InVID

Deliverable D5.1: Technology Roadmap

Max Göbel, Arno Scharl, Albert Weichselbraun (webLyzard)
Jan Thomsen (Condat)
Gerald Innerwinkler (APA-IT)
Evlampios Apostolidis, Symeon Papadopoulos, Vasileios Mezaris (CERTH)
Lyndon Nixon (MOD)

30/06/2016

Work Package 5: System Integration and Platform Development

InVID - In Video Veritas: Verification of Social Media Video Content for the News Industry

Innovation Action

Horizon 2020, Research and Innovation Programme

Grant Agreement Number 687786

Dissemination level	PP
Contractual date of delivery	30/06/2016
Actual date of delivery	30/06/2016
Deliverable number	D5.1
Deliverable name	Technology Roadmap
File	`InVID_D5.1_v1.0.tex`
Nature	Report
Status & version	Final & V1.0
Number of pages	33
WP contributing to the deliverable	5
Task responsible	webLyzard
Other contributors	Condat, CERTH, APA-IT, MOD, UdL, EXO
Author(s)	Max Göbel, Arno Scharl, Albert Weichselbraun (webLyzard) Jan Thomsen (Condat) Gerald Innerwinkler (APA-IT) Evlampios Apostolidis, Symeon Papadopoulos, Vasileios Mezaris (CERTH) Lyndon Nixon (MOD)
Quality Assessors	Jan Thomsen / Condat
EC Project Officer	Miguel Montarelo Navajo
Keywords	System Architecture, Integration, API Specification

Abstract

This deliverable summarises the architecture and technology roadmap of the InVID project. It covers both high-level design decisions for the overall system as well as technical considerations for the individual information services to be deployed.

All data enrichment components of the InVID platform will be provided as RESTful Web services accessible to all components of the platform. InVID has three data producers that govern the control flow of data across these Web services. Two central, cross-referenced data repositories have been configured to hold all relevant InVID datasets required by the verification components, of both binary and contextual formats. A InVID meta-document model is presented to enable effective and flexible data sharing among all InVID partners. This document motivates the design decisions behind this setup and gives technical details about data management, data integration, data security, and service verification aspects.

Development of the InVID prototype will follow an evolutionary approach, building a working system early in the project life-cycle to ensure that technical quality and usability meet the highest standards. We plan to publicly announce a first version of the InVID platform in September 2016.

This deliverable is structured as follows: We begin with an overview of the InVID platform in Section 2, followed by more detailed technical descriptions of the data acquisition (Section 3) and knowledge extraction (Section 4) components. Section 5 presents the data integration strategy that was devised to guarantee a seamless integration of all technical contributions of the project partners into a coherent and effective video verification platform. Section 6 provides technical details of the information services and applications to be built. Finally, we summarize the technical infrastructure that has been put in place to enable efficient collaboration among the InVID partners in Section 7.

History of the document

Table 1: History of the document.

Date	Version	Name	Comment
2016/03/01	V0.1	A Scharl, webLyzard	Document structure and introduction
2016/06/09	V0.2	G Innerwinkler, APA-IT	Details about the UGV upload app
2016/06/11	V0.3	V Mezaris, CERTH	Details about CERTH components
2016/06/12	V0.4	M Göbel, webLyzard	Core platform description
2016/06/14	V0.5	J Thomsen, Condat	Description of integrated workflow
2016/06/17	V0.51	A Weichselbraun, webLyzard	Document revision
2016/06/20	V0.52	M Göbel, webLyzard	Integration and cleanup
2016/06/21	V0.6	V Mezaris, CERTH	Revision
2016/06/22	V0.7	J Thomsen, Condat	Quality assessment
2016/06/28	V0.8	M Göbel, webLyzard	Revision and consolidation
2016/06/30	V1.0	M Göbel, A Scharl, webLyzard	Final review and formatting

0 Content

1 Introduction

The purpose of the document is to provide an overview of the architecture, data and software integration plans, as well as to outline the roadmap towards a first InVID prototype.

InVID aims to develop a content verification platform to detect, authenticate and check the reliability and accuracy of newsworthy video files and video content spread via social media. As user applications, the platform will provide both a management system for user-generated content (UGC) as well as a visual analytics dashboard to track information flows and evolving stories, including participating actors (persons, organisations) and the relations among them, as outlined in Fig. 1. UGC content will be fed into the system in the form of videos via the so-called UGV-Upload App (user-generated video). To build such a comprehensive content verification platform, InVID will integrate a portfolio of content acquisition, metadata annotation, information retrieval, content verification and visualisation services - all powered by highly scalable and interconnected components through Application Programming Interfaces (APIs), as outlined in Section 5.

Data and software integration are key aspects to data-intensive projects such as InVID. Projects that do not pay close attention to devise pragmatic and clear integration plans are likely to fail to deliver sound common results, even if the individual components and tools developed within the project are of high quality. This document stipulates the integration procedures and plans, encouraging all to participate actively in the integration discussions. It provides the guiding principles to ensure that the data, components and tools provided by all partners interact in a manner that helps to achieve the project goals.

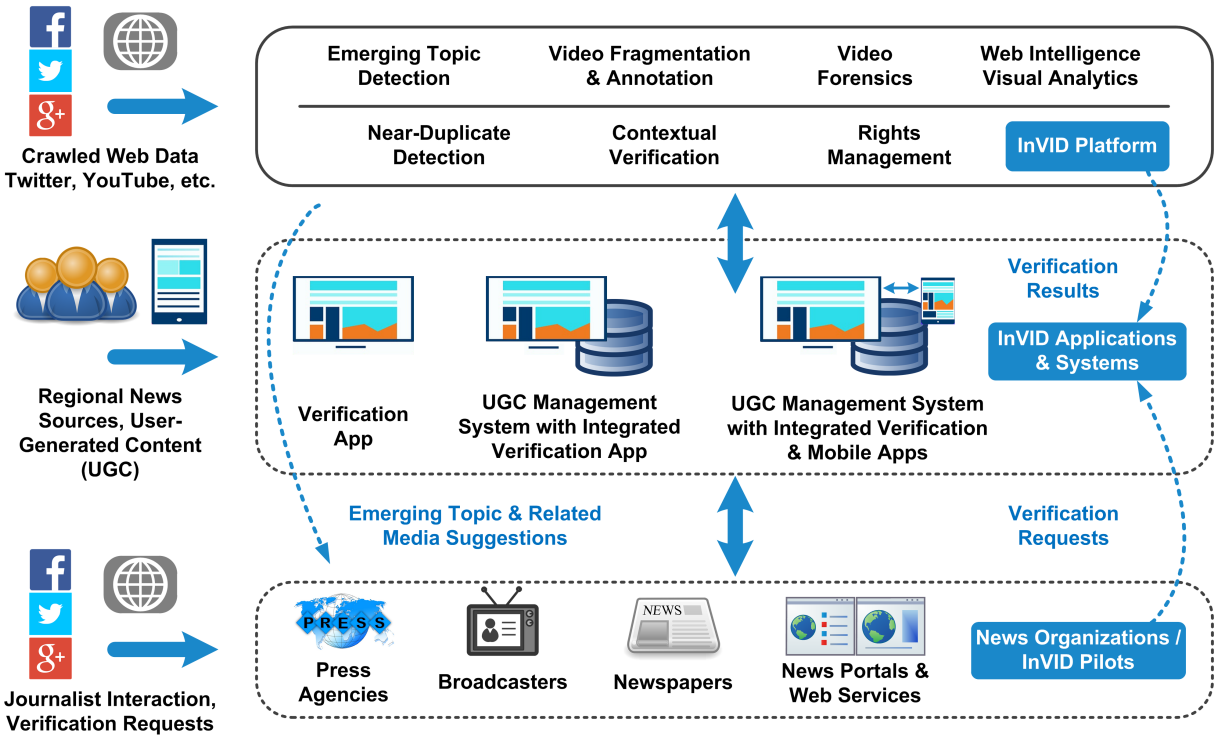

Figure 1: Functional Overview of the InVID Platform.

2 Overview of Workflow and System Architecture

Fig. 2 illustrates the InVID system architecture. A dynamic task scheduling component allocates resources to content acquisition and pre-processing tasks - i.e., focused crawls of Web sites and RSS feeds, real-time updates of social media input channels, or keyword analyses. The webLyzard platform (Scharl, Weichselbraun, Göbel, Rafelsberger, & Kamolov, 2016; Scharl, Hubmann-Haidvogel, Sabou, Weichselbraun, & Lang, 2013) uses the scalable and flexible messaging broker service Rab-

bitMQ (www.rabbitmq.com) to distribute these tasks to worker instances, which perform the content acquisition and processing tasks. Docker images enable the dynamic creation and destruction of worker instances based on the current system load and number of pending tasks. Although virtualisation ensures a scalable content acquisition and processing pipeline, additional steps are taken to manage the volume and dynamic nature of InVID content. Efficient pre-processing and filtering removes irrelevant content at an early stage of the processing pipeline. This reduces the number of documents that later need to be processed by the computationally more expensive knowledge extraction algorithms. Locality sensitive hashing to identify near duplicates and a domain specificity measure allow removing irrelevant content at an early stage. The domain specificity measure uses a combination of blacklists and whitelists to assess the relevance of gathered documents in terms of newsworthiness.

