UNIMORE
UNIVERSITÀ DEGLI STUDI DI
MODENA E REGGIO EMILIA

Evoluzione del Web

Tecnologie Web

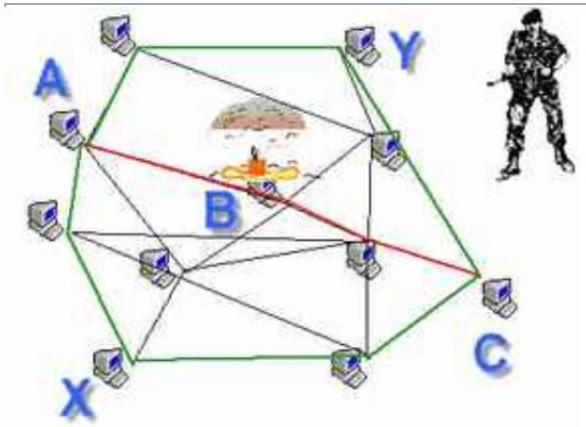
Claudia Canali

Internet e il Web

Differenze?

Internet

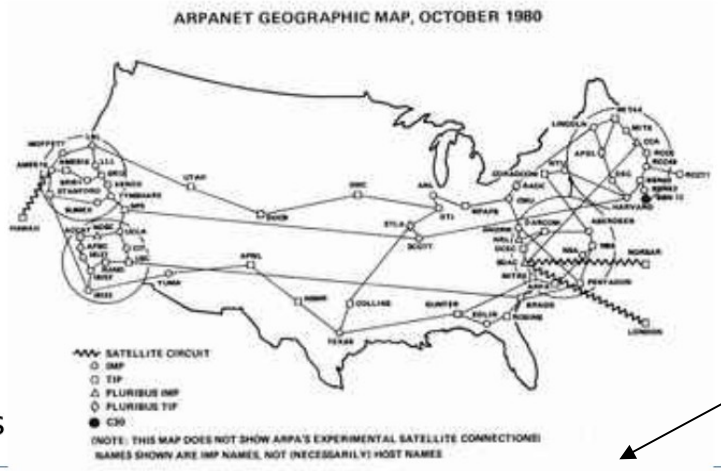
Origine del progetto: ARPA (Advanced Research Projects Agency)
creata dal Dipartimento Difesa Stati Uniti nel 1967



Il progetto iniziale
ARPANET (1969)

http://www.vox.com/a/internet-maps

Connessioni nel
1969



Connessioni nel 1980

1972: 15 nodi

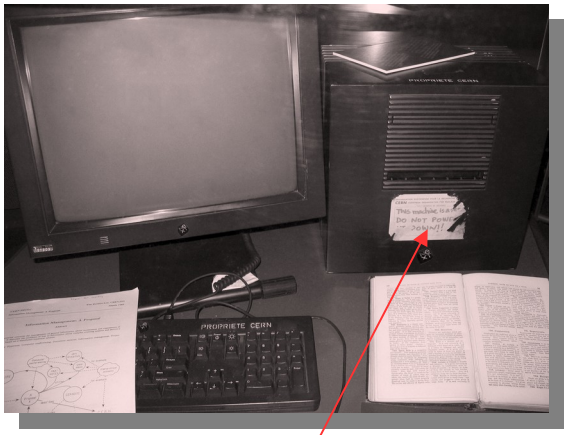
Usi specialistici:
Ricerca, Università

Internet e Web

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

1982

- Definizione del **protocollo TCP/IP**
- Nascita della parola "**Internet**"



1991

- Nasce il **World Wide Web** (WWW)
- Uno dei principali **servizi** di Internet
- **Tim Berners Lee** (CERN di Ginevra) pubblica il primo sito Web
- Concetto di **hypertext**
- **Linguaggio HTML** e **protocollo HTTP**

La nascita del Web

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

Prima degli anni '90: Internet = rete dedicata alle comunicazioni all'interno della comunità scientifica e tra le associazioni governative e amministrative

1982: Definizione del **protocollo TCP/IP** e nascita della parola "**Internet**"

Tim Berners-Lee (CERN di Ginevra)

- Definisce nel 1991 il protocollo *HTTP (HyperText Transfer Protocol)*
- *"The current **incompatibilities of the platforms and tools** make it impossible to access existing information through a common interface, leading to **waste of time**, …"*
- *"A link is specified as an **ASCII string** from which the browser can deduce a suitable method of contacting an appropriate server. When a link is followed, the browser addresses the request for the document to the server."*

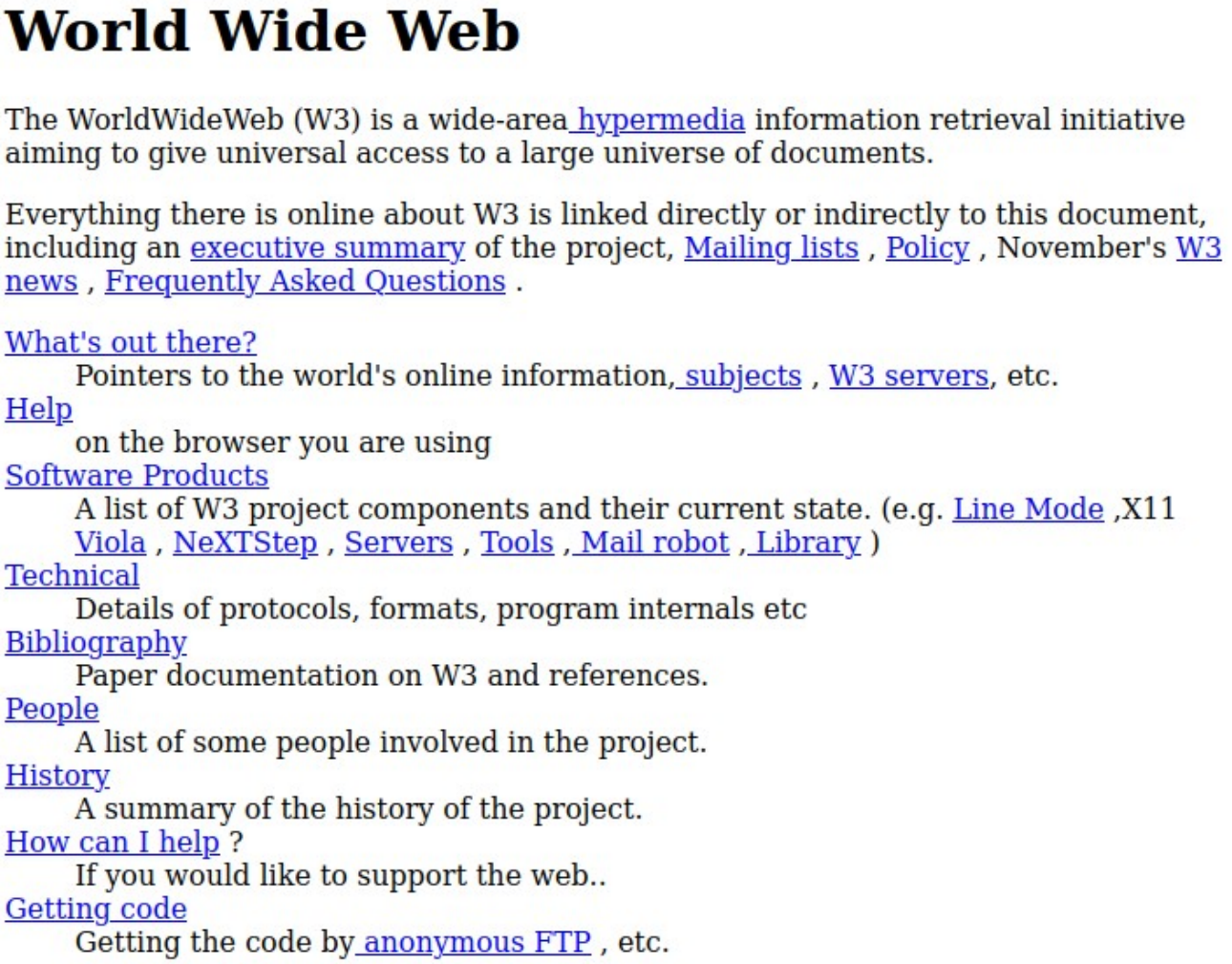
*Nasce il **World Wide Web...***

Il Web in origine

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

Sistema che permette una **lettura ipertestuale, non-sequenziale** dei documenti, saltando da un punto all'altro mediante l'utilizzo di rimandi (**hyperlink**)

Il 6 agosto 1991 Berners-Lee pubblicò il **primo sito Web della storia**, presso il CERN https://web.archive.org/web/20150717103715/http://info.cern.ch/hypertext/WWW/TheProject.html



World Wide Web

The WorldWideWeb (W3) is a wide-area hypermedia information retrieval initiative aiming to give universal access to a large universe of documents.

