[SE-21] YOUTUBE CONNECTOR

Design Requirement Specification Document

DIBRIS – Università di Genova. Scuola Politecnica, Corso di Ingegneria del Software 80154

**DATA – 11/04/2018**

**VERSION: 1**

**Autori**

Annalisa Bovone

**REVISION HISTORY**

|  |  |  |  |
| --- | --- | --- | --- |
| Versione | Data | Autori | Note |
| 1 | 20/04/2018 | Annalisa Bovone | Prima consegna |
| 2 | 16/08/2018 | Annalisa Bovone | Consegna finale |
|  |  |  |  |
|  |  |  |  |

Indice dei Contenuti

1 Introduzione

* 1. Scopo del Documento
  2. Descrizione Generale del Sistema
  3. Riferimenti Bibliografici
  4. Definizioni e Acronimi
  5. Overview del Documento

1. Descrizione del Progetto
   1. Introduzione
   2. Architettura del Sistema
   3. Vincoli e Assunzioni
2. Interfacce
   1. Dashboard
   2. Informazioni su video scelto
3. Data Stores
4. Use Cases
5. Structural Design
   1. Diagramma delle classi
      1. YoutubeVideo
         1. Attributi
         2. Metodi
      2. UpdateVideo
         1. Attributi
      3. YoutubeChannel
         1. Attributi
         2. Metodi
      4. YoutubeAnswer
         1. Attributi
         2. Metodi
      5. YoutubeComment
         1. Attributi
         2. Metodi
      6. YoutubeConnector
         1. Attributi
         2. Metodi
   2. Object Diagram
6. Dynamic Model
   1. Requisito 1
   2. Requisito 2
   3. Requisito 3
   4. Requisito 6

# Introduzione

## Scopo del Documento

Questo documento ha lo scopo di mostrare l'architettura del sistema software chiamato Youtube Connector. Questo documento è stato redatto per gli sviluppatori, i progettisti e i tester del sistema. Alcune parti dell’interfaccia grafica potrebbero essere mostrate ad analisti del marketing in modo da comprendere tutte le funzionalità del sistema.

## Descrizione Generale del Sistema

Il sistema, una volta avviato e configurato correttamente, esegue un processo che estrae i dati dei video di YT scelti e li salva sul DB, e costruisce una web application per la presentazione e interrogazione dei risultati dell'analisi.

Il processo di estrazione dei dati viene lanciato ogni giorno ad un orario prestabilito su un insieme di video scelti. L’orario e gli url dei video devono essere configurati prima dell’esecuzione del sistema. Le informazioni estratte vengono salvate su un database che permetta di interrogare i dati in maniera efficiente.

Le informazioni relative ad uno dei video precedentemente analizzati vengono visualizzate mediante un’interfaccia grafica html.

I dati che vengono mostrati all’operatore che utilizza YoutubeConnector sono: titoli dei video analizzati e nomi dei creatori dei contenuti; in particolare, per ogni video viene inoltre mostrato: numero di iscritti al canale yt, data e ora di pubblicazione del video, durata, descrizione, immagine di anteprima del video, numero di “mi piace”, “non mi piace” e numero di visualizzazioni del video. Inoltre vengono visualizzati i commenti relativi al video e, tramite il sentiment engine X2Check, viene generato un valore numerico sulla base della positività del commento. Grazie ai dati raccolti, vengono realizzati dei grafici, per vedere l'andamento nel tempo di, per esempio, quantità di commenti per giorno e valore medio del sentiment, numero di “mi piace” per giorno, ecc…

Il servizio X2Check è fornito dall’azienda App2Check e, preso in input un testo, restituisce un valore tra 1 e 5 dove 1 indica un commento negativo, 3 neutro e 5 positivo.

## Riferimenti Bibliografici

URD

## Definizioni e Acronimi

|  |  |
| --- | --- |
| **Acronimo-Nome** | **Definizione** |
| YT | Youtube |
| SW | Software |
| API | Application Programming Interface |
| REST | REpresentational State Transfer. E’ un’architettura che viene utilizzata nei sistemi distribuiti. Tramite un url ben definito viene identificata univocamente una risorsa o un insieme di risorse. Questo tipo di servizio viene utilizzato per il recupero di informazioni tramite richieste http. |
| DB | Database |

## Overview del documento

Nella sezione 2 viene presentata una descrizione ad alto livello dell'applicazione. Nella sezione 3 vengono descritte le interfacce del sistema, mentre nella sezione 4 viene descritto come vengono memorizzati i dati. Nella sezione 5 vengono mostrate le attività che l’utente può svolgere sulla piattaforma YoutubeConnector. La sezione 6 descrive la struttura del sistema, mentre nella sezione 7 viene descritto in dettaglio come i requisiti sono implementati.

# Descrizione del Progetto

## Introduzione

Una volta scelti i video da analizzare e l’orario in cui avviare il sistema, viene avviato un processo che estrae i dati relativi a ciascun video tramite le API fornite da Youtube e li salva in un database orientato agli oggetti. Viene poi costruito un servizio di REST API che interroga il database per restituire i dati in maniera compatta. Infine il servizio di REST viene utilizzato dall’applicazione web per mostrare i risultati dell’analisi agli utenti.

Le tecnologie utilizzate per lo sviluppo del software sono:

* ECLIPSE JAVA EE IDE for Web Developers Oxygen con JAVA (versione 1.8) per l’estrazione e il salvataggio dei dati da YT
* SPRING BOOT (versione 3.9.5.201807030714) tool usato per la costruzione del servizio di REST API, per la costruzione del server e per il l’avvio dell’applicazione web
* MONGO DB (versione 3.1.1) database ad oggetti per il salvataggio dei dati
* BOOTSTRAP per lo sviluppo dell’applicazione web
* JAVASCRIPT per lo sviluppo dell’applicazione web
* CHART JS libreria usata per la costruzione dei grafici nell’interfaccia web

Per disegnare i diagrammi UML sono stati utilizzati i tools Dia e draw.io.

## Architettura del Sistema

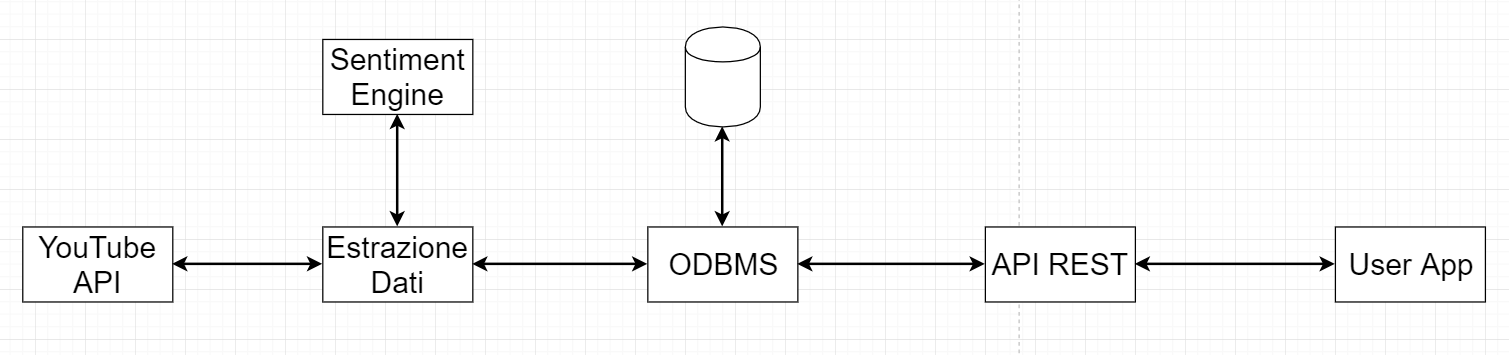


Figura 1

In Figura 1 viene rappresentata l’architettura del sistema: il sistema è composto da un processo che, dato un url di un video di YT, estrae dati (quali titolo, autore, testo dei commenti, numero di like, dislike e numero di visualizzazioni, ecc…), tramite l’utilizzo di YT API, e li salva sul DB. I commenti al video vengono estratti sempre dalle YT API e analizzati dal Sentiment Engine che restituisce un numero intero in base al livello di positività del commento (5 positivo, 3 neutro, 1 negativo); anche questo dato verrà salvato sul DB .I dati salvati vengono poi estratti da un servizio REST e visualizzati tramite un’applicazione web.

## Vincoli e Assunzioni

Youtube fornisce agli utenti la possibilità di creare commenti relativi ad un particolare video. I commenti possono raccogliere risposte: è presente, infatti, per ogni commento, il pulsante RISPONDI (vedi fig. 2) che permette di inserire il proprio commento nel thread di risposte a quel commento. Nonostante questo tasto sia presente anche sulle risposte, Youtube non permette di creare risposte annidate tra loro.

Figura 2

Esiste quindi un solo livello di risposta e pertanto nell’architettura di YoutubeConnector troviamo la classe padre YoutubeAnswer che contiene tutti le caratteristiche del commento/risposta e la classe figlia YoutubeComment che, oltre ai campi della classe da cui eredita, contiene una lista di risposte.

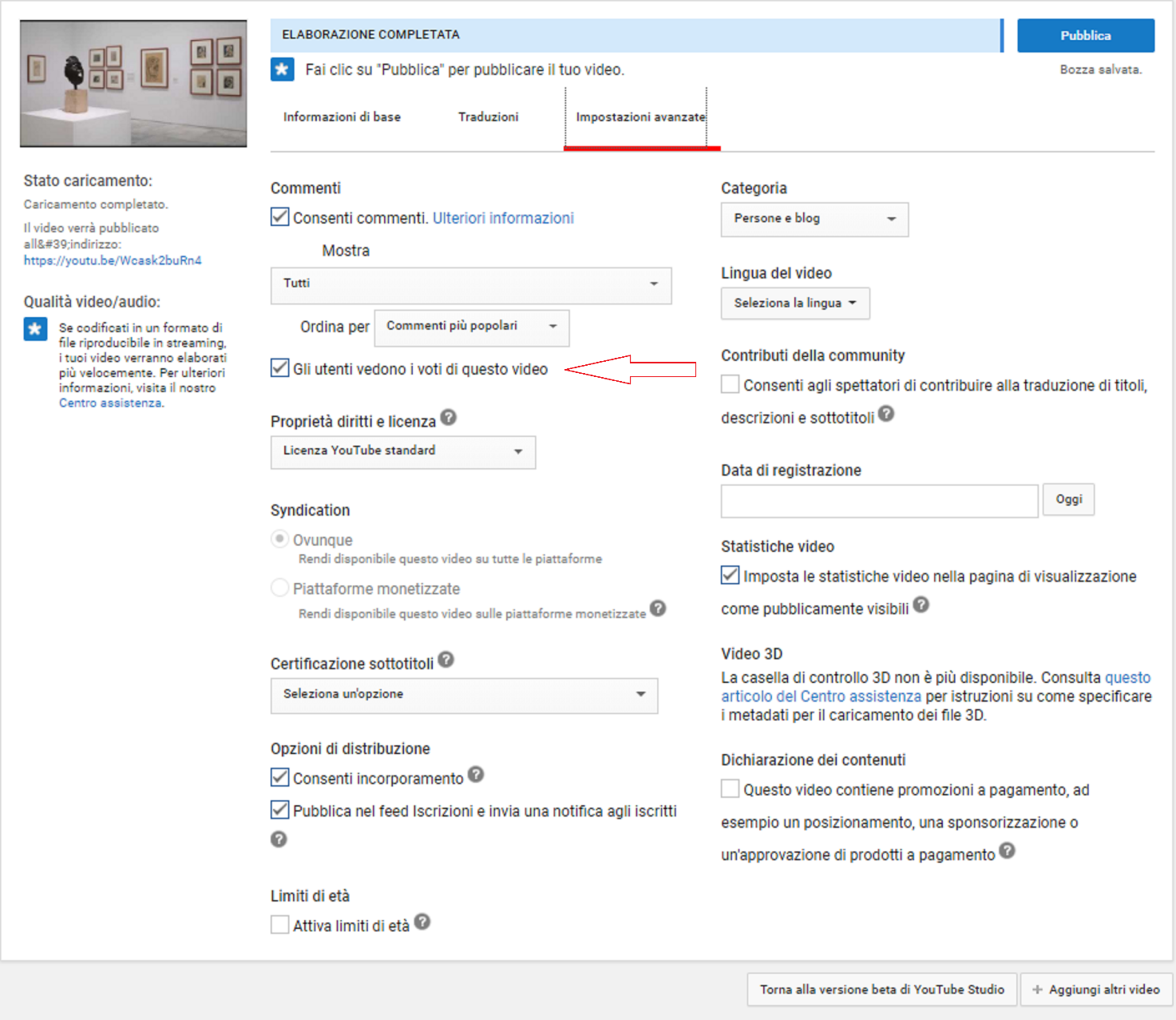
Gli utenti di Youtube che decidono di caricare i propri contenuti video sulla piattaforma, possono decidere se rendere visibile agli utenti che guarderanno il video il contatore di “like” e “dislike” al video.

Figura 3

Se chi carica il video decide di lasciare la checkbox (indicata in fig 3) spuntata, fornisce la possibilità agli utenti di YT di visualizzare i contatori dei “mi piace” e “non mi piace”, relativi al video, come in figura 4. Se invece, la checkbox viene deselezionata, i pulsanti di “mi piace” e “non mi piace” sono comunque presenti, ma non viene visualizzato il conteggio totale (vedi fig 5)



Figura 4

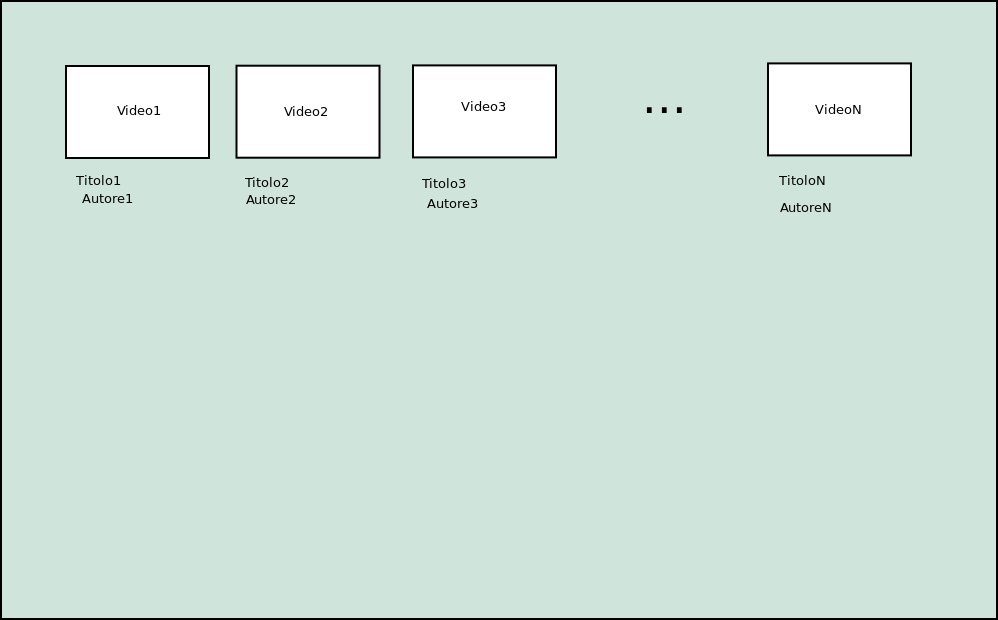


Figura 5

Infine, il tasto “disike” (vedi fig 2) nei commenti non contiene mai il contatore, ma, come nel caso precedente, può essere attivo o disattivo per il singolo utente.

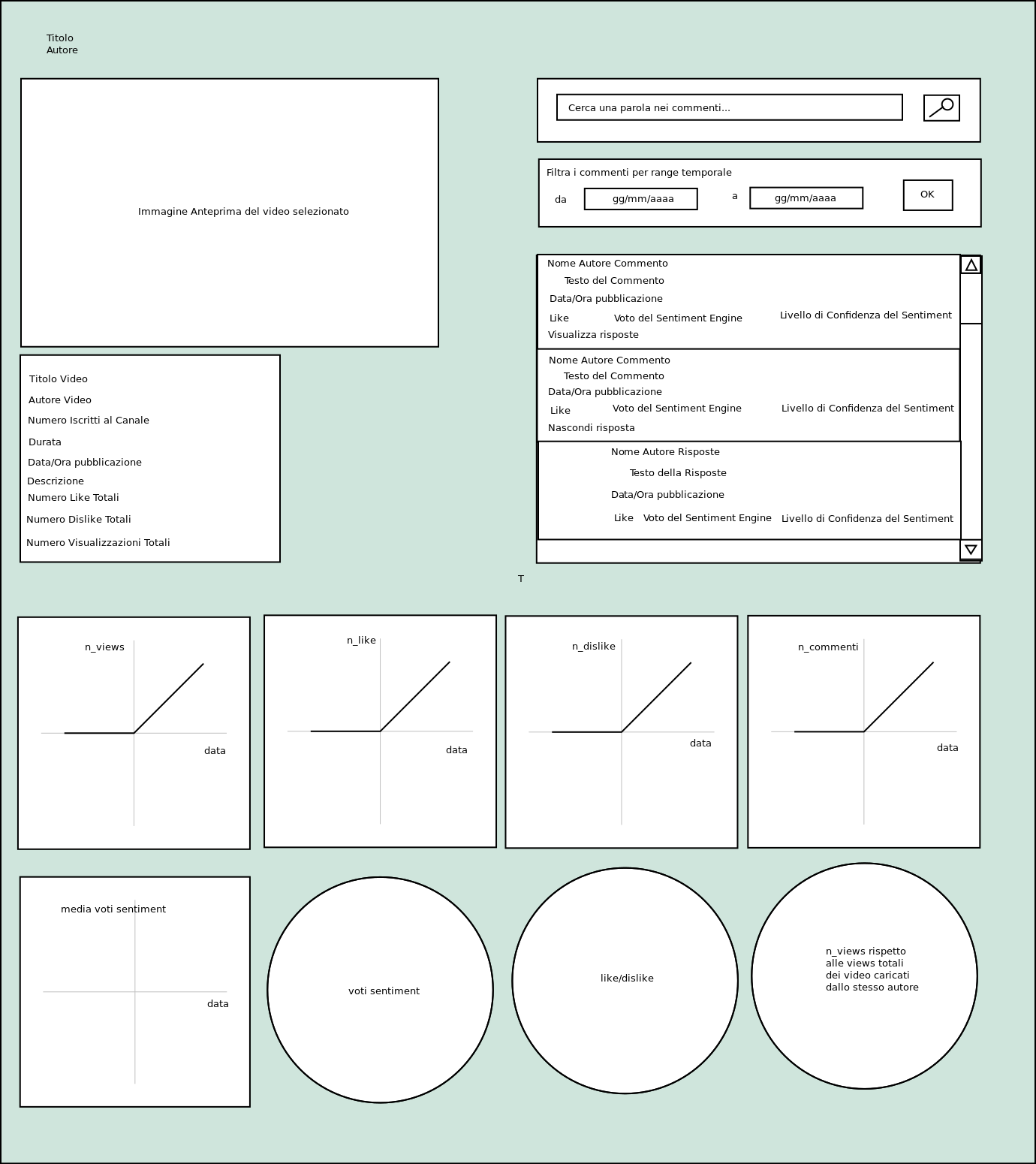
# Interfacce

## Dashboard



Nella dashboard sono presenti tutte le miniature dei video già analizzati dal tool. Ogni video mostra l’immagine di anteprima, il titolo e l’autore corrispondente. Ogni rettangolo è cliccabile e il click sul video scelto reindirizza alla schermata successiva che mostra tutte le informazioni relative al video scelto.

## Informazioni su video scelto



Questa interfaccia mostra nel dettaglio, le informazioni del video richiesto. Vengono mostrati innanzitutto il titolo, l’autore (nome del canale youtube che ha caricato il video sulla piattaformaYT) e l’immagine di anteprima del video. Nel riquadro in basso vengono riportati inoltre: il numero di iscritti al canale yt proprietario del video, la durata, la data e l’ora di pubblicazione del video, la descrizione, il numero di “mi piace” (volte in cui è stato cliccato il tasto mi piace relativo al video), “non mi piace” e visualizzazioni totali del video.

A destra, invece, sono presenti tutti i commenti lasciati dagli utenti sotto al video. Ogni commento riporta l’autore, il testo, la data e l’ora di pubblicazione, il numero di “mi piace” al commento, il voto stabilito dal Sentiment Engine di X2Check e il livello di certezza associato al voto. Se il commento è provvisto di una o più risposte troviamo il tasto “Visualizza risposta” o “Visualizza risposte” (in base al numero di risposte al commento). Se questo tasto viene cliccato vengono mostrate le risposte al commento indentate al di sotto del commento principale. Ogni risposta mostra gli stessi campi dei commenti, ad eccezione del pulsante per visualizzare le risposte perché non è possibile avere risposte annidate tra loro (vedi 2.3 Vincoli e Assunzioni). I commenti possono essere filtrati dall’utente secondo la ricerca di una parola/frase e/o secondo un intervallo temporale. Una volta aggiunto un filtro sarà possibile visualizzare nuovamente tutti i commenti togliendo il filtro attraverso l’apposito tasto “X”.

Infine, scorrendo la pagina troviamo i grafici relativi ai dati estratti. L’utente visualizza solamente i grafici disponibili e passando col mouse sopra gli indicatori del grafico può visualizzare le percentuali e/o le occorrenze relative a quel grafico.

# Data Stores.

Il sistema salva i dati in un database ad oggetti, in particolare viene consigliato l’utilizzo di MongoDB. Le collections che memorizzano i dati sul database sono: Videos, YoutubeChannels, Comments.

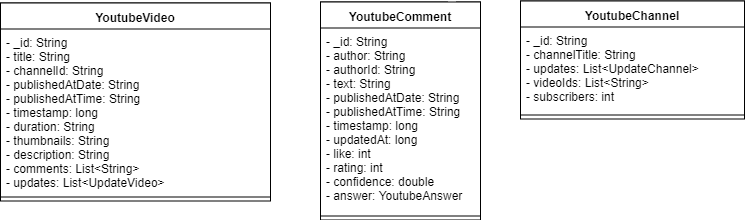


Figura 6

La collection Videos contiene una lista di oggetti di tipo YoutubeVideo aventi:

\_id: chiave primaria che identifica univocamente un video sulla piattaforma YT, è una stringa di 11 caratteri composta da lettere, numeri e simboli;

title: stringa che contiene il nome del video (YT impone un massimo di 100 caratteri);

channelId: identifica univocamente un canale YT sulla piattaforma, è una stringa di 24 caratteri composta da lettere, numeri e simboli;

publishedAtDate: stringa che contiene la data di pubblicazione del video, salvata nel seguente formato: gg/MM/aaaa;

publishedAtTime: stringa che contiene l’orario di pubblicazione del video convertito al fuso orario italiano (YT fornisce la data e l’ora di pubblicazione del video secondo il fuso orario di Greenwich). Il formato dell’ora è il seguente: hh:mm:ss;

timestamp: long che contiene i millisecondi trascorsi dal 1 Gennaio 1970 alla data considerata. Visto che l’orario fornito da YT non contiene i millisecondi, per convenzione viene considerato lo scoccare del secondo.

duration: tipo string salva la durata del video nel formato hh:mm:ss;

thumbnails: contiene l’url dell’immagine di anteprima del video;

description: contiene la descrizione del video;

comments: lista di stringhe contenente gli id dei commenti relativi al video;

updates: lista di oggetti di tipo UpdateVideo. UpdateVideo salva:

like: valore intero che indica il numero di utenti che hanno cliccato il tasto “mi piace”. Il numero di like non è sempre conteggiabile: l’utente proprietario del video infatti durante il caricamento dello stesso ha la possibilità di scegliere questa opzione (vedi 2.3 Vincoli e Assunzioni);

dislike: valore intero che indica il numero di utenti che hanno cliccato il tasto “non mi piace”;

views: valore intero che indica il numero di utenti che hanno visualizzato il video;

date: stringa che contiene la data in cui vengono estratti questi dati. Se l’estrazione viene fatta più volte lo stesso giorno, l’oggetto update viene aggiornato.

Ogni volta che viene lanciato il processo per il video viene aggiunto un nuovo record alla lista per tener traccia dell’andamento dei dati nel tempo.

La collection youtubeChannels contiene una lista di oggetti di tipo YoutubeChannel aventi:

\_id: chiave primaria che identifica univocamente un canale yt sulla piattaforma, è una stringa di 24 caratteri composta da lettere, numeri e simboli;

channelTitle: stringa che contiene il nome del canale YT da cui è stato caricato il video.

videoIds: lista di stringhe contenente gli id dei video caricati da questo canale yt;

subscriber: valore intero che indica il numero di utenti iscritti al canale yt in questione.

La collection Comments contiene una lista di oggetti di tipo YoutubeComment aventi:

\_id: chiave primaria che identifica univocamente un commento sulla piattaforma, è una stringa di 26 caratteri composta da lettere, numeri e simboli;

author: stringa che contiene il nome dell’utente della piattaforma che ha scritto il commento;

authorId: stringa di 24 caratteri che identifica univocamente un utente della piattaforma YT;

text: contiene il testo del commento (yt impone una lunghezza massima di 10090 caratteri);

publishedAtDate: stringa che contiene la data di pubblicazione del commento, salvata nel seguente formato: gg/MM/aaaa;

publishedAtTime: stringa che contiene l’orario di pubblicazione del commento convertito al fuso orario italiano (YT fornisce la data e l’ora di pubblicazione del commento secondo il fuso orario di Greenwich). Il formato dell’ora è il seguente: hh:mm:ss;

timestamp: long che contiene i millisecondi trascorsi dal 1 Gennaio 1970 alla data considerata. Visto che l’orario fornite da YT non contiene i millisecondi, per convenzione viene considerato lo scoccare del secondo.

updatedAt: long che contiene i millisecondi trascorsi dal 1 Gennaio 1970 all’ultima modifica del commento. Le API di YT forniscono, per ogni commento, la data e l’ora della creazione e dell’ultima modifica del commento.

like: intero che indica il numero di utenti che hanno cliccato il tasto “mi piace”. Il numero di like non è sempre conteggiabile.

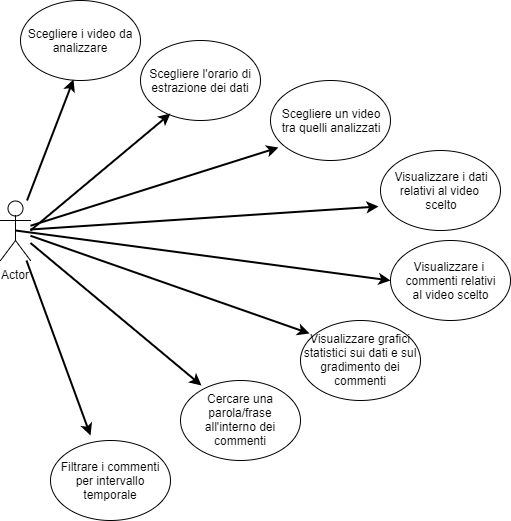
rating: intero compreso tra 1 e 5 che indica il grado di positività di un commento (1=commento negativo, 3=commento neutro, 5=commento positivo). Questo numero viene fornito dalle API del Sentiment Engine X2Check;

confidence: numero reale compreso tra 0 e 1 che indica quanto il Sentiment Engine sia sicuro del risultato ottenuto nella valutazione (0=insicuro, 1=molto sicuro);

answer: lista di oggetti di tipo YoutubeAnswer, con gli stessi attributi dell’oggetto YoutubeComment descritto sopra a meno del campo answer. YT infatti non fornisce la possibilità di rispondere ad una risposta (vedi 2.3 Vincoli e assunzioni).

# Use Cases

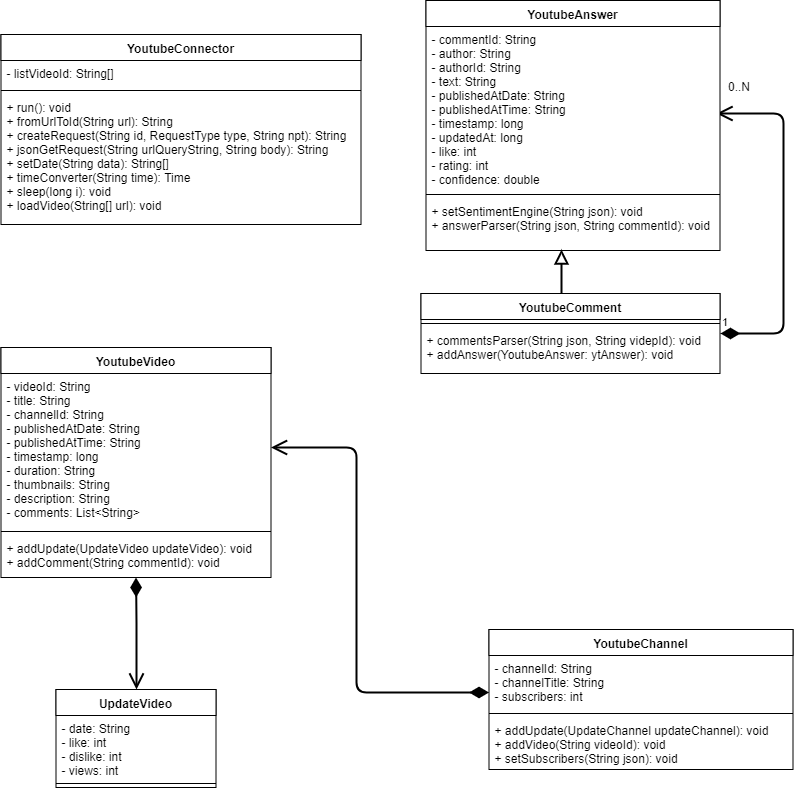
In questa sezione sono descritte le funzionalità del sistema con cui l’utente è in grado di interagire. Per una definizione delle diverse funzionalità fare riferimento all'URD.



L’utente che voglia utilizzare YoutubeConnector deve configurare l’orario di esecuzione e la lista dei video su cui fare analisi. In un secondo momento, l’utente può visualizzare nella dashboard tutti i video che sono stati analizzati dal sistema e può scegliere di visualizzare tutte le informazioni riguardo uno specifico video. L’utente visualizza quindi, per il video scelto, i dati strutturati (numero di like, dislike, visualizzazioni, ecc…) e non strutturati (commenti) e i grafici statistici relativi ai dati estratti. È inoltre possibile filtrare i commenti secondo intervallo temporale oppure secondo parole/frasi.

# 6 Structural Design

## 6.1 Class Diagram



### 6.1.1 YoutubeVideo

La classe YoutubeVideo è l’istanza dell’oggetto YoutubeVideo e contiene quindi i dati per identificare un video della piattaforma YT.

**6.1.1.1 Attributi**

videoId: chiave primaria che identifica univocamente un video sulla piattaforma YT, è una stringa di 11 caratteri composta da lettere, numeri e simboli;

title: stringa che contiene il nome del video (YT impone un massimo di 100 caratteri);

channelId: identifica univocamente un canale YT sulla piattaforma, è una stringa di 24 caratteri composta da lettere, numeri e simboli;

publishedAtDate: stringa che contiene la data di pubblicazione del video, salvata nel seguente formato: gg/MM/aaaa;

publishedAtTime: stringa che contiene l’orario di pubblicazione del video convertito al fuso orario italiano (YT fornisce la data e l’ora di pubblicazione del video secondo il fuso orario di Greenwich). Il formato dell’ora è il seguente: hh:mm:ss;

timestamp: long che contiene i millisecondi trascorsi dal 1 Gennaio 1970 alla data considerata. Visto che l’orario fornite da YT non contiene i millisecondi, per convenzione viene considerato lo scoccare del secondo.

duration: tipo string salva la durata del video nel formato hh:mm:ss;

thumbnails: contiene l’url dell’immagine di anteprima del video;

description: contiene la descrizione del video;

comments: lista di stringhe che contiene gli id dei commenti lasciati sotto al video in questione;

updates: lista di oggetti di tipo UpdateVideo. Ogni volta che viene lanciato il processo per il video viene aggiunto un nuovo record alla lista per tener traccia dell’andamento dei dati nel tempo.

**6.1.1.2Metodi**

addUpdate: metodo void. Prende come argomento un oggetto di tipo UpdateVideo e lo aggiunge alla lista delle update;

addComment: metodo void. Prende come argomento una stringa e la aggiunge alla lista degli id dei commenti.

### 6.1.2 UpdateVideo

**6.1.2.1 Attributi**

like: valore intero che indica il numero di utenti che hanno cliccato il tasto “mi piace”. Il numero di like non è sempre conteggiabile: l’utente proprietario del video infatti durante il caricamento dello stesso ha la possibilità di scegliere questa opzione;

dislike: valore intero che indica il numero di utenti che hanno cliccato il tasto “non mi piace”;

views: valore intero che indica il numero di utenti che hanno visualizzato il video;

date: stringa che contiene la data in cui vengono estratti questi dati.

### 6.1.3 YoutubeChannel

**6.1.3.1 Attributi**

channelId: chiave primaria che identifica univocamente un canale yt sulla piattaforma, è una stringa di 24 caratteri composta da lettere, numeri e simboli;

channelTitle: stringa che contiene il nome del canale YT da cui è stato caricato il video;

videoIds: lista di stringhe contenente gli id dei video caricati da questo canale yt;

subscribers: intero che contiene il numero di utenti iscritti al canale youtube in questione

**6.1.3.2 Metodi**

addUpdate: metodo void. Prende come argomento un oggetto di tipo UpdateChannel e lo aggiunge alla lista delle update;

addVideo: metodo void. Prende come argomento una stringa e la aggiunge alla lista degli id dei video.

setSubscribers: metodo void. Prende come argomento un json con la risposta alla richiesta per conoscere il numero di iscritti a quel determinato canale yt e setta questo valore.

### 6.1.4 YoutubeAnswer

**6.1.4.1 Attributi**

commentId: chiave primaria che identifica univocamente un commento sulla piattaforma, è una stringa di 26 caratteri composta da lettere, numeri e simboli;

author: stringa che contiene il nome dell’utente della piattaforma che ha scritto il commento;

authorId: stringa di 24 caratteri che identifica univocamente un utente della piattaforma YT;

text: contiene il testo del commento (yt impone una lunghezza massima di 10090 caratteri);

publishedAtDate: stringa che contiene la data di pubblicazione del commento, salvata nel seguente formato: gg/MM/aaaa;

publishedAtTime: stringa che contiene l’orario di pubblicazione del commento convertito al fuso orario italiano (YT fornisce la data e l’ora di pubblicazione del commento secondo il fuso orario di Greenwich). Il formato dell’ora è il seguente: hh:mm:ss;

timestamp: long che contiene i millisecondi trascorsi dal 1 Gennaio 1970 alla data considerata. Visto che l’orario fornite da YT non contiene i millisecondi, per convenzione viene considerato lo scoccare del secondo.

updatedAt: long che contiene i millisecondi trascorsi dal 1 Gennaio 1970 all’ultima modifica del commento. Le API di YT forniscono, per ogni commento, la data e l’ora della creazione e dell’ultima modifica del commento.

like: intero che indica il numero di utenti che hanno cliccato il tasto “mi piace”. Il numero di like non è sempre conteggiabile: l’utente proprietario del video infatti durante il caricamento dello stesso ha la possibilità di scegliere questa opzione;

rating: intero compreso tra 1 e 5 che indica il grado di positività di un commento (1=commento negativo, 3=commento neutro, 5=commento positivo). Questo numero viene fornito dalle API del Sentiment Engine X2Check;

confidence: numero reale compreso tra 0 e 1 che indica quanto il Sentiment Engine sia sicuro del risultato ottenuto nella valutazione (0=insicuro, 1=molto sicuro);

**6.1.4.2 Metodi**

setSentimentEngine: funzione void. Prende come argomento una stringa corrispondente al json che restituisce il Sentiment Engine. Questa funziona setta il rating e la confidence della risposta e del commento.

answerParser: funzione che prende come argomento una stringa, ovvero il json di risposta alla richiesta di avere tutte le risposte di un commento e l’id del commento stesso. La funzione effettua il parsing del json e restituisce una lista delle YoutubeAnswer di quel commento.

### 6.1.5 YoutubeComment

YoutubeComment è classe figlia di YoutubeAnswer quindi eredita tutti gli attributi e tutti i metodi della classe padre.

**6.1.5.1 Attributi**

answer: lista di oggetti di tipo YoutubeAnswer. YT infatti non fornisce la possibilità di rispondere ad una risposta. (vedi Vincoli e Assunzioni).

**6.1.5.2 Metodi**

commentsParser: funzione che prende come argomento una stringa, ovvero il json di risposta alla richiesta di avere tutti i commenti di un video e l’id del video stesso. La funzione effettua il parsing del json e restituisce una lista delle YoutubeComment di quel video.

addAnswer: funzione void che prende come argomento una YoutubeAnswer e, se questa non è ancora presente nella lista delle risposte la aggiunge, altrimenti la sostituisce.

### 6.1.6 YoutubeConnector

La classe YoutubeConnector ha come attributi le interfacce per la creazione delle repository su MongoDB

**6.1.6.1 Attributi**

listVideoId: lista di stringhe che contiene gli url dei video che dovranno essere analizzati;

**6.1.6.2 Metodi**

**-** run: è il metodo che attiva il thread di esecuzione dell’intero sistema. Ogni ora controlla se è passata la mezzanotte del giorno e a quel punto esegue l’estrazione dei dati di ciascun video, salvando su DB i dati mancanti e aggiornando i cambiamenti. Dopo aver esaminato tutti i video torna a dormire, in attesa della mezzanotte del giorno successivo;

- fromUrlToId: prende in ingresso l’url del video da analizzare e restituisce una stringa contente solo l’id del video;

- createRequest: prende in ingresso un identificativo, il tipo di richiesta e il nextpagetoken (stringa che contiene l’identificativo della pagina successiva che permette di visualizzare altri commenti, ciascuna pagina riporta fino a 20 risultati). Questa funzione restituisce la richiesta che dovrà essere fatta in base al tipo di richiesta che viene passato alla funzione;

- jsonGetRequest: prende in ingresso l’url della richiesta e il body (per richieste di tipo POST) e restituisce il json sotto forma di stringa;

- setData: prende come input una stringa, che il json ha restituito come data di pubblicazione, e, tramite regular expression, restituisce un’array di stringhe che contiene la data, l’ora e il timestamp della pubblicazione rispetto al fuso orario italiano;

- timeConverter prende come input una stringa, che il json ha restituito come durata del video, e la converte in una stringa contenente la durata del video;

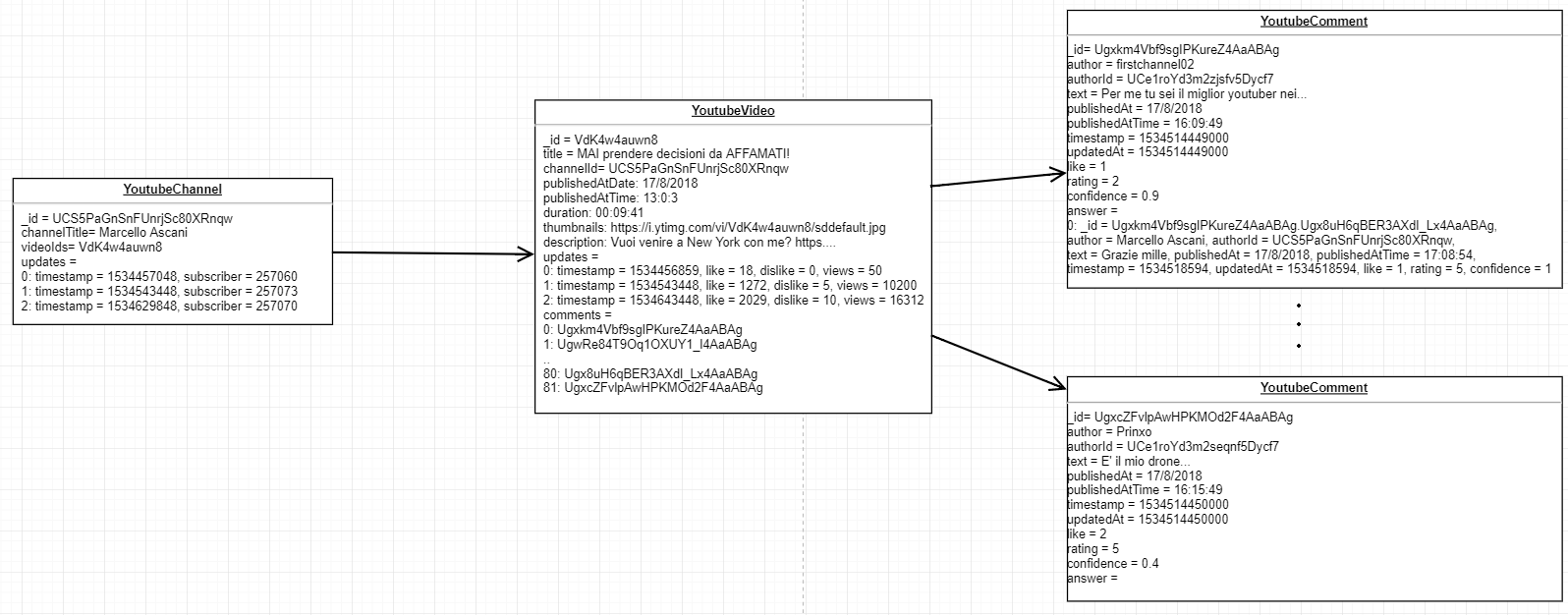
-sleep: prende in ingresso i minuti durante i quali il thread va a dormire e sospende l’esecuzione;

