Enhancing and Re-Purposing TV Content for Trans-Vector Engagement (ReTV)
H2020 Research and Innovation Action - Grant Agreement No. 780656

ReTV

Enhancing and Re-Purposing TV Content for Trans-Vector Engagement

Deliverable 4.3 (M36)

Trans-Vector Platform, Final and Optimized Version

Version 1.0

This document was produced in the context of the ReTV project supported by the European Commission under the H2020-ICT-2016-2017 Information & Communication Technologies Call
Grant Agreement No 780656

DOCUMENT INFORMATION

Delivery Type	Report
Deliverable Number	4.3
Deliverable Title	Trans-Vector Platform, Final and Optimized Version
Due Date	M36
Submission Date	30.12.2020
Work Package	WP4
Partners	WLT, GENISTAT, MOD, CERTH
Author(s)	Arno Scharl, Max Göbel, Katinka Böhm (WLT), Basil Philipp (Genistat), Kostas Apostolidis, Evlampios Apostolidis, Nikolaos Gkalelis, Damianos Galanopoulos, Elena Adamantidou, Vasileios Mezaris (CERTH), Lyndon Nixon, Jakob Steixner, Arlind Avdullahi (MOD)
Reviewer(s)	RBB, NISV
Keywords	System Architecture, Integration
Dissemination Level	PU
Project Coordinator	MODUL Technology, Am Kahlenberg 1, 1190 Vienna, Austria.
Contact Details	Coordinator: Dr Lyndon Nixon (nixon@modultech.eu) R&D Manager: Prof Dr Arno Scharl (scharl@weblyzard.com) Innovation Manager: Bea Knecht (bea@zattoo.com)

Revisions

Version	Date	Author	Changes
0.1	07.10.2020	Arno Scharl	Initial version based on D4.2
	18.11.2020	Lyndon Nixon	Started adding MODUL updates
	20.11.2020	Kostas Apostolidis, Nikolaos Gkalelis, Damianos Galanopoulos, Elena Adamantidou	Started adding CERTH updates
	25.11.2020	Basil Philipp	Started adding GEN updates
0.3	26.11.2020	Arno Scharl	Revision and extension
0.4	27.11.2020	Vasileios Mezaris, Evlampios Apostolidis, Kostas Apostolidis	Completed CERTH parts
0.5	27.11.2020	Lyndon Nixon	Completed MODUL parts
0.6	01.12.2020	Lyndon Nixon, Katinka Böhm, Jakob Steixner, Arlind Avdullahi	Added further details on Events and Anniversaries API, Prediction Service and Prediction API
0.7	08.12.2020	Arno Scharl	Revision and extension
0.8	11.12.2020	Lyndon Nixon, Kostas Apostolidis, Vasileios Mezaris	Further revisions and extensions to CERTH and MODUL contributions
0.9	18.12.2020	Rasa Bocyte, Miggi Zwicklbauer	QA
1.0	30.12.2020	Arno Scharl	Revision and final version
	30.12.2020	Lyndon Nixon	Final check and submission

Statement of Originality

This deliverable contains original unpublished work except where clearly indicated otherwise. Acknowledgement of previously published material and of the work of others has been made through appropriate citation, quotation or both.

This deliverable reflects only the authors' views and the European Union is not liable for any use that might be made of the information contained therein.

TABLE OF CONTENTS

GLOSSARY

Term	Description
Application Programming Interface (API)	Well-defined technical machine-to-machine interface for data exchange.
Electronic Program Guide (EPG)	A metadata feed of scheduling information for upcoming video broadcasts.
Semantic Knowledge Base (SKB)	A knowledge graph service jointly operated by MODUL and WLT.
Trans-Vector Platform (TVP)	An open architecture of components and subsystems that, in flexible configurations, enable a content owner or creator to optimise their content's impact and reach through contextualisation, prediction, and adaptation mechanisms.
Vector	A place where content can be published. This can be a social media channel like Facebook or Twitter or a more traditional channel like linear television.

Executive Summary

At the end of the third project year, ReTV has completed all planned extensions and revisions of the **Trans-Vector Platform (TVP)**. The final version of the TVP combines the technical results from the project partners into a comprehensive modular platform, enabling broadcasters, media archives and other digital content owners to tap into the next generation of digital content publication. The provision of prediction, personalisation and content transformation capabilities will help to **maximise the impact** of media organisations' communication activities and the **engagement** with their audience.

In response to stakeholder feedback received from the use cases, the target architecture has evolved both technically and conceptually into a comprehensive platform of loosely-coupled software components, organised along the three main functional capabilities: **listening**, **prediction** and **adaptation**. At the end of the third year all components have been integrated, evaluated and optimized, extending the functional capacity of the TVP in line with the continuously refined **use case** specifications. To validate the correctness and completeness of the TVP, we present **four TVP configurations** that showcase the added value of the TVP by solving industry limitations confirmed in stakeholder demonstrations.

The ReTV partners are committed to the continued **maintenance** and **evolution** of the platform and anticipate a strong demand for TVP-enabled services and applications. The achieved **modularity** and **flexibility** of the platform will be a major asset when adapting the products to changing **customer needs**, or when reconfiguring the workflow to rapidly deploy new services for yet unforeseen **business opportunities**.

1 INTRODUCTION

This document presents the final version of the **Trans-Vector Platform** (TVP) at M36, building on the initial prototype reported in Deliverable D4.1 (M12) and the extended version of Deliverable D4.2 (M24). The TVP is a system of modular components that can be used together in different configurations by digital content owners to recommend and repurpose their media semi-automatically and distribute it across multiple publication vectors. It combines the technical outputs from the ReTV project, composed as a system of independent, distributed modules that communicate with each other through a set of **Application Programming Interfaces (APIs)**. The end user accesses the functionalities of the TVP through applications that access these APIs and bundle the relevant TVP components together. The applications address distinct use case scenarios developed and refined by the use case partners of the consortium (see Sections on "TVP for Professional Scenarios" and "TVP for Personalization").

The main focus of WP4 lies in the **architecture design** and **integration** of software components as developed by the technical consortium partners towards a working platform that addresses and solves the user needs as elaborated by the use case partners. Since the last reporting in M24, the ReTV consortium has extended and continuously refined the **four applications** (*Topics Compass, Content Wizard, 4u2 Messenger* and *Abendgruss*) based on feedback of the use case partners as well as selected third parties.[1] The four scenarios use TVP components in different configurations, relying on three types of technical components:

1. **Individual Web Services** delivered by WPs 1-3 that can operate standalone via REST APIs, or as part of an integrated content processing workflow. Those components provide data management (collection, annotation), predict usage patterns (based on events, Web and social media metrics, TV audience figures) and analyse and repurpose digital media for recommendation and (re)publication across vectors.

2. **Integration Components** to combine individual services into a workflow and a data mapping between the data generated and consumed by distinct components (as delivered by WP4). The chosen set of integration choices defines the configuration of multiple components into a TVP instance.

3. **TVP Applications** for supporting different use cases, including scenario-specific user interfaces on top of TVP components, in order to provide access to TVP data and services to both professional users on the media content owner side as well as content consumers on the public side (WPs 5 and 6 have defined, designed and evaluated a set of scenarios that the TVP should support).

In this deliverable, we present the TVP as a fully integrated and optimized portfolio of software components (see **Appendices A-C**), ready to be utilised by the TVP applications. Extensive testing and iterative improvements have considerably improved the **reliability** and **scalability** of the TVP platform, as compared to the M24 prototype.

The goal of Deliverable D4.3 is to provide comprehensive, consistent documentation that does not require readers to access and read previous deliverables. Since this inevitably leads to a certain degree of overlap with the preceding Deliverable D4.2 (M24), the next section summarizes the key innovations of Year 3 to highlight the extensions and improvements implemented between M25 and M36 of the project.

[1] See deliverables D5.3 and D6.3 for detailed descriptions of the professional user and consumer scenarios, respectively, including the evaluations conducted in Year 3 of the project.

1.1 Key Innovations in the Third Year of the Project

Structured into one subsection for each technical partner, the following overview summarises the main innovations in the third year of ReTV (cf. the Appendices for the details specifically organized according to the components of the TVP), while Sections 2 ff. then present the latest M36 version of the platform, serving as a TVP reference covering the system architecture as well as individual services and their integration to support specific use case scenarios.

WEBLYZARD

- **REST API Framework** (see Section 3): Major extension to support the integration of TVP Services, compatible with new developments of the technical WPs 1-3, incl. support of predictive analytics and full integration of the ReTV data ingestion pipeline into the core platform.

- **TVP Visual Dashboard** (see Section 4.1): In addition to frontend adaptations to support a fully reworked content processing workflow (keywords, entities, lexemes), improving the platform's contextualisation and disambiguation capabilities, the final *Topics Compass* release supports predictive analytics expanded to more content sources. It offers a selector to use published dates or referenced dates to anchor the documents contained in the search results along the temporal axis. This enables the display of predicted topics, as referenced dates can point to both past and future events. Year 3 development work also yielded a new mobile client, enabled dashboard localisation and improved user interaction design based on feedback from the use cases.

- **Visualisation Components** (see Section 3.2.4)**:** (i) Extension of all charts to support multi-colour codings, to support e.g. the opinion mining extension to classify content according to Plutchik's *Wheel of Emotions*; (ii) Extension of all charts to support prediction mode; (iii) updated WYSDOM Chart to ensure its generic applicability, using standard bookmark / attribute colours instead of specific desired/undesired scheme; (iv) revised *Story Graph* to improve label quality and layout.

- **Widget Embedding** (see Sections 3.2.4 and 4.2): To support a generic Visualisation-as-a-Service (VaaS) approach, embeddable versions of (i) various visualisations including the *Story Graph,* the entire mobile client (see previous bullet) and the Storypact editing environment have been developed.

- **Scalable Infrastructure** (see Section 2, particularly 2.2.3 and 2.3): Year 3 optimisations helped to transform the TVP services from early proof-of-concept configurations into a commercially viable technical platform, with particular focus on high availability and rapid scaling out capabilities. This process included a migration to Elastic 7.10 and CockroachDB 20.2, batch data ingestion, improved knowledge graph caching and the development of a term cache (e.g. synonyms or third-party metadata attributes).

MODUL

- **Final Full End-to-End Data Integration Pipeline** (Data Collection, NLP, Semantic Annotation) (see Section 2.3.1 and Appendix A)
    - Means for adaptation to new domains
    - Data Source Categorization
    - Expanded sources for stakeholders (input collection via Sheets)
    - Addition of NLP/NER support for Dutch

    - Improved Keyword-Entity Disambiguation using the SKB and Label Translations (e.g. TermEntities)
    - Improved Keyword and Entity Detection in Text, with a focus on Works (e.g. TV Programme Brands)
- **Final Semantic Knowledge Base (SKB)** (see Appendix A)
    - expanded Event collection (input collection via Sheets)
    - broader support for Person entities (born/died < 1900)
    - categorization of Events
    - inclusion of Event descriptions
    - more granular Location entities (street level, from OSM)
- **SKBBrowser > SKBEditor** (see Appendix A)
    - Added advanced search filter
    - Edit capability on all properties
    - Added 'add new entity' functionality
- **Events and Anniversaries API** (see Sections 3.3.1 and 3.3.2)
    - Support for scenario filters
    - Extended metadata response
- **Final Hybrid Prediction Model** (see Sections 2.3.2 and 3.5.1)
    - Time Series Forecasting of Future Topic Popularity
    - Final Temporal Reference Detection from Text
    - Addition of Event and Temporal Reference Features
    - Prediction API for finding the best topic for a date or the best date for a topic

CERTH

- **Video Fragmentation and Annotation** (see Appendix A)
    - More accurate DNNs based on, among others, network pruning techniques, to improve accuracy and computational efficiency of video annotation
    - New Sandmannchen video analysis methods (domain-specific fragmentation and annotation)
    - Software optimization for enabling ingesting large video files (>2h, to be able to ingest full-length movies)
- **Brand Detection** (see Appendix A)
    - New car brand detection functionality
    - Introduction of merged objection detection model to improve processing time and memory footprint
- **Video Adaptation and Repurposing** (see Appendix C)
    - Introduction of learning-based video summarization, using Generative Adversarial Networks
    - Joint consideration of editor-specific rules (supported by video annotation results) and learning-based summarization for generating video summaries
    - Support for different video summarization profiles (e.g. Movie, News-in-Film)
    - Smart-cropping for adapting video summaries to different aspect ratios
    - Software optimization for speeding-up the summary generation process (e.g. multi-threaded read and decode workflow)
- **Text-to-Video Embeddings Extraction** (see Appendix C)
    - Extraction of Text2Video embeddings from video, using a dual encoder attention network, integrated in the Video Analysis service
    - New REST endpoint for real-time extraction of similar Text2Video embeddings from text

GENISTAT

- **Content Wizard use case** (See Section 4.2)
  - Scalability: Storing vector embeddings of CERTH API in Elasticsearch indexes to allow for efficient querying
  - Built a new search API on top of the CERTH API for Text2Video search including a background process that processes new videos
  - Scheduler: Automatic daily uploads of RBB / NISV content into Content Wizard
- **4u2 Messenger use case** (See Section 5.1)
  - Scalability: Using Faiss library to allow for performant recommendations using the vectors from CERTH
  - Rebuilt the 4u2 Messenger with a better Chatbot framework that allows for easier control of the conversation flow
  - Scheduler: Built a scheduling and deliver process for the messages of the 4u2 Messenger
- **Abendgruss use case** (See Section 5.2)
  - Built a backend to allow the processing of episodes through the CERTH API and for editors to set descriptions
  - Scalability: Create m3u8 streams of Abendgruss episodes, which allows for real-time creation and delivery of personalised episodes.

1.2 Deliverable Structure

The remainder of D4.3 starts with a technical overview of the TVP in Section 2, incorporating updates from all technical work packages where applicable. Section 3 then describes the integration and optimization efforts undertaken to bundle the various technical components into a holistic, coherent toolkit based on specified data exchange formats and interfaces. Section 3 also introduces processes for platform evaluation that allow us to assert the validity, correctness, and completeness of the TVP integration. Finally, Sections 4 and 5 demonstrate the use of the TVP by highlighting the technical integration of the two use case scenarios, *Engagement Monitoring* and *Personalisation*, as defined in WP5 and WP6, respectively. As such, Section 4 discusses the TVP as a back end for the TVP applications *Topics Compass* and *Content Wizard*, and Section 5 highlights the TVP as a back end for the TVP applications *4u2 Messenger* and *Abendgruss*. The *4u2 Messenger* is a newly developed use case based on the research done for the *Content sWitch*. We have decided to focus our efforts on the 4u2 Messenger, as we believe it has a more substantial potential for addressing the needs of our stakeholders than the *Content sWitch*. As we described in the relevant paper[2], both the Content sWitch and the Messenger use the same core recommendation and scheduling engine. Therefore, the *4u2 Messenger did not* require changes to the TVP architecture. We conclude this deliverable with how the TVP will be sustained and maintained after the project ends, in line with the business and exploitation plan (D7.4).

During the reporting period M25-M36, three WP4 tasks were active. Progress of Task 4.1, *System Architecture and Integration* is reported on in Sections 2, *Trans-Vector Platform*, and 3, *Platform Integration And Evaluation*. Task 4.2, *Visualization of Content and Audience Flows*, is covered in Sections 3.2, *Metadata APIs*, and 4.1, *Topics Compass*. Progress on Task 4.3, *TVP Visual Dashboard*, is reported in Section 4.1, *Topics Compass*, which is the ReTV brand name

[2] B. Philipp, K. Ciesielski, and L. Nixon. Automatically adapting and publishing tv content for increased effectiveness and efficiency. In Proceedings of the 1st International Workshop on AI for Smart TV Content Production, Access and Delivery, pages 51–52, 2019.

that has replaced *TVP Visual Dashboard*. Finally, Task 4.4, *Scalability and Distributed Content Processing Strategy*, is reported in Sections 2, *Trans-Vector Platform*, and 3, *Platform Integration and Evaluation*.

2 Trans-Vector Platform

The TVP is the back end system that enables all applications as identified and specified in the use cases WP5 and WP6 - to be reported in D5.3, *Second Validation of Engagement Monitoring Prototype*, and D6.3, *Second Validation of Personalization Prototype*.

2.1 Overview and Objectives

The content industry faces a diversification of its audience across a growing set of media vectors, while at the same time lacking the tools to efficiently adapt its content to these new publication vectors for optimised impact. The TVP addresses the problem that adapting content for a growing set of publishing vectors is still a manual, labour-intensive task (c.f. Listing 1, excerpt from the ReTV Description of Work).

> ReTV aims to provide **broadcasters** and **content distributors** with technologies and insights to leverage the converging **digital media landscape.** By advancing the state of the art in the **analysis** of this media landscape and providing novel methods to dynamically **re-purpose** content for an array of **media vectors** (= all relevant digital channels), a **Trans-Vector Platform (TVP)** will provide these stakeholders with the ability to *"publish to all media vectors with the effort of one".* It will empower broadcasters and brands to continuously **measure** and **predict** the **success** of their content and advertisements in terms of **reach** and **audience engagement** across vectors, allowing them to optimize **decision-making** processes.

Listing 1: Objective of the TVP, as described in the ReTV Description of Work.

In ReTV, the automation of content repurposing for cross-vector publication has been aligned within three distinct capacities:

- **Listening** - for an existing TV show, monitor audience feedback and the public online debate. This helps to contextualise content publication and transformation as a central KPI (success metric), and to repurpose archival content, e.g. to provide historical context about a particular topic currently discussed online.
- **Prediction** - for current world events and trends, predict the most relevant content for publication, at the best time (scheduling), and along the highest-impact vector.
- **Adaptation** - for an existing TV format, allow fully-automatic content transformation for various audiences (personalisation) and different publication vectors.

The TVP as described in this deliverable is a flexible toolkit of individual software components developed within WP1, WP2, and WP3, respectively, integrated into a closely-knit mesh of interconnected TVP subsystems that address the very core capacities outlined above. Through its flexibility − achieved by a loosely-coupled microservice architecture of single-functional components with strictly-defined interfaces − the TVP allows for a multitude of functional instances to be realised that address the above-mentioned core capacities. In the scope of the ReTV project, we have defined four use case scenarios for which applications are being developed. The chosen use case applications are guided by the project's content partners, and have integrated multiple feedback rounds from industry stakeholders received during the first two years of the project. With the TVP (in the form of the professional scenarios) a media

professional can maximise the impact of their media content publication guided step by step by AI-powered prediction and recommendation functionalities, without additional manual effort or any technical background required.

2.2 Platform Architecture

WP4 is concerned with the overall architecture of the TVP as well as the integration of the individual components into coherent, scalable, and adaptive workflows that optimally reflect the functional requirements of the ReTV applications and thereby best address the use case scenarios from WP5 and WP6, respectively. Over the course of the last two years, the TVP architecture has evolved into a tightly-integrated mesh of components that supports rapid prototyping of new integration ideas from user testing in the use case scenarios into the ReTV applications. This flexibility is achieved by strict adherence of all technical consortium partners to the microservice-based TVP architecture, where each TVP component is required to have a well-defined API and must be limited to a fixed number of related functionalities.

2.2.1 Technical Innovation

The core motivation behind the TVP is an industrial need for an open toolkit of easy-to-use tools for the content creator and scheduler to optimise the reach and impact of video content to be broadcast across an ever-growing set of publication vectors. The TV industry is strongly relying on rigid and closed systems to organise video content creation and publication. With the TVP, we introduce an open and flexible architecture for the media industry that opens new possibilities to connect and integrate with existing and new components and thereby to freely accommodate for new uses as they arise in the future.

The TVP follows a modular design to allow for the highest flexibility in terms of component reuse across the various user-facing TVP applications. Its modular architecture is the technical realisation of the bold vision of reinventing TV, allowing for a reorganisation of modules akin to a toolkit to best match a wide variety of real-world TV content publication scenarios. It further facilitates the commercial dissemination of the TVP beyond the lifetime of the ReTV project by mitigating the risk of blockage of individual TVP components in larger, integrated deployments, with each component of the TVP being fully operational in isolation as a containerised micro application, independent of existing code bases.

Where possible, *application state* should be achieved via single-purpose data storages (both for binary and metadata content), and possibly equipped with full CRUD (Create, Retrieve, Update and Delete) API access. The microservice components operate stateless on the content streams. The microservice-based TVP architecture is shown in Figure 1, where the integration is particularly apparent in the many data flows that cross between the metadata workflows on the left of the figure, and the binary data workflows on the right.

2.2.2 Data Flows

From a technical perspective, the TVP is driven by two central processing workflows: (i) a *binary processing workflow*, where TV video (binary) content is ingested, analyzed, transformed, and the results pushed into the TVP applications *Content Wizard*, *4u2 Messenger*, and *Abendgruss*. The binary workflow is the backbone of content repurposing in ReTV; and (ii) a *metadata processing workflow*, where TV content and news metadata is ingested, integrated, analyzed, and the results pushed into the TVP applications *Content Wizard*, *Topics Compass, 4u2 Messenger*, and *Abendgruss*. The metadata workflow enables prediction via contextualization and alignment of binary data with the public debate.

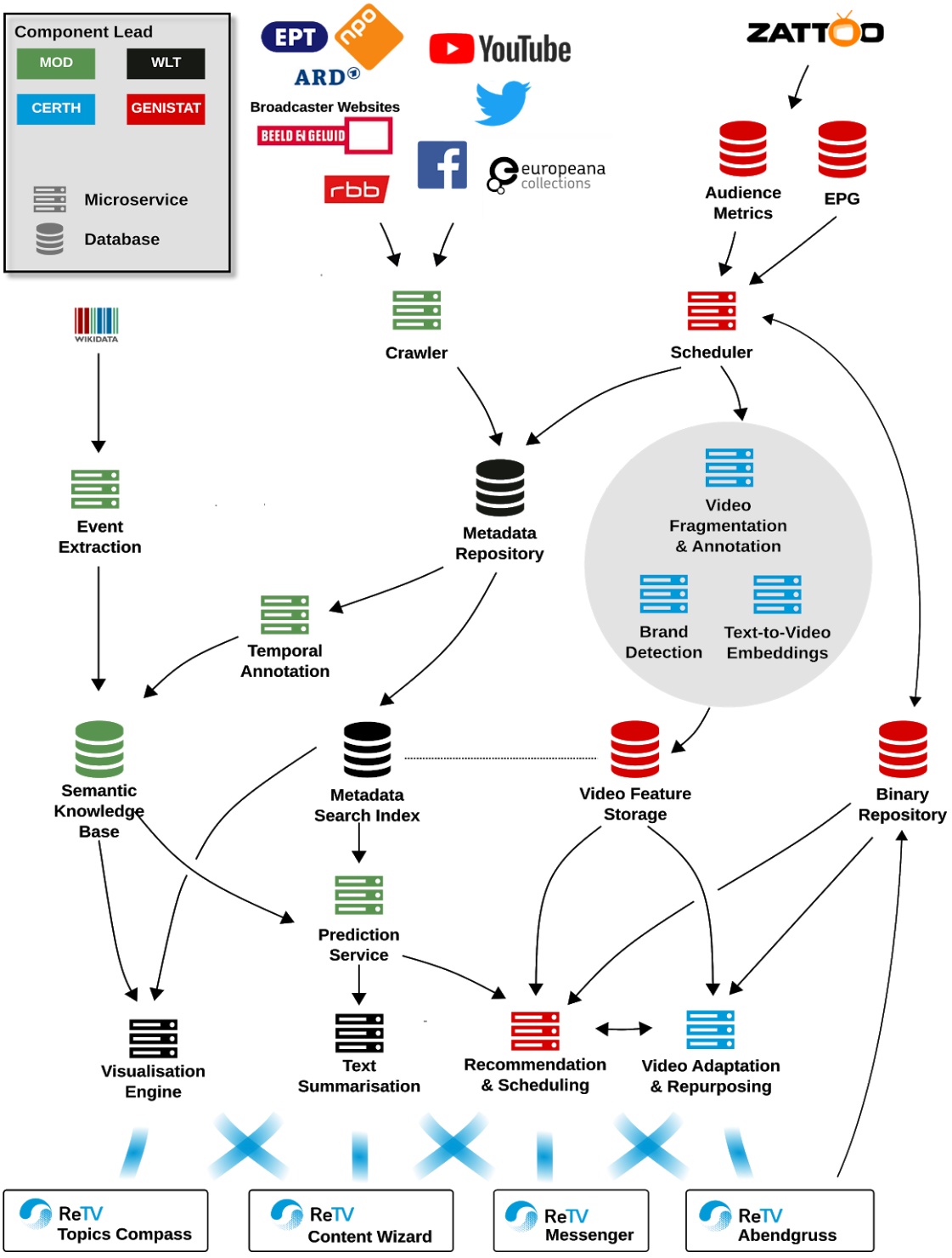


Figure 1: TVP architecture as a modular design of microservices, controllers and databases, colour-coded by the lead developer of each component.

Both workflows have data flow from raw formats towards enriched or transformed formats in a one-way direction. The two main workflows along the various TVP components into the applications are shown in Figure 1. In this figure, the core components of the TVP are described as either a microservice or a database component. A *microservice* here is an enrichment or transformation service that does not maintain state, and is therefore by design

horizontally scalable. A *database* represents any form of persistent storage component, generally also providing APIs for retrieval or ingestion. The left half of Figure 1 illustrates data flows for metadata (textual formats), whereas the right half of the figure shows the data flows for video data (binary format). Both data flows start with data retrieval from either public (Web, YouTube, Twitter, Wikidata, etc.) content providers or from our commercial industry partner Zattoo. As the data items flow downwards through the TVP, the data is continuously refined into representations and patterns that make up the statistical support which in turn drive the insights as required by the TVP applications, shown at the very bottom of the same figure.

2.2.3 Platform Scalability

The TVP was designed and built with scalability in mind. In the TVP architecture, scalability considerations show in the microservice architecture, in the choice of database systems, and in the distribution of persistence and caching layers throughout the TVP. The system state of the TVP is maintained in the seven database systems that are distributed across the architecture, holding varying levels of data abstractions. The microservices are stateless and can be scaled horizontally as the data flow requires. All components interact via APIs, with all APIs having well-defined functional scopes and a consistent payload abstraction. Where possible, APIs maintain buffers of incoming requests, while processing their workload asynchronously via messaging queues. This design principle applies to the CERTH Video Fragmentation and Annotation, Text-to-video Embeddings and Brand Detection microservices (which are collectively known as the Video Analysis service), the CERTH Video Adaptation and Repurposing service (aka the Video Summarization service), all MODUL data ingestion services, as well as the GENISTAT scheduler service for data pre-aggregation from Zattoo.

TVP Operations

The operation of the TVP is de-centralised, with every technical partner of the consortium overseeing all operations of their respective TVP contributions. The partners synchronise on best practices of service operations in the monthly WP4 meetings to guarantee an optimal knowledge transfer on operations, including the orchestration and monitoring of each component. Also, all technical partners of the ReTV consortium provide their services fully containerised through *Docker*, allowing for fully-managed service provisioning via orchestration frameworks such as *Kubernetes* or *Portainer*. *Load Balancing* and *Service Discovery* are used both on microservice and database level where fitting.

Software choices for persistence layers were made in favour of high-availability systems that are distributed across large hardware setups with high resilience to failure. In particular, at consortium partner WLT, we employ large Linux-based hardware clusters to run distributed, horizontally sharded CockroachDB[3] and ElasticSearch[4] systems as backbones to the metadata repositories. Similar setups are operated by Genistat for binary repositories.

TVP Monitoring

Multiple levels of monitoring in the TVP exist, each addressing a different class of failures: Hardware monitoring of the Metadata Repository and APIs as well as all prediction services are monitored via the *Open Monitoring Distribution*[5] (OMD) together with its flexible *CheckMK*[6]

[3] https://www.cockroachlabs.com/
[4] https://www.elastic.co/
[5] https://omdistro.org/
[6] https://checkmk.com/

extension. This monitoring covers all potential hardware failure scenarios and allows for quick recoveries via battle-proven alerting policies. On top of hardware monitoring, software-level monitoring is employed via *Prometheus*[7] and *Logstash*[8], with *Grafana*[9] and *Kibana*[10] analytical front ends for detailed insights and alerting. Alerts are sent using the Opsgenie platform[11]. Front end application monitoring is achieved via *Sentry*[12] application level monitoring as we use Logstash*/Kibana* for usability and access monitoring.

2.3 TVP Components

Name (Lead)	Description
Semantic Knowledge Base (SKB) including an SKB Browser (MOD)	An RDF triple store (Apache Fuseki) containing entities and their descriptions as a knowledge graph. Includes the event entities from the event extraction service. This now includes a Web-based interface to browse and directly edit entity descriptions in the SKB.
Metadata Repository (WLT)	A distributed, high volume, high performance persistence layer for fully integrated, cross-vector metadata items.
Metadata Search Index (WLT)	A fast and expressive ElasticSearch index with a rich set of functional API abstractions.
Audience Metrics (GENISTAT)	Audience data from Zattoo. Consists of aggregated real-time data and detailed viewing sessions. Metrics are anonymised to preserve user privacy. Data is being pushed regularly into the Metadata Search Engine.
EPG Repository (GENISTAT)	Program metadata for the relevant TV channels, regularly pushed into the Metadata Search Engine.
Binary Repository (GENISTAT)	A distributed storage solution that holds video files for 10 TV channels since the beginning of 2019. Provides a REST API for search and retrieval of binary content.
Video Feature Storage (GENISTAT)	The features that CERTH video analysis service extracts from video are stored in a database for quick later retrieval without the need for recomputation. The video feature storage has been extended with a powerful search based on ElasticSearch, allowing to find videos based on their Text-to-Video Embeddings. It is described in detail in Section 3.3.1 of D3.3.

Table 1: Overview of storage components of the TVP architecture.

[7] https://prometheus.io/
[8] https://www.elastic.co/products/logstash
[9] https://grafana.com/
[10] https://www.elastic.co/products/kibana
[11] https://www.atlassian.com/software/opsgenie
[12] https://sentry.io/welcome/

Similarly, Table 2 lists all ten microservices in the TVP, describing their respective inputs and outputs, as well as denoting their respective development lead from the consortium.

Name (Lead)	Input	Output
Temporal Annotation (MOD)	textual content from Web pages or social media posts	detected temporal references (position of date-time values)
Event Extraction (MOD)	event sources: WikiData, iCal files and CSV files	event instances and their descriptions in RDF
Crawler (MOD)	list of Web sites or social media channels or (regexp) terms	documents each representing one content item (Web page, social network post) and its description
Prediction Service (MOD)	events, documents, time series success metrics from the documents, audience metrics	prediction of a value at a future time (success metric for a topic on a certain vector, topic for a success metric on a certain vector)
Events and Anniversaries API (MOD)	date (range)	a query endpoint for future anniversaries and events based on the entity descriptions in the SKB
Video Fragmentation and Annotation (CERTH)	videos	video keyframes, video temporal segmentation information, keyframe concept annotations, supporting signals for subsequent components
Text-to-Video Embeddings (CERTH)	video keyframes or text strings	Text-to-Video Embeddings to support video search and text-query-driven summarization
Brand Detection (CERTH)	video keyframes	brand/channel logo annotations on keyframes
Recommendation and Scheduling (GENISTAT)	video segments from the binary repository, metadata features from the video feature storage, topics from the prediction service	a linear sequence of potentially adapted video segments tailored to the personal preference of the viewer
Video Adaptation and Repurposing (CERTH)	features extracted by the Video Fragmentation and Annotation component	summary script (segments of the original video that constitute a meaningful summary) or the summary rendered to video files (mp4, webm and gif), which can be retargeted to a specific aspect ratio using a smart-cropping technique.

Table 2: Overview of enrichment components of the TVP architecture.

Across the consortium, the TVP is composed of nine database components and ten microservice components. All services provide RESTful APIs as an interface for data exchange. Within the consortium, we maintain documentation of all services in the shared GitLab instance we operate for technical communication between project partners. Table 1 gives a short overview of all storage components from the TVP, also denoting their respective development lead from the consortium.

2.3.1 Content Aggregation and Alignment

WP1 retrieves video content and metadata context from a wide set of commercial and public resources. From *Genistat,* in association with industry partner *Zattoo*, we retrieve Electronic Program Guide data (EPG) as metadata and associate it with binary video files extracted on demand from IP-based TV streams. This correlation of metadata to binary content via EPGs is complemented by indirect audience monitoring through listening to the online discourse surrounding TV programming and its topics both along news and social media vectors.

1. **TV Show content**: the broadcast video content as provided by industry partner Zattoo, together with metadata information extracted from it.
2. **Public metadata content on TV shows**: this includes descriptions and reactions online, as well as the broadcaster's schedule. It is retrieved by the *MOD crawler* from publicly accessible Web sources such as social media APIs and Web sites published by TV producers.
3. **Public metadata content from Europeana**: this relates to the collection of video content from consortium partner NISV that is being employed in the use cases. The content is accessible on the Europeana platform[13], from which the metadata descriptions are aggregated for special use case evaluations.
4. **Event data related to TV shows**: information on what event a TV show relates to. Examples of events are the FIFA World Cup or a royal wedding. This data is aggregated by the *MOD Event Extraction* component in structured format from the public Wikidata repository.

Metadata Aggregation and Alignment

The content aggregation dataflows required by the engagement monitoring scenarios for professional users as described in D5.2 (WP5) are summarized in Figure 2. We collect data from relevant Web sites as well as from public social media platforms that are related to the topics of interest (to a media organisation as stakeholder) as defined for the *scheduler* component. Details on the final form of the content aggregation can be found in WP1 Deliverable D1.3, *Data Ingestion, Analysis and Annotation, Final Version*.

The *MOD Crawler* scrapes different data sources for information relevant to ReTV. Its targets are social media, Web sites and archives and it can be configured to listen for certain keywords through "term lists" or monitor for new content on Web sites or social network accounts through "channel lists". It can, for example, listen to keywords or hashtags like *#Tatort*, which usually accompany posts on the popular German-language crime TV show. It analyses and annotates all Web and social media content found matching its term and channel lists and writes this content metadata into the *Metadata Repository*.

[13] https://www.europeana.eu/portal/en

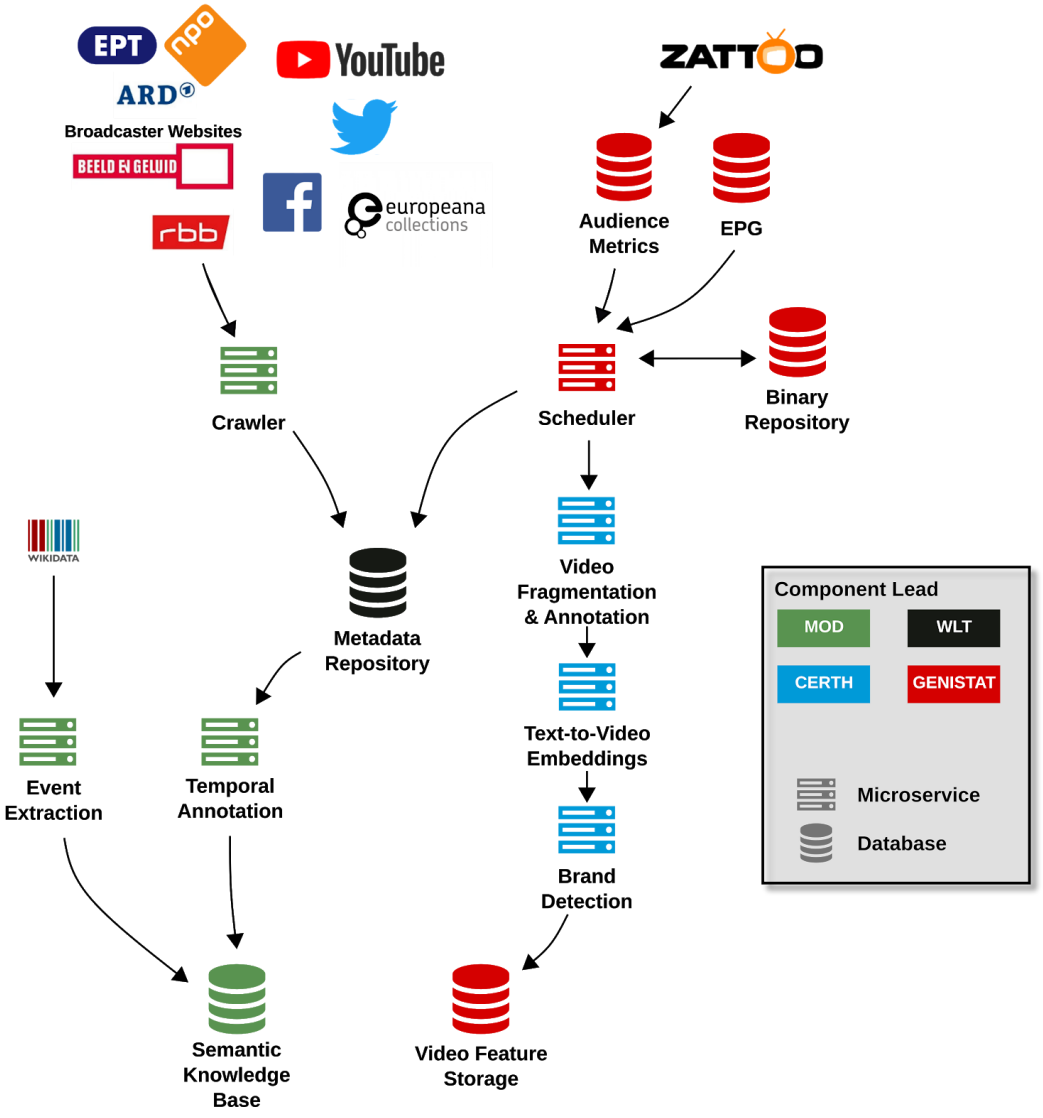


Figure 2: Content Aggregation for Engagement Monitoring (Topics Compass, Content Wizard), from data source (top) to the internal TVP components *Metadata Repository* and *Semantic Knowledge Base*.

Another source of content is the Electronic Program Guide (EPG) data that Genistat receives from Zattoo (stored in the *EPG* database). After it is enriched with metadata and start/end times have been added, the *Scheduler* pushes it into the *Metadata Repository* at regular intervals. Consortium partner Genistat also provides audience statistics on selected EPG publications to enhance the analytical capabilities of the prediction components of the TVP.

Binary Aggregation and Alignment

The source for the raw video material is Zattoo. It is stored in the *Binary Repository.* In compliance with copyright laws, this video data is treated as a private recording and is therefore not redistributed without the explicit approval of the copyright holder.

The *Video and Fragmentation and Annotation* component splits the videos into meaningful temporal fragments, selects some representative frames (keyframes) for each fragment, annotates the keyframes with concepts and extracts a set of additional auxiliary features to support the services of WP3. The *Brand Detection* component extracts information on brand logos detected in keyframes. It also performs advertisement detection. The *Text-to-Video*

Embeddings component extracts text-to-video embeddings from video's shots keyframes as well as text strings, to support video search and text-query-driven video summarization in subsequent components. The metadata results from all three components discussed in this paragraph are stored in the *Feature Storage* of ReTV (they include temporal segmentation data, concept annotations, text-to-video embeddings and other supporting signals).

The latest status update of WP1 components is compiled in Appendix A.

2.3.2 Content Enhancement and Predictive Analytics

The second major subsystem within the TVP is formed by those components that enable the predictive scenarios of the ReTV use cases (c.f. WP2). They incorporate the database systems *Semantic Knowledge Base* and *Search Index*, as well as the microservice component *Prediction Service*. This prediction subsystem of the TVP is highlighted in Figure 3. The *Semantic Knowledge Base* (SKB) stores all the information on relevant entities for the syntactic and semantic annotation of the crawled content. As a knowledge graph of entities and relations, it supports the accuracy of keyword and entity extraction from the content metadata by modelling synonyms, alternative written forms, lemmas, as well as entity relations (for graph-based disambiguation of entities in MOD's Named Entity Recognition tool *Recognyze*). New and updated APIs enable the *Prediction Service* to use both events and anniversaries identified in the SKB according to a stakeholder's requirements as well as the metrics extracted from the Metadata Search Index for both temporal reference detection and time series forecasting.

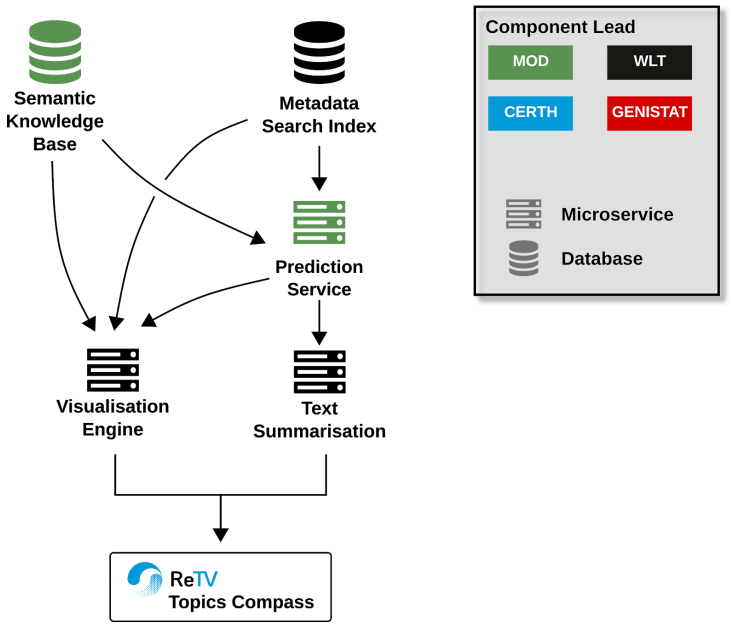


Figure 3: TVP subsystem to enable predictive analytics.

Prediction Service

WP2 delivered D2.3, *Metrics-based Success Factors and Predictive Analytics, final version* in month 30. The *Prediction Service* presented in this deliverable incorporates the following prediction models:

1. Prediction of audience metrics based on past audience and including content and event features for improved accuracy. Input: past audience, content categorization and events per TV show/channel; Output: future audience per TV show/channel.
2. Prediction of success metrics for content based on past success metrics. Input: past success metrics per topic; output: future success metric per topic.
3. Prediction of future keyword popularity based on the *Temporal Annotation* component. Input: keyword or topic; output: success metric for that keyword or topic on future dates.
4. Prediction of future events of relevance based on the *Event Extraction* component. Input: future date; output: list of events on that date.

The *Temporal Annotation* and *Event Extraction* components produce data which can be used to interpret knowledge about the future, whether that is the keyword(s) likely to have a peak in popularity on a certain future date or the event(s) which are likely to be occurring on a future date. As such, both now provide APIs which are used by the Content Wizard tool to expose this knowledge to end users. The *Prediction API* recommends the optimal time for publication of content related to a certain topic based on the predicted peak in usage of the topic keywords within a given future date range. The *Event and Anniversaries API* returns a list of events and anniversaries on a given future date matching a stakeholder's specified interests (configured as a template on the API side).

Since D2.3, we have completed our research and development on predictive analytics, including how to achieve the optimal accuracy for predicting future content success both alone with time series forecasting and through a hybrid model incorporating additional features with knowledge about the future.

Following the linear regression and ARIMA approaches reported in D2.2 and D2.3 respectively, we chose to compare our selected best performing model (SARIMAX(7,1,2) with annual seasonality) with the use of a neural network for time series forecasting. In the scientific literature, LSTMs (Long Short Term Memories) have been reported as being particularly effective for the forecasting task.

Taking a "vanilla" LSTM as the baseline (single layer, 50 neurons, relu activation) we can train it on input-output sequences of length (7,1), i.e. use the last week of data to predict the next day. Multi-step predictions are handled using the same autoregressive approach we used with linear regression or ARIMA, so equally can not take into account longer term trends or seasonality in the data (the improvements in prediction when using SARIMA, which included an annual seasonality component, suggests seasonality can indeed be useful when the keyword/topic is present throughout the year in the data source and can be expected to demonstrate some regular variations, e.g. sports which peak during some specific sports events in the year). We also tested with some variations in LSTM configurations (using the Encoder-Decoder model and/or Bidirectional LSTMs) as well as the TimeSeriesGenerator for modelling the training and test data in an explicit time series model. While these variations did improve the forecasting accuracy compared to the vanilla model, the LSTMs were still less accurate than our best performing ARIMA model.

We wanted to perform multi-step forecasting directly from the prediction model whereas TimeSeriesGenerator only allows single step forecasts and therefore autoregression needs to be used. For this, we looked at seq2seq models which were also being reported in the literature as the best performing LSTMs for forecasting. Considering current research, we tested with the addition of attention mechanisms (Luong attention) which further improved the accuracy. Finally, we also experimented with different activation functions and found a

recent function proposed by Google engineers called Swish was best for our prediction task. So our final LSTM model, based on comparison with different configurations of accuracy with different datasets (keywords), is an Encoder-Decoder model with input-output sequences of length (200, 30), using Luong attention and Swish activation function, trained for 50 epochs and validated with test data using a 80%/20% split.

The latest status update of WP2 components is compiled in Appendix B.

2.3.3 Recommendation and Scheduling

With our decision to turn the Content sWitch into the 4u2 Messenger, we change the way *Recommendation & Scheduling* works. Instead of working on the level of viewer segments, the system can now recommend videos to a specific user and take into account their reactions. The detailed recommendation algorithm used in the *Recommendation & Scheduling* component is described in D3.3. To predict the best date for optimal success of the future publication of a piece of content, we directly use the *Prediction Service* in the Content Wizard.

Video Adaptation

The *Video Adaptation* service fetches the analysis results for videos that were previously ingested by the WP1 services from the *Video Feature Storage*. Then, shorter versions of the original video are generated by employing video summarization methods. The generated summaries can be retargeted to a user-defined aspect ratio different from that of the original video using a smart cropping technique which is reported in Section 2.2 of D3.3. This service has undergone several revisions in a continuous effort to be adapted to newly presented challenges and requirements of content partners, as reported in Deliverable 3.3.

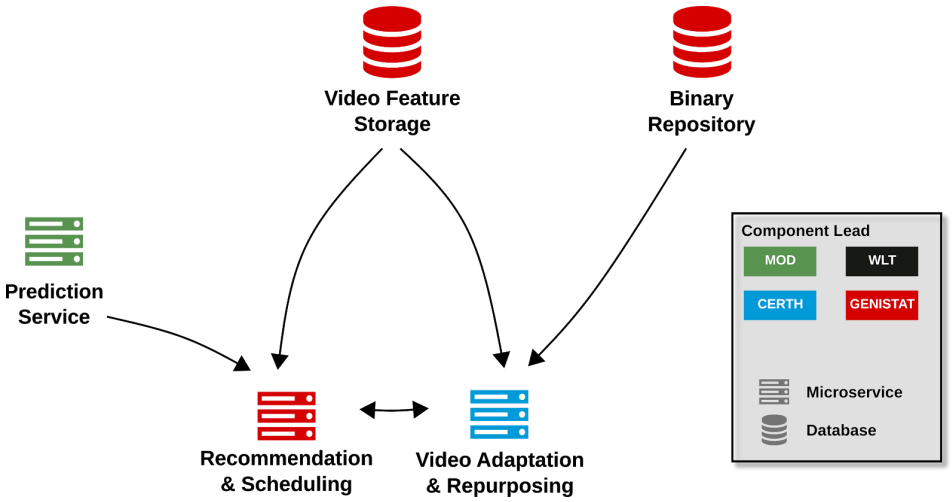


Figure 4: TVP subsystem to enable video adaptation and repurposing.

Text Summarisation

Content creators need effective methods to optimise the textual content accompanying (re-purposed) video assets in order to increase its impact on target audiences. In analogy to what the *Video Adaptation and Repurposing* component applies to binary content, the Text Summarisation component adapts and repurposes a piece of textual content in preparation for dissemination across multiple publication vectors, e.g. press release, blog posting, social media posting, or a long read (see Section 4.2 for results of the initial evaluation).

Term significance values in a sentence are combined with additional metadata to change and optimize the ranking of the sentences contained in the shortened document. The component provides three ranking options to adapt the condensed representation, tailored to the users' specific goals and tasks:

- **Content.** Focus on key aspects by selecting the most characteristic sentences - computed via keyness metrics that rely on co-occurrence analysis and the distribution of sentences between a target corpus (= current text) and a reference corpus (= entire set of documents published within the past 30 days in the selected content source).
- **Google Search** favors sentences that contain popular search terms, using the average search volume over the past <n> weeks to distinguish words likely to increase organic search traffic from words that are less popular as a search term. Historic values are stored in a term cache, for increased scalability and improved recall.
- **Sentiment** ranks sentences higher that are likely to convey a positive attitude, avoiding expressions with a negative connotation in the public debate. It is based on the same lexical sentiment analysis algorithms that are also used to highlight keyword sentiment with green and red colours in *Storypact* and the *Topics Compass*.

The corresponding REST API call can include **keyword sets** (containing up to ten n-grams) to further optimize the wording and scheduling of posts. The ranking process boosts sentences that contain terms co-occurring with one of more of those keywords (optionally, a synonym lookup sub-module can be activated to increase the recall of the query). This can either increase the likelihood of the summary to be associated with *desired keywords*, or the *expected impact* when embedded into a message to be sent out at a given future point in time - if a keyword set extracted via the *prediction service* is used, for a specific day of a future time range that the author is targeting.

3 Platform Integration and Evaluation

Section 2 presented the TVP from a high-level technical perspective as a microservice architecture, interleaving both binary and metadata workflows to uncover content transformations and publication triggers that support content creators for state-of-the art TV experiences. Section 3 discusses the technical integration of the TVP components into a singular platform for content enhancement, prediction and repurposing. A central aspect of component integration lies in the definition of clear interchanges for data exchange between individual TVP components. In the TVP, the metadata repository with its metadata exchange format defines the methods available to all ReTV partners to connect to the TVP.

3.1 Metadata Exchange Format

All TVP components dealing with textual metadata content were specified at the outset of the ReTV project to be compatible with the WLT Metadata Exchange Format. This is a JSON based document description model, specifying the set of required fields as well as the set of optional fields to be expected in metadata items as passed through the TVP. Whereas the model has been originally developed outside of the ReTV project, amendments have been made in order to make it suitable to the TVP data space. The amendments include:

1. an optional field **`media_brand`**.
2. an optional field **`media_broadcaster`**.
3. an optional field **`temporal_start`**, as a UNIX Timestamp.
4. an optional field **`temporal_end`**, as a UNIX Timestamp.

These changes allow for a close alignment of the EPG data with the audience data, along the temporal, the channel and the brand dimensions. The extended metadata model is shown in the sample document *document.json*, as shown in Listing 2. This model can be used in all the metadata API examples to follow in this chapter.

3.2 METADATA APIS

ReTV ensures a high degree of flexibility of the TVP by offering a rich set of *single-purpose APIs*, allowing for a maximum level of combinations for heterogeneous data flows. The metadata repository takes a central role in the TVP, since it integrates all non-binary contents and analytical results into a single persistence service with a rich set of APIs. The metadata APIs contain:

- Authentication API, one-stop token valid for all metadata APIs.
- Ingestion APIs (Document API and Statistics API) for multiple metadata vectors.
- A Search API, for fast aggregation across mult-vector metadata and vast dimensions.
- A Visualisation API, for high-quality, fully interactive visualisations across large, multi-vector metadata.

All metadata APIs are documented online under *api.weblyard.com*, and special programming tutorials are being shared within the consortium for fast access.


```
{
	"content": "Die Kulturstiftung Pro Helvetia soll das Schweizer
Kunstschaffen im Ausland bekannt machen. Sie verhilft hiesigen
Kulturschaffenden nicht nur zu Ausstellungen und Veranstaltungen in aller
Welt, sie gewährt ihnen auch Atelier- und Recherchestipendien auf vier
Kontinenten. \"Kulturplatz\" fragt, wie nachhaltig diese Fördermaßnahmen
sind.",
	"content_type": "text/plain",
	"repository_id": "retv.weblyzard.com/api_retv_epg",
	"uri": "http://api.bee.genistat.ch/program/zattoo/158",
	"title": "Was bringt Kulturaustausch?",
	"meta_data": {
		"published_date": "2018-03-08T16:10:20.335472",
		"language_id": "de",
		"media_brand": "Kulturplatz",
		"media_broadcaster": "sf1",
		"temporal_start": "1520219400",
		"temporal_end": "1520220900",
		"source": "zattoo_program"
	}
}
```


Listing 2: File *document.json*, an example of a GENISTAT EPG document encoded in the WLT metadata format (JSON).

3.2.1 WLT Authentication API

At consortium partner WLT, all API access is restricted to user login via *Java Web Token*. For this purpose, a stand-alone *WLT Authentication API* is employed, with login credentials provided to all ReTV consortium partners. The Authentication API exposes a single GET endpoint for token generation, with the validity of a single token being eight hours. No quota restrictions are used for the ReTV users.

```
#!/bin/bash
TOKEN=$(curl -s -u username:password https://api.weblyzard.com/0.3/token)
```

Listing 3: Creating an access token valid for all webLyzard APIS, with validity of 8 hours.

3.2.2 WLT Search API

Exclusively for ReTV, the WLT Search API has been extended to cater for the various prediction aspects of WP2 and WP3.