Everything there is online about W3 is linked directly or indirectly to this document, including an executive summary of the project, Mailing lists , Policy , November's W3 news , Frequently Asked Questions .

What's out there?
Pointers to the world's online information, subjects , W3 servers, etc.
Help
on the browser you are using
Software Products
A list of W3 project components and their current state. (e.g. Line Mode ,X11 Viola , NeXTStep , Servers , Tools , Mail robot , Library)
Technical
Details of protocols, formats, program internals etc
Bibliography
Paper documentation on W3 and references.
People
A list of some people involved in the project.
History
A summary of the history of the project.
How can I help ?
If you would like to support the web..
Getting code
Getting the code by anonymous FTP , etc.

World Wide Web

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

Nel World Wide Web (WWW):

- Risorse disponibili organizzate secondo un **sistema di librerie, o pagine**
- **Accesso** tramite appositi programmi detti **Web browser** che permettono di **navigare** visualizzando file, testi, ipertesti, suoni, immagini, animazioni e filmati

Rilascio pubblico: nel 1993, il World Wide Web è stato reso pubblico e il software necessario per eseguire un browser è stato messo a disposizione gratuitamente

Primo browser con caratteristiche simili a quelle attuali: Mosaic

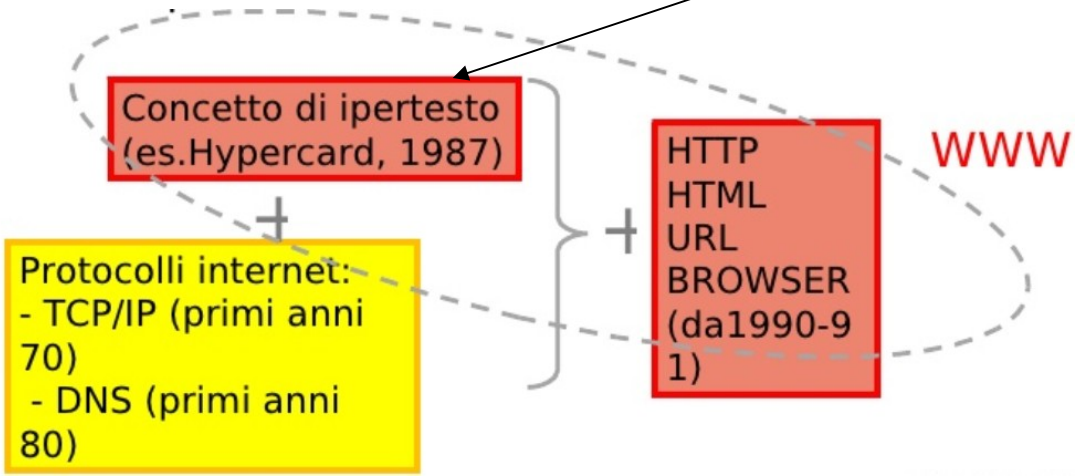
Crescita e Standardizzazione (1994-1995): Il web ha conosciuto una crescita esplosiva, con la creazione di nuovi siti web e l'adozione di standard come i browser Mosaic e Netscape. Nel 1995, il World Wide Web Consortium (W3C) è stato fondato per sviluppare e mantenere gli standard web.

I veri motivi alla base del successo iniziale

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- **Digitalizzazione dell'informazione**
  - (Qualsiasi informazione come sequenza di 0 e 1)
- **Diffusione di Internet (dagli anni '70)**
  - (Trasporto dell'informazione ovunque, in tempi rapidissimi e a costi bassissimi)
- **Diffusione dei PC (dagli anni '80)**
  - (Accesso all'informazione da parte di chiunque a costi bassi)
- **Interfacce utente semplificate (dagli anni '80)**
- **Presenza delle tecnologie sottostanti!**

HyperCard: software ipertestuale introdotto da Apple



"I just had to take the hypertext idea and connect it to the TCP and DNS ideas and—ta-da!—the World Wide Web" Berners-Lee

Il Web 1.0 - statico

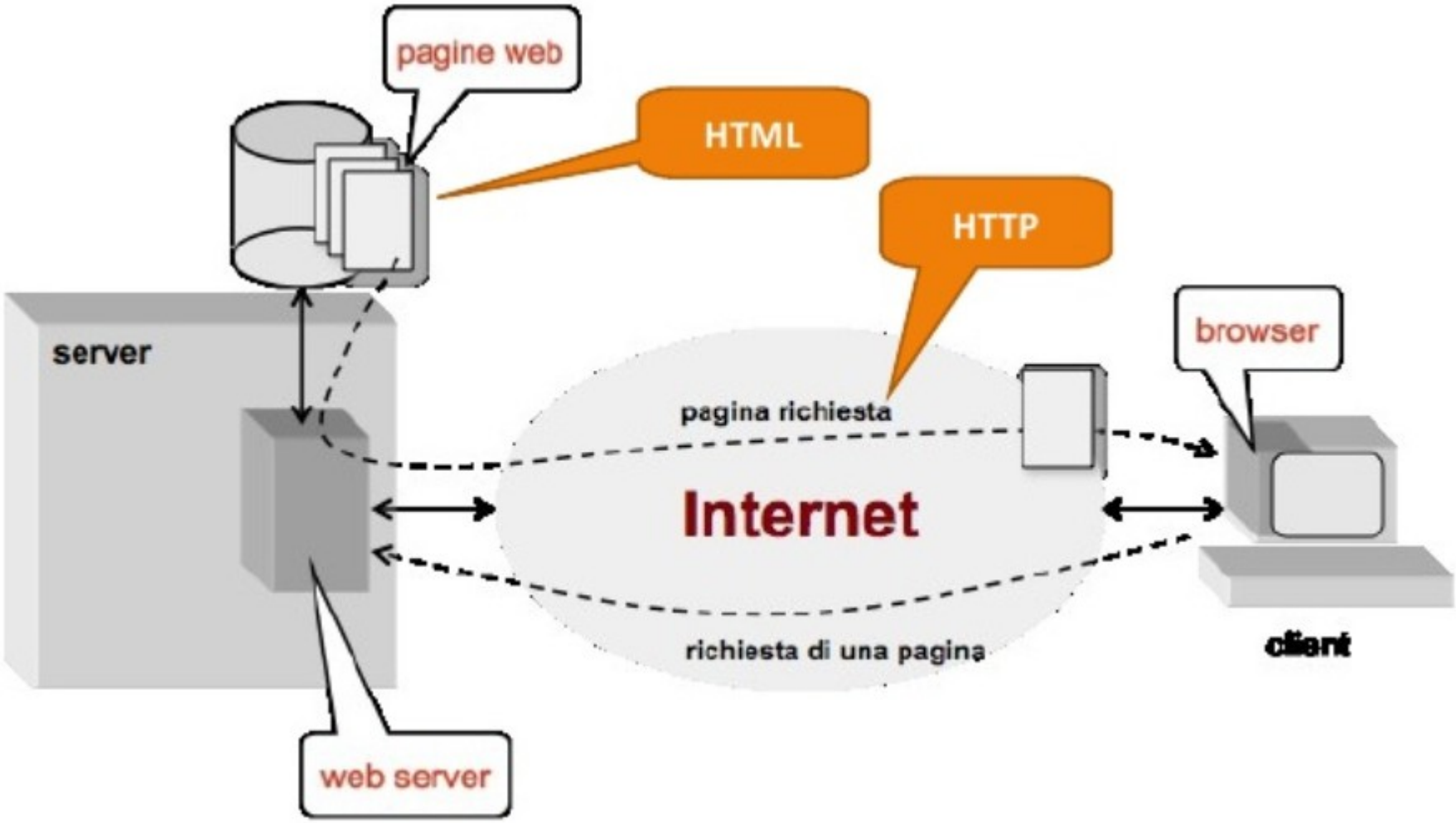
All'inizio il Web era **statico** - l'utente **non interagiva con il contenuto**: gli URL corrispondevano a documenti HTML che venivano restituiti al client

