Webservices

Omar Vazquez Gonzalez

Modelo OSI

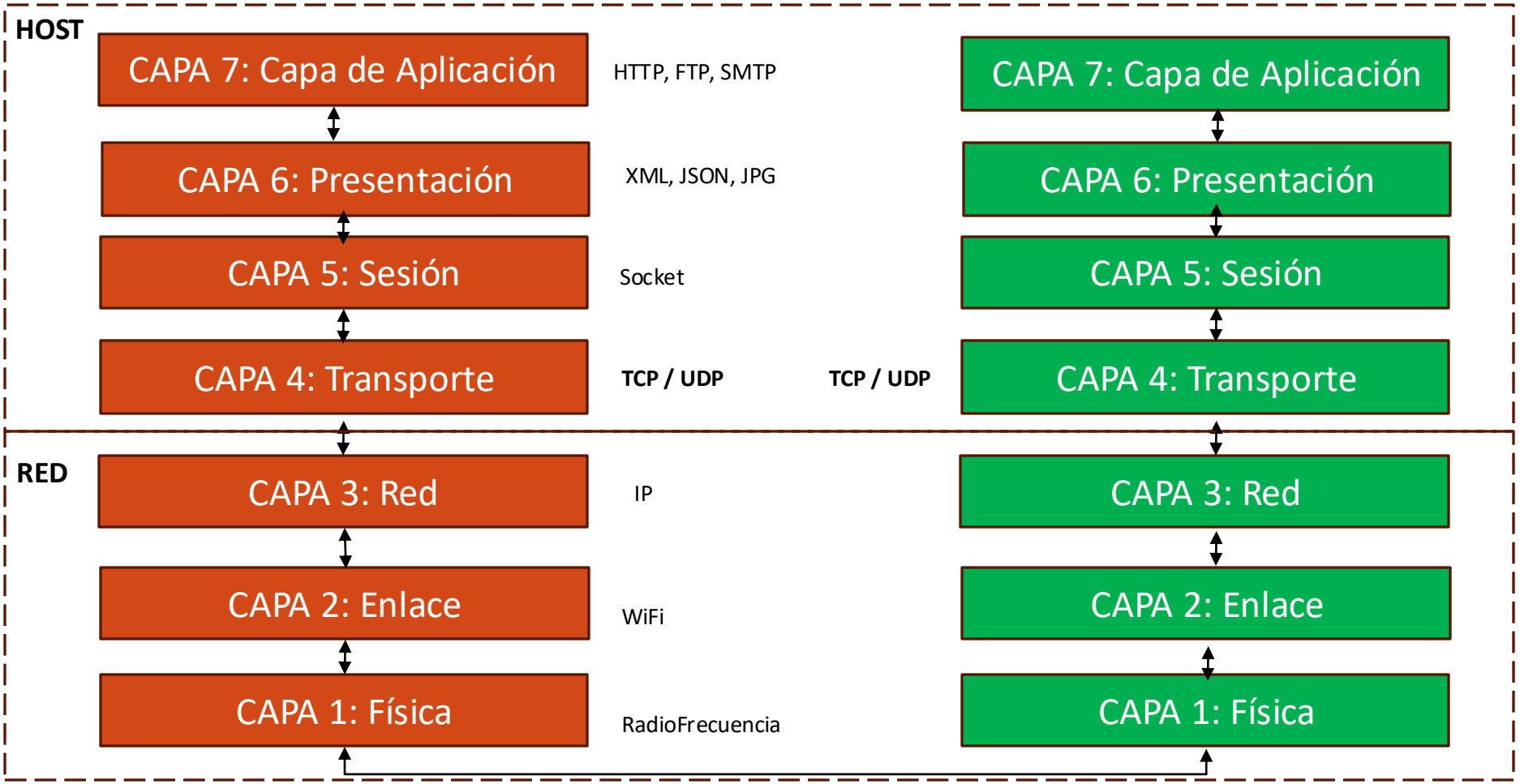
- Unifica las comunicaciones entre diferentes fabricantes de hardware
- Basado en 7 capas
- Útil para el diseño de redes y el diagnóstico de problemas de red

1 Capa física: fibra óptica, ondas

2 Capa de enlace de datos

3 Capa de red: Protocolo IP

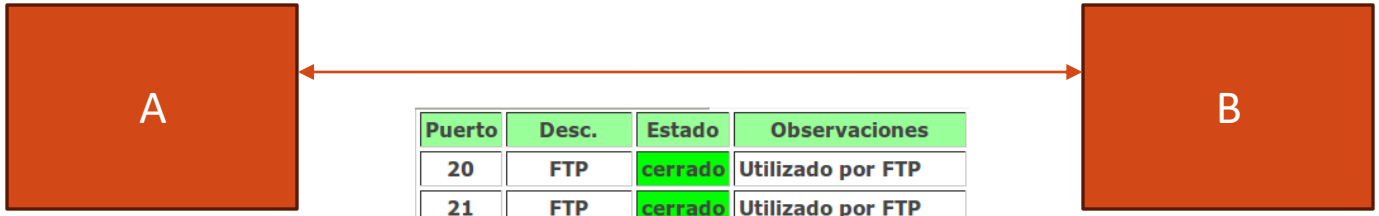
4 Capa de transporte: Protocolos TCP y UDP


HOST
CAPA 7: Capa de Aplicación
HTTP, FTP, SMTP
CAPA 7: Capa de Aplicación
CAPA 6: Presentación
XML, JSON, JPG
CAPA 6: Presentación
CAPA 5: Sesión
Socket
CAPA 5: Sesión
CAPA 4: Transporte
TCP / UDP
TCP / UDP
CAPA 4: Transporte
RED
CAPA 3: Red
IP
CAPA 3: Red
CAPA 2: Enlace
WiFi
CAPA 2: Enlace
CAPA 1: Física
RadioFrecuencia
CAPA 1: Física

Protocolo: **TCP**/UDP
Número de Puerto
Dirección IP

Cliente

Servidor



Puerto	Desc.	Estado	Observaciones
20	FTP	cerrado	Utilizado por FTP
21	FTP	cerrado	Utilizado por FTP
22	SSH	cerrado	Secure Shell.
23	TELNET	cerrado	Acceso remoto
25	SMTP	cerrado	Servidor de correo SMTP
53	DNS	cerrado	Servidor DNS
79	FINGER	cerrado	Servidor de información de usuarios de un PC
80	HTTP	cerrado	Servidor web
110	POP3	cerrado	Servidor de correo POP3
119	NNTP	cerrado	Servidor de noticias
135	DCOM-scm	cerrado	Solo se puede cerrar a través de un cortafuegos
			Compartición de

MySQL: 3306


Cliente
Servidor
A
B
Request
GET / HTTP/1.1
Host: localhost
VERBO HTTP – NOMBRE DEL RECURSO – HTTP/VERSION
Response
HTTP/1.1 200 OK
Content-Type: text/json
Content-Lenght: 31
{"nombre":"Enrique", "edad":20}
HTTP/VERSION – CÓDIGO – Descripción
HEADER: VALOR
HEADER: VALOR
CONTENIDO

VERBOS HTTP

HEAD	-	Solicitar el encabezado de un recurso (Saber que existe)
GET	-	Solicitar un recurso
POST	-	Enviar un recurso (Si no existe lo crea)
PUT	-	Enviar un recurso (Si existe lo actualiza)
PATCH	-	Enviar un recurso (Si existe lo actualiza)
DELETE	-	Eliminar un recurso

¿Qué puede ser ese recurso? = Archivo (HTML,XML, JSON, JPEG, TEXT/PLAIN, BINARIO)

HTTP Verb	Path	Action	Used for
GET	/photos	index	display a list of all photos
GET	/photos/new	new	return an HTML form for creating a new photo
POST	/photos	create	create a new photo
GET	/photos/:id	show	display a specific photo
GET	/photos/:id/edit	edit	return an HTML form for editing a photo
PATCH/PUT	/photos/:id	update	update a specific photo
DELETE	/photos/:id	destroy	delete a specific photo

HTTP Status Codes

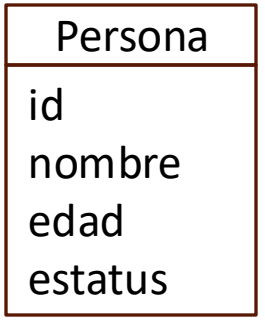
When a browser request a service from a web service, a response code will be given.
These are the list of HTTP Status code that might be returned

1XX Information	
100	Continue
101	Switching Protocols
102	Processing
103	Early Hints
2XX Success	
200	OK
201	Created
202	Accepted
203	Non-Authoritative Information
205	Reset Content
206	Partial Content
207	Multi-Status (WebDAV)
208	Already Reported (WebDAV)
226	IM Used (HTTP Delta Encoding)
3XX Redirection	
300	Multiple Choices
301	Moved Permanently
302	Found
303	See Other
304	Not Modified
305	Use Proxy
306	Unused
307	Temporary Redirect
308	Permanent Redirect
4XX Client Error	
400	Bad Request
401	Unauthorized
402	Payment Required
403	Forbidden
404	Not Found
405	Method Not Allowed
406	Not Acceptable
Compiled by Ivan Tay.	

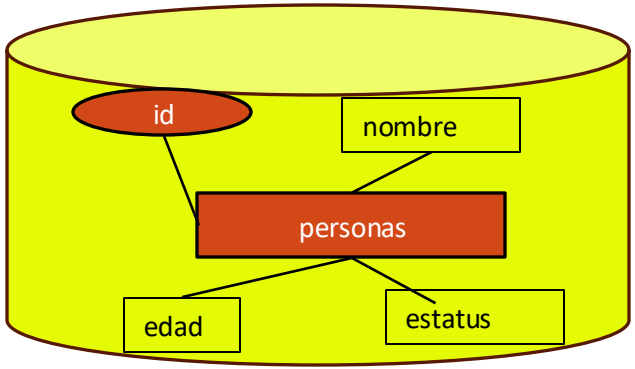
4XX Client (Continue)	
407	Proxy Authentication Required
408	Request Timeout
409	Conflict
410	Gone
411	Length Required
412	Precondition Failed
413	Payload Too Large
414	URI Too Large
415	Unsupported Media Type
416	Range Not Satisfiable
417	Exception Failed
418	I'm a teapot
421	Misdirected Request
422	Unprocessable Entity (WebDAV)
423	Locked (WebDAV)
424	Failed Dependency (WebDAV)
425	Too Early
426	Upgrade Required
428	Precondition Required
429	Too Many Requests
431	Request Header Fields Too Large
451	Unavailable for Legal Reasons
499	Client Closed Request
5XX Server Error Responses	
500	Internal Server Error
501	Not Implemented
502	Bad Gateway
503	Service Unavailable
504	Gateway Timeout
505	HTTP Version Not Supported
507	Insufficient Storage (WebDAV)
508	Loop Detected (WebDAV)
510	Not Extended
511	Network Authentication Required
599	Network Connect Timeout Error

XML (eXtensible Markup Languaje)

```
<Persona>
    <id>1</id>
    <nombre>Jesus</nombre>
    <edad>25</edad>
    <estatus>true</estatus>
</Persona>
```



Lenguaje POO



XML

```
<Persona>
    <id>1</id>
    <nombre>Jesus</nombre>
    <edad>25</edad>
    <estatus>true</estatus>
    <fecha>2023-02-28T13:21:35<fecha>
</Persona>
```

```
<Persona id="1" nombre="Jesus" edad="25" estatus="true" />
```

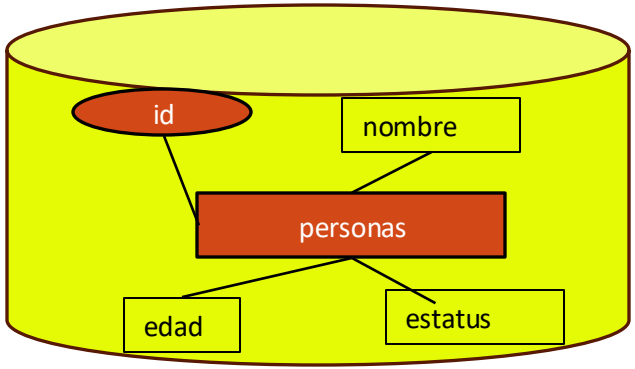
```
<Persona id="1" edad="25">
    <nombre>Jesus</nombre>
    <estatus>true</estatus>
</Persona>
```

JSON

{“id”: 1, “nombre”: ”Juan”, “edad”:25, “estatus”: true}

Persona
id
nombre
edad
estatus

Lenguaje POO



NOTA: JSON no tiene un formato estándar para las fechas

Una cadena JSON es al menos 33% mas corta que un XML

{“id”: 1, “nombre”: ”Juan”, “edad”:25, “estatus”: true}

<Persona><id>1</id><nombre>Jesus</nombre><edad>25</edad><estatus>true</estatus></P

[{“id”: 1, “nombre”: ”Juan”, “edad”:25, “estatus”: true},
{“id”: 2, “nombre”: ”Maria”, “edad”:20, “estatus”: true}]

<Personas>
<Persona><id>1</id><nombre>Jesus</nombre><edad>25</edad><estatus>true</estatus></Pe
<Persona><id>1</id><nombre>Jesus</nombre><edad>25</edad><estatus>true</estatus></Pe
</Personas>

EJERCICIO:

Implementar los siguientes métodos:

toJson()

toXml()

fromJson(String)

fromXml(String)

```
<Persona>
    <id>1</id>
    <nombre>Jesus</nombre>
    <edad>25</edad>
    <estatus>true</estatus>
</Persona>
```

{“id”: 1, “nombre”: ”Juan”, “edad”:25, “estatus”: true}

REVISION 12:15

REST SERVICE
(MAPEO VERBOS HTTP-CRUD BASE DE DATOS)

GET	-	Consulta
POST	-	Inserción de un dato (Creación)
PUT	-	Actualización
DELETE	-	Eliminar

REST SERVICE vs SOAP WEB SERVICE

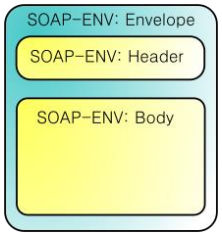
Verbos: GET, POST, PUT, DELETE

Formato: JSON, XML (plain ó POX)

Representational State Transfer

Verbo: POST

Formato: XML (SOAP envelope)



XSD – XSDL - WSDL

Simple Object Access Protocol

SOAP ENVELPE (XML)

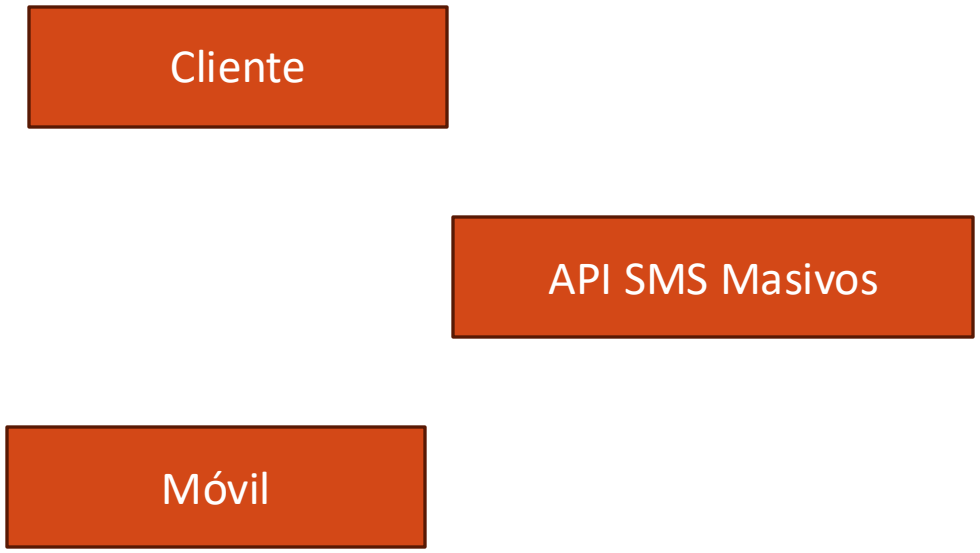
```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:tem="http://tempuri.org/">
   <soapenv:Header/>
   <soapenv:Body>

      <tem:sumar>
         <!--Optional:-->
         <tem:x>?</tem:x>
         <!--Optional:-->
         <tem:y>?</tem:y>
      </tem:sumar>

   </soapenv:Body>
</soapenv:Envelope>
```

RESPONSE

<s:Envelope
xmlns:s="http://schemas.xmlsoap.org/soap/envelope/"><s:Body><sumarResponse
xmlns="http://tempuri.org/">
<sumarResult>15</sumarResult>
</sumarResponse></s:Body></s:Envelope>



a) Aplicación Cliente va a solicitar su nombre y número de teléfono
b) Generar un número aleatorio de 4 dígitos (No se muestra en pantalla solo se envía
c) Va a enviar al código de 4 dígitos a la API SMS
d) El usuario recibe el código en su celular.
e) El usuario ingresa el código en la aplicación. Si es válido, muestra un mensaje de éxito.

Ejercicio

Crear los métodos para el webservice de:

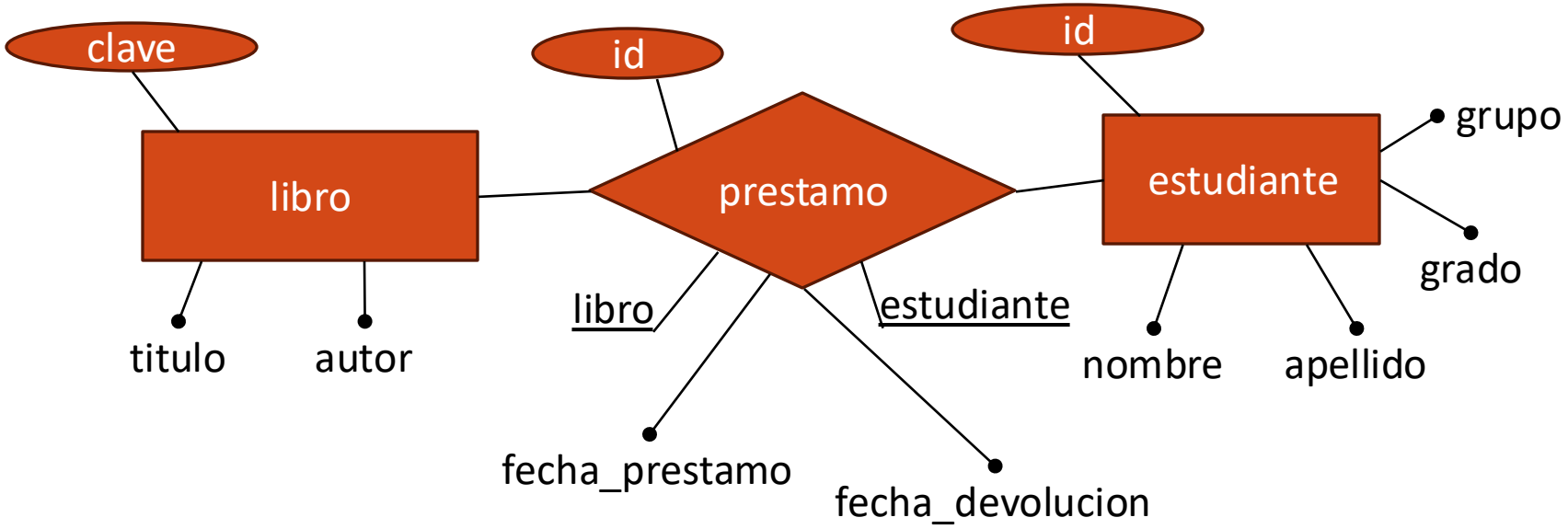
Estudiante

Préstamo

Sobre el WebController de Biblioteca

Crear las clases y dependencias necesarias.

