



Dirección de
Innovación y
Desarrollo
Docente
Universidad Andrés Bello

Evolución de la Web

*Basado en apuntes del profesor Dr. José Emilio Labra,
Universidad de Oviedo, España.*

Richard Rodríguez Rivas (richard.rodriguez.r@gmail.com)

Magister en Tecnologías de la Información

- Ingeniero Civil Informático



FORMAR

01/2016

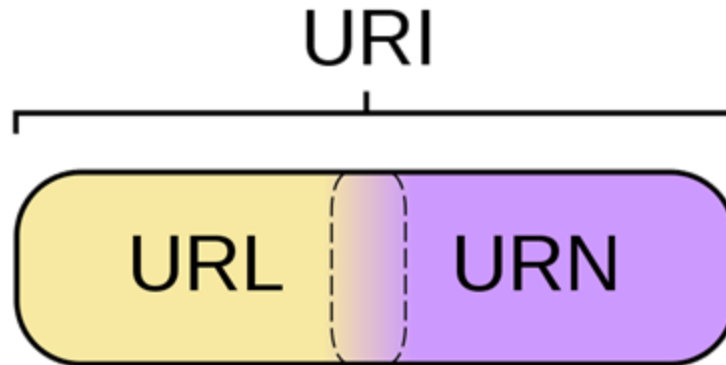
TRANSFORMAR



UNIVERSIDAD
ANDRÉS BELLO

URIs

URIs



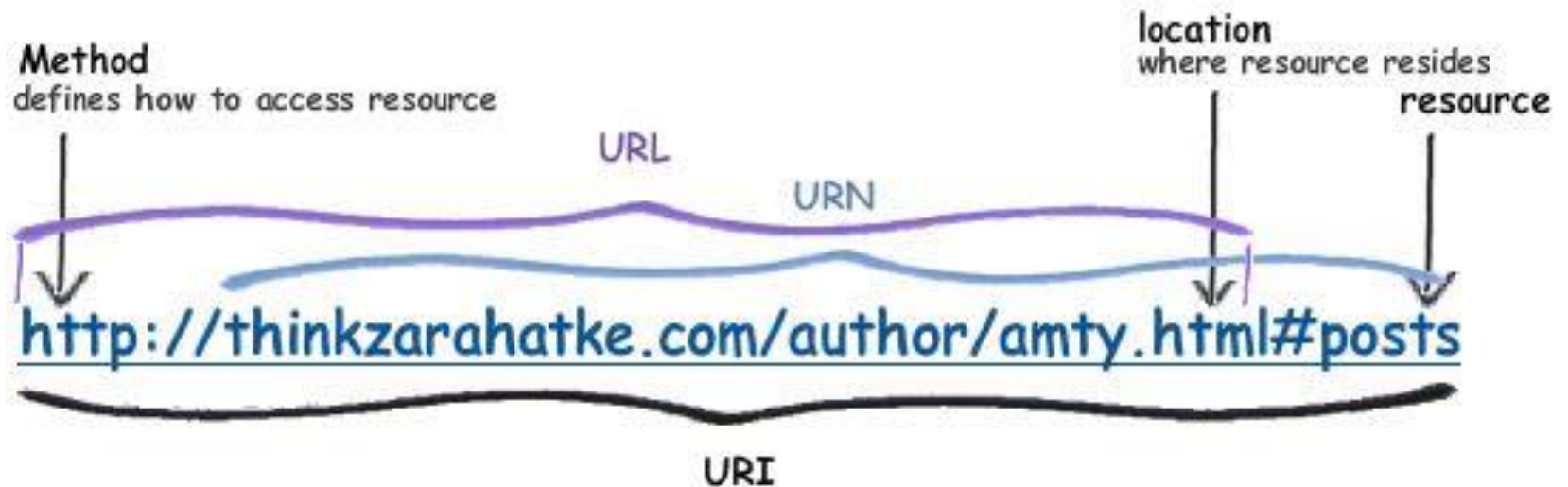
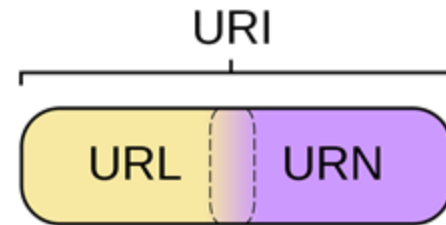
Las definiciones son:

URI Uniform Resource Identifier (Identificador Uniforme de Recursos)

URL Uniform Resource Locator (Localizador Uniforme de Recursos)

URN Uniform Resource Name (Nombre Uniforme de Recursos)

URIs



Recursos

Uniform Resource Identifier

Recurso = Unidad básica de la Web

Cualquier cosa que se identifique con una URI

URI \neq Recurso \neq Representación

URI

<http://tiempo.com/Asturias/Oviedo>

Identifica

Representa

Tiempo en Oviedo



Recurso

Representación

Metadatos:

Content-type: text/html

Datos:

<html>

<head><title>Tiempo</title></head>

<body>

<h1>Tiempo en Oviedo</h1>

<p>Nubes y claros</p>

</body>

</html>

FORMAR

TRANSFORMAR



UNIVERSIDAD
ANDRÉS BELLO

¿Qué se puede identificar con una URI?

Cualquier cosa concreta o abstracta

Ejemplo: Una página Web

<http://www.uniovi.es>

identifica

```
<!DOCTYPE html>
<html>
  <head>
    <title>Universidad de Oviedo</title>
  </head>
  <body>
    <h1>Universidad de Oviedo</h1>
    <p>Fundada en el año 1608 en
      <a href="http://www.wikipedia.org/Oviedo">
        Oviedo</a></p>
    . . .
  </body>
</html>
```

Una página Web
Recurso de información
Formato HTML

¿Qué se puede identificar con una URI?

Cualquier cosa concreta o abstracta

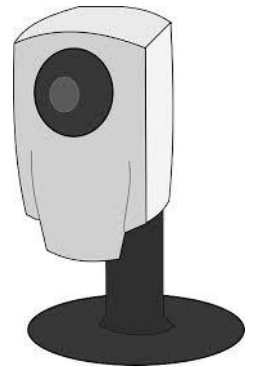
Ejemplo: Una fotografía (recurso multimedia)

<http://www.di.uniovi.es/~labra/images/asturias.jpg>

identifica



Una fotografía
Recurso de información
Formato JPG



¿Qué se puede identificar con una URI?

Cualquier cosa concreta o abstracta

Ejemplo: Una persona

`http://www.w3.org/People/Berners-Lee/card#i`

identifica



Una persona (Tim Berners-Lee)
Recurso de no información

¿Qué se puede identificar con una URI?

Cualquier cosa concreta o abstracta

Ejemplo: Conjunto de todas las personas

`http://xmlns.com/foaf/0.1/Person`

identifica



Conjunto de Personas (concepto abstracto)
Recurso de no información

¿Qué se puede identificar con una URI?

