



# WorldSensitive



GAD

Prof. Gennaro Costagliola

A.A. 2015/2016

Alessia Antelmi

Martina Garofalo

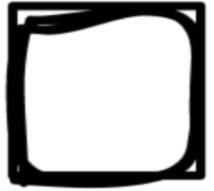
Enrica Loria

Maria Angela Pellegrino

*That's what games are, in the end. Teachers.  
Fun is just another word for learning.*

**Raph Koster**

# Obiettivi



Integrazione di dati eterogenei e autonomi



Gestione consistente delle risorse



Strumento per l'apprendimento  
semplice ed intuitivo



Separazione delle aree di conoscenza

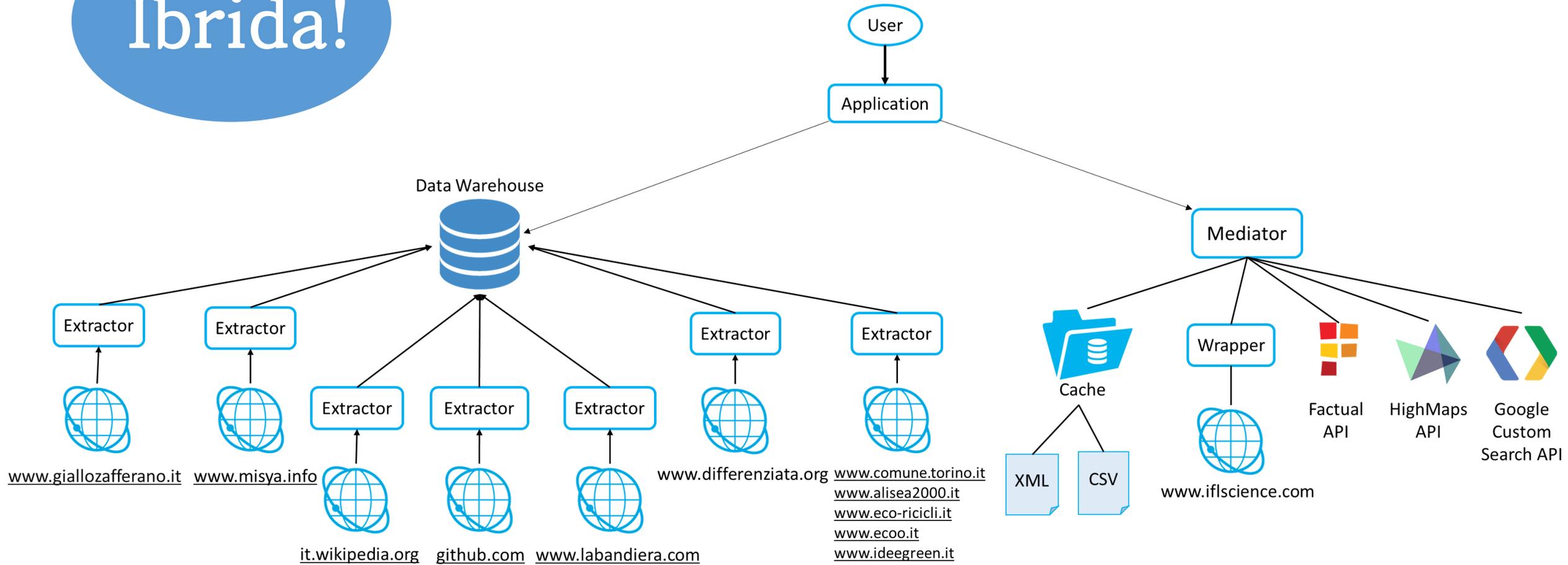


Piacevole interazione con l'utente



# Architettura

Ibrida!





Il numero di fonti totale è 19.

# Ecologia



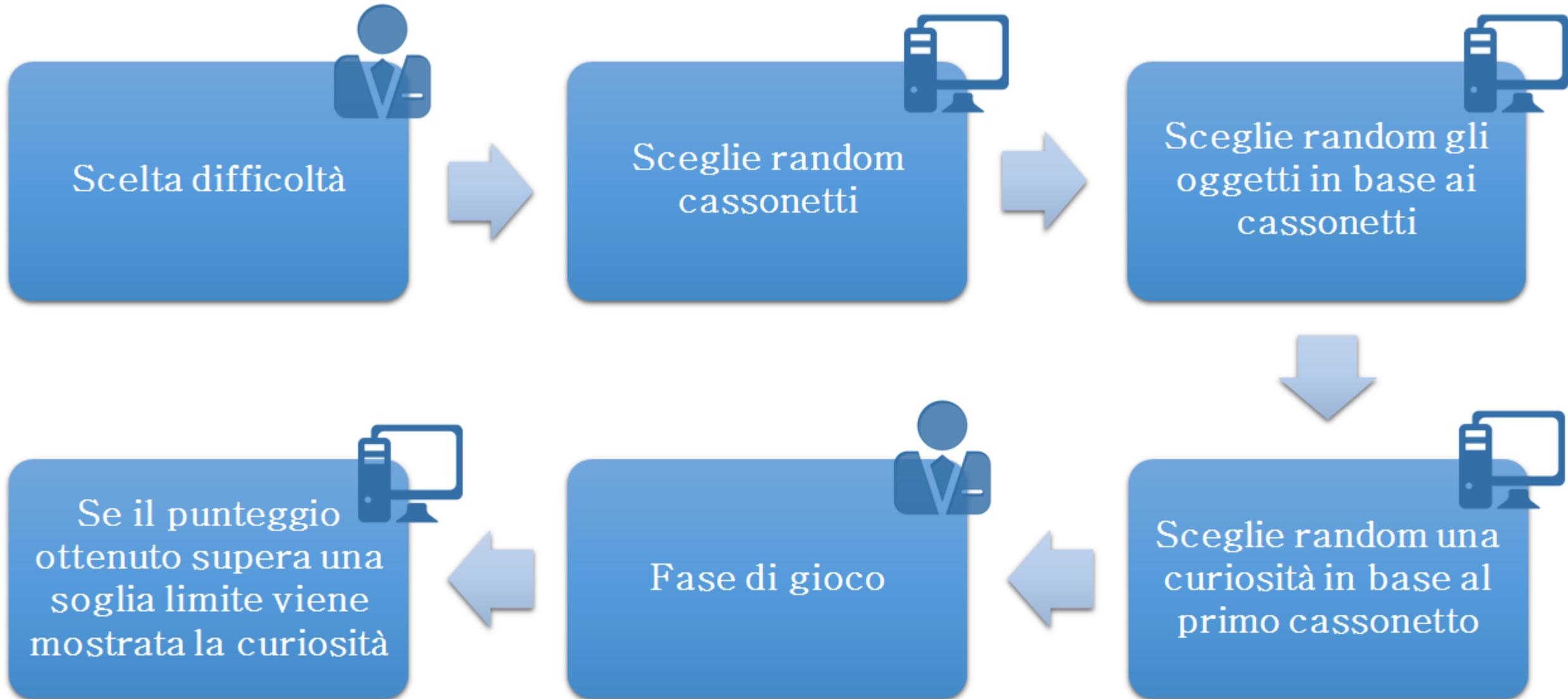
Degli oggetti scelti in modo casuale cadono dall'alto verso il basso e l'utente può muovere uno o più cassonetti in modo che l'oggetto in caduta libera sia raccolto nel cassonetto corretto.

Il numero di cassonetti dipende dalla difficoltà del gioco.