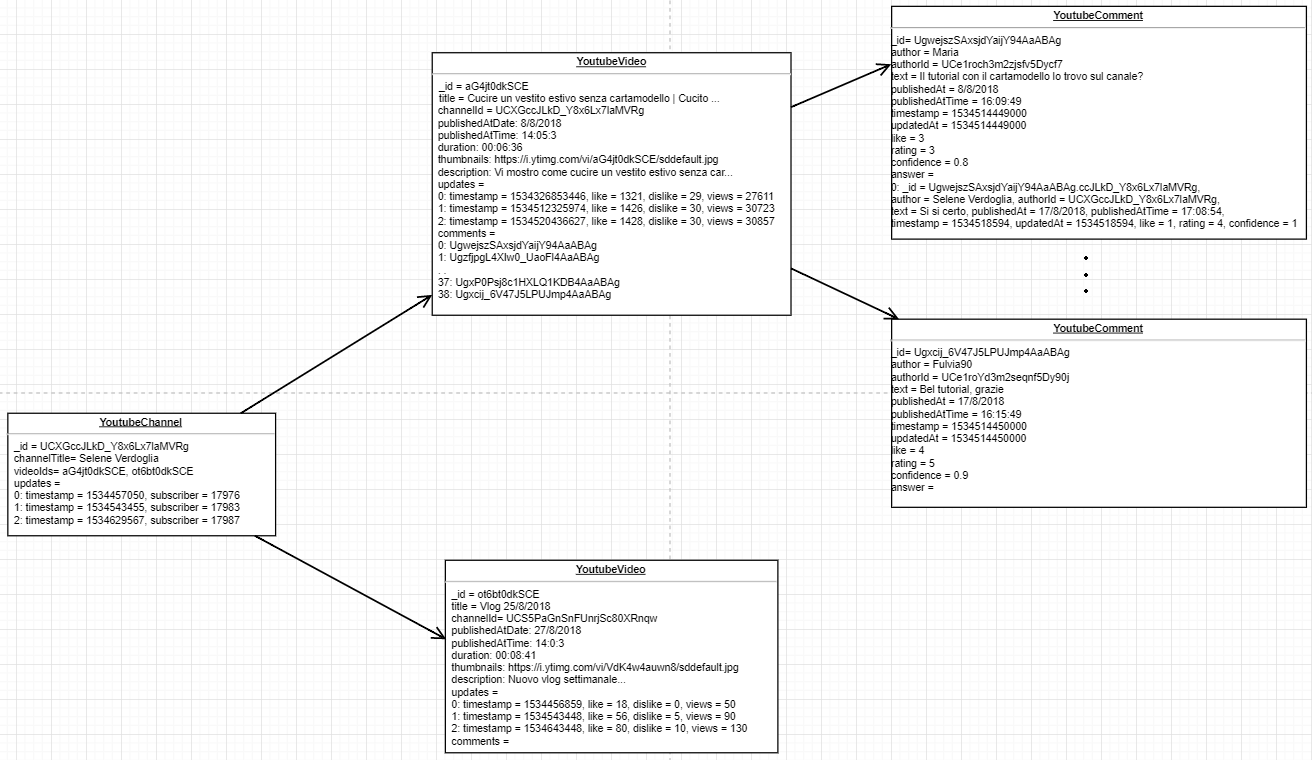
-loadVideoId: prende il input il nome del file che contiene gli url dei video da analizzare e carica nel sistema gli id di questi video.

## 6.2 Object Diagram

Il diagramma in figura rappresenta una situazione in cui il sistema potrebbe trovarsi, in particolare il diagramma descrive una situazione in cui:

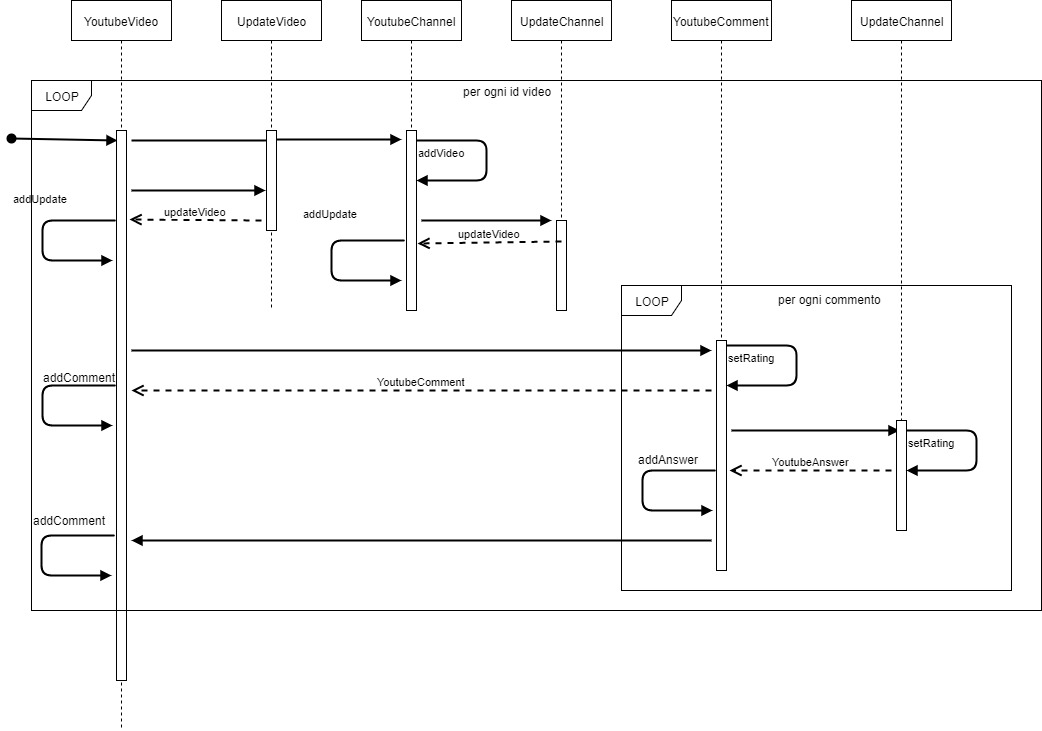
1. il sistema ha analizzato 3 video, 2 dei quali appartenenti allo stesso canale youtube
2. per ciasun video il sistema ha rilevato la presenza di commenti e li ha salvati