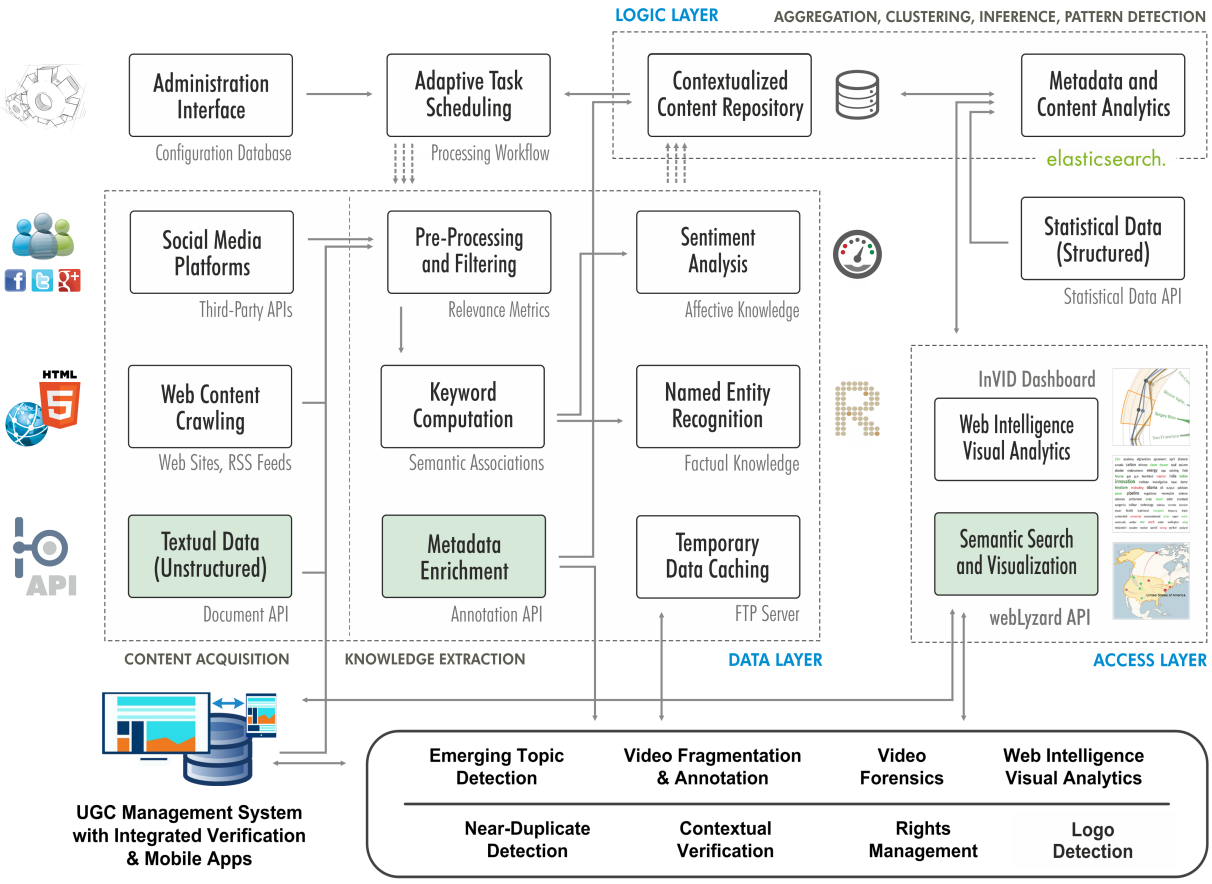

Figure 2: Overview of Application Programming Interfaces (highlighted in green color) for uploading, managing and querying unstructured and structured datasets as part of the InVID content repository.

The API Layer shown in Fig. 2 ensures the required integration of input data (relevant social media content) as well as the extracted metadata. By providing a simple and unified interface to InVID services, the API layer serves as a central building block for InVID applications. All APIs will require authentication and represent content as a set of JSON objects. The envisioned functionality will be described in detail in Section 5 of this document.

The APIs of the webLyzard platform include a *Document API* to ingest unstructured social media content. Its core data structures are Documents, Sentences and Annotations. The Document API can be used for uploading documents regardless of their provenance, as well as annotations from InVID knowledge extraction services - sentiment, named entities, etc. The *Search API* returns a set of query results in the form of unstructured text documents. Its core data structure is Query. The *Embeddable Visualisation API* allows embedding individual visualisations of the webLyzard dashboard into the UGC Management System or third-party applications. Such a visualisation is typically generated based on the results of a search query. Its core data structure is Visualisation. Depending on their

specific functionality, individual InVID services to add or validate metadata elements will be called from the webLyzard core platform and/or directly from the UGC Management System. This is a design decision that reflects the consideration of whether the metadata elements create permanent value within the central data repository, or whether they just support a specific user workflow and thus can be disregarded after the successful completion of the user request (which might entail storing the final result of the user request in the webLyzard content repository). Accordingly, the storage layer distinguishes the fully annotated repository and the temporary data caching via the FTP server (to be hosted by Condat) - the latter for storing temporary results not required for real-time content queries. This dynamic composition of services via RESTful APIs ensures maximum flexibility when developing the InVID dashboard and the UGC Management System, and minimises potential points of failures.

3 Data Acquisition and Pre-Processing

The very first step in the InVID workflow will be to gather online content from social media as an indicator for breaking news events and ongoing news stories. Social media sites serve as today's central media exchange hubs for covering and distributing news events through vast user networks as they unfold. These media documents are of highest interest to journalists, yet they must be verified as authentic first. Linked to the **Data Acquisition** task is the gathering and analysis of more generic social media chatter in order to also detect and extract the most current news topics and thereby to refine the data acquisition queries and improve relevance in the social network content retrieval (see Section 4.2).

When content is ingested into the platform, it must also be prepared for the further processing in the InVID workflow, i.e. the verification and rights management processes. In webLyzard, each content item's metadata is serialised as a *Document* which can be modified and updated throughout the content lifecycle.

For videos uploaded by users via the Mobile App (UGV-Upload App) the workflow may be different depending on the type of integration. For the platform itself, the upload is considered to be just another input channel.

The **Pre-processing** step includes *language detection* to ensure that the gathered documents are in a language supported by the platform's annotation services, and a *newsworthiness* filter to filter out irrelevant content and classify the remaining content by keywords which will support subsequent processing steps (such as refinement of social network queries and clustering for topic detection).

3.1 Social Media Platforms

The initial InVID platform release will support gathering social media from: Twitter, Facebook, YouTube, Instagram, DailyMotion and Vimeo.

webLyzard uses components known as *mirrors* that query Web sources for content. These are the components labelled as Social Media Platforms in Fig. 2. Each mirror type has implemented its own client software that can access, query, and process the mirror's public API (using API keys). All mirrors are part of the same platform component so that they can also be handled together in the system architecture, e.g. to share configurations or to collect multiple content in parallel. Queries for content are based on configuration files known as *input filters* which define the terms to use in a textual query or a list of user accounts to get the most recent posts from. Where possible, queries are executed as exact matches on the terms on each line of the input filter file. To capture a larger number of potential query permutations, the query text can be expressed as a regular expression. Mirrors run as processes connected to a scheduler, repeating the configured queries on the social network API at regular intervals.

For InVID, we extended the existing mirror implementation for Twitter, YouTube and Instagram. We also implemented new mirrors based on the vert.x application framework (`http://vertx.io`) for the Vimeo and DailyMotion APIs. A significant new requirement of InVID is the need to programmatically update the input term list dynamically from the processed content, so that social networks get queried for input terms derived directly from previous queries, thereby allowing for *topic drifts*. A RESTful API implementation has been added as part of the new vert.x mirrors that allows mirrors (currently Vimeo and DailyMotion) to programmatically update its input term list.

3.2 Other Online Sources

While not the core focus on InVID, the webLyzard platform already manages the ingestion of news articles on a regular basis (the Web Content Crawling component of Fig. 2) from international sources. Since access to that content and its enriched metadata can also be of use to journalists and newsrooms using the platform, we will also include this news content feed in the InVID platform.

3.3 Videos sent with the UGV-Upload App

User-generated videos are fed into the system via the UGV-Upload App (Mobile App), for which there are two workflows available:

- After the video was uploaded by the user it is stored by the UGV-Upload-Server. The metadata of the video including the link to the video-file is ingested into the platform for further processing. Afterwards it is processed in the same way as other videos.
- The UGV-Upload-Server will store the video in a streaming platform, from where it will be ingested by APA-IT-ContentHub (a content management system (CMS) also used for curating news content). During the curation process, the user may select a video for verification, thereby initiating the verification workflow through the Verification App. This second workflow is intended for a pilot run organised by partner APA-IT.

Further details on the UGV-Upload App are given in Chapter 6.3.

3.4 Content Preparation

3.4.1 Natural Language Processing

The webLyzard platform includes components for tokenisation, part-of-speech (POS) tagging, and for detecting the language of a given text sample. WP2 will customise these components to the specific requirements of InVID. Language detection, for example, is vital to determine the subsequent steps in the processing pipeline - otherwise collected content can not be classified or annotated, and hence finally searched for and retrieved when relevant. The InVID platform will ensure indexed content is always in one of the (currently) supported languages: at the time of launch, this will be English, French and German.

3.4.2 Video Fragmentation and Annotation

The Video Fragmentation and Annotation service is a web service (hosted at CERTH[1]) that performs temporal decomposition of a video to shots and scenes based on variations of the algorithms described in (Apostolidis & Mezaris, 2014) and (Sidiropoulos et al., 2011). Shots are sequences of frames captured without interruption by a single camera and can be seen as the elementary structural units of the video, while scenes are groups of consecutive shots that correspond to the story-telling parts of the video, i.e. video segments covering either a single event or several related events taking place in parallel. After the fragmentation of the video into these two different levels of temporal granularity, the service identifies the semantics of the video by detecting a number of high level visual concepts (323 concepts selected from list of concepts defined in the TRECVID SIN task (Over, Awad, Michel, et al., 2012)) using the method of (Markatopoulou, Mezaris, & Patras, 2015) and analysing one representative keyframe per shot.

An indicative example of the analysis outcomes can be seen in Fig. 3. The figure shows a screenshot of a web application[2] that uses the same CERTH technologies for video fragmentation and concept-based annotation. Right below the video player, there is a dashed light-blue bar where each fragment of the bar corresponds to an individual shot of the video. The keyframes of the automatically defined shots appear in the left side of the User Interface (UI), below the brief description of the web application (see screenshot). This widget contains only the keyframes of the first 21 shots of the video, while the entire set of extracted keyframes can be accessed by clicking on the "See all

[1] http://multimedia2.iti.gr:8080
[2] http://multimedia2.iti.gr/onlinevideoanalysis/service/start.html

