

Composición

La composición es una representación de una relación fuerte "todo - parte", donde una clase y una entidad dependen de otra si o si para existir.

Eso decir, si el objeto "todo" se destruye sus partes también deben de existir.

O sea el "todo" tiene a las partes y controla su ciclo de vida.

En UML se representa con un rombo negro.

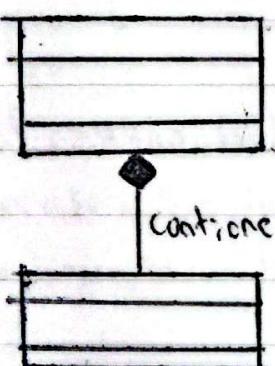
Características Principales:

- Relación fuerte.
- El "todo" crea y destruye sus partes.
- Las partes no existen o no tienen sentido sin el "todo".

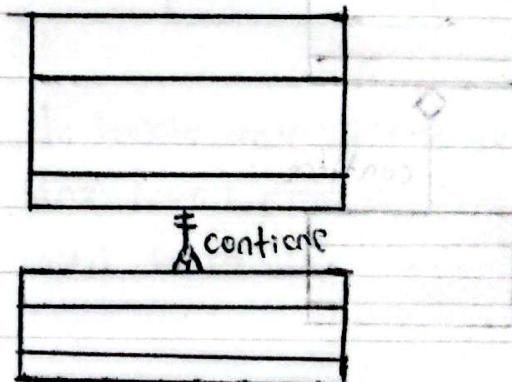
En MER equivale a una relación identificadora.

La identidad de la entidad depende de la cantidad fija de partes para poder existir y para establecer su clavijón se representa con un rombo fuerte y una tira sólida.

Ejemplo UML:



Ejemplo MER:



Agregación

La agregación es una representación de una relación donde "todo-parte" o "tire-un", donde los partes pueden existir de una manera independiente del todo.

El todo usa o usa objetos, pero no los tiene completamente.

En UML se representa con un rombo blanqueo.

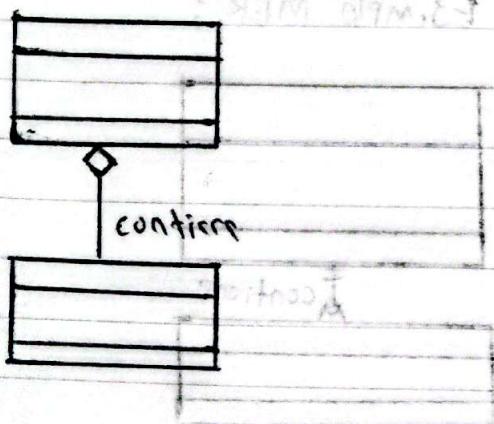
Características Principales:

- Relación débil
- los partes no dependen del todo para poder existir
- No existe destrucción automática de los partes

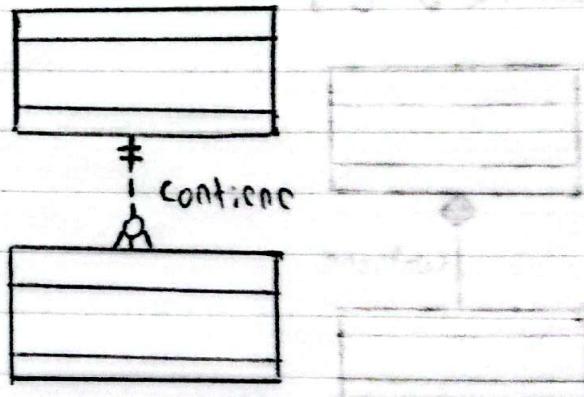
En MEF equivale a una relación no identificadora.

Ambos extremos tienen su propia clase primaria y la relación se representa con un rombo simple o con una linea discontinua.

Ejemplo UML:

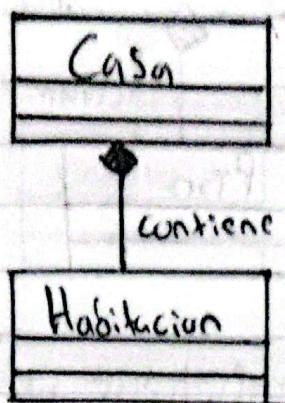


Ejemplo MEF:

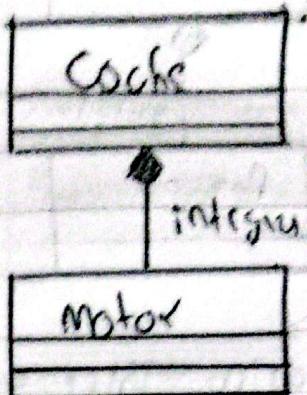


Composición (Diagrama de Clases UML)

1. Casa - Habitación



2. Coche - Motor

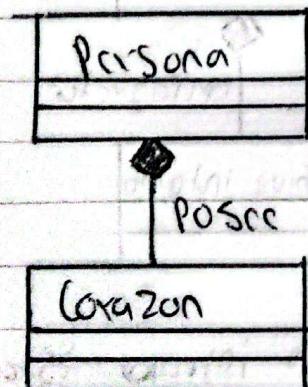


Los habitaciones dependen de una casa.

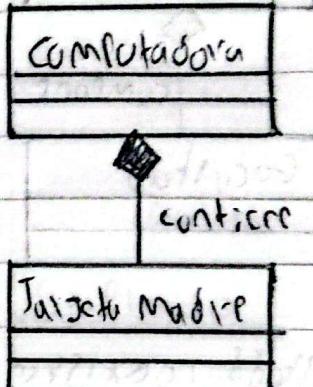
Si la casa se destruye los habitaciones desaparecerán.

El motor está diseñado y ensamblado para un coche en específico. Si el coche deja de existir, el motor pierde su contexto.

3. Persona - Corazón



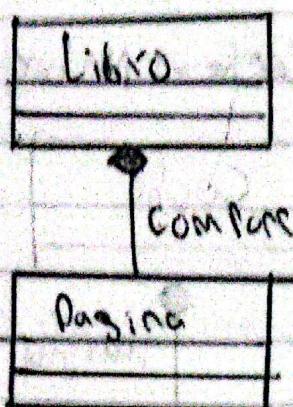
4. Computadora - Tarjeta Madre



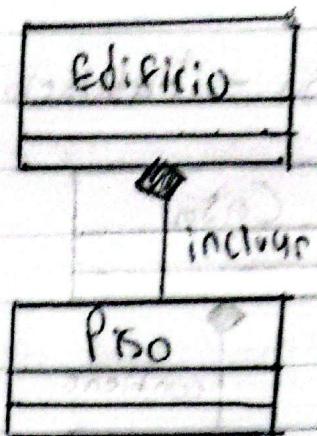
El corazón pertenece a una persona concreta. Sin la persona, el corazón no tiene razón de ser.

