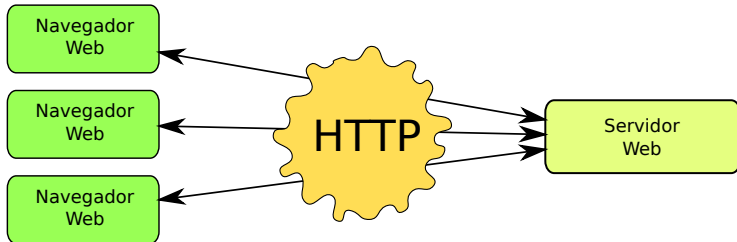


# Aplicaciones Web Desconectadas

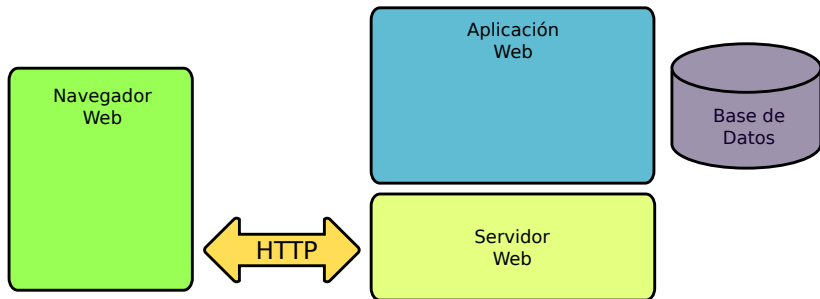
Defossé Nahuel, van Haaster Diego Marcos

15 de Diciembre de 2009

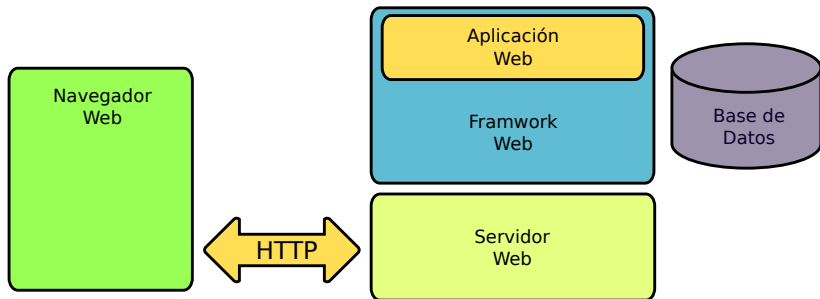
# ¿Qué es Web?



# ¿Qué es una aplicación Web?



# ¿Qué es una aplicación Web Hoy?



# Objetivos Principales

## Objetivo Principal

“Extender un framework de aplicaciones web existente, **Open Source**, de manera que una aplicación realizada sobre éste pueda ser ejecutada en el cliente de manera desconectada con un mínimo de modificaciones. Para permitir que la aplicación pueda ejecutarse en el cliente, se implementará”

- Persistencia del modelo de datos en el cliente.
- Subconjunto de acciones disponibles en modo desconectado
- Primitivas de sincronización entre la aplicación del cliente y la aplicación web que le dio origen

# Objetivos Secundarios

- **Open Source**

Coste de licenciamiento nulo y aseguramiento de la continuidad

- **Multiplataforma**

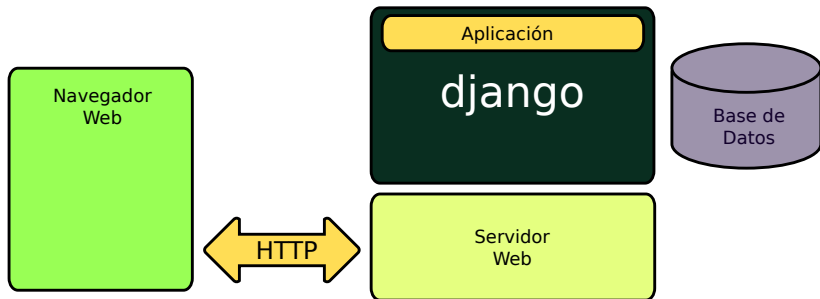
Windows, Linux, Mac y móviles (donde exista un browser)

- **Adaptación mínima de aplicaciones existentes**

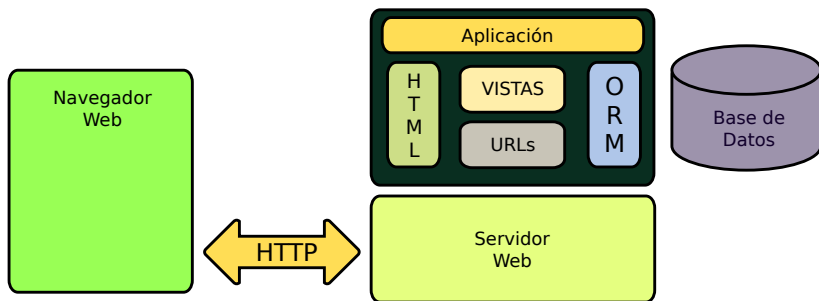
Integración con un Frameworks Web

Reutilizar los conceptos/patrones del framework para una rápida asimilación de los desarrolladores.