- Siti e portali con contenuti prodotti dal publisher
- Organizzazione gerarchica dell'informazione e navigazione attraverso menu
- Data base
- Interazione sito ↔ singolo utente
- Directory e motori di ricerca
- E-commerce
- Servizi "chiusi" ("attrarre l'utente sul sito, e tenercelo")
- Banda stretta

Web publishing

Prima forma di interattività
Web dinamico

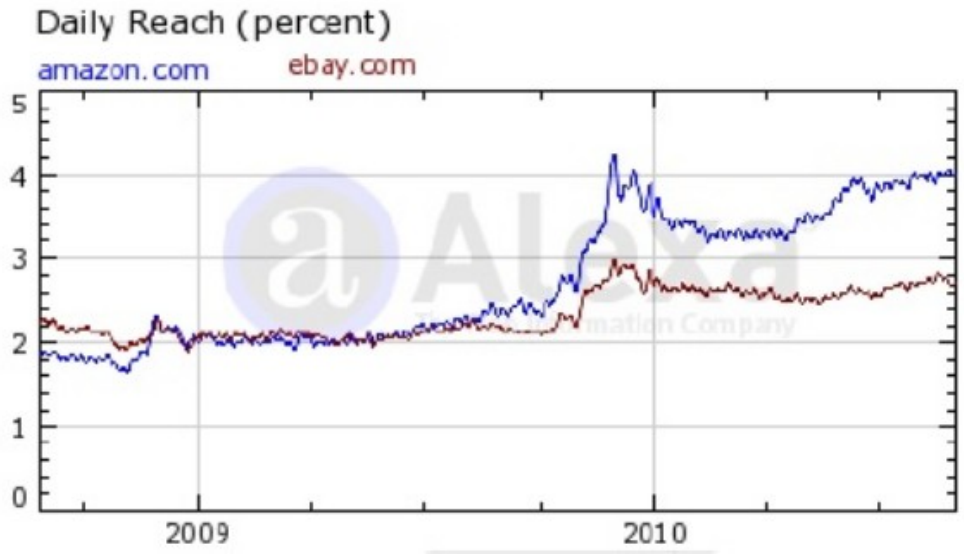
Architettura di base



E-commerce: gli albori

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- http://www.amazon.com
  (dal 1995, http://en.wikipedia.org/wiki/Amazon.com)
- http://www.ebay.com
  (dal 1995, http://en.wikipedia.org/wiki/EBay)



Revenues 2009:
- **Amazon 24,5 B$ (+28%)**
- **eBay 8,7 B$ (+2%)**

Web 2.0

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

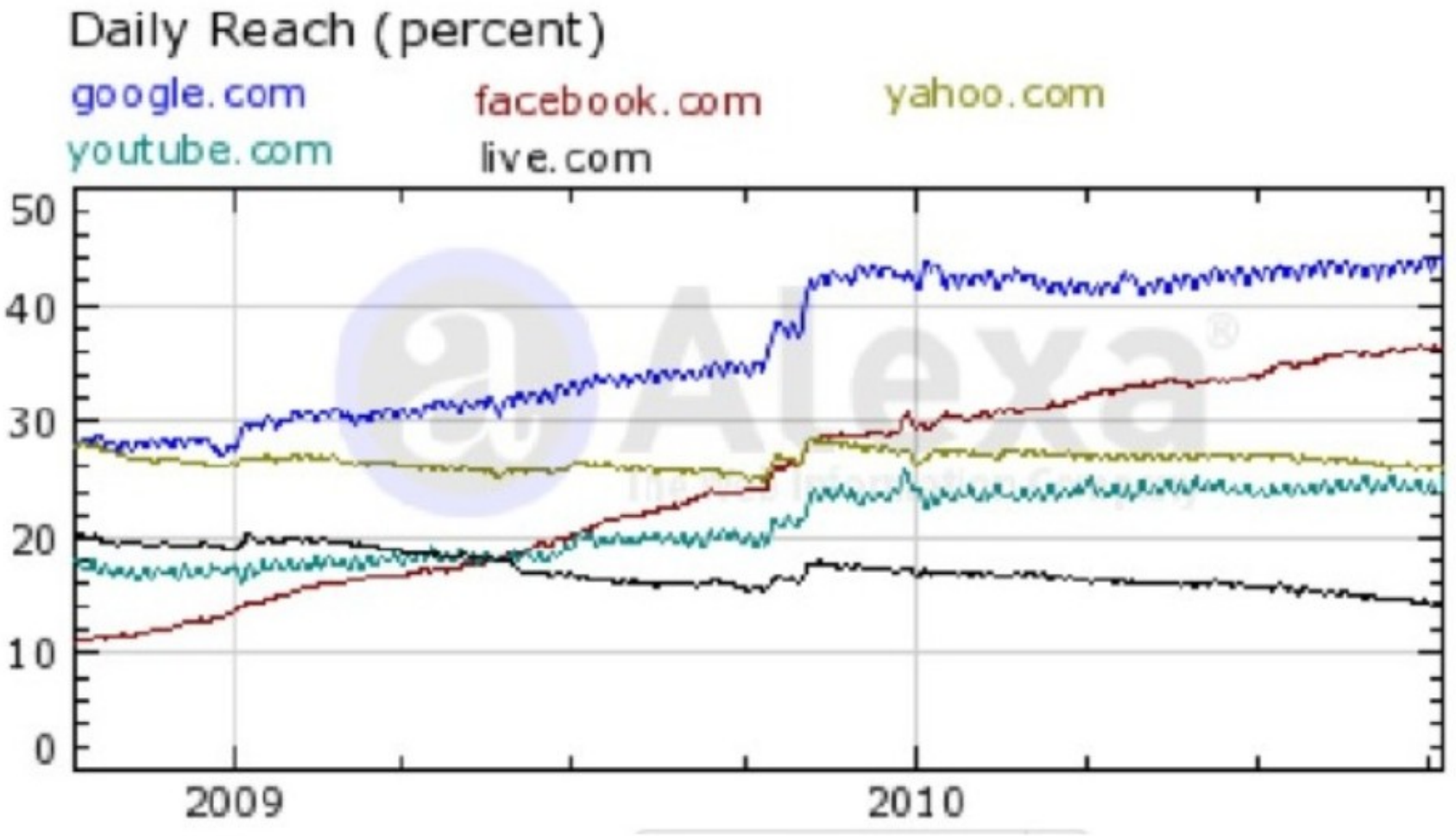
- Termine coniato nel **2004** – **prima Web 2.0 Conference**
- **“Web as a platform”**: i siti Web 2.0 sono piattaforme che consentono grande interazione tra utenti
- **User-centric**: gli utenti forniscono il loro valore aggiunto con l’autoproduzione di contenuti e la condivisione della conoscenza
  - Non più utilizzatori passivi ma creatori di contenuto
  - Centralità e protagonismo dell’utente
  - Comunicazione ‘da uno a molti’ → ‘da molti a molti’
- Da Web ‘read only’ a Web ‘read-write’
- Anche detto ***Social Web***

Time, 2006


TIME
PERSON OF THE YEAR
You.
Yes, you.
You control the Information Age.
Welcome to your world.

Top 5 sites (2010)

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA



Web 3.0

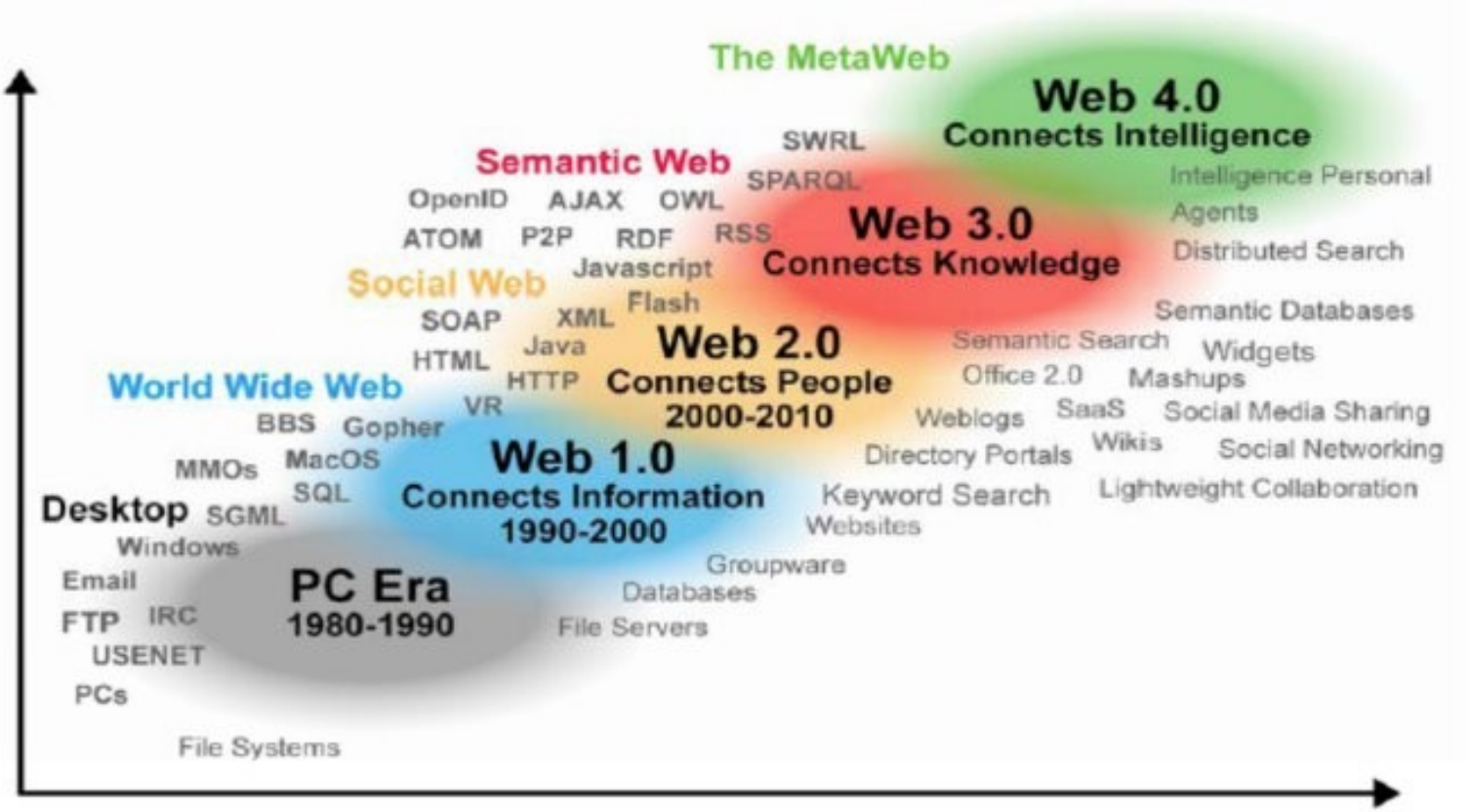
UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- **Web 3.0 → Semantic Web**
- Algoritmi intelligenti sono incorporati nei sistemi Web per interpretare dati e fornire **servizi in una forma più strutturata e personalizzata per ciascun utente**
- Analizzano i dati storici degli utenti e li utilizzano per migliorare la successiva esperienza di navigazione sul Web
  - Es. Sistemi di recommendation: consigli su altri articoli da acquistare in base alle precedenti ricerche e/o acquisti dell'utento stesso e di altri utenti caratterizzati come "simili"
  - Es. Servizi personalizzati sulla posizione geografica dell'utente (meteo, promozioni, eventi, …)
- Apprendimento automatico, reti neurali e intelligenza artificiale, Internet of Things (IoT) ecc., sono alcune delle tecnologie caratteristiche del Web 3.0

Web X.0

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Le definizioni si fanno un pò più confuse



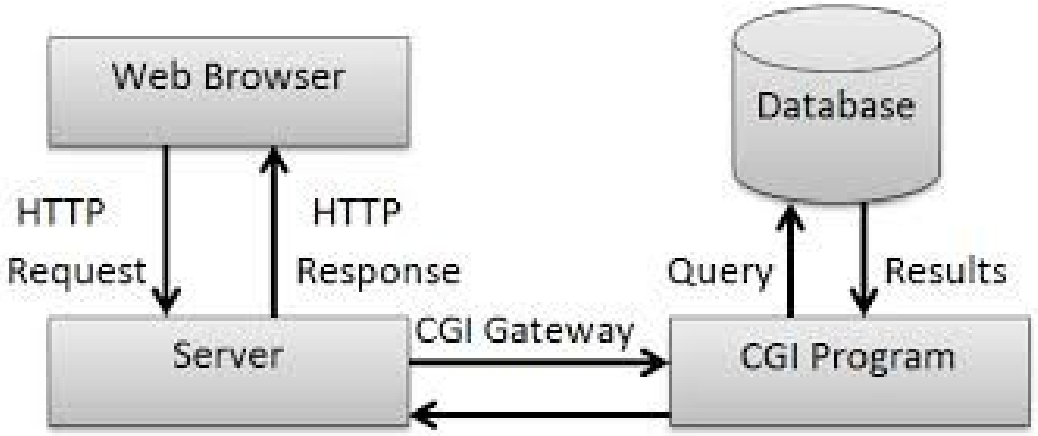
Il Web dinamico

Web dinamico: applicazioni Web che restituiscono pagine dinamiche, ossia che contengono informazioni modificate (personalizzate) in base a dati ricevuti dal client/utente

Primo strumento di **Web dinamico**: uso di **Common Gateway Interface (CGI)** - standard per l'interfacciamento dei server Web con applicazioni esterne per la generazione di contenuti dinamici

URL corrispondenti a programmi CGI eseguiti dal server in tempo reale: il programma CGI interagisce con il DB per generare il contenuto dinamico – personalizzato

Il programma CGI restituisce al server la pagina HTML che contiene informazioni personalizzate, che il server restituisce al client



CGI

CGI

- Programma che può essere scritto con **qualsiasi linguaggio di programmazione** (più usati: C e Perl)
- **Standard di comunicazione** col server circa la modalità per scambiare parametri di input (variabili di ambiente, riga di comando o std input) e l'output

Problema: scalabilità!

- Un nuovo processo invocato per ogni richiesta al server

  → **carico pesante** sul server al crescere della quantità di richieste
- Negli anni 1995-1996 la crescita di pagine Web dinamiche aumenta in modo significativo con l'avvento del commercio elettronico
- *E ci sono altri problemi con CGI...*

Esempio CGI

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

Consideriamo uno script CGI scritto in Python...

```
#!/usr/bin/env python
import MySQLdb
print("Content-Type: text/html\n\n")
print("<html><head><title>Books</title></head><body><ul>")
connection = MySQLdb.connect(user='me', passwd='letmein', db='my_db')
cursor = connection.cursor()
cursor.execute("SELECT name FROM books ORDER BY pub_date DESC LIMIT 10")
for row in cursor.fetchall():
    print("<li>%s</li>" % row[0])
print("</ul></body></html>")
connection.close()
```

Cosa può rappresentare un potenziale problema?

Esempio CGI

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

Consideriamo uno script CGI scritto in Python...

```
#!/usr/bin/env python
import MySQLdb
print("Content-Type: text/html\n\n")
print("<html><head><title>Books</title></head><body><ul>")
connection = MySQLdb.connect(user='me', passwd='letmein', db='my_db')
cursor = connection.cursor()
cursor.execute("SELECT name FROM books ORDER BY pub_date DESC
    LIMIT 10")
for row in cursor.fetchall():
    print("<li>%s</li>" % row[0])
print("</ul></body></html>")
connection.close()
```

Riferimento esplicito a DB con credenziali incluse

SQL inserito nello script

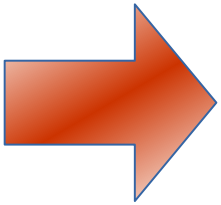
HTML mescolato a logica applicativa

Problemi CGI

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

Consideriamo l'esempio precedente analizzandolo in un'ottica di progetto complesso

- Codice potenzialmente ripetuto: connessione e interrogazioni a DB in SQL
- Codice di dettaglio: settare content-type
- Codice poco riusabile (credenziali DB hardcoded)
- Combinazione codice Python e HTML
- Combinazione di output e application logic



Sconsigliabile anche per un **progetto piccolo one-of-a-kind!**

Tecnologie di scripting

- CGI → **scalabilità limitata**
- Limite alla scalabilità: dovuto alla necessità di attivazione di un nuovo processo per ogni richiesta dinamica
- Per aumentare le prestazioni, servono tecnologie che evitino la creazione di un nuovo processo ad ogni richiesta
- Nascono le **tecnologie di scripting**
    - **FastCGI**: permette di condividere un'istanza di un programma CGI
    - **Server API** (Netscape NSAPI, MS ISAPI): Librerie condivise caricate nello spazio del server HTTP, in grado di servire richieste multiple senza creare nuovi processi. Ma... **poca portabilità** (legate al server HTTP) e **vulnerabili**
    - **mod_perl, mod_python, mod_php**: moduli di Apache, in grado di interpretare script all'interno del processo HTTP

Tecnologie di scripting

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Come ulteriore evoluzione, nascono i **linguaggi di scripting:** codice inserito nelle pagine HTML ed interpretato direttamente dal server HTTP
  - **Active Server Pages** (ASP): tecnologia Microsoft, pagine contenenti funzioni in JScript o VBScript (Microsoft's Visual Basic Scripting) → Soluzione specifica per server HTTP, *problemi di manutenibililtà del software HTML/script integrato*

Problema già visto nel CGI

```
<html>
 <head>
  <title>Pagina in Asp</title>
 </head>
 <body>
  <script language="VBScript" runat="Server">
     response.write "Hello World!"
  </script>
 </body>
</html>
```

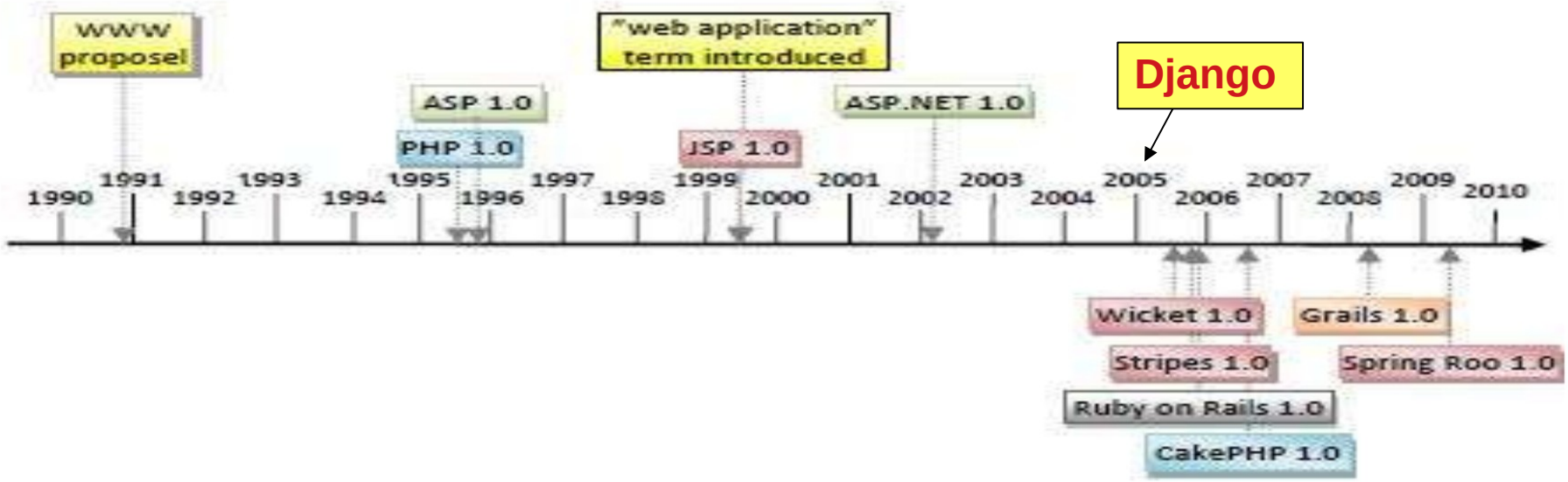
  - **Java servlets**: programmi server-side scritti in Java in grado di servire multiple richieste HTTP con un solo processo (processo multi-thread eseguito lato server)
  - **Java Server Pages (JSP)** – pagine HTML con speciali tag per invocare funzioni predefinite sotto forma di codice Java (più semplice rispetto a Java servlets)
  - **PHP** “Programming Language of 2004”

PHP

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- **PHP: Hypertext Preprocessor**
- Nato a metà anni ‘90: in origine una **raccolta di script CGI**
- Evoluzione: possibilità di **integrare il codice PHP nel codice HTM**L per semplificare la creazione di pagine dinamiche
  - Usato per funzionalità di accesso al Database e processing dati
- Grande popolarità
  - Nel **1997** circa 50mila siti Web erano basati su PHP
  - **1998** PHP 3.0 installato sul 10% dei server Web in Internet
  - 2001 supera il milione di siti che lo utilizzano
  - Attualmente ancora molto popolare per applicazioni Web lato server
    - *Gran quantità di siti Web ancora scritti in PHP*
    - *CMS popolari (Wordpress, Drupal, …) basati su PHP*

Evoluzione….

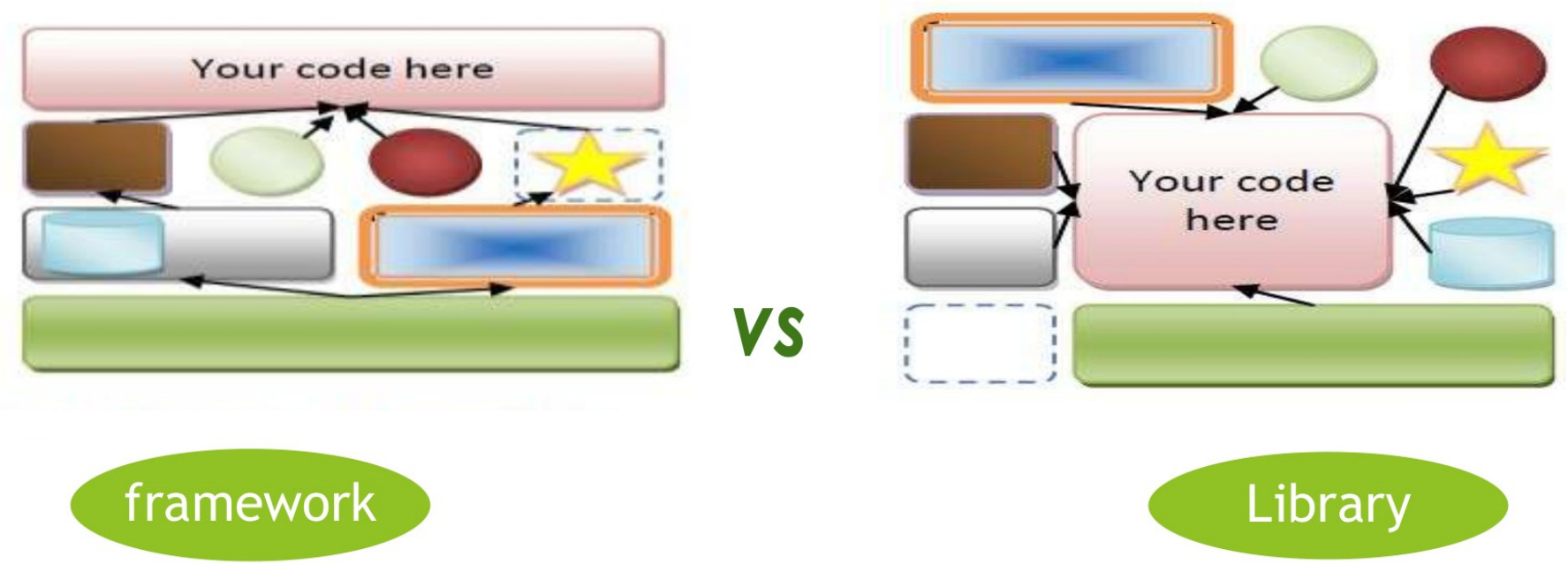


… verso i Framework Web

- **Dal 2002 in poi: nascita e sviluppo di Framework Web**
- **Django nasce nel 2005**