La tarjeta madre forma parte de una computadora. No tiene vida útil fuera de ella.

5. Libro - Página



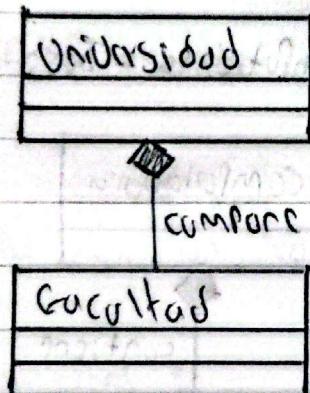
6. Edificio - Piso



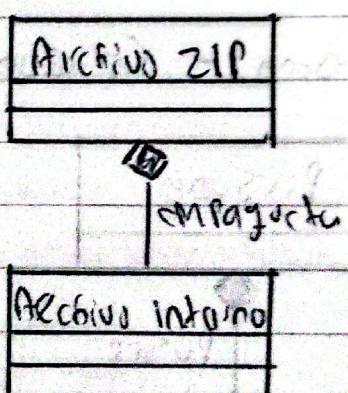
Los páginas no existen solas pertenecen a un libro ; si se destruye el libro, su conjunto de páginas pierden su contexto.

Un piso pertenece al edificio que lo contiene . Si el edificio se derrumba , los pisos desaparecen.

7. Universidad - Facultad



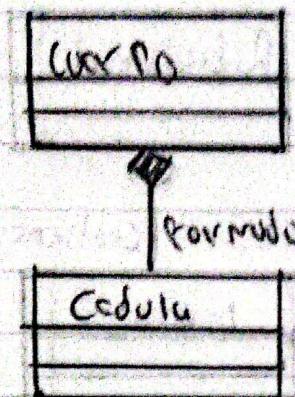
8. Archivo ZIP - Archivo interno



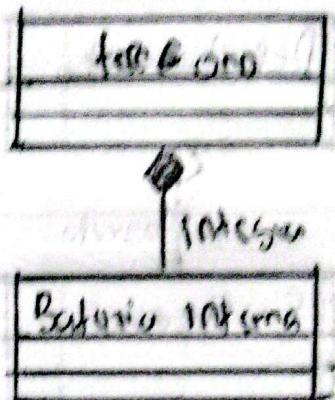
La facultad existe dentro de la estructura universitaria.

Los Archivo interno están comprimidos una vez zip, sin embargo la forma en que existen.

9. CURSO - CEDULA



10. telefono - Bateria interna

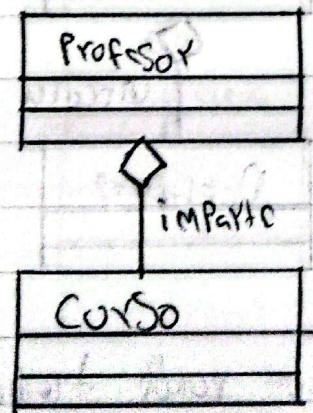


Las cedulas manejan si el curso
dura de funcionar

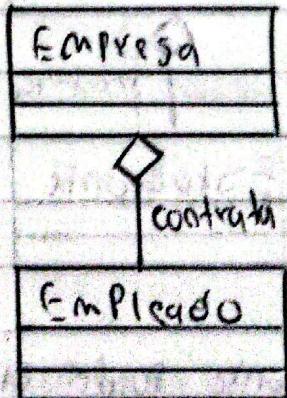
La bateria esto integrada
es decir al telefono sin el no
tendria control

Agregacion (Diagrama de clases UML)

1. Profesor - CURSO



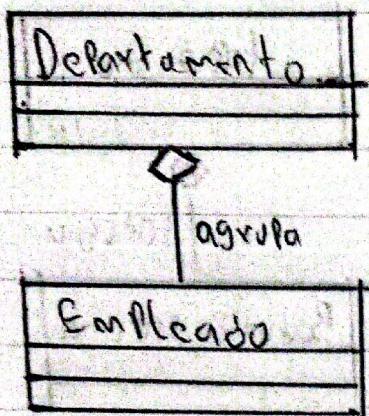
2. Empresa - Empleado



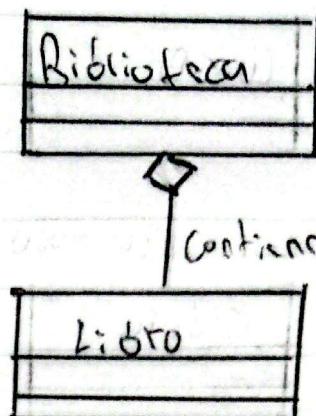
Los cursos pueden existir sin un
profesor (ojo profesor podria dictarlo)

Un empleado puede existir sin
la empresa (puede trabajar en otra)

3. Departamento - Empleado



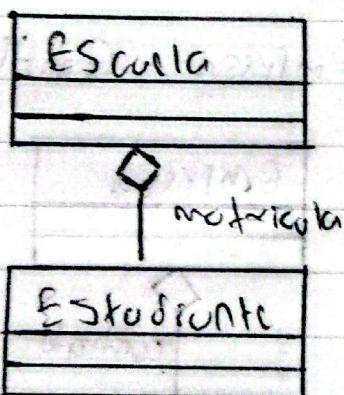
4. Biblioteca - Libro



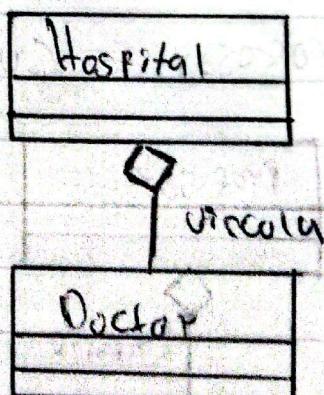
El empleado puede combinar de
departamento.

Los libros existen independientemente
de una biblioteca los agrupa
temporalmente.

