



UNIVERSIDADE DA CORUÑA

FACULTADE DE INFORMÁTICA

TRABALLO FIN DE GRAO

GRAO EN ENXEÑARÍA INFORMÁTICA

Mención en Enxeñaría do Software

**Aplicación web para a análise do
comportamento humano en zoas transitadas**

Autor: Xoan Iago Suárez Canosa

Director: Cancela Barizo, Brais

Director: González Penedo, Manuel Francisco

Director: Novo Buján, Jorge

Director: Ortega Hortas, Marcos

A Coruña, 13 de agosto do 2015

*A Meus pais Elvira e Manolo, que grazas ao seu
esforzo e cariño permitíronme chegar ata aquí*

Agradecementos

En primeiro lugar grazas a Brais Cancela por toda a axuda e o apoio que me brindou ao longo destes meses, xa que sen a súa achega este proxecto non tería nin sequera comezado. Grazas tamén a tódolos compañeiros da facultade que coñecín ao longo deste increíbles anos, deles aprendín a meirande parte do que sei e ademais pasamos momentos inesquecibles. Por último agradecer á miña familia e ao resto dos meus amigos por soportar día a día cada un dos meus defectos e permitirme gozar das vosas virtudes.

Resumo

No mundo da seguridade, un dos maiores retos que se propoñen hoxe en día é a detección de condutas sospeitosas. Esta situación vólvese cada vez máis complexa ao aumentar o número de cámaras a vixiar, polo que é imprescindible dispoñer dunha ferramenta que facilite esta tarefa.

Este proxecto consiste nunha aplicación web que emprega funcionalidades relativas á análise do comportamento en zoas transitadas, incluíndo visualización de vídeo e distintas capas que mostran información de alto nivel sobre o comportamento detectado.

En concreto esta ferramenta encargase de detectar a todos aqueles obxectos ou persoas que aparecen nunha secuencia de vídeo, aplicando un algoritmo para medir o 'grao de anormalidade' da súa conduta en base aos movementos que realizan.

Palabras chave

Análise Comportamento, Comportamento Humano, Aplicación Web, Django.

Índice xeral

1. Introducción e Obxectivos	1
2. Fundamentos Teóricos e Conceptos Previos	3
2.1. Arquitectura web	3
2.2. Análise do comportamento	4
2.3. Programación Web	5
2.3.1. Desenvolvemento Áxil	5
2.3.2. Soporte para transaccións	6
2.3.3. Object-Relational Mapping (ORM)	6
2.3.4. Xestión de Layout	6
2.3.5. AJAX	6
2.3.6. Outras cuestións da web	7
2.4. O Vídeo	7
2.4.1. Codec's	7
2.4.2. Formato de Vídeo	8
2.4.3. Streaming de Vídeo	9
2.4.4. Pseudo Streaming ou Descarga Progresiva	9
2.4.5. Streaming	9
3. Análise de antecedentes e alternativas	11
4. Metodoloxía seguida no desenvolvemento de Proxecto	13
4.1. As metodoloxías áxiles	14

4.2. Persoas	14
4.2.1. ProductOwner	14
4.2.2. ScrumMaster e Development Team	14
4.3. Reunións	14
4.3.1. Sprint Planning Meeting	14
4.3.2. Daily Scrum	14
4.3.3. Sprint Review	15
4.3.4. Sprint Retrospective	15
4.4. Control de Versións con GitHub	15
4.5. Integración Continua con Travis CI	15
4.6. Control da cobertura con Coveralls	17
4.7. Xestión de Incidencias e Control de Proxecto con YouTrack	17
5. Estudo comparativo das tecnoloxías web	19
5.1. Back-End	20
5.1.1. Java	20
5.1.2. C#	20
5.1.3. C - C++	21
5.1.4. Python + Django	21
5.2. Front-End	21
5.2.1. Vídeo en Flash	22
5.2.2. Vídeo HTML5 + Javascript	22
6. Tecnoloxías Empregadas	25
6.1. HTML5	25
6.2. CSS3	26
6.2.1. Twitter Bootstrap	26
6.3. Javascript	27
6.3.1. jQuery	28
6.3.2. Qunit	29
6.3.3. Image-picker	29

6.3.4. jQuery UI	29
6.3.5. tablesorter	29
6.4. OpenCV	30
6.5. Extensible Markup Language (XML)	30
6.6. ffmpeg	31
7. Funcionalidades Destacadas	33
7.1. Control de Usuarios	33
7.2. Carga de Vídeo	33
7.3. Lista de vídeos e Imaxe de Portada	34
7.4. Reprodución de Vídeo	37
7.5. Análise de Vídeo	38
7.5.1. Interface de Liña de Comandos	38
7.5.2. Ficheiro XML	39
7.5.3. Paquete XmlRecognition	40
7.5.4. A análise dende a capa web	41
7.6. Mostrar Deteccións	43
7.7. Traxectorias	47
7.8. Lista de Deteccións	47
7.9. Comportamento anormal	47
8. Probas Realizadas	49
8.1. Probas Unitarias	49
8.1.1. Política de acceso ás páxinas web	49
8.1.2. Probas da Capa Web con Javascript	50
8.2. Probas de Integración	51
8.2.1. Probas Funcionais Selenium	51
8.3. Probas de Sistema	52
8.4. Probas de Aceptación	52
9. Calidade	53

10. Planificación e Avaliación de Custes	55
11. Resultados e Conclusións	57
12. Liñas Futuras	59
12.1. Vídeo en Directo	59
A. Título	61
A.1. Lista de Acrónimos	61
A.2. Manual de Usuario	61
A.3. Manual de referencias Técnicas	61
A.4. Notas acerca da Terminoloxía	61

Índice de figuras

2.1. Clásica arquitectura dunha aplicación web empresarial	5
2.2. Diagrama conexión RTPS	10
4.1. Imaxe de parte do ficheiro .travis.yml	16
7.1. Diagrama de secuencia do proceso seguido cando se fai o Submit do Formulario de creación de vídeos	35
7.2. Captura de pantalla da páxina web SuccessfulUpload	36
7.3. tag en html5, coas súas fontes e coas capas < <i>canvas</i> > asociadas	37
7.4. Interfaze de liña de comandos do Sistema de recoñecemento	39
7.5. Diagrama de clases do paquete XmlRecognition	41
7.6. Capturas de pantalla do sistema de Notificacións	42
7.7. Diagrama de Clases do sistema	43
7.8. Diagrama de secuencia da análise do vídeo na capa web	44
7.9. Diagrama de clases do patrón observador na capa web	45
7.10. Xeración do atributo fps no lado servidor	46

Índice de Táboas

7.1. My caption	38
---------------------------	----

Capítulo 1

Introdución e Obxectivos

O seguimento de obxectos é o proceso de estimar no tempo a localización de un ou mais obxectos en movemento empregando as imaxes captadas por una cámara. A crecente mellora na potencia de cálculo dos procesadores actuais, xunto coa dixitalización dos sensores de imaxe propiciou dende comezos de século a aparición de novos algoritmos de análise que aportan cuantiosas melloras a este campo.

Neste aspecto, o Grupo de Visión Artificial e Recoñecemento de patróns (VARPA) da UDC leva anos investigando para aportar á comunidade científica os seus propios algoritmos e desenvolver novas aplicación que empreguen estes algoritmos para detección de persoas, vehículos, ou calquera outro obxecto susceptible de seres estudado. En concreto, o grupo posúe ferramentas que permiten o seguimento en zoas transitadas nas que poden aparecer multitude de obxectos a seguir simultaneamente.

Co fin de achegar estes métodos de análise á súa aplicación final, propónse dende o laboratorio a construción dunha web, que sexa capaz de reproducir vídeos, e sobre eles mostrar distintas capas con información de alto nivel, como pode ser a resultante de detectar obxectos, medir o seu grao de anormalidade, a súa velocidade, etc.

Seleccionase unha arquitectura web xa que a diferenza das arquitecturas de escritorio, proporciona ás persoas que acceden á web independencia do Sistema Operativo empregado e dispoñibilidade dende calquera lugar con acceso a rede, evitando así as dificultades asociadas coa instalación ou actualización da aplicación.

Capítulo 2

Fundamentos Teóricos e Conceptos Previos

O desenvolvemento dunha aplicación web non é algo trivial e mais se temos en conta todas as peculiaridades que este proxecto contén. Para a súa comprensión é preciso coñecer unha serie de conceptos teóricos que se expoñen a continuación:

2.1. Arquitectura web

Neste proxecto séguese unha Arquitectura Web baseada no modelo cliente-servidor, que consiste nun lado servidor que distribúe os recursos como poden ser o contido multimedia (vídeos, imaxes, etc) ou as páxinas web ao outro lado, o cliente, que típicamente corre nun navegador web interpretando as páxinas html e o código javascript asociado a estas.

No noso caso a parte servidor estará dividida en dúas compoñentes claramente diferenciadas, o sistema para a análise do comportamento, e a aplicación web que permitirá o acceso a este, os fundamentos de ámbalas dúas partes explícanse a continuación.

2.2. Análise do comportamento

É un dos campos de investigación mas activos hoxe en día. A idea principal na que se centran estes sistemas como o que nos ocupa é a de detectar calquera acción levada a cabo polos obxectos involucrados nunha escena de vídeo. Un obxecto é calquera cousa que debe ser seguida, polo que dependendo do tipo de problema estes obxectos poden ser dunha natureza ou doutra. O tipo de accións a detectar tamén depende da tipoloxía do sistema, xa poden ser comportamentos individuais(camiñar, correr, loitar...) ou grupais (reunirse, abandonar un grupo de persoas...).

Tanto neste proxecto como nos sistemas para a análise do comportamento en xeral, pódense discernir tres tarefas importantes que colaboran entre si[1]:

- **Detección de Obxectos:** Partindo dunha secuencia de vídeo como entrada obtéñense os distintos obxectos que aparecen en cada fotograma da escena. Para este fin empréganse técnicas de visión por computador.
- **Seguimento de Obxectos:** A partires da información obtida na detección, asígnanselle identificadores a cada obxecto detectado no vídeo, agrupando se procede distintos obxectos baixo o mesmo identificador en caso de considerarse que estes obxectos forman parte de un grupo ou unha mesma detección.
- **Análise do comportamento de Alto Nivel:** Unha vez obtida a información dos dous pasos anteriores pódese catalogar o comportamento de cada detección empregando técnicas de recoñecemento de patróns.

Os resultados mais destacables destas técnicas cos que a aplicación terá que traballar serán:

- A lista de obxectos detectados para cada un dos fotogramas e a súa posición neles
- A traxectoria de cada un dos obxectos detectados
- O grao de anormalidade da traxectoria seguida por un obxecto en cada un dos fotogramas
- A velocidade de un obxecto determinado.

Estas tres tarefas requiren dunha serie de cálculos matemáticos baseados en técnicas moi diversas, para realizar toda esta serie de cálculos, empregase algunha biblioteca de código que simplifique o traballo a realizar, e neste caso, esta biblioteca é OpenCV.

2.3. Programación Web

A programación web de aplicacións de carácter empresarial require do coñecemento da rede, ademais do de unha serie de ferramentas e estratexias para chegar a un deseño sostible e de calidade.

A arquitectura clásica das aplicacións web pódese ver no gráfico 2.1

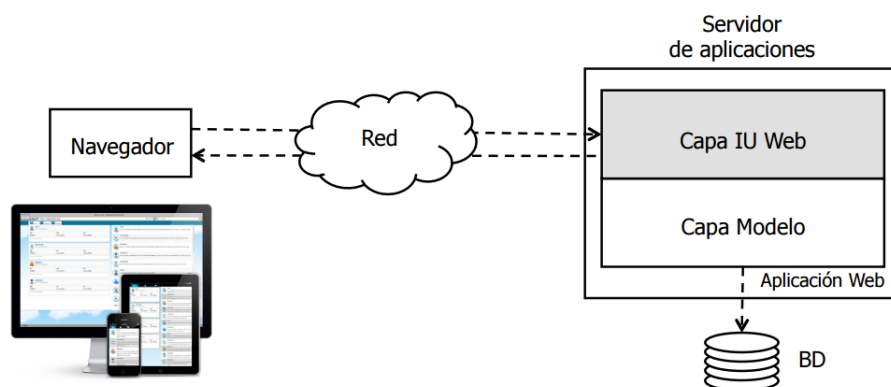


Figura 2.1: Clásica arquitectura dunha aplicación web empresarial

As técnicas e estratexias mais importantes á hora de construír unha aplicación web relátanse nos puntos subseguintes:

2.3.1. Desenvolvemento Áxil

Para reducir custos e poder proporcionar solucións rápidas é preciso que as aplicacións web's de carácter empresarial se leven a cabo en pouco tempo e con bos principios de enxeñaría, a isto contribúen en gran medida as tecnoloxías que se amosan a continuación.

2.3.2. Soporte para transaccións

Unha transacción nun Sistema Xestor de Base de Datos (SGBD) é un conxunto de ordes que se executan formando unha unidade de traballo, de forma invisible e atómica. As transaccións cobran gran importancia nas aplicacións web debido á inestabilidade da rede e á concorrencia dos distintos clientes conectados, polo que é axuda a un desenvolvemento moi áxil que a tecnoloxía traia a súa xestión integrada.

2.3.3. Object-Relational Mapping (ORM)

Os mapeado obxecto-relación é unha das técnicas de programación que mais velocidade imprimen na construción de webs, xa que converte os datos dunha linguaxe Orientada a Obxectos (OO) a datos de un sistema relacional no que son persistidos e viceversa, aforrando ao programador o traballo de ter que programar o código para esta tarefa. É desexable pois que a tecnoloxía a empregar dispoña de un mapeador obxecto-relacional ben integrado, algúns exemplos disto poden ser: a combinación Java+Maven+Hibernate, o EF(Entity Framework) de Microsoft ou os Models de Django.

2.3.4. Xestión de Layout

Tamén resulta moi practico dispoñer dunha linguaxe de prantilla que permitan xerar contido html ben estruturado dinamicamente. Algúns exemplos son o Sistema de Templates de Django, o Sistema JSP de Spring, a librería Thymeleaf ou os compoñentes de ASP.NET. Todos eles axudan a xerar contido HTML de xeito sinxelo e escalable que logo será enviado ao cliente, de todos os xeitos esta parte cliente as veces precisa comunicarse co servidor sen que sexa preciso unha recarga da páxina e para elo empregase AJAX.

2.3.5. AJAX

AJAX ou Asynchronous JavaScript And XML é unha tecnoloxía da web empregada para crear aplicacións interactivas, estas aplicacións executanse no navegador dos usuarios mentres manteñen unha comunicación asíncrona co lado servidor en segundo

plano. Normalmente emprégase javascript coma linguaxe para a realización das chamadas asíncronas en combinación con algunha linguaxe para a definición de obxectos como XML ou JSON, para as que ademais os propios navegadores adoitan a facilitar ferramentas de parsing.

2.3.6. Outras cuestións da web

A maiores existe toda unha gama de outras funcionalidades que cobran importancia cando deseñamos e construímos unha web como o manexo de erros nos formularios, internacionalización (i18n), visualización de grande cantidades de datos(en listas ou táboas), seguridade...

2.4. O Vídeo

O vídeo permite gravar, procesar, almacenar e transmitir información en forma de imaxe en movemento, esta imaxe en movemento soe estar composta por unha serie de imaxes estáticas chamadas fotogramas, que se manexan a unha velocidade alta causando o efecto de que o que se está a ver está en movemento.

Non obstante o vídeo en formato electrónico non almacena necesariamente todas as imaxes de forma individual xa que isto suporía moita información redundante. En lugar disto empréganse técnicas de codificación-decodificación (codec's) que comprimen e descomprimen os datos para facilitar o seu manexo.

2.4.1. Codec's

Os codec's teñen como función principal a de transformar unha sinal de vídeo para que poida ser visto. A maioría dos codec's provocan con cada transformación unha perda de información para conseguir un tamaño final o mais pequeno posible, estes codec's chámanse lossless (con perda) e a pesares de que perden calidade soe compensar pola cantidade de espazo que aforran.

A parte de este fenómeno da compresión tamén cobra moita importancia outros aspectos relacionados co vídeo como a reprodución e sincronización de son, os subtítulos

do vídeo... Todo isto depende do formato no que se almacene o vídeo.

2.4.2. Formato de Vídeo

O formato dun vídeo determina como se almacenan os distintos tipos de información involucrada como as imaxes que poden estar codificadas en varios codec, o son, os subtítulos... este formato correspóndese cunha extensión específica do arquivo que o contén, como por exemplo:

- **AVI (Audio Video Interleaved):** Sendo un dos formatos máis famosos pode conter un vídeo dunha calidade excelente pero soe requiren dunha gran capacidade. Os codec's que se soen empregar neste formato pola súa capacidade de compresión e calidade aceptable son DivX y XviD, inda que tamén se permiten outros como DV(Digital Video), CinePak...
- **MKV (Matroska):** É un formato de código aberto que basea o seu nome nas clásicas bonecas Matrioskas. Ten capacidade para conter tanto vídeo, son e subtítulos en diferentes idiomas, empregándose como códec de vídeo normalmente algunha implementación de H.264, como por exemplo x264. Mentres que para o son é habitual empregar o codec de audio Vorbis.
- **WebM (Google, 2010):** Un dos formatos máis recentes é o WebM (WebMovie), un proxecto lixeiramente baseado en Matroska adquirido e liberalizado por Google en 2010 co obxectivo de empregalo con HTML5 como estándar libre. O formato ten un excelente rendemento e xunto ao codec VP9, motivo polo cal forma parte dos recomendados pola W3C.
- **Formato OGG (Xiph.Org, 1993):** O formato contedor OGG é un formato libre deseñado para incluír vídeo, son, subtítulos e metadatos. O vídeo en este formato soe estar codificado co codec Theora, que se basea nunha versión liberada de VP3. Tamén se emprega para este tipo de empaquetado a extensión .OGV, mais o que marca o estándar é a extensión .OGG.
- **MP4 - MPEG (Moving Pictures Expert Group):**

O Moving Picture Experts Group (MPEG) é un grupo de expertos da ISO (Organización Internacional de Normalización) e da Comisión Electrotécnica Internacional (IEC) para crear estándares en canto ao mundo do audio e o vídeo.

Froito do traballo deste grupo naceron os formatos MPEG-1 (calidade CD), MPEG-2 (calidade DVD), MPEG-3 (orientado ao audio MP3) e MPEG-4 que é o que mais nos interesa xa que vai enfocado á compresión de vídeo e son na web, podendo incluír tamén subtítulos ou imaxes de referencia. O ficheiros de este último formato teñen extensión .mp4.

2.4.3. Streaming de Vídeo

Dado que este proxecto está centrado no tratamento de vídeo, é de especial importancia ver de que xeitos podemos distribuílo e reproducilo a través da rede. A estes efectos existen dúas grandes alternativas que varían en canto ao seu grao de escalabilidade, dificultade de implementación, e calidade final do servizo:

2.4.4. Pseudo Streaming ou Descarga Progresiva

Consiste na descarga do vídeo por fragmentos, típicamente empregando o protocolo HTTP. Neste formato, o reprodutor vai acumulando fragmentos de vídeo ata obter os precisos como para comezar a reprodución, mais se o ancho de banda fose insuficiente, o vídeo remataría por pararse. Este sistema é o empregado por servizos como YouTube, Vimeo, DailyMotion...

Será a opción empregada por motivos de simplicidade, mais compre explicar tamén o verdadeiro Streaming, xa que é a diferenza do pseudo-streaming pode ser empregado para a emisión de contido en directo como o dunha cámara de seguridade.

2.4.5. Streaming

O verdadeiro streaming (do inglés True Streaming) consiste na emisión en directo do contido multimedia a través da rede, que o reprodutor reproduce no momento que recibe. Este outro xeito de distribuír vídeo, apoiase en axustar a calidade do vídeo ao ancho de banda do que dispón o cliente, evitando así interrupcións na reprodución.

O protocolo mais destacable á hora de empregar este tipo de streaming é RTSP (Real-Time Streaming Protocol) que operando a nivel de aplicación permite controlar un ou varios fluxos sincronizados de contido multimedia.

Por unha parte RTSP soe empregar o Real-Time Transport Protocol (RTP) sobre UDP (User Datagram Protocol) para o transporte de contido multimedia, maximizando así o emprego da rede pero sen garantir un mínimo na calidade do servizo.

E por outra parte RTSP emprega o Real-time Control Protocol (RTCP) sobre TCP (Transmission Control Protocol) para a transmisión periódica de paquetes de control da sesión, o diagnóstico de fallos e o control de la calidade da transmisión.

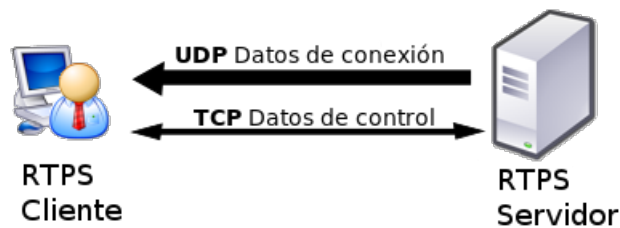


Figura 2.2: Diagrama conexión RTPS

RTSP asemellase a HTTP no formato das peticións/reposas e na sintaxe, pero dispoñendo dun estado que permite tanto a clientes como a servidores facer peticións.

Tamén existen outros protocolos propietarios como MMS (Microsoft Media Server) ou RTMP (Real-Time Messaging Protocol) e RTMFP (Real-Time Media Flow Protocol) de Adobe.

Capítulo 3

Análise de antecedentes e alternativas

Se trata de realizar un estudio de alternativas o “estado del arte” o un análisis comparativo de alternativas.

Se exponen las diferentes alternativas que se han evaluado o que se consideran de interés, a lo realizado en el proyecto. Fundamentalmente se trata de otras herramientas existentes que realizan algo similar, sean o no comerciales, o de prototipos de investigación relacionados, o de estudios que tratan aspectos similares.

Buscar por internet produtos que fagan algo similar...

Capítulo 4

Metodoloxía seguida no desenvolvemento de Proxecto

Os diferentes obxectivos do proxecto abordáronse seguindo a Metodoloxía SCRUM, adaptada a un proxecto de un solo Developer.

Esta metodoloxía áxil tamén chamada melé pola súa inspiración no Rugby, permite un desenvolvemento rápido en situacións de requisitos inestables. Apoíase no seu carácter iterativo e incremental, dividindo traballo a realizar en períodos de aproximadamente un mes chamados Sprint's.

Para a realización deste traballo de fin de grao foi preciso adaptala, pois está pensada en principio para organizar equipos de entre 3 a 9 persoas (Team). Por outro lado, o marco de traballo planifica reunións diarias (Daily Scrum), ao supoñer que todos os membros do equipo traballan unha xornada laboral enteira entre cada unha destas reunións, o cal tampouco se dá no caso deste proxecto, xa que a dedicación será de determinadas horas nos momentos dispoñibles.

4.1. As metodoloxías áxiles

4.2. Persoas

Os tres papeis que se definen nesta metodoloxía [2] foron adaptados do seguinte xeito:

4.2.1. ProductOwner

O papel de ProductOwner, que define os requisitos da aplicación estivo representado polo director de proxecto Brais Cancela, que participou na creación do Anteprojecto. En certos momentos o señor Cancela tamén desempeñou a función de membro do equipo, posto que é foi autor do algoritmo de análise de vídeo.

4.2.2. ScrumMaster e Development Team

Ambos papeis leváronse a cabo polo autor, xa que carece de sentido definir ambas figuras nun equipo de unha soa persoa. De este xeito á par que se desenvolvía o proxecto, íase asegurando o cumprimento das regras de SCRUM.

4.3. Reunións

As reunións pola súa parte modifícanse do seguinte xeito:

4.3.1. Sprint Planning Meeting

Esta reunión mantén o mesmo formato que no SCRUM puro, xuntando ao autor co ProductOwner e concretando as tarefas do Product Backlog que se realizarán no seguinte Sprint, pasando por tanto a formar parte do Sprint Backlog.

4.3.2. Daily Scrum

Dado que o equipo de Desenvolvemento e o ScrumMaster están conformados pola mesma persoa e que o número de horas diarias adicadas é moito menos ao dunha xornada laboral, considerouse oportuno substituír esta reunión diaria por unha reunión

dúas veces á semana (Martes e Xoves pola tarde normalmente). Na que se mostrase ao ProductOwner o avance do proxecto.

4.3.3. Sprint Review

Esta reunión fusionase co Sprint Planning Meeting, xa que ao mesmo tempo valorase o traballo realizado no Sprint que remata e, en base a el, planifícase a videira Iteración.

4.3.4. Sprint Retrospective

Pola súa parte, esta reunión toma un carácter unipersoal, pasando a ser unha valoración do propio autor sobre as persoas, relacións, procesos e ferramentas implicadas no último Sprint. Nela avalíase os elementos con éxito e os suxeitos a melloras, creando un plan para implementar estas melloras na Videira iteración.

4.4. Control de Versións con GitHub

Os sistemas de control de versións permiten a xestión dos distintos cambios efectuados sobre un produto software ou sobre a súa configuración. Facilitando a administración das distintas versións do produto.

En concreto, GitHub é unha plataforma de desenvolvemento colaborativo que emprega o sistema de control de versións Git. Escolleuse empregar este sistema pola súa potencia e simplicidade, xa que proporciona libre acceso aos titores para comprobar o avance do proxecto, e a súa vez asegura que o código este sempre a bo recaudo.

A páxina do proxecto é: <https://github.com/iago-suarez/ancoweb-TFG>

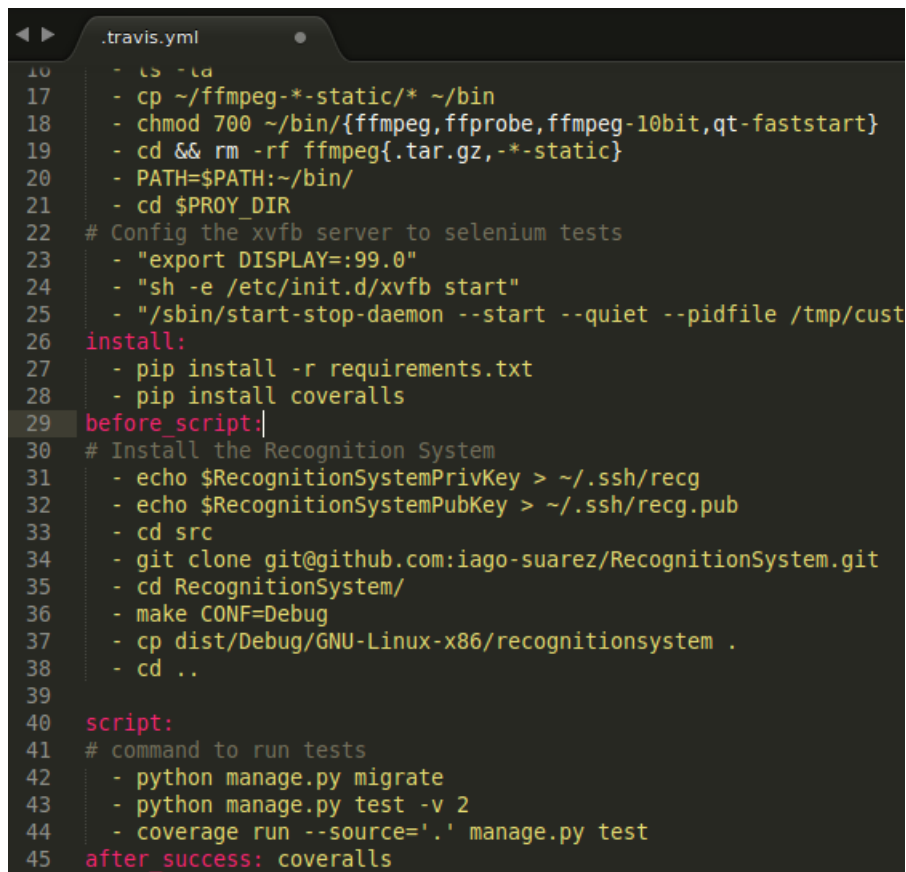
4.5. Integración Continua con Travis CI

A Integración Contínua (CI do inglés Continuous Integration) é un modelo informático que consiste en facer integracións automáticas dun proxecto o mais a miúdo posible para así poder detectar os posibles erros o antes posible, minimizando as súas posibles consecuencias. Outro factor importante é o feito de garantir que a versión subida ao repositorio segue a funcionar con independencia do entorno de desenvolvemento.

Enténdense como pertencentes á integración continua a compilación e a execución das probas de todo un proxecto.

Travis CI é unha plataforma de integración continua para proxectos aloxados en GitHub, que detecta automaticamente cando se produce un cambio no repositorio, e executa unha serie de pasos definidos no ficheiro `.travis.yml` 4.1, que contén as accións a realizar antes, durante e tras a as probas.

Escolleuse Travis CI, pola súa integración con GitHub, pola súa potencia (permite executar practicamente todo o que se pode executar nunha máquina local) e pola súa sinxela integración con outras ferramentas como Coveralls.

A screenshot of a code editor showing a portion of a `.travis.yml` file. The file is dark-themed with syntax highlighting. The visible lines are numbered 16 to 45. The content includes commands for installing FFmpeg, configuring xvfb, installing Python dependencies (requirements.txt, coveralls), cloning a repository, building a project, running tests with coverage, and finally running coveralls after a successful build.

```
16 - ls -la
17 - cp ~/ffmpeg-*-static/* ~/bin
18 - chmod 700 ~/bin/{ffmpeg,ffprobe,ffmpeg-10bit,qt-faststart}
19 - cd && rm -rf ffmpeg{.tar.gz,-*-static}
20 - PATH=$PATH:~/bin/
21 - cd $PROY_DIR
22 # Config the xvfb server to selenium tests
23 - "export DISPLAY=:99.0"
24 - "sh -e /etc/init.d/xvfb start"
25 - "/sbin/start-stop-daemon --start --quiet --pidfile /tmp/cust
26 install:
27 - pip install -r requirements.txt
28 - pip install coveralls
29 before_script:
30 # Install the Recognition System
31 - echo $RecognitionSystemPrivKey > ~/.ssh/recg
32 - echo $RecognitionSystemPubKey > ~/.ssh/recg.pub
33 - cd src
34 - git clone git@github.com:iago-suarez/RecognitionSystem.git
35 - cd RecognitionSystem/
36 - make CONF=Debug
37 - cp dist/Debug/GNU-Linux-x86/recognitionssystem .
38 - cd ..
39
40 script:
41 # command to run tests
42 - python manage.py migrate
43 - python manage.py test -v 2
44 - coverage run --source='.' manage.py test
45 after_success: coveralls
```

Figura 4.1: Imaxe de parte do ficheiro `.travis.yml`

4.6. Control da cobertura con Coveralls

4.7. Xestión de Incidencias e Control de Proxecto con YouTrack

Capítulo 5

Estudo comparativo das tecnoloxías web

Para a elaboración de este traballo de fin de grao, é preciso seleccionar unha serie de tecnoloxías tanto para o lado servidor coma para o lado cliente, vamos a empregar os seguintes criterios para poder comparalas entre si e escoller entre elas a que mellor se adecúan ás necesidades do proxecto.

■ **Plataforma e Portabilidade**

Segundo as especificacións iniciais do proxecto, e de cara a facilitar a implementación deste, as ferramentas empregadas deben de executarse baixo os Sistemas Operativos baseados en Linux.

■ **Compatibilidade co algoritmo de Procesado de Vídeo**

Dado que o algoritmo que procesa o vídeo foi parcialmente implementado en C++, a tecnoloxía que se empregue para o desenvolvemento da parte web debe dispoñer da máxima compatibilidade con esta linguaxe de programación.

■ **Desenvolvemento Áxil**

A tecnoloxía que se seleccione ten que minimizar o custe en tempo e esforzo da implementación, por este motivo valorarase positivamente que dispoña de IDE's axeitados, facilidades de acceso a BD(Base de Datos),se é posible ORM(Object-

Relational Mapping) integrado, recarga en quente... En xeral todo aquilo que permita axilidade e flexibilidade.

Posto que empregamos unha arquitectura baseada no modelo cliente-servidor, temos que determinar por un lado en que tecnoloxías vamos a construír o Servidor ou Back-End, e por outra parte o Cliente ou Front-End que se executará nun navegador.

5.1. Back-End

En canto ás tecnoloxías para a elaboración do Back-End, hai que ter en conta que procesará os datos proporcionados polo Front-End, atendendo as súas peticións, e xestionando o modelo de datos e os procesos implicados na aplicación. A maiores neste caso en concreto, o Back-End será o encargado de interactuar directamente co sistema de Análise de Vídeo.

As tecnoloxías estudadas para esta parte do sistema son as seguintes:

5.1.1. Java

Java é unha das linguaxes mais empregadas actualmente. Ademais existen diversos frameworks web como Tapestry ou SpringMVC e en canto as BD Hibernate, que facilitan o seu uso, mais é preciso integralos xa que non forman parte da plataforma en si. A súas posibilidades de integración con c++ son altas grazas á interface JNI(Java Native Interface), pero a súa configuración pódese volver tediosa. Un dos proxectos estudados para ver o seu funcionamento é Red5 [3].

5.1.2. C#

Esta linguaxe en combinación con .NET resulta unha combinación bastante áxil de cara á programación web, integrando na propia plataforma un deseñador Web e un ORM moi intuitivo. O gran problema polo que se descartou este entorno foi polo seu baixo grao de compatibilidade cos sistemas operativos Linux.

5.1.3. C - C++

C++ presenta como era de esperar a maior compatibilidade co algoritmo implementado, non obstante, inda que existen algunhas utilidades que facilitan o desenvolvemento web con esta linguaxe como Wt (Web Toolkit)[4], o grado de axilidade está moi por baixo do que facilitan o resto das combinacións. Tamén pode resultar de interese o coñecido proxecto Icecast[5], que fai streaming de vídeo sobre unha interface web.

5.1.4. Python + Django

Python presentase como a mellor opción para desenvolver o lado servidor, por unha parte dispón do módulo Subprocess[6] que permite executar calquera comando pola terminal maximizando así a modularidade e a integración co algoritmo en C++. Por outra parte Django[7] contén un potente ORM e un sistema de "Templates" que simplifica a parte web. Para concluír cabe destacar o feito de que python sexa unha linguaxe interpretada, xa que isto evita o paso previo de compilación.

5.2. Front-End

Unha vez seleccionada a tecnoloxía do lado servidor, é hora de ver que opcións existen para o Front-End, a parte do Sistema encargada da interacción co usuario.