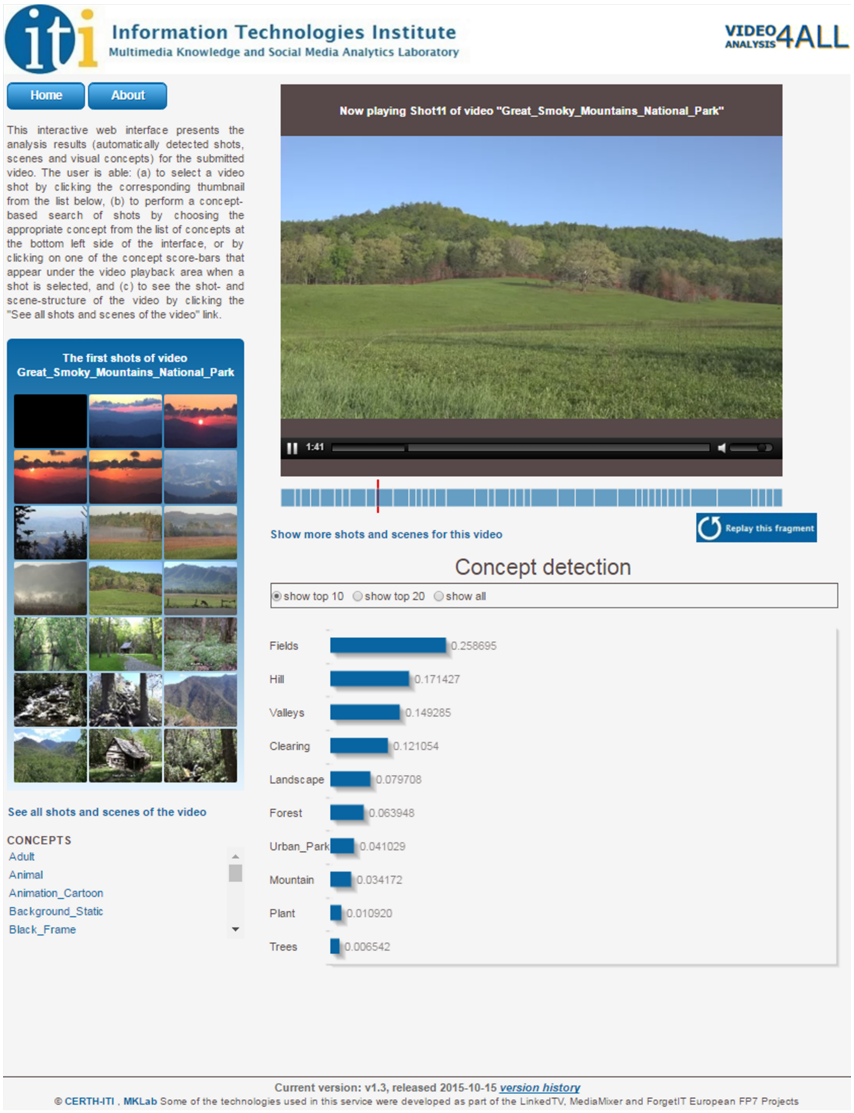

Figure 3: A screenshot of a web application that uses the same CERTH technologies for video fragmentation and concept-based annotation.

shots and scene of the video" link right underneath. By clicking on one of the keyframes, the video player loads and displays the corresponding shot segment of the video. Moreover, the area below the player shows, by default, the 10 most relevant visual concepts detected for the selected shot, providing a concept-based annotation for the selected media fragment (also the top-20 and the full list of concepts can appear by clicking on the corresponding radio buttons of the UI). By performing this annotation procedure for the entire set of video fragments, the service provides a shot- and scene-level concept-based annotation of the video.

The processing starts right after fetching the video file. The video shot and scene segmentation analysis is approximately 5-6 times faster than real-time processing (where real-time analysis would require processing time equal to the video's duration) depending on the resolution of the given video. Following, the required time for concept detection is related to the number of identified shots (since this type of analysis is performed on a per shot / keyframe basis as described above). Based on the fact that the service needs approximately 1.2 sec per keyframe for identifying the visual concepts that appear in it, and according to our evaluations that included several different types of videos (e.g. news videos, documentaries, sitcoms, talk shows), we anticipate that the entire video fragmentation and annotation analysis is approximately 1.5 times faster than real-time analysis (depending again on the number of the processed keyframes).

The input parameters of the service include: (a) the URL of the video file that needs to be analysed (supporting both HTTP and FTP protocols), (b) a set of access credentials (i.e. username and password) for authentication in case of password-protected content repositories, and (c) a user-key for authenticating the authorised users of the service and logging their activities. The output of the service contains: (a) an XML file with the shot segmentation analysis results, (b) an XML file with the scene segmentation analysis results, (c) an XML file with the concept detection analysis results, (d) a JSON file with the same shot, scene and concept detection analysis results, and (e) a set of image files that correspond to the extracted keyframes for the automatically defined shots of the video (indicating three keyframes per shot).

The communication between the user and the service is enabled via its RESTful API and is synchronous only during the transmission of the HTTP POST/GET calls. In particular, after submitting a video for analysis via committing a properly formed HTTP POST call, the user gets the response "*The RESTful call has been received. Processing will start as soon as the provided content is downloaded.*" and the communication turns to asynchronous (i.e. it is terminated until the next HTTP POST/GET call of the user), enabling the submission of a new analysis request from the same user. However, for time efficiency purposes the service applies a queuing strategy for ordering the incoming analysis requests and the analysis is performed in a one-by-one basis (and not in parallel). During the analysis the user is able to get information about the progress and the status of the processing while after the end of it the user is allowed to retrieve the analysis results (i.e. the created XML and JSON files) and the extracted keyframes, via a set of HTTP GET calls.

4 Knowledge Extraction

4.1 Detection of Newsworthiness

We want to maximise the relevance of content indexed and stored by the platform. To support this, in InVID we will adapt the webLyzard platform components for Pre-Processing and Filtering, Keyword Computation and Named Entity Recognition to identify, filter and rank content according to its newsworthiness. While this term means literally "interesting or important enough to report as news" (Merriam-Webster), in InVID we are particularly seeking news events which potentially lead to the posting and sharing of User Generated Video around those events. As such, in InVID we will include:

- Newsworthiness filtering on the extracted keywords. Since we consider the extracted keywords from input content feeds a useful indicator of current news we need a view on those keyword lists which highlights the keywords directly relevant to a news story or event (such as a person, a type of event or a location) and removes those which may well be occurring regularly in the content set but do not contribute to a news-centric view (e.g. typically from news articles we see the current month or the name of the providing press agency occurring in the keyword lists).

- Keyword computation to focus on removing duplicates and disambiguating terms to news-centric entities such as persons, organisations and locations. A keyword list can contain for example "Donald", "Trump" and "Donald Trump" as separate keywords. The current computation approach uses statistically significant n-grams. This needs to be extended to identify where separate keywords refer to the same thing (entity), so that e.g. the previously mentioned three keywords can be merged to one single entity.

- Named Entity Recognition (NER) for news events. The current NER component is configured for other use cases (e.g. environmental), only performs location detection for locations with a population over 500 000 inhabitants and is not integrated with the keyword computation. We propose to use labels of entities detected by the NER component to support the keyword computation as mentioned in the previous point, so that e.g. "Donald Trump" could be recognised as a label of a relevant entity (a politican) and single keywords like "Donald" and "Trump" could be correctly disambiguated as references to that entity.

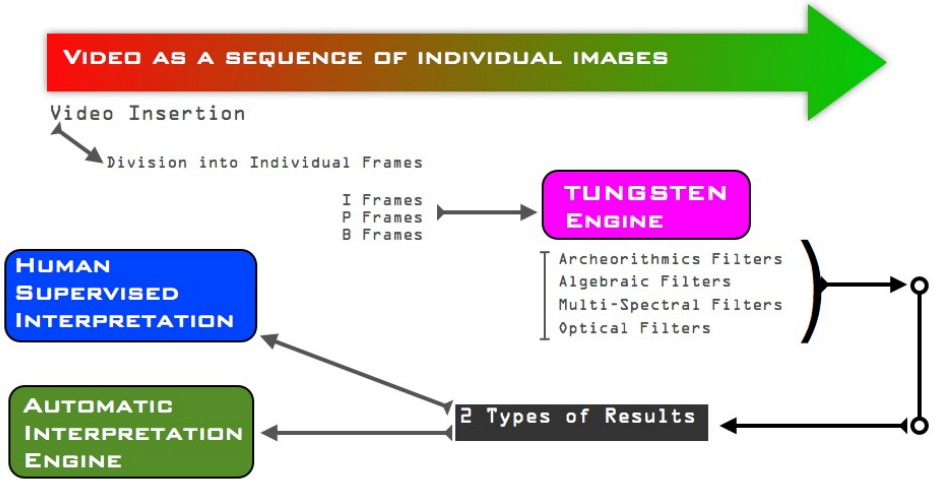

Figure 4: Overview of the video forensic analysis process using the TUNGSTEN engine.

4.2 Topic Detection

The data acquisition aims to ensure all potentially relevant, newsworthy social media content gets collected and indexed in the InVID platform. We add a topic detection component to the platform that surfaces current newsworthy events to the user. The topic detection component extracts keywords/entities from the content streams (from social media platforms and/or web content crawls) to identify breaking news events, allowing applications to obtain a pre-filter on the content to be analysed.

To guide journalists to more specific ongoing and emerging news events, the acquired content is being similarity-clustered such that distinct clusters may represent distinct news events. These clusters are visible to users of the Web portal via a cluster map visualisation. Determining newsworthiness of a cluster will be based on occurrence counts of news-relevant keywords. Clusters determined to be representative of newsworthy events can be used to automatically add new topics to explore, either directly appearing in the InVID dashboard or via the API to other InVID applications.

4.3 Content Verification

4.3.1 Video Forensics

TUNGSTEN already offers numerous and various types of filters in order to deeply analyse and assess still images and let us try to expose alterations and forgeries. One approach is based on the fact that a video flow is in reality not a continuous stream. From the image and frame perspective this pseudo flow is made with three different types of still frames: I, P and B. We intend to use I frames, which are said to be "full", as source images in TUNGSTEN engine. We then intend to apply on these I frames the algorithms and filters from TUNGSTEN which are distributed in 4 different families. Each family stands for a particular forensic approach and for a specific type of results. Fig. 4 illustrates the forensic analysis process using the TUNGSTEN engine, yielding partially automatic interpretations, with some human supervision required. Fig. 5 gives more details of the analytical steps implemented in TUNGSTEN and applied to binary videos through the forensic analysis service as part of the InVID platform.

We anticipate that we will be able to gather some filters and their results in heuristics that will be the base for and automatic interpretation engine. However, for some other filters and results the human supervision shall remain necessary.

We do not know yet if we will be able to design and code an automatic interpretation engine dedicated to this aspect of the forensic analysis. Indeed, if we already know that we will be able to retrieve numerous and various type of information, we still have to discover how to link them and use them in order to produce or, at least, participate, to the final decision. If these links and relations can

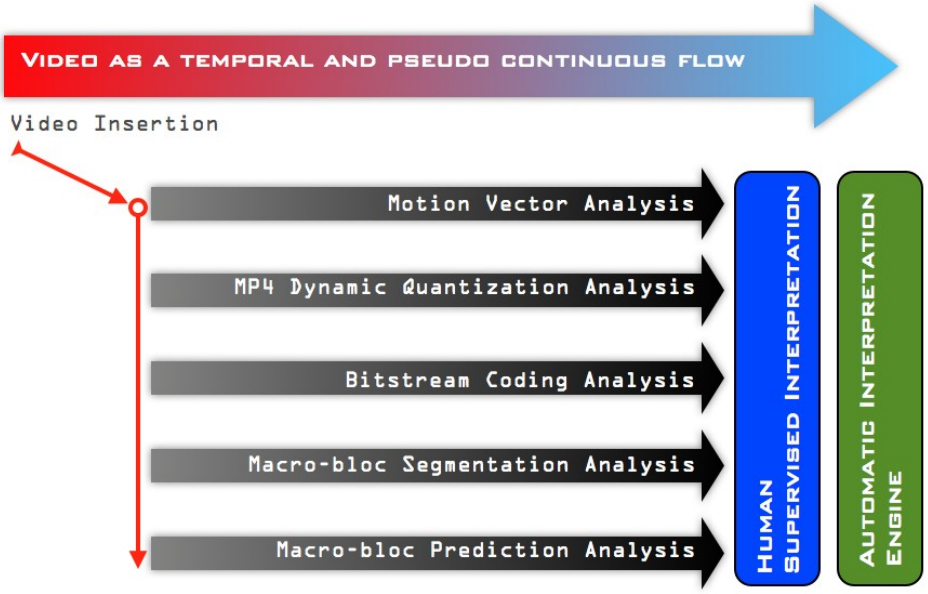

Figure 5: Details of the video analysis steps performed by the TUNGSTEN forensic engine.

be gathered in a limited number of heuristics, we then shall be able to code a software engine for automatic interpretation. However, we strongly anticipate that the human supervision might remain useful or even necessary for some filters and results.