5. Escuela - Estudiante



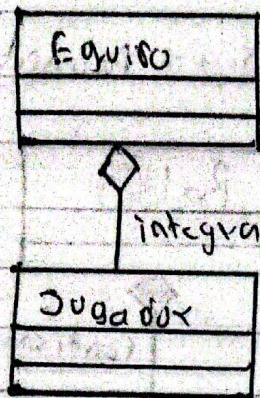
6. Hospital - Doctor



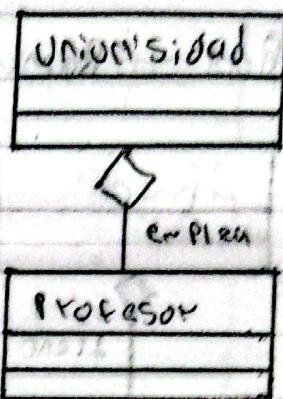
El estudiante puede matricular sin
estar inscrito en una escuela.

El doctor puede trabajar en
distintos hospitales o por su
propia cuenta.

7. Equipo - Jugador



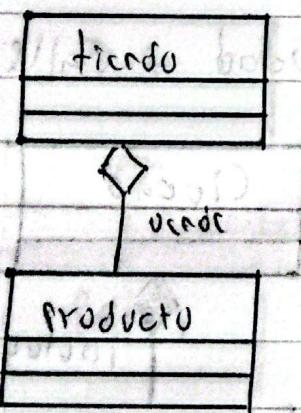
8. Universidad - Profesor



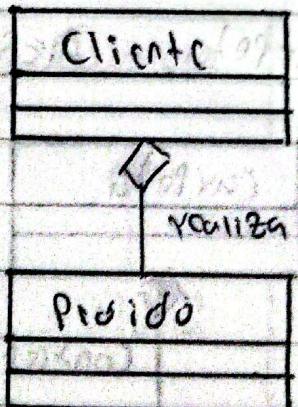
El Jugador puede cambiar de equipo o retirarse.

El profesor puede trabajar en varias universidades o en ninguna.

9. Tienda - Producto



10. Cliente - Pedido

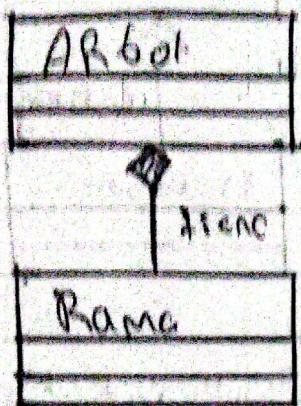


El producto puede existir fuera de la tienda en inventario o en otra tienda.

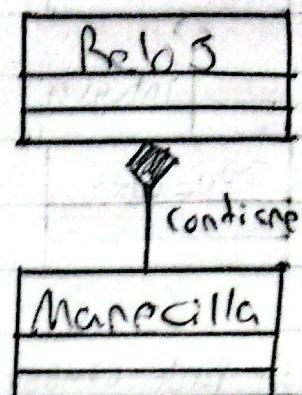
El cliente puede existir sin tener pedidos activos.

Composición (Diagrama de clases UML)

1. Árbol - Ramo



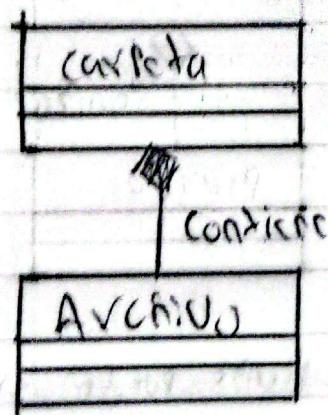
2. Reloj - Manecilla



Una rama NO existe por si sola;
Fueron partes biológicas del árbol

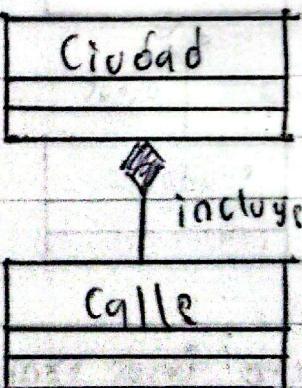
los manecillas NO existen sin
el reloj que los contiene y
controla

3. Carpeta - Archivo



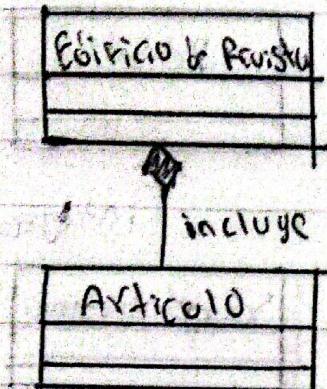
En este contexto: en un sistema donde
borrar la carpeta borra también
todo su contenido

4. Ciudad - Calle

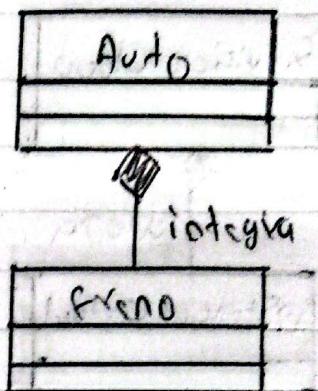


Una calle forma parte de la
ciudad: Sin la ciudad esa calle
 pierde identidad administrativa.

5. Edificio de Revista - Artículo



6. Auto - Freno



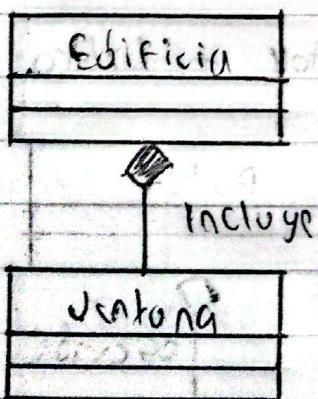
Los artículos pertenecen a una edición específica. Si esa edición no existe, ese conjunto de artículos tampoco.

La pieza forma parte física del vehículo. Un carro sin auto pierde sentido funcional.

7. Mesa - Patos



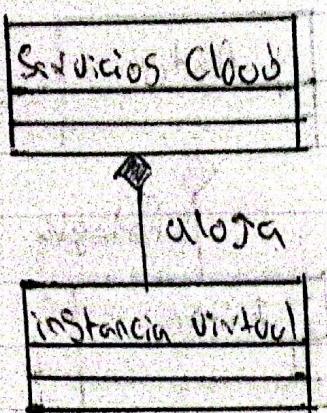
8. Edificio - Ventana



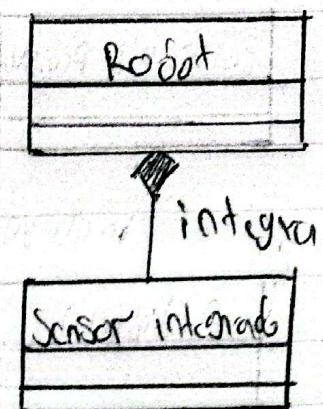
Los patos pertenecen a una mesa específica y no existen independiente

La Ventana forma parte física del edificio. Si esto se destruye, las ventanas desaparecen.