DIAGRAMA ENTIDAD RELACIÓN



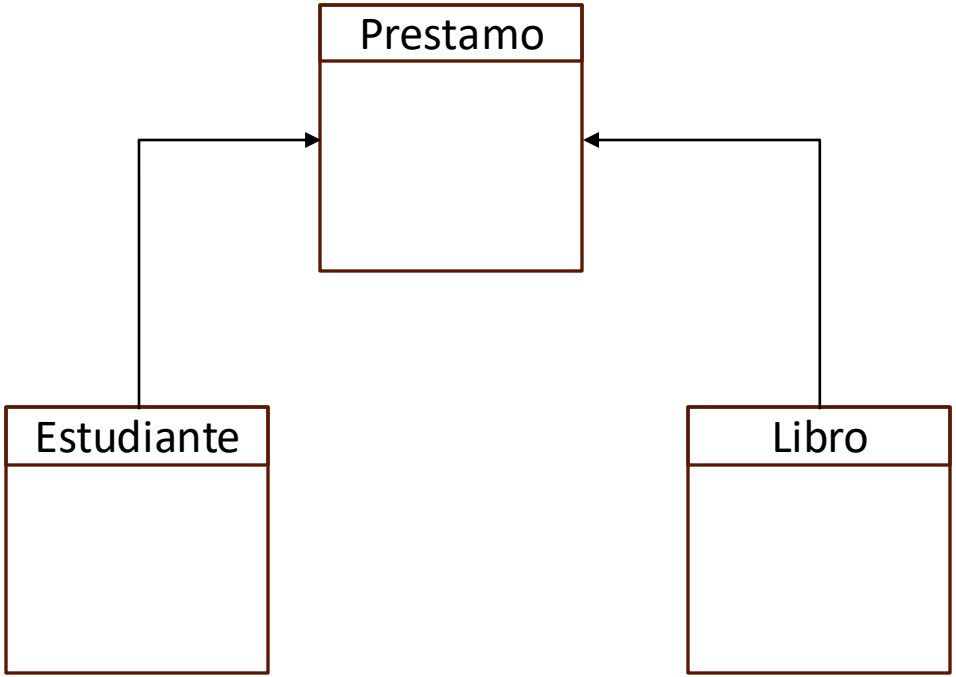
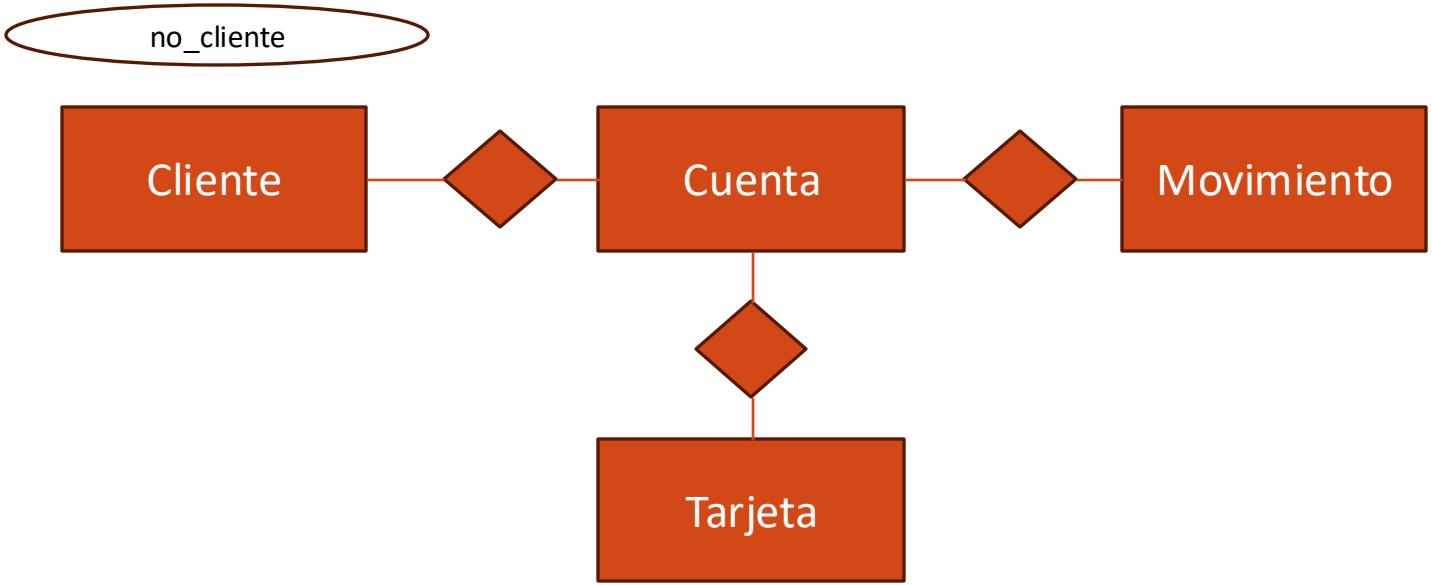