Por unha parte están as tecnoloxías que pola súa transcendencia e nivel de aceptación cosideranse xa imprescindibles no desenvolvemento web, estamos a falar de HTML e CSS linguaxes de facto para definir o contido e o aspecto visual respectivamente dunha páxina web. De estas linguaxes seleccionaremos as súas versións mais recentes, que a día de hoxe son HTML5 e CSS3.

Sen embargo, noutros campos coma son a reprodución de vídeo e o control de elementos dinámicos non existe unha tecnoloxía que abarque a case completitude da rede, é por elo que neste capítulo estudaremos aqueles xeitos que permitan a reprodución de vídeo e a maiores o debuxado de figuras en movemento sobre este vídeo.

En canto á reprodución de vídeo e o control deste destacan principalmente dúas

alternativas:

5.2.1. Vídeo en Flash

Vídeo Flash é a tecnoloxía de reprodución de vídeo mais empregada e madura en internet dende hai anos. Inicialmente creada por Macromedia e mercada por Adobe en 2005, permite crear elaboradas animacións vectoriais, que logo poden proxectarse sobre un vídeo, mentres que tamén manexa os eventos de reprodución de vídeo como o play ou o stop.

Ten certos problemas en tanto ao Posicionamento Web(SEO), reprodución en dispositivos móbiles, accesibilidade... pero o meirande de todos eles é que mentres que outros dos exemplos estudados son 100 % libres, Flash é un programa propietario para o que é preciso adquirir unha licenza.

5.2.2. Vídeo HTML5 + Javascript

Esta é outra das combinacións mais empregadas actualmente, xa que segue o estándar do W3C (World Wide Web Consortium)[8] no que se define como se han de mostrar e obter os vídeos dunha páxina codificada coa linguaxe HTML5., e destaca por seres extremadamente sinxelo en comparación con Flash ou outras tecnoloxías.

Se o comparamos con Flash podemos ver a seguintes **vantaxes**:

- Resulta moito mais sinxelo de codificar grazas a que é o navegador o que se encarga da reprodución do vídeo mentres que o programador só define o xeito de obtelo.
- Non precisa da instalación de ningún Plugin que poida dar problemas por exemplo en dispositivos móbiles.
- Mentres que HTML5 e Javascript son libres, Flash é unha tecnoloxía propiedade de Adobe.
- HTML5 + CSS3 dispón de mais facilidades se buscamos un deseño “responsive”.

Será por tanto a alternativa escollida para a construción deste proxecto, e en canto ao debuxo de figuras sobre o vídeo escolleremos o elemento `< canvas >` tamén de HTML5. Todas estas tecnoloxías e moitas mais explícanse en detalle no seguinte capítulo.

Capítulo 6

Tecnoloxías Empregadas

O feito de traballar na web, e moito mais o de facelo no ámbito da análise de vídeo, requiren que este sexa un proxecto cuns altos niveis de integración no que toman parte toda unha serie de librerías e ferramentas software que axudan a alcanzar os fins desexado.

A continuación explicase detidamente cal é a función de cada unha das tecnoloxías que forman parte deste proxecto co fin de comprender o explicado nos capítulos seguintes. Comezamos polas tecnoloxías empregadas no lado servidor:

6.1. HTML5

HTML (HyperText Markup Language) é a linguaxe de marcas empregada en internet para a elaboración de páxinas web, define unha estrutura básica e un código para a definición do contido da páxina como poden ser texto, imaxes, vídeos... Existen diversas versións deste estándar, mais para este proxecto empregarase a súa última versión HTML5, que inclúe toda unha serie de novos elementos.

Un dos compoñentes que mais empregaremos para a construción desta web será o elemento `< video >` de HTML5 que achega unha serie de Métodos, Eventos e Propiedades [9] que poden empregarse dende o código Javascript.

A maiores do propio contido da páxina, HTML permite incluír referencias a outros ficheiros que están tamén asociados á páxina como poden ser ficheiros javascript ou css

que se mostran de seguido.

6.2. CSS3

HTML permite unha perfecta estruturación dos elementos que compoñen unha páxina web, pero en caso de que queiramos personalizar o estilo da páxina web, ou adaptala a diferentes dispositivos precisamos CSS. CSS(Cascading Style Sheets) é unha linguaxe para definir e crear a presentación dun documento HTML.

Para elo defínense unha ou mais follas de estilos, que definen para cada elemento seleccionado nelas unha serie de características de estilo. No caso da aplicación a desenvolver a política de follas de estilo será a de crear unha folla de estilo xeral para conter as características de estilo comúns a todo o proxecto e a maiores as que sexan precisas para páxinas ou elementos concretos, todo elo traballando coa versión 3 desta linguaxe.

Por sorte, para simplificar o traballo na definición do estilo xeral da páxina existen librerías CSS como bootstrap que definen parte do código preciso, isto dá un estilo uniforme á web ao tempo que axuda a reducir o tempo de implementación.

6.2.1. Twitter Bootstrap

Twitter bootstrap é un framework ou conxunto de ferramentas de código aberto para deseñar webs. Contén todo un conxunto de platillas como tipografía, botons, formularios, táboas, barras de navegación... Estas plantillas normalmente consisten simplemente nun estilo CSS, pero ás veces tamén hai un procesado en javascript como o da función popover que permite xerar unha ventá flotante para amosar os datos dos obxectos detectados na aplicación.

A pesar de todas as cousas que permiten facer HTML + CSS moitas veces é preciso un tratamento dinámico da información e neste caso é preciso o emprego dunha linguaxe de scripting como javascript.

6.3. Javascript

Javascript é a linguaxe de programación interpretada que se executa nos navegadores cumprindo co estándar ECMAScript, permite executar ordes podendo interactuar coa páxina web a través da arbore DOM (Document Object Model) e co servidor mediante o sistema de chamadas asíncronas. Cada script en javascript está asociado a unha páxina web, de forma que todo o que executemos terá efecto sobre a páxina actual e nunca poderemos prolongar a execución dun código .js mais aló da vida desta páxina.

A pesar de executarse nun navegador javascript é unha linguaxe cunha potencia considerable, esta potencia ven da súa orientación a obxectos que tamén permite programación imperativa con un tipado débil e dinámico.

Javascript pode incluírse directamente no documento HTML, mais isto é problemático porque mestura dúas linguaxes e non permite a reutilización do código javascript en distintas páxinas, polo que na practica sempre se traballará con código contido en ficheiros .js que logo serán importados dende as páxinas web que o precisen. Mediante referencias tamén se pode engadir código de librerías, que pode achegarse de dous xeitos distintos, ou ben cun enlace á páxina onde están publicadas, ou ben descargando estas librerías ao servidor da aplicación e ofrecéndoas dende aí. O enfoque seguido neste proxecto é o segundo, pois de este xeito a aplicación está auto-contida, podendo traballar sen conexión con internet, e mantendo sempre a integridade a pesar de que o proveedor da biblioteca decida deixar de ofrecela.

Mediante Javascript tamén pode manexarse o elemento `< canvas >` de HTML5, que xera un mapa de bits para construír gráficos, manipular imaxes e crear dinamicamente animacións nunha páxina web. A única dúbida que soe xurdir sobre esta tecnoloxía está en canto ao seu rendemento e alcance, pero exemplos como os que amosa Kevin Roast na súa páxina web[10] despexan toda dúbida posíbel.

Por desgraza, a execución en navegador ten tamén os seus inconvenientes como a exe-

cución multi-threading empregando **Web Workers**, estes elementos pensados para permitir a execución paralela de código javascript deixan polo de agora moito que desexar xa que cada thread ten as súas propias variables estancas, impedindo polo tanto o acceso á arbore DOM dende threads paralelos co problema engadido de que algúns navegadores como Opera en vez de facer un multi-threading sobre threads do propio sistema tan só simulan este fenómeno nun único thread. O paralelismo cobrará especial importancia cando executemos os algoritmos que mostran os datos da análise en XML, pois ao executarse todo no mesmo fio debemos prestar especial atención a non bloquear a interface de usuario con tarefas moi prolongadas.

Javascript tamén é especialmente potente e sinxelo á hora de parsear documentos, xa sexa un documento HTML para acceder á arbore DOM ou ben un XML como o que xera o sistema de análise. Por desgraza, o manexo de excepcións, e as veces a selección de elementos dentro de un documento poden ser tarefas tediosas con javascript, co fin de simplificar isto empregaremos a biblioteca jQuery.

6.3.1. jQuery

jQuery é unha biblioteca javascript pensada para simplificar os aspectos mais complexos desta linguaxe como poden ser a manipulación de documentos HTML/XML ou as chamadas asíncronas mediante AJAX. Está escrita en javascript e é 100 % libre, tal vez por isto sexa a biblioteca javascript mais empregada.

O sistema de selección de jQuery é o mesmo que se emprega en CSS, permitindo seleccionar de xeito sinxelo un conxunto de elementos dentro dun documento, a maiores tamén dispón de un sinxelo acceso e modificación tantos destes elementos como dos seus atributos.

Os mesmos autores de jQuery, en vista de que no mundo javascript existían outros moitos aspectos a simplificar decidiron outras bibliotecas como as que se amosan a continuación:

6.3.2. Qunit

QUnit é un framework de probas unitarias para código javascript doado de empregar e bastante poderoso. Empregase tanto en jQuery, jQuery UI e os proxectos de jQuery Mobile sendo capaz de probar calquera código javascript xenérico.

Para facer probas emprega un conxunto de sentencias assert como todas as bibliotecas que realizan tests de unidade. No caso da nosa aplicación empregárase para probar todo aquel código independente da arbore DOM da páxina. É importante destacar que co fin de maximizar a facilidade de proba do código na capa javascript seguíronse os consellos amosados na páxina de Qunit[11] e unha interpretación flexible do MVC(Modelo Vista Controlador) onde a vista está conformada polo código HTML+CSS, o controlador polo manexadores dos ficheiros video-player.js, video-controls.js e por último as clases do modelo en javascript nos ficheiros Detection.js, DetectionObserver.js, Video-Detections.js e en menor medida suspicious-popup.js.

6.3.3. Image-picker

Outra librería tamén baseada en jQuery é Image-Picker [12] que permite xerar un formulario cun campo de tipo “select” baseado en imaxes en vez de nunha entrada despregable, todo elo de forma extremadamente sinxela.

6.3.4. jQuery UI

jQuery UI é unha biblioteca de compoñentes para jQuery que lle engade un conxunto de plugins, widgets e efectos visuais. Pódese descargar dende a súa páxina oficial o nucleo da biblioteca e os compoñentes nos que se estea interesado, que no caso deste proxecto empregase o compoñente slider[13] que xera unha barra selectora.

6.3.5. tablesorter

A derradeira biblioteca empregada, tamén baseada en jQuery é tablesorter. Tablesorter é un plugin baseado en jQuery para transformar unha táboa HTML estándar coas etiquetas `<thead>` e `<tbody>` nunha táboa que se pode ordenar polo contido das distintas columnas sen ter que recarga-la páxina. No noso caso será de moita utilidade

á hora de amosar os resultados da análise, pois así poderanse ordenar as deteccións segundo aparecen no vídeo, segundo o tempo que pasan en pantalla...

Todas estas tecnoloxías de capa web axudarannos a mostrar con mais facilidade a análise que o sistema de recoñecemento faga do vídeo, e en canto as ferramentas empregadas para esta análise a ferramenta fundamental que se empregará é OpenCV.

6.4. OpenCV

OpenCV é unha biblioteca libre de visión artificial escrita en código C/C++ optimizado. Dende a súa aparición publicada por Intel en Xaneiro de 1999, empregouse en infinidade de proxectos, tanto para detección de movemento como para aplicativos de control de procesos que requiren recoñecemento de obxectos.

OpenCV é multiplataforma, existindo versión para GNU/Linux, Mac OS X e Windows. Contén mais de 500 funcións que abarcan unha ampla gama de áreas como o proceso de visión, recoñecemento de obxectos (tamén recoñecemento facial), calibrado de cámaras, realidade aumentada e visión robótica.

Por todas estas características OpenCV é unha das bibliotecas mais empregadas hoxe en día, e é por elo tamén que se escolleu para a implementación do sistema de recoñecemento que a aplicación web empregará para a análise do vídeo.

Non obstante, xa que se desexa que a aplicación web sexa o mais versátil posible, contemplase a posibilidade de que poida empregar para a análise sistemas desenvolto noutras tecnoloxías como pode ser Matlab. E para dotala deste grao de versatilidade decídese definir unha interface de liña de comando a través da cal se chamará ao sistema, e un formato de ficheiro XML (Extensible Markup Language) no que este sistema de recoñecemento deben escribir os datos da súa análise.

6.5. Extensible Markup Language (XML)

XML é unha linguaxe de marcas desenvolvida polo W3C (World Wide Web Consortium) e empregado para almacenar datos de forma clara e lexible. Permite definir a gramática de linguaxes específicas para estruturar así grandes documentos.

Os documentos XML seguen una estrutura xerárquica baseada en etiquetas(tag's) e atributos, que se poden definir nunha Definición de Tipo de Documento ou DTD. [14]

Cando un documento en formato XML segue as directrices definidas no ficheiro DTD asociado, dise que este ficheiro esta ben formado(well formed en ingles), e para validar iso empregárase na elaboración do traballo algún avaliador de XML en liña como por exemplo o da W3Chools.[15]

XML é especialmente útil para comunicar varias aplicacións que traballan en tecnoloxías diferentes grazas á súa simplicidade que permite integrar os datos de xeito moi sinxelo. Precisamente por iso, servirá como nexa de unión entre o sistema de análise e a aplicación web.

Para a modificación da súa aparencia pódese empregar *CSS3* (Follas de Estilos en Cascada), e para modificar o seu comportamento, o elemento `< video >` de HTML5 achega unha serie de Métodos, Eventos e Propiedades[9] que poden empregarse dende código *Javascript*.

6.6. ffmpeg

Capítulo 7

Funcionalidades Destacadas

7.1. Control de Usuarios

É preciso un mínimo control de usuarios para controlar que sube vídeos á plataforma.

7.2. Carga de Vídeo

Dado que a aplicación traballará con vídeos subidos polos usuarios, o primeiro paso é lograr a subida exitosa de vídeos á plataforma. Para este fin empregarase un formulario HTML que viaxa sobre unha chamada POST de HTTP. Cando o navegador faga esta chamada incluíndo o vídeo como parte do formulario, este vídeo comezará a subirse ao servidor en pequenos anaquiños (data chunk).

É de especial importancia que no caso de que o vídeo teña un peso considerable e precise duns cantos segundos para subirse á plataforma, o usuario poida coñecer de forma gráfica o avance deste proceso.

Con este fin, crease un sistema de notificación de progreso baseado no django-progressbarupload [16], este sistema apoia-se nunha compoñente fundamental chamada VideoUploadHandler, que é unha extensión da interface de Django TemporaryFileUploadHandler [17], e que basicamente manexa a subida dun ficheiro de tamaño considerable.

Esta compoñente componse dunha función de inicio (`handle_raw_input`) que crea unha entrada na Cache de Django, almacenando como chave un número aleatorio e a IP do cliente que está a subir o vídeo, e como valor o tamaño do ficheiro e o porcentaxe de este que xa está subido. Esta entrada será actualizada cada vez que o servidor reciba un novo anaco de vídeo (mediante a outra función do compoñente `VideoUploadHandler`, `receive_data_chunk`).

Por outra parte, para que visualmente o cliente poida ver o avance da subida a través unha barra de progreso, crease unha función asíncrona en javascript (Tecnoloxía AJAX), que periodicamente consulta ao servidor para obter o valor da cache que indica a porcentaxe de subida do vídeo, e unha vez obtido, actualiza a barra de progreso para mostralo. Todo isto ten lugar no navegador mentres este inda está a subir o arquivo de vídeo.

Unha vez que a subida se completa, o POST é manexado pola vista `UploadView`, que se todos os datos do formulario son correctos, encargase de crear un modelo `VideoModel`. Como parte desta creación o vídeo pasa do directorio temporal no que foi almacenado (baixo linux por defecto é `/tmp`) a un directorio calculado pola función `get_valid_filename`. Esta función pásaselle ao modelo como parte do seu campo "video" do tipo `FileField`.

Unha vez subido o vídeo satisfactoriamente xa está asentada a base da nosa aplicación. Con esta parte completada a seguinte funcionalidade a atender será a de mostrar todos os vídeos dispoñibles na plataforma.

7.3. Lista de vídeos e Imaxe de Portada

Co fin de que os usuarios accedan aos distintos vídeos subidos, deseñase unha páxina web que será a principal do módulo `video_manager` na cal un usuario pode visualizar unha **lista paxinada dos vídeos** dispoñibles. Esta lista estará ordenada comezando polo vídeo mais recente e rematando polo mais antigo, tamén se contempla a posibilidade de poder filtralos por exemplo polo nome.

Para a elaboración de esta páxina empreganse o elementos dos que dispón Django

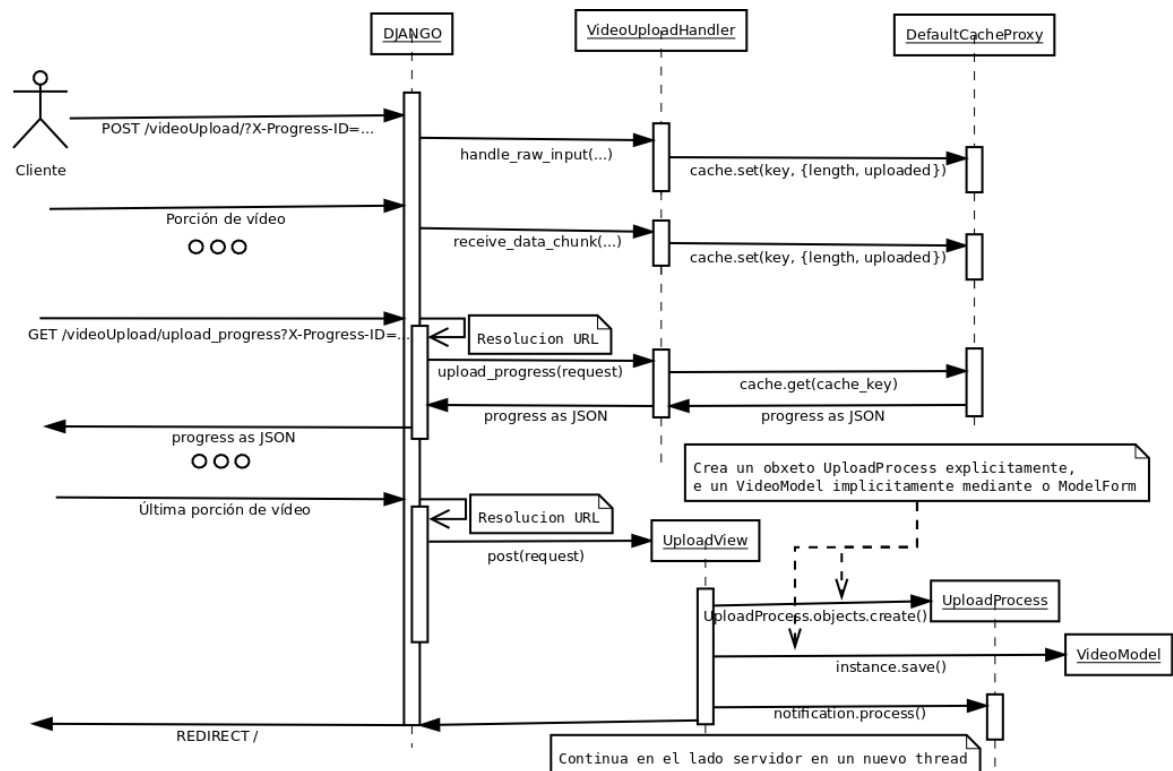


Figura 7.1: Diagrama de secuencia do proceso seguido cando se fai o Submit do Formulario de creación de vídeos

como son a clase `ListView` co seu atributo `paginate_by` e o filtrado de resultado co método `QuerySet.filter(...)`, mentres que para o paso das palabras claves polas que buscar un vídeo empregase un parámetro de URL chamado `'name'`.

Outra funcionalidade interesante de cara a amosar unha lista de vídeos é a de poder mostrar unha imaxe representativa de cada un deles. Con este fin crease a vista `SuccessfulUpload`, á que se redirecciona unha vez o vídeo é subido correctamente para seleccionar a súa **imaxe de portada**.



Figura 7.2: Captura de pantalla da páxina web `SuccessfulUpload`

Con este fin extraíense mediante `ffmpeg` unha serie de fotogramas do vídeo, que se gardan nun directorio temporal para que unha vez o vídeo estea subido e analizado, as imaxes se integren como parte dun formulario na páxina `SuccessfulUpload`. Mediante este formulario, xerado co plugin `image-picker`[12], o usuario poderá escoller o fotograma que lle pareza mais representativo do vídeo e unha vez que pulse no botón de "Submit" este fotograma gardarase como parte do Modelo de Django `VideoModel`, quedando pois accesible para que a lista de vídeos poida amosalo.

7.4. Reprodución de Vídeo

Desexase que a aplicación permita a reprodución dos vídeos contidos, mediante técnicas de streaming ou pseudo-streaming. Neste caso empregarase o pseudo-streaming polo sinxela que resulta esta implementación empregando as capacidades da etiqueta `< video >` de HTML5 en conxunto con un servidor HTTP como Apache ou o servidor para desenvolvemento de Django.

```

▶ <div class="col-sm-4 col-md-3">...</div>
▼ <div class="col-sm-8 col-md-6">
  ▼ <video id="video-player" controls>
    <source src="/media/videos/2/v_7760.mp4" type="video/webm" fps="10">
    <source src="/media/videos/2/v_7760.webm" type="video/webm" fps="10">
    "Your browser does not support the video tag."
  </video>
  <canvas id="training-canvas" class="drawing-layer" width="462" height="347" style="top: 7729px; left:
  1008.5px; padding-left: 0px; padding-top: 0px; display: none;">
  <canvas id="objects-canvas" class="drawing-layer" width="462" height="347" style="top: 7729px; left:
  1008.5px; padding-left: 0px; padding-top: 0px; display: none;">
  <canvas id="trajectories-canvas" class="drawing-layer" width="462" height="347" style="top: 7729px; left:
  1008.5px; padding-left: 0px; padding-top: 0px; display: none;">
  <span id="xml_detected_objs" hidden> /media/xml/2/x199.xml </span>
  </div>
▶ <div class="col-sm-12 col-md-3">...</div>

```

Figura 7.3: tag en html5, coas súas fontes e coas capas `< canvas >` asociadas

na figura 7.3 podemos ver o resultado en HTML5, vense claramente a etiqueta `< video >` coas súas fontes de datos `< source >`, cabe destacar que aquí engadiuse o atributo `fps` (Fotogramas Por Segundo do inglés Frames per second) que non pertence ao estándar definido pola W3C?? mais no caso da nosa aplicación é fundamental para poder coñecer a velocidade á que o navegador vai amosar os fotogramas do vídeo.

