Tiene
!insert (Cliente1, Pedido1) into Pide
!insert (Cliente1, Producto1) into TieneProductos
!insert (Pedido1, Articulo1) into PedidoArticulo
!insert (Empresa1, Pedido1) into Vende
!insert (Almacen1, Producto1) into AlmacenProducto

!-- To check constraints
check
```

DIAGRAMA DE OBJETOS: