TvPad - Raccolta dati dai social network

Analisi e implementazione

Sommario

[Nomenclatura 2](#_Toc411090118)

[Requisiti 3](#_Toc411090119)

[Requisiti funzionali 3](#_Toc411090120)

[Criteri di selezione dei contenuti 3](#_Toc411090121)

[Acquisizione dei contenuti 3](#_Toc411090122)

[Memorizzazione dei contenuti 3](#_Toc411090123)

[Architettura dell’applicazione 4](#_Toc411090124)

[Il modello dei dati 4](#_Toc411090125)

[Diagramma Entity-Relationship 7](#_Toc411090126)

[La logica dell’applicazione 8](#_Toc411090127)

[Architettura dell’applicazione 8](#_Toc411090128)

[I produttori 8](#_Toc411090129)

[I consumatori 9](#_Toc411090130)

[Lettura dei parametri di configurazione 9](#_Toc411090131)

[Stato dello sviluppo 10](#_Toc411090132)

[Informazioni generali sullo stato dello sviluppo 10](#_Toc411090133)

[Informazioni dettagliate sullo stato dello sviluppo 10](#_Toc411090134)

[Dowlad dei dati 10](#_Toc411090135)

[Criteri di selezione dei contenuti 11](#_Toc411090136)

[Contenuto cartelle e utilizzo 12](#_Toc411090137)

# Nomenclatura

|  |  |
| --- | --- |
| **Nome** | **Descrizione** |
| Social Network |  |
| Sorgente | Oggetto dal quale vengono prodotti contenuti. Può essere un social network in generale o una sua componente specifica (ad esempio una pagina facebook) |
| Contenuto | Dato prodotto da una sorgente. Può essere di diversi tipi (testo, link, foto…) |
| Criterio di selezione (filtro) | Modalità con la quale l’utente può definire quali contenuti soddisfino il suo bisogno informativo. Sono previsti 4 criteri di selezione   1. Selezione in base alle parole del contenuto 2. Selezione in base alla data di creazione del contenuto 3. Selezione in base al lingua del contenuto 4. Selezione in base alla località associata al contenuto |
| Modalità di acquisizione dei contenuti | Approccio utilizzato per l’acquisizione dei contenuti. Due modalità di acquisizione dei contenuti sono state definite   1. L’applicazione richiede i dati al Social Network (REST) 2. Il social network invia dati in tempo reale all’applicazione (STREAM) |
| Richiesta | Insieme dei criteri di selezione dei contenuti |
| Produttore | Componente software che gestisce l’acquisizione contenuti di una sorgente utilizzando una modalità di acquisizione |
| Consumatore | Componente software che s |
|  |  |

# Requisiti

## Requisiti funzionali

* L’utente deve poter memorizzare sul database le sorgenti
* L’utente deve poter memorizzare sul database le richieste
* All’avvio dell’applicazione l’utente deve poter selezionare la richiesta a cui è interessato.

## Criteri di selezione dei contenuti

* Il criterio di selezione dei contenuti basato sulle parole deve consentire all’utente di definire un’espressione booleana i cui termini sono parole chiave[[1]](#footnote-1)
* Il criterio di selezione dei contenuti basato sulla data di creazione del contenuto deve essere definito mediante di un intervallo temporale (data di inizio e data di fine)
* Il criterio di selezione basato sulla lingua (opzionale) del contenuto deve permettere all’utente di specifica una o più lingue[[2]](#footnote-2)
* Il criterio di selezione basato sulla località nella quale il contenuto è stato prodotto (opzionale) deve permettere all’utente di definire una località (es. Roma) oppure un insieme di coordinate geografiche[[3]](#footnote-3)

## Acquisizione dei contenuti

* L’applicazione deve poter acquisire i contenuti sia in modalità REST che STREAM
* L’utilizzo delle diverse modalità di acquisizione dei contenuti dipende dal criterio di selezione basato sulla data di creazione del contenuto
  + Se la data di inizio è successiva alla data corrente l’applicazione dovrà utilizzare la modalità di acquisizione STREAM[[4]](#footnote-4)
  + Se la data corrente è compresa tra la data di inizio e quella di fine
    - L’applicazione deve acquisire i contenuti creati a partire dalla data di inizio fino alla data corrente utilizzando la modalità REST[[5]](#footnote-5)
    - L’applicazione deve acquisire i contenuti creati a partire dalla data corrente fino alla data di fine utilizzando la modalità STREAM
  + Se la data di fine è precedente alla data corrente il l’applicazione deve acquisire i contenuti in modalità REST[[6]](#footnote-6)

## Memorizzazione dei contenuti

* I contenuti che soddisfano i criteri di selezione devono essere memorizzati su una base dati
* Per ogni contenuto memorizzare i seguenti dati devono essere memorizzati (compatibilmente con il social network da cui provengono):
  + Contenuto testuale
  + Link
  + Foto
  + Indice di gradimento
* Per ogni contenuto deve essere possibile individuare la richiesta e la sorgente che dalla quale è stato generato

# Architettura dell’applicazione

Per lo sviluppo dell’applicazione è stato scelto il pattern architturale Model-View-Control (MVC) in modo da tener separate la rappresentazione dei dati, la logica dell’applicazione e la presentazione[[7]](#footnote-7).

## Il modello dei dati

* Una richiesta deve contenere i seguenti dati

|  |  |  |
| --- | --- | --- |
| **Dato** | **Nullable** | **Nome sul database** |
| Nome della richiesta | NO | NAME |
| Descrizione della richiesta | NO | DESCRIPTION |
| Criterio di selezione basato su parole chiave | NO | KEYWORD\_FILTER |
| Criterio di selezione basato sul linguaggio | SI | LANGUAGE\_FILTER |
| Criterio di selezione basato sulla località | SI | LOCATION\_FILTER |
| Data d’inizio | NO | START DATE |
| Data di fine | NO | STOP\_DATE |

* Una sorgente deve contenere i seguenti dati

|  |  |  |  |
| --- | --- | --- | --- |
| **Dato** | **Nullable** | **Nome sul database** | **Note** |
| Nome del social network | NO | SOCIAL\_NETWORK |  |
| ID dell’oggetto assegnato dal social network | SI | SRC\_SN\_ID |  |
| Nome dell’oggetto sul social network | SI | SRC\_SN\_NAME |  |
| Tipo di oggetto | SI | TYPE | Valori supportati  GENERIC per twitter  PAGE per facebook  GROUP per facebook |
| Granularità dell’informazione da estrarre | SI | INFO\_LEVEL | Valori supportati  POST per facebook  COMMENT per facebook |

* Ad ogni richiesta devono essere associate una o più sorgenti
* La stessa sorgente può essere utilizzata da più richieste
* Twitter
  + Ogni tweet deve contenere i seguenti dati (<https://dev.twitter.com/overview/api/tweet> , <https://dev.twitter.com/overview/api/users> , <https://dev.twitter.com/overview/api/tweets#obj-coordinates>)

|  |  |  |  |
| --- | --- | --- | --- |
| **Dato** | **Nullable** | **Nome campo (JSON)** | **Nome campo (DATABASE)** |
| Data di creazione | NO | created\_at | CREATION\_DATE |
| Numero di volte che il tweet è stato segnalato tra i favoriti | NO | favorite\_count | FAVORITES\_COUNT |
| Lingua del tweet | NO | Lang | LANGUAGE |
| Latitudine del luogo in cui il tweet è stato prodotto | SI | Oggetto Coordinates, secondo campo | LATIDUE |
| Longitudine del luogo in cui il tweet è stato prodotto | SI | Oggetto Coordinates, primo campo | LONGITUDE |
| Testo contenuto nel tweet | NO | Text | TEXT |
| Id del tweet (assegnato da twitter) | NO | Id | TWEET\_ID |
| Numero di followers dell’autore del tweet | NO | Oggetto User, campo followers\_count | USER\_FOLLOWER\_COUNT |
| Id dell’autore del tweet (assegnato da twitter) | NO | Oggetto User, campo id | USER\_ID |
| Numero di liste a cui l’utente è iscritto | NO | Oggetto User, campo listed count | USER\_LISTED\_COUNT |
| Luogo dichiarato dall’utente | SI | Oggetto User, campo location | USER\_LOCATION |

* + Ogni tweet deve essere associato ad una richiesta
  + Ogni tweet deve essere associato ad una sorgente
  + Ogni tweet può avere associati i seguenti dati[[8]](#footnote-8)
    - Uno o più Hashtag
    - Uno o più Link
    - Uno o più Media (memorizzare il link al contenuto multimediale)
* Facebook
  + Ogni dato scaricato da facebook deve contenere i seguenti dati (<https://developers.facebook.com/docs/graph-api/reference/v2.2/post?locale=it_IT> , <https://developers.facebook.com/docs/graph-api/reference/object/comments>, <https://developers.facebook.com/docs/graph-api/reference/object/likes> )

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Dato** | **Nullable** | **Nome campo (JSON)** | **Nome campo (DATABASE)** | **Note** |
| Id dell’autore (assegnato da facebook) | NO | From | AUTHOR\_ID |  |
| Didascalia del post | NO | caption | CAPTION | Solo per i post |
| Data di creazione | NO | created\_time | CREATION\_DATE |  |
| Descrizione del link nel post | SI | descritpion | DESCRIPTION |  |
| Id del post a cui il commento è associato | SI |  | FEED\_REF | Solo per commenti |
| Numero di likes del post/commento | NO | postID/likes | LIKES |  |
| Link contenuto nel post | SI | link | LINK |  |
| Messaggio di stato contenuto nel post | SI | message | MESSAGE |  |
| Foto inclusa in un post | SI | picture | PICTURE |  |
| Id del dato (post o commento) assegnato da facebook | NO | id | SN\_ID |  |
| Tipo di dato | NO |  | TYPE | Valori supportati :  POST  COMMENT |

### Diagramma Entity-Relationship



## La logica dell’applicazione

All’avvio dell’applicazione l’utente seleziona la richiesta che deve essere soddisfatta (attualmente tramite passaggio di parametri da riga di comando – parametro *req\_id=NUM* dove *NUM* deve corrispondere all’id della richiesta memorizzata sul database).

L’applicazione segue il seguente flusso logico

1. Carica la richiesta dal database con le relative sorgenti
2. In base ai Social Network specificati nelle sorgenti vengono letti i file di properties necessari
3. In base alle sorgenti e al criterio di selezione temporale vengono inizializzati i produttori
4. Vengono avviati i produttori
   1. I produttori di tipo REST vengono avviati immediatamente
   2. I produttori di tipo STREAM
      1. Vengono avviati immediatamente se la data corrente è successiva alla data di inizio
      2. Viene programmato il loro avvio alla data di inizio se questa è successiva alla data corrente
5. Vengono avviati i consumatori

### Architettura dell’applicazione

* L’applicazione è strutturata secondo un modello produttore/consumatore. La comunicazione tra produttori e consumatori avviene mediante un buffer (implementato con una coda bloccante).
* Per ogni produttore e consumatore viene creato un Thread
* Tutti i consumatori condividono la stessa connessione al database

### I produttori

Ogni produttore si occupa del download dei dati da una sorgente secondo una specifica modalità di acquisizione. (Per ogni sorgente possono essere necessari i produttori che gestiscono la modalità REST, la modalità STREAM o entrambe). Ogni dato acquisito dai produttori viene inserito nel buffer.

Le classi che implementano i produttori sono contenute nel package *it.tvpad.control.producer*. All’interno di questo package ci sono le seguenti classi che rappresentano un’astrazione dei produttori:

* AbstractDataProducer : produttore generico
* AbstractRestDataProducer : produttore generico per la modalità REST
* AbstractStreamDataProducer : produttore generico per la modalità STREAM

#### Produttore per Twitter in modalità STREAM

Il produttore per Twitter in modalità STREAM è stato implementato utilizzando la libreria twitter4j (<http://twitter4j.org/en/index.html> versione 4.0) ed è contenuto nel package *it.tvpad.control.producer.twitter*. Questo produttore consiste in un event listener che resta in ascolto delle notifiche che arrivano da Twitter. Quando arriva una notifica la libreria lancia un evento che viene gestito dal listener. In particolare il metodo onStatus viene attivato quando arriva una notifica. Questo metodo estrae dal tweet i dati di interesse, crea un oggetto della classe Tweet e lo inserisce nel buffer di comunicazione.

#### Produttore per Facebook in modalità REST

Il produttore per Facebook in modalità REST è contenuto nel package *it.tvpad.control.producer.facebook*. Questo produttore è formato da due componenti principali

* Un server http asincrono contenuto nel package *it.tvpad.control.producer.facebook.server* (implementato tramite la libreria *Apache HttpComponents* versione 4.3[*https://hc.apache.org/httpcomponents-core-ga*](https://hc.apache.org/httpcomponents-core-ga)) necessario per gestire l’autenticazione per accedere alle API di Facebook
* Il produttore vero e proprio contenuto nel package *it.tvpad.control.producer.facebook.data* (implementato tramite la libreria *restfb* <http://restfb.com/> versione 1.7.1).