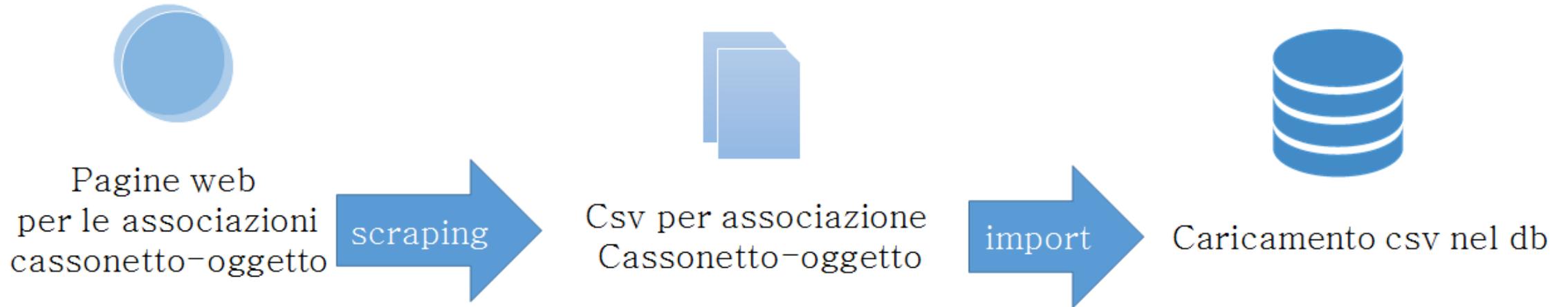


7 fonti

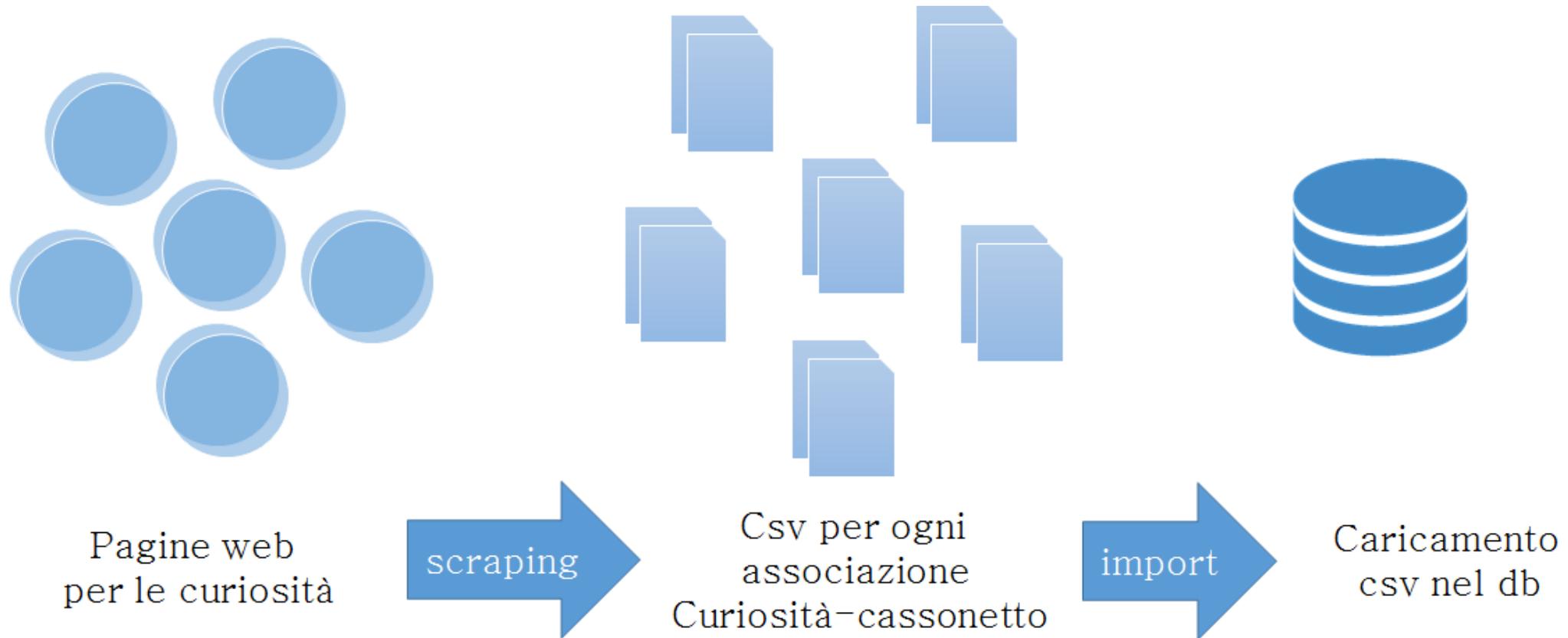
# Processo di esecuzione



# Processo di estrazione dati



# Processo di estrazione dati



# Cucina

La sezione della cucina offre un mini quiz per (quasi) ciascuna regione italiana. Il quiz prevede quattro domande a risposta multipla, con quattro opzioni.

Al termine del gioco se l'utente risponde correttamente ad almeno tre domande otterrà una curiosità riguardo uno degli ingredienti della ricetta in questione.



2 fonti



# Processo di esecuzione



Selezione  
regione.



Sceglie 1  
ricetta tipica  
della regione e  
3 ricette di  
altre regioni.



Sceglie 3  
ingredienti  
non utilizzati  
nella ricetta  
tipica.



Elabora le  
domande sulla  
base delle  
informazioni  
ricevute

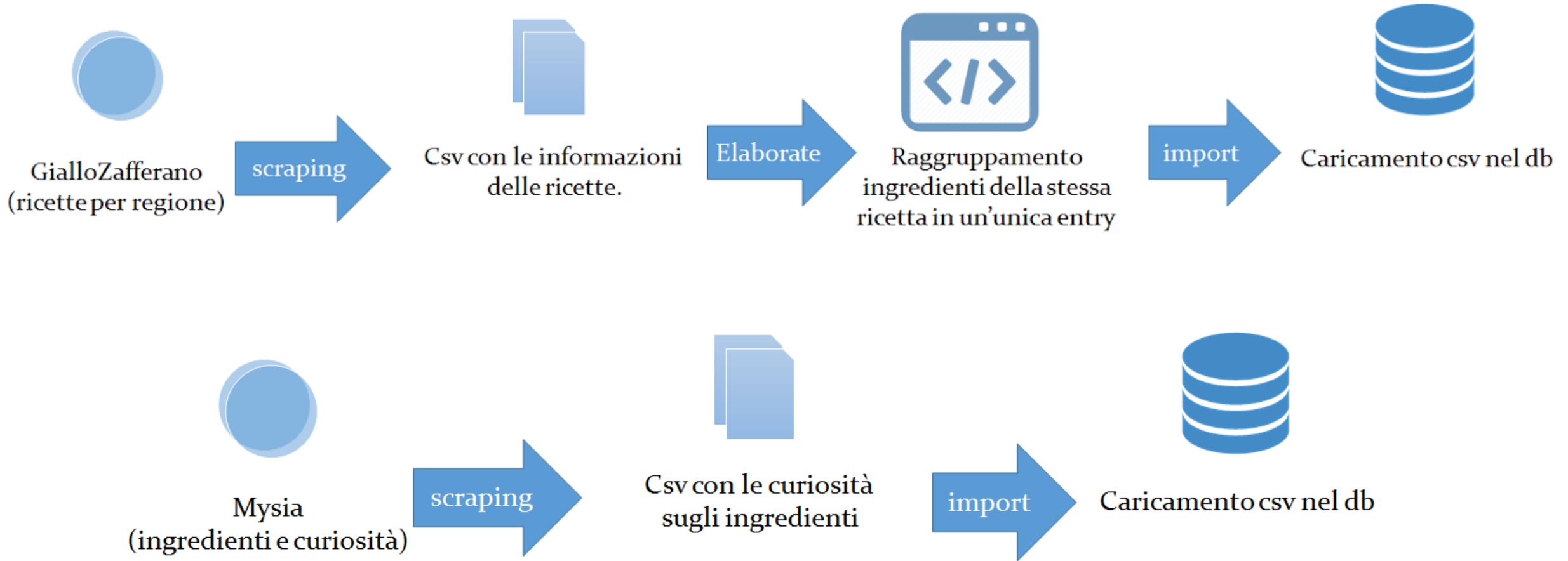


Seleziona le  
risposte.



Se le risposte  
corrette sono  
più di 3  
mostra una  
curiosità.

# Processo di estrazione dati





# Geografia

Raggruppamento bandiere in base ai colori.

Mappa interattiva del mondo con cui l'utente può interagire per associare le bandiere ai relativi stati.



5 fonti

# Processo di esecuzione



Individua i  
colori  
disponibili.



Sceglie un  
colore.



Sceglie 10  
bandiere  
contenenti il  
colore  
indicato.



Seleziona una  
bandiera.

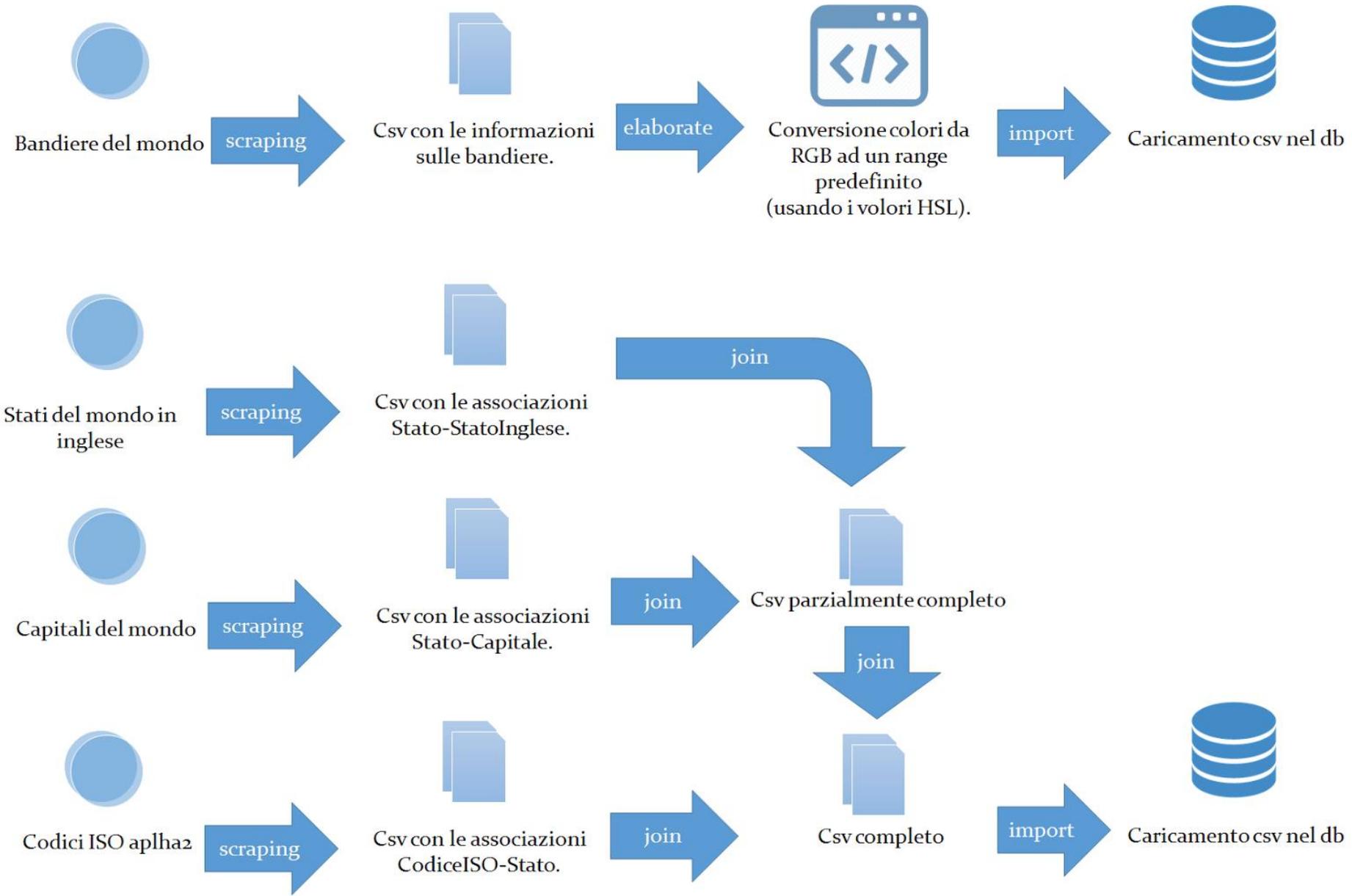


Seleziona uno  
stato



Se la risposta  
è corretta  
mostra la  
capitale dello  
stato.