At this time, the WLT Search API supports querying for *key entities* (Person, Organisation, GeoLocation, Event) as defined by the MOD *Semantic Knowledge Base* according to the annotations in the WLT *Metadata Repository*. Due to the integration of future timeline projection into the WLT *Metadata Repository*, this query not only supports historic or current date ranges, but also retrieves predicted future mentions of said entity types.

```
curl -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" -X POST
-d @template_request.json https://api.weblyzard.com/1.0/_retv/keyentities
```

Listing 4: Querying the WLT Trending API for key entities using a query template.

```
"query": {
	"bool": {
		"should": [
			{
				"text": {
					"phrase": "Sandmännchen"
				}
			},
			{
				"text": {
					"phrase": "Sandmaennchen"
				}
			},
			{
				"text": {
					"phrase": "Kalle Kuchenzahn"
				}
			},
			{
				"text": {
					"phrase": "Raketenflieger Timmi"
				}
			},
			{
				"text": {
					"phrase": "Lennart im Grummeltal"
				}
			},
			{
				"text": {
					"phrase": "Pondorondo"
				}
			},
			{
				"text": {
					"phrase": "Antjes Fischkoppgeschichten"
				}
			},
			{
				"text": {
```

```
                                        "phrase": "Lola Langohr"
                                }
                        },
                        {
                                "title": {
                                        "phrase": "Sandmännchen"
                                }
                        },
                        {
                                "title": {
                                        "phrase": "Sandmaennchen"
                                }
                        }
                ]
        },
      "beginDate": "2020-01-13",
      "endDate": "2020-01-19"
}
```

Listing 5: A JSON-based topic template designed by use case partner RBB to restrict the search space of the contextual WLT *Metadata Repository* to the RBB TV show *Sandmännchen*, with a time window projected to the third week in January, 2020.

3.2.3 WLT Document API

The WLT Document API provides a simple HTTP wrapper for data ingestion into the WLT metadata repository. It allows for standard CRUD functionality, and has been configured and activated for data ingestion from both the *MOD Crawler* and the *GENISTAT Scheduler* components. The former pushes metadata content as retrieved from various public news, TV and social media sources into the Metadata Repository, whereas the latter pushes EPG program data provided by Genistat into the TVP (c.f. Section 5, *Content Aggregation and Alignment, WP1*). Sample *cURL* requests for the Document API are shown in Listing 6-9.

```
curl -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json"
-X POST
-d @document.json
https://api.weblyzard.com/0.3/documents/retv.weblyzard.com/api
```

Listing 6: Push (POST) a JSON document to the Metadata Repository.

```
curl -H "Authorization: Bearer $TOKEN" -X GET
https://api.weblyzard.com/0.3/documents/retv.weblyzard.com/api/446202312674
1424383
```

Listing 7: Retrieve (GET) a JSON document to the Metadata Repository.

```
curl -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json"
-X PUT
--data @new_document.json
https://api.weblyzard.com/0.3/documents/retv.weblyzard.com/api/446202312674
1424383
```

Listing 8: Update (PUT) a JSON document to the Metadata Repository.

```
curl -H "Authorization: Bearer $TOKEN" -X DELETE
https://api.weblyzard.com/0.3/documents/retv.weblyzard.com/api/446202312674
1424383
```

Listing 9: Remove (DELETE) a JSON document to the Metadata Repository.

3.2.4 WLT Visualization API

The Topics Compass in its "full" form offers a comprehensive, fully-fledged and feature-rich Data Analytics and Visualisation dashboard customised to the needs of ReTV stakeholders. In order to support use cases that require a more granular approach, the WLT Visualization API enables the integration of distinct dashboard components into third-party Web applications. Version 1 uses *<iframe>* tags to embed these components. While this approach ensures ease of use and widespread compatibility across platforms, it also comes with shortcomings if a deeper integration is desired (e.g. incompatibility with certain browser navigations). This will be addressed by additional features in future versions of the API, complementing (but not replacing) the *<iframe>* approach.

Configuration of visualizations (i.e. the exact topics/search terms behind the visualization) are managed by webLyzard and can be defined on request by the use case partners. Access to a configuration for embeddings is achieved via a token in the URL, which tells the webLyzard Visualization API what configuration to apply to the rendered iframe embedding. The required *<token>* is provided by webLyzard and might change over time. This token ensures that the correct search configuration is used for the visualizations.

The iframe should be provided with all necessary attributes:

- *width* - the iframe's width in pixels (int)
- *height* - the iframe's height in pixels (int)[14]
- *src* - URL of the visualization, see also next section "URL Schema" (string)
- *frameborder* - should be "0" to avoid rendering standard iframe border (string) (deprecated in HTML5 and will later be replaced in CSS)
- *scrolling* - should be "no" to avoid scrolling bars (no,yes,auto) (deprecated in HTML5 and will later be replaced in CSS)
- *sandbox* - not required, but if present a value of "allow-same-origin allow-scripts" is required

Story Graph Visualisation

The TVP story detection module identifies and describes groups of related documents (= referred to as stories) from digital content streams. The following figure summarizes the results of such a clustering process based on a query on "Tesla" in the form of an interactive *Story Graph.* Hovering individual stories indicates their duration, the number of documents that belong to a particular story, and the associated keywords. Various graph rendering options to adapt the visualisation to specific use case requirements include labels (on or off), the underlying metric to be used (document count vs. weight) and the methods to stack (silhouette, expand, zero, wiggle) and sort (default, inside-out, reverse) the stories.

```
<iframe width=960 height=600 frameborder="0" scrolling="no"
src="https://api.retv.weblyzard.com/embed/PDna4PFm9huwWPpFGjonK2NC2WwCkvrV/
storygraph"></iframe>
```

Listing 10: Story graph <iframe> embed for HTML integration of story detection results.

[14] Width and height can be preset to the desired dimensions, similar to the approach of e.g. YouTube.

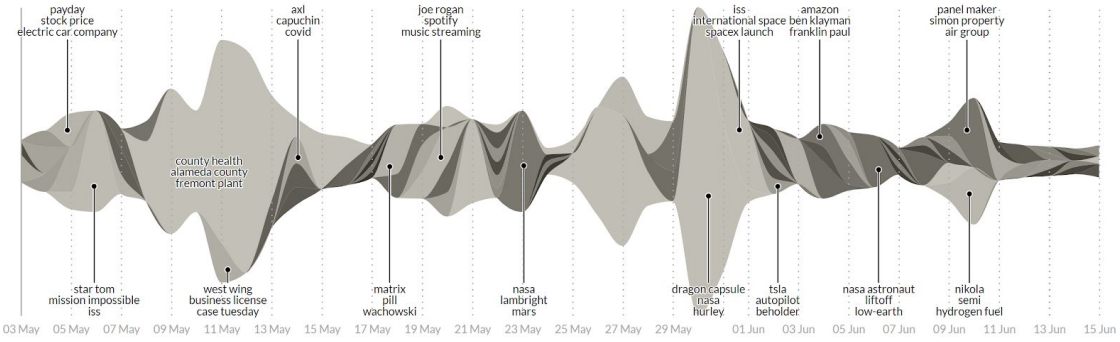


Figure 5: Story graph <iframe> embed for HTML integration of story detection results.

Geographic Visualisation

The geographic map module projects documents from the WLT metadata repository within their location of publication as well as within the locations referenced within their content. It allows the user of the *Topics Compass* to determine the geographic scope of their topics of interest quickly and effectively, e.g. to investigate the regional distribution of news articles or social media postings. Figure 6 shows an iframe embed of the GeoMap for a given ReTV topic. This embed can be rendered with the HTML embedding shown in Listing 10.

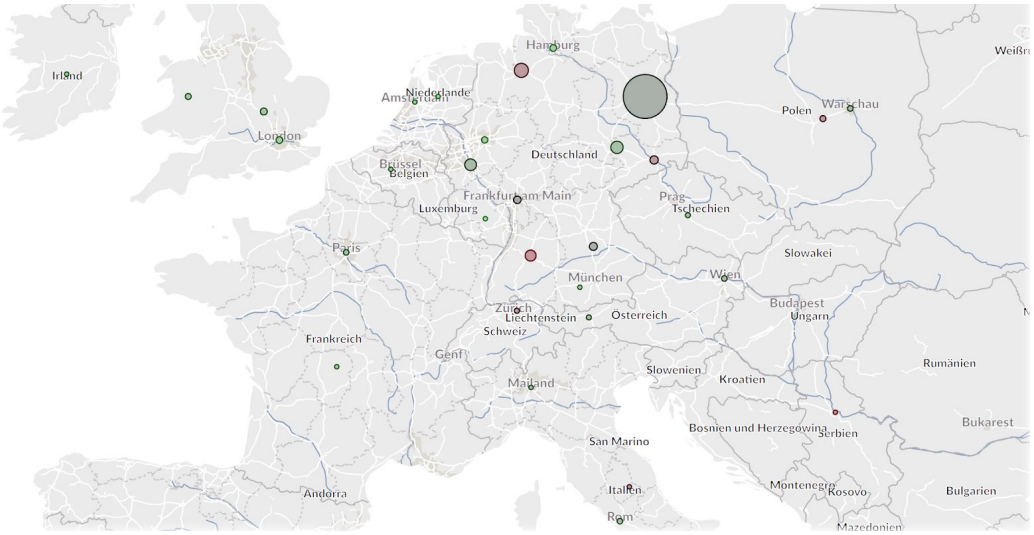


Figure 6. Geomap <iframe> embed for HTML integration of a predefined ReTV Topic.

```
<iframe width=960 height=600 frameborder="0" scrolling="no"
src="https://api.retv.weblyzard.com/embed/ump7lbw1OTInsgfyiuYoClQZFY/geomap
"></iframe>
```

Listing 11: Geomap <iframe> embed for HTML integration of a predefined ReTV Topic.

Tag Cloud Visualisation

The *Tag Cloud* uses dynamic transitions to adapt its content to the current search. It arranges the most relevant keywords alphabetically, uses colour coding to show their positive or negative sentiment and adjusts the font size according to their relative importance (= number of occurrences in the list of search results). The colour coding of various widgets including the tag cloud has been made more flexible. The component now supports both the drill down and

comparison modes, for example to not only visualise positive versus negative sentiment, but also multi-dimensional emotions based on Plutchik's *Wheel of Emotions*. Figure 7 shows an iframe embed of the Tag Cloud for a given ReTV topic. This embed can be rendered with the HTML embedding shown in Listing 11.

Figure 7. Tag cloud <iframe> embed for HTML integration of a predefined ReTV Topic.

```
<iframe width=340 height=280 frameborder="0" scrolling="no"
src="https://api.retv.weblyzard.com/embed/ump7lbw1OTInsgfyiuYoClQZFY/tagcloud"></iframe>
```

Listing 12: Tag cloud <iframe> embed for HTML integration of a predefined ReTV Topic.

Keyword Graph Visualisation

The keyword graph shows the search term together with its strongest *associations* within the search results. The resulting *semantic network* is a powerful tool to guide the development and ongoing adaptation of an organization's communication strategies. The keyword graph allows investigating how different stakeholder groups perceive brands or topics, what negative and positive associations they have with a topic (the colour of graph nodes encodes their sentiment), and how the meaning of specific terms changes over time.

Figure 8 shows an iframe embed of the Keyword Graph for a given topic. This embed can be rendered with the HTML embedding shown in Listing 12.

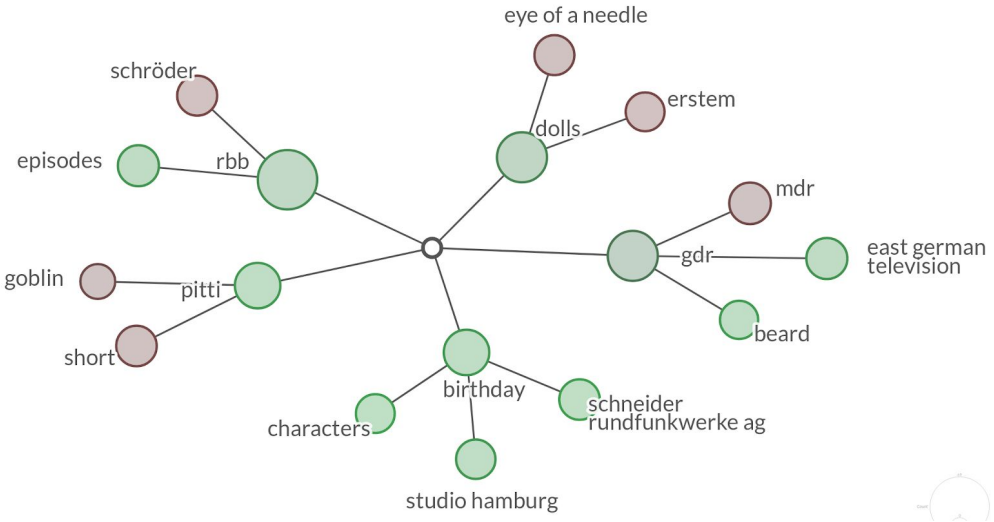


Figure 8: Keyword Graph <iframe> embed for HTML integration of a predefined ReTV Topic.

```
<iframe width=600 height=400 frameborder="0" scrolling="no"
src="https://api.retv.weblyzard.com/embed/ump7lbw1OTInsgfyiuYoClQZFY/keywor
d"></iframe>
```

Listing 13: Keyword Graph <iframe> embed for HTML integration of a predefined ReTV Topic.

Integrated Visualisation

This integration provides a powerful search interface alongside four different analytic visualisations tools. It allows for comparison of multiple topics by retrieving the most relevant documents for each topic and additionally setting a search context by defining date ranges, data sources and an optional filter term. The documents are presented both in a simple and straight-forward List View as well as in analytic visualisations like the Tag Cloud, Keyword Graph, Story Graph and Cluster Map, in which each visualisation provides a unique perspective on key characteristics of the documents.

Figure 9 shows an iframe embed of the Visualisation integration for the given search. This embed can be rendered with the HTML embedding shown in Listing 14.

```
<iframe width="800" height="500" frameborder="0" scrolling="no"
src="https://api.retv.weblyzard.com/embed/bgKqJZPr8Y4ZTiTtVrxrMZgDFMXMFJ97"
></iframe>
```

Listing 14: Visualisation integration <iframe> embed for HTML integration.

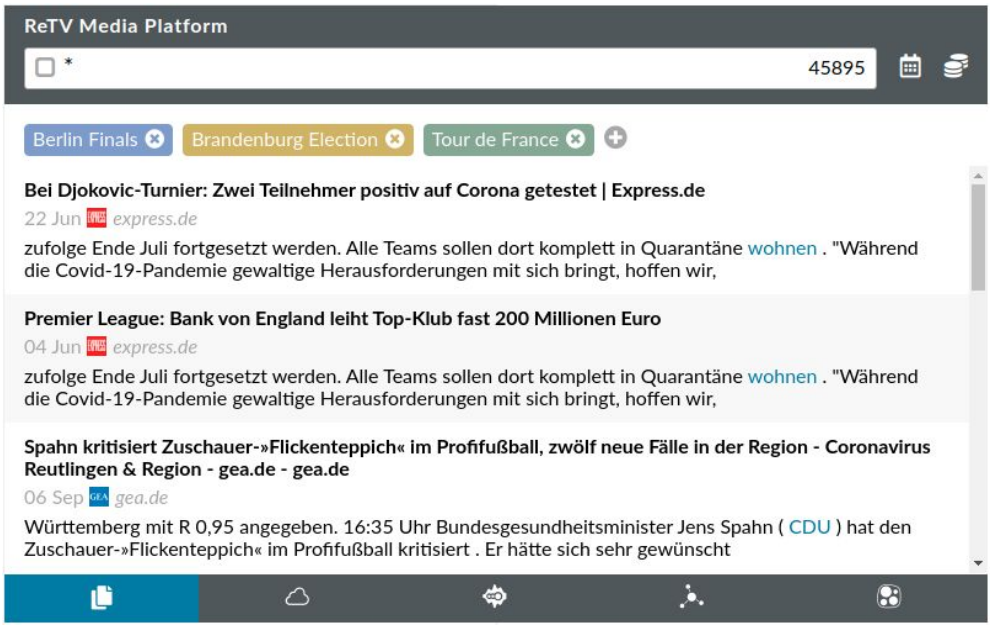


Figure 9: Visualisation integration <iframe> embed for HTML integration.

Embeddable Storypact Editor

The Storypact editor is a data-driven text optimization tool to help content creators maximise the impact of their stories. It supports the authoring of various types of publications including web pages, press releases, blog entries and social media postings. Storypact analyzes the text while you write, suggesting modifications to increase the expected impact and align the text with your specific communication goals. The suggestions are based on a real-time analysis of

the public debate across various online channels - identifying keywords, emerging stories and opinion leaders that trigger and drive these stories.

Figure 10 shows an iframe embed of the Visualisation integration for the given search. This embed can be rendered with the HTML embedding shown in Listing 15.

```
<iframe frameborder="0" style="width:100%;height:100%;"
src="https://www.storypact.com/editor/?languageMode=manual"
></iframe>
```

Listing 15: Visualisation integration <iframe> embed for HTML integration.

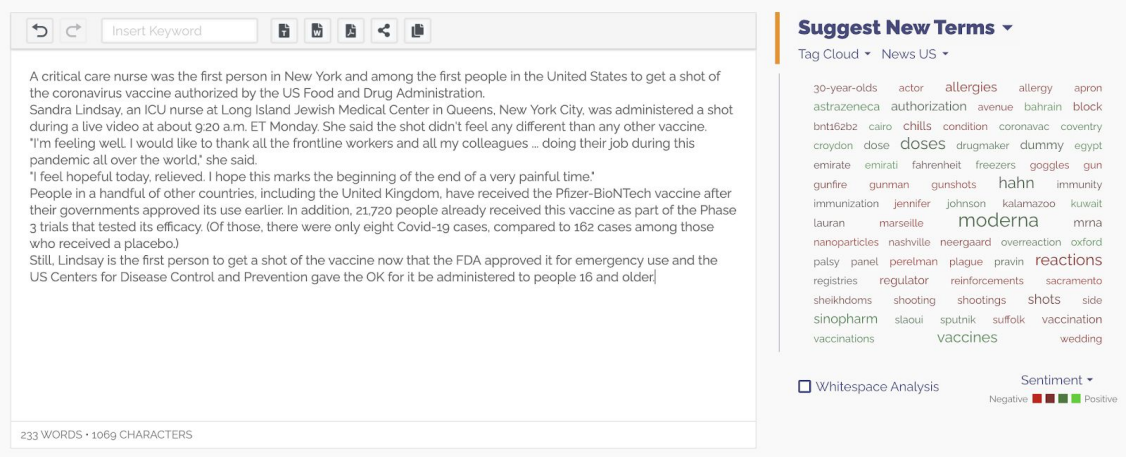


Figure 10: Storypact <iframe> embed for HTML integration.

