FIND MY GAME

GAD
Prof. Gennaro Costagliola
A.A. 2015/2016

Donato Concilio
Francesco Orciuoli

SPECIFICA DEL PROBLEMA

Offrire un servizio di ricerca riguardante i videogiochi per le console old gen(Xbox 360, Ps3), next gen (Ps4,Xbox One) e per Pc. Rispetto ad altri motori di ricerca compara recensioni, prezzi e voti del gioco cercato

- Vendita videogiochi delle principali piattaforme:
- Amazon
- Ebay

- Informazioni Specifiche sui Videogiochi:
- Multiplayer
- EveryEye

DESCRIZIONE FUNZIONALE

Find My Game:

- Informazioni aggiornate sui videogiochi
- Possibiltà di ricerca per Nome, Piattaforma e Genere
- Possibilità di ricerca in combinazioni multiple su Piattaforme e Genere
- Informazioni aggiuntive ricavate da Amazon e Ebay
- Intefaccia user-friendly

Firefox File Modifica Visualizza Cronologia Segnalibri Strumenti Finestra Aiuto 65% Mer 10:17

Home | Find My Game

localhost/gadProject/FindMyGame/index.php

Cerca

FIND MY GAME

Search

GENERI

- TUTTI
- AZIONE
- AVVENTURA
- CASUAL
- GESTIONALE
- GIOCO DI GUIDA
- GIOCO DI RUOLO
- ONLINE
- PICCHIADURO
- PLATFORM
- PUZZLE
- SIMULAZIONE
- SPARATUTTO
- SPORTIVO

TUTTE PC PS4 PS3 XBOX 360 XBOX ONE

Double Dragon Trilogy

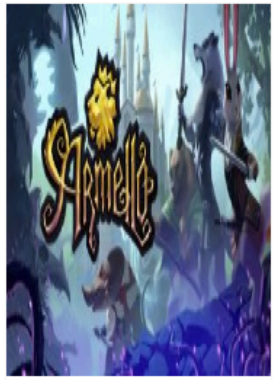
Dettagli

Resident Evil

Dettagli

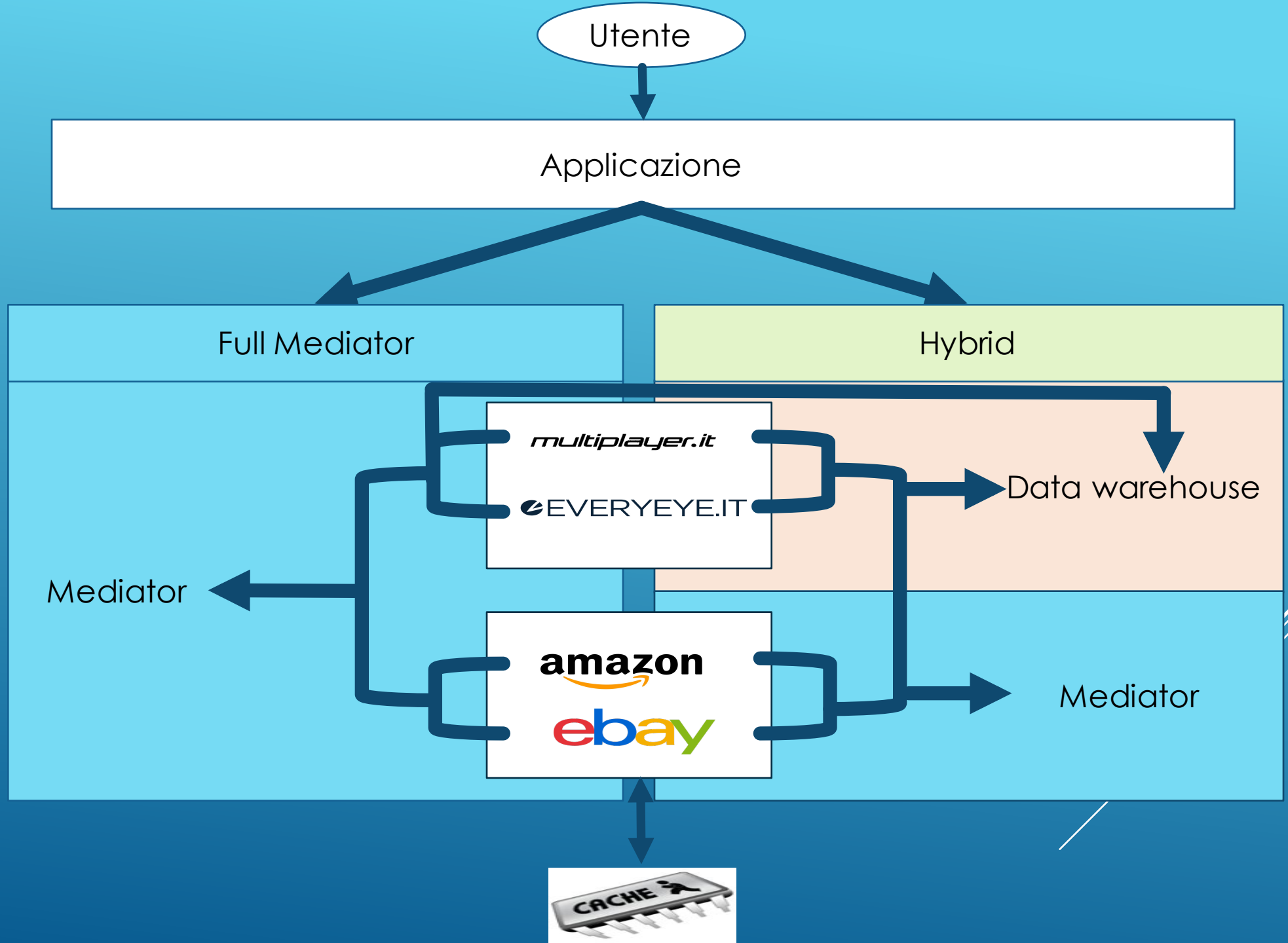
Assassin's Creed Unity: Segreti della Rivoluzione