# Processo di estrazione dati



# News

Tale sezione offre la possibilità di visualizzare delle notizie tratte dal sito web <http://www.iflscience.com>.

All'interno di tale fonte, è possibile trovare i post pubblicati divisi in nove categorie distinte: Editor's blog, Space, Technology, Health&Medicine, Chemistry, Environment, The Brain, Plants&Animals, Physics.



# Processo di esecuzione

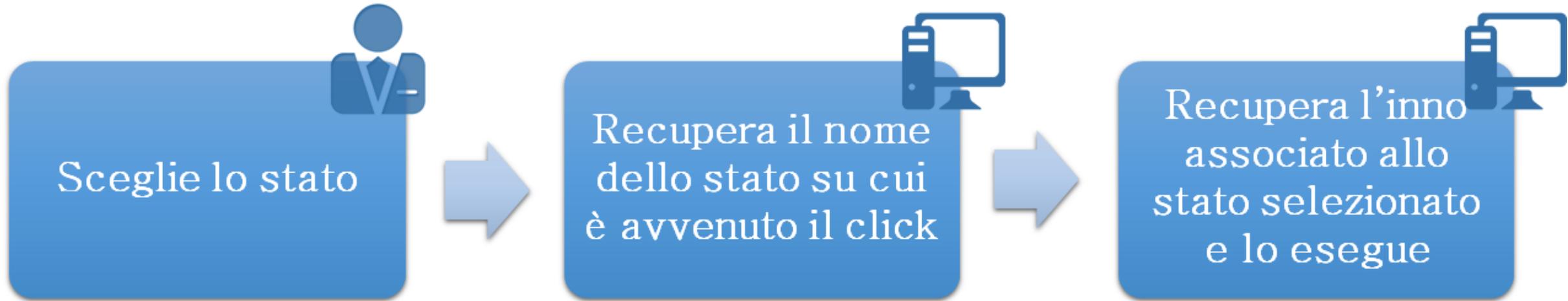


# Musica

L'idea è che all'utente viene mostrata una mappa di tutto il mondo e viene suonato l'inno dello stato che è stato scelto dall'utente.



# Processo di esecuzione



# Processo di estrazione dati



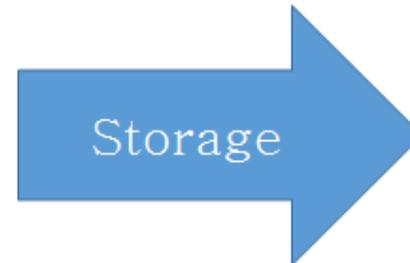
[www.labandiera.com](http://www.labandiera.com)



File in formato .mid



File in formato .mp3



Caricamento mp3  
in una cartella

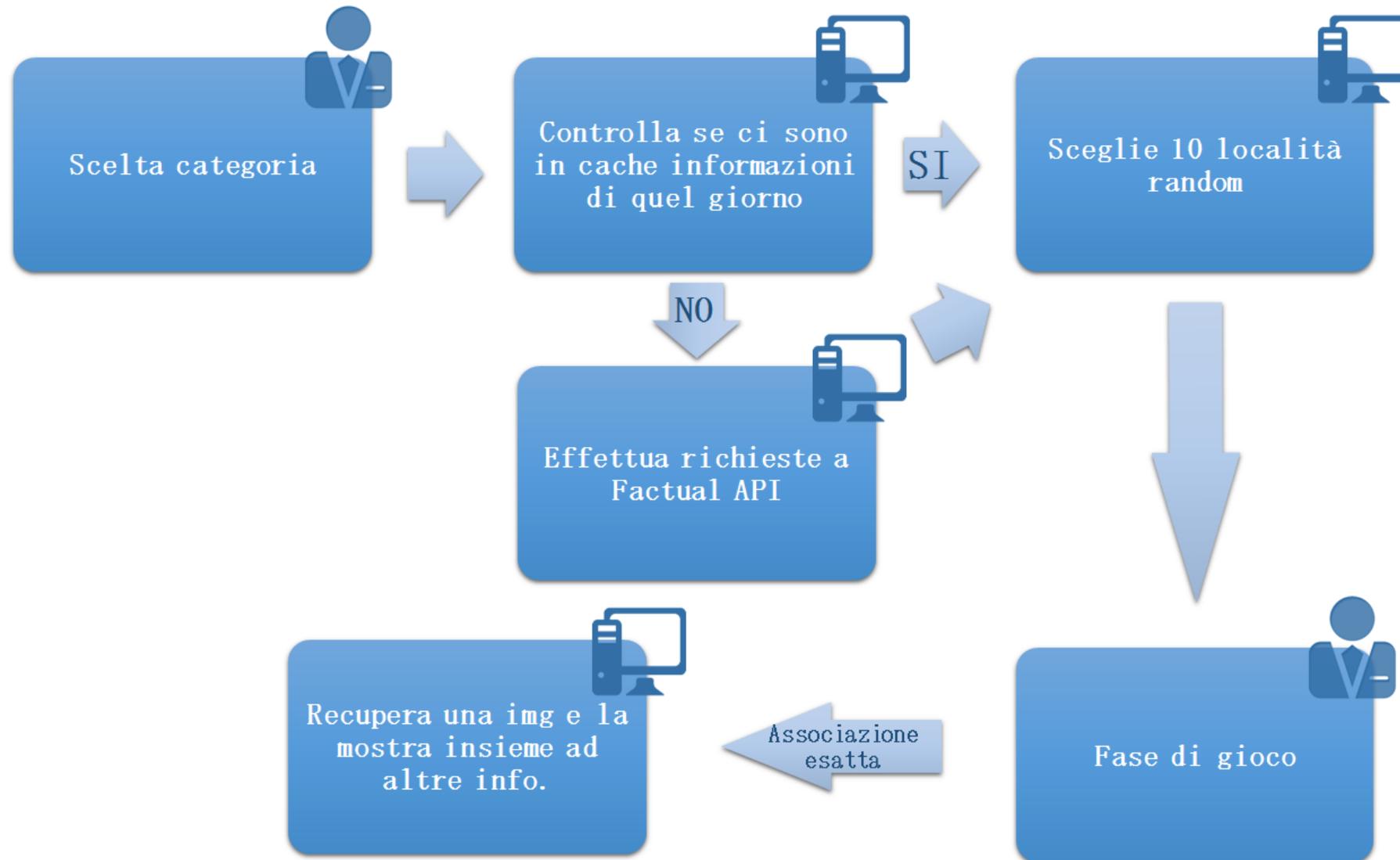
# Arte

Mappa dell'Italia con cui l'utente poteva interagire associando monumenti italiani e la rispettiva regione.