4.3.2 Near-duplicate Detection

Near-duplicate detection aims at identifying videos that are highly similar to a query video or keyframe. Being able to perform such detection is extremely valuable for journalists during verification, especially in cases where they can identify an earlier version of a video (which typically means that the video is a reposting from a previously captured scene and not from the breaking story of interest). There are different ways in which InVID will support near-duplicate search:

- by linking to third-party services (e.g. TinEye, Google Reverse Image search) that perform near-duplicate image search at web scale; this type of search is only applicable to selected video keyframes (or other images of choice) and will be part of the content verification application (Section 4.3.3);
- by performing near-duplicate video search on a collection of videos that have been indexed by the InVID platform; this will support both image-to-video (i.e. provide a keyframe or image as input and get a list of videos as output) and video-to-video search (provide a video as input and get a list of videos as output).

The latter functionality will be implemented through a sophisticated video analysis pipeline that comprises the following steps: a) keyframe extraction and selection (with the use of the video fragmentation service described in Section 3.4.2), b) feature extraction (per keyframe), c) aggregation of keyframe-specific features to video-level features, d) feature indexing, and e) search in the set of indexed features. Hence, two methods will be exposed by this module to the InVID platform (in the form of a RESTful service):

- **Index:** Add a new video to the InVID index. The index can be optionally organised in collections with different names (which may facilitate data management and improve the efficiency and effectiveness of the near-duplicate search module).
- **Search:** Perform near-duplicate search in the index of InVID videos and return a ranked list of videos that are considered to be near-duplicates to the input video.

These methods will be also applicable to single keyframes (or images), i.e. it will be possible to index and search on a per keyframe basis. Note that the index construction, maintenance and search is controlled (orchestrated) by the content verification application (section 4.3.3).

In terms of technical methodology, we consider three main parts in the Near-duplicate Detection module: a) keyframe features, b) feature aggregation (to move from a keyframe-level to a video-level representation), c) feature indexing. For the first, we plan to evaluate the optimised VLAD features of (Spyromitros-Xioufis, Papadopoulos, Kompatsiaris, Tsoumakas, & Vlahavas, 2014), along with the CNN features extracted from the video annotation service (Section 3.4.2), as well as the inexpensive LBP features (Ojala, Pietikäinen, & Mäenpää, 2002). In terms of feature aggregation, we will test the approach by (Shang, Yang, Wang, Chan, & Hua, 2010), which relies on clustering keyframe descriptors into "keyframe words" and creating a bag-of-keyframe-words vector, which is representative of the whole video. Finally, in terms of video indexing, we will rely on inverted indices, while keyframe-level indexing will also be supported by means of Product Quantisation and Asymmetric Distance Computation, which was proven to be highly scalable in (Spyromitros-Xioufis et al., 2014). Base implementations for the above elements are available and different module configurations are evaluated using the CC_WEB_VIDEO dataset as benchmark (Wu, Hauptmann, & Ngo, 2007), in order to select the one that will form the core of the first version of the component. More details on this will be presented in the upcoming deliverable D3.1.

4.3.3 Logo Detection

Logo detection in videos is often useful as a means of linking the video to particular sources, channels or groups with known background. Consider the example of Fig. 6, in which the video features the logo of the China Central Television, which may be unknown to several journalists and news professionals working in Western countries. Logo detection will be exposed as a RESTful service that will extract candidate logos from an input video and perform a query against a database of known logos (and the associated Wikipedia articles). In case of matches, the end user (journalist) will be presented with the Wikipedia article(s) related to the detected logos. This could be helpful in the analysis of contextual information for that video.

A preliminary logo extraction approach based on constructing and processing the histogram of keyframe differences is currently tested and more details on the initial implementation of the module will be made available as part of the upcoming deliverable D3.1.

4.3.4 Contextual Verification

The contextual verification RESTful service will take as input online videos and compute a number of credibility indicators that can be used by journalists as cues to the verification process. These indicators will relate to the following aspects:

- **Poster:** Several valuable cues can be derived by collecting information about the online account/source that posted the video. Such cues can for instance include the age of the account (based on the premise that old accounts are typically more reliable), the activity of the account (e.g. to quickly identify cases where an account was set up to post only a single video), the avatar (to check whether it is original or copied from elsewhere on the Web), etc.

- **Comments:** One may often discover valuable verification cues in the comments under the video or in messages posted in other platforms such as Twitter. The service will retrieve such comments and will rank them based on their interest for verification (e.g. comments containing keywords such as “fake” or “tampering” will be ranked higher).

- **Web context:** In case the video is embedded in a website, or the poster of the video has a known homepage, additional verification cues can be extracted for the respective domains by bringing together ratings such as Web-of-Trust[3] and scores derived from the position (importance) of the webpage in the Web graph[4].

[3] https://www.mywot.com/

[4] http://wwwranking.webdatacommons.org/

Figure 6: Depiction of foreseen logo detection functionality.

– **Location:** In case location cues are possible to extract from a video, either using text processing or image processing methods (e.g., by comparing the depicted scene against a repository of scenes in an area of interest), these will be automatically extracted and returned by the service.

Several of the credibility indicators that will be extracted for a video will be based on the approach presented in (Boididou, Papadopoulos, Kompatsiaris, Schifferes, & Newman, 2014). This was particularly designed for Twitter posts, but several of the proposed indicators are directly applicable for online videos, while new credibility indicators that are suitable for videos will be added. Location cues will be extracted using a combination of Natural Language Processing (in particular, Named Entity Detection) and statistical language models such as the ones described in (Kordopatis-Zilos, Papadopoulos, & Kompatsiaris, 2015). More details on the progress of development regarding the contextual verification component will be provided as part of the upcoming deliverable D3.1.

4.4 Copyright Management

The module responsible for Copyright management is developed in WP4. The module offers a RESTful API that provides access to the data model depicted in Fig. 7. As shown in the diagram, the current model represents generic concepts for users of User Generated Video as well as specific concepts for the YouTube application, which will be the focus of the first version of the copyright management module.

Spring Boot and the Spring Data RESTful are being used to generate the API based on the previous domain model. This way, it is possible to post the URL of a YouTube video for which we want to gather authorship information. For instance, Code 1 shows an example using CURL where a particular YouTube video is registered using its URL.

Code 1: Example of a curl command to register a YouTube video in the Copyright Management module.

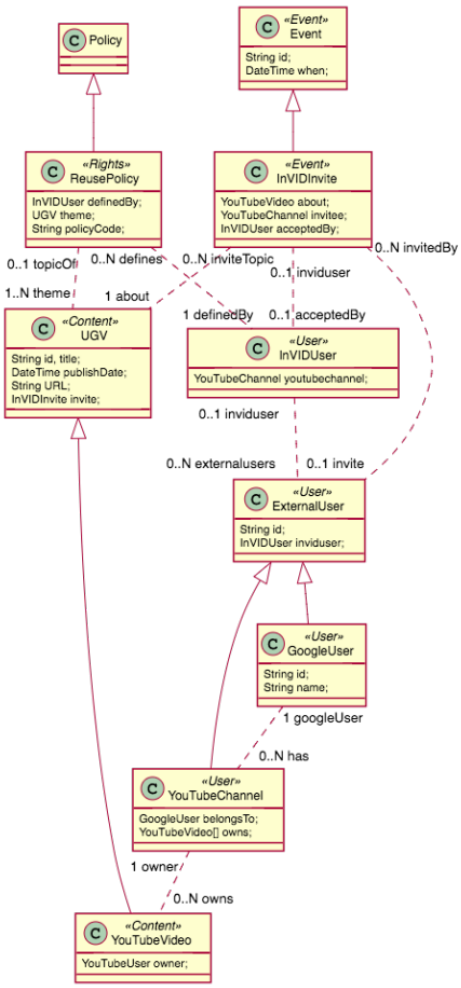

Figure 7: Domain model of the Copyright Management module.

```
curl 'http://localhost:8080/youTubeVideos' -i -u 'user:password' -X POST
-H 'Content-Type: application/json' -H 'Accept: application/hal+json' -d
'{
  "url" :"https://www.youtube.com/watch?v=Xb0P5t5NQWM"
}'
```

The service's response is to create a new entity of type YouTubeVideo, which is enriched with metadata gathered using the YouTube API. The response is a JSON message, shown in Code 2, that contains the retrieved metadata (for instance title or publishing date) plus links to additional entities that provide additional functionality.

Code 2: Response after registering a YouTube video in the Copyright Management module.

```
{
  "title": "Golden Eagle Snatches Kid",
  "publishDate": "2012-12-19T13:55:28Z",
  "url": "https://www.youtube.com/watch?v=Xb0P5t5NQWM",
  "_links": {
    "self": {
      "href": "http://localhost:8080/youTubeVideos/Xb0P5t5NQWM"
    },
    "youTubeVideo": {
      "href": "http://localhost:8080/youTubeVideos/Xb0P5t5NQWM"
    },
```

```
    "invite": {
      "href": "http://localhost:8080/youTubeVideos/Xb0P5t5NQWM/invite"
    },
    "youTubeChannel": {
      "href": "http://localhost:8080/youTubeVideos/Xb0P5t5NQWM/youTubeChannel"
    }
  }
}
```

For instance, "_links.youTubeChannel.href" links to the YouTube Channel the registered video corresponds to. Additional information about the channel, also retrieved during video registration using YouTube API, is available by interacting with the API performing and HTTP GET to the corresponding link URL, as shown in Code 3.

Code 3: HTTP command to retrieve the YouTube Channel corresponding to the previously registered video.

```
GET http://localhost:8080/youTubeVideos/Xb0P5t5NQWM/youTubeChannel
```

The response to the previous HTTP call is a JSON message with all the gathered information about the YouTube video channel, including links to the corresponding Google user, as shown in Code 4.