Framework Web

- Framework Web progettati per supportare lo sviluppo di siti Web dinamici
- **Include librerie** che offrono funzionalità comuni a questo tipo di applicazioni
- **Base o scheletro su cui costruire**



Vantaggi dei Framework Web

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Tutte le applicazioni Web moderne hanno un **set comune di requisiti:** user management (es., secure user login, password recovery), group management, funzioni di autenticazione/autorizzazione, funzioni admin, supporto per la sicurezza (protezione da attacchi più comuni)
- Un Web Application Framework supporta già queste funzionalità, lasciando la possibilità agli sviluppatori di **focalizzarsi sugli specifici requisiti delle loro applicazioni**
- Web Application Framework offrono **supporto per interazione trasparente con i più popolari database relazionali (*e non solo*)** sia in termini di creazione della base di dati che di funzioni per l'accesso e la manipolazione dei dati contenuti nel DB

Modello MVC

Modello MVC

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Tutti i framework Web moderni si basano sul **paradigma di programmazione MVC - Model View Controller** *(o simile)*
- MVC fornisce una struttura logica e modulare per sistemi fortemente interattivi e complessi
- Aderisce a best practices di progetto e ingegnerizzazione del software
  - *Information hiding*
  - *Disaccoppiamento*
  - *Don't Repeat Yourself (DRY)*
- Nato (e ampiamente usato) nell'ambito delle GUI (desktop computing) legate alla programmazione ad oggetti
- La sua popolarità ne ha determinato l'adozione anche in ambito Web

Concetti base MVC

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

Model:

Modello che astrae la rappresentazione dei dati

Tipicamente si usa un modello basato su object-relational database (ORD) per mappare classi e tabelle

View:

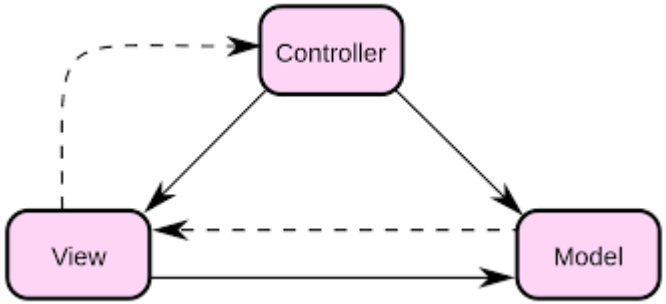
Si occupa del rendering (visualizzazione) dei dati contenuti nel model che vengono mostrati all'utente

Un solo model può avere un numero elevato di rendering diversi (importantissimo per applicazioni con requisiti di multicanalità)

Controller:

Gestore di eventi che processano le interazioni con l'utente (alcune interazioni arrivano attraverso il view)

Modifica il modello (se l'interazione lo richiede): tale modifica può richiedere poi l'aggiornamento del view



Model

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Si occupa della **gestione dei dati** dell'applicazione
- Fornisce le funzioni per la **manipolazione** e la **modifica** dei dati
  - Contiene i metodi usati dagli altri componenti per svolgere le loro funzioni sui dati
  - Esporta le **interfacce** dei metodi
- Deve essere **completamente indipendente** dagli altri componenti Controller e View (***information hiding***)
  - **Flessibilità e robustezza**
  - **Best Practice**

View e Controller

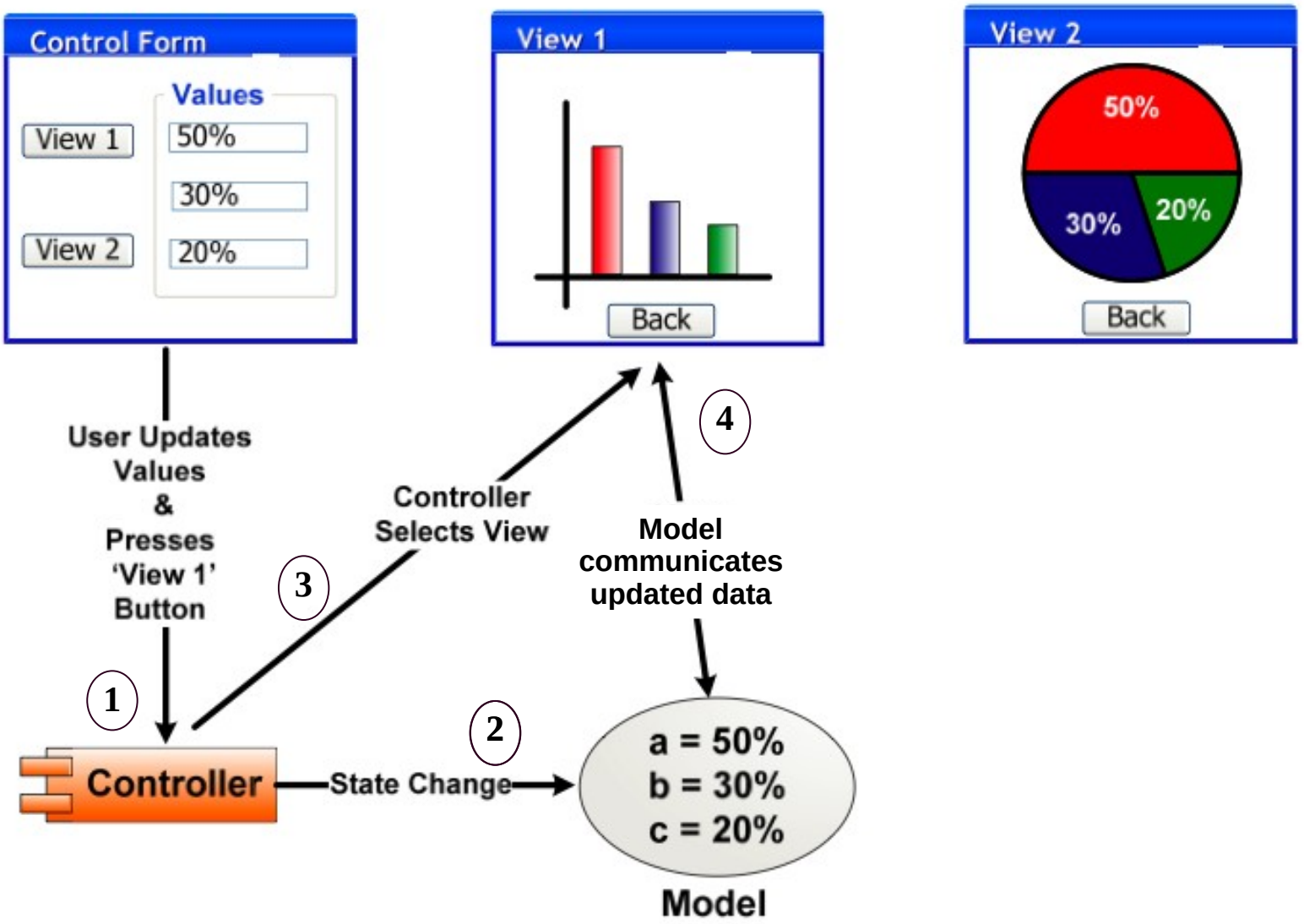
View

- View mostra all'utente lo **stato (aggiornato) del modello**
  - Usa le interfacce dei metodi del modello per ricevere i dati aggiornati
  - Ruolo di osservatore passivo, non ha impatto sul modello

Controller

- Il controller decide cosa fare col modello in risposta alle azioni dell'utente
  - Può contenere parte della logica applicativa
- Il progetto del controller può dipendere dal modello
  - Non vero il viceversa