Dettagli



Armello

Dettagli


Utente
Applicazione
Full Mediator
Hybrid
multiplayer.it
EVERYEYE.IT
Data warehouse
Mediator
amazon
ebay
Mediator
CACHE

ARCHITETTURA DEL SERVIZIO

L'approccio usato è quella dell'architettura ibrida.

- Utilizziamo l'approccio mediator per due fonti recuperando i prezzi per tutte le piattaforme disponibili
- Utilizziamo l'approccio data warehouse per le altre due fonti che riguardano i videogiochi e le loro informazioni

Vista la volatilità delle sorgenti il data warehouse è aggiornato tramite E.T.L. in due modi possibili:

1)

- Inzialmente viene eseguito l'E.T.L. completo per popolare il database con tutti i giochi
- Giornalmente viene eseguito l'E.T.L. solo sul mese corrente scartando tutti i giochi già inseriti

2)

- Se il gioco cercato non è presente nel data warehouse viene recuperato tramite mediator dalle fonti e inserito nel database

Inoltre è prevista una cache per i dati delle fonti Mediator (Amazon e Ebay) riempita quando un videogioco viene selezionato e che ha validità giornaliera

NUMERO E TIPO DELLE FONTI

Le fonti utilizzate sono quattro:

- www.amazon.it è un sito di e-commerce e viene utilizzato tramite API filtrando i risultati con la categoria Videogiochi. E' una fonte che subisce aggiornamenti giornalieri in questo ambito
- www.ebay.it è un sito di e-commerce e viene utilizzato tramite API filtrando i risultati con la categoria Videogiochi. E' una fonte che subisce aggiornamenti giornalieri in questo ambito
- www.multiplayer.it è un sito di videogiochi e viene utilizzato tramite le API di import.io ed è una fonte che subisce aggiornamenti giornalieri riguardanti i giochi usciti più recentemente
- www.everyeye.it è un sito di videogiochi e viene utilizzato tramite le API di import.io ed è una fonte che subisce aggiornamenti giornalieri riguardanti i giochi usciti più recentemente

WRAPPER SVILUPPATO PER AMAZON

Amazon APi

Richiesta GET alla url:

endpoint = "webservices.amazon.it";

Parametri:

aws_access_key_id = acces_key;

aws_secret_key = secret_key;

uri = "/onca/xml";

"Service" = "AWSECommerceService",

"Operation" = "ItemSearch",

"AWSAccessKeyId" = "AKIAJHZYEYD6EWCKBYWQ",

"AssociateTag" = "prova0cb-21",

"SearchIndex" ="VideoGames",

"ResponseGroup" = "Offers,OfferFull,OfferListings,ItemAttributes",

"Sort" = "relevancerank",

"Keywords" = gioco da cercare,

"Version" = "2015-12-145"