3.2.5 WLT Statistics API

The WLT *Statistics API* provides a wrapper to ingest statistical time-series data (optional: geotags) into an ElasticSearch persistence layer, with full CRUD abstraction. Any content ingested in the Statistics API can be correlated against content from the WLT Metadata Repository, and the resulting insights can in turn be visualised using the WLT Visualisation API.

The webLyzard *Stakeholder Dialog and Opinion Model* (WYSDOM) is particularly relevant for ReTV, as it provides a dynamic assessment of communication success that goes beyond sentiment and allows insights into the impact of published content. Figure 11 shows the WYSDOM chart with ingested audience data points provided by Genistat, stored in the WLT Metadata Repository. Audience data is shown by the blue bars, correlated with desired (green) and undesired (red) keyword associations as defined by use case partner RBB. Measuring attention and sentiment is descriptive in nature, whereas the hybrid WYSDOM success metric, reveals whether communication targets have been reached. It measures to what extent the chosen strategy has an impact on observable patterns in online coverage.

Instead of making the WYSDOM representation embeddable, a feature not deemed a priority by any of the use case partners, development work in Year 3 focused on the integrated *Topics Compass* version - from a very specific component relying on hard-coded metadata categories (sentiment, desired/undesired associations) into a more flexible stacked bar chart component that supports all types of checkboxes from the dashboard's left sidebar including bookmarks, associations and metadata categories.

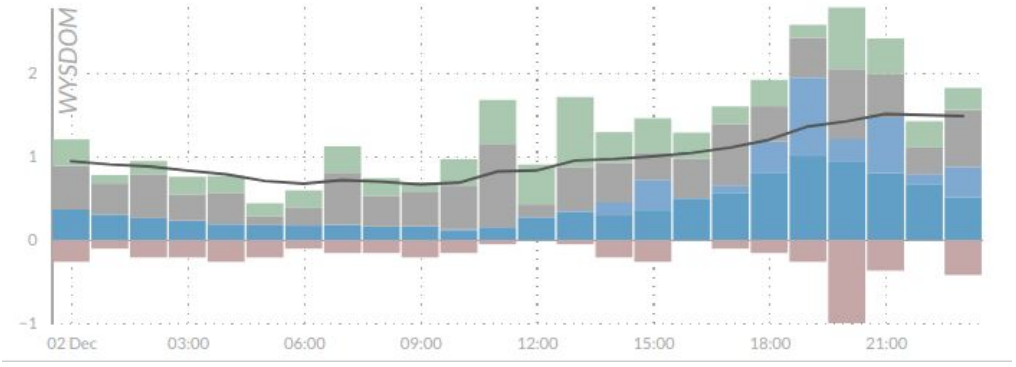


Figure 11: WYSDOM chart with desired versus undesired topics (green, red) and audience metrics (blue)

3.3 Prediction Components

The prediction capabilities of the TVP are realised through the prediction components *Prediction Service*, *Events and Anniversaries API* and *Semantic Knowledge Base*. Together, these offer a rich set of freely-configurable search queries against future time frames.

3.3.1 MOD Events API

The Event API grants query access to the events (EventEntities) stored in the MOD *Semantic Knowledge Base* (SKB). It supports different search functionalities, such as a search for events in a given timespan, and an advanced search for events that match given properties with the use of filters.

Date Range Search

To retrieve all events within a specific timespan the (optional) parameters `from_date` and `to_date` exist. A sample *cURL* request for a date range search on the MOD Event API is shown in Listing 16.

```
curl -X POST -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json"
--data '{"from_date": "2020-04-01", "to_date": "2020-06-31"}'
https://api.weblyzard.com/1.0/skb/events
```

Listing 16: Date range search, returning all events that happen between 1st April and 31th June 2020.

Property names that can be used with filters are:

- `skbprop:eventCategory`, the general event category, e.g. Sport, NewsEvent, Politics
- `skbprop:confirmation_status`, status of the event, e.g. confirmed/unconfirmed
- `skbprop:temporal_start`, starting datetime
- `skbprop:temporal_end`, ending datetime
- `skbprop:location`, location where the event takes place, either exact location, e.g. Emirates Stadium, or if unknown country, e.g. Bolivia (wd:Q750) or Kingdom of Spain (geonames:2510769)
- `skbprop:country`, country wikidata or geonames id of the location country, e.g. Austria (wd:Q40)
- `skbprop:countryCode`, ISO-code of the location country, e.g. DE, AT
- `skbprop:mdDate`, month and day date value as a string, e.g. 12-10
- `skbprop:year`, year the event takes place as an integer value, e.g. 2017

- `skbrop:next_instance`, for repeating events, such as elections, holidays or awareness days
- `schema:description`, a description of the event, e.g. 1. Bundesliga, 14. matchday
- `dct:source`, the original (web)source, e.g. ical calendar link or official Web site
- `dct:type`, the event type or subtype, e.g. Premier League (wd:Q9448) or *http://weblyzard.com/skb/events/recurring_event*
- `dct:date`, date of a full-day (repeating) event, e.g. 2020-07-28
- `skos:altLabel`, optional alternative labels, e.g. Erster Weihnachtsfeiertag

To ensure consistent data for search, a normalization attempt is made to include respective geoname ids for geo-relevant properties `skbprop:location` and `skbprop:country` even if the original data source only provides wikidata ids or name strings.

When a natural-language string is given as a property value in a search filter, it is - if possible - resolved into a wikidata and geonames id that are then included in the performed search, i.e. `property_value:london` also returns results for `wd:Q84` (London) and `geonames:2643744` (City of London).

To improve useability of the Events API the previously necessary filter parameter `property_format` has been completely removed and is now automatically derived from the provided `filter_operator`. At the same time supported text search operators were extended to now include `term`, `match`, `prefix`, `match_phrase_prefix`, `wildcard` and `contains` based on the underlying ElasticSearch query language. Compatibility with the previous version has been ensured.

Full-Text Search

Basic full-text search filters use `filter_operator:match`. Alternatively `filter_operator:match_phrase_prefix` can be used to match an (incomplete) phrase. Using `contains` has the same functionality as `match` for normal text queries but additionally allows more complex query strings such as leading wildcards, e.g. `*_DE` and simple logical operators (`AND`, `OR`) within a property value. Sample *cURL* requests for the full-text search on the MOD Event API are shown in Listing 17 and Listing 17.

```
curl -X POST -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json"
--data '{"filters": [{"property_name":"rdfs:label", "property_value":"wien",
"filter_operator":"match"}]}'
https://api.weblyzard.com/1.0/skb/events
```

Listing 17: Full-text search, returning all events with arbitrary date that match the query term “wien” in a `rdfs:label` property.

```
curl -X POST -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json"
--data '{"filters": [{"property_name":"schema:description", "property_value":"North
America", "filter_operator":"match_phrase_prefix"}]}'
https://api.weblyzard.com/1.0/skb/events
```

Listing 18: Full-text search for a phrase, returning all events with arbitrary date that match the phrases “North America” as well as “North American” in a `schema:description` property.

Exact Match Search

Exact match search filters are done with `filter_operator:term` or `filter_operator:prefix to` match a given prefix of a term. A sample *cURL* request for the exact match search on the MOD Event API is shown in Listing 19.

```
curl -X POST -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json"
--data '{"filters": [{"property_name":"dct:type", "property_value":"wd:Q16466010",
"filter_operator":"term"}]}'
https://api.weblyzard.com/1.0/skb/events
```

Listing 19: Exact-match search, returning all events of type Association football match.

Integer Search

Integer search allows one to search the MOD Event API by year directly. A sample *cURL* request for the integer year search on the MOD Event API is shown in Listing 20.

```
curl -X POST -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json"
--data '{"filters": [{"property_name":"skbprop:year", "property_value":"2001",
"filter_operator":"lte"}]}'
https://api.weblyzard.com/1.0/skb/events
```

Listing 20: Integer search query, returning all events that happened before and including the year 2001.

Conjunctive Search

Any number of filters can be combined, i.e. the following curl example returns association football matches in March 2019 with a description containing premier league. A sample *cURL* request for the conjunctive search capabilities on the MOD Event API is shown in Listing 21.

```
curl -X POST -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json"
--data '{"from_date":"2019-03-01", "to_date": "2019-03-31",
"filters": [{"property_name":"dct:type", "property_value":"wd:Q16466010",
"filter_operator":"term"}, {"property_name":"schema:description",
"property_value":"premier league", "filter_operator":"match"}]}'
https://api.weblyzard.com/1.0/skb/events
```

Listing 21: Conjunctive search query, using a combination of search constraints in conjunction.

Negation

By adding `negate:true` to a filter, it is inverted and events that match this property filter are not included in the results. A sample *cURL* request for the integer year search on the MOD Event API is shown in Listing 22.

```
curl -X POST -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json"
--data '{"filters": [{"property_name":"rdfs:label", "property_value":"Christmas",
"filter_operator":"match"}, {"property_name":"dct:type",
"property_value":"http://weblyzard.com/skb/events/holiday OR wd:Q1445650",
"filter_operator":"contains", "negate":"true"}]}'
https://api.weblyzard.com/1.0/skb/events
```

Listing 22: Search query with a negation filter, returning all events with "Christmas" in `rdfs:label` that do not have a `dct:type` associated with holiday.

Language Filters

There exists an optional filter to only retrieve labels and descriptions in a certain language. If applied the properties `rdfs:label`, `schema:description` and `skos:altLabel` are only returned in that language, instead of a list with all known labels and descriptions. Labels and descriptions with unknown language are always returned. A sample *cURL* request for the language filters for the MOD Event API is shown in Listing 23.

```
curl -X POST -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json"
--data '{"language": "de", "filters": [{"property_name":"rdfs:label",
"property_value":"Weltmeisterschaft", "filter_operator":"contains"}]}'
https://api.weblyzard.com/1.0/skb/events
```

Listing 23: Query with output language specified to be German.

Provenance Filters

An optional filter allows retrieving events of a certain source (provenance). This is most useful to filter events that originate from wikidata from other events extracted through, for example, *iCal*. Provenance uses substring match, so using `provenance:wiki` will work as well. Similarly it is also possible to filter for e.g. *HTTP*. A sample *cURL* request for the language filters for the MOD Event API is shown in Listing 24.

```
curl -X POST -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json"
--data '{"provenance": "wiki", "filters":
[{"property_name":"skbprop:eventCategory", "property_value":"Politics",
"filter_operator":"term"}]}'
https://api.weblyzard.com/1.0/skb/events
```

Listing 24: Provenance query, returning only events in the category *Politics*, with provenance *wikidata*.

3.3.2 MOD Anniversaries API

Querying the *Semantic Knowledge Base (SKB)* for anniversaries, i.e. events and significant occurrences that take place on a given day, where the day is provided as a month-day pair. The response returns events such as birthdays, dates of death, inception dates of organizations, holidays or sporting events that match that day. A sample *cURL* request for the anniversary search on the MOD Anniversaries API is shown in Listing 25.

```
curl -X POST -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json"
--data '{ "date": "04-01", "entity_type": "PersonEntity"}'
https://api.weblyzard.com/1.0/skb/anniversaries
```

Listing 25: Anniversaries query, returning all persons that have an anniversary happening on 1st April.

To restrict the anniversary results two optional parameters are supported:

- `anniv_num` takes an integer *n* to filter for *n*-th anniversaries (and multiples of n), i.e. *n=50* returns 50th, 100th, 150th etc. anniversaries this year
- `search_phrase` takes a string to include in the search, matched to any property value

A sample *cURL* request for the anniversary search with optional parameters on the MOD Anniversaries API is shown in Listing 26.

```
curl -X POST -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json"
--data '{"date": "12-12", "entity_type" : "PersonEntity", "anniv_num":50,
"search_phrase":"German"}'
https://api.weblyzard.com/1.0/skb/anniversaries
```

Listing 26: Anniversaries query with additional parameters, returning all persons whose properties mention German, that this year have a 50th or 100th etc. anniversary on the 12th December.

3.3.3 Scenario Template API for Events and Anniversaries

Based on collected requirements, individual scenario templates were defined for project partners NISV and RBB. Each template consists of predefined requests to the Events API and the Anniversaries API that together reflect a particular interest. A single template could be for example summarized as "events and anniversaries relevant for a Dutch audience".

In addition to the requested `template`, the Scenario Template API takes a `daterange` for event queries and `md-date` and `anniv_num` for anniversary queries. It returns the aggregated results of the queries as defined in the template, with the given parameters filled in. A sample *cURL* request for the scenario template API is shown in Listing 27.

Additional templates can be defined by MODUL for additional stakeholders as required.

```
curl -X POST -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json"
--data '{"template": "retv.weblyzard.com/nisv", "parameters": {"daterange":
{"from":"2020-10-01", "to":"2020-10-31"}, "md-date": "10-31", "anniv_num": 10}}'
https://api.weblyzard.com/1.0/skb/anniversaries
```

Listing 27: Request aggregated results for events in October and for round anniversaries on the 31st October applying the NISV scenario template.

3.3.4 MOD Temporal Annotation API

The *Temporal Annotation API* from ReTV partner MODUL allows the identification of mentions of dates, both relative and absolute, within a document and the addition of the referenced dates as an annotation to the document in the *Metadata Repository*. This component constitutes one of the core steps that enable prediction. A sample *cURL* request for the text annotation with temporal information via the MOD Annotation API is shown in Listing 28.

```
curl -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" -X POST
-d @document.json https://api.weblyzard.com/0.3/annotate/temporal
```

Listing 28: Annotate (POST) a JSON document with temporal information for prediction.

3.3.5 MOD Prediction API

The *Prediction API* from ReTV partner MODUL allows to identify a future date in a given date range on which a provided keyword or topic is predicted to experience highest popularity among the audience (i.e. the success metric of frequency of mentions for the keyword or topic is predicted to peak on this date within the date range).

The API takes a JSON request such as the example below:

```
{
   "sources": [
     "retv.weblyzard.com/new_news_en"
   ],
   "query": {
     "text": {
       "phrase": "inauguration"
     }
   },
   "averaging": {
     "units": "1"
   },
   "interval": "day",
   "beginDate": "2020-10-01",
   "endDate": "2021-04-30"
}
```

Listing 29: MOD Prediction API JSON Request

Here, the data source of global English news coverage is to be used to predict the relative popularity of the keyword "inauguration" in the time period 1 October 2020 to 30 April 2021. It is based on predicted absolute values for the frequency metric for each day ("averaging/units" = 1) from the beginDate until the end Date. The query, just like the WLT Search API, can combine keywords so that a composite prediction may be acquired for a specific topic (which is defined as a set of keywords relevant to that topic). The predicted values may also be returned as an n-day moving average which better reflects the likelihood that a topic remains in the attention of the audience for some days after a significant peak in appearance (e.g. 7 day moving average values through "averaging/units" = 7). The query can also be used to retrieve predicted sentiment or share of voice metric values as well, using the metrics calculation support of the webLyzard platform.

The API then responds with histogram data in JSON format like this:

```
{
   "result": {
     "datapoints": [
       {
         "date": 1601510400000,
         "frequency": 4.0
       },
       {
         "date": 1601596800000,
         "frequency": 4.0
       },
…
     ]
   }
}
```

Listing 30: MOD Prediction API JSON Result

The default success metric is the 'frequency' metric, as discussed in D2.3. The dates use the Unix timestamp which can be converted to other calendar values by most programming languages. The result is also shown below, with a clear peak predicted for the date of 21 January 2021, the date of the inauguration of the new President of the United States.

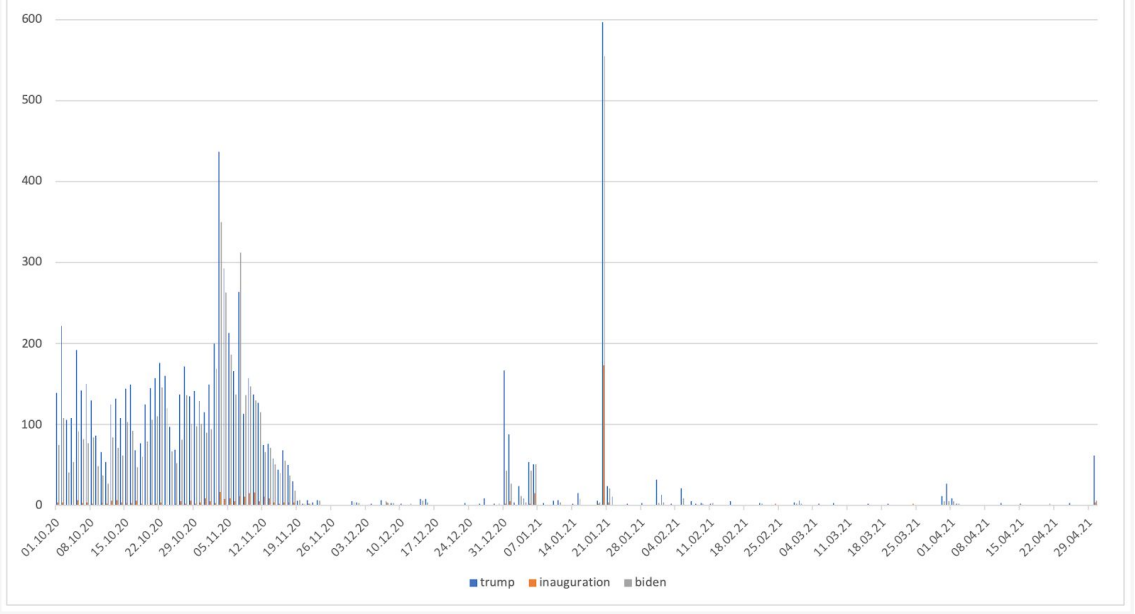


Figure 12. Prediction results for the 'inauguration' keyword, Oct 2020 - Apr 2021

3.4 Software Quality

Software quality management is overseen by WP8 (c.f. D8.1, *Quality and Data Management Plan*). WPs 1-3 maintain their own software quality management in accordance with these recommended software quality guidelines, and these are reported respectively in D1.2, D2.2, and D3.2. All work in WP4 is following similar recommendations on software quality. All software produced in this work package adheres to:

- Strict implementation of issue tracking and code review.
- Test-driven design principles. To achieve best test coverage of software at unit-level, all developers are trained and reminded to adhere to state-of-the-art principles from Test-Driven Development (TDD).
- Latest development infrastructures. Programming languages in use are constantly updated to their latest stable release versions (Python3.7, Java11) to optimise security, stability, features, and support. Build, test, and development infrastructures are kept up-to-date for similar reasons.
- Static code analysis tools run both on Java and Python environments to counter software defects and code smells as early as possible (Sonarqube, Sentry).
- Continuous Delivery allows for a high volume of release cycles, establishing a sense of software ownership.
- Detection of integration defects through a full-fledged staging environment.