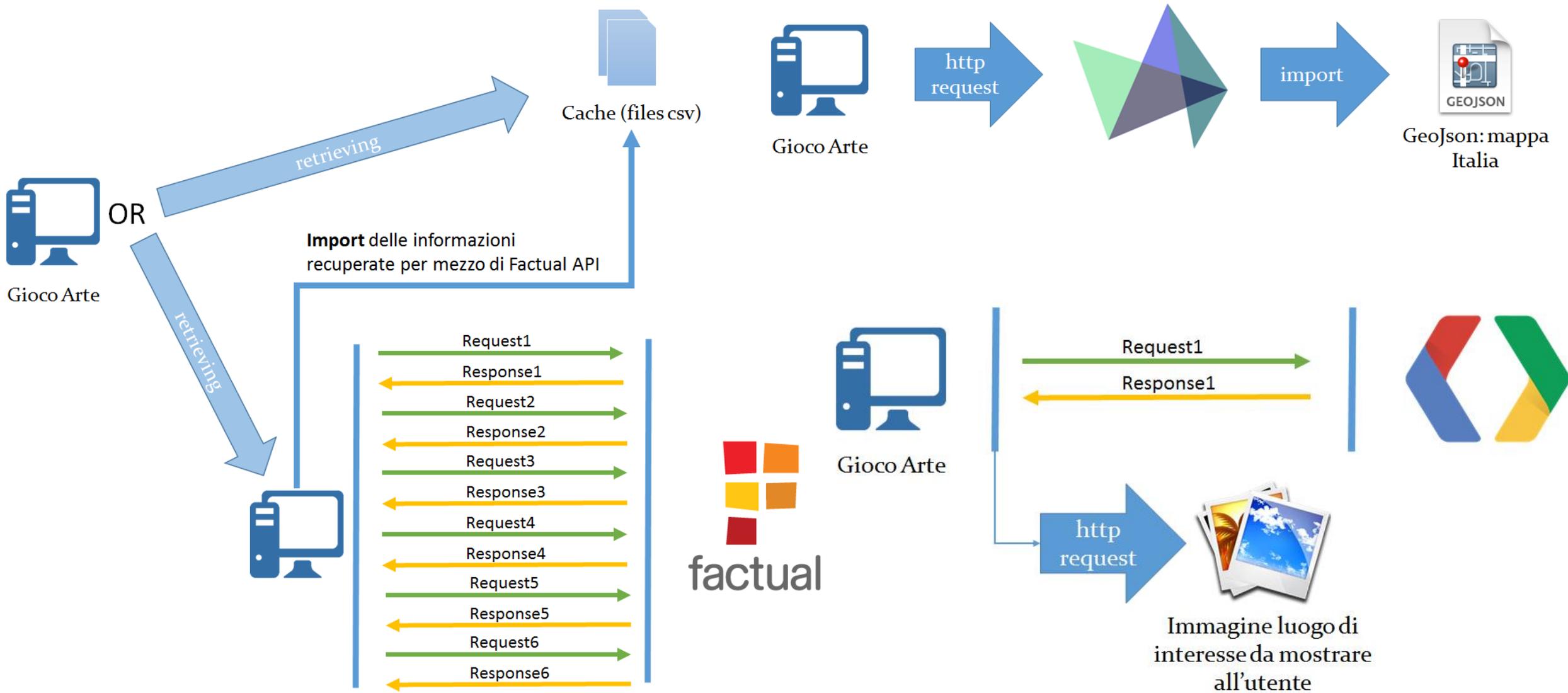
Estensione delle associazioni tra luoghi generici di interesse e la regione di appartenenza.



# Processo di esecuzione



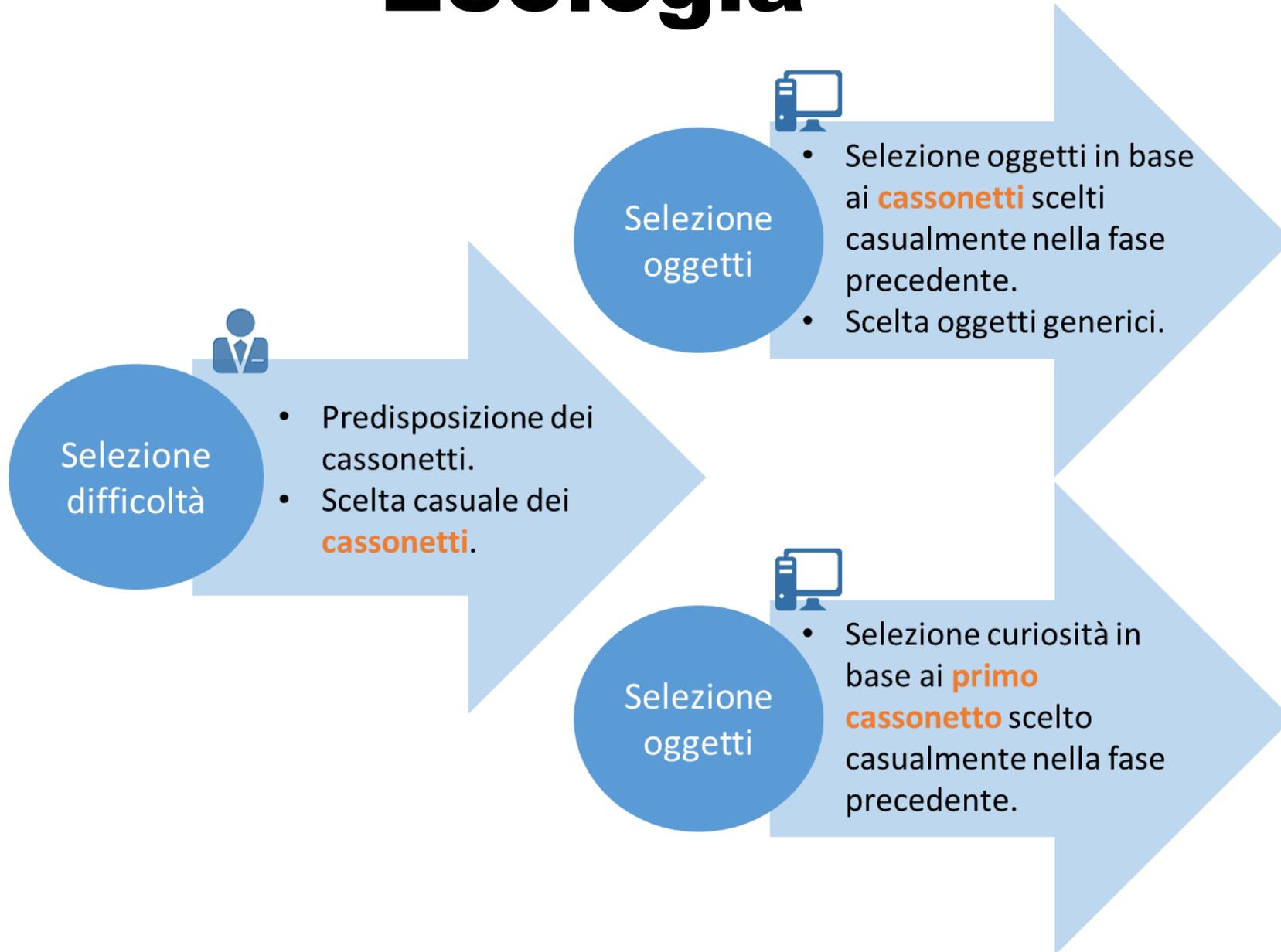
# Processo di estrazione dati



# JOINT



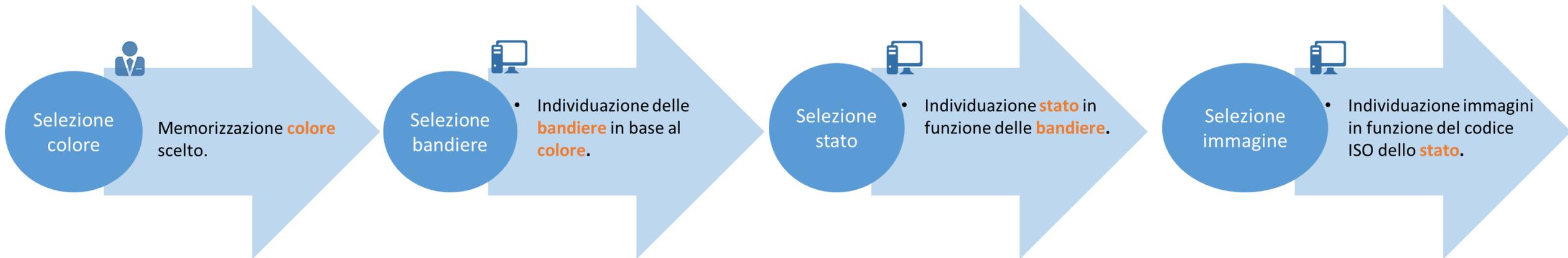
# Ecologia



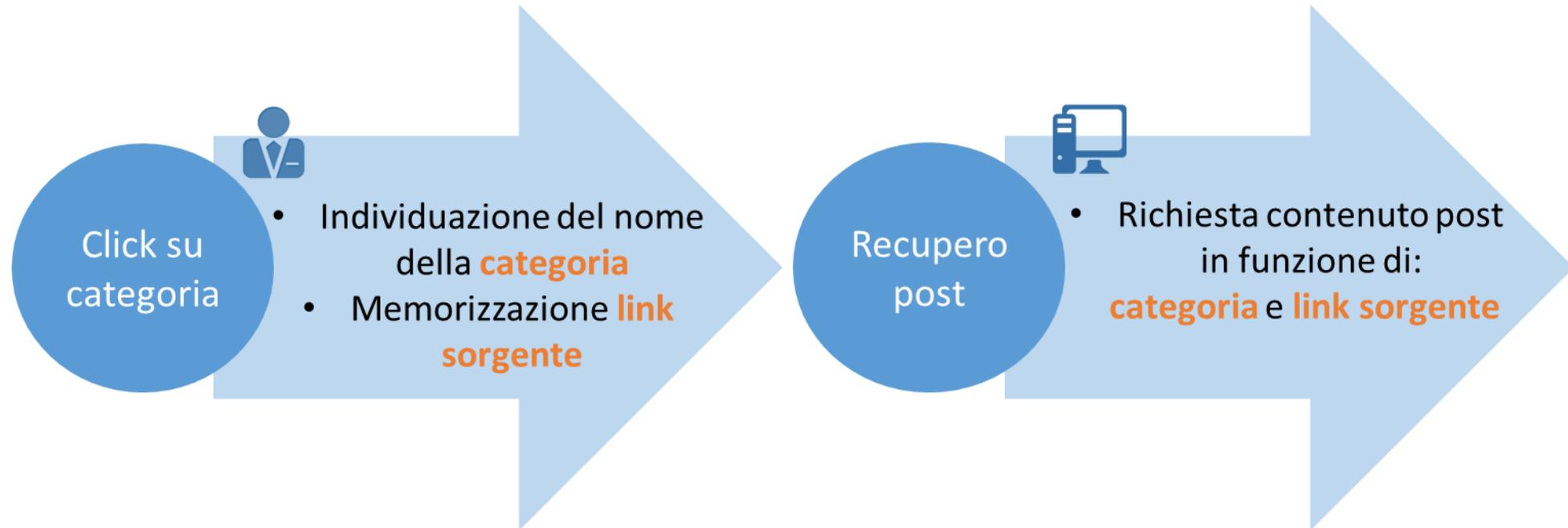
# Cucina



# Geografia



# News



# Musica



# Arte

Click su  
luogo



- Individuazione della **provincia**
- Memorizzazione **nome**  
**luogo**

Click su  
provincia



- Individuazione della **provincia**
- Controllo correttezza  
associazione

Recupero  
img



- Richiesta di img in  
funzione del **luogo**

# Schema

Locale

Globale

# Ecologia

## Fonte Differenziata

*oggetto\_cassonetto(cassonetto, oggetto).*

## Fonte Comune di Torino

*lo\_sapevate\_che(curiosita)*

## Fonte Alisea2000

*altri\_rifiuti(curiosita)*

## Fonte Eco-Ricicli

*riciclare\_vetro(curiosita)*

## Fonte Ecoo

*riciclo\_creativo(nome\_lavoretto, descrizione\_lavoretto)*

## Fonte IdeeGreen

*smaltimento\_umido(curiosita)*

## Fonti Differenziata

*ecologia(oggetto, cassonetto)*

## Fonti Comune di Torino, Alisea2000, Eco-Ricicli, Ecoo ed IdeeGreen

*curiosita\_ecologia(cassonetto, curiosita)*

locale

globale

# Cucina

## Fonte GialloZafferano

*ricetta\_regione*(nomeRicetta, immagine, difficoltà, preparazione, cottura, persone, costo, descrizione, ingredienti, ricetta, consiglio, regione)

## Fonte Mysia

*ingredienti*(nome\_ingrediente, curiosita).

locale

globale

## Fonte GialloZafferano

*ricette*(regione, nome, difficoltà, tempoPreparazione, tempoCottura, nPersone, costo, descrizione, ingredienti, consiglio)

## Fonte Mysia

*ingredienti*(nome, curiosita)

# Geografia

## Fonte Nations Online

*english\_italiano*(country, paese, local\_name)

## Fonte Wikipedia ISO alpha-2

*iso*(numero, codice, stato, note)

## Fonte Wikipedia Lista Bandiere Nazionali

*bandiera*(stato, descrizione, storia, note, altri\_progetti, collegamenti, proporzioni, simbolo\_fiav, colori, tipologia, adozione, nazione, altre\_bandiere)

## Fonte Wikipedia Lista delle capitali del mondo

*stati\_indipendenti*(stato, capitale, anno)

*stati\_autoproclamati*(stato, capitale, anno)

## Fonte GitHub

Ha una cartella flag/4x3 con le immagini di tutte le bandiere del mondo.

## Fonte Wikipedia Lista Bandiere Nazionali

*bandiere*(stato, colori, colori\_coded)

## Fonti Nations Online, Wikipedia ISO alpha-2, Wikipedia Lista delle Capitali del Mondo

*stato*(nome, capitale, nome\_eng, code)

locale

globale

### \*Fonte: HighMaps API

sia nello schema locale che globale. Sarà spiegato in seguito il suo schema.

# News

**Fonte: *import.io API***

Campi restituiti nel recupero dell'elenco dei post:

```
importIO( link, link_prices, fieldmain_link,  
           fieldmain_image, photocredit_label,  
           photocredit_description,  
           photocredit_description_prices,  
           content_contents, content_contents_prices,  
           content_links, noreadmore_link,  
           readfull_label, readfull_value,  
           comments_link)
```

Campi restituiti nel recupero del contenuto dei post:

```
importIOPost( title, content, fieldmain_image,  
              photocredit_label, photocredit_description,  
              photocredit_description_prices,)
```

**Fonte: *import.io API***

```
informazioniPost(link, title, image, linkSource, timestamp)
```

```
contenutoPost(title, image, content)
```

locale

globale

# Musica

## Fonte La Bandiera

`innoL(stato, inno)`

## Fonte: HighMaps API

`featuresL(type, id, properties, geometry)`

`propertiesL( hc-key, hc-a2, labelrank, hasc, alt-name, name, country, region, longitude, latitude)`

`geometryL(type, coordination)`  
`coordinationL(x, y)`

locale

globale

## Fonte La Bandiera

`innoG(stato, inno)`

## Fonte: HighMaps API

`featuresL(type, id, properties, geometry)`

`propertiesL( hc-key, hc-a2, labelrank, hasc, alt-name, name, country, region, longitude, latitude)`

`geometryL(type, coordination)`  
`coordinationL(x, y)`

# Arte

## Fonte: Factual API

*factual*( Id\_factual, nome\_monumento, indirizzo, città, regione[stato,provincia], codice\_postale, nazione, vicini, telefono, sito\_web, latitudine, longitudine, orari, categoria)

## Fonte: Google Custom Search API

*google*(displayLink, fileFormat, htmlSnippet, htmlTitle, image, kind, link, mime, snippet, title)

## Fonte: HighMaps API

*featuresL*(type, id, properties, geometry)  
*propertiesL*( hc-key, hc-a2, labelrank, hasc, alt-name, name, country, region, longitude, latitude)  
*geometryL*(type, coordination)  
*coordinationL*(x, y)

## Fonte: Factual API e Google Custom Search API

*monumento*(nome\_monumento, regione, orari, sito\_web, telefono, src\_img)

## Fonte: HighMaps API

*featuresG*(type, id, properties, geometry)  
*propertiesG*( hc-key, hc-a2, labelrank, hasc, alt-name, name, country, region, longitude, latitude)  
*geometryG*(type, coordination)  
*coordinationG*(x, y)

locale

globale

# Wrappers

# Ecologia

Fonte [http://www.comune.torino.it/ambiente/rifiuti/cosa\\_rifiuti/rifiuto-indifferenziato-non-recuperabile.shtml](http://www.comune.torino.it/ambiente/rifiuti/cosa_rifiuti/rifiuto-indifferenziato-non-recuperabile.shtml)

*Selezione curiosità(css path)* → `p:nth-of-type(n+5)`

Fonte

<http://www.differenziata.org/>

*Selezione rifiuti(css path)* → `:div.Block:nth-of-type(1)`  
`ul.menu_menu a`



# Ecologia

**Fonte**

<http://www.alisea2000.it/raccolta-differenziata/altri-rifiuti/>

*Selezione curiosità (css path) → div.paragraph-block  
p:nth-of-type(2)*

**Fonte**

<http://www.eco-ricicli.it/riciclare-vetro.htm>

*Selezione curiosità (css path) → h2, td td p:nth-of-type(1)*

**Fonte**

<http://www.ideegreen.it/smaltimento-del-rifiuto-umido-15555.html>

*Selezione curiosità (css path) → p:nth-of-type(n+3)*



# Ecologia

**Fonte**

<http://www.ecoo.it/articolo/riciclo-creativo-plastica-10-idee-facili-e-veloci-foto/24227/>

*Selezione curiosità (css path) → div.the\_content p:nth-of-type(n+3)*

**Fonte**

<http://www.ecoo.it/articolo/riciclo-creativo-della-carta-dei-quotidiani-e-giornali-10-idee-originali-foto/27287/>

*Selezione curiosità (css path) → p*



# Cucina

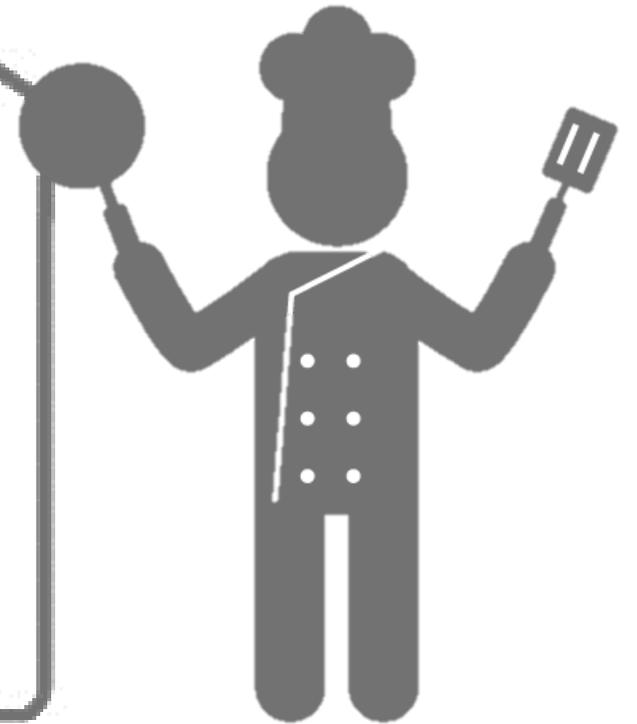
Fonte

<http://www.giallozafferano.it/ricette-cat/regionali>

*Selezione regione (css path)* → `li:nth-of-type(n+2) a.filter`

*Selezione ricetta (css path)* → `h2 a`

*Selezione informazioni ricetta (css path)* → `h1 .fn,`  
`li.difficolta strong, li.preptime strong, li.cooktime strong,`  
`li.yield strong, li.costo strong, div.intro p, dd.ingredienti a,`  
`div.consiglio p`