9. Servicios Cloud - instancia virtual



10. Robot - Sensor integrado

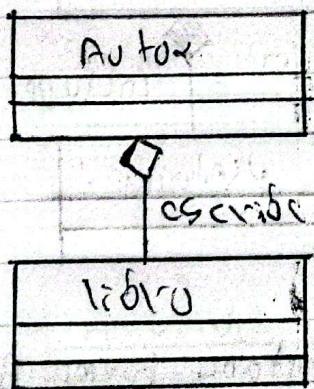


Una instancia virtual solo existe porque el proveedor mantiene la mantendrá.

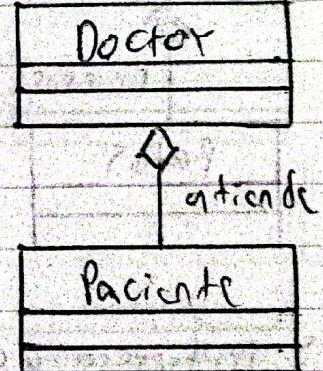
Los sensores integrados están ensamblados físicamente al robot y dependen de él.

Agregación (Diagrama de clases UML)

1. Autor - Libro



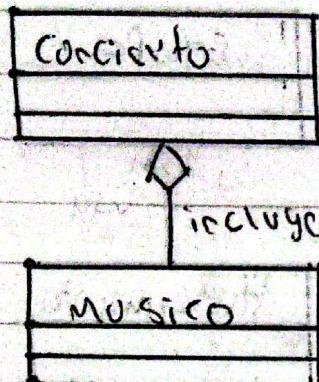
2. Doctor - Paciente



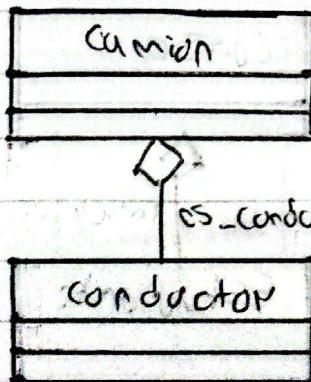
El autor puede existir sin escribir libros. Un libro puede existir sin un autor. (por ejemplo basa de datos)

Ambos existen independientemente. Solo se relacionan temporalmente.

3. Concierto - Músico



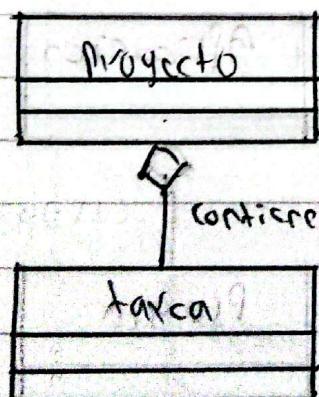
4. Camión - Conductor



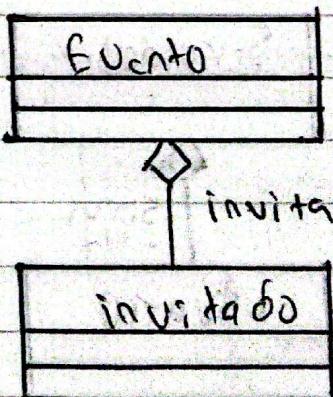
El músico puede tocar en distintos conciertos o ~~trabajando~~

el conductor puede manejar distintos camiones o trabajar en otro lugar.

5. Proyecto - tarea



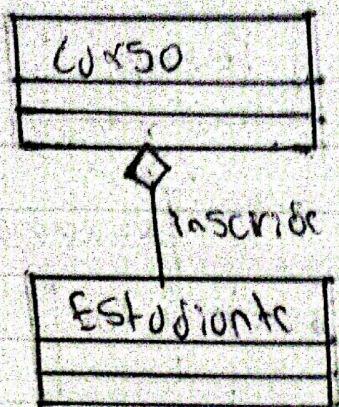
6. Evento - invitado



Las fuerzas pueden transferirse entre proyectos o existir si asignación temporal

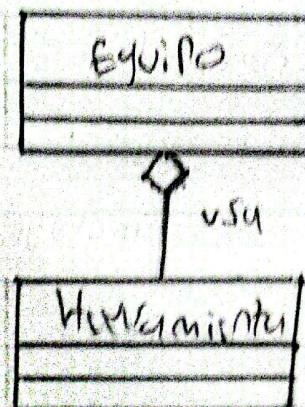
un invitado existe fuera del evento

7. Curso - Estudiante



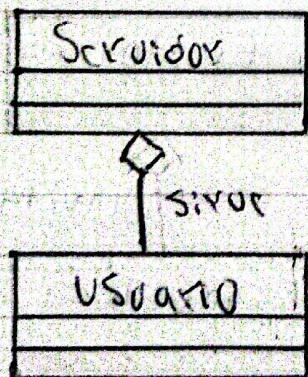
El estudiante puede estar en
varios cursos o en ninguno

8. Equipo - Herramienta



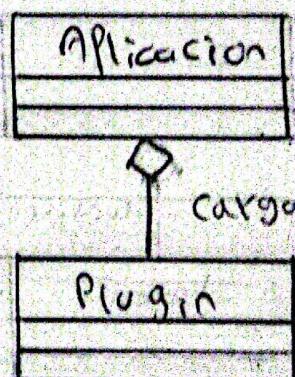
Las herramientas existen
independientemente

9. Servidor - Usuario



El usuario existe aunque no
use el servidor

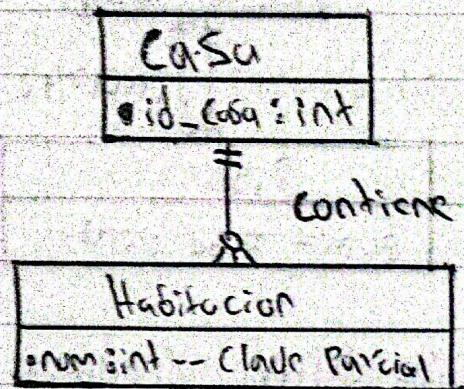
10. Aplicacion - Plugin



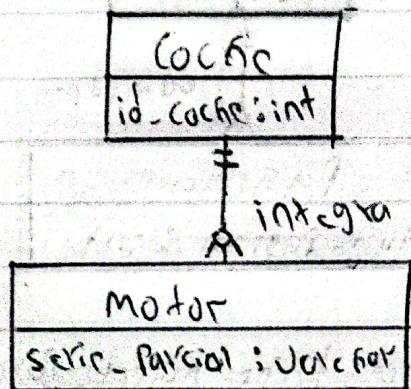
El plugin se puede instalar en
multiples aplicaciones o
existir desinstalado.

Composición (MER)

1. Casa - Habitación



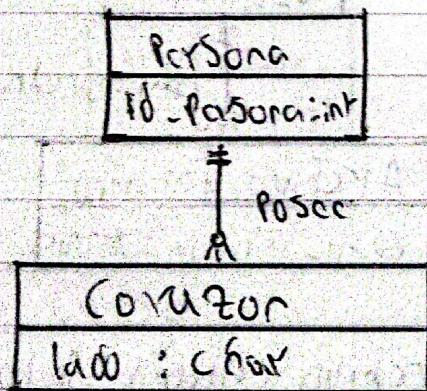
2. Coche - Motor



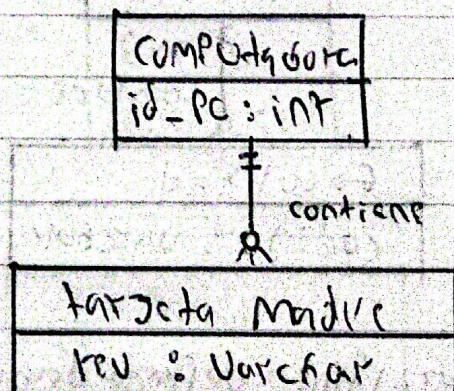
Habitación tiene clave Parcial
Ocupa sierte de casa

Motor no tiene identida sin
coche

3. Persona - Corazón



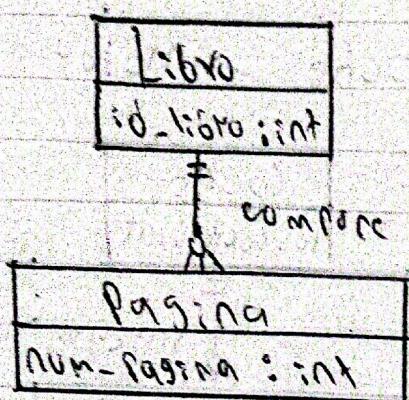
4. Computadora - tarjeta Madre



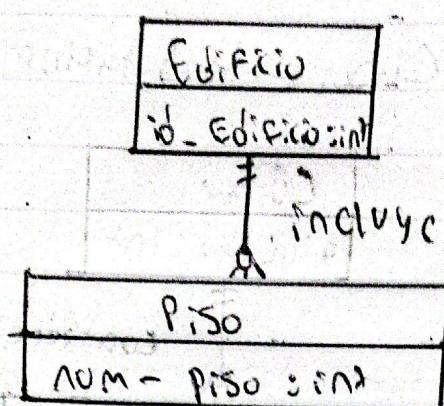
Corazón pertenece a una
persona específica

tarjeta madre identificada
por su computadora

5. Libro - Página



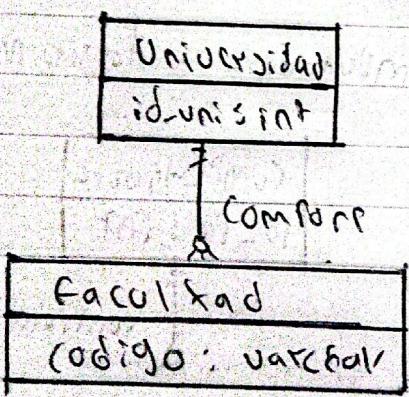
6. Edificio - Piso



Página tiene número oficial dentro del libro

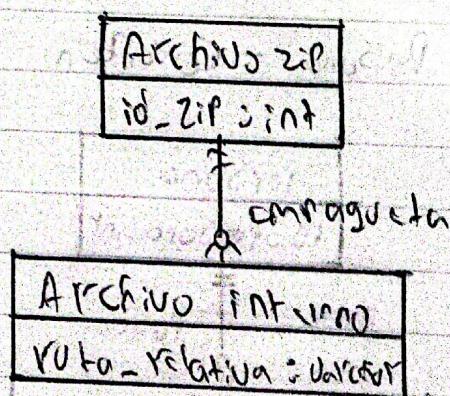
Piso define definición de edificio

7. Universidad - Facultad



Facultad define definición de universidad

8. Archivo ZIP - Archivo interno



Archivo interno, solo existe dentro del ZIP

9. Cuerpo - Cedula

10. telefono - Bateria Interna

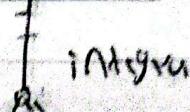
| |
|-----------------|
| Cuerpo |
| id_cuerpo : int |



formado_por

| |
|--------------|
| Cedula |
| indice : int |

| |
|--------------|
| telefono |
| id_tel : int |



| |
|------------------|
| Bateria interna |
| Modulo : parcial |

Cedula pertenece a cuerpo

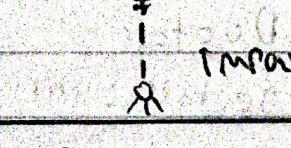
Bateria interna dentro del teléfono

Agregacion (MER)

1. Profesor - CURSO

2. Empresa - Empleado

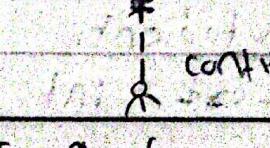
| |
|---------------|
| Profesor |
| id_Prof : int |



imparte

| |
|----------------|
| CURSO |
| id_CURSO : int |

| |
|--------------|
| Empresa |
| id_emp : int |



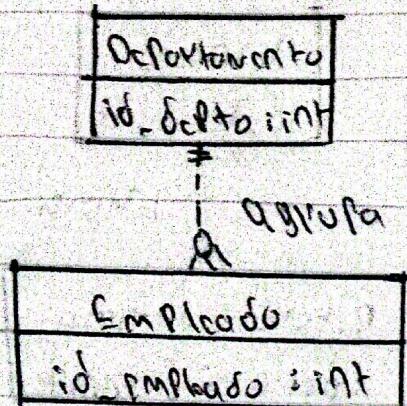
contrata

| |
|-------------------|
| Empleado |
| id_empleado : int |

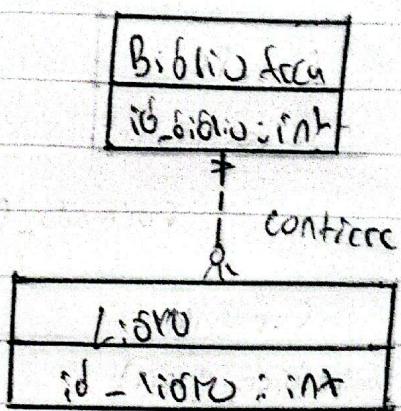
CURSO tiene su propia
relación simple "imparte"