Code 4: JSON response message with all the metadata about the channel for the registered YouTube video and other associated resources.

```
{
  "title": "The MozaiK",
  "_links": {
    "self": {
      "href": "http://localhost:8080/youTubeChannels/UCna6ThFB4_PsYrYNZB7Vx0w"
    },
    "youTubeChannel": {
      "href": "http://localhost:8080/youTubeChannels/UCna6ThFB4_PsYrYNZB7Vx0w"
    },
    "inVIDUser": {
      "href": "http://localhost:8080/youTubeChannels/UCna6ThFB4_PsYrYNZB7Vx0w/inVIDUser"
    },
    "userVideos": {
      "href": "http://localhost:8080/youTubeChannels/UCna6ThFB4_PsYrYNZB7Vx0w/userVideos"
    },
    "googleUser": {
      "href": "http://localhost:8080/youTubeChannels/UCna6ThFB4_PsYrYNZB7Vx0w/googleUser"
    },
    "invite": {
      "href": "http://localhost:8080/youTubeChannels/UCna6ThFB4_PsYrYNZB7Vx0w/invite"
    }
  }
}
```

The complete and live documentation for the copyright management module is available online[5].

5 Data Integration

5.1 Integration Strategy

InVID provides a number of individual components for content verification in near real-time from both video and news content. These components will be developed for multiple use cases, by several project partners. The success of the InVID project depends on how well the developed components can be integrated into a coherent, interlocking framework.

[5] http://copyrightly-api.invid.rhizomik.net/docs/index.html

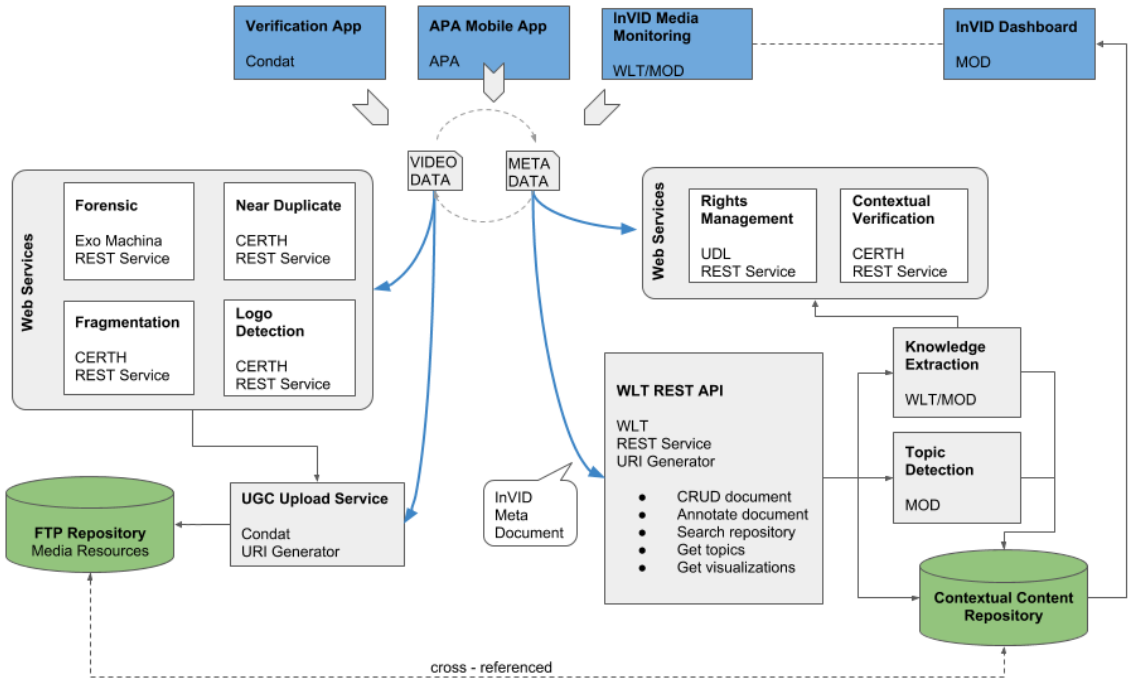

Figure 8: Overview of the data flow from the applications to the data repositories.

The overall integration strategy for InVID is to design all components as black-box modules with an explicit RESTful end point for data transfer, and to integrate these modules via a set of use-case driven data workflows that are managed via the partners' applications (see Section 6). The design of distributed data flow control was chosen to reduce the overall dependency on a single point of failure and thereby to guarantee the success of the individual partners' use cases — without compromising on the overall integration of all components into a singular, coherent architecture. An overview of the current status of the respective RESTful endpoints is provided in Appendix A.

5.2 Data Management

The InVID platform maintains two separate data repositories to be used by all project partners for exchanging and persisting data: a FTP-based repository holding binary data, and a contextual content repository for metadata. Documents within these two repositories are cross-referenced via unified resource identifiers (URIs), allowing to obtain both metadata and binary data for any given document if available. Each data repository can be accessed via a dedicated service endpoint that also provides unique resource idenifiers (URI) to be used for cross-referencing documents between the repositories. Fig. 8 shows the data flow from the applications as data producers along the various data enrichment services to the repositories. Data is fed into the platform via the three core InVID applications either in video or in metadata format and it gets persisted in the respective repository after being enriched by the InVID web service layers. For binary (video) formats, Forensic Analysis, Near Duplicate Detection, Keyframe Fragmentation, and Logo Detection services are available. For textual metadata formats, Digital Rights Management and Contextual Verification are available. Topic Detection and Knowledge Extraction as described above are provided by the webLyzard RESTful API. Video data processing produces metadata, and metadata processing may produce video data (e.g. YouTube data, HTML data through HTML anchors). All extracted data will be made available to the InVID partners via the webLyzard RESTful API for querying, modification (e.g. enrichment), and visualisation.

5.3 Document Exchange Format

As the central structure of data interchange between the two data repositories, a unified document model has been designed that describes all relevant information required and/or provided by each InVID project partner. This *InVID meta document* shall be used to upload any content to the InVID portal

for efficient data persistence and exchange between the InVID partners. Since the InVID dashboard acts as the central repository to hold any document metadata (as extracted from both video and text documents) for visualisation, it needs to integrate all fields required and/or provided for the partners.

In particular, the proposed format was designed to fulfill:

1. **Functional Contract** - using a functional abstraction, the document model provides a contract to the user of what each field provides, providing transparency and intuition to its users.
2. **Extensibility** - the format needs to be flexible to allow for potential extensions to be made during the lifetime of the project
3. **Completeness** - the document model should be as simple as possible, while at the same time cover all use cases defined in the project proposal.

The document model is provided as JSON structure to be interchanged via the APIs. In Fig. 8, the InVID meta document is used as standard format to transfer all metadata required for visualisations by the InVID dashboard via the webLyzard RESTful API to the respective contextual content repository.

We propose the following *functional sections* to be used in the InVID document model, providing a clear contract to the user about their function to the InVID platform:

1. **Metadata** - providing generic information regarding the complete document
2. **Sentences** - the document's textual content representation, after sentence tokenisation, part-of-speech tagging, etc.
3. **Cross-document Relations** - defining any document-to-document relation, including cross-references to objects from the binary DAV repository, re-tweet relations, nested comment threads, etc.
4. **Document Features** - document features that impact UI selections in the portal (e.g. filters), such as channel IDs, video license information, verification features from the InVID partners, etc.
5. **Annotations** - surfaced document annotations, such as named entities (text), video fragments (video), etc.
6. **Document Statistics** - time-stamped document statistics such as social media metrics (likes, view counts, re-tweets, etc.)

By modeling each of the above *functional sections* as a dictionary, we ensure that the model can be extended by new features, relations, etc. at any time without requiring any changes to the model itself. Code 5 shows an example document represented using the proposed document model.

Code 5: A sample document showing the proposed InVID unified document model.

```
{
      "repository_id": "invid.weblyzard.com",
      "uri": "http: //www.bbc.com/news/science-environment-33524589",
      "title": "New Horizons: Nasa's spacecraft speeds past Pluto",
      "meta_data": {
            "author": "Jonathan Amos",
            "published_date": "2016-04-12",
            "polarity": "0.342",
            "reference_uri": "http://..."

      },
      "sentences": [{
            "id": "595f44fec1e92a71d3e9e77456ba80d1",
            "value": "New Horizons: Nasa's spacecraft speeds past Pluto",
            "is_title": "TRUE",
            "pos_list": "NN NN :\\' NN :NN CC JJ NN . \\'",
            "tok_list": "0,2 3,1919,2021,2222,3333,3435,4243,4647,5556,6262,6363,64",
            "sentence_number": 0,
            "polarity": -0.783
```

```
20      }],
21      "annotations": [{
22            "annotation_type": "InVID_video",
23            "start_pos": "0:12:59",
24            "end_pos": "0:34:34",
25            "properties": {
26                  "frame": 12312,
27                  "truth_class": "unconfirmed",
28                  "algorithm": "ILCv3.1",
29                  "story_id": 1232122,
30                  "scene_id": "Sc1"
31            }
32      }],
33      "relations": {
34            "invid:ref": "http://multimedia2.iti.gr:8080/keyframe/1a2b3c4d5e/shot4_1"
35      },
36      "features": {
37            "license": "MIT",
38            "truth_class": "unconfirmed"
39      },
40      "statistics": [{
41            "views": 1202,
42            "likes": 23,
43            "timestamp": 1464868697
44      }]
45 }
```

5.4 Application Programming Interfaces of the Core Platform

5.4.1 Document API

The Document API acts as a RESTful end-point to the contextual content repository. It supports basic create-retrieve-update-delete (CRUD) functionality. It uses artificial content identifiers (numerical) to uniquely identify documents within in the repository. On document creation, the RESTful request returns this identifier for future reference to the created document. The Document API accepts and returns content of MIME type `application/json`, in UTF-8 encoding. Creating a document (upload) converts the submitted JSON representation into an internal structure which is then send for processing through the document enrichment stages of the InVID platform (see Section 2). The Document API additionally offers an annotation service that allows to annotate documents without adding them to the contextual content repository. The annotation service provides any of the following metadata enrichments:

- Part-of-speech Tagging (POS)
- Named Entity Recognition and Resolution (NER)
- Keyword extraction
- Semantic orientation

The Document API is constantly monitored for availability and correctness.

5.4.2 Search API

The Search API acts as a RESTful end-point to the elastic search index sitting on top of the contextual content repository. The elastic search index provides additional search functionality on large-scale data that would not be possible on an unindexed content repository, such as pre-computed filter aggregations and content indexing. The Search API therefore enables real-time full-text search over very large data repositories, data filtering across any fields provided by the InVID meta document such as video license information or any other meta information provided by any of the InVID partners.

JSON query language is used to issue queries to the Search API, returning JSON results. Code 6 demonstrates some example queries for the Search API.