L’autenticazione (User Access Token) viene gestita nel seguente modo (<https://developers.facebook.com/docs/facebook-login/access-tokens?locale=it_IT#usertokens> )

1. Una richiesta http GET viene inviata alla url <https://www.facebook.com/v2.2/dialog/oauth> per l’autenticazione impostando come parametri (attualmente effettuata aprendo il browser perché viene richiesto il login dell’utente)
   1. client\_id : identificativo dell’applicazione registrata su facebook
   2. redirect\_uri : uri a cui facebook risponderà (uri su cui il server è in ascolto)
   3. scope : code
   4. grant : permessi richiesti in formato csv
2. Facebook inviera al server un codice (da estrarre dalla uri)
3. Una richiesta GET inviata alla url <https://graph.facebook.com/oauth/access_token> impostando come parametri
   1. client\_id : identificativo dell’applicazione registrata su facebook
   2. redirect\_uri : uri a cui facebook risponderà (uri su cui il server è in ascolto)
   3. client\_secret : chiave dell’applicazione registrata su facebook
   4. code : codice identificativo ottenuto al punto 2
4. Facebook risponde fornendo lo User Access Token richiesto

Una volta ottenuto lo User Access Token il produttore richiederà a Facebook i dati specificati dalla sorgente.

### I consumatori

Ogni consumatore preleva i dati dal buffer e li memorizza sul database. Attualmente è stato implementato un solo consumatore. La classe che lo implementa è chiamata *DataConsumer* ed è contenuta nel package *it.tvpad.control.consumer*.

Per la memorizzazione dei dati i consumatori utilizzano un componente che si occupa della connessione con il database e della persistenza dei dati. Questo componente è stato implemenato nella classe *DatabaseService* contenuta nel package *it.tvpad.control.database*.

### Lettura dei parametri di configurazione

I parametri di configurazione sono gestiti attraverso l’utilizzo di file. In particolare si utilizzano file con estensione *.properties* la cui sintassi è della forma *NOME\_PARAMETRO = VALORE\_PARAMETRO*. Tutti i file di configurazione devono essere contenuti nella cartella *config* nella presente nella root directory dell’applicazione.

I files richiesti per il funzionamento dell’applicazione sono i seguenti

* database.properties : contiene i parametri per la connessione al database
  + database\_host : host su cui è installato il database (nome o indirizzo IP)
  + database\_type : tipo di database
  + database\_driver : nome del package in cui è contenuta la classe driver (utilizzata per la comunicazione tra java e il database)
  + database\_port : porta su cui il database è in ascolto
  + database\_name : nome del database
  + datbase\_user\_name : nome utente per la connessione al database
  + database\_user\_password : password per la connessione al database
* log4j.properties : contiene le proprietà per il logging (<http://logging.apache.org/log4j/1.2/manual.html> Sezione Configuration)
* facebook.properties
  + facebook.app.id : id dell’applicazione registrata su Facebook
  + facebook.app.secret : chiave dell’applicazione registrata su Facebook
  + facebook.app.redirect.base.uri : URI sulla quale è in ascolto il server
  + facebook.app.grant : permessi richiesti dall’ applicazione in formato csv
  + facebbok.app.access.token.uri : uri per il login a facebook
  + facebook.server.port : porta su cui è in ascolto il server
* twitter.properties : contiene i parametri necessari alla librera twitter4j (<http://twitter4j.org/en/configuration.html> )

Le classi per la lettura dei parametri di configurazione sono contenute nel package *it.tvpad.control.properties*. All’interno di questo package c’è la classe astratta *AbstractPropertiesReader* rappresenta un generico lettore di file di properties e contiene il metodo *loadProperties* che legge il file.

Per ogni file di properties è stato sviluppato uno specifico parser per la lettura dei parametri e la loro validazione.