Empleado tiene su propia
clave y puede estar
vinculado en otra empresa

3. Departamento - Empleado



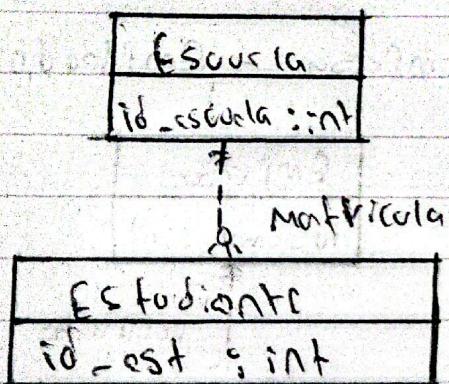
4. Biblioteca - Libro



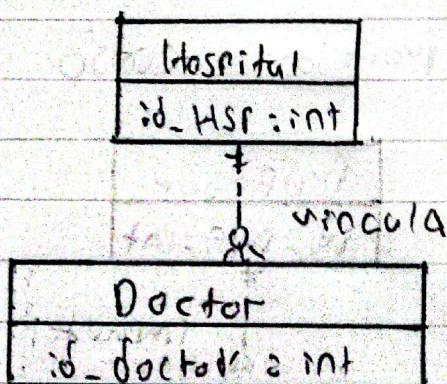
Asociación 1..n: Un empleado
puede estar o no en un empleo

Libro es independiente de
Biblioteca

5. Escuela - Estudiante



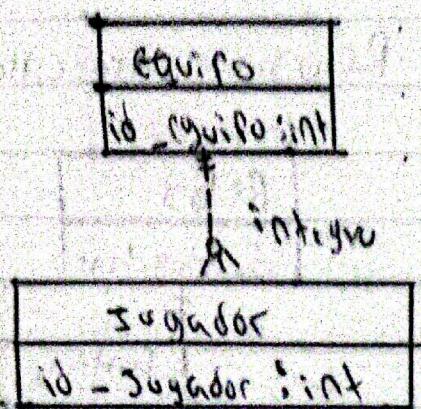
6. Hospital - Doctor



Estudiante tiene su propia identidad
aunque sea de la escuela.

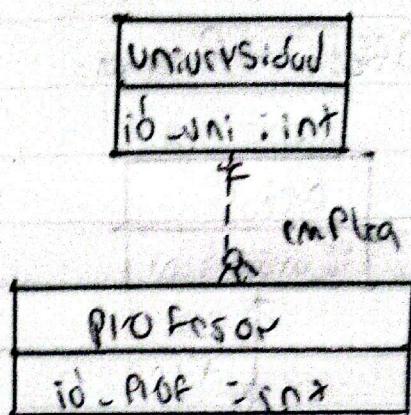
Doctor puede estar o
no asociado a varias facultades

7. Equipo - Jugador



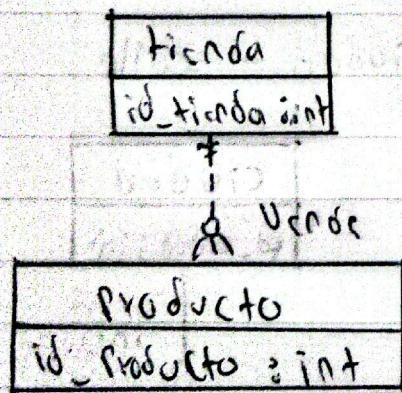
Jugador puede participar en varios equipos o lisos

8. Universidad - Profesor



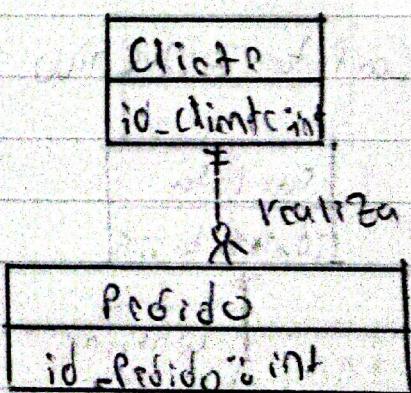
profesor okupa distintos
mentres mismo labora
flexible

9. tienda - Producto



Producto tiene existencia para
en la tienda.

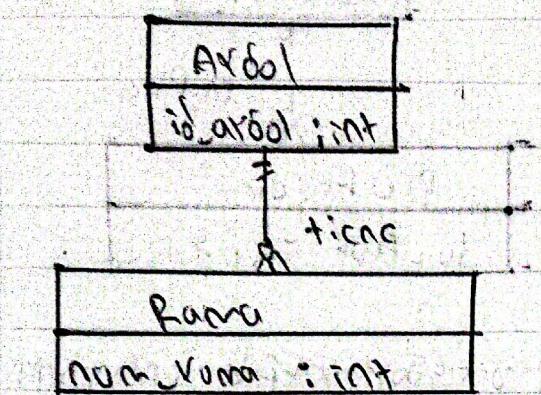
10. Cliente - Pedido



Pedido referencia a un
cliente pero al cliente
sin existiendo sin pedido

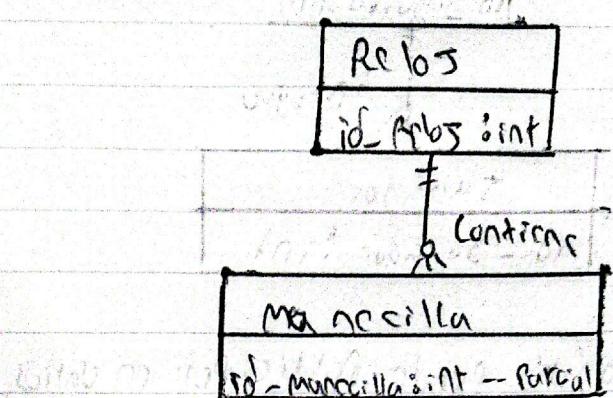
Composición (MER)

1. Árbol - Rama



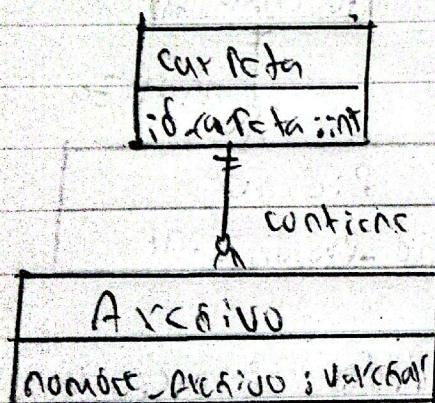
Rama no puede existir sin Árbol.
 (lava de Rama = (id_Arbol, num_Rama)).