Code 6: A sample query to the Search API, with `gte` being short for greater than equal, and `lte` for less than equal, respectively.

```
{
 "sources" :["invid_media"],
 "fields" :["title", "summary"],
 "query" :{
  "text" :{
    "phrase" :"happing right now"
  }
 },
 "filter" :{
  "date" :{
    "gte" :"2016-05-01",
    "lte" :"2016-05-31"
  }
 },
 "count" :100,
 "offset" :0,
 "ranking" :{}
}
```

The Search API accepts and returns content of MIME type `application/json`, in UTF-8 encoding. The Search API service requires authentication via JSON Web Tokens and it is constantly monitored for availability and correctness.

5.4.3 Visualisation API

The InVID dashboard offers a feature-rich and customisable solution for visual analytics, semantic search and Web intelligence applications. To support use cases that require a more granular approach, the Visualisation API enables the integration of distinct portal components into third-party Web applications. In particular, this external integration is achieved via <iframe> embeddings of HTML components into a partner's Web application. We refer to Code 7 for examples of InVID embedded visualisation, showcasing some of the parameter configurations supported.

Code 7: Examples of <iframe> embeddings of visualisation components from the InVID dashboard for use in external Web applications (authentication required).

```
<!-- The tag cloud using the default search term -->
<iframe width='400' height='600' src='/embed/tags' frameborder='0'>
<!-- The geo map using a custom search term -->
<iframe width='400' height='600' src='/embed/geo?search=fracking' frameborder='0'>
```

The Visualisation API service requires authentication via JSON Web Tokens and it is constantly monitored for availability and correctness.

6 Applications

6.1 Introduction

InVID supports different kinds of use cases, the requirements of which have been described in detail *D6.1 Initial Industrial Requirements*. There are the following use case scenarios which involve different workflows:

- **Trending Topics:** The first scenario is triggered by the automatic searching for newsworthy video content through the **Emerging Topic Detection** component within social networks, particularly Twitter. This results in a list of resources which are then ingested and registered by the Platform. At this moment this includes only the metadata of the video not a download of the

physical copy. The Verification App user gets notifications about these results according to her interests or profile. From these results she selects videos which she wants to be verified and the respective properties, such as context, forensics, near duplicates, source, legal rights, etc. If the analysis requests involve an examination of the video itself then the video will be fetched and analysed through downloading a physical copy.

- **Search:** This scenario is very much the same as the first one with the difference that the search for resources is triggered from a Verification App user through specific search terms.
- **Single Verification Request:** In this scenario the Verification App user requests the verification of a specific video which she might have received vie email or found herself on the Internet. In this case the video will be uploaded to the Platform directly; in case of a physical file it will also be stored in InVID Media Resource Repository; otherwise only the URI will be stored. Then the video analysis and verification process will be triggered and the results presented in the Verification App.
- **UGC Upload:** In this scenario a user directly uploads his own video content to the Platform from his own device, be it a mobile, tablet or PC. The video itself is stored in the media resource repository. As the user is registered at the system (in the UGC Management Application), this UGC can be considered trustworthy, so verification should not be necessary. Nevertheless, the user of Verification App can trigger all InVID analysis services to get more background information as well as information about the video itself.

To support these different use case scenarios, the following applications are provided by InVID:

- The Verification App
- The UGV Upload App
- The Multimodal Analytics Dashboard

These applications will be described in more detail in the following.

6.2 Verification App

The Verification Application will be built on top of the InVID platform, integrating its plentiful features. The application will provide a graphical user interface using Web technologies that allow users to select UGC from different sources for verification; and, following the best practices of the industry for this, invoke the InVID platform's components for investigating e.g. video file manipulations, previous re-use of the same content, use the other supported verification functionalities, and also identify the rights associated with each UGC media item and its re-use costs. For the purpose of helping its users to follow the best practices of the industry for verification and rights management, the InVID Verification Application will not simply expose the different InVID verification functionalities to them, but will also define a number of workflows and guidelines that will help users to optimally use these functionalities in their everyday work. This will allow users in news organisations, where journalists are typically working, to decide under consideration of public interest, time and cost conditions whether to publish an item or not. The Verification Application will help its users to get, for a chosen media item (video), answers to at least the questions, who posted this video, what do we know about the poster's reputation, possible intentions, views, what is shown in the video, has the video file been manipulated, what do we know about the rights to re-use the video and whether the author can be contacted.

The Application is designed according to the following general concepts, it

- guides through the verification workflow by gathering clues and making decisions following the verification best practice and ethic guidelines
- collects videos and labels the relevant items for a story to perform verification activities
- supports multiple users working in parallel on the same UGV item, share annotations and verification state thus enabling an easy integration in the work process

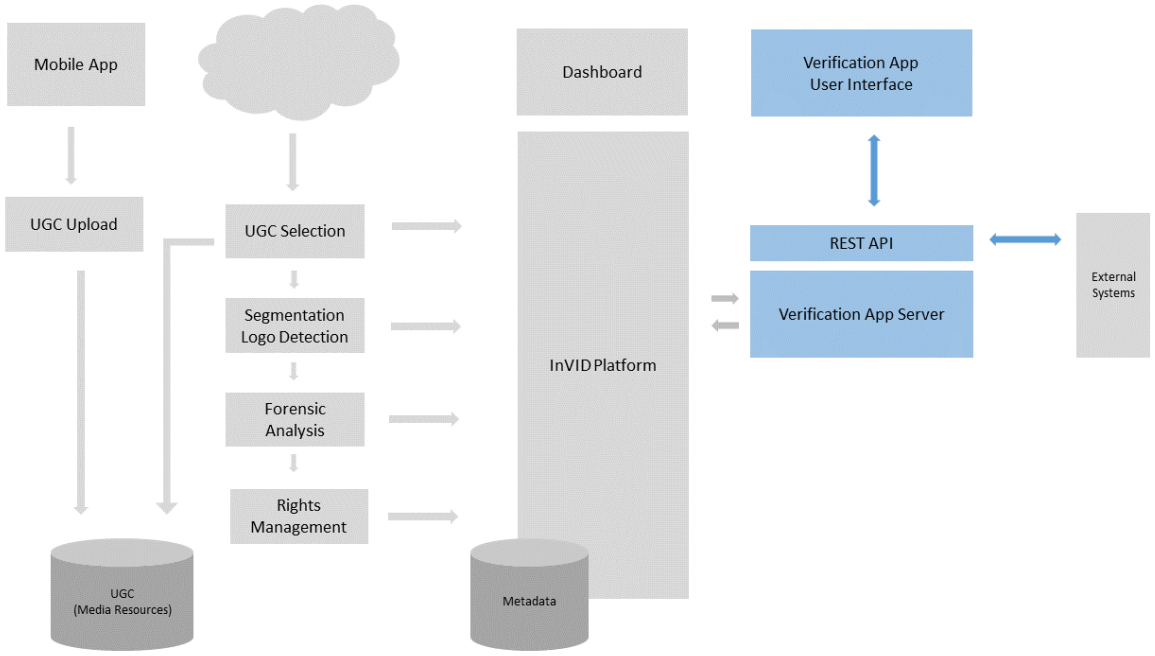

Figure 9: Architecture of the Verification App within the InVID system context.

- stores a certain amount of selected videos
- offers a look and feel which is customisable regarding colours and layout to comply with the customer's NCS UI and
- keeps the user in the loop, who manually decides to label a source as trusted.

The Application offers features for the user to

- geo-locate the focused materials during the retrieval by positioning the video and related social network feeds on a map with different views and zooming
- retrieve videos and images and order the results
- gather metadata for verification regarding the who, what, where, when, why automatically and displays them in a structured summary.
- find and verify the author by identification of Social Network use and cross check of reliability
- check for near duplicates and see if the file is original or if it has been processed through software or tampering
- call 3rd party tools to check the weather at the location of the video, for retrieval in social networks, checking ICANN registration, and translation of keywords in different languages
- support the rights clearing process and in a traceable way, retrieve content owner's contact and attribution, and set the credit.

The Application provides a RESTful API that is functionally compatible to the MOS[6] protocol which

- exports a list of verified UGC items from the InVID application to other systems containing annotations, basic metadata, validation state, legal, cost
- includes UGV received from other sources , from my computer, from NCS for verification
- generates notifications to inform other applications about new relevant items.

[6] http://mosprotocol.com/

6.2.1 App Architecture

The high-level architecture of the Verification App is shown in Fig. 9. The Verification App is composed of a user interface for user interaction, and a server application hosting the verification logic. The two components interact via a RESTful API that also acts as an interface to external systems. The Verification App Server is being invoked through the Knowledge Extraction process provided by partners webLyzard/MOD, and consumes contextual content from the contextual content repository.

6.2.2 RESTful API

This section describes the RESTful API calls used and provided by the Verification App for the verification process and verification management. Most of these REST URL are just wrappers of RESTful API calls provided by the Platform or other InVID services. This list is only preliminary and subject to extension.

Code 8: Planned RESTful web service functionality for programmatically accessing the Verification App.

```
Base URL: http://invid.condat.de/va/api

GET /search?<parameters>
=> list of mediaresources

GET /resource/?<filter>
=> list of resources, filtered by <filter>

GET /resource/{id}
=> single resource with all metadata (complete document)

GET /resource/{id}/status
=> get the processing status of a mediaresource

GET /mediaresource/{id}/author
=> get a reference to an author

GET /author/{id}/
=> get all metadata for an author; like twitter id, email, verification status, etc.

POST /resource/
=> creates an InVID resource; provide URI and metadata in the body;
return uuid

POST /verify/{id}/author/{id}

POST /verify/{id}/nearduplicates
=> list of duplicates of a resource; empty if none found

POST /verify/{id}/forensic
```

6.3 UGV Upload App

The UGV Upload App (Mobile App) helps news publishers to work with their user community on user generated (UG) videos - either with all of them or just with specific user groups (like members of emergency services or local sports clubs). The UGV-Upload App offers the users a direct channel to provide UG video content to their news publisher. On the other hand, it also gives the news publisher the option to interact with their community. An overview of the UGV-Upload App is given in Fig. 10. The tool consists of the following three components:

- The Mobile App itself as a frontend for the users

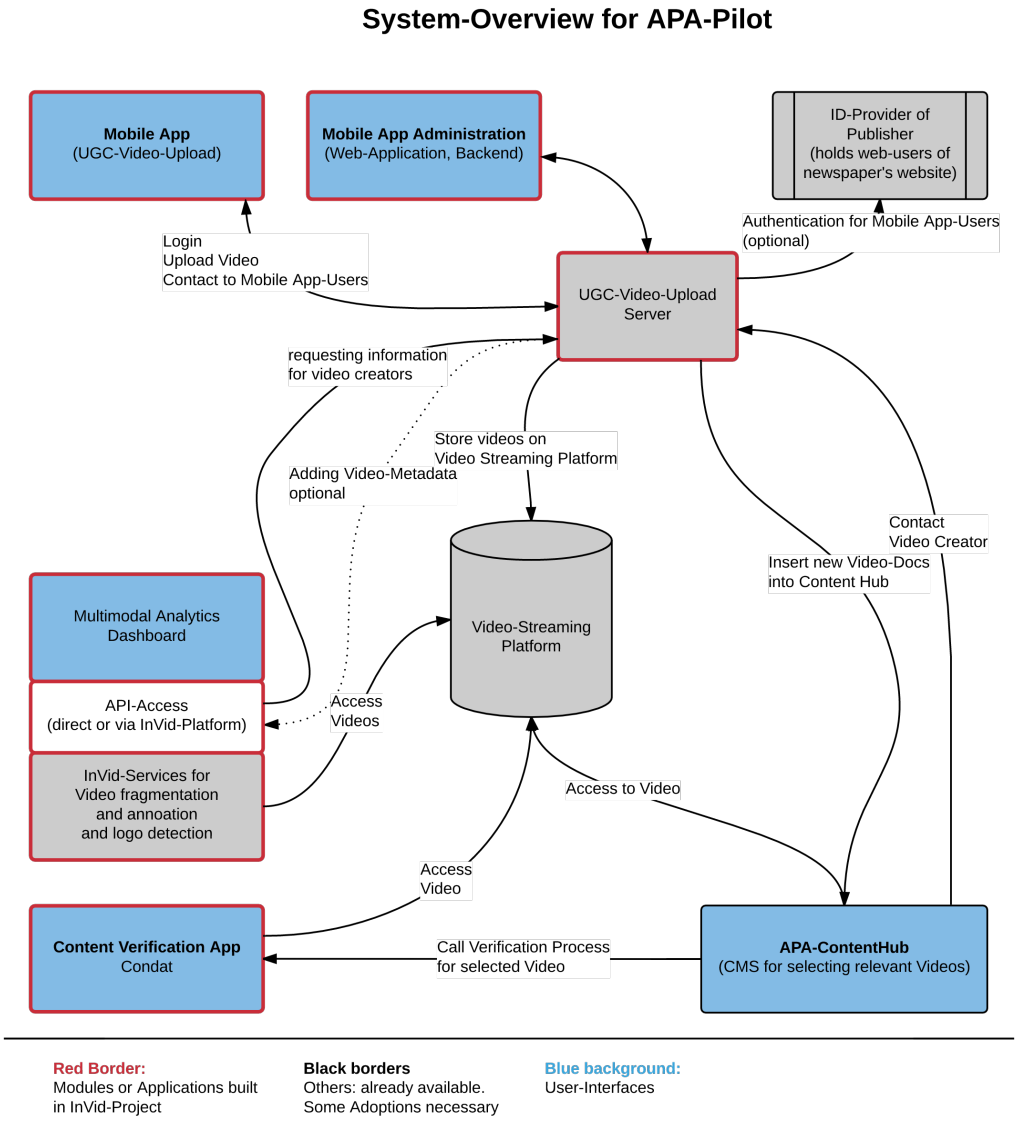

Figure 10: Overview of the UGV Upload App as used for the APA-Pilot.

- The Mobile App Administration Interface for the backend-management used by the news publisher
- The UGV-Upload-Server which connects the Mobile App. the administration interface, the video storage system and other services used for curating the video content.

For the APA-Pilot the editorial user will use the APA-ContentHub to monitor the incoming video-feed, view the attached metadata provided with the video or extracted by other services (e.g. video fragmentation and annotation, logo detection) and decide which video should be verified. For verification purposes the user will call the Content Verification App for the selected video.

6.3.1 The Mobile App

The Mobile App is a native white-label application (iOS and Android). The main use case for this App is providing the user an interface for upload videos for distribution by the publisher. In addition the user will have some administrative functions and a notification area for a direct contact with the publisher.

- **Use Case: Upload Video -** The user may upload a video which is either already stored in his video library or is just to be recorded using this App. He may add pictures and additional metadata like a description, location-data or a phone number if there are questions.
- **Use Case: Record Video -** The user may record the video directly in the App for uploading it to the server.
- **Use Case: Registration Process -** The user may register directly in the App. During this process he will have to accept the usage terms once. In addition he has to provide the initial settings (ref. to use Case: Edit Settings). For verification purposes an email will be sent to verify the email address.
- **Use Case: Edit Settings -** The user can adopt which data may be sent to the publisher (e.g. geo-data), may enter specific groups (e.g. fire brigade, local sports club), change his communication settings. In the settings the user may also accept to be contacted for coverage-requests by the news publisher.
- **Use Case: Sign Up with existing account -** For users that already have an existing account but who log into the App for the first time, the user has to accept the usage terms and define his initial settings.
- **Use Case: Respond to coverage-request -** When a coverage-request from the publisher arrives the App will notify the user (The notification is just sent to the user if he agreed to this in his settings.). The user may read the request which automatically sets it to the status "read". This request gives the user the option to upload a video "in response" to this request which helps the news publisher to identify related videos.

6.3.2 Mobile App Administration Interface

The Mobile App Administration Interface constitutes the backend of the system. It allows the news publisher to manage users and user groups, to respond to uploads of videos (e.g. asking for more details, additional videos, pictures or similar) and to issue requests to user groups for coverages of events.

- **Use Case: Manage user groups** The administrator may create/edit/delete groups. A user group will have a name, a description and a secret key which is issued to specific users. With this secret key, users may join groups like the "fire brigade A", "lifesavers" or "sportsclub B".
- **Use Case: Send a coverage request** The administrator may send a note to groups, individual users or to users within a defined region to provide coverage of any event. This event is identified by the system and links all incoming videos with this request again.
- **Use Case: Manage users** The administrator may view and edit users - e.g. assign them to groups, disable users, change passwords and similar.

6.3.3 UGV-Upload-Server

The UGV-Upload-Server deals with all requests from the Mobile App and the Administration Interface. In addition, it is the interface to the video-storage-system and both curation system (webLyzard-Platform or APA-ContentHub).

The backend system also implements interfaces to authentication-services (ID-providers) provided by the news publishers usually used on their web-sites to have the same login for already registered users of the news portal. The following interfaces are provided by the backend system:

- interface to news publishers' authentication service
- interface to Mobile App
- interface to a APA-ContentHub or webLyzard-platform to store uploaded video documents (metadata) containing links to the video-file itself

- interface for verification service to provide details about the user (group)
- interface for contacting user for further details, comments, etc. used e.g. within the verification app or the curation tool
- interface for contacting users, specific groups or users in special areas
- mail-interface for verification of newly registered users (may be using the news publishers portal)
- optional: an interface to request details (keyframes, logos, annotations) from other InVid-services to display this data also in the curation tool (APA-ContentHub).

6.4 Multimodal Analytics Dashboard

The multimodal dashboard of InVID will provide access to the processed content streams from social media, enabling interactive exploration along various semantic dimensions - using multiple coordinated view technology for the desktop version and a cross-platform HTML5 application to access analytic function through smart-phones and other mobile devices. The dashboard will integrate a semantic search platform with the ability to track evolving stories, including participating actors (persons, organisations) and the relations among them. Users should be able to customise the representation according to their requirements, selecting the type of entities that are particularly relevant for their current task.

The dashboard's real-time synchronisation mechanisms will allow the tracking of information flows within the contextualised information space built in WP2. This will include the ability to display image and video content and use thumbnails to represent related stories and content clusters, thereby integrating visual content into the existing Web intelligence and knowledge co-creation workflow. This will require a number of core innovations - including (i) a "story" view as a new default representation to represent clusters of documents - including a lead article or posting, and related content from the chosen sources, (ii) extended interface representations to show and interact with InVID metadata elements, (iii) selection and storage mechanisms to provide the most relevant image content to be used as thumbnails in the dashboard and the UGC application, (iv) improved access mechanisms to ensure rapid response times for multimedia content, achieving response times similar to those achievable for text documents.

The result will both extend the capabilities of the current webLyzard platform product, as well as enable data visualisation widgets that can be integrated into the InVID applications created in WP6.

7 Software Development

7.1 Infrastructure and Hosting

At its core, the InVID platform utilises two interlinked data repositories that are used both externally as the basis for the developed applications, as well as internally as data provisioning services for the project partners. This design strategy is motivated to (i) reduce the overall platform complexity and to (ii) increase awareness of potential deficiencies in the platform by making the project partners to also be platform users. The two core data repositories of the InVID platform are (i) a FTP server to hold binary video data collected and enriched by the video analytics services and tools in the project, provided in form of the UGV-Upload server; and (ii) a relational database for storing document meta-documents, inclusive the various content acquisition channels (news media, social media, public video hosting platforms, etc.) set up for monitoring trending videos. Document upload to the meta-document repository will be provided via a REST API service.

- The UGV-Upload-Server will be hosted in the data center of APA-IT. For the **APA-Pilot** it will use the Video-streaming platform of APA-IT. The APA-ContentHub will also be run in the data center of APA-IT.
- The REST-based meta-document upload service and the relational database storage solution will be hosted and maintained by webLyzard technologies in collaboration with Modul University.

Both services contain mechanisms to create unique identifiers to be used in cross-referencing documents between the two data repositories. In addition to the two storage modules, each project partner hosts their own service endpoints and ensures high-availability as well as security maintenance in accordance with EU guidelines.

All services holding personal or any other sensible data will be encrypted where necessary and actively protected against unauthorised access and abuse.

7.2 Workflow Support

The development and maintenance of the InVID platform is made possible through a set of core infrastructure services provided by selected InVID partners to the platform that enable effective collaboration and data sharing between the project partners. The requirements for the shared InVID infrastructure have been set by the project partners to be (i) as simple as possible; (ii) containing as few service as possible; (iii) each service having a central responsibility in form of a dedicated project partner. In particular, the following infrastructure services have been agreed upon by the partners to be setup:

1. Wiki page (CERTH)
2. GitLab Code Repository and Issue Tracking (webLyzard)
3. Public Github Code Repository (Individually)
4. FTP Server (Condat)

All infrastructure services have been set-up and are in active use at the time of writing.

7.2.1 Wiki Page

InVID maintains an internal project Wiki to centrally collect and share team data such as meeting minutes, online and offline documents from both internal documentation and external publications. On top of being a team organisation and communication tool, the Wiki additionally provides transparency of the work undertaken for documentation and team synchronisation.

7.2.2 Gitlab Code Repository

A Gitlab repository for team-internal code sharing and issue tracking has been setup by partner webLyzard technologies at `gitlab.weblyzard.com` and respective user accounts were created. The code repository is being monitored and maintained for high availability and actively protected against external abuse via firewalls, encrypted disk space, and frequent software updates. Regular data backups are put in place to prevent potential data loss on hardware failure.

7.2.3 Public Github Code Repository

For open source exploitation of InVID components, the InVID partners will use their already established distribution channels on the code sharing platform *github.com*. Reuse of existing accounts guarantees optimal exposure of our work and maximises source credibility to the users.