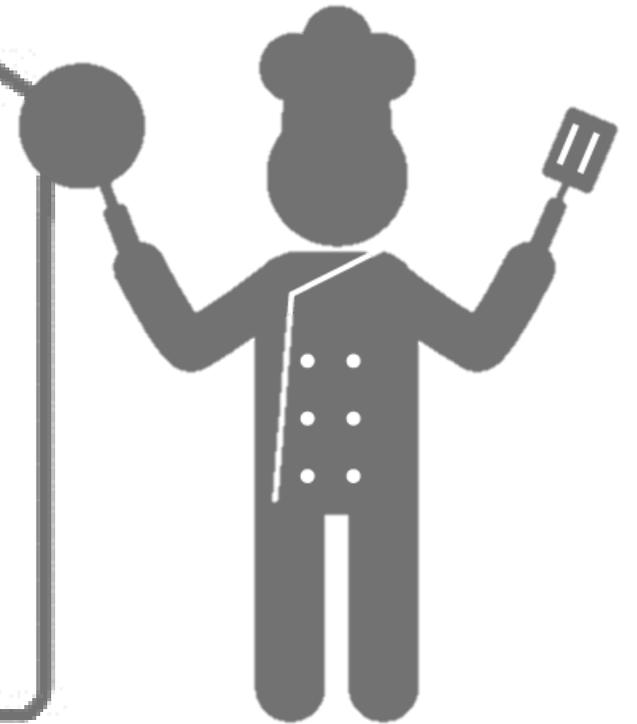
# Cucina

Fonte

<http://www.misya.info/elenco-ingredienti>

*Selezione ingredienti (css path)* → li.In-a a, li.In-b a, li.In-c a, li.In-d a, li.In-e a, li.In-f a, li.In-g a, li.In-h a, li.In-i a, li.In-j a, li.In-k a, li.In-l a, li.In-m a, li.In-n a, li.In-o a, li.In-p a, li.In-q a, li.In-r a, li.In-s a, li.In-t a, li.In-u a, li.In-v a, li.In-w a, li.In-x a, li.In-y a, li.In-z a

*Selezione curiosità ingrediente (css path)* → p



# Geografia



## Fonte

[https://it.wikipedia.org/wiki/Capitali degli Stati del mondo](https://it.wikipedia.org/wiki/Capitali_degli_Stati_del_mondo)

*Selezione tabella (css path)* → table.wikitable

*Selezione header(css path)* → thead tr

*Selezione row (css path)* → tbody tr

## Fonte

<http://www.nationsonline.org/>

*Selezione tabella (css path)* → table.tb86 tr:nth-of-type(n+3)

*Selezione nome inglese (css path)* → td:nth-of-type(1)

*Selezione nome italiano (css path)* → td:nth-of-type(2)

# Geografia



## Fonte

[https://it.wikipedia.org/wiki/  
ISO\\_3166-1\\_alpha-2](https://it.wikipedia.org/wiki/ISO_3166-1_alpha-2)

*Selezione tabella (css path)* → `tr:nth-of-type(n+2)`

*Selezione codice (css path)* → `td:nth-of-type(2)`

*Selezione nome stato (css path)* → `td:nth-of-type(3)`

## Fonte

[https://it.wikipedia.org/wiki/  
Lista\\_di\\_bandiere\\_nazionali](https://it.wikipedia.org/wiki/Lista_di_bandiere_nazionali)

*Selezione tabella (css path)* → `table:nth-of-type(n+2) td`

*Selezione bandiera (css path)* → `a:nth-of-type(2)`

*Selezione colori (css path)* → `table.sinottico`

`tr:contains('Colori') td`

# News



## *Import.io API*

Richiesta GET alla url:

<https://api.import.io/store/connector>

### *Parametri:*

**apikey** = developer key

**connector** = /connector/id dello scraper generato per recuperare i dati

**query** = webpage/url da cui effettuare lo scraping

# Musica

Fonte

<http://www.labandiera.com/inni/Home.htm>

*Download degli inni(css path) → css path: td>a*

I file sono salvati in una cartella.



# Arte

Factual API

Richiesta GET alla url:

<https://api.factual.com/t/places-it>

*Parametri:*

**KEY**= developer key

**filters**= oggetto che consente di filtrare i risultati

**filters.country**= paese rispetto al quale valutare i luoghi di interesse

**filters.category\_labels**= lista di categorie (monumenti, arte, memoriali, etc)

**limit**= numero massimo di tuple restituite

**offset**= discostamento dall'inizio, da cui iniziare a restituire i valori.

# Arte

Google Custom  
Search API

Richiesta GET alla url:

<https://www.googleapis.com/customsearch/v1>

**Parametri:**

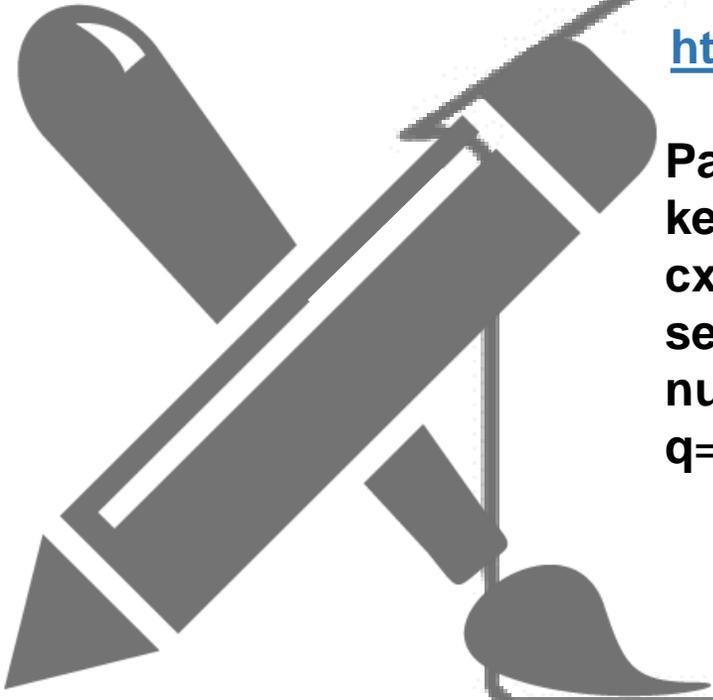
**key**= google developers key

**cx**= world sensitive custom search key

**searchType**= tipologia richiesta da effettuare

**num**= numero di valori da restituire

**q**= query

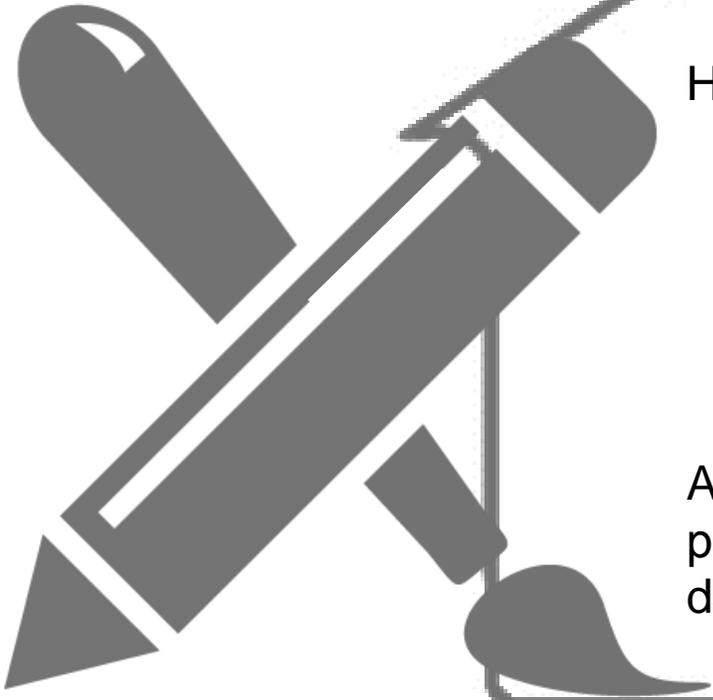


# Arte

## HighMaps API

Highcharts.geojson  
(Highcharts.maps[<codice\_mappa\_desiderata>])

Allo stesso modo vengono recuperate le mappe per musica e geografica, con l'unica differenza data dal codice della mappa che si desidera recuperare.



GAV - LAV



# Ecologia GAV

*ecologia(oggetto, cassonetto):- oggetto\_cassonetto(cassonetto, oggetto).*

*curiosita\_ecologia(cassonetto, curiosita):- lo\_sapevate\_che(curiosita), casonetto=="indifferenziato".*

*curiosita\_ecologia(cassonetto, curiosita):- altri\_rifiuti(curiosita), cassonetto=="ecocentro".*

*curiosita\_ecologia(cassonetto, curiosita):- riciclare\_vetro(curiosita), cassonetto=="vetro".*

*curiosita\_ecologia(cassonetto, curiosita):- riciclo\_creativo(curiosita), cassonetto=="plastica".*

*curiosita\_ecologia(cassonetto, curiosita):- riciclo\_creativo(curiosita), cassonetto=="carta".*

*curiosita\_ecologia(cassonetto, curiosita):- smaltimento\_umido(curiosita), cassonetto=="umido".*



# Ecologia LAV

*oggetto\_cassonetto(cassonetto, oggetto):- ecologia(oggetto, cassonetto).*

*lo\_sapevate\_che(curiosita):- curiosita\_ecologia(cassonetto, curiosita), cassonetto=="indifferenziato".*

*altri\_rifiuti(curiosita):- curiosita\_ecologia(cassonetto, curiosita), cassonetto=="ecocentro".*

*riciclare\_vetro(curiosita):- curiosita\_ecologia(cassonetto, curiosita), cassonetto=="vetro".*

*riciclo\_creativo(nome\_lavoretto, descrizione\_lavoretto):-*

*curiosita\_ecologia(cassonetto, descrizione\_lavoretto),*

*cassonetto=="plastica".*

*riciclo\_creativo(nome\_lavoretto, descrizione\_lavoretto):-*

*curiosita\_ecologia(cassonetto, descrizione\_lavoretto), cassonetto=="carta".*

*smaltimento\_umido(curiosita):-*

*curiosita\_ecologia(cassonetto, descrizione\_lavoretto), cassonetto=="umido".*



# Cucina GAV

*ricette*(*regione, nome, difficolta, tempoPreparazione, tempoCottura, nPersone, costo, descrizione, ingredienti, consiglio*) :-

*ricette\_regione*(*nome, \_ , difficolta, tempoPreparazione, tempoCottura, nPersone, costo, descrizione, ingredienti, \_ , consiglio, regione*).