Outra cuestión a aclarar é o motivo polo cal non se subministra a fonte de vídeo en formato .ogv que a W3C recomenda. A resposta é que o codec theora que ffmpeg inclúe soporta decodificación pero NON codificación, polo cal é posible pasar de vídeos en .ogv a outros formatos pero non de outros formatos a .ogv imposibilitando pois que se poida ofrecer o vídeo neste formato mentres o codec de ffmpeg non o permita. Non obstante, isto non supón un problema, xa que como se pode ver na táboa seguinte todos os navegadores permiten a reprodución baseándose nestes dous formatos:

Táboa 7.1: My caption

Internet Explorer	SI	NON	NON
Chrome	SI	SI	SI
Firefox	SI dende Firefox 21 (win) dende Firefox 30 (linux)	SI	SI
Safari	SI	NON	NON
Opera	SI dende Opera 25	SI	SI

7.5. Análise de Vídeo

Para a análise de vídeo será preciso definir unha interface de liña de comandos mediante a cal a aplicación web chamará ao Sistema de Recoñecemento, indicándolle aqueles parámetros que sexan precisos^{7.4}.

7.5.1. Interface de Liña de Comandos

A aplicación web indicarlle a este sistema o vídeo que debe empregar como entrada para o recoñecemento e o ficheiro de saída onde ten que escribir os datos da análise. A maiores, pódesele indicar con que frecuencia se desexa que o Subsistema Behaviour System, encargado da análise de alto nivel, entre en funcionamento. Po último a opción `-standar` fai que se mostre o resultado da detección de obxectos por liña de comandos, esta saída é empregada pola capa web para calcular o progreso do proceso de análise.

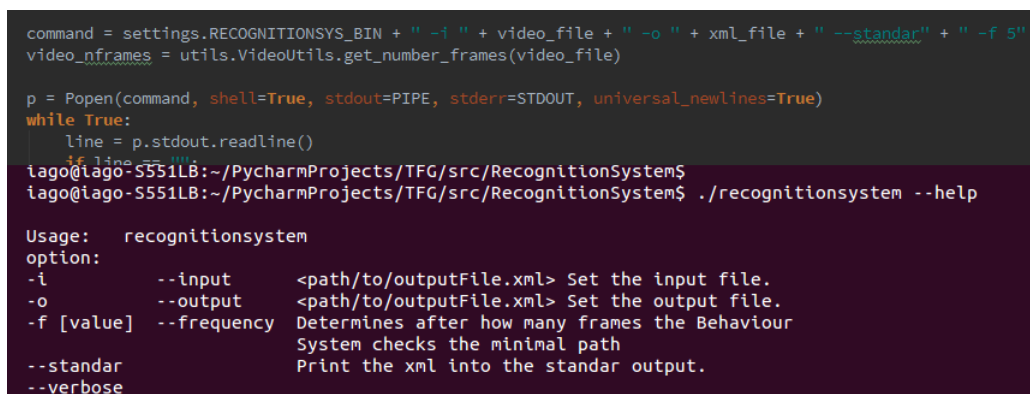
Estas opcións tamén se poden visualizar mediante o comando `-help`:

```
iago@UbuIago:~/TFG/src/RecognitionSystem$ ./recognitionsystem --help
Usage: recognitionsystem
option:
-i          --input      <path/to/outputFile.xml> Set the input file.
```

```

-o          --output      <path/to/outputFile.xml> Set the output file.
-f [value]  --frequency   Determines after how many frames the Behaviour
                        System checks the minimal path
--standar          Print the xml into the standar output.
--verbose

```



```

command = settings.RECOGNITIONSYS_BIN + " -i " + video_file + " -o " + xml_file + " --standar" + " -f 5"
video_nframes = utils.VideoUtils.get_number_frames(video_file)

p = Popen(command, shell=True, stdout=PIPE, stderr=STDOUT, universal_newlines=True)
while True:
    line = p.stdout.readline()
    if line == "":
        break
iago@iago-S551LB:~/PycharmProjects/TFG/src/RecognitionSystem$
iago@iago-S551LB:~/PycharmProjects/TFG/src/RecognitionSystem$ ./recognitionssystem --help

Usage:  recognitionssystem
option:
-i          --input      <path/to/outputFile.xml> Set the input file.
-o          --output      <path/to/outputFile.xml> Set the output file.
-f [value]  --frequency   Determines after how many frames the Behaviour
                        System checks the minimal path
--standar          Print the xml into the standar output.
--verbose

```

Figura 7.4: Interface de linha de comandos do Sistema de reconhecimento

7.5.2. Ficheiro XML

Como resultado desta chamada, o sistema de reconhecimento deve criar no ficheiro indicado para a saída, um XML co formato que se define no esquema .dtd situado no directorio

`src/static/detections_schema.dtd`

Neste ficheiro podemos ver a definición dos seguintes elementos:

- *< objects >*

Contén para cada un dos fotogramas *< frame >* a lista de obxectos detectados segundo o explicado no apartado 2.2, indicando para cada un deles a distancia á parte esquerda, e superior da escena (xc, yc) e o alto e ancho do obxecto detectado (h, w).

- *< trajectories >*

Este outro elemento garda para cada un dos obxectos detectados a traxectoria que seguiu ao longo do vídeo, esta traxectoria estará composta de unha serie de puntos *< point >*, para cada un dos cales, a parte das súas coordenadas e o número de frame, indicase un grao de anormalidade entre 1 e 0 que indica como de anómala é a conduta dese obxecto no momento no que se atopa sobre ese punto.

Para xerar este ficheiro XML foi preciso desenvolver unha serie de funcionalidades en C++, que están contidas no paquete XmlRecognition.

7.5.3. Paquete XmlRecognition

Inicialmente o proxecto conta cun código escrito en C++ e apoiado en OpenCV que está distribuído en dúas librerías EllipseLib e BehaviorLib. A primeira delas encargada da detección de obxectos, e a segunda encargada de analizo comportamento a alto nivel a partir dos resultados que proporciona a primeira.

Para a construción do Sistema de recoñecemento integráranse estas dúas librerías como módulos, e creárase un terceiro módulo C++ chamado XmlRecognition. Este novo módulo será o responsable de definir e implementar a interface de liña de comandos, xestionar a comunicación entre as dúas librerías e gardar o resultado da análise en formato XML.

Para elo, creáse un ficheiro principal chamado **main.cpp** que describe a interface de liña de comandos independente da librería que se emprega para as deteccións, un ficheiro **XmlUtils** que contén as funcionalidades para a escritura do XML en base a unha detección simplificada: DetectionDto (Data Transfer Object), e por último un ficheiro **RecognitionFacade** que variará en caso de cambiar as librerías, transformando o resultado destas en DetectionDto's para logo poder escribilo coas funcionalidades de XmlUtils.

Para maximizar o rendemento evitase crear clases innecesarias: para o caso da

DetectionDto é dabondo cunha estrutura, e no caso de XmlUtils y RecognitionFacade, ao non ter estado chega con ficheiros que definen funcións.

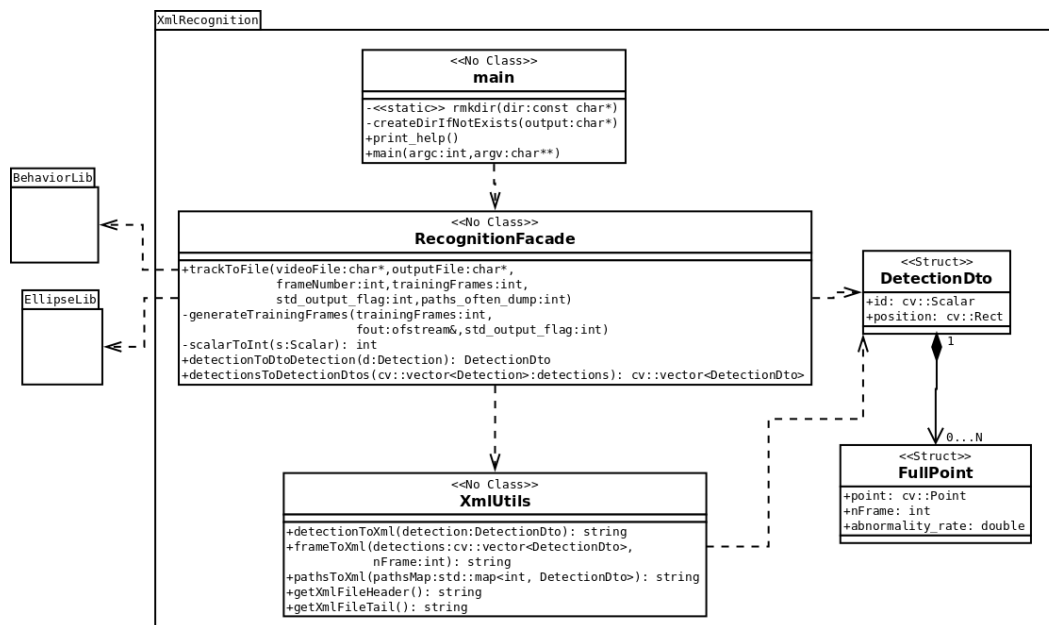


Figura 7.5: Diagrama de clases do paquete XmlRecognition

Coa elaboración deste paquete queda listo o sistema de recoñecemento, agora solo resta que a capa web sexa capaz de chamar ao sistema e mostrar as análises en XML sobre o vídeo. Será o que abordaremos no seguinte apartado.

7.5.4. A análise dende a capa web

Cando deseñamos unha aplicación web é de capital importancia que o usuario este informado de que está a acontecer na aplicación para que non sinta que está perdido, ou que a aplicación non responde. Tendo isto en conta, e sabendo que tanto o proceso de análise coma o de conversión do vídeo a outros formatos poden requirir dun tempo prolongado, prantexase un problema: como manter ao usuario informado destes longos procesos e evitar a sensación de bloqueo?

A solución deseñada é un sistema de notificacións que permite ao usuario rexistrado navegar libremente pola aplicación mentres o vídeo se está a analizar, mostrando en

todo momento unha barra de progreso para o proceso que se está a seguir nestes intres.

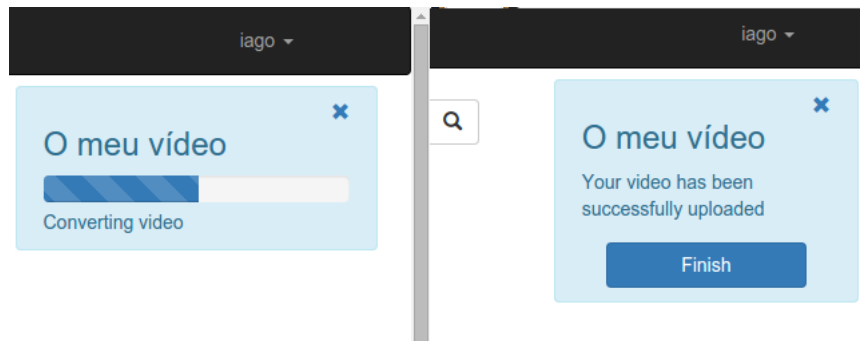


Figura 7.6: Capturas de pantalla do sistema de Notificacións

Para albergar tanto o sistema de notificacións como os procesos que engloba, creouse o módulo de Django `video_upload` composto polas clases que se poden observar no seguinte diagrama:

`UploadProcess` representa o proceso de subida, análise, conversión e extracción de imaxes a partir de un vídeo. Mentres que `AnalysisProcess` representa o proceso que se segue no caso de que un vídeo xa subido á plataforma sexa analizado de novo. Os distintos estados nos que pode estar un proceso modelanse mediante a clase abstracta `ProcessState`, que nas súas implementacións define tanto o traballo a realizar neste estado coma a mensaxe que se amosará ao usuario cando este se execute.

Dado que é o `ProcessState` que executará a tarefa, tamén será o encargado de actualizar a través do método `set_progress(self, progress)` o progreso do proceso asociado (`UploadProcess` ou `AnalysisProcess`).

É importante destacar que as figuras etiquetadas co estereotipo `<< Model >>` son modelos de BD manexados por Django. Nótese tamén que pese a que `UploadProcess` e `AnalysisProcess` comparten a meirande parte do seu código, non foron refactorizados nunha clase abstracta, dadas as complicacións de base de datos que isto carrega. En lugar diso empregase o tipado dinámico de Python para pasar os obxectos tanto de `UploadProcess` como de `AnalysisProcess` á clase `ProcessState`, que ao invocar só os

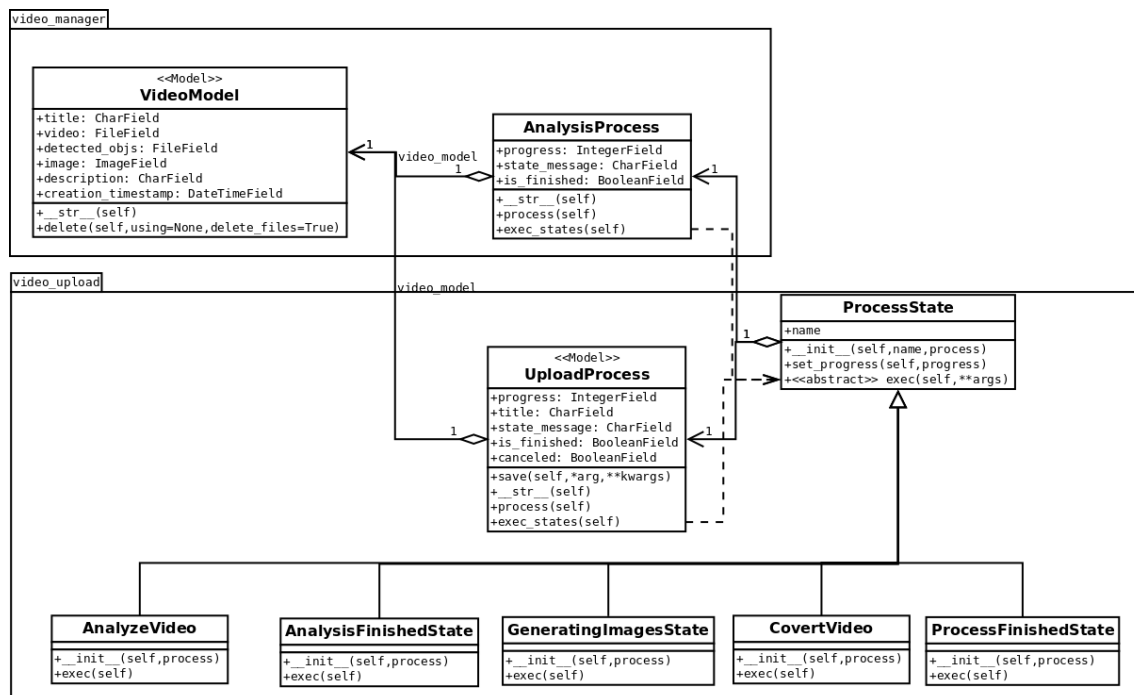


Figura 7.7: Diagrama de Clases do sistema

métodos comúns non é capaz de percibir a diferenza entre ambas.

O seu funcionamento mostrase no gráfico:

7.6. Mostrar Deteccións

Unha vez que o vídeo xa está subido e correctamente analizado, o que resta é comezar a construír na capa web as vistas que mostren sobre o elemento `< video >` de HTML5 as distintas capas de análise obtidas a partires do ficheiro XML, así como unha serie de paneis que permitan configurar estas vistas.

Todo este traballo realizase na vista `DetailView` do módulo `video_manager` e principalmente consiste nun ficheiro HTML que contén as referencias a:

- O vídeo a mostrar
- O ficheiro XML que contén a análise realizada polo sistema de recoñecemento.
- Os ficheiros de estilo.

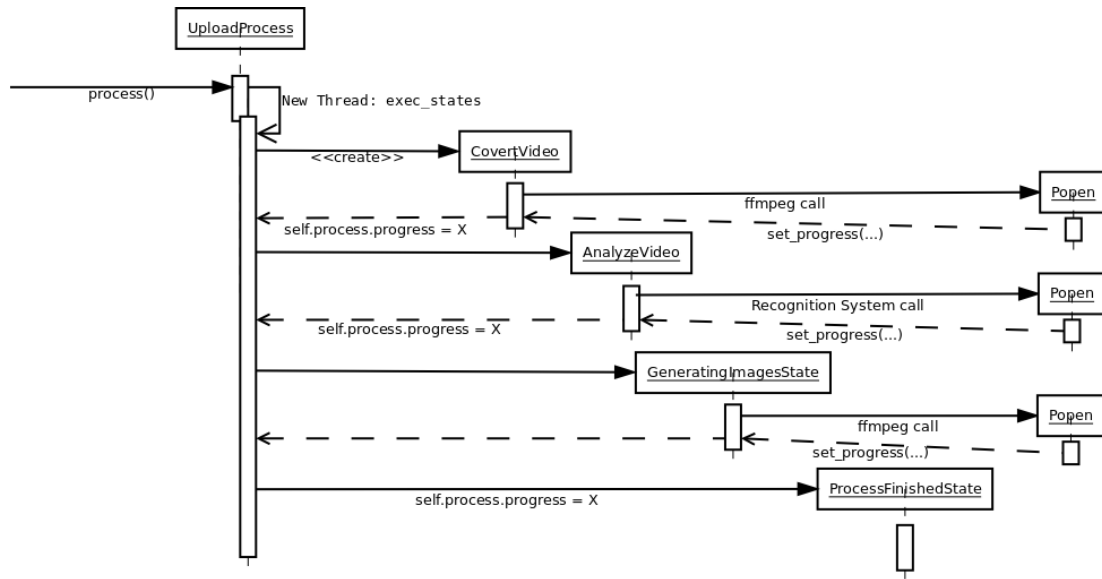


Figura 7.8: Diagrama de secuencia da análise do vídeo na capa web

- Os distintos ficheiros de código javascript involucrados.

Para amosar a análise do ficheiro XML sobre o vídeo a estratexia será a de a de superpoñer distintos elementos `< canvas >` sobre o elemento vídeo `< video >` como se pode ver na figura 7.3. Para axustar todas estas capas etiquetadas como `class="drawing-layer"` sobre o vídeo crease a función `adjustCanvasExtended` do ficheiro `video-player.js`, esta función chamase cando o vídeo se carga na páxina, en caso de que o tamaño do vídeo cambie ou cando se entra e sae do modo pantalla completa, axustando de novo o tamaño de todas estas capas ao actual tamaño do vídeo.

Unha vez axustados os elementos `< canvas >` nos que se desexa amosar a información, tense que extraer esta do ficheiro XML. O XML cargase mediante AJAX, cunha chamada de jQuery `$.get(...)` ao dirección do servidor que indica a etiqueta oculta con `id xml_detected_objs`, unha vez cargado este arquivo iniciase a carga inicial do sistema.

Esta carga inicial consiste na creación dun obxecto `VideoDetections` creado a partir do XML, e que conterà a lista de deteccións así como unha referencia ao elemento `< video >`. Este elemento `VideoDetections`, é o suxeito de un patrón Observador no cal un suxeito ou obxecto central notifica ao seus observadores (Observers) os cambios

no seu estado. Neste caso, os observadores serán os encargados de actualizar cada un dos elementos da páxina, incluídas as capas *< canvas >*, dos cambios no suxeito VideoDetections, estes cambios poden ser por exemplo o avance na reprodución do vídeo, un cambio de preferencias no panel de control... Deste xeito unifícase a xestión das deteccións que só se manexa no elemento VideoDetections e faise moito mais sinxelo engadir ou eliminar algunha capa de información xa que basta con engadir ou eliminar o observador que a manexa.

Nótese que a maiores do propio patrón observador, tamén se engaden os métodos enable e disable á clase DetectionsObserver, isto faise para que no caso de que algún observador non se atope activo nun momento determinado non reciba as notificacións de actualización. O motivo deste cambio é o incremento de eficiencia que produce, moi importante dado que o proceso de actualización do Suxeito e todo-los seus observadores tense que executar entre 10 e 30 veces por segundo.

A continuación pódese ver un diagrama de clases que amosa o deseño desta capa:

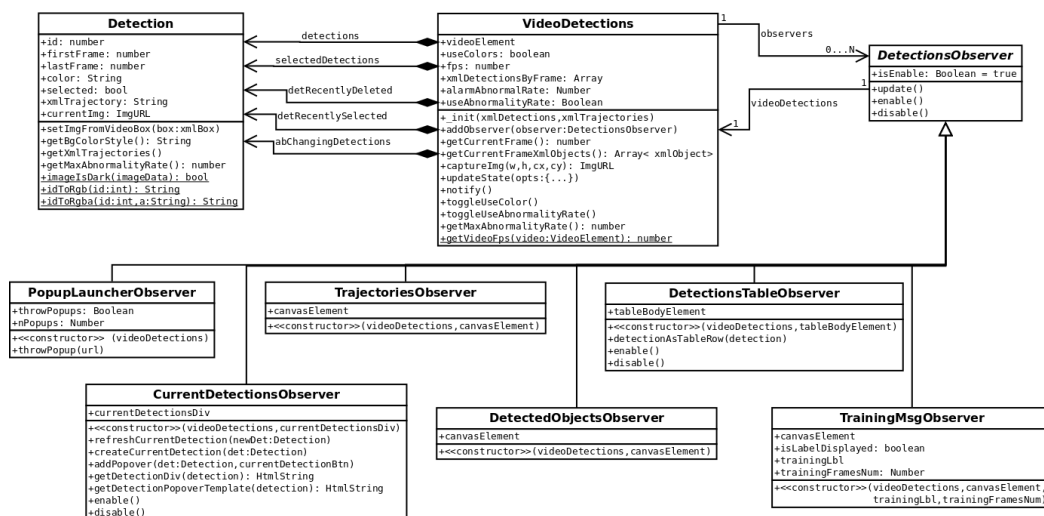


Figura 7.9: Diagrama de clases do patrón observador na capa web

Agora que se coñece o xeito no que as deteccións pasan de un formato xml a un modelo obxectual no cal poden ser consultadas con maior eficiencia veremos como actualizar o estado dos elementos *< canvas >* cada vez que se mostra un novo fotograma ou se modifica algunha opción dos paneis da páxina.

Nun comezo, pensouse en asociar a actualización de estado ao evento `timeupdate` do elemento `< video >`, que segundo a súa definición lanzase cando a posición do vídeo varía. Por desgraza, e como reflexa a reflexión que podemos atopar no libro ... COMPLETAR CON REFERENCIA AO LIBRO DE VIDEO EN HTML5 ... este evento é lento de mais para o seu emprego, polo que o que se fará será crear un bucle que se execute cando o vídeo se estea a reproducir. Para tentar que o bucle se execute unha soa vez en cada un dos fotogramas empregase a función `setTimeout` que fará que o código agarde unha cantidade de tempo antes de volver a executarse. Esta cantidade de tempo calculase en base a execucións anteriores e ao número de fotogramas por segundo desa fonte de vídeo, que o servidor inclúe no HTML mediante o atributo `fps` tras obtelos datos dunha chamada ao método `VideoUtils.get_fps` que á súa vez chama ao `ffmpeg`.