2. Reloj - Manecilla



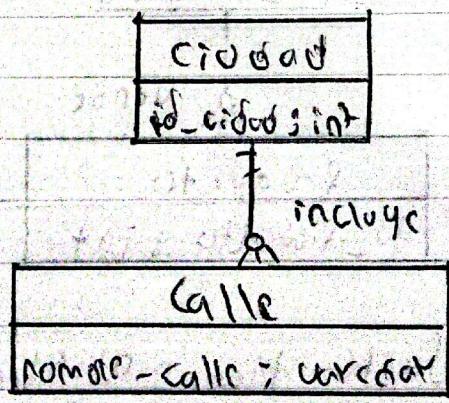
Manecilla depende del Reloj

3. Carpeta - Archivo



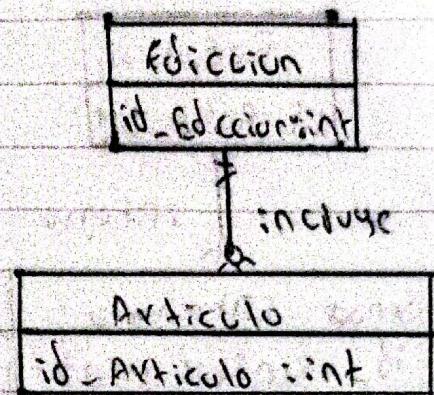
Archivo depende de carpeta

4. Ciudad - calle

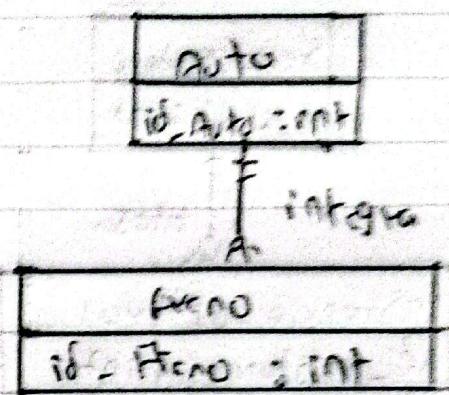


Calle necesita una ciudad
 para existir administrativamente

5. Edicion - Articulo



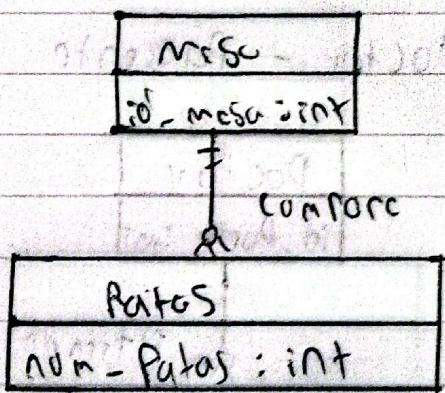
6. Auto - Freno



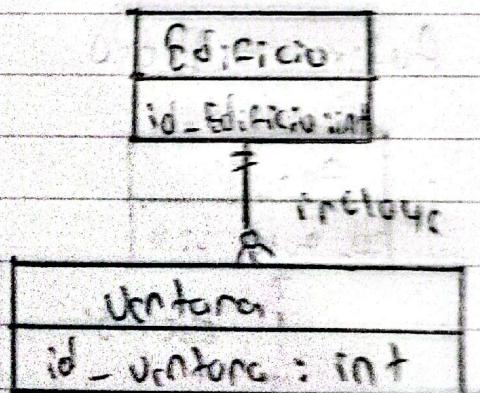
los Articulos dependen de la Edicion
a la que pertenezcan

los frenos se identifican por
el auto al que pertenezcan

7. Mesa - Patos



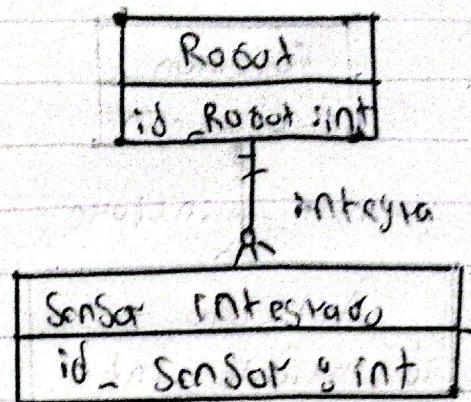
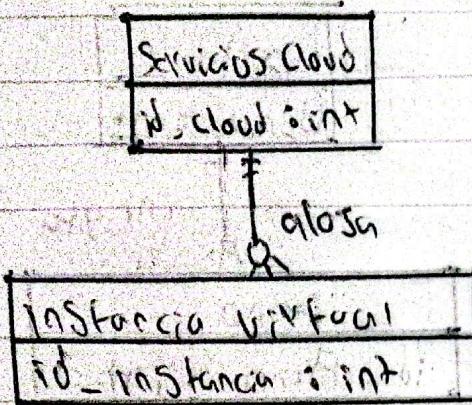
8. Edificio - Ventana



los Patos dependen de la mesa

las ventanas dependen del
edificio

9. Servicios Cloud - instancia Virtual 10. Robot - Sensor integrado

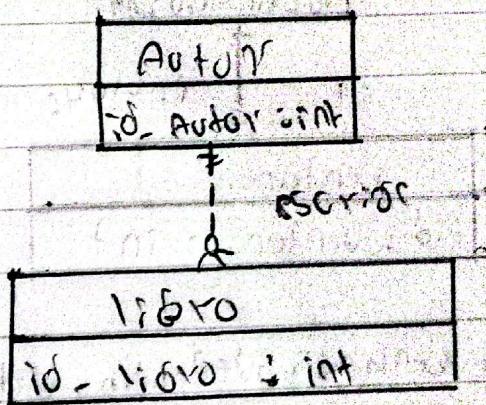


Una instancia virtual no existe si su servidor no existe

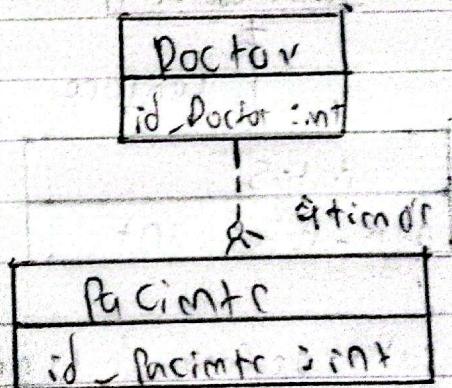
el sensor depende del robot, incluso en claus