*ingredienti*(*nome\_ingrediente, curiosita*) :-

*ingredienti* (*nome\_ingrediente, curiosita*).



# Cucina LAV

*ricette\_regione(nomeRicetta, immagine, difficolta, preparazione, cottura, persone, costo, descrizione, ingredienti, ricetta, consiglio, regione) :-*

*ricette(regione, nome, difficolta, tempoPreparazione, tempoCottura, nPersone, costo  
descrizione, ingredienti, consiglio).*

*ingredienti(nome\_ingrediente, curiosita) :-*

*ingredienti(nome\_ingrediente, curiosita).*



# Geografia GAV

*bandiere(stato, colori, colori\_coded) :-*  
  stati\_indipendenti(stato,\_,\_),  
  stati\_autoproclamati(stato,\_,\_),  
  bandiera(stato, \_,\_,\_,\_,\_,\_, colori,\_,\_,\_,\_),  
  colori\_coded = codifica(colori)\*.

*stato(nome, capitale, nome\_eng, code) :-*  
  stati\_indipendenti(nome, capitale, \_),  
  stati\_autoproclamati(nome, capitale, \_),  
  english\_italiano(\_, nome, nome\_eng),  
  iso(\_,code,nome,\_).

\**codifica(colori)* è una semplificazione della computazione necessaria per restringere il range dei colori (a partire da un codice RGB si ricava il nome di uno dei colori basici). In tale computazione dal campo colori (una lista di colori) vengono estratti i valori RGB dei singoli colori. Viene, poi, calcolato il valore HSL del colore ed in base al risultato viene individuato il range. In seguito, viene ricomposta la lista e memorizzata come un elenco.



# Geografia LAV

*english\_italiano(country, paese, local\_name) :-  
stato(paese, local\_name, \_).*

*iso(numero, codice, stato, note) :-  
stato(nome, \_, \_, codice).*

*bandiera(stato, descrizione, storia, note, altri\_progetti, collegamenti, proporzioni, simbolo\_fiav, colori, tipologia,  
adozione, nazione, altre\_bandiere) :-  
stato(stato, \_, \_, \_),  
bandiere(stato, colori, \_).*

Le informazioni riguardo gli *stati indipendenti* e *stati autoproclamati* non sono più reperibili distintamente in quanto vengono conservate come *stati*.



# News GAV

*informazioniPost*(link, title, image, linkSource, timestamp) :-

*importIO*( link, \_, \_, image, \_, \_, \_, title, \_, \_, linkSource, \_, \_, \_).

*contenutoPost*(title, image, content) :-

*importIOPost*( title, content, image, \_, \_, \_).



# News LAV

```
importIO( link, link_prices, fieldmain_link, image, photocredit_label,  
          photocredit_description, photocredit_description_prices, title,  
          content_contents_prices, content_links, linkSource, readfull_label,  
          readfull_value, comments_link ) :-  
    informazioniPost(link, title, image, linkSource).
```

```
importIOPost( title, content, fieldmain_image, photocredit_label, photocredit_description,  
             photocredit_description_prices ) :-  
    contenutoPost(title, image, content).
```



# Musica GAV & LAV

## GAV

innoG(stato, inno) :- innoL(stato, inno).

## LAV

innoL(stato, inno):- innoG(stato, inno)

Inoltre la sezione musica fa uso della mappa del mondo, esattamente come la sezione geografia.



# Arte GAV

```
monumento(nome_monumento, regione, orari, sito_web, telefono, src_img) :-  
  factual( _, nome_monumento, _, _, regione, _, nazione, _, telefono, sito_web, _, _,orari, categoria),  
  nazione == "it",  
  categoria == (("arts"&&"museum") || ("landmarks"&&"monuments and memorials") ||  
               ("community and government" &&"religious" &&"churches") ||  
               ("landmarks"&&"parks"&&"natural parks")),  
  google( _, _, _, _, _, _, src_img, _, _, nome_monumento).
```

```
featuresG(type, id, properties, geometry) :-
```

```
  featuresL("Italy", id, properties, geometry).
```

```
propertiesG( hc-key, hc-a2, labelrank, hasc, alt-name, name, country, region, longitude, latitude):-
```

```
  propertiesL( hc-key, hc-a2, labelrank, hasc, alt-name, name, country, region, longitude, latitude).
```

```
geometryG(type, coordination) :- geometryL(type, coordination).
```

```
coordinationG(x, y) :- coordinationL(x, y).
```



# Arte LAV

*factual( Id\_factual, nome\_monumento, indirizzo, città, regione, codice\_postale, nazione, vicini, telefono, sito\_web, latitudine, longitudine, orari, categoria) :-  
monumento(nome\_monumento, regione, orari, sito\_web, telefono, src\_img).*

*google(displayLink, fileFormat, htmlSnippet, htmlTitle, image, kind, link, mime, snippet, nome\_monumento) :-  
monumento(nome\_monumento, regione, orari, sito\_web, telefono, link).*

*featuresL(type, id, properties, geometry) :-  
featuresG(type, id, properties, geometry).*

*propertiesL( hc-key, hc-a2, labelrank, hasc, alt-name, name, country, region, longitude, latitude):-  
propertiesG( hc-key, hc-a2, labelrank, hasc, alt-name, name, country, region, longitude, latitude).*

*geometryL(type, coordination) :- geometryG(type, coordination).*

*coordinationL(x, y) :- coordinationG(x, y).*





# Ecologia SQL

Per quanto riguarda SQL abbiamo preferito inserire l'intero processo per la costruzione della query per dimostrare che la clausola *where* dipende dai parametri ricevuti in input, quali i cassonetti selezionati in modo random: sia il numero che la tipologia dei cassonetti non è nota apriori.

```
//selezione oggetti corretti
```

```
$numero_cassonetti = sizeof($cassonetti);
```

```
for($i=0; $i<$numero_cassonetti; $i++){
```

```
    $where = $where . "cassonetto=" . $cassonetti[$i] . "";
```

```
    if($i<($numero_cassonetti-1))
```

```
        $where = $where . " || ";
```

```
}
```

```
$query = "SELECT oggetto, cassonetto FROM ecologia WHERE " . $where;
```

```
//selezione oggetti corretti e sbagliati
```

```
$query2 = "SELECT oggetto, cassonetto FROM ecologia";
```

# Ecologia Datalog

*oggetti\_plastica(oggetto, cassonetto):- ecologia(oggetto, cassonetto), cassonetto=="plastica".*

*oggetti\_carta(oggetto, cassonetto):- ecologia(oggetto, cassonetto), cassonetto=="carta".*

*oggetti\_umido(oggetto, cassonetto):- ecologia(oggetto, cassonetto), cassonetto=="umido".*

*oggetti\_vetro(oggetto, cassonetto):- ecologia(oggetto, cassonetto), cassonetto=="vetro".*

*oggetti\_ecocento(oggetto, cassonetto):- ecologia(oggetto, cassonetto), cassonetto=="ecocentro".*

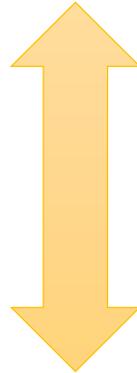
*oggetti\_indifferenziata(oggetto, cassonetto):- ecologia(oggetto, cassonetto), cassonetto=="indifferenziata".*

*oggetti(oggetto, cassonetto):- ecologia(oggetto, cassonetto).*

# Geografia

SQL

```
$query = "SELECT stato, code, capitale, nome_eng  
FROM bandiere INNER JOIN stato ON bandiere.stato = stato.nome  
WHERE colori_coded like '%" . $name . "%";
```