Cualquier cosa concreta o abstracta

Ejemplo: Propiedad de creación

<http://purl.org/dc/terms/creator>

identifica

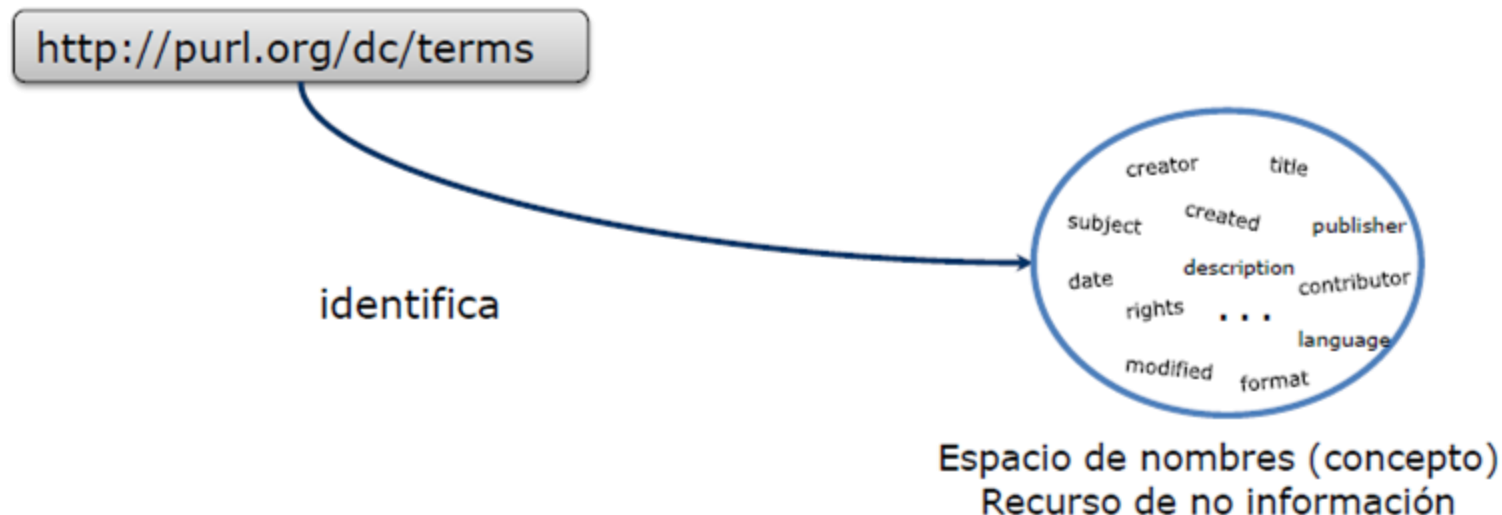


Propiedad de creación (concepto abstracto)
Recurso de no información

¿Qué se puede identificar con una URI?

Cualquier cosa concreta o abstracta

Ejemplo: Espacio de nombres



Formato de una URI

esquema : // autoridad camino ?consulta#fragmento

<http://ejemplo.com:8042/libros/castellano?autor=Cervantes#capitulo2>

Nota: los caracteres deben codificarse. Significado especial de espacios, ?, /, etc.

Otros ejemplos de URIs:

<ftp://ftp.is.co.za/rfc/rfc1808.txt>

<mailto:pepe@ejemplo.com>

<telnet://192.0.2.16:80/>

<urn:oasis:names:specification:docbook:dtd:xml:4.1.2>

Nota: las URNs identifican nombres únicos solamente. Sin protocolo

Más información: Especificación
<http://tools.ietf.org/html/rfc3986>

Dereferenciación

Dereferenciar una URI = Acceder al contenido de una URI

Obtener una representación del recurso identificado por la URI

Habitualmente se utiliza protocolo HTTP

Pueden existir diferentes representaciones

La representación puede incluir enlaces a otras URIs con información relacionada

Principio: Follow your nose (*"Sigue tu instinto"*)

A partir de una URI, se puede ir encontrando más información y más recursos relacionados fácilmente y de casualidad (serendipia)

Estabilidad de las URIs

URIs = pilar fundamental de cualquier aplicación Web

Objetivo: Esquema de URIs estable

Lema: *Cool URIs don't change*

Modificar una URI puede romper aplicaciones existentes

Evitar URIs que dependen de detalles de implementación

Ejemplo: <http://156.35.41.34:8080/pagina.php>

Recomendaciones:

Una URI genérica + 1 URI para cada representación

Ejemplo:

<http://periodico.com/noticias/101> - URI genérica para la noticia 101

<http://periodico.com/noticias/101.en> - URI para la noticia en inglés

<http://periodico.com/noticias/101.es> - URI para la noticia en español

Importancia de nombres adecuados para URIs

<http://www.w3.org/Provider/Style/URI>

Formatos de Representación

FORMAR

TRANSFORMAR

Formatos de representación

En la Web, el formato más habitual es HTML

Existen muchos más formatos: XML, JSON, RDF, PNG, ...

Un recurso puede tener diferentes tipos de representación

Cada tipo de representación sirve para un propósito



FORMAR



TRANSFORMAR

HTML

Tipo de representación más popular en la Web

Objetivo: representar hipertexto

Ejemplo:

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>Ejemplo</title>
  </head>
  <body>
    <h1>Lista de enlaces</h1>
    <p>Mis enlaces preferidos</p>
    <ul>
      <li><a href="http://www.wikipedia.org">Wikipedia</a>
      <li><a href="http://www.w3c.org">Consortio W3c</a>
    </ul>
  </body>
</html>
```

XML

Facilita intercambio de información

Objetivo: procesamiento automático

Comercio electrónico