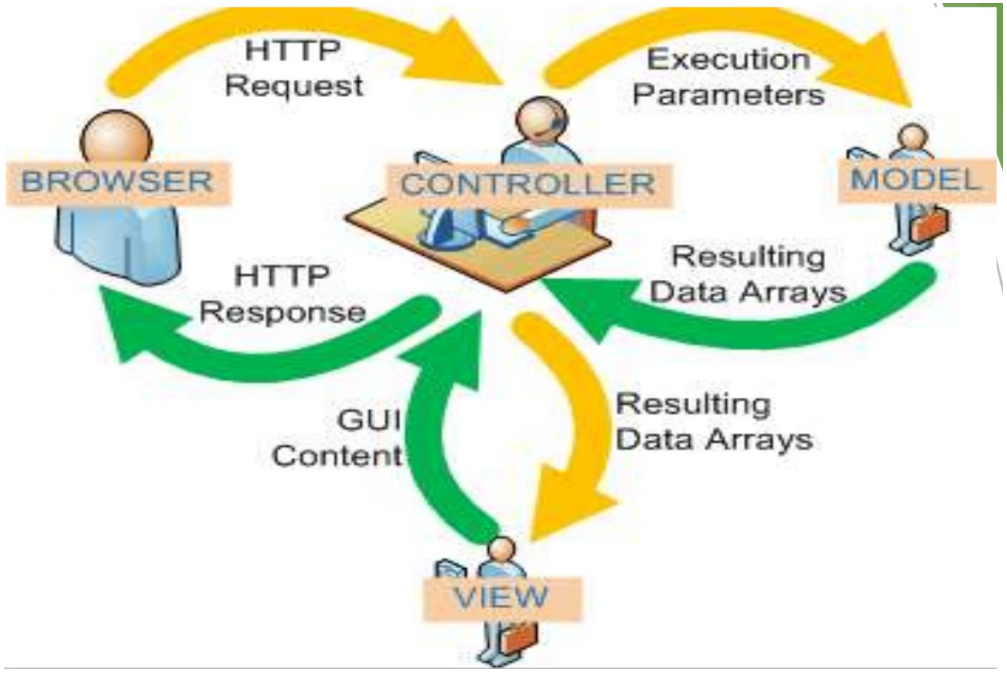
Esempio Interazione MVC



Relazione tra componenti

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

Possibili interpretazioni “leggermente” diverse



Riflesse nella pratica: diversi approcci nei vari framework Web esistenti *(vedremo in Django)*

MVC nel Web: osservazione

- L'approccio MVC si adatta molto bene a GUI in **ambito desktop computing** (per cui è nato)
- L'approccio MVC nel **contesto Web** richiede più attenzione
- Più piattaforme coinvolte (client-server)
  - La visualizzazione dei dati avviene sul client → piattaforma diversa da quella su cui risiedono i dati
- Nel Web la **visualizzazione (view)** avviene sul client

→ *Problema di possibili view con dati non aggiornati rispetto al server*

- In quest'ottica, si distinguono **diverse interpretazioni del modello MVC** applicato al Web:
  - *Diversa divisione dei task tra client e server*

MVC nel Web: osservazione

- Approccio **server-side (classico)**: tutti i componenti MVC risiedono sul server
  - **Thin client**
  - Il client manda richieste, che saranno gestite dal controller, e riceve in risposta una pagina Web completa dal componente view
  - Possibili soluzioni: **polling con refresh periodico** → richieste periodiche di refresh da parte del client
  - Potenzialmente inefficiente (stale information, carico inutile del server, trade-off)
- **Tecnologie client-side:**
  - Far girare parti dei componenti MVC sul client
    - Incluse tecnologie basate su AJAX, JQuery,...

Django Framework

Django

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- Web application framework
- Free e open source
- Interamente scritto in Python
- Rilasciato nel 2005 (Django Software Foundation)
- Numerosi siti Web molto popolari ne fanno uso: Instagram, Pinterest, Mozilla, The Washington Times,…
- Sempre più usato a livello aziendale
- **Obiettivo**: rendere *facile e veloce* la creazione di applicazioni Web complesse e basate su database (*database-driven Web sites*)
- **Punti di forza: riusabilità e "pluggability" dei componenti, sviluppo rapido, e principio DRY (Don't Repeat Yourself)**

Django come framework MVC

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

Due diverse interpretazioni

1) Separazioni degli elementi MVC in tre elementi (file) diversi

models.py → model

urls.py → controller

views.py → view

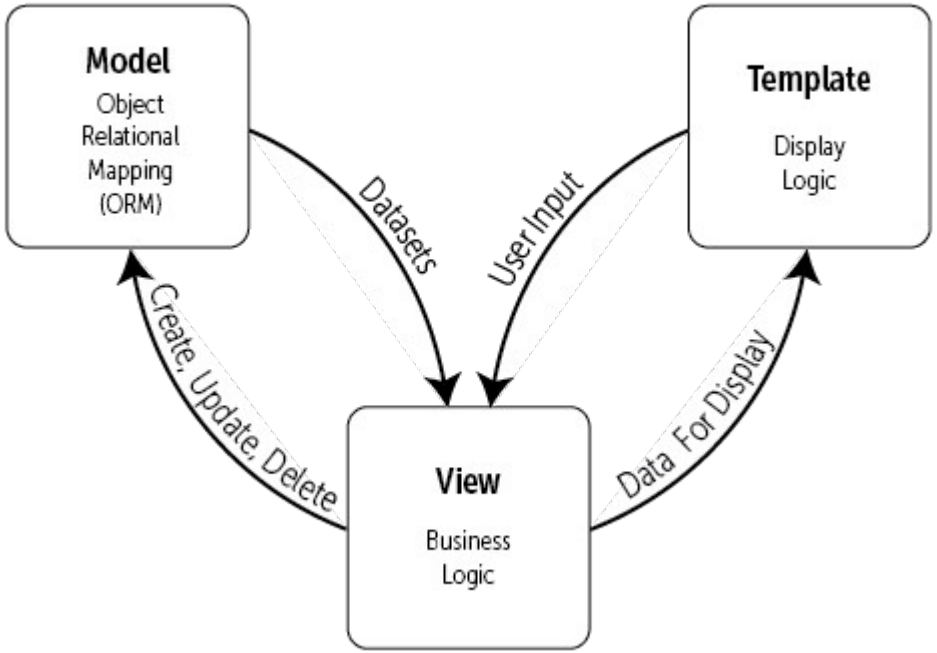
2) Visione alternativa: modello MTV

Model

Template

View

Controller



Model in Django

- Model implementato direttamente come collezione di classi Python che andranno a rappresentare le tabelle del database
- Modello Object-Relational Database (ORD)
  - Convergenza di modello relazionale e a oggetti
  - Approccio Object-Relational Mapping (ORM)
- Il codice SQL per creare lo schema del DBMS è generato ed eseguito automaticamente da Django quando si esegue il deployment del modello
- Creazione di un 'virtual object database'
  - Accessibile direttamente attraverso API del linguaggio di programmazione Python (es. query/join)
  - *Non avremo bisogno di scrivere codice SQL*

Filosofia di progetto: interazione con DB

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- **Interazione trasparente**
- **Django usa moduli Python per interfacciarsi con DB specifici e aggiunge un proprio strato software**
- **Principali DB supportati (e non solo)**
  - MySQL
  - PostgreSQL
  - SQLite3 (default) – *conoscete? Uno dei 5 sw più installati al mondo...*
  - Oracle
- **Ulteriori estensioni supportano formati di dati geo-referenziati e altre caratteristiche**

Filosofia di progetto: API Python per DB

API Python per il DB

- Efficienza del codice SQL prodotto: limitare al massimo le interazioni col database
- Sintassi semplice per interagire coi dati:
  - Dati accessibili da ogni modulo
  - Codice di operazioni complesse (es. join) generato automaticamente dallo strato di interfacciamento software

Include la possibilità di scrivere direttamente codice SQL (*solo quando veramente necessario*)

View e controller in Django

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

View in Django:

- Funzioni Python (view.py) per fornire le risorse (pagine e dati) richieste dall'utente → ***include logica di business!***
- Accesso a dati del model attraverso API Python
- Uso di template Django per generare diversi tipi di output (HTML, XML, CSV, etc.)