For integration testing, we define a set of integration tests in the next subsection 3.6, *Platform Evaluation*. While this evaluates the live TVP, similar tests are performed in staging setups, where possible. Note that due to the distributed nature of the TVP, not all integration tests can be run isolated in staging environments.

3.5 Platform Evaluation

The TVP is a complex and distributed software architecture that integrates a wide range of service components and database repositories from multiple technical partners, tied together via data exchanges and abstractions. For such complex architecture to work both in evaluation as well under load in the field, we are taking multiple measures to guarantee the platform's correctness and performance.

We evaluate the TVP via integration and scalability testing and individual TVP components are evaluated via unit testing and software quality assessments.

3.5.1 Integration Testing

Success criteria for the correctness and completeness of the TVP is asserted via the four use case scenarios described in WPs 5 and 6.

Data Retrieval Subsystem

Since data retrieval drives the entire TVP platform, it is of utmost importance to guarantee a steady flow of data on the binary streams as well as on the metadata streams. In WP4, we have established tight data quality monitoring and alerting on the storage tier following the data ingestion. This allows us to assert the correctness of the integration of the complete data retrieval subsystem from data intake to persistence.

The monitoring system is designed to detect anomalies in any data intake component within at most an hour, and to alert the responsible service operator to guarantee small response times on errors. Table 3 shows that all four data retrieval components are working in a satisfactory manner.

Data Source	Component	Status	Outage/Week
WikiData Events	Temporal Annotation	Running	Low
Web sites/Social Media APIs	Crawler	Running	Low
EPG	Scheduler	Running	Low
Audience Metrics	Scheduler	Running	Low

Table 3: Operational overview of the data retrieval subsystem of the TVP. *Low Outage*: less than 1 hour of downtime per week; *Medium Outage*: between 1 and 3 hours of downtime per week; *High Outage*: above 3 hours of downtime per week.

Prediction Subsystem

The prediction subsystem takes data inputs from the SKB and the Metadata Search Index as well as produces the data output (the prediction) to consuming components for the scenarios (the Visualisation Engine for the Topics Compass or Content Wizard, Text Summarization for the Content Wizard or the Recommendation & Scheduling for the Content Wizard and 4u2 Messenger). The subsystem covers four different cases of prediction:

1. Prediction of audience metrics uses audience metrics from the Metadata Search Index and events from the Semantic Knowledge Base, outputs to the Visualisation Engine and the Recommendation & Scheduling components. Correctness of operation is

shown by response to queries from these components (e.g. channel or show + future time period) with predicted audience.

2. Prediction of success metrics uses success metrics from the Metadata Search Index and outputs to the Visualisation Engine, the Text Summarization and the Recommendation & Scheduling components. Correctness of operation is shown by response to queries from these components (topic + success metric + future time period) with predicted success metric value.

3. Prediction of future keyword popularity uses the *Temporal Annotation* through the SKB and outputs to the Visualisation Engine, the Text Summarization and the Recommendation & Scheduling components. Correctness of operation is shown by response to queries from these components (future time period, optional keywords) with predicted keyword popularity at that future time.

4. Prediction of future events of relevance uses the *Event Extraction* via the SKB and outputs to the Text Summarization component. Correctness of operation is shown by response to queries from these components (future time period, optional topics of interest) with future events in that time period.

As mentioned in Chapter 2, we have completed our testing of the case (2) which uses neural networks (LSTM) and concluded on how to best manage a hybrid solution within ReTV depending on the characteristics of the prediction task:

EVALUATION OF FORECASTING

Prediction accuracy for our neural network (LSTM) is illustrated below with two of our evaluation datasets, for the keywords “cycling” and “filmfestival” respectively (from global news media, 995 data points each where 1 data point represents 1 day). ‘Test’ data is the 20% split of the dataset, following training with the first 80%. ‘Val’(idation) data is the next 30 days of actual values (not in the input dataset for training/testing), where comparison of predicted values is made to the 7-day moving average of the actual values (as explained in D2.3).

Metrics are reported as the average from 5 runs. E-D refers to Encoder-Decoder and AR to autoregression for single step forecasts. TS is TimeSeriesGenerator. (7,1) and (200,30) are the input-output sequences used. SARIMAX is also given as a comparative benchmark.

Cycling	Test MAE	Test RMSE	Val MAE	Val RMSE
SARIMAX	11.5	15.4	15.3	18.5
LSTM E-D AR(7,1)			22.1	26.1
LSTM E-D TS AR(7,1)	11.6	15.9	21.1	23.6
LSTM E-D (200,30)		14.9	26.8	28.1
LSTM Seq2seq+attention with Swish (200,30)		**15.7**	**9**	**10.4**

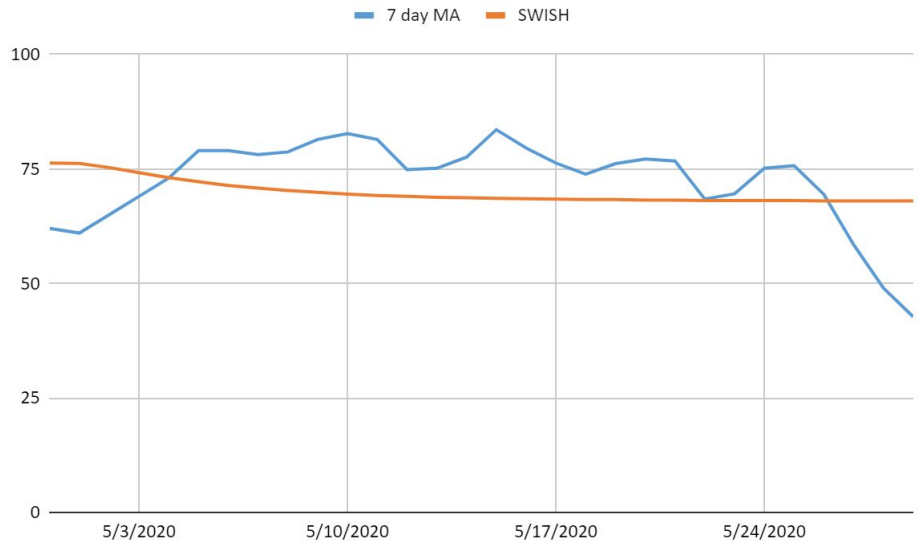


Figure 13: Predicted values vs real 7 day moving average for "Cycling"

Filmfestival	Test MAE	Test RMSE	Val MAE	Val RMSE
SARIMAX	12.8	16.9	20.3	24.5
LSTM E-D AR(7,1)			18.4	21.3
LSTM E-D TS AR(7,1)	17.1	23.1	25.9	35.6
LSTM E-D (200,30)		24.6	10.6	12.8
LSTM Seq2seq+attention with Swish (200, 30)		**8.4**	**10.5**	**13**

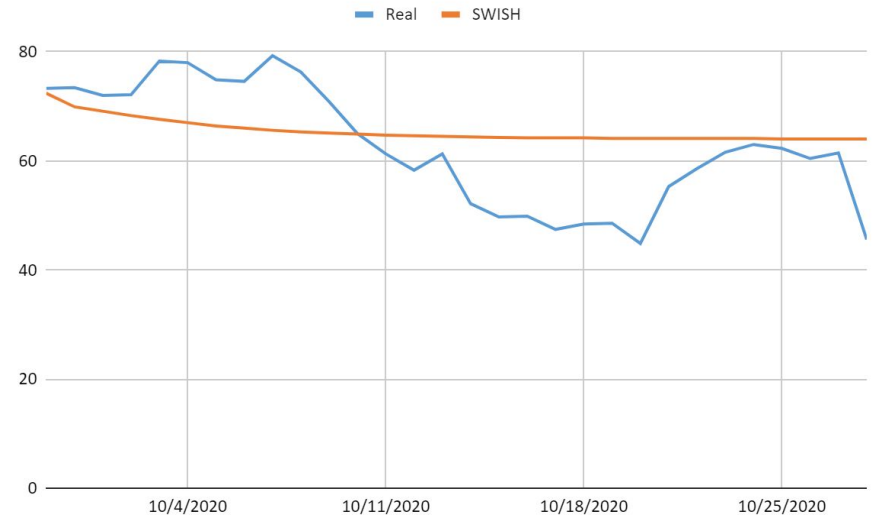


Figure 14: Predicted values vs real 7 day moving average for "Filmfestival".

EVALUATION OF FORECASTING WITH OTHER FEATURES

The above results suggest we have a time series forecasting model that can be used to provide "close" predictions for an arbitrary keyword's (or topic's) future success metrics (frequency). This complements the other methods available in ReTV, where a date can be used as a seed to get predictions for which keywords will be most popular on that date or which events or anniversaries are relevant to the audience on that date. These methods work less well the other way around, i.e. input a keyword (or topic) and get the date(s) where this keyword (or topic) will be more relevant, simply because this relies on the chosen keyword (or topic) being associated in the document corpus to dates in the range specified, or being present in entity metadata of entities relevant to the future date. Only a small subset of all keywords (natural language terms) will be significant for any selected prediction task with these methods, therefore the forecasting method addresses this gap (e.g. a stakeholder wants to compare predicted future success of content based on different keywords available to them). On the other hand, forecasting relies on a longer period of past time-series values relevant to the current prediction task, which in turn suggests it works better with keywords whose relevance (presence in the document corpus) have been relatively constant over the past year at least.
To conclude our technical investigation of prediction, we need to consider two further cases that would currently affect very negatively the accuracy of our forecasting:

1. Newly emerging keywords. Our LSTM is trained with input sequences of 200 time series data points. Some keywords emerge in documents only recently, where there is not sufficient past time series data for metrics to train adequately our LSTM model.

2. Future significant disruptions. The resulting prediction series is currently based purely on the past behaviour in the time series data, so its accuracy is predicated on the same behaviour persisting into the future. If something happens in the prediction time period which is significant in changing the usual level of discourse about the keyword/topic, the prediction results will likely prove to be far less accurate. Not all, but some, significant disruptions in the future can be known at the time of prediction, and including that knowledge in the model can help improve its predictive accuracy.

Case 1: Newly Emerging Keywords

Especially in the world of news and television, new topics and keywords can appear in recent time and thus suggest themselves as potentially useful indicators of content for future publication success. However, a significant amount of past time series data is missing to be able to train our prediction model.

For example, on 6 May 2020, Elon Musk and Grimes announced that they named their new baby "X Æ A-12", thus generating a trending keyword on global news media that definitely had not been seen before. When considering prediction with such a keyword, there needs to be some minimum 'lead time' for the keyword to persist in the online discourse (to exclude keywords which appear as one-off occurrences in the document corpus, returning to zero metrics in the space of days). We choose two weeks, as we considered 7 days the minimum input sequence length for a one step forecast and this ensures at least 7 input sequences in training, with at least one input sequence as a test.

Baseline: using our chosen approach, we take the frequency metric for the keyword from 1 Jan 2018 to 20 May 2020 (two weeks after the initial keyword-related event). We want to predict the next 30 days. Note that the metric is zero until (first instance of the keyword) May 2, 2020

which is only 18 days before the end of the time series. We have an input data set of 864 data points. However, as expected, the predicted values output are all almost zero and it is established that we need a different approach for keywords which only experience a significant presence in the online discourse since a recent time.

Comparison: an input-output sequence of (200,30) might be effective for longer term prediction of long present keywords in the online discourse but cannot work for newly emerged keywords. We re-consider our Encoder Decoder model from the initial LSTM tests with an input-output sequence of (7,1), feeding it only three weeks of data (the heuristic can be to start on the first date where the keyword metric is greater than zero) and using auto-regression to predict for the next 7 days. It seems a fairer training period for metrics prediction of a keyword which has emerged only a few weeks ago is to use at least two weeks or 14 data points, ideally longer of course. Our resulting prediction (values of 8.3 to 11.9) was initially close to the actual values (17 and 13 on the first two days) then stayed in this range while actual values rose again (the baby was in the news again) so RMSE was 37-39 and MAE was 29-30. While not better, such keyword choices are exceptionally non-deterministic (often peaking quickly then falling away permanently) and thus difficult to predict accurately.

Conclusion: if the beginning sequence of the metrics of the chosen keyword or topic is zero, we should start the time series data at the first date with a value greater than zero. Based on the length of the past time series data we have as a result, we should switch between (200,30) and (7,1) input-output sequences. An alternative Encoder-Decoder LSTM configuration can use the (7,1) sequences to predict future metrics for very recently emerged keywords which reflect more realistic numbers (otherwise we are guaranteed zero values), though we can expect as much accuracy as with our longer term time series forecasting approach.

Case 2: Anticipating Future Disruptions

Knowledge that we can have now about the future may reveal variations in future metrics of the keywords that cannot be anticipated by training a prediction model purely on the past data. The case of “elections” can be explored, with the example of a prediction task for the next 30 days (being November 1-30, 2020) based on training with past data from 13 months (October 1 2019 - October 31 2020). Since an election is typically every four years, for example, the training data might not even include the past relevant peak, and if there is an election in the prediction period, it can not be anticipated from that past data. This would be the case for the US election which took place on November 3, 2020.

In previous work, we looked at training the prediction model with relevant events in the past, so that the prediction could take into account the occurrence of an event in the future. While this demonstrably worked for a specific event - our example case was the “Eurovision” Song Contest - it is hard to generalise this approach. Firstly, what if the future event does not have similar events in the past period of the training data? Secondly, the effect on the keyword metric in the past does not directly correlate to a similar effect in the future for more general categories of events. While a specific event like Eurovision both occurs annually and naturally causes a peak in the synonymous keyword, “elections” provides a perfect counter-example. The change in the metric due to the occurrence of (some) elections in the past period of the training data will not likely correlate with any metric change due to the US election 2020, since such events can not be assumed to be comparable. This was confirmed also in tests where the same prediction model (using past events in training, then prediction of a period where another related event occurs) was used with sports-related keywords and could not improve the accuracy (e.g. “cycling” - not every past cycling event caused the same change in the metric

and the change in predicted metric was not more accurate). While filtering and ranking events might help, it is likely that this would always be very context-dependent and not possible to automate. We therefore consider as the alternative option the metrics available for frequency of documents which associate the chosen keyword with a future date (the "temporal reference detection" of the *Temporal Annotation* component).

Considering the keyword "elections" (which we used with RBB testing in D5.2 for the Brandenburg regional elections), there is clearly a peak in the temporal references to elections on the 3rd November (due to the US election). As the past US election took place four years earlier, it is outside of the training data for our time series forecasting. Therefore we can compare the 30 day prediction for the keyword based purely on the past frequency metrics as training data with the model including temporal reference metrics (which is available as a frequency metric for any date range, where n is the number of documents in the corpus which reference the date as a future date and are annotated with the keyword).

Standard forecasting: we use the frequency metrics from global English language news media for training, testing and validation (in the prediction time period). The training/testing dataset is 397 data points. The validation dataset has 30 data points and uses the 7-day moving average of the frequency metric. As expected, the predicted values - based on the "usual" news cycle without major election events - are much lower than the actual as the forecasting is unable to take into account the US election on 3 November (see chart below).

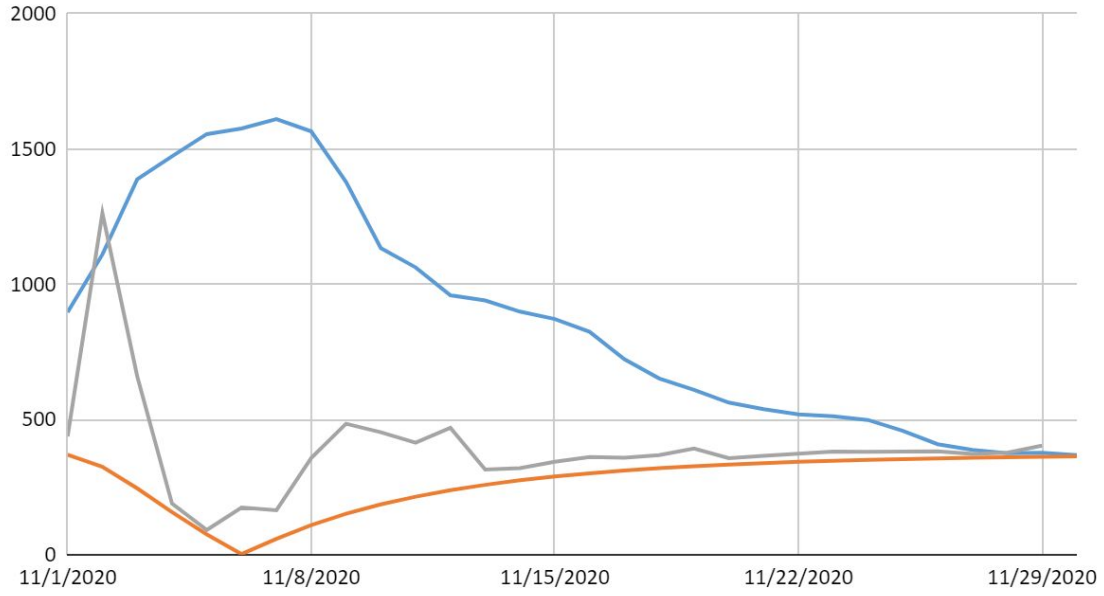


Figure 15: Predicted values without temporal references (orange), predicted values with temporal references (grey) and real 7 day moving average for "elections" (blue).

With temporal references: looking at the temporal reference frequency in November, there is a significant peak on 3 November (as we would hope) and generally a much larger association of the keyword with dates in the first half of the month (in the second half the numbers fall again to something more normal, varying between 18 and 40 - see the middle line of the chart below). Considering the integration of this feature, it is not relevant to include temporal references in the past data while training, since the references are created by detecting future dates in documents going back six months from the present (e.g. there would be a significant

lack of past data to use in training). Also we can not expect patterns in the temporal reference data as its main use is in detecting for us the out-of-trend peaks that can not be anticipated by forecasting. So rather we consider addition of temporal references to the predicted values, thus integrating peaks in temporal reference data into the usually much smoother forecasting.

Regarding the chart, the blue line is the real (7 day moving average) values for 'elections' in November. The orange line is the forecasting model predictions which clearly fall much lower (based rather on the 'usual' coverage of elections in the news media than on the 'out-of-trend' occurrence of the US election). The grey line in between is the sum of forecast values and temporal reference metrics. While still lower (it was also impossible to anticipate the longer period of focus on the election several days after election day itself due to ongoing counts and disputes) the peak on 3 November is clear (and interestingly, very close to the real value) and can now be considered in prediction.

We conclude that the temporal reference metrics can be valuable in combination with the forecasting model to incorporate out-of-trend peaks in keyword popularity that would otherwise not be part of our predictions. The Topics Compass uses for its prediction capabilities exclusively the temporal reference metrics as these can be calculated very efficiently and in a scalable fashion on the fly from the document corpus for either dashboard visualisations or the Prediction API. The time series forecasting using LSTMs is more resource intensive (predicting for the single "elections" keyword took 70 seconds on a latest-spec laptop with 1 GPU) and it is therefore not feasible to expect the same on-the-fly responsiveness. Such a prediction component can be maintained separate from the platform and used for ad hoc prediction calculations in advance when specific keywords need forecasting for a future date range (and the temporal reference metrics alone are insufficient).

3.5.2 Scalability Testing

Another important aspect of the TVP's evaluation concerns the assessment of its processing capacity and scaling out capabilities. It should be able to process a large number of documents at any given moment, and under heavy-load scenarios (e.g., a sudden increase in concurrent content to be processed in parallel), it should be straightforward to increase the computational capacity of the TVP to ensure uninterrupted operation without data loss, system failure, or negative impact on the usability of any of the ReTV applications. Processing capacity and scaling out capabilities were analyzed by adding load-test jobs using *Artillery*[15] in our CI pipeline that we can run on any version of an application.

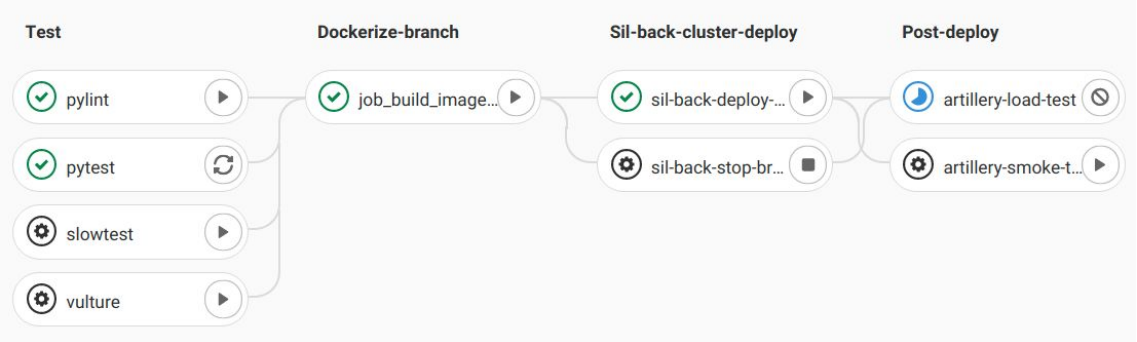


Figure 16: CI/CD pipeline where load tests using artillery.io can be manually triggered *post-deployment*

[15] https://artillery.io/

The evaluations of Figures 17-19 are based on the configuration shown in Listing 31. We computed sustained *Requests per Second* (RPS) while increasing the number of concurrent users. The evaluation focused on the previous computational bottlenecks of the ReTV content processing pipeline. This includes the entire NLP preprocessing stack (about 300 RPS as of December 2020), the knowledge extraction component (100 RPS) and the translation module (1000 RPS). The lower number of the knowledge extraction component reflects the throughput of the external API, its performance for internal computations is significantly higher.

The tests are continually performed to assess changes in implementation and architecture. The scalability testing also included scenarios to check the application throughputs when multiple replicas of the applications are deployed.

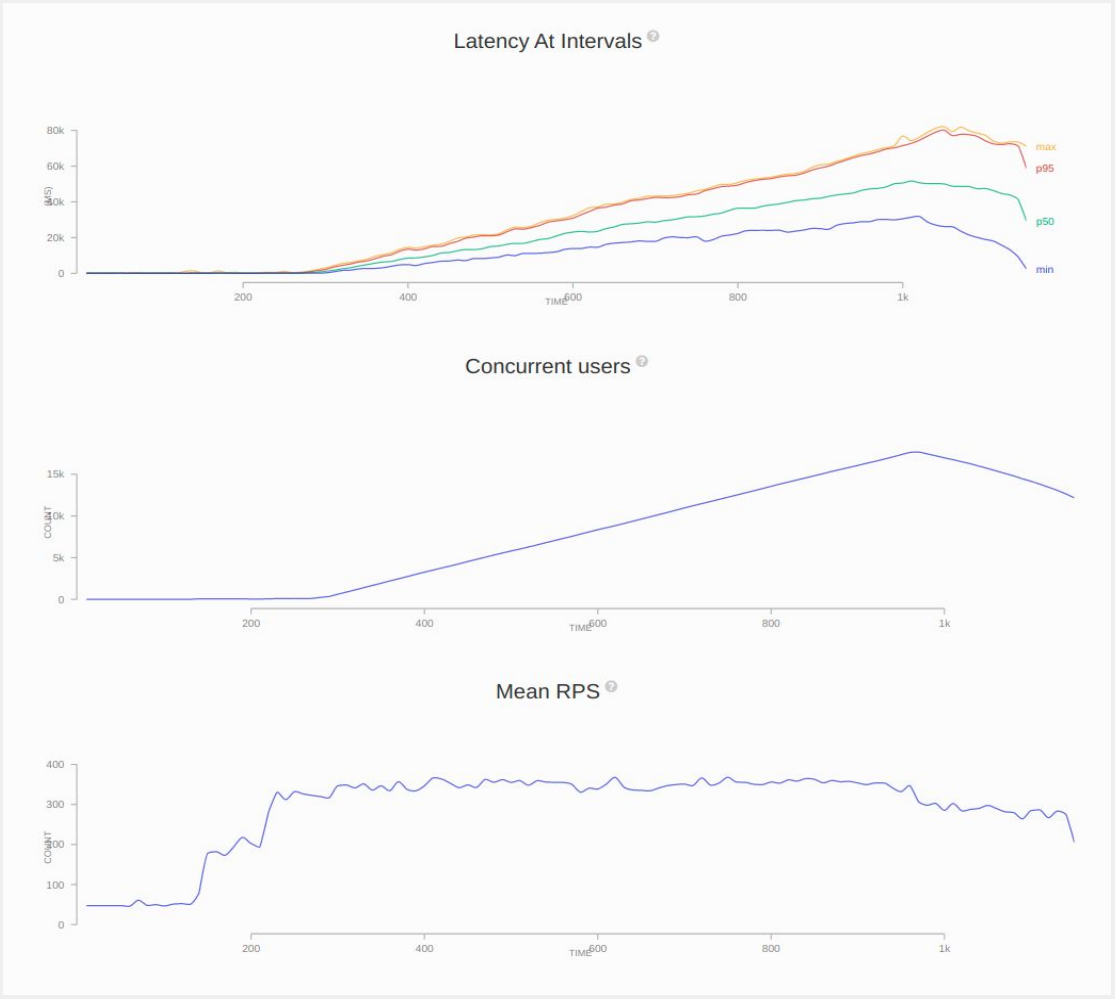
```
config:
  target: "http://skb-rest-translation.prod.i.weblyzard.net:8443/"
  phases:
    - name: Warm up
      arrivalRate: 5
      duration: 60
    - name: Ramp up
      arrivalRate: 5
      rampTo: 50
      duration: 300
    - name: Sustained load
      arrivalRate: 50
      duration: 600
      maxVusers: 2000
```

Listing 31: Load test warm up, ramp up and sustained load phase parameters
(each arrival sends 100 sequential requests, depending on the application being tested)

Year 3 of the ReTV project also brought a significant shift in our scalability strategy, triggered by new insights stemming from the Horizon 2020 big data project EVOLVE.[16] Instead of the more traditional approach of optimizing and evaluating each service in isolation (classified into *listening, prediction* and *adaptation* sub-systems), the migration to *Kubernetes*[17] allows a much more dynamic allocation of resources, making sure that it is easy to adapt to processing needs by adding additional nodes of commodity hardware. Kubernetes 1.19 is used for the Java and Python components of ReTV that are not inherently scalable - *Elasticsearch* 7.x[18] and *CockroachDB* 20.x[19] being the most notable exceptions that do not require this additional layer. Glusterfs[20] provides scalable storage to our Kubernetes applications (5x2TB at the moment, easily extendable). In terms of hardware, the Kubernetes cluster comprises seven physical nodes as of December 2020 (CPU: 32*5 + 40*2 cores = 240 cores; memory: 252*7 = 1764 GB). The cluster for Elasticsearch and CockroachDB comprises five nodes, to be extended to seven machines as well in the first quarter of 2021.

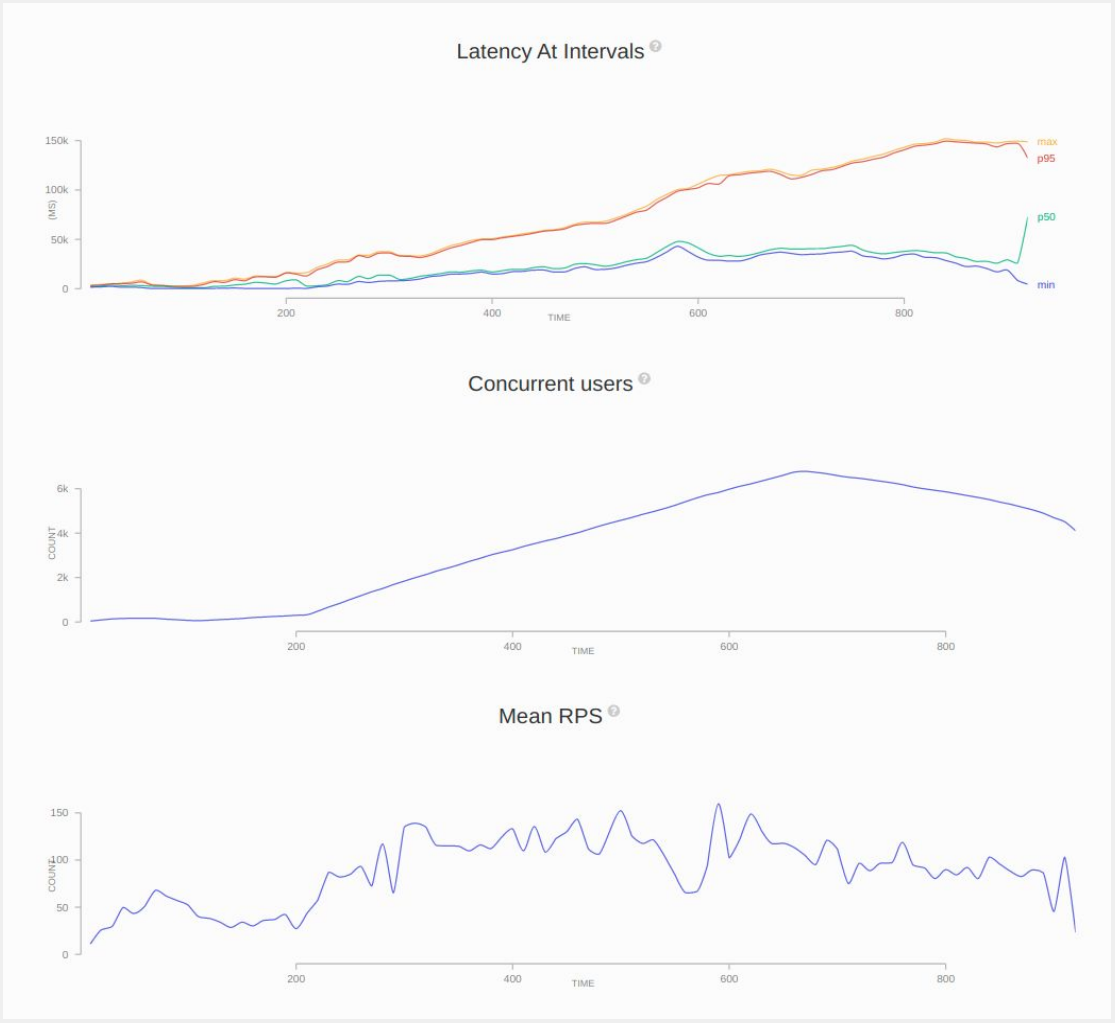
Additional insights were gained during the reprocessing tasks required in Q4-2020 to bring the metadata annotations of the entire ReTV knowledge repository in line with the latest versions of all modules. The English and German news samples, for example, comprised 300,000+ documents, which significantly exceeds the ongoing computational requirements.

[16] https://www.evolve-h2020.eu/
[17] https://kubernetes.io/
[18] https://www.elastic.co/
[19] https://www.cockroachlabs.com/
[20] https://www.gluster.org/




```
All virtual users finished
Summary report @ 02:04:42(+0000) 2020-12-30
  Scenarios launched:  38641
  Scenarios completed: 38641
  Requests completed:  386410
  Mean response/sec: 298.52
  Response time (msec):
    min: 3.5
    max: 76930.7
    median: 24062.8
    p95: 53581.2
    p99: 69783.1
  Scenario counts:
    submit_document: 38641 (100%)
  Codes:
    200: 386390
    502: 19
    504: 1
```

Figure 17: Load test results for the NLP preprocessing component




```
All virtual users finished
Summary report @ 03:25:33(+0000) 2020-12-30
  Scenarios launched:  10070
  Scenarios completed: 10052
  Requests completed:  100617
  Mean response/sec: 65.34
  Response time (msec):
    min: 134.1
    max: 151899.6
    median: 33068.1
    p95: 123798.1
    p99: 144847.8
  Scenario counts:
    annotate: 10070 (100%)
  Codes:
    200: 100617
  Errors:
    ESOCKETTIMEDOUT: 18
```

Figure 18: Load test results for the knowledge extraction component


```
All virtual users finished
Summary report @ 23:50:18(+0000) 2020-12-29
  Scenarios launched:  37299
  Scenarios completed: 37299
  Requests completed:  3729900
  Mean response/sec: 1706.06
  Response time (msec):
    min: 3.5
    max: 40066.3
    median: 6224.8
    p95: 29755.7
    p99: 33401.7
  Scenario counts:
    translate: 37299 (100%)
  Codes:
    200: 3729900
```

Figure 19: Load test results for the translation module

4 TVP FOR PROFESSIONAL SCENARIOS

Since Deliverable D4.1, where the revised prototype of the TVP was outlined, the use cases have been further refined through recurrent internal and external stakeholder feedback loops. The resulting professional user applications with clear focus on addressing the market needs developed from the use case scenarios. This section provides a brief overview of ReTV's two use case applications for the professional user, with a particular focus on how they are technically enabled through the TVP as outlined in the previous sections.

4.1 TOPICS COMPASS

Topics Compass is the flagship version of a Data Exploration and Visualisation System used for the TV content and news metadata and its analyses within the TVP. As such, the *Topics Compass* uses the TVP metadata analysis workflow and drives the definition and configuration of all content parameters used in the metadata visualisations that are shared via API with the *Content Wizard*. Whereas the pre-tailored visualisation components as integrated into the *Content Wizard* via the WLT Visualisation API are limited in their exploratory functionalities, the *Topics Compass* allows for full-fledged, multi-modal context explorations over arbitrary topics, date ranges and across languages including English, German, Dutch and French. The *Topics Compass* is a JavaScript-based Dashboard implementation that is directly connected to the WLT Search Index (ElasticSearch), providing full access to the WLT Metadata Repository as well as the MOD Prediction Service. The advantages of *Topics Compass* over the integrated visualisation in the *Content Wizard* are its exploratory capabilities, supporting drill down operations on specific aspects of a story. The context of the story is shown within the public debate - by associated keywords, observed agreement and disagreement, as well as entity and geolocation co-occurrence patterns.

4.1.1 Dashboard Structure

The main content area of Topics Compass provides a hierarchical menu with five main categories – *Documents, Sentences, Sources, Entities* and *Relations*. Each of these categories contains at least one list view and one visualization (e.g. word tree for sentences or scatter plots for entities and sources).

The colours are assigned dynamically based on the selected *bookmarks, associations* or *metadata attributes* such as *"Sources - Aggregated"* shown in Figure 20. In the final M36 version of Topics Compass, this also applies to the WYSDOM chart. The M24 prototype still used a hard-coded list of categories including predefined colours for desired / undesired associations. This increases the flexibility of the component and avoids having to cross out the checkboxes of other sidebar elements while in WYSDOM mode.

4.1.2 TVP Integration

The *Topics Compass* is the front end system to the metadata repository, abstracted by the metadata search index for fast aggregations. It uses the visualisation engine for rendering of the analytical charts. Also, the SKB provides entity-related metadata for the visualisation engine in the form of entity thumbnails and internationalised label and descriptions. Another integration for the *Topics Compass* comes from the textual summarisation and prediction services.

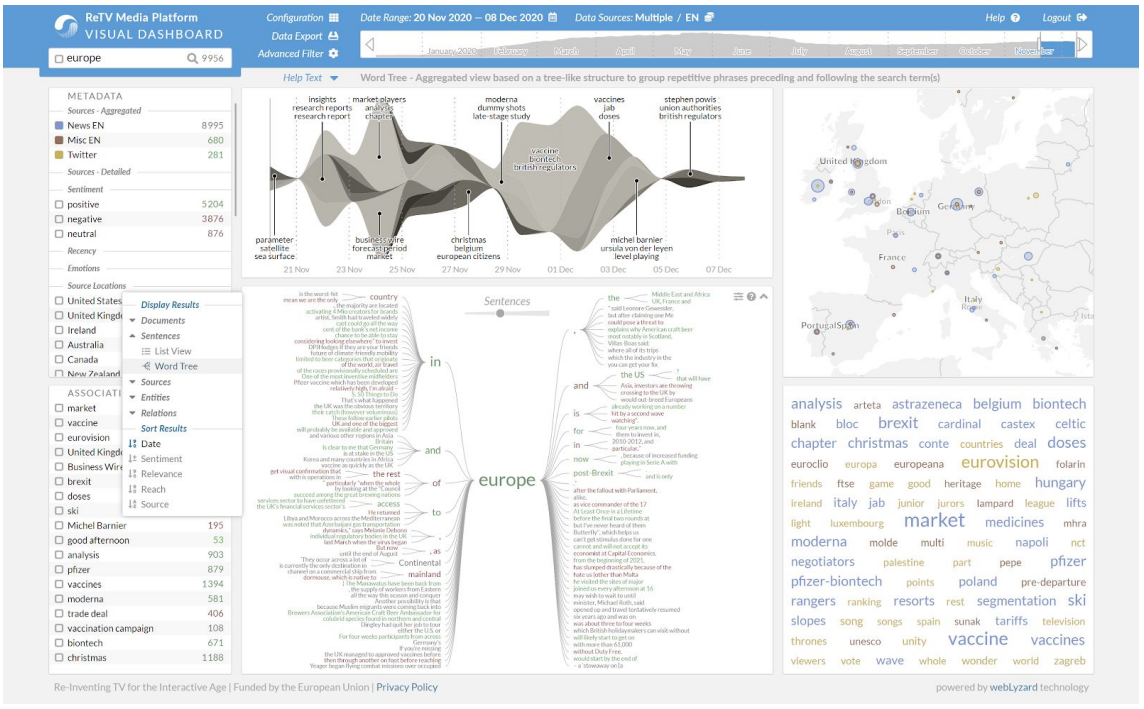


Figure 20: Topics Compass, one of the TVP applications, with a query on "Europe" between 20 Nov and 08 Dec 2020 to highlight the successful integration of the TVP components

The Topics Compass benefits from the TVP as follows:

- **Prediction Mode** opens new doors to the analytical capacity of the Topics Compass, accessible via the "date processing" selector to switch between *listening* and *prediction mode*. The latter is the new default mode and uses date references found in the documents instead of publication dates to project documents along the temporal axis in the trend chart, unlocking the ability to apply the full analytic capabilities of the dashboard to future time frames.

- **Text Summarisation**, the Topics Compass makes use of the Text Summarisation service to better abstract news article content into relevant content. It uses "Relevance" as the variable to determine the selection of sentences to be included in the summary (as compared to the *Storypact Editor*, which provides users with the option to select among several variables including *Sentiment* and *Google Search Popularity*).

- **Semantic Knowledge Base**, with the integration of the SKB into the Visualisation Engine, the Topics Compass gains the possibility of a whole new level of metadata overlays from the LOD space to elevate its keyword-centric visualisations into real-world entities.

- **SKB Viewer and Editor.** Complementing the dashboard, this separate user interface delivered by MOD (WP1) allows *Topics Compass* users to search for entities in the Elasticsearch index (which is synced with entities in MOD's SKB), supporting filtered search for entity type, time range, anniversary date, as well as property types and property values, and allows modifying thumbnails in addition to property values in multiple languages.

4.2 ReTV Content Wizard

The *ReTV Content Wizard* is the user-facing application that builds upon all aspects of the TVP, integrating results from both the metadata and the binary analyses. As such, it directly or indirectly makes use of the WLT Content Recommendation API, the WLT Search and Visualisation APIs from the metadata workflows, as well the Genistat Recommendation and Scheduling API and the CERTH Video Adaptation and Repurposing API from the binary workflows. The integration of the ReTV Content Wizard into the TVP is shown in Figure 12.

In particular, the following functional aspects are provided by the TVP in the form of microservice API requests from the *Content Wizard*:

- **Text Summarisation**: Smart, textual content creation support and impact optimisation, based on trends from the public debate (i.e. metadata context) and prediction.
- **Visualisation Engine**: rich, interactive trend visualisations and top trending keywords derived from the public debate (i.e. metadata context) and prediction.
- **Recommendation & Scheduling**: Smart video scheduling and adaptation for multi-vector impact optimisation.

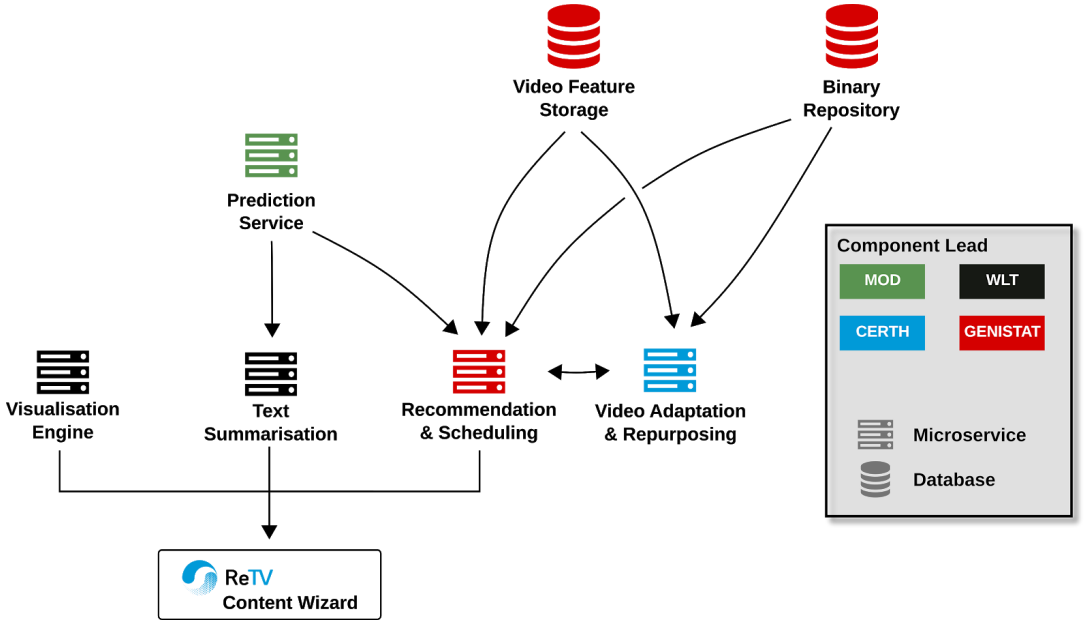


Figure 21: The TVP configuration of the ReTV Content Wizard.

Integration with the WLT Search API

The WLT Search API allows to query the WLT metadata repository composed of multi-modal television-specific as well as general-interest content collected by MOD in an expressive and efficient way. The API identifies the top stories, their opinion leaders, and the most sought-after keywords occurring in the current debate, but also, thanks to the prediction capabilities provided by partner MODUL, in the near future. This allows content creators such as users of the Content Wizard to guide their idea creation and content refinement process towards more effective publication turnover.

Each search result (stories, opinion leaders, keywords) is queried with precise date range selectors provided in the application to the user's choice. The results are presented as ranked lists *Top Stories*, *Opinion Leaders*, and *Top Keywords*.

Integration with the WLT Visualisation Engine

All visualisations that make up the *Topics Compass* are available as embeddable widgets via the WLT Visualisation API. In particular, this interface allows for the integration of contextual metadata analytics including prediction into the use case application *Content Wizard*. Requests against this API allow precise configurations regarding the data sources, search queries, as well as data ranges to be defined. In addition, the WLT Visualisation API further broadens the context shown in a visualisation widget via similarity search based on specific URIs. This is useful if the seed of the explorative context analysis is provided by a single document rather than a topic definition or search terms.

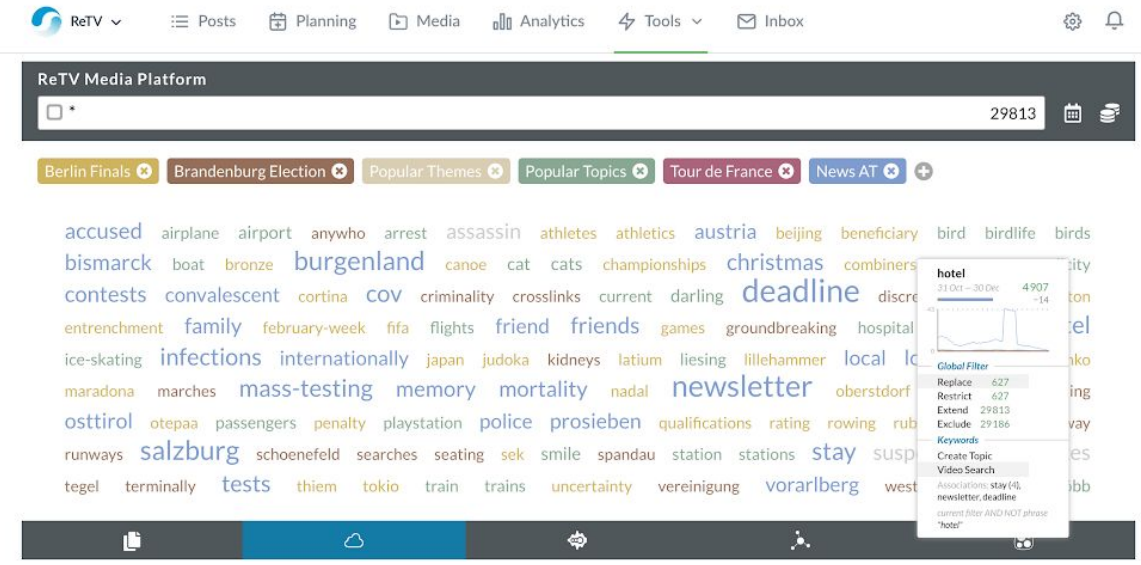


Figure 22: Integration of the WLT Topics Compass into the planning view of the *Content Wizard*.

WLT Text Editor

The WLT Text Editor refers to embeddable UI elements for the text summarisation component that will be used in the Content Wizard. Based on *TinyMCE* editing platform (www.tiny.cloud), it is composed of a stand-alone Web application and a rich set of RESTful Web Services (APIs) for TVP integration. The editor is a content optimisation, text summarization and ambient search tool that helps content creators maximise the impact of their textual content across multiple vectors (press releases, Twitter, Facebook, blog, etc.) by suggesting lexical, stylistic and contextual text replacements that can increase the impact of publications. The replacement suggestions are determined by measuring the impact of past publications and observing trending stories as well as keywords and references. The MOD prediction service will provide additional evidence to fine-tune the data-driven content recommendations. As part of its individual exploitation plan, WLT plans to release a stand-alone version of the editor under the name *Storypact*.

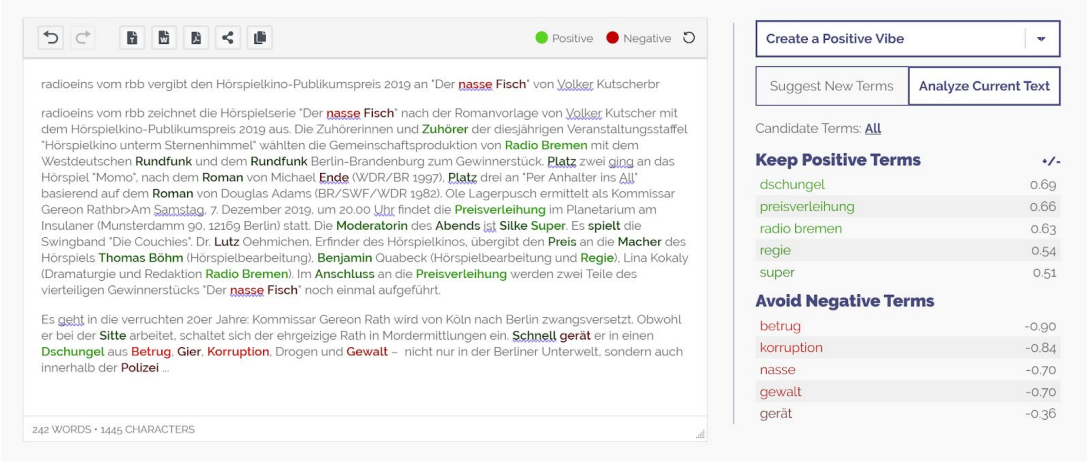


Figure 23: Screenshot of the WLT Text Editor, showing an annotated press release and sentiment analysis to highlight positive and negative terms.

Up until now, we have conducted two evaluation cycles for the editor component, initially in February 2020 (n=5) and based on a revised version in November 2020 (n=10). A third evaluation cycle is planned for the first quarter of 2021, prior to presenting the tool to a larger number of third-party users. The results in Figure 24 based on ten test users reflect the quality of the text summarisation sub-module, as it clearly outperforms the other editor components including the *suggestions of new terms*, the *exploration of related content* and the *analysis of the current text*.

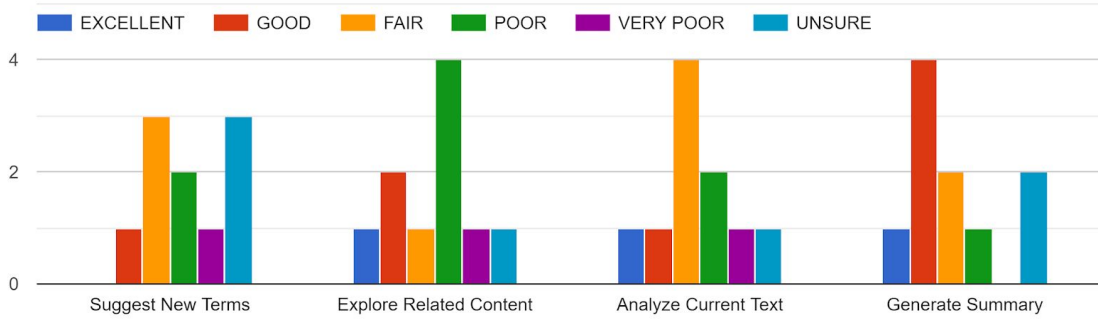


Figure 24: Results of the Evaluation of the Storypact components

Integration with Video Adaptation & Repurposing

When the user selects a video for editing in the Content Wizard, they are given the option to have it automatically summarised (see Figure 25). The summarisation is then requested from the Video Adaptation & Repurposing service. Since the video analysis is run in advance, storing extracted features in the Video Feature Storage, the summarisation is almost instant. Figure 25 also shows how the video editor cuts the video into the scenes that the Video Adaptation & Repurposing proposed. The user can still make manual adjustments if so desired.

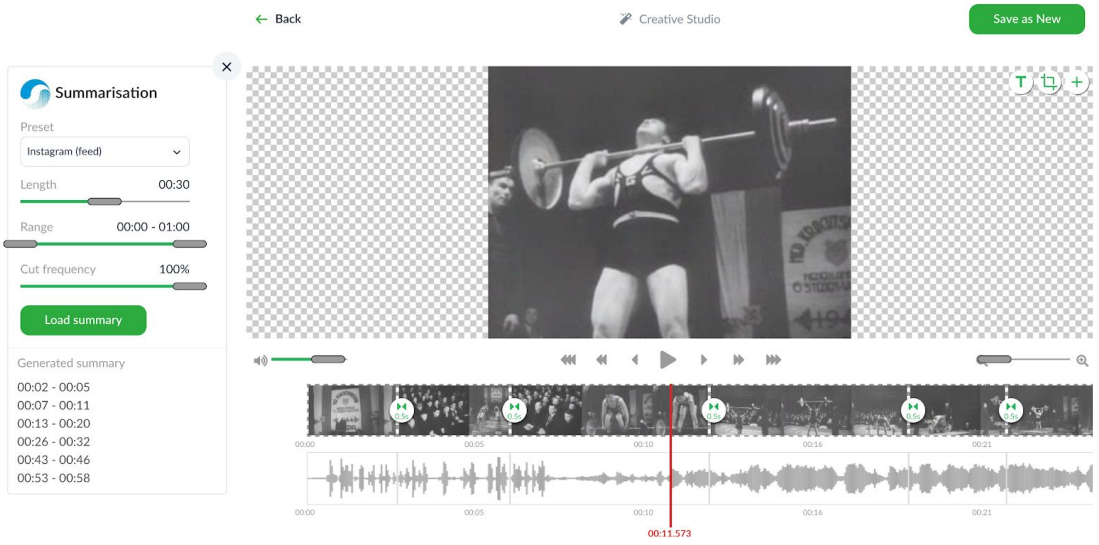


Figure 25: "Load Summary" button for a video in the *Content Wizard*. Different presets are supported.

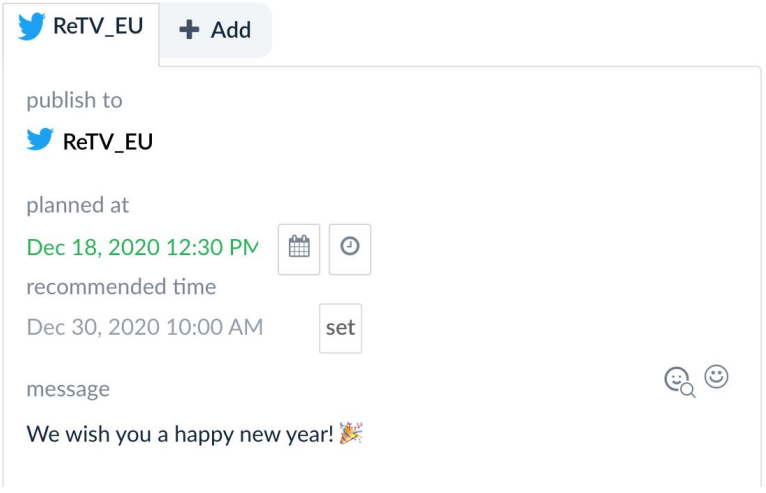


Figure 26: Choosing a publication time for a social media post.

Integration with Recommendation & Scheduling

When publishing a video or any other type of content to social media, the Content Wizard allows the user to schedule the post (see Figure 26). The Content Wizard now calls the Recommendation & Scheduling API to get the ideal date of publication.

Integration with Events and Anniversaries API

The Content Wizard is now integrated with the Events and Anniversaries API and displays relevant events in the calendar view.