WRAPPER SVILUPPATO PER EBAY

Ebay APi

Richiesta GET alla url:

http://svcs.ebay.com/services/search/FindingService/v1

Parametri:
operation-name= findItemsAdvanced
global-id= EBAY-IT
response-data-format= json

categoryId= 1249

itemFilter[].name=condition,listingType

itemFilter[].value=new, fixedPrice

Keywords= gioco da cercare

WRAPPER SVILUPPATO PER EVERYEYE

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

Sono state utilizzate 2 tipi di query:

1. Per recuperare tutti i giochi di uno specifico mese
2. Per recuperare il dettaglio di un gioco

WRAPPER SVILUPPATO PER MULTIPLAYER

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

Sono state utilizzate 2 tipi di query:

1. Per recuperare tutti i giochi di uno specifico mese
2. Per recuperare il dettaglio di un gioco

MERGE

Viene eseguito un merge tra i nomi dei videogiochi uguali recuperati dalle fonti Multiplayer ed EveryEye

- Se il gioco è presente solo su una delle due viene presentato solo con le informazioni della sua fonte
- Se è presente su entrambe le fonti il gioco viene presentato fondendo le informazioni

DATALOG SCHEMA LOCALE

Amazon(name,platform,price,link).

Ebay(name,platform,price,link).

MultiPlayer(name,date,platform,genre,publisher,image,vote,review).

EveryEye(name,date,platform,genre,publisher,image,vote,review).

DATALOG SCHEMA GLOBALE

VideoGame(name,date,platform,genre,publisher,image,vote,review).

Price(name,platform,price,link).

QUERY 1

JOINT 

Visualizzare i dettagli di un gioco

- findDetailGame(name,date,platform,genre,publisher,image,vote,review,price,link):- VideoGame(name,date,platform,genre,publisher,image,vote,review), Price(name,platform,price,link).

- Select *

From VideoGame Join Price

On VideoGame.Name=Price.Name

On VideoGame.Platform=Price.Platform

QUERY 2

Filtrare i giochi per piattaforma

- findGamesByPlatform(name,platform):- VideoGame(name,v1,platform,v2,v3,v4,v5,v6);
- Select Name From VideoGame Where Platform=P;

QUERY 3

Filtrare i giochi per genere

- findGamesByGenre(name,genre):- VideoGame(name,v1,v2,genre,v3,v4,v5,v6);
- Select Name From VideoGame Where Genre=G;

GAV MAPPING

1)

VideoGame(name,date,platform,genre,publisher,image,vote,review)

$\subseteq$

MultiPlayer(name,date,platform,genre,publisher,image,vote,review).

VideoGame(name,date,platform,genre,publisher,image,vote,review)

$\subseteq$

EveryEye(name,date,platform,genre,publisher,image,vote,review).

2)

Price(name,platform,price,link)

$\subseteq$

Amazon(name,platform,price,link), min(price).

Price(name,platform,price,link)

$\subseteq$

Ebay(name,platform,price,link), min(price).

UNFOLDING – ES QUERY 1

findDetailGame(name,date,platform,genre,publisher,image,vote,review,price,link):-
VideoGame(name,date,platform,genre,publisher,image,vote,review),

Price(name,platform,price,link).

1)

- findDetailGame(name,date,platform,genre,publisher,image,vote,review,price,link):-
  MultiPlayer(name,date,platform,genre,publisher,image,vote,review),

  Price(name,platform,price,link).

- findDetailGame(name,date,platform,genre,publisher,image,vote,review,price,link):-
  MultiPlayer(name,date,platform,genre,publisher,image,vote,review),

  Amazon(name,platform,price,link), min(price).

UNFOLDING – ES QUERY 1

2)

- findDetailGame(name,date,platform,genre,publisher,image,vote,review,price,link):- **MultiPlayer(name,date,platform,genre,publisher,image,vote,review)**,

  Price(name,platform,price,link).

- findDetailGame(name,date,platform,genre,publisher,image,vote,review,price,link):- MultiPlayer(name,date,platform,genre,publisher,image,vote,review),

  Ebay(name,platform,price,link),min(price).