Controller in Django

- Realizzato attraverso file urls.py
- Obiettivo: mappare gli URL richiesti sulle opportune risorse richieste dall'utente → funzioni Python in view.py
- Uso di espressioni regolari Python per fare il match con gli URL richiesti - *Regular expression matching*

Strumento Template

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

Template: strumento molto potente

- **Separare la logica (view) dalla presentazione (template)**
- Definizione di standard API per il caricamento e il rendering dei templates (API invocate nelle view)
- Non inventare un nuovo linguaggio di programmazione: *c'e' già Python...*
- Uso di **codice** apposito nei template per includere un minimo di **logica**
- Assunzione: chi scrive i template non dipende da strumenti WYSIWYG
  - Uso di variabili {{ var }} passate dalla view in un context

    My first name is {{ first_name }}. My last name is {{ last_name }}
  - Uso di condizioni e cicli

```
{% if user.is_authenticated %}Hello, {{ user.username }}.{% endif %}
<ul>
{% for athlete in athlete_list %}
    <li>{{ athlete.name }}</li>
{% endfor %}
</ul>
```

Strumento Template

- Evitare template in XML: lento il parsing...
- Evitare la ridondanza: meccanismo di ereditarietà nei template
  - Consente di creare uno “scheletro” di modello di base che contiene tutti gli elementi comuni del sito e definisce i blocchi che i modelli secondari possono sovrascrivere.

```
#base.html
<!DOCTYPE html>
<body>
        <div id="content">
        {% block content %}{% endblock %}
    </div>
</body>
</html>
```

```
#child.html
{% extends "base.html" %}

{% block content %}
{% for entry in blog_entries %}
    <h2>{{ entry.title }}</h2>
    <p>{{ entry.body }}</p>
{% endfor %}
{% endblock %}
```

- Security e Safety by design (protezione contro codice malevolo)

Django come framework MTV

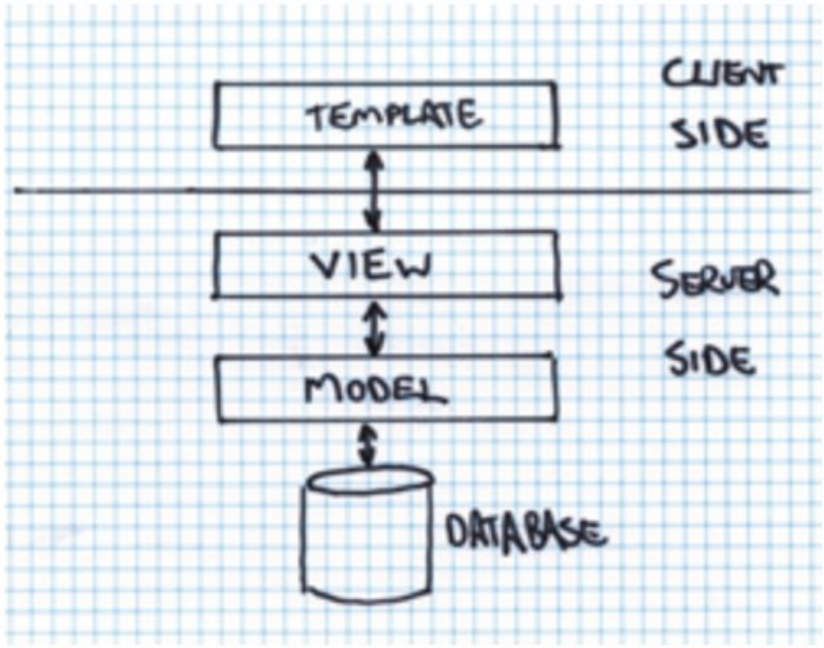
UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

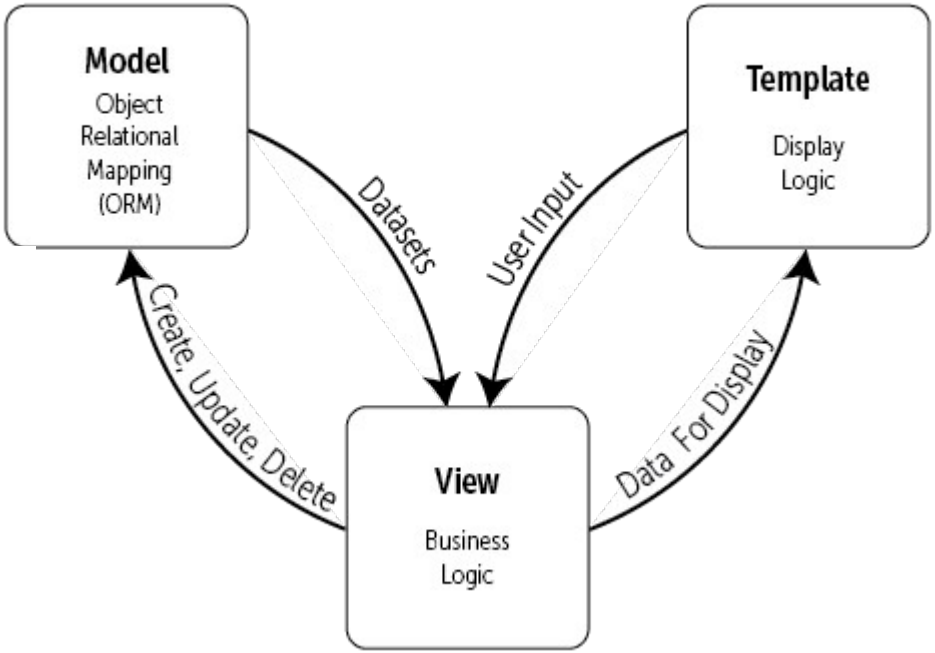
Visione alternativa: modello MTV

Model

Template

View





Struttura di un progetto Django

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

mysite (dir top-level)

manage.py

mysite (package Python)

__init__.py

settings.py

urls.py, ...

app1 (package Python: __init__.py, urls.py, models.py, views.py, ...)

Progetto (es. sito Web) creato con Django e chiamato mysite

Tassonomia di un progetto Django

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- mysite/: cartella top-level di progetto
- manage.py: script per automatizzare le operazioni di Django
- mysite/mysite/: cartella della applicazione principale (omonima) del progetto – package Python
- settings.py: impostazioni per il funzionamento dell'applicazione, tra cui:
    - Definizione dei path dell'applicazione
    - Configurazione accesso al DBMS
    - …
- urls.py: configurazione degli URL per il progetto (livello root)

Tassonomia di un'applicazione django

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- **wsgi.py:** file che serve da entry-point per l'interazione con un Web server (es. Apache, Nginx) esterno mediante standard WSGI
  - Interazione tra Web server e applicazioni Python
- **urls.py:** struttura dello spazio degli URL dell'applicazione - mappa gli URL richiesti in invocazioni verso le componenti del sistema (implementa il controller del progetto)
- **File non presenti nello scheletro:**
  - **models.py: modello**
  - **views.py: view**
- **Sono spesso presenti solo nelle sotto-applicazioni (non di default) che svolgono azioni specifiche**

Progetto vs. applicazione in Django

- Un singolo progetto/sito Django può essere spezzato in diverse sotto-applicazioni
- Ogni applicazione è un package Python separato

app1/: cartella top-level della applicazione app1 del progetto (conterrà __init__.py e tutte le cartelle e i file relativi all'applicazione)

  - Ogni applicazione ha i propri componenti MVC (model.py, views.py, urls.py) / o MTV
  - Le applicazioni possono interagire includendo l'una il package dell'altra

Django-admin/manage

UNIMORE
UNIVERSITÀ DEGLI STUDI DI MODENA E REGGIO EMILIA

- **Sintassi:**
  - manage.py <command> [options]
- **Principali comandi:**
  - **startproject: crea scheletro di un progetto**
  - **startapp**: crea scheletro applicazione
  - **runserver**: avvia server Web di development
  - **migrate**: creazione e gestione DB
  - **dumpdata**: scarica dati DB
  - **loaddata**: carica dati in DB
  - **test**: esecuzione di test
  - ...