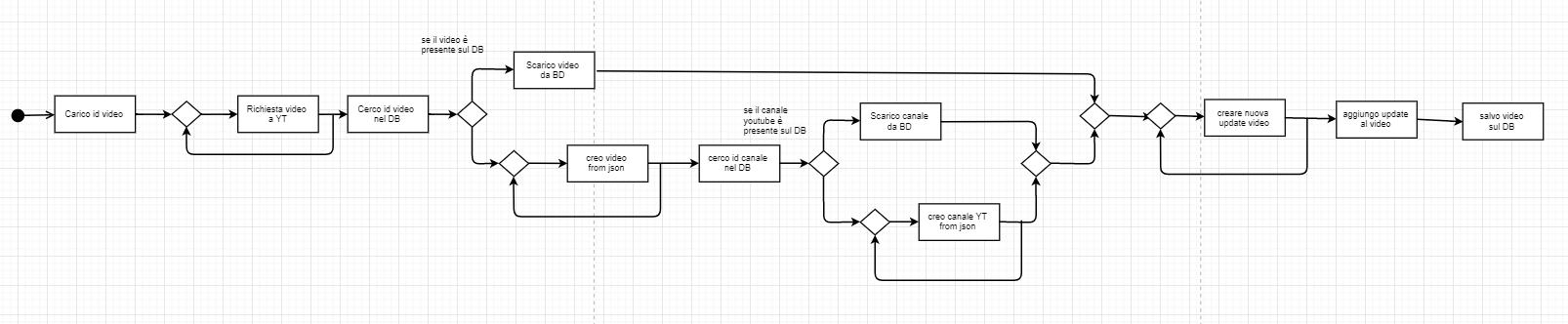


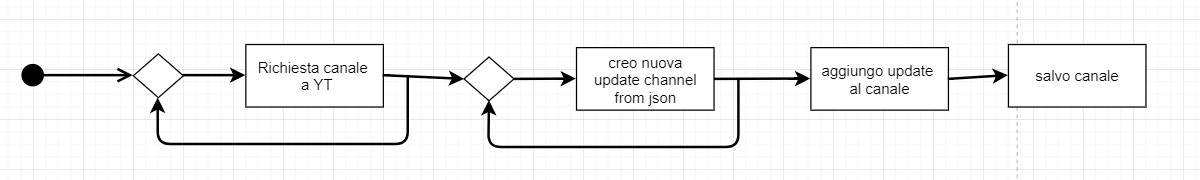
# Dynamic Model

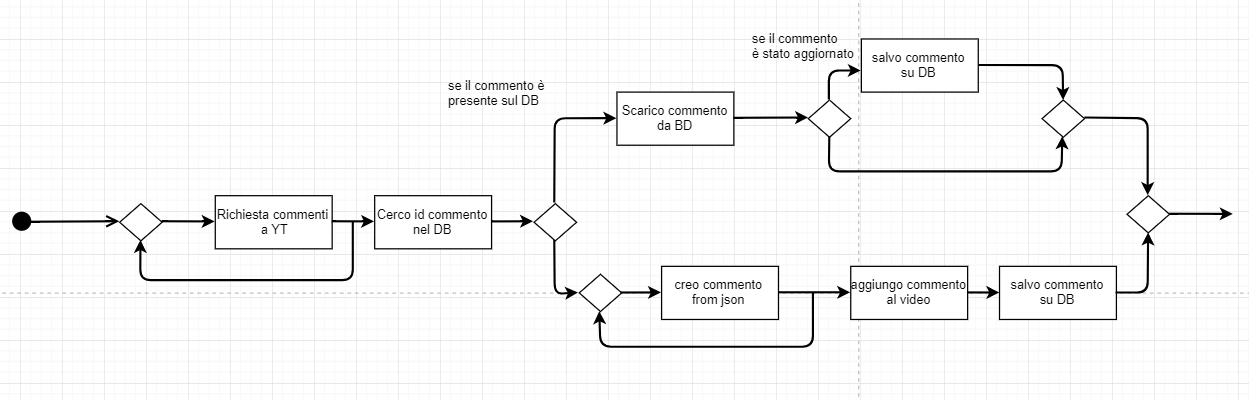
## 7.1 Requisito 1

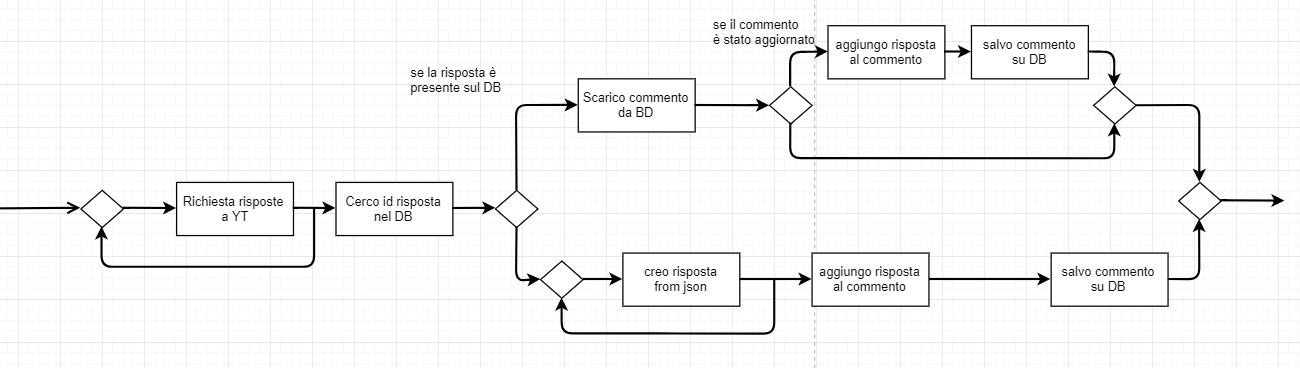


## 7.2 Requisito 2

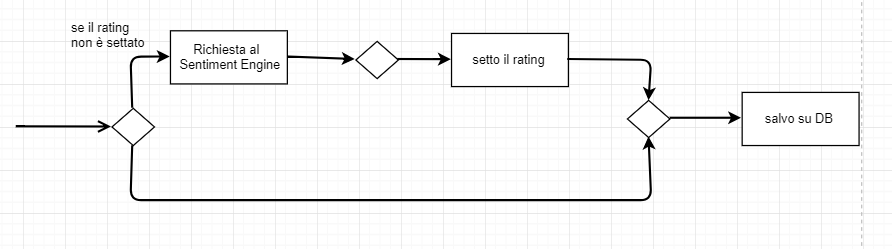








## 7.3 Requisito 3



## 7.4 Requisito 6

