Cuadernillo práctico para el diseño de objetos

Nivel I y II

VERSIÓN 1, REVISIÓN 2, 14/07/2015

Eugenia Bahit

http://cursos.eugeniabahit.com

Objeto:
todo sustantivo o “cosa” con una o más características distintivas

Algunos ejemplos de objetos que pueden aparecen en los sistemas informáticos:

Sistemas comerciales	e-Learning	Informática Médica y afines
Cliente Producto Servicio Pedido Factura	Alumno Examen Docente Asignatura Curso	Paciente HistoriaClinica ProfesionalMedico Medicamento PrincipioActivo

Propiedad (o atributo):
cada una de las características distintivas de un objeto

Algunos ejemplos de propiedades que pueden aparecen en los sistemas informáticos:

Cliente	Producto	Medicamento
denominacion Domicilio DatoDeContacto (colección)	denominacion precio	denominacion PrincipioActivo (colección) Excipiente (colección)
Curso	**Editorial**	**Autor**
denominacion Asignatura (colección)	denominacion Libro (colección)	apellidos nombres

Propiedad compuesta:
propiedad cuyo valor es otro objeto

Algunos ejemplos de propiedades compuestas que pueden aparecen en los sistemas informáticos:

Cliente	Empleado	Paciente	Asignatura
Domicilio	ContratoDeTrabajo	HistoriaClinica	Docente

Propiedad colectora:
propiedad cuyo valor es una matriz (array) con más de un objeto del mismo tipo

Algunos ejemplos de propiedades colectoras que pueden aparecen en los sistemas informáticos:

Editorial	Cliente	Evento
Libro_collection	DatoDeContacto_collection	Invitado_collection

Propiedad simple:
propiedad cuyo valor es de un único tipo de datos (string, int, float, boolean), exceptuando las propiedades compuestas y las colectoras

Algunos ejemplos de propiedades simples que pueden aparecen en los sistemas informáticos:

Cliente	Empleado	Libro	Producto
denominacion	apellido nombre	denominacion isbn	denominacion precio

Objeto Compuesto:
un objeto que es compuesto por otro objeto (todo objeto con al menos 1 propiedad compuesta o 1 propiedad colectora)

Algunos ejemplos de objetos compuestos que pueden aparecen en los sistemas informáticos:

Objeto compuesto (se compone de)	Objeto compositor (compone a)
Cliente	Domicilio DatoDeContacto (muchos)
Empleado	ContratoDeTrabajo DatoDeContacto (muchos)
Pais	Provincia (muchos)
Provincia	Ciudad (muchos)
Paciente	HistoriaClinica Domicilio DatoDeContacto (muchos)
ProfesionalMedico	Paciente (muchos)
Pedido	DomicilioDeEntrega Producto (muchos)

Objeto Compositor:
Objeto que compone a otro objeto *(para ejemplos ver tabla anterior)*

Objeto Colector:
Objeto que se compone SOLO Y ÚNICAMENTE de la colección COMPLETA de objetos de un mismo tipo. Todos los objetos tienen a su propio colector.

Compositor único (de pertenencia o exclusivo):

objeto compositor que solo puede componer a un objeto y no más de uno

Algunos ejemplos de compositores únicos que pueden aparecen en los sistemas informáticos:

Objeto Compositor	Es compositor exclusivo de:
DatoDeContacto	Cliente
Domicilio	Cliente
Pedido	Cliente
Provincia	Pais

Un compositor exclusivo siempre llevará una propiedad simple denominada propiedad de pertenencia, cuyo valor será solo la ID del objeto al que compone:

Compositor	Compuesto	Propiedad de pertenencia (en el compositor)
DatoDeContacto	Cliente	*int* cliente
Domicilio	Cliente	*int* cliente
Pedido	Cliente	*int* cliente
Provincia	Pais	*int* pais

IMPORTANTE:

Solo los compositores exclusivos requieren de una

propiedad de pertenencia + un método *get* auxiliar.

Para que un compositor sea considerado “exclusivo”, el compuesto deberá componerse por más de uno de estos compositores.

Compositor reutilizable:

objeto compositor que puede componer a más de un objeto.

Algunos ejemplos de compositores reutilizables que pueden aparecen en los sistemas informáticos:

Objeto Compositor	Puede componer a más de un...
Producto	Pedido
PrincipioActivo	Medicamento
Invitado	Evento
Docente	Asignatura

Un compositor reutilizable siempre requiere de un tercer objeto denominado "Conector Lógico" que establezca la relación entre compositores y compuestos.

Conector Lógico Relacional:

objeto cuya única finalidad es establecer la relación entre compositores reutilizables y sus compuestos.

Algunos ejemplos de conectores lógicos que pueden aparecen en los sistemas informáticos:

Compositor reutilizable	Compuesto	Conector Lógico
Producto	Pedido	ProductoPedido
PrincipioActivo	Medicamento	PrincipioActivoMedicamento
Invitado	Evento	InvitadoEvento
Docente	Asignatura	DocenteAsignatura

Todos los conectores lógicos tienen las mismas propiedades:

- *Compuesto:* el objeto compuesto completo
- *Compositor_collection:* propiedad colectora del objeto compuesto

Clasificación de objetos según sus características:

Si el objeto....	Objeto Estándar	Objeto Compuesto	Objeto Compositor	
			Exclusivo	Reutilizable
Tiene propiedades simples	X			
Tiene al menos una propiedad compuesta		X		
Tiene al menos una propiedad colectora		X		
Compone a otro objeto			X	
Muchos, componen a un solo objeto			X	
Muchos, componen a muchos objetos				X

Tipo de propiedades según su composición interna:

	Simple	Compuesta	Colectora	De pertenencia (o de dependencia directa)
No se compone de ningún objeto	X			
Se compone de un único objeto		X		
Se compone de más de un objeto			X	
Su valor es la ID de otro objeto				X
Su valor es su propia ID	X			

Propiedades aplicables a cada tipo de objeto:

OBJETO / PROPIEDAD	Estándar	Compuesto	Compositor		Colector	Conector Lógico
			Exclusivo	Reutilizable		
ID	1	1	1	1	0	1
Simple	Todas	0 o más	0 o más	0 o más	0	0
Compuesta	0	1 o más	0 o más	0 o más	0	1
Colectora	0	0 o más	0 o más	0 o más	1	1
De pertenencia	0	0	1	0	0	0

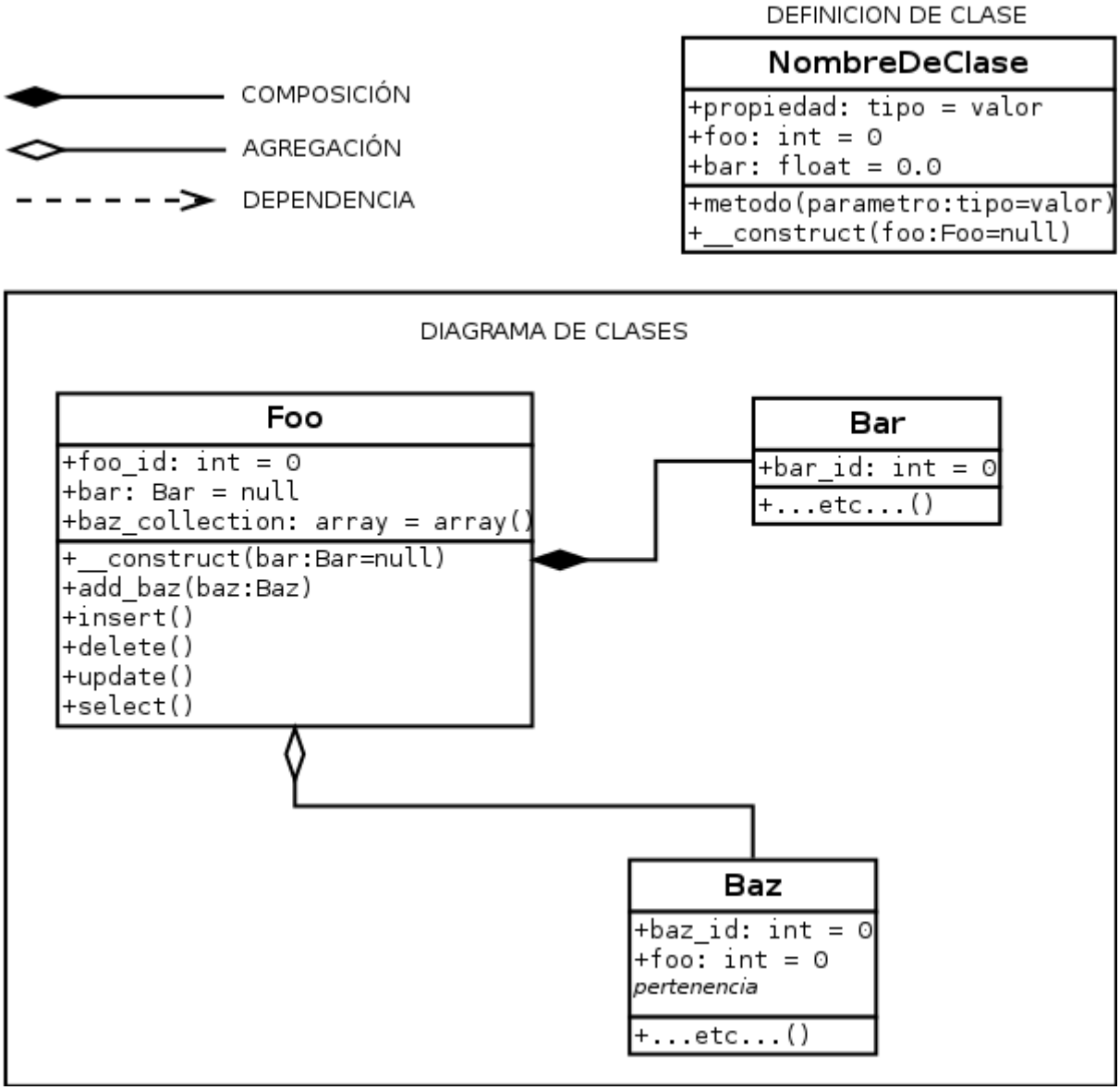
Métodos presentes en los objetos:

OBJETO / MÉTODO	Estándar	Compuesto	Compositor		Colector	Conector Lógico
			Exclusivo	Reutilizable		
`Select|get`	X	X	X	X	X	X
Get auxiliar			X			
`Update|save`	X	X	X	X		
`insert|save`	X	X	X	X		X
`delete|destroy`	X	X	X	X		X
Método de agregación		0 o más				

Estructura interna de los métodos de un objeto:

`insert()` *guarda un nuevo objeto en la db*	**`select()`** *recupera un objeto por su id*	**`update()`** *guarda los cambios de un objeto preexistente*
1. Definir SQL 2. Definir array de datos (propiedades) 3. Ejecutar query → retorno setea ID	1. Definir SQL 2. Definir array de datos (solo ID) 3. Ejecutar query 4. Setear propiedades 5. (opcional) componer propiedades compuestas 6. (opcional) componer propiedades colectoras	1. Definir SQL 2. Definir array de datos (propiedades + ID) 3. Ejecutar query
`delete()` *elimina un objeto preexistente de la DB*	**`add_OBJETO(OBJETO $objeto)`**	**`get_auxiliar($COMPUESTO_id)`** *Recupera la ID de todos los compositores exclusivos que componen a COMPUESTO*
1. Definir SQL 2. Definir array de datos (ID) 3. Ejecutar query	*Agrega OBJETO a la propiedad colectora OBJETO_collection*	1. Definir SQL 2. Definir array de datos ($compuesto_id) 3. Ejecutar query 4. Retornar resultado

UML: Tarjeta de referencias rápida



Software para realizar diagramas UML en GNU/Linux:

```
apt-get install dia
```

Análisis de objetos del sistema

OBTENER OBJETOS:
Los objetos se obtienen a partir de la *acción a realizar* de una Historia de Usuario:

Como [tipo de usuario] puedo [acción a realizar]

Ejemplos de HU:

HU de un sistema de gestión para consultorios médicos
Como secretaria puedo *agregar una nueva agenda de turnos*
Como secretaria puedo *agregar un nuevo turno a la agenda de turnos*
Como paciente puedo *consultar los turnos con mi médico*
Como médico puedo *crear una receta electrónica*

HU de un administrador de contenidos Web (CMS)
Como redactor puedo *crear un nuevo artículo en la wiki*
Como administrador puedo *crear una nueva encuesta*
Como usuario registrado puedo *agregar fotos a mi perfil de usuario*
Como visitante del sitio puedo *ver la lista de artículos de la wiki*

La acción a realizar se divide en:

- verbo en infinitivo:
  recurso del controlador
- sustantivo principal:
  generalmente, nuevo objeto
- sustantivos complementarios:
  generalmente, objetos preexistentes o derivados
- otros:
  son descartados

Acción a realizar	**Recurso**	**Objeto principal**	**Objeto(s) secundario(s)***
Agregar un nuevo turno a la agenda de turnos	Agregar	**Turno**	Agenda
Crear una receta electrónica	Crear	**Receta**	
Crear una nueva encuesta	Crear	**Encuesta**	
Ver la lista de artículos de la wiki	Ver	**Articulo**	Wiki

(*) si no están aún creados, se trata de objetos derivados; sino, de objetos preexistentes.

OBTENER PROPIEDADES:
Se extraen -inicialmente- de los Criterios de Aceptación de una Historia de Usuario:

Historia de Usuario # 35	
Como secretaria puedo agregar una nueva agenda de turnos	PRIORIDAD: VALOR: SPRINT: ACEPTADA:

Criterios de aceptación:
• Se muestra un formulario con los siguientes campos (ref: mockup #153):
  • Especialidad: [select]
  • Médico: [select] (depende de especialidad)
  • Nombre: [text] valor x defecto: Agenda Dr. [medico]
  • [btn:Guardar]
  • Al pulsar el botón:
  • …. (etc...)	

Obtención de **objetos** desde la HU:

Acción	**Nuevo objeto**	**Objeto(s) secundario(s)***
Agregar una nueva agenda de turnos	Agenda	- - -

Obtención de **objetos** desde los CA:

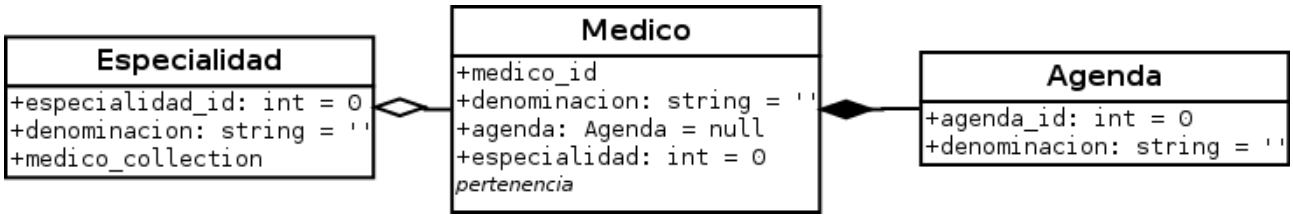
Campo	**Objeto derivado**
Especialidad*	Especialidad
Médico	Medico
Nombre	*(pertenece a Agenda, obtenido de la HU)*

(*) Un campo de tipo *select*, generalmente, denota una colección de objetos preexistentes (o *mockeados* -simulados-)