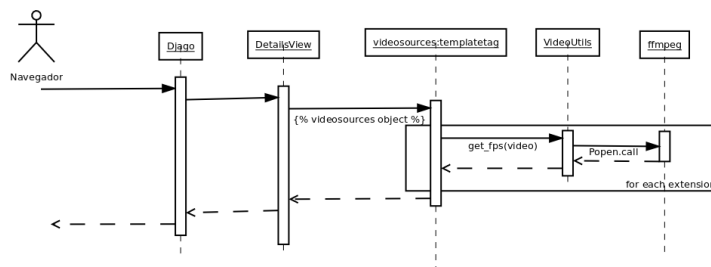


Figura 7.10: Xeración do atributo `fps` no lado servidor

Coñecido pois o xeito no que refresca o estado da interface de usuario e o sistema que crea un modelo obxectual para a representación gráfica das deteccións, agora com-pre ver como é que se remarcan os obxectos detectados no vídeo. Isto faise mediante o elemento `< canvas >` con `id="objects-canvas"` no que o observador da clase `DetectedObjectsObserver` encargase de debuxar un recadro da cor axeitada. Para coñecer esta cor que depende das opcións do panel o observador consulta a cada obxecto `Detection` empregando o método `getCurrentColor()`.

Tamén cabe destacar aquí a función do observador `TrainingMsgObserver`, que é o encargado de mostrar a etiqueta de Fotogramas de Adestramento e o recadro de cor amarelo nos primeiros fotogramas do vídeo que corresponden aos empregados polo

sistema de recoñecemento para obter o fondo da imaxe e así poder distinguir os obxectos que se moven sobre el.

7.7. Traxectorias

As traxectorias son outra parte fundamental do noso sistema, debese mostrar a ruta seguida por cada un dos obxectos detectados dende o momento no que entra en escena ata a súa saída. Para seres detectadas no sistema de recoñecemento hai que invocar a biblioteca de análise de alto nivel que require de unha cantidade de tempo considerable, é por elo que se precisa un argumento de frecuencia na interface de liña de comando que indica cada cantos fotogramas se realizará a análise de alto nivel (por defecto cada 5 fotogramas).

As traxectorias chegan ao sistema web como parte do XML resultado do sistema de análise, para mostralas empregárase unha nova capa canvas en conxunto cun novo observador `TrajectoriesObserver` que itera sobre as deteccións actuais pintando as liñas entre cada un dos puntos da súa ruta, dende o comezo desta ate o punto actual.

7.8. Lista de Deteccións

7.9. Comportamento anormal

Capítulo 8

Probas Realizadas

Á hora de deseñar probas é importante abarcar a maior parte do código posible, neste proxecto isto foi todo un reto, pois o alto nivel de integración dificulta enormemente a realización das probas. Pese a todo, logrouse probar tanto o código realizado en Python-Django así como o código da capa cliente en javascript, empregando para elo distintos modelos e bibliotecas de probas que vemos a continuación.

8.1. Probas Unitarias

As probas unitarias proban as funcionalidades mais básica do software. Executanse sempre no ámbito de un só modulo para probar o correcto funcionamento de este, simulando se fose preciso a súa interacción con outros módulos (este proceso chamase *mocking*).

No ámbito da nosa aplicación as probas de cada módulo recollense nos ficheiros `tests.py` de cada un deles. Estas probas inclúen tanto comprobacións de funcións illadas, como a correcta xeración de algunha webs independentes do resto dos módulos.

8.1.1. Política de acceso ás páxinas web

Como resulta lóxico non todo o mundo pode acceder a tódalas páxinas da aplicación, algunhas delas están reservadas para o administrador, outras para o propio usuario logueado, unhas terceiras para calquera usuario logueado e por último hai páxinas de

dominio público.

É importante de cara a non cometer erros de seguridade, que este correcto comportamento sexa comprobado, así na táboa excel que se atopa no ficheiro docs/urlsMap.ods pódese observar con detalle que política de acceso segue cada unha das páxinas da aplicación segundo a súa URL. Foi a partir deste mapa de urls dende onde se elaborou a base dos test de unidade para o acceso ás páxinas, que comprobar para cada unha das páxinas da aplicación que un usuario cos permisos adecuados poida acceder e que calquera dos demais reciba o erro axeitado.

Pero só con probas unitarias non se pode asegurar a calidade dos software xa que moitos dos módulos están pensados para interactuar entre eles, polo que fanse necesarias as Probas de Integración.

8.1.2. Probas da Capa Web con Javascript

As probas da capa web en escrita en javascript apoiaranse no framework Qunit de jQuery que proporciona un xeito sinxelo de crear probas unitarias sempre e cando o código javascript esté convintemente separado do HTML que forma a vista da capa web.

Como resulta lóxico, estas probas estarán escritas en javascript e almacenadas no directorio do proxecto /src/static/site/tests, podendo executarse de dous xeitos diferentes: Ou ben como unha páxina web pertencente á aplicación (isto favorece o desenvolvemento áxil), ou ben como unha proba das realizadas polo comando `python manage.py test`.

Para poder executar un código javascript dende a execución común dos tests da aplicación, precisamos un lanzador ou runner que lance estes tests contra algún navegador de liña de comandos, neste caso a opción seleccionada foi a combinación do paquete `django.js` (v0.8.1) en combinación co navegador de liña de comandos `phantomJS`.

Django.js é un conxunto de utilidades que permiten a integración de código javascript en Django. Mais en concreto neste proxecto empréganse aquelas que teñen que ver co testing de aplicacións??, destacando as clases `QUnitSuite` e `PhantomJsRunner` que se empregan para lanzar os tests como parte dos test da aplicación mediante a clase

creada `StaticJsTestCase` que obtén os resultados do modo de páxina web co navegador de liña de comandos, e tamén a clase `QUnitView` que é a peza central para a execución a neste modo de páxina web.

Po outro lado `PhantomJS` é un navegador `WebKit` de liña de comandos, cunha API Javascript que da soporte rápido e nativo para varios estándares web que resultan moi do noso interese, como son a manipulación DOM, os selectores CSS, JSON, Canvas e SVG. `PhantomJS` será chamado implícitamente polo runner de `Django.js` cando se executen os test, mentres que no caso da vista `QUnitView` os tests executaranse directamente no navegador que realice a petición.

8.2. Probas de Integración

As probas de integración so aquelas que proban o funcionamento conxunto de varios módulos da aplicación, realízanse tras o éxito das probas unitarias tamén sen que haxa interacción humana.

No noso caso agruparemos as probas de integración nun paquete a parte, para evitar así mesturalas coas probas unitarias de cada módulo. A estes efectos creamos a clase `SeleniumAncowebTest` que estende `StaticLiveServerTestCase` engadindo ademais os métodos `login_user(self, user, password)` e `logout_user(self, user)` xa que todo-los tests que comprobren outros módulos precisando de un usuario logueado consideranse tests de integración.

8.2.1. Probas Funcionais Selenium

As probas funcionais son aquelas nas que se lle dita ao sistema cales serán as saídas a unha determinada serie de entradas co fin de comprobar que a funcionalidade é a correcta. No caso da nosa aplicación empregaremos tests funcionais para as probas de integración apoiandonos en Selenium.

Selenium é un framework para a realización de probas funcionais que permite lanzar un navegador e indicar as accións a realizar sobre él xunto cunha serie de comprobacións para verificar que estas accións provocan na páxina web o efecto desexado. Resultan de especial relevancia na programación web, xa que o seu funcionamento asemellase

moito ao que un humano faría para comprobar o correcto funcionamento da web sendo por tanto moi intuitivo.

No noso caso comprobábase con Selenium, o logueado de Usuarios, a subida de vídeos e o listado de vídeos.

8.3. Probas de Sistema

8.4. Probas de Aceptación

Por último están as probas de Aceptación que se fan co obxectivo de comprobar se o software cumpre coas expectativas que o cliente tiña de el. A estes efectos cada vez que se finalizaba unha funcionalidade realizouse acorde coa metodoloxía unha proba completa por parte do titor Brais Cancela que garante que todo o implementado era acorde cos que se desexaba. Ocasionalmente o proxecto tamén foi revisado polos outros titores aportandolle así un toque mais plural.

Capítulo 9

Calidade

Os parámetros de calidade empregados para a codificación do código fonte son: JavaScript Style Guide and Coding Conventions[18] JavaScript Best Practices:[19] PEP8 Style Guide for Python Code: [20]

Capítulo 10

Planificación e Avaliación de Custes

Capítulo 11

Resultados e Conclusións

Podemos concluir que ...

Capítulo 12

Liñas Futuras

Aquí meter todas as tarxetas que queden pendentes + paridas varias...

12.1. Vídeo en Directo

Escribir aquí todo lo documentado del vídeo en streaming

Apéndice A

Título

Bla bla bla

A.1. Lista de Acrónimos

Bla bla bla

- IDE
- BD

A.2. Manual de Usuario

A.3. Manual de referencias Técnicas

A.4. Notas acerca da Terminoloxía

- Scrum Team
- Product Owner
- Development Team
- Scrum Master
- Spring

- Sprint Planning Meeting
- Sprint Goal
- Daily Scrum
- Sprint Review
- Sprint Retrospective
- Product Backlog
- Sprint Backlog

Se pueden escribir unas líneas en las que se justifique el motivo de la inclusión de términos en inglés, por ejemplo por formar parte de la jerga, ser de uso frecuente y estar completamente aceptados por comités científicos de la temática del proyecto, etc.; en otro caso deben buscarse traducciones adecuadas.

Bibliografía

- [1] Barizo, B. C., *Understanding target trajectory behavior: a dynamic scene modeling approach*. PhD thesis, Universidade da Coruña, 2015.
- [2] Schwaber, K. and Sutherland, J., *La Guía Definitiva de Scrum: Las Reglas del Juego*. Scrum.Org and ScrumInc, 2013.
- [3] “Red5 web server on github.” Available on: <https://github.com/Red5/red5-server>.
- [4] “Web toolkit for c++ page.” Available on: <http://www.webtoolkit.eu/wt>.
- [5] “Icecast server on github.” Available on: <https://github.com/paluh/icecast>.
- [6] “Python subprocess module.” Available on: <https://docs.python.org/3.4/library/subprocess.html>.
- [7] “Django web page.” Available on: <https://www.djangoproject.com/>.
- [8] “W3schools html5 video tag web page.” Available on: http://www.w3schools.com/html/html5_video.asp.
- [9] “Html audio and video dom reference.” Available on: http://www.w3schools.com/tags/ref_av_dom.asp.
- [10] “Kevin roast canvas examples.” Available on: <http://www.kevs3d.co.uk/dev/index.html>.
- [11] “Recomendacións de jquery-qunit á hora de escribir código testeable.” Available on: <http://qunitjs.com/intro/#make-things-testable>.

-
- [12] “Páxina web da librería image picker empregada no proxecto.” Avaliable on: <http://rvera.github.io/image-picker>.
 - [13] “Compoñente slider de jquery ui.” Avaliable on: <http://api.jqueryui.com/slider/>.
 - [14] “Document type definition w3chools tutorial.” Avaliable on: http://www.w3schools.com/xml/xml_dtd_intro.asp.
 - [15] “Validador xml da w3chools.” Avaliable on: http://www.w3schools.com/xml/xml_validator.asp.
 - [16] “Proxecto django-progressbarupload publicado por ouhouhsami.” Avaliable on: <https://github.com/ouhouhsami/django-progressbarupload>.
 - [17] “Clase temporaryfileuploadhandler.” Avaliable on: https://docs.djangoproject.com/en/1.8/_modules/django/core/files/uploadhandler/#TemporaryFileUploadHandler.
 - [18] “Javascript style guide and coding conventions.” Avaliable on: http://www.w3schools.com/js/js_conventions.asp.
 - [19] “Javascript best practices.” Avaliable on: http://www.w3schools.com/js/js_best_practices.asp.
 - [20] “Pep8 style guide for python code.” Avaliable on: <https://www.python.org/dev/peps/pep-0008/>.