Agregación (MER)

1. Autor - Libro



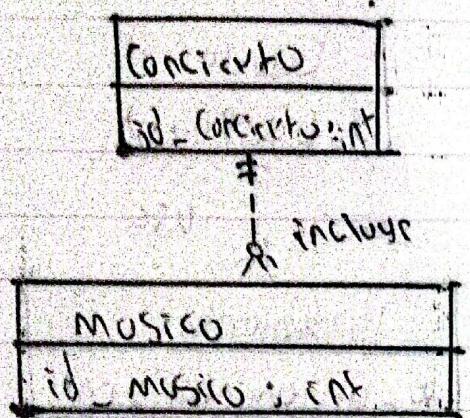
2. Doctor - Paciente



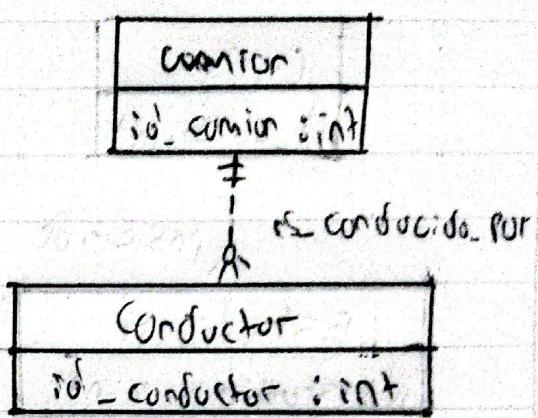
Libro tiene su propio ID
no depende del autor para existir

Entidades totalmente
independientes

3. Concierto - Músico



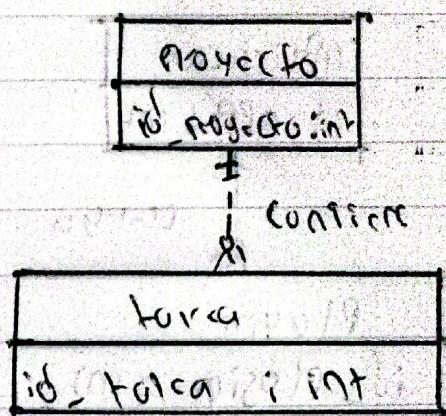
4. Comión - conductor



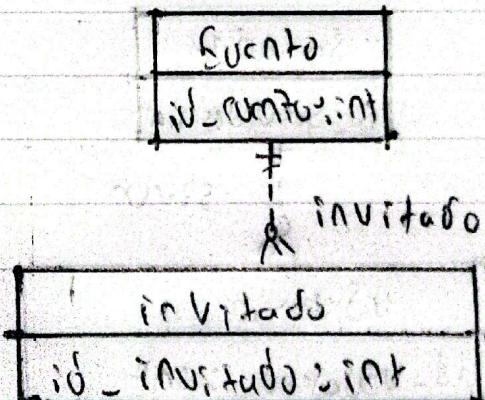
Músico tiene su propia identidad

Conductor y Comión son independientes

5. Proyecto - tarea



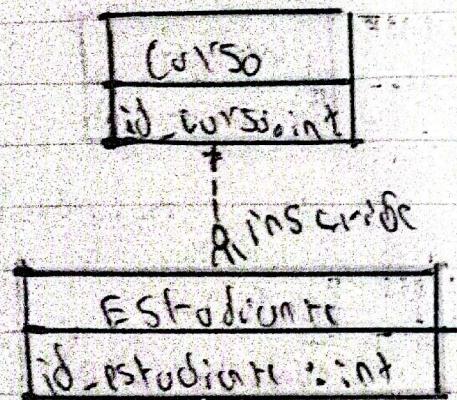
6. Evento - invitado



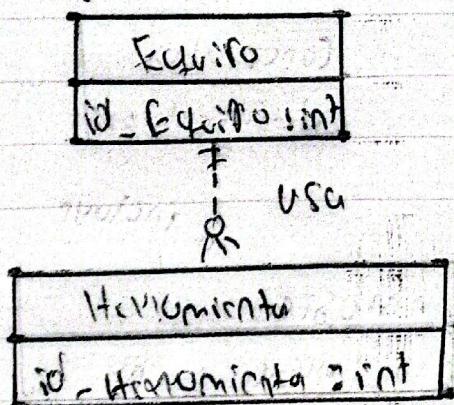
Tarea tiene su propio ID no depende de Proyecto

invitado existe fuera del evento

7. Curso - Estudiante



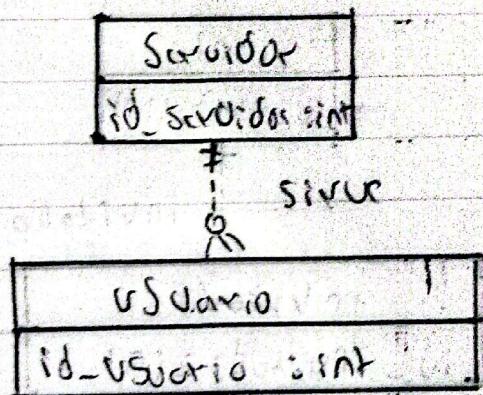
8. Equipo - Herramienta



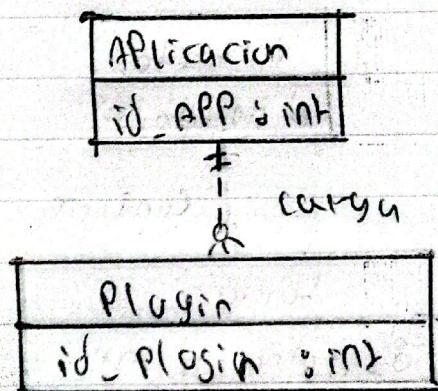
Relación estudiantil: no se puede
asistir sin curso

la herramienta existe sin
equipo

9. Servidor - Usuario



10. Aplicacion - Plugin



usuario tiene identidad propia

el plugin tiene su propia
identidad y puede existir sin
la aplicación