7.2.4 FTP Server

For downloading and storing media resources (videos and images) a central FTP repository has been setup by partner Condat under `ftp.condat.de` and an invid user has been created.

7.3 Deployment Procedures

The InVID platform constitutes of a consortium of distributed software solutions developed and maintained by the various InVID partners as described above. Each partner in the platform is responsible to ensure the necessary quality and timeliness of their respective software components so to ensure the

Table 2: Selected system validation metrics and their validation measures.

Validation Metric	Validation Measure	Description
Acceptance Tests	Test-driven Development (TTD), Build Infrastructure	Checks alignment with requirement specification
Integration Tests	TTD, Build Infrastructure	Checks how well a software module integrates into an existing infrastructure
System Tests	TTD, Build Infrastructure	Checks the overall correctness of a software system
Usage Tests	UI Statistics, usage statistics	Checks if users use a system as intended (c.f. use cases)
Load Tests	TTD, Build Infrastructure	Checks the scalability capabilities of a service
Up-time Statistics	Monitor service health stats	Checks if services are available and sane
User Tests	Implemented through platform design: partners are users	Checks if data uploads to public toolsets yield expected results

correctness and timely deployment of cross-partner features and solutions. We strive to install continuous integration (CI) tools and workflows where such measures are not already in place. Deployment validation is to be achieved through the installation of a rigorous testing and build infrastructure, provisioning system tests, acceptance tests, and integration tests (also see 7.4). As a consortium, a high-level integration plan has been initiated to coordinate the dependencies between the work of the various InVID partners so to guarantee an optimal integration of the various solutions into the overall InVID platform during the life time of the project. The integration plan includes the planning and organisation of high-interval technical synchronisation meetings between involved parties surrounding scheduled deployments, so to minimise the risk of potential wait times between dependent components.

7.4 Validation Measures

In order to validate the correctness of the distributed software systems in InVID, validity metrics need to be designed for (i) all system components involved in the software system to be validated, as well as for (ii) all use cases associated with the software system. Validity metrics may be defined as software acceptance or integration tests, as quantifiable expectancy metrics (e.g. minimum service up-times, service qualities such as precision/recall, system user statistics), or as fail-over strategies (through mechanisms such as load-balancing, service discovery, horizontal and/or vertical node clustering). Scalability should be addressed as validation measure prior to development of software systems, to ensure that the produced systems are designed against the expected scope of the system's usage. Together with each validity metric, one or more validity measures should be identified that describe a series of steps to be taken in case a validity metric has not been met. The validity metrics and their corresponding validity measures will be reviewed as the project progresses, to incorporate feedback from the experiences gained and/or to integrate unexpected insights obtained due course.

8 Summary

In this document we have presented a roadmap for the design and development of key technologies from video processing to information visualisation towards an integrated video verification platform. The distributed nature of the InVID project across multiple project partners with varying expertise requires a clear deployment strategy for the final project product, the InVID platform for video verification. We devised a modular approach that encapsulates the promised software tools of all project partners as RESTful web services—accessible by all other partners. This architecture design reduces the interdependence of individual software components and thereby reduces the impact of failure a single component has on the complete platform. Three core applications act as data producers of

multiple formats (binary versus text) and of different source (user generated versus public crawls), and they control the flow of the data through the individual service components in alignment with the respective InVID use cases.
For each of the service modules, we have presented the technical status quo, together with initial specifications of the application programming interface to be used to integrate the module into the InVID platform. We have created an iterative deployment strategy with a first prototype release candidate scheduled for the last quarter of the year. All partners have agreed to strict software quality assurance measures in the form of continuous integration (CI) testing.
An InVID metadata model has been designed to integrate the various metadata information from all software components of the platform into a coherent structural representation to be used for visualisation. Two data repositories (binary and contextual) are being used to persist and query datapoints needed for online video verification by the platform. Data from both repositories are cross-referenced, so that all information extracted and collected by the platform can be retrieved for any video file.
To enable efficient collaboration between the project partners, we have setup a service infrastructure of core collaboration tools for reporting, software hosting, issue tracking, team communication, and team documentation. This infrastructure is in full use and already provides the basis for the project's development.

A Appendix

Table 3: Overview of planned InVID services

Service Name	Service Description	WP	Partner
webLyzard Visualisation	Embeddable Visualisations	WP5	webLyzard
webLyzard Search	Access Data from the Content Repository	WP5	webLyzard
webLyzard Annotation	Annotate Data in the Content Repository	WP5	webLyzard
webLyzard Statistical Data	Upload Statistical Data to the Content Repository	WP5	webLyzard
webLyzard Document	Upload Documents to the Content Repository	WP5	webLyzard
UdL Rights Management	Copyright Management	WP4	UdL
CERTH Image Forensics	Splice Detection in Images	WP3	CERTH
CERTH Near-Duplicate Search	Retrieval of Near-Duplicate Images and Videos from the InVID Index	WP3	CERTH
CERTH Contextual Verification	Assessment of Contextual Cues of Social Media Posts	WP3	CERTH
CERTH REST service for video analysis	REST service for video fragmentation and annotation	WP2	CERTH
CERTH Logo Detection	Module that detects logos (from a database of known logos) in videos	WP3	CERTH
EXO Video Forensics	Provide Results of Video Forensics Checks	WP3	ExoMakina
Video - Trust in Sources / Users	provide indication of trust in video form source/user	WP3	ExoMakina
APA-IT: UGC-Authentication	user authentication und authorisation service for UGC-App	WP6	APA-IT
Video-Verification	frame to be embedded for verification purposes in other applications	WP6	Condat
RightsManagement-Workflow	frame to be embedded for rights management purposes in other applications	WP6	Condat
Metadata-Annotation for Video	provide UG-videos and retrieve metadata for those videos (e.g. geo-data, ..)	WP5	Condat
MODUL Topic Detection	Determine timely, newsworthy news topics	WP2	MODUL
Social Media Mirrors	Extract for a topic relevant social media posts and store annotations in platform	WP2	MODUL

Table 4: Status of planned InVID services

Service Name	Status	API	Authentication	Service URL
webLyzard Visualisation	Alpha	REST	JSON Web Tokens (JWT)	www.weblyzard.com/api
webLyzard Search	Alpha	REST	JSON Web Tokens (JWT)	www.weblyzard.com/api
webLyzard Annotation	Alpha	REST	JSON Web Tokens (JWT)	www.weblyzard.com/api
webLyzard Statistical Data	Alpha	REST	JSON Web Tokens (JWT)	www.weblyzard.com/api
webLyzard Document	Alpha	REST	JSON Web Tokens (JWT)	www.weblyzard.com/api
UdL Rights Management	Development	REST	Basic Auth + HTTPS, but updatable to JWT	

Table 4: Status of planned InVID services

Service Name	**Status**	**API**	**Authentication**	**Service URL**
CERTH Image Forensics	Alpha	REST	TBD	http://reveal-mklab.iti.gr/reveal/verify.html
CERTH Near-Duplicate Search	Alpha	REST	TBD	https://github.com/MKLab-ITI/multimedia-indexing
CERTH Contextual Verification	Alpha	Other	TBD	https://github.com/MKLab-ITI/computational-verification
CERTH REST service for video analysis	Alpha	REST	Provision and check of user_keys	http://multimedia2.iti.gr:8080/shot-scene-concept
CERTH Logo Detection	Development	Other	TBD	
EXO Video Forensics	Development	REST	TBD	
Video - Trust in Sources / Users	Development	REST	TBD	
APA-IT: UGC-Authentication	Development	REST	TBD	
RightsManagement-Workflow	Development	Other	TBD	
Video-Verification	Development	Other	TBD	
Metadata-Annotation for Video	Development	Other	TBD	
MODUL Topic Detection	Alpha	No	TBD	
Social Media Mirrors	Alpha	No	TBD	

References

Apostolidis, E., & Mezaris, V. (2014, May). Fast shot segmentation combining global and local visual descriptors. In *2014 ieee international conference on acoustics, speech and signal processing (icassp)* (p. 6583-6587). doi: 10.1109/ICASSP.2014.6854873

Boididou, C., Papadopoulos, S., Kompatsiaris, Y., Schifferes, S., & Newman, N. (2014). Challenges of computational verification in social multimedia. In *Proceedings of the companion publication of the 23rd international conference on world wide web companion* (pp. 743–748).

Kordopatis-Zilos, G., Papadopoulos, S., & Kompatsiaris, Y. (2015). Geotagging social media content with a refined language modelling approach. In *Intelligence and security informatics* (pp. 21–40). Springer.

Markatopoulou, F., Mezaris, V., & Patras, I. (2015, Sept). Cascade of classifiers based on binary, non-binary and deep convolutional network descriptors for video concept detection. In *Image processing (icip), 2015 ieee international conference on* (p. 1786-1790). doi: 10.1109/ICIP.2015.7351108

Ojala, T., Pietikäinen, M., & Mäenpää, T. (2002). Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, *24*(7), 971–987.

Over, P., Awad, G., Michel, M., et al. (2012). TRECVID 2012 – An overview of the goals, tasks, data, evaluation mechanisms and metrics. In *Proc. of trecvid 2012.*

Scharl, A., Hubmann-Haidvogel, A., Sabou, M., Weichselbraun, A., & Lang, H.-P. (2013). From web intelligence to knowledge co-creation - a platform to analyze and support stakeholder communication. *IEEE Internet Computing*, *17*(5), 21-29.

Scharl, A., Weichselbraun, A., Göbel, M., Rafelsberger, W., & Kamolov, R. (2016). Scalable knowledge extraction and visualization for web intelligence. In *49th hawaii international conference on system sciences (hicss-2016)* (p. 3749-3757). IEEE Press.

Shang, L., Yang, L., Wang, F., Chan, K.-P., & Hua, X.-S. (2010). Real-time large scale near-duplicate web video retrieval. In *Proceedings of the 18th acm international conference on multimedia* (pp. 531–540).

Sidiropoulos, P., Mezaris, V., Kompatsiaris, I., Meinedo, H., Bugalho, M., & Trancoso, I. (2011, Aug). Temporal video segmentation to scenes using high-level audiovisual features. *IEEE Transactions on Circuits and Systems for Video Technology*, *21*(8), 1163-1177. doi: 10.1109/TCSVT.2011.2138830

Spyromitros-Xioufis, E., Papadopoulos, S., Kompatsiaris, I. Y., Tsoumakas, G., & Vlahavas, I. (2014). A comprehensive study over vlad and product quantization in large-scale image retrieval. *Multimedia, IEEE Transactions on*, *16*(6), 1713–1728.

Wu, X., Hauptmann, A. G., & Ngo, C.-W. (2007). Practical elimination of near-duplicates from web video search. In *Proceedings of the 15th international conference on multimedia* (pp. 218–227).