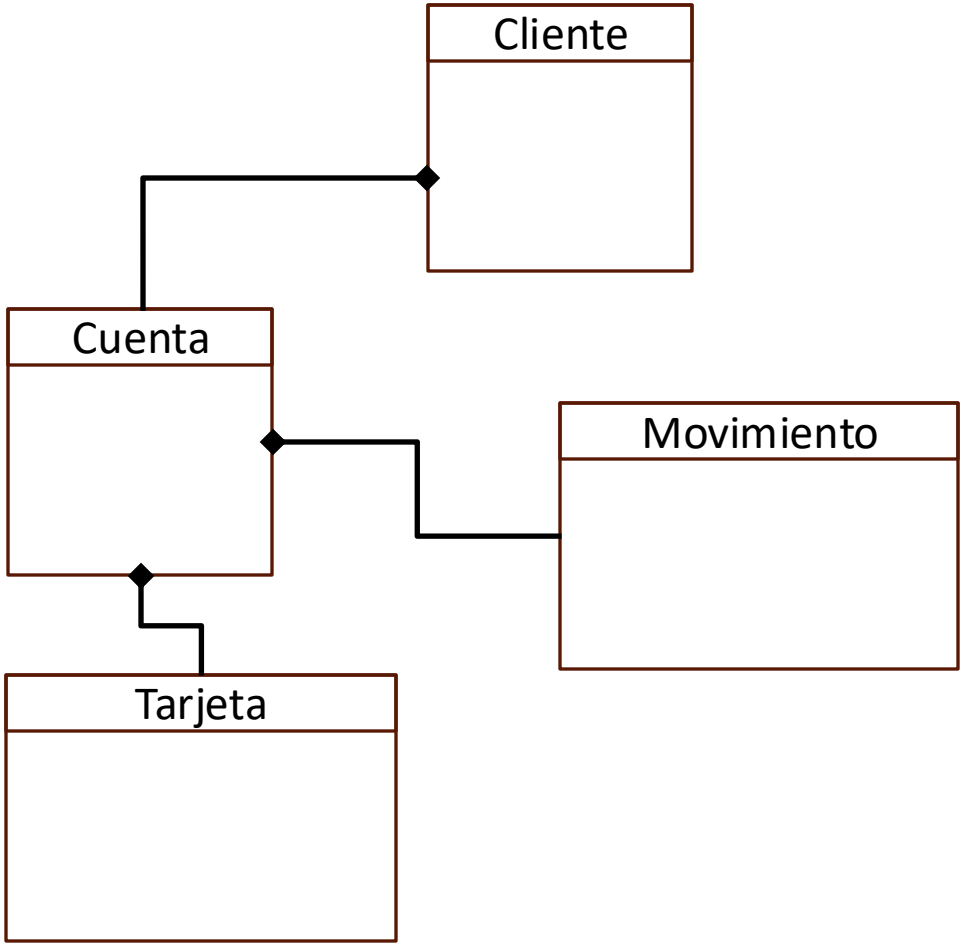

Prestamo
Estudiante
Libro

Diagrama Entidad-Relación




Cliente
Cuenta
Movimiento
Tarjeta