

# **Recomendador de películas inteligente**

**Entrega final DASI 2018/19**

**Daniel Bastarrica Lacalle  
Gabriel Sellés Salvà**

# Contenido de la memoria

<b>Contenido de la memoria</b>	<b>2</b>
<b>1. Introducción</b>	<b>3</b>
<b>2. Componentes del sistema</b>	<b>4</b>
2.1. Almacenamiento del sistema	4
2.2. Módulo de aprendizaje y generación de recomendaciones	4
2.3. Módulo de reconocimiento de intents	5
2.6. Controlador	6
2.7. Disposición multi-agentes	6
2.8. Diagrama de clases	8
<b>3. Funcionalidades del sistema</b>	<b>9</b>
3.1. Obtener información sobre una película	9
3.2. Solicitar una recomendación	10
3.3. Recibir opiniones sobre películas del usuario	11
3.4. Gestionar la lista de pendientes	12
<b>4. Uso del sistema</b>	<b>14</b>
4.1. Manual de instalación	14
4.2. Manual de ejecución y customización	15
4.3. Ejemplos de uso	17
4.3.1. Obtención de información de una película	17
1. Obtención del director de una película	17
2. Obtención de la trama de una película	18
3. Obtención de las notas de una película en un sitio web	18
4. Obtención de información “general” de una película	19
4.3.2. Recibir opiniones sobre las películas	20
1. Opinión positiva	20
2. Opinión negativa	20
4.3.3. Solicitar una recomendación	21
1. Recomendar una película	21
2. Recomendar una película de forma acotada	22
4.3.4. Gestión de la lista de pendientes	22
1. Insertar películas en la lista de pendientes	22
2. Eliminar películas de la lista de pendientes	23
3. Listar el contenido de la lista de pendientes	23
<b>5. Informe de división de tareas</b>	<b>23</b>

# 1. Introducción

El presente proyecto consiste en desarrollar un sistema de Recomendación de películas inteligente.

El sistema realizará recomendaciones acotadas a los gustos del usuario y proporcionará información sobre distintos aspectos de las películas que contempla en su almacenamiento. Algunos de los aspectos tratados son el director de la película, su trama, los principales actores que participan y su calificación en múltiples sitios web, entre otros aspectos. Además permitirá que el usuario gestione una lista de “películas pendientes” en las cuales podrá insertar o eliminar películas.

Para realizar esta funcionalidad se ha implementado un Sistema basado en Agentes que permitirá utilizar múltiples técnicas de aprendizaje. De esta forma, en función del número de “opiniones” que el usuario proporcione al sistema, el “gerente” del sistema podrá seleccionar la técnica más adecuada y así realizar mejores recomendaciones.

Las peticiones que el usuario solicite al sistema se realizarán mediante la introducción al sistema de sentencias en lenguaje natural. El sistema tendrá la capacidad de procesar dichas sentencias y responderá como corresponda.

El sistema estará compuesto por 2 agentes, uno encargado de la interacción con el usuario y otro encargado de gestionar toda la información relativa a las películas.

La implementación del proyecto se ha realizado en Python y, para ello, se han utilizado múltiples librerías que se especifican en el apartado de Manual de Instalación.

## 2. Componentes del sistema

En este apartado se muestran los distintos elementos que forman el sistema final. Las partes implementadas son las siguientes:

### 2.1. Almacenamiento del sistema

Toda la información que utiliza el sistema para realizar sus funcionalidades se encuentra distribuida en distintos componentes:

- **Base de datos MongoDB** que almacena información asociada (trama, director, actores, etc) a cada una de las películas que el sistema utiliza. Para generar la información que almacena dicha base de datos se han realizado múltiples peticiones a la *API* de *OMDB* (*Open Movie Database*) y se ha almacenado el resultado, previamente procesado, de estas. El script que genera la información y la inserta se encuentra en *“db\_creation\_script.py”*. El contenido de este componente se utiliza tanto para proporcionar información sobre una película al usuario como para obtener la información que el propio sistema utilizará para su aprendizaje.
- **Ficheros que almacenan las opiniones del usuario** y que el sistema utilizará cada vez que se deba reentrenar el modelo de aprendizaje del sistema. Consisten en los ficheros *“file.test”* y *“file.train”*.
- **Ficheros que almacenan información de acceso rápido.** Almacenan información a la que el sistema accede de forma frecuente y que tiene un formato simplificado. Al lanzar el sistema implementado, la mayoría de esta información se almacena en estructuras cuyo acceso es rápido (diccionarios). De esta forma, no es necesario acceder, por ejemplo, a la base de datos implementada en *MongoDB* para consultas menores y el rendimiento del sistema mejora considerablemente. Estos ficheros son *“films\_names”*, *“ntrmdb”*, *“pldb”* y *“rmdb”*.

### 2.2. Módulo de aprendizaje y generación de recomendaciones

Es el componente del sistema encargado de inferir posibles películas que puedan gustarle al usuario a partir de las opiniones que este le ha proporcionado previamente. Toda su lógica se encuentra en el fichero *“trained\_model.py”*.

Este módulo contiene un modelo entrenado con las opiniones (si le ha gustado o no) que ha introducido el usuario sobre ciertas películas. Utilizando estas opiniones y la información extraída de la base de datos del sistema sobre dichas películas, este modelo analizará si debe recomendar una película o no en el futuro.

Para implementar el modelo entrenado se utiliza *PySpark*, un *framework* en *Python* que nos permite generar distintos modelos de aprendizaje y a su vez proporciona un gran abanico de operaciones sobre dichos modelos.

Ya que, debido a la naturaleza del problema que se trataba en este proyecto, no existe una técnica de aprendizaje que dé, en todo momento, mejores resultados que las demás, decidimos implementar varias.

Entonces, modificando un parámetro en el código del sistema, este puede utilizar las siguientes técnicas:

- Regresión logística
- Clustering con el algoritmo K-nn
- Árboles de clasificación
- Redes neuronales

El parámetro que hay que modificar es “*model\_type*” y se encuentra en el fichero “*systemconfiguration.py*”.

Además, modificando otro parámetro del sistema, se puede seleccionar qué información sobre la película utiliza el modelo de entrenamiento para su aprendizaje.

En este caso, es necesario modificar los parámetros “*parameters*”, “*string\_columns*” y “*numeric\_columns*” que se encuentran en el mismo fichero.

## 2.3. Módulo de reconocimiento de intents

Para realizar el procesamiento que el usuario solicita mediante el texto que introduce, es necesario entender qué quiere que haga el sistema.

Este módulo es el encargado de identificar cuál es la intención (*intent*) que tiene el usuario al introducir una petición mediante una sentencia en lenguaje natural.

Para ello, utiliza el *framework RASA NLU*. Esta librería nos permite entrenar un modelo que, con el entrenamiento adecuado, obtendrá la capacidad de reconocer los *intents* que deseemos.

Además, *RASA NLU* genera modelos persistentes de forma que no será necesario entrenar dicho modelo cada vez que se lance el sistema.

Los elementos que conforman el módulo se ubican en:

- “intent\_recognition\_model”: directorio que almacena el modelo entrenado.
- “modelconfiguration.ipynb”: fichero con la información de entrenamiento del modelo.
- “intent\_recognition.py”: almacena la lógica para interactuar con el modelo entrenado y, dada una sentencia en lenguaje natural, reconocer el *intent* asociado.

## 2.6. Controlador

Este módulo será el responsable de conectar el agente encargado de procesar las peticiones del usuario con el almacenamiento del sistema (que contiene la información de las películas y la lista de pendientes) y el modelo entrenado. Cuando el agente conversacional detecte una petición, el agente encargado del procesamiento utilizará dicho módulo para realizar la funcionalidad correspondiente y luego devolver el resultado.

Este módulo está constituido por la clase Controller (implementada en el fichero *controller.py*).

## 2.7. Disposición multi-agentes

El sistema está construido utilizando el *framework* de modelado de agentes *MESA*. En él los agentes están gobernados por un Modelo, el cual contendrá todos los agentes del sistema y gestionará la comunicación entre ellos. Concretamente el sistema contendrá 2 agentes: el conversacional (*ChatAgent*) y el encargado de gestionar la información de las películas (*DBAgent*).

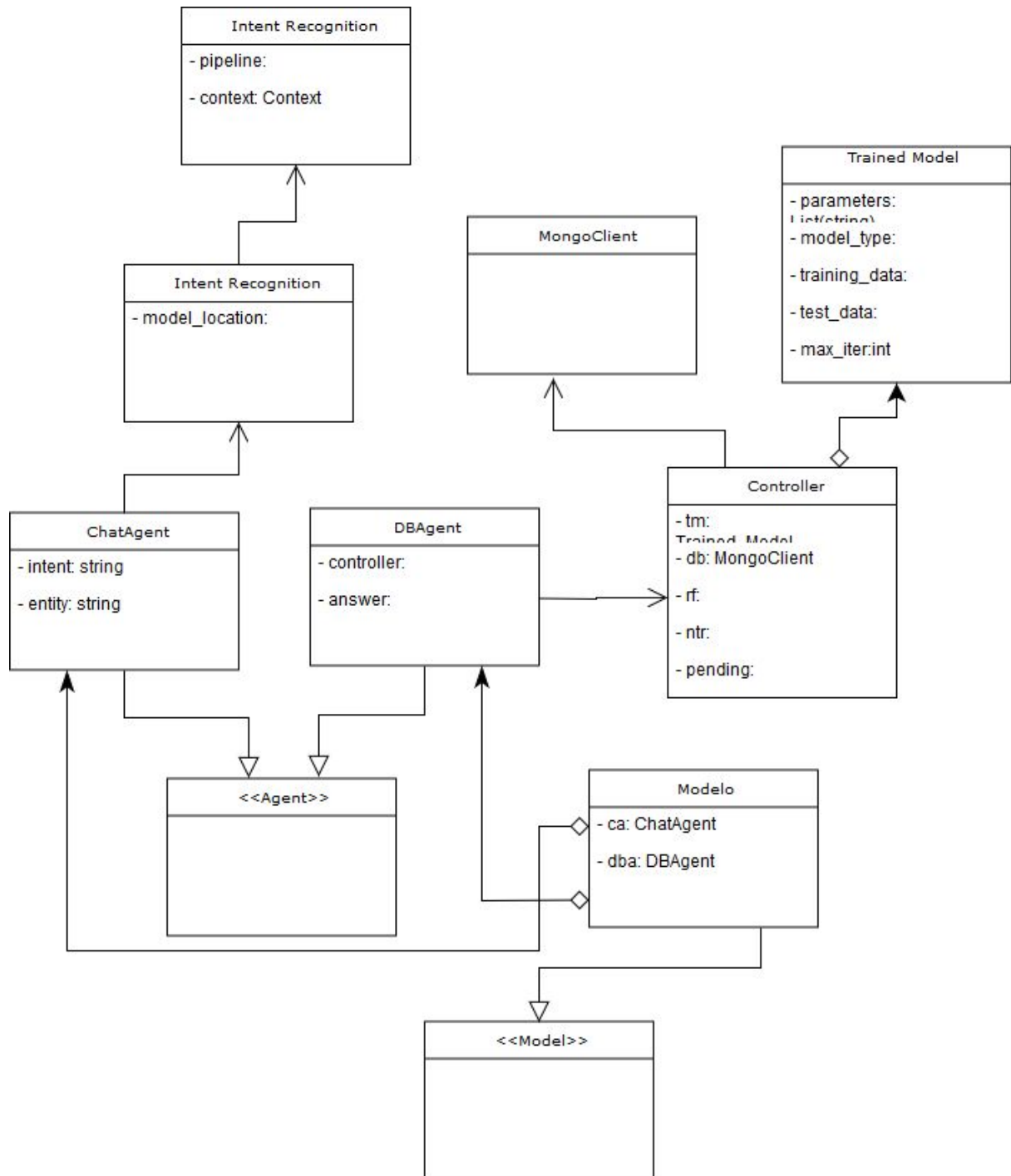
A continuación comentaremos los aspectos más relevantes de cada uno de los 3 componentes:

- Modelo: es el encargado de crear los agentes y de transmitir la información entre ellos. Debido a las restricciones de *MESA*, se ha intentado simular el modelo de paso de mensajes. Para ello, los agentes guardan en el Modelo la información que han obtenido y desean transmitir al otro agente, así como una variable que indicará si tienen información pendiente de procesar. Una vez que hayan procesado dicha información, indicarán mediante el modelo que ya han realizado la tarea correspondiente. Está localizado en “*Modelo.py*”.

- *ChatAgent*: es el agente encargado de gestionar toda la interacción con el usuario. Para ello utiliza la consola del sistema como medio de comunicación. Para modelar el comportamiento se ha utilizado una aproximación similar a las Reglas. Esta aproximación consiste en diccionarios en los que se almacena qué acción debe tomar el agente en función de la entrada que ha recibido. Por simplicidad, se han agrupado las posibles acciones en 2: si la consulta requiere una entidad (una película) o no. Más concretamente, para cada tipo de consulta realizada, si ésta es del tipo que requiere hacer referencia a una película en específico pero no ha sido capaz de reconocerla, el agente mostrará al usuario un mensaje correspondiente al *intent* inferido para solicitar la información restante. Está localizado en "*ChatAgent.py*".
- *DBAgent*: es el encargado de gestionar las peticiones del usuario una vez han sido procesadas. En función del *intent* y la configuración del sistema, accederá a los recursos correspondientes para llevar a cabo la acción solicitada por el usuario. En caso de haber solicitado alguna información devolverá la misma, y si por el contrario ha solicitado modificar un recurso (lista de pendientes, opiniones de películas), devolverá un mensaje para confirmar que la acción ha sido llevada a cabo con éxito. Está localizado en "*DBAgent.py*".

## 2.8. Diagrama de clases

Se adjunta el siguiente diagrama de clases. La información del flujo se encuentra más detallada en las correspondientes secciones del siguiente apartado:





## 3. Funcionalidades del sistema

En este apartado se muestran las distintas funcionalidades que ofrece el sistema implementado. También se indica, mediante diagramas de secuencia, cómo interactúan los componentes descritos anteriormente para dar lugar a cada funcionalidad.

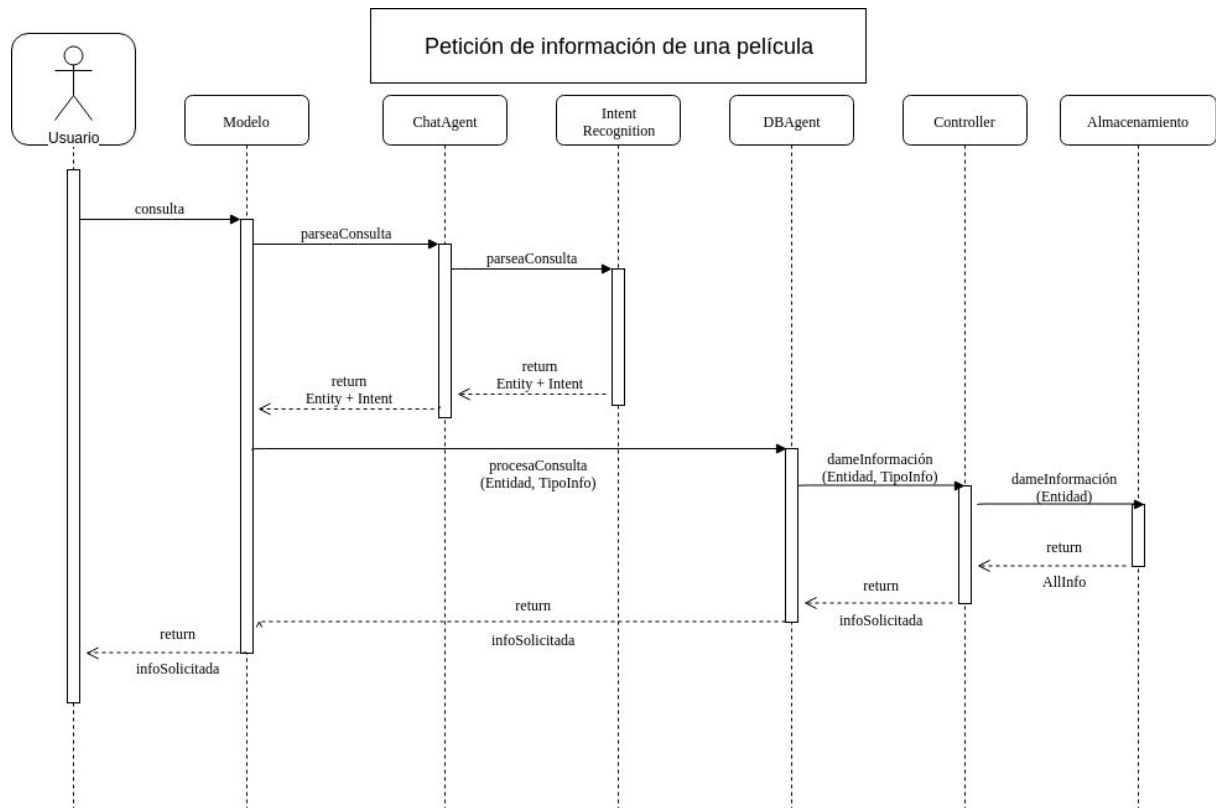
### 3.1. Obtener información sobre una película

El sistema tiene la capacidad de proporcionar al usuario información sobre múltiples aspectos de una película.

Los aspectos que el sistema puede proporcionar al usuario son:

- Director de la película
- Duración de la película
- Resumen de la trama
- Reparto principal de la película
- Premios que la película ha ganado
- Su calificación en Metacritic
- Su calificación en IMDB
- Su calificación en Rotten Tomatoes

El flujo que sigue el sistema para dar lugar a esta funcionalidad se muestra en el siguiente diagrama:



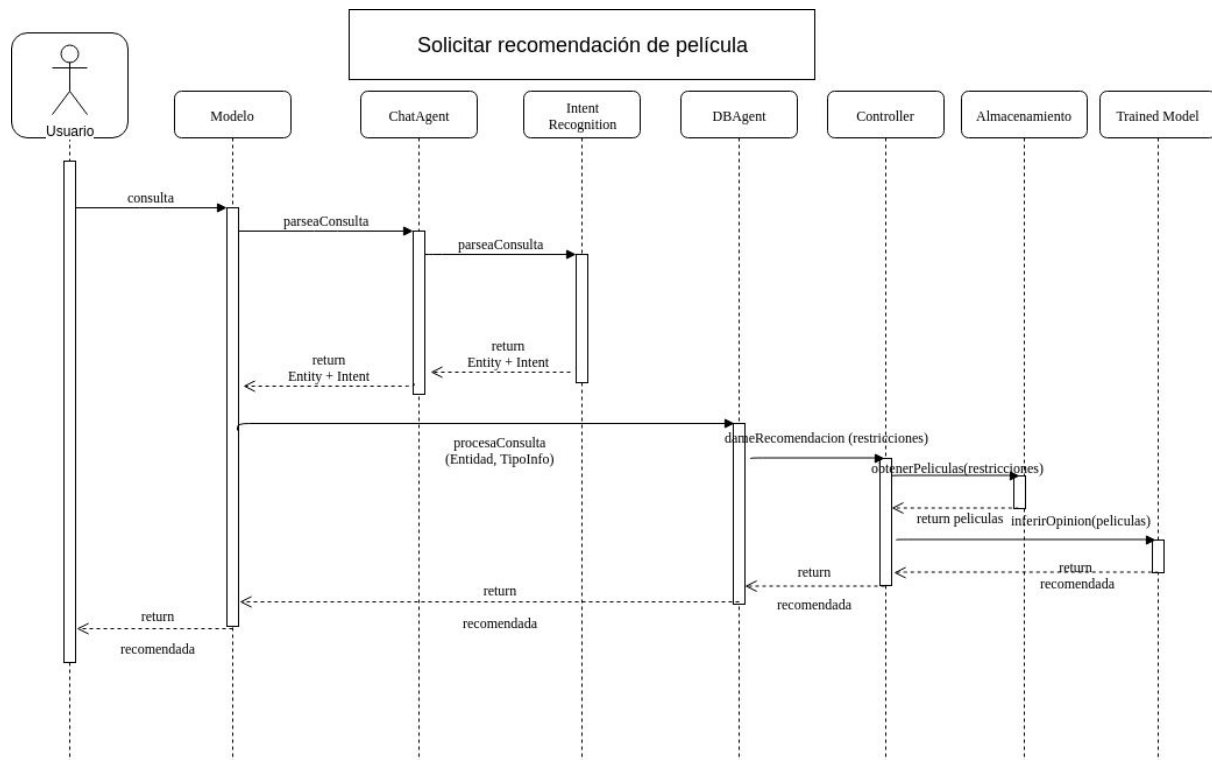
### 3.2. Solicitar una recomendación

El usuario puede pedir al sistema que le recomiende una película. La recomendación se realizará teniendo en cuenta las opiniones que el usuario le ha proporcionado anteriormente y, en general, la interacción que haya tenido el usuario con el sistema.

Además, esta recomendación se puede acotar por múltiples parámetros como:

- Género
- Actor que aparezca en la película
- Director
- Duración de la película:
  - Película corta: duración menor que 90 minutos.
  - Película mediana: mayor que 90 y menor que 120 minutos.
  - Película larga: mayor que 120 minutos.

En el siguiente diagrama se muestra como interactúan los distintos componentes del sistema cuando el sistema realiza una recomendación:



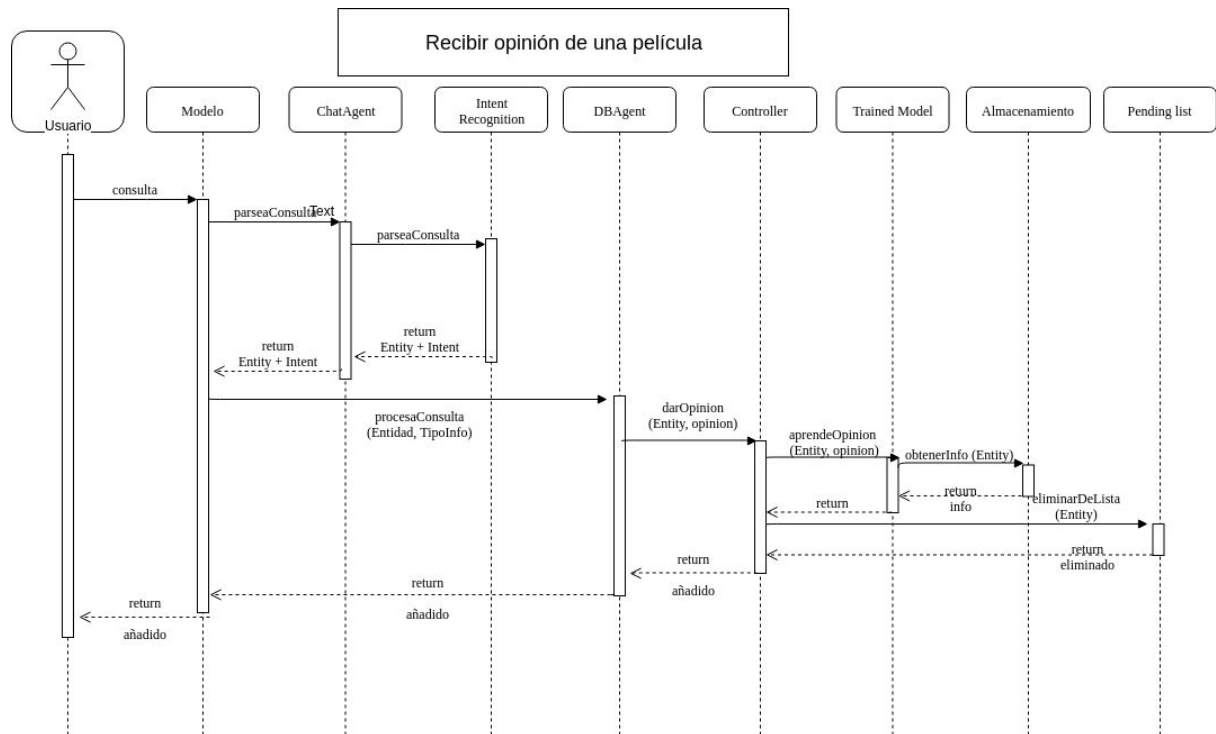
### 3.3. Recibir opiniones sobre películas del usuario

El sistema procesa las opiniones que el usuario introduce sobre una película (si le ha gustado una película o no).

Al procesar estas opiniones, el sistema obtiene información sobre esta película y analiza dicha información para ver qué características son las que pueden hacer que la película le guste al usuario.

El resultado del “análisis” de la información obtenida, se almacena en el sistema y lo utiliza para realizar mejores recomendaciones en el futuro.

Para ello, los componentes descritos anteriormente interactúan del siguiente modo:



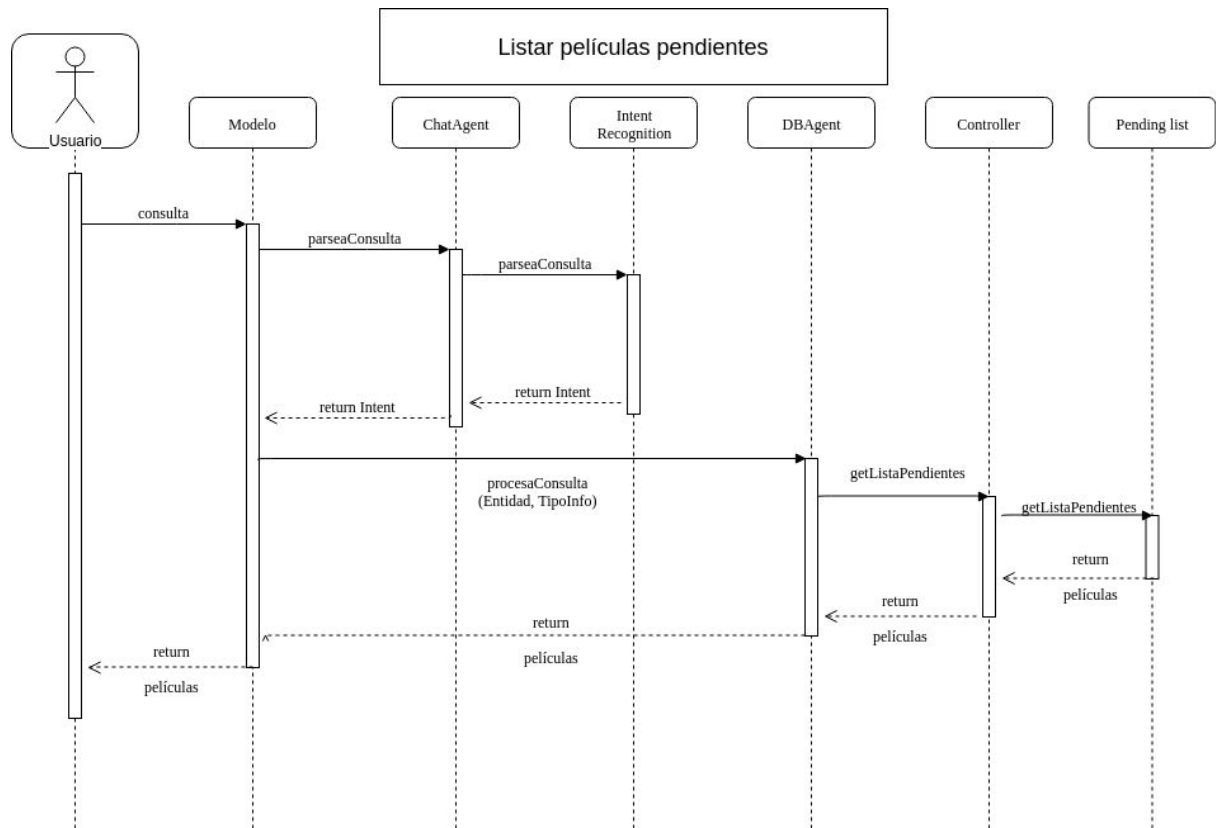
### 3.4. Gestionar la lista de pendientes

El sistema permite que el usuario gestione una colección de películas en la cual podrá insertar o eliminar las que desee.

Estas serán las películas que el usuario considera “pendientes” y podrá solicitar en cualquier momento que el sistema se las recuerde.

Además, cuando el usuario aporta una opinión sobre una película, el sistema la elimina automáticamente de la lista, ya que se supone que si le da una opinión, sea positiva o negativa, ya ha visto dicha película y por lo tanto ya no es una “película pendiente”.

En los siguientes diagramas se muestra el flujo del sistema cuando se añade/elimina una película de la lista de pendientes y cuando se devuelven los títulos que la lista contiene:



## 4. Uso del sistema

En este apartado se incluye toda la información necesaria para la instalación, customización y uso del sistema. Además, muestra un conjunto de ejemplos de ejecución para mostrar su funcionamiento y guiar al propio usuario en su uso.

### 4.1. Manual de instalación

Para usar el prototipo, es necesario:

- Instalar las dependencias descritas abajo.
- Descomprimir el fichero del campus virtual.
- Obtener unas credenciales para usar la API de OMDb e incluirlas en el código (en el fichero *omdb\_module.py*). El sistema proporcionado incluye unas credenciales a modo de ejemplo. Para obtener las credenciales se puede utilizar el siguiente enlace: <http://www.omdbapi.com/apikey.aspx>

```
'''  
    PARÁMETROS DE CONFIGURACIÓN DEL MÓDULO  
    '''  
omdb_endpoint='http://www.omdbapi.com/?apikey='  
omdb_apikey='passhere'
```

- Crear una base de datos MongoDB y nombrarla “recomendadorpeliculas”. Lanzarla en “[mongodb://127.0.0.1:27017](http://localhost:27017)”.
- Lanzar el script ubicado en “*db\_creation\_script.py*”. De esta forma, se creará la base de datos “recomendadorpeliculas” y contendrá la información sobre las películas que utilizará el sistema.

intent_recognition_model	21/05/2019 15:37	Carpeta de archivos	
spark-warehouse	21/05/2019 16:00	Carpeta de archivos	
ChatAgent.py	21/05/2019 15:37	Python File	5 KB
Controller.py	21/05/2019 15:37	Python File	15 KB
dataset_manager.py	21/05/2019 15:37	Python File	1 KB
db_creation_script.py	21/05/2019 15:37	Python File	3 KB
DBAgent.py	21/05/2019 15:37	Python File	1 KB
file.test	21/05/2019 16:01	Archivo TEST	1 KB
file.train	21/05/2019 15:37	Archivo TRAIN	1 KB
films_names	21/05/2019 15:37	Archivo	9 KB
intent_recognition.py	21/05/2019 15:37	Python File	6 KB
modelconfiguration.ipynb	21/05/2019 15:37	Archivo IPYNB	11 KB
Modelo.py	21/05/2019 15:37	Python File	2 KB

Para la ejecución del sistema, es necesario instalar distintas librerías Python:

- **MESA**: Modelador de un sistema multi-agente en Python.  
<https://mesa.readthedocs.io/en/master/>














- **RASA**: Librería de procesamiento de sentencias en lenguaje natural. Se utiliza para la obtención de los *intents* y la entidades de la entrada introducida por el usuario. <https://rasa.com/>
- **Spacy**: Librería para procesamiento de sentencias en lenguaje natural. Necesario para el uso de RASA. <https://spacy.io/>
- **TensorFlow**: Librería para el entrenamiento de sistema ML (Machine Learning). Necesario para el uso de RASA. <https://www.tensorflow.org/>
- **Pyspark**: Librería para el procesamiento de grandes volúmenes de datos y entrenamiento de modelos de aprendizaje. <https://spark.apache.org/docs/2.2.1/api/python/pyspark.html>
- **Pymongo**: Librería que incluye un conjunto de herramientas para comunicarse con bases de datos MongoDB. <https://api.mongodb.com/python/current/>

## 4.2. Manual de ejecución y customización

Después de seguir las instrucciones marcadas en el apartado anterior para instalar el sistema y sus dependencias, ya solo hace falta ejecutar el sistema. Para ello, debemos **ejecutar el fichero “run.py”**.

Luego, tan solo es necesario introducir sentencias en lenguaje natural solicitando servicios al sistema, tal y como se muestra en el siguiente apartado.

En el caso de que se desee modificar algún parámetro de la ejecución del sistema, es necesario modificar el fichero “*systemconfiguration.py*”.

 films_names	21/05/2019 15:37	Archivo	9 KB
 intent_recognition.py	21/05/2019 15:37	Python File	6 KB
 modelconfiguration.ipynb	21/05/2019 15:37	Archivo IPYNB	11 KB
 Modelo.py	21/05/2019 15:37	Python File	2 KB
 nlu.md	21/05/2019 15:37	Archivo MD	7 KB
 nlu_config.yml	21/05/2019 15:37	Archivo YML	1 KB
 ntrmdb	21/05/2019 15:37	Archivo	1 KB
 omdb_module.py	21/05/2019 15:37	Python File	5 KB
 pldb	21/05/2019 15:37	Archivo	0 KB
 rmdb	21/05/2019 16:01	Archivo	1 KB
 run.py	21/05/2019 15:37	Python File	2 KB
 systemconfiguration.py	21/05/2019 15:37	Python File	2 KB
 trained_model.py	21/05/2019 15:37	Python File	20 KB

Los parámetros que el sistema permite modificar son:

1. **Técnica de aprendizaje utilizada:** para seleccionarla, introducir el valor deseado en el parámetro *“model\_type”*.

```
'''
Técnica que el modelo entrenado utilizará para su aprendizaje

POSIBLES VALORES:
    1.Red neuronal: neural_net
    2.Árbol de decisión: classification_tree
    3.Clustering mediante kNN: clustering
    4.Regresion logística: logistic_regression
'''

model_type="clustering"
```

2. **Atributos utilizados en el aprendizaje:** indicar qué aspectos (asociados a una película) el sistema debe tener en cuenta en el aprendizaje. De forma predeterminada, se encuentran todos los posibles. Cualquier modificación realizada sobre este parámetro no debe incluir aspectos distintos a los que se muestran en la siguiente imagen. Los aspectos que se eliminan de “parameters” también deben eliminarse de *“numeric\_columns”* y *“string\_columns”*.

```
parameters=['director','Runtime','Genre','Subgenre',
'imdb','rotten','metacritic']

numeric_columns=['Runtime','imdb','rotten','metacritic']
string_columns=['director','Genre','Subgenre','recommend']
```



## 4.3. Ejemplos de uso

A continuación se adjuntan unas pequeñas capturas de ejemplos de uso. Dadas las similitudes presentes entre algunas de las funciones, no se introducirán ejemplos de todas ellas.

### 4.3.1. Obtención de información de una película

#### 1. Obtención del director de una película

```
In [1]: runfile('C:/Users/Bielo/Desktop/Recomendador-de-pel-culas-  
inteligente/run.py', wdir='C:/Users/Bielo/Desktop/Recomendador-de-  
pel-culas-inteligente')
```

What can I help you with?

Who's the director of Gladiator?

-- The director of Gladiator is Ridley Scott

What can I help you with?

Who have directed Gladiator?

-- The director of Gladiator is Ridley Scott

## 2. Obtención de la trama de una película

What can I help you with?

Tell me the plot of Avatar

-- Here you go a summary of the plot of Avatar:  
When his brother is killed in a robbery, paraplegic Marine Jake Sully decides to take his place in a mission on the distant world of Pandora. There he learns of greedy corporate figurehead Parker Selfridge's intentions of driving off the native humanoid "Na'vi" in order to mine for the precious material scattered throughout their rich woodland. In exchange for the spinal surgery that will fix his legs, Jake gathers intel for the cooperating military unit spearheaded by gung-ho Colonel Quaritch, while simultaneously attempting to infiltrate the Na'vi people with the use of an "avatar" identity. While Jake begins to bond with the native tribe and quickly falls in love with the beautiful alien Neytiri, the restless Colonel moves forward with his ruthless extermination tactics, forcing the soldier to take a stand - and fight back in an epic battle for the fate of Pandora.

What can I help you with?

|

## 3. Obtención de las notas de una película en un sitio web

What can I help you with?

What do think the people from rotten of Hidalgo?

-- They don't seem to like it a lot. It has a 4.6

What can I help you with?

Tell me the score of Hidalgo from metacritic

-- They don't seem to like it a lot. It has a 5.4

What can I help you with?

Which is the score of Avatar in imdb?

-- It has a 7.8. Seems that they kind of like it but its not the best film they've seen.

What can I help you with?

#### 4. Obtención de información “general” de una película

What can I help you with?

What can you tell me about The Bourne Legacy?

What film is the one you want general information?  
The Bourne Legacy

-- Well, let's see what I find about The Bourne Legacy  
It's release date was: 10 Aug 2012  
Some people from the cast: Jeremy Renner, Scott Glenn, Stacy Keach, Edward Norton  
And the director is Tony Gilroy  
Summary of the plot:  
The events in this movie take place around the same time as the events in The Bourne Ultimatum (2007). When a British reporter was writing an expose about Black Ops operations Treadstone and Black Briar, and the ones responsible for them are concerned. And when Jason Bourne, former Treadstone operative got the file on Treadstone and Black Briar and gave it to Pamela Landy who then passed it to the media. When the men behind Treadstone and Black Briar learn of this, they're concerned how this will affect other ops they have. They decide it's best to shut down all ops and make sure make everyone involved disappears. They try to take out Aaron Cross who is part of another op called Outcome, but he manages to survive. He then seeks out Dr. Marta Shearing who worked on him when he began. It seems part of the program is for all subjects to take medications but he has run out, which is why he seeks her. But someone tries to kill her. He saves her and she tells him, he should have stopped taking the medications long ago. They go to Manila so that she can help him. Later the men behind Outcome learn that Cross and Shearing are still alive. They try to get them.  
IMDB score:  
It has a 6.7. Seems that they kind of like it but its not the best film they've seen.  
Rotten Tomatoes score:  
They don't seem to like it a lot. It has a 5.6  
Metacritic score:  
It has a 6.1. Seems that they kind of like it but its not the best film they've seen.

What can I help you with?

## 4.3.2. Recibir opiniones sobre las películas

### 1. Opinión positiva

What can I help you with?

I love Insurgent

Could you repeat the movie you enjoyed please?  
Insurgent

-- Did you like it? Perfect. I'll remember it :)

What can I help you with?

Next time recommend me films like American Gangster

-- Did you like it? Perfect. I'll remember it :)

What can I help you with?

|

### 2. Opinión negativa

What can I help you with?

I've watch Hugo and I didn't enjoyed it

Could you repeat the movie you didn't like please?  
Hugo

It seems you don't like Hugo.I'll remember it

What can I help you with?

I don't like Suicide Squad

Could you repeat the movie you didn't like please?  
Suicide Squad

It seems you don't like Suicide Squad.I'll remember it

What can I help you with?



### 4.3.3. Solicitar una recomendación

#### 1. Recomendar una película

What can I help you with?

My favourite film is Avatar

-- Did you like it? Perfect. I'll remember it :)

What can I help you with?

I don't like The Great Gatsby

Could you repeat the movie you didn't like please?  
The Great Gatsby

-- It seems you don't like The Great Gatsby.I'll remember it

What can I help you with?

Tell me a film to watch

-- The Sorcerer's Apprentice

What can I help you with?

I

## 2. Recomendar una película de forma acotada

What can I help you with?

Recommend me a Christopher Nolan film

-- The Dark Knight

What can I help you with?

Recommend me an Action film

-- John Carter

What can I help you with?

Recommend me a film where appears Johnny Depp

Sorry who did you mean?

Johnny Depp

-- Pirates of the Caribbean: Dead Man's Chest

What can I help you with?

Recommend me a large duration film

-- Spider-Man 3

What can I help you with?

## 4.3.4. Gestión de la lista de pendientes

### 1. Insertar películas en la lista de pendientes

What can I help you with?

Put Elysium into the pending list

-- Elysium it's now in your pending list

What can I help you with?

Insert Alexander into the pending list

-- Alexander it's now in your pending list

What can I help you with?

I want you to insert Gladiator into the pending list

-- Gladiator it's now in your pending list

## 2. Eliminar películas de la lista de pendientes

```
What can I help you with?  
Pop Hidalgo from the pending list  
-- Hidalgo is not in your pending list  
What can I help you with?  
pop Gladiator from the pending list  
-- Gladiator popped from the pending list  
What can I help you with?
```

## 3. Listar el contenido de la lista de pendientes

```
What can I help you with?  
list the pending list  
-- This is your pending list:  
Elysium  
Alexander  
  
What can I help you with?  
Tell me whats in the pending list  
-- This is your pending list:  
Elysium  
Alexander  
  
What can I help you with?  
,
```

## 5. Informe de división de tareas

Ver excel adjunto Informe de Reparto. En él se agrupan todas las tablas Excel que se han ido enviando en los entregables de cada semana.