```
<?xml version="1.0">
<pedido>
  <producto codigo="R23">
    <nombre>Rotulador RX2</nombre>
    <cantidad>20</cantidad>
    <comentarios>Comprobad que escriben</comentarios>
  </producto>
  <producto codigo="G56">
    <nombre>Grapadora Lin</nombre>
    <cantidad>2</cantidad>
    <comentarios>Envuelta para regalo</comentarios>
  </producto>
</pedido>
```

Tipos de representación

Los tipos de representación se identifican con MIME
MIME (Multipurpose Internet Mail Extensions)

Identificar el tipo de contenido (Cabecera Content-type)

Formato tipo/subtipo

Ejemplos:

text/html: Página Web en formato HTML

text/xml, application/xml : Documento XML

application/json: Documento JSON

application/pdf: Fichero PDF

image/jpeg: Imagen JPEG

application/xhtml+xml: Documento XHTML

application/rdf+xml: Documento RDF

text/turtle: Documento Turtle

...

Lista oficial: <http://www.iana.org/assignments/media-types>

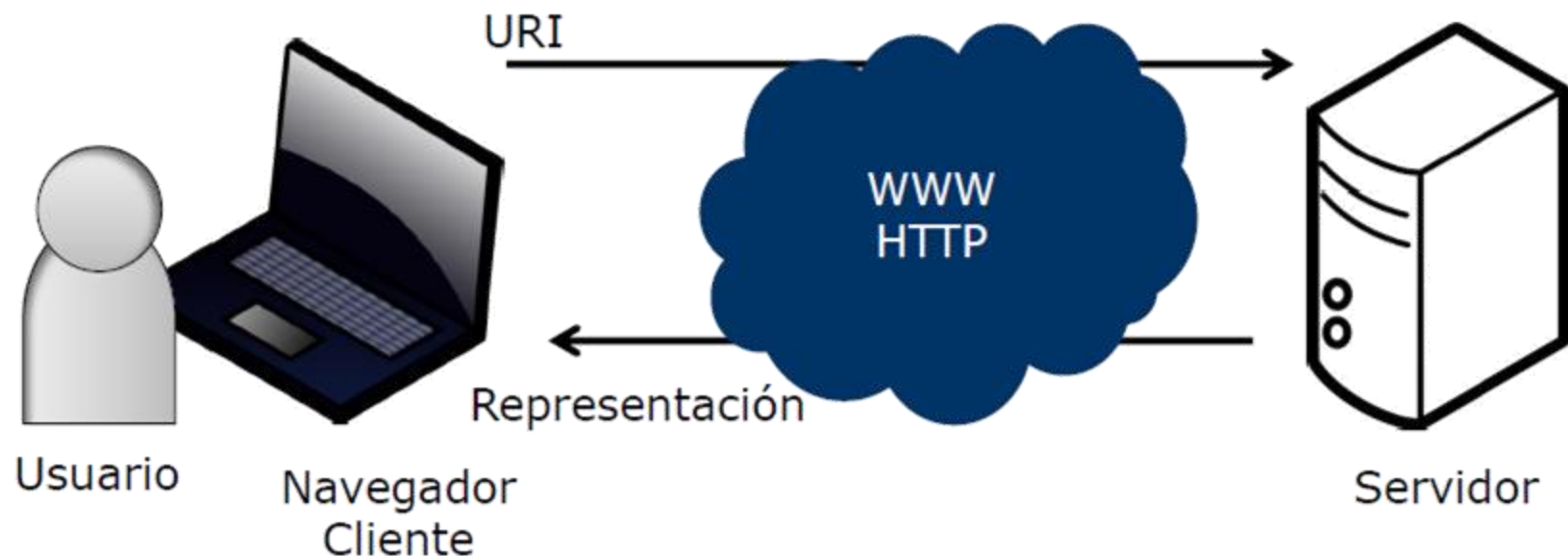
Funcionamiento de la Web

2 computadores conceptuales: Cliente y Servidor

La representación puede calcularse dinámicamente

Computación en Cliente

Computación en servidor



Cliente

También se conoce como Agente de Usuario

Normalmente es un navegador (browser)

Múltiples tipos de agentes de usuarios y navegadores

- Navegadores: Internet Explorer, Chrome, Firefox, Lynx, ...

- Dispositivos móviles

- Lectores de pantalla