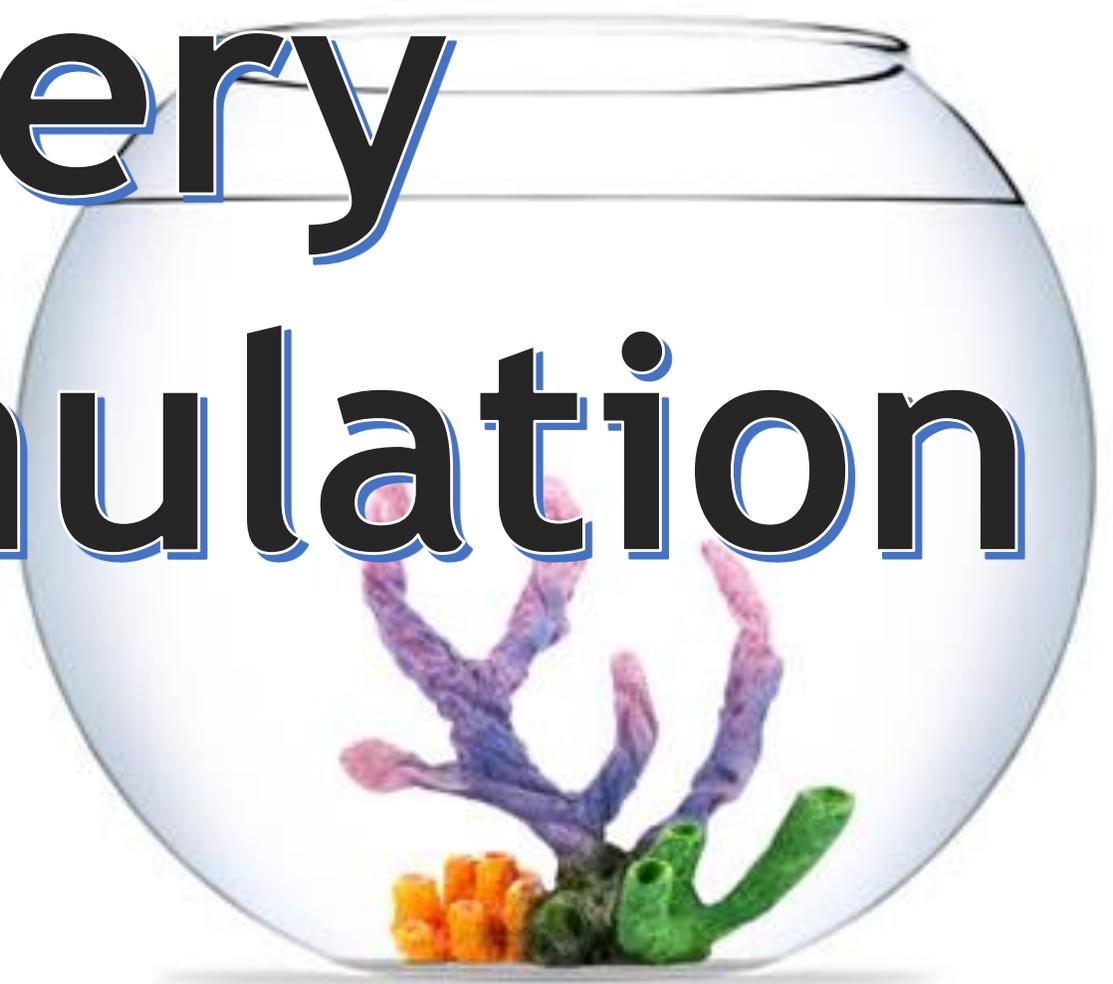


```
selezione_bandiere(stato, code, capitale, nome_eng) :-  
bandiere(stato, _, colori),  
stato(stato, capitale, nome_eng, code),  
colori == "colore".
```

Datalog

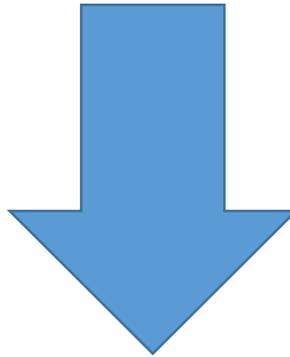


# Query Reformulation



# Unfolding

*ingredienti\_ricetta\_tipica(nome\_ingrediente, curiosita):-*  
ingredienti(nome\_ingrediente, curiosita),  
ricette(\_, nome, \_, \_, \_, \_, nome\_ingrediente, \_),  
nome == "ricetta tipica".

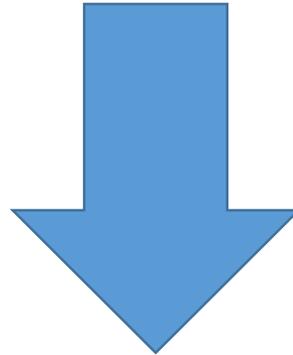


**Applicando l'unfolding:**

*ingredienti\_ricetta\_tipica(nome\_ingrediente, curiosita):-*  
ingredienti (nome\_ingrediente, curiosita),  
ricette\_regione(nome, \_, \_, \_, \_, \_, nome\_ingrediente, \_, \_, \_),  
nome == "ricetta tipica".

# Bucket

ingredienti\_ricetta\_tipica(nome\_ingrediente, curiosita):-  
ingredienti(nome\_ingrediente, curiosita),  
ricette(\_, nome, \_, \_, \_, \_, nome\_ingredienti, \_).  
nome == "ricetta tipica".



**Query Candidata**

ingredienti\_ricetta\_tipica(nome\_ingrediente, curiosita):-  
ingredienti(nome\_ingrediente, curiosita),  
ricette\_regione(nome, \_, \_, \_, \_, nome\_ingrediente, \_, \_),  
nome== "ricetta tipica".

# Bucket

Dobbiamo verificare se  $Q1 \subseteq Q2$ . Con Q1 query candidata e Q2 query iniziale.



*estensione Q1*

ingredienti\_ricetta\_tipica(nome\_ingrediente, curiosita):-  
    ingredienti(nome\_ingrediente, curiosita),  
    ricette(A, nome, B, C, E, F, G, H, nome\_ingrediente, I),  
    nome== "ricetta tipica".

*Le variabili A-I sono variabili esistenziali.*

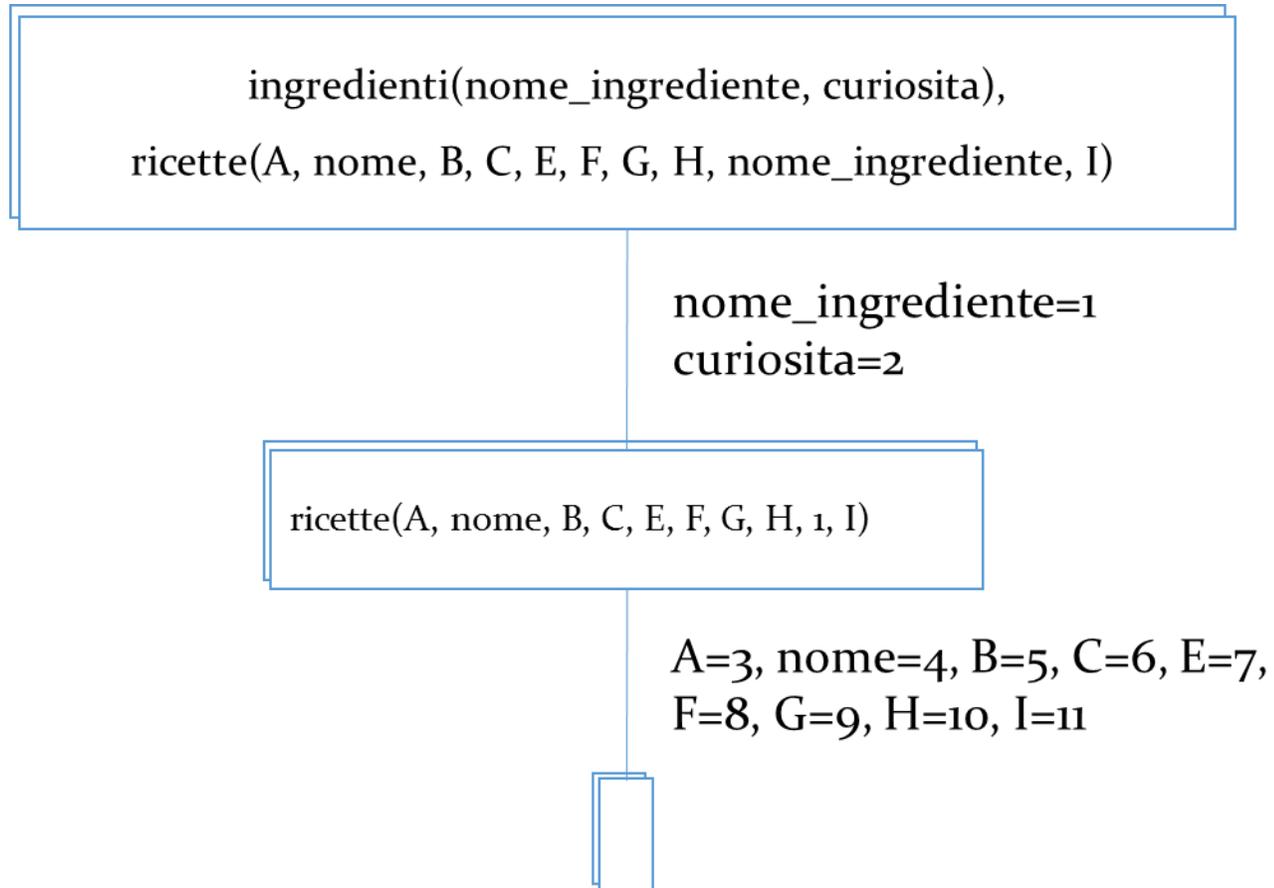
## Database canonico di Q1

*nome\_ingrediente=1, curiosita=2, A=3, nome=4, B=5, C=6, E=7, F=8, G=9, H=10, I=11*

*D={ingredienti(1,2), ricette(3, 4, 5, 6, 7, 8, 9, 10, 1, 11)}*

La frozen head di Q1 è Q1(1, 2).

# Bucket



**ans(Q2, frozen(Q1)) = {(1,2)}**

La frozen head di Q1 è contenuta nello spazio delle soluzioni di Q2 computata sul database canonico di Q1, quindi  $Q1 \subseteq Q2$ .

# Tecnologie



# Tecnologie Utilizzate



## Linguaggi di programmazione:

- PHP
- JavaScript

## Formato trasferimento/memorizzazione dati:

- XML
- JSON
- CSV

## API:

- Factual API
- HighMaps API
- Google Custom Search API
- import.io API

## Altre tecnologie di scraping impiegate:

- Web Scraper e suo utilizzo tramite CSSPath



Alessia  
Antelmi



Martina  
Garofalo

THANKS



Enrica  
Loria



Maria Angela  
Pellegrino