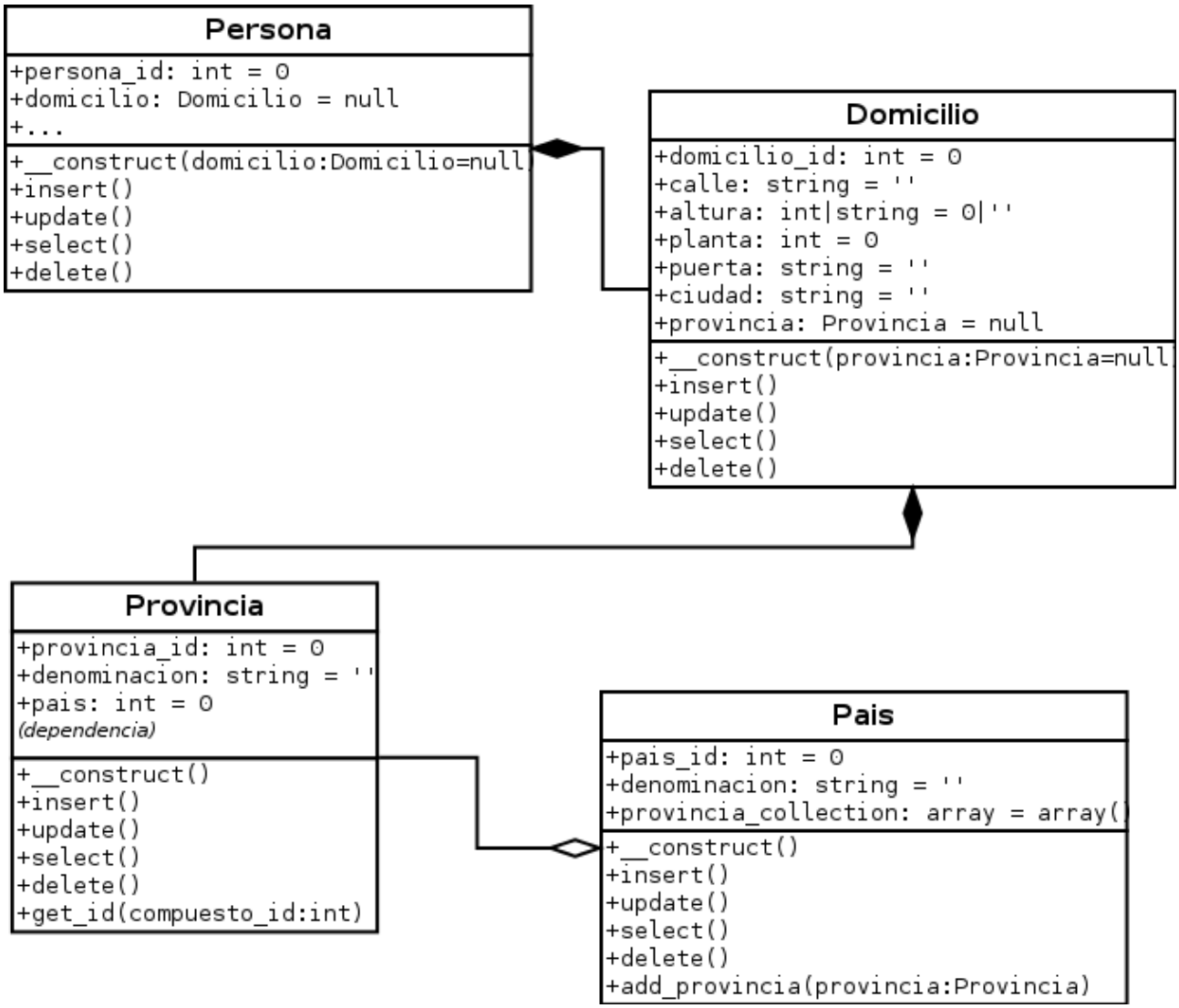
Análisis final de **propiedades*** para cada objeto, desde los CA:

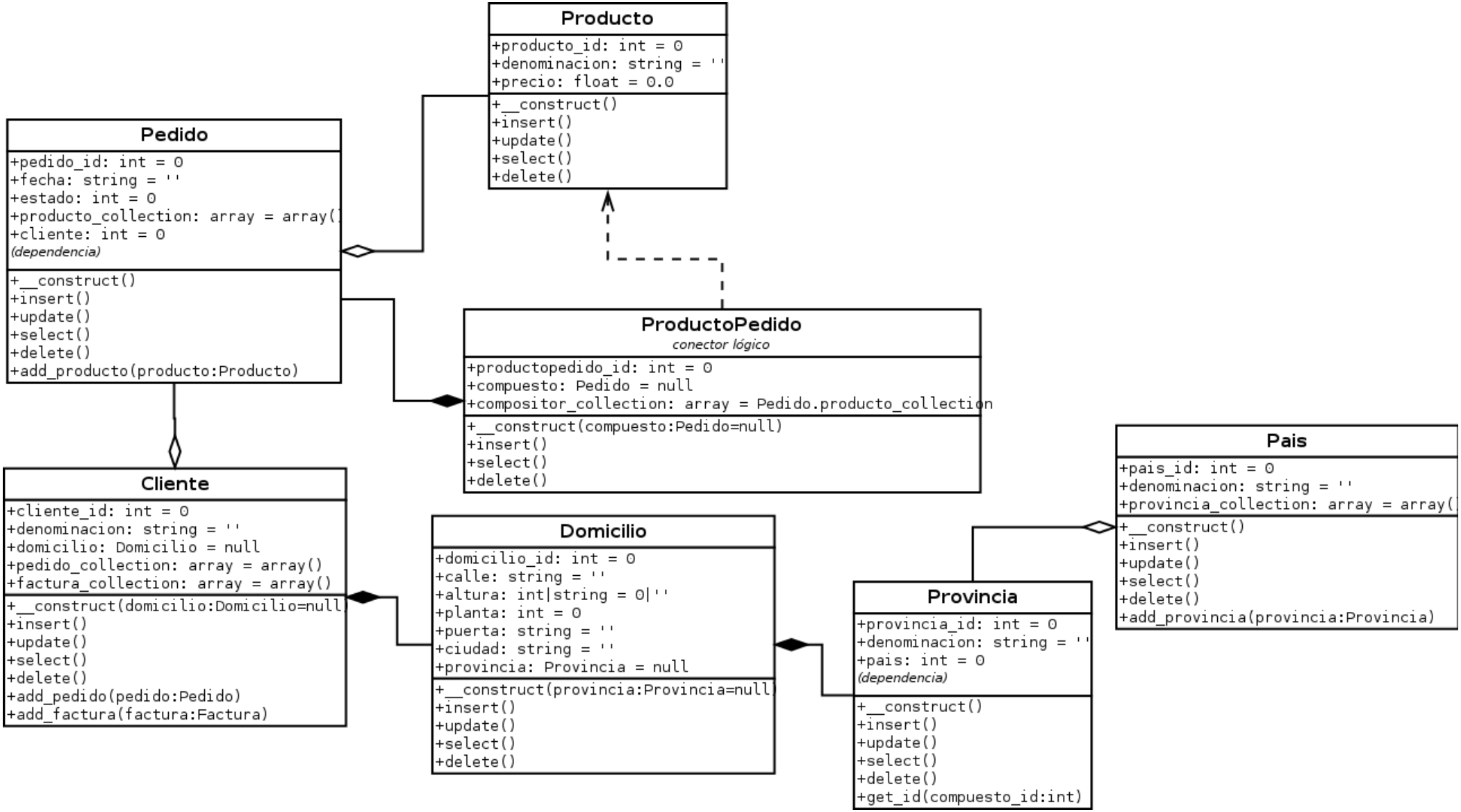
Campo	**Objeto derivado**	**Propiedad**		
		Simple	**Compuesta**	**Colectora**
Especialidad	Especialidad	especialidad_id denomiancion		medico_collection
Médico	Medico	medico_id denominacion *especialidad*	agenda	
Nombre	Agenda	agenda_id denominacion		

Relaciones obtenidas:



Ejemplo de objetos comunes en sistemas informáticos





Mapeo relacional de objetos

Tablas

Clase → nombre de tabla (en minúsculas)

Campos de tabla para objetos simples, compuestos y compositores

<table>
<tr><th>Propiedad</th><th>Campo</th><th colspan="2">Características especiales</th></tr>
<tr><td>Simple</td><td rowspan="5">SI</td><td colspan="2"></td></tr>
<tr><td>ID</td><td colspan="2">INT(11)
NOT NULL
AUTO_INCREMENT
PRIMARY KEY</td></tr>
<tr><td>Compuesta</td><td rowspan="2">INT(11)
FOREIGN KEY</td><td>DELETE → SET NULL</td></tr>
<tr><td>Pertenencia</td><td>DELETE → CASCADE
INDEX</td></tr>
<tr><td>Colectora</td><td>NO</td><td colspan="2"></td></tr>
</table>

Tabla para un conector lógico relacional

<table>
<tr><th>Campo</th><th colspan="2">Características especiales</th></tr>
<tr><td>{relacional}_id</td><td colspan="2">INT(11)
NOT NULL
AUTO_INCREMENT
PRIMARY KEY</td></tr>
<tr><td>compuesto</td><td rowspan="2">INT(11)
FOREIGN KEY
DELETE → CASCADE</td><td>INDEX</td></tr>
<tr><td>compositor</td><td></td></tr>
<tr><td>factor_multiplicacion</td><td colspan="2">INT
DEFAULT 1</td></tr>
</table>

ANOTACIONES