- eBooks

- TVs

- ...

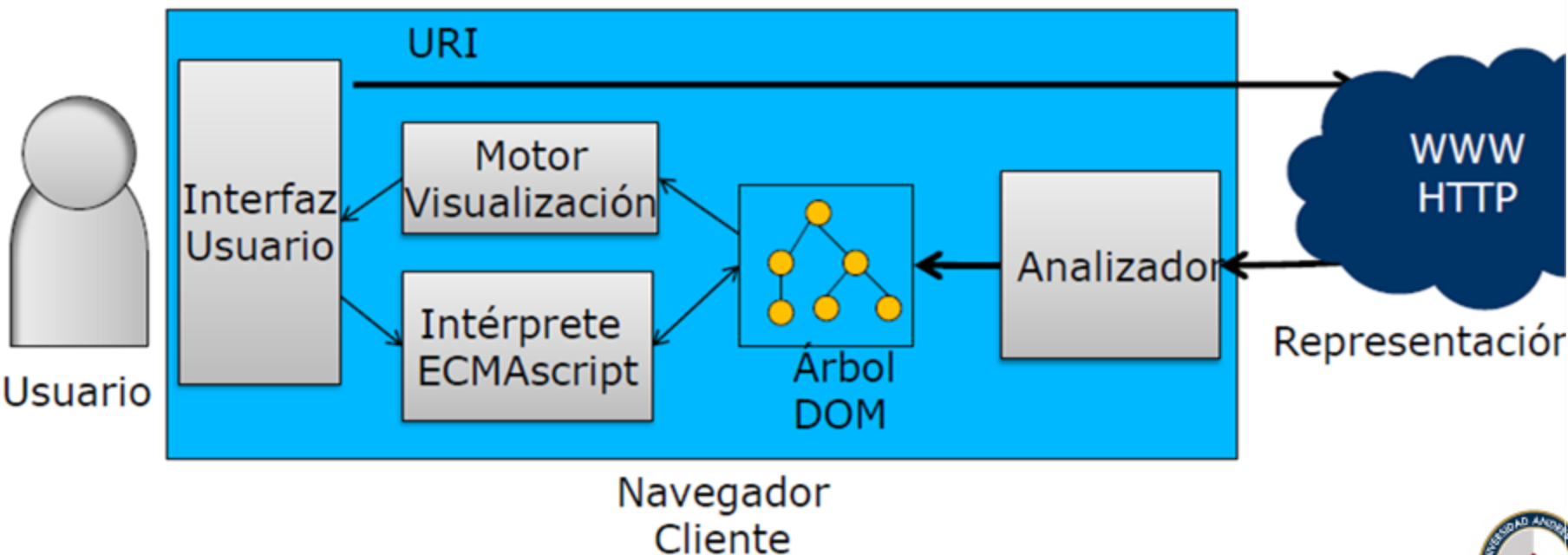
Componentes de un navegador

Interfaz

Analizador






Motor visualización

Intérprete ECMAScript procesa eventos y modifica árbol



Motor de visualización

A veces los navegadores comparten el mismo motor de visualización (rendering engine)

Navegadores		Motor de visualización
Internet Explorer		Trident
Firefox		Gecko
Opera		Presto
Chrome		Webkit (Webcore)
Safari (iPhone, iPad)		Webkit

ECMAScript

Lenguaje interpretado basado en prototipos

Origen: Brendan Eich, Netscape (1995)

ECMAScript = estándar con dialectos Javascript, Jscript...






Permite la interacción entre el usuario del navegador y el árbol DOM

Los navegadores utilizan APIs para crear objetos que pueden manipular el árbol DOM

Intérprete de ECMAScript

Competición entre intérpretes

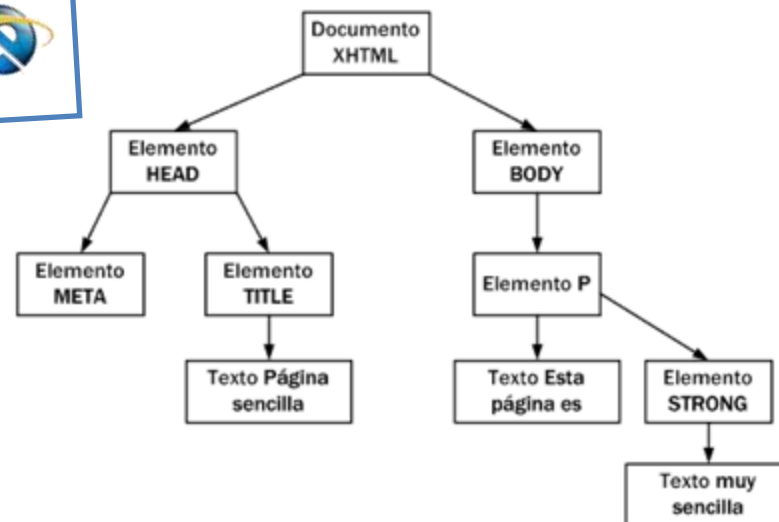
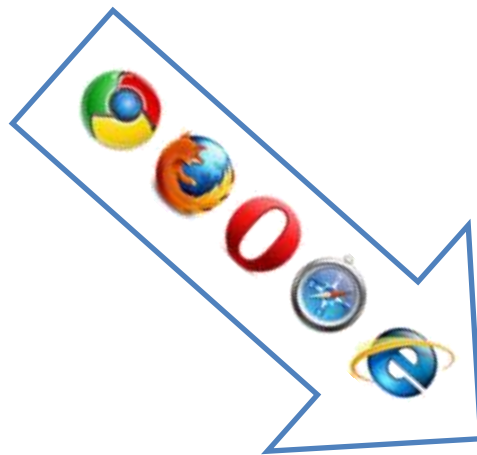
Quién
entiende a JS

Navegadores	Lenguaje	Implementación
Internet Explorer	 JScript	Chakra
Firefox	 Javascript	Rhino Tracemonkey IonMonkey
Chrome,	 Javascript	V8
Safari (iPhone, iPad)	 Javascript	Squirrelfish (Nitro)
Opera	 Javascript	Carakan

Document Object Model

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
    <title>Página sencilla</title>
  </head>

  <body>
    <p>Esta página es <strong>muy sencilla</strong></p>
  </body>
</html>
```