# Framework Web - Django



# Framework Web - Django - Componentes





# Carencias de los navegadores

- **Servidor Web**

Sin servidor, el navegador solo posee la **cache**

- **Base de Datos**

Un navegador no posee un mecanismo de almacenamiento

- **Lenguaje de Programación Consistente**

Cada navegador implementa a su manera **JavaScript** y **DOM**

- **Concurrencia**

Los eventos son atendidos en el bucle principal

- **Conectividad con el entorno del cliente**

Escritorio  $\neq$  Espacio de URLs

# Principales alternativas

- Microsoft Silverlight
- Sun JavaFX
- Adobe AIR
- Mozilla XUL



Microsoft®  
**Silverlight™**

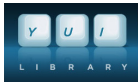


ADOBE® AIR™

# JavaScript

Un lenguaje con *mala* reputación con:

- Objetos
- Patrones Propios (Closures, Module, etc.)
- Muchas librerías



# JavaScript - Según Mozilla

- Generadores e Iteradores
- Varias ayudas para una mejor programación OO
- Sabor Pythonico :)



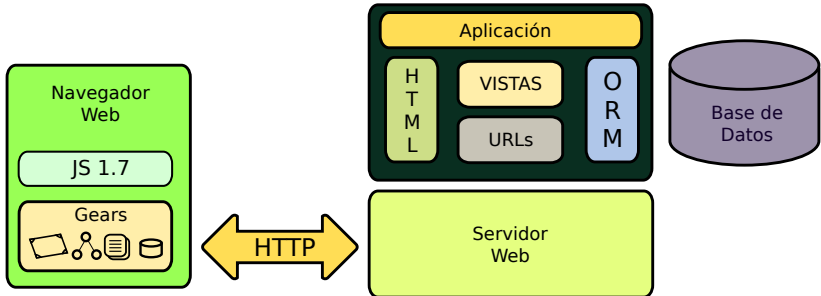
# Google Gears

Plugin para los navegadores

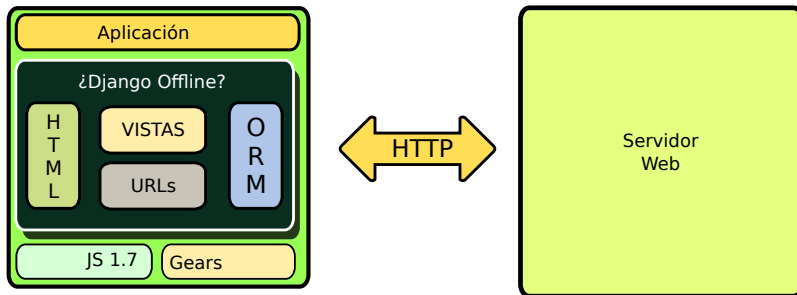
- **Local Server**
- **Data Base**
- **Worker Pool**
- **Desktop**



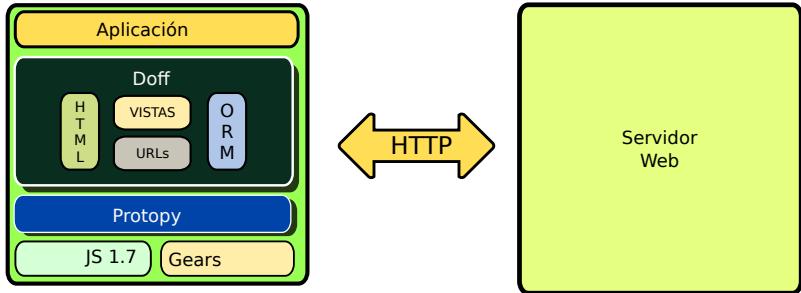
# Gears



# ¿Django Desconectado?



# Doff y Protopy





# Protopy - Lobera de JavaScript

## Objetivos:

- Interfase Pythonica para desarrolladores
- Promover la reutilización de código
- Control de la página y eventos(DOM) - Trabajo cotidiano del desarrollador web de *front-ends*
- Soporte de el framework desconectado.
- Interfase con las tecnologías de persistencia y ejecución offline.

# Componentes de Protopy

- Módulos
- Clases
- DOM
- Eventos
- AJAX



# Componentes de Protopy

Doff (Django sobre Protopy)

Módulos distribuidos con Protopy en "packages"

logger

copy

gears

rpc

urls

json

Protopy

modulos

paquetes

types

builtin

sys

ajax

dom

exceptions

event

# Doff - Objetivos

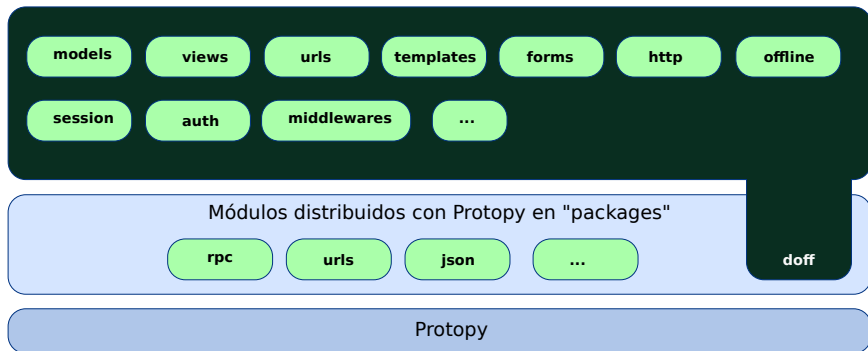
## Objetivos:

- Facilitar el desarrollo desconectado
- Reutilización de recursos del proyecto en línea
- Emulación de HTTP
- Control de URLs e historial del navegador
- Persistencia de la aplicación, datos y *bootstrapping*
- Sincronización

# Doff - Componente

- **API de Modelos**
- **Templates**
- **Formularios**
- **URLs**
- **Vistas**
- **Proyecto**
- **Aplicaciones adicionales**
- Sincronización, Autenticación, Sesión

# Estructura de Doff



# Proyecto Desconectado

- El desarrollador debe reescribir las vistas en JavaScript y URL en JavaScript.
- El desarrollador puede reutilizar los plantillas de Django
- El desarrollador **debe reescribir** los modelos nuevamente JavaScript

## Offline - Objetivos

- Automatización de creación de proyectos desconectados
- Servidor estático del código de Protopy y Doff
- Servidor estático del código de cada proyecto

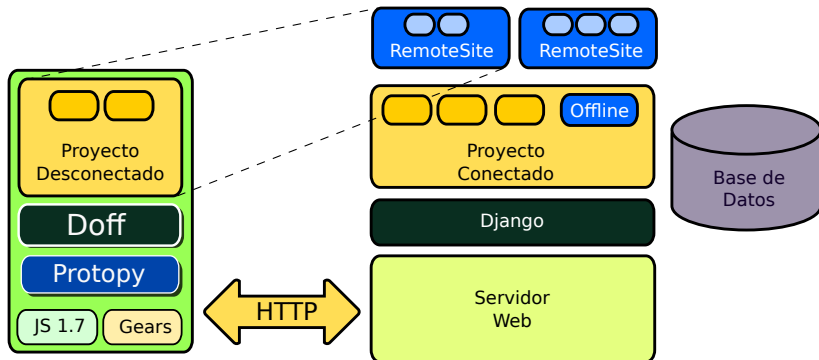


# Offline - RemoteSites

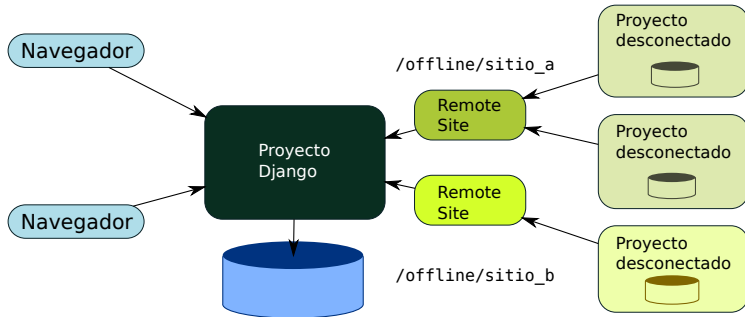
**Objetivo** Definición de desconectado en el cliente. Cada sitio remoto es una *vista* de el proyecto y se publica en una *URL*

- Conversión de modelos de Python a JS mediante *introspección*
- Definición de acceso a datos del servidor
- Acceso a filas (*managers*)
- Acceso a columnas (*modificación de modelos*)
- Publicación de plantillas

# Offline - RemoteSites estructura



# Esquema



# Objetivo

Entidades definidas en el servidor se repliquen en el cliente y que las creadas y modificadas por el cliente se transfieran al servidor.

- Integridad de la sincronización
- Transporte de datos
- Detección de cambios en instancias en el servidor
- Detección de cambios en el cliente

# Primitivas

- **PUSH**
- **PULL**
- **UPDATE**
- **PURGE**

# Conflictos

- **PUSH**
- **PULL**
- **UPDATE**
- **PURGE**

# Conclusiones

- **Extender un framework**
- **Persistencia del modelo de datos en el cliente**
- **Acciones disponibles en modo desconectado**
- **Primitivas de sincronización**
- **Desarrollo de software libre** <http://code.google.com/p/protopy>

## Lineas Futuras

- **Conversión de Código Python en JavaScript**
- **Sitio de Administración**
- **Workers con Soporte para JavaScript 1.7**
- **Compatibilidad con ES5 y HTML5**
- **Optimizaciones en Base a Permanencia de Estado**
- **Implementación de Storage o Almacenamiento en el Cliente**
- **Compilación de JavaScript**
- **Manejo de Migraciones de Esquema**



## Miscelánea

- **Firefox** Plataforma
- **Firebug** Depuración
- **Django** Framework Web
- **Python** Lenguaje Server Side, Scripting, Sphinx Hacking, etc
- **Mercurial** Control de versiones
- **Sphinx** Para crear la documentación,  $\text{\LaTeX}$  y HTML
- **$\text{\LaTeX}$  - beamer** Para crear *esta* presentación