# Stato dello sviluppo

## Informazioni generali sullo stato dello sviluppo

|  |  |  |
| --- | --- | --- |
| **Download dei dati** | | |
| **Social Network** | **Modalità** | **Stato implementazione** |
| Twitter | Stream | Completato |
| Twitter | Rest | Non implementato |
| Facebook | Stream | Parzialmente implementato |
| Facebook | Rest | Completato |
| **Criteri di selezione dei dati** | | |
| **Social Network** | **Criterio** | **Stato implementazione** |
| Twitter | Parole chiave | Parzialmente implementato |
| Twitter | Intervallo di tempo | Parzialmente implementato |
| Twitter | Lingua | Completato |
| Twitter | Luogo | Non implementato |
| Facebook | Parole chiave | Parzialmente implementato |
| Facebook | Intervallo di tempo | Parzialmente implementato |
| Facebook | Lingua | Non implementato |
| Facebook | Luogo | Non implementato |

## Informazioni dettagliate sullo stato dello sviluppo

### Dowlad dei dati

#### Twitter

Per twitter è stato implementato il download e la memorizzazione dei contenuti in modalità STREAM. Per la modalità REST è stato fatto solo uno studio preliminare.

#### Facebook

Per poter utilizzare le API di facebook, in entrambe le modalità di download, è stato realizzato un server http. Tale server è necessario per gestire la fase di autenticazione senza la quale non è possibile invocare le API.

Il download e la memorizzazione dei dati in modalità REST sono stati implementati.

Per quanto riguarda la modalità STREAM è necessario l’utilizzo del server. Il server sviluppato è predisposto per ricevere gli aggiornamenti provenienti da facebook. Per completare lo sviluppo del download dei dati in modalità stream occorre:

1. Gestire le sottoscrizioni per la ricezione degli aggiornamenti (<https://developers.facebook.com/docs/graph-api/real-time-updates/v2.2?locale=it_IT> Sezione “Creating and Updating Subscriptions”)
2. Scrivere un gestore delle risposte del server (<https://developers.facebook.com/docs/graph-api/real-time-updates/v2.2?locale=it_IT> Sezione “Setting Up your Callback URL”)
3. Gestire la memorizzazione dei dati

### Criteri di selezione dei contenuti

Il cirterio di selezione basato su parole chiave consente all’uteIl filtro sulle parole chiave l’applicazione permette di utilizzare soltanto gli operatori booleani AND e NOT e non consente l’uso di sotto-espressioni

Il filtro sull’intervallo di tempo è strettamente legato allo sviluppo del download dei dati:

1. La modalità di download STREAM consente di scaricare dati dalla di inizio a fino a quella di fine
2. Se la data di inizio è precedente alla data in cui il programma viene avviato
   1. La modalità di download REST recupera i dati prodotti dalla data di inizio fino alla data corrente
   2. La modalità di download STREAM recupera i dati prodotti dalla corrente fino alla data di fine
3. La data di fine deve essere posteriore alla data corrente

Il filtro sulla lingua dei contenuti è supportato in modo nativo dalle API di Twitter mentre le API di Facebook non offrono alcun supporto.

Il filtro sul luogo in cui i contenuti sono stati prodotti è supportato in modo nativo dalle API di Twitter mentre le API di Facebook non offrono alcun supporto.

# Contenuto cartelle e utilizzo

* SocialNetworkDataDownloader
  + config : contiene i file di configurazione
  + db\_dumps: contiene i dump del database
  + log : contiente i file su cui vengono memorizzati i log dell’applicazione
  + src : contiente i codici sorgenti (esportati come progetto per Eclipse versione Luna)
  + SocialNetworkDataDownloader-0.0.1-SNAPSHOT-jar-with-dependencies : file jar contenente l’applicazione e tutte le librerie necessarie per il funzionamento
  + docs : contiene questo file

Per lanciare l’applicazione

1. Aprire il Prompt dei comandi
2. Posizionarsi nella cartella SocialNetworkDataDownloader
3. Digitare “java –jar SocialNetworkDataDownloader-0.0.1-SNAPSHOT-jar-with-dependencies req\_id=NUM”

1. Supportati solo gli operatori logici AND e NOT. L’utilizzo di espressioni booleane annidate non è supportato [↑](#footnote-ref-1)
2. Supportato solo per il social network Twitter [↑](#footnote-ref-2)
3. Non supportato [↑](#footnote-ref-3)
4. Supportato solo per il social network Twitter [↑](#footnote-ref-4)
5. Supportato solo per il social network Facebook [↑](#footnote-ref-5)
6. Funzionalità non supporta (emersa alla consegna del 29/01/2015) [↑](#footnote-ref-6)
7. Il livello di view non è stato definito per questa fase di studio [↑](#footnote-ref-7)
8. I dati per le tabelle Hashtag, Link e Media sono molto semplici (https://dev.twitter.com/overview/api/entities) [↑](#footnote-ref-8)