Componentes de un Servidor

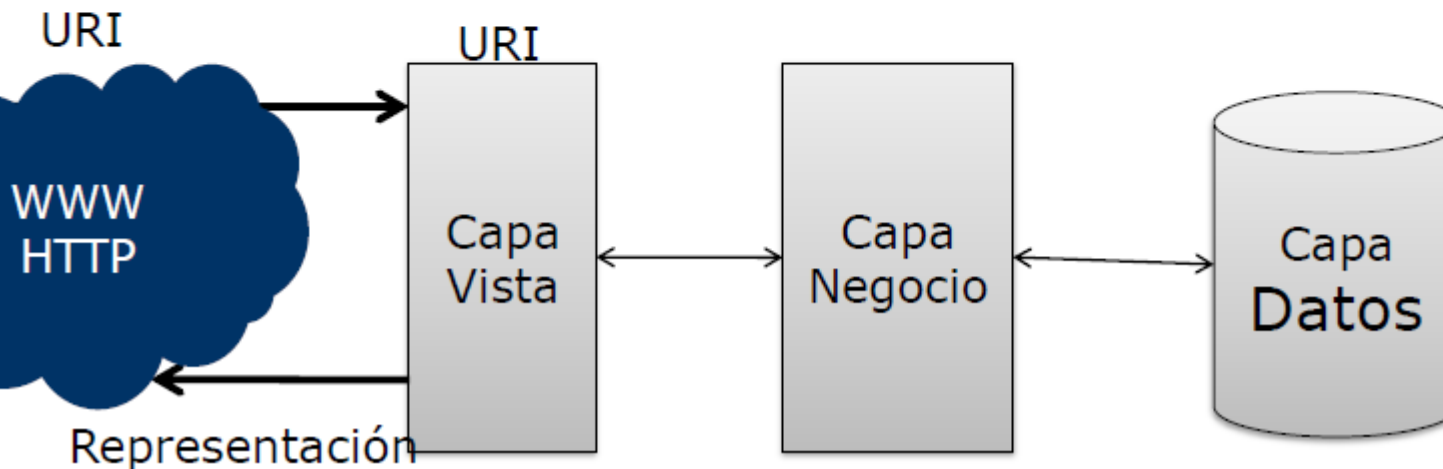
Capas

La arquitectura del servidor suele descomponerse en varias capas

Vista: Se encarga de preparar la representación

Negocio: Gestión de objetos de negocio

Datos: Modelos de datos



FORMAR

TRANSFORMAR

Tecnologías del lado servidor

Múltiples frameworks y lenguajes

Java: J2EE, Spring,...

Ruby: Ruby on Rails, Sinatra, Padrino...

Python: Django,...

Scala: Lift,...

PHP: Zend... Symfony, Cake, ...

Modelo de datos

Bases de datos relacionales

Modelos NoSQL

Modelos RDF



"That's all Folks!"