UNFOLDING – ES QUERY 1

3)

- findDetailGame(name,date,platform,genre,publisher,image,vote,review,price,link):- **EveryEye(name,date,platform,genre,publisher,image,vote,review)**,

  Price(name,platform,price,link).

- findDetailGame(name,date,platform,genre,publisher,image,vote,review,price,link):- EveryEye(name,date,platform,genre,publisher,image,vote,review),

  Amazon(name,platform,price,link),min(price).

UNFOLDING – ES QUERY 1

4)

- findDetailGame(name,date,platform,genre,publisher,image,vote,review,price,link):-
  EveryEye(name,date,platform,genre,publisher,image,vote,review),

  Price(name,platform,price,link).

- findDetailGame(name,date,platform,genre,publisher,image,vote,review,price,link):-
  EveryEye(name,date,platform,genre,publisher,image,vote,review),

  Ebay(name,platform,price,link),min(price).

LAV MAPPING

MultiPlayer(name,date,platform,genre,publisher,image,vote,review) ⊇
VideoGame(name,date,platform,genre,publisher,image,vote,review)

EveryEye(name,date,platform,genre,detail,publisher,image,vote,review) ⊇
VideoGame(name,date,platform,genre,publisher,image,vote,review)

Amazon(name,platform,price,link) ⊇ Price(name,platform,price,link)

Ebay(name,platform,price,link) ⊇ Price(name,platform,price,link)

BUCKET ALGORITHM

findDetailGame (name,date,platform,genre,publisher,image,vote,review, price,link):-
VideoGame(name,date,platform,genre,publisher,image,vote,review),
Price(name,platform,price,link).

Bucket	VideoGame	Price
	MultiPlayer(name,date,platform,genre,publisher,image,vote,review) EveryEye(name,date,platform,genre,detail,publisher,image,vote,review)	Amazon(name,platform,price,link) Ebay(name,platform,price,link)

R1: Multiplayer(name, ...), Amazon(name,..,price,link), min(price)
R2: Multiplayer(name, ...), Ebay(name,..,price,link), min(price)
R3: EveryEye(name, ...), Amazon(name,…,price,link), min(price)
R4: EveryEye(name, ...), Ebay(name,…,price,link), min(price)

- Espandiamo R1 secondo il Mapping:

  R1: Multiplayer(name, ...), Amazon(name,...,price,link), min(price)

  R1exp :- **VideoGame(name,date,platform,genre,publisher,image,vote,review)**, Amazon(name,..,price,link), min(price)

  R1exp :- VideoGame(name,date,platform,genre,publisher,image,vote,review), **Price(name,platform,price,link)**

Query Containment per verificare se:

R1exp(name,date,platform,genre,publisher,image,vote,review,price,link)
⊆ findDetailGame(name,date,platform,genre,publisher,image,vote,review,price,link)

- R1exp(name,date,platform,genre,publisher,image,vote,review,price,link):- VideoGame(name,date,platform,genre,publisher,image,vote,review), Price(name,platform,price,link)

- findDetailGame (name,date,platform,genre,publisher,image,vote,review, price,link):- VideoGame(name,date,platform,genre,publisher,image,vote,review), Price(name,platform,price,link)

$R1exp \subseteq findDetailGame$

LINGUAGGI DI PROGRAMMAZIONE

- Client-side
  JavaScript, AJAX, HTML5, CSS

- Server-side
  Php, MySQL.

MULTI AJAX

```
var xhttp = new XMLHttpRequest();
  xhttp.onreadystatechange = function() {
    if (xhttp.readyState == 4 && xhttp.status == 200)
    {

     document.getElementById("detailGame").style.display = "";
     document.getElementById("detailGame").innerHTML = xhttp.responseText;
     document.getElementById("mainSection").style.display = "none";

     //chiamata ajax per mediator
     var xhttp2 = new XMLHttpRequest();
     xhttp2.onreadystatechange = function() {
     if (xhttp2.readyState == 4 && xhttp2.status == 200) {
              document.getElementById("comparaPrezzi").innerHTML = xhttp2.responseText;
            }
        };
    var idGame = document.getElementById("idGame").innerHTML;
    xhttp2.open("GET", "getPrezzi.php?id="+idGame, true);
    xhttp2.send();
    }
  };

  xhttp.open("GET", "getGame.php?name="+nameGame, true);
  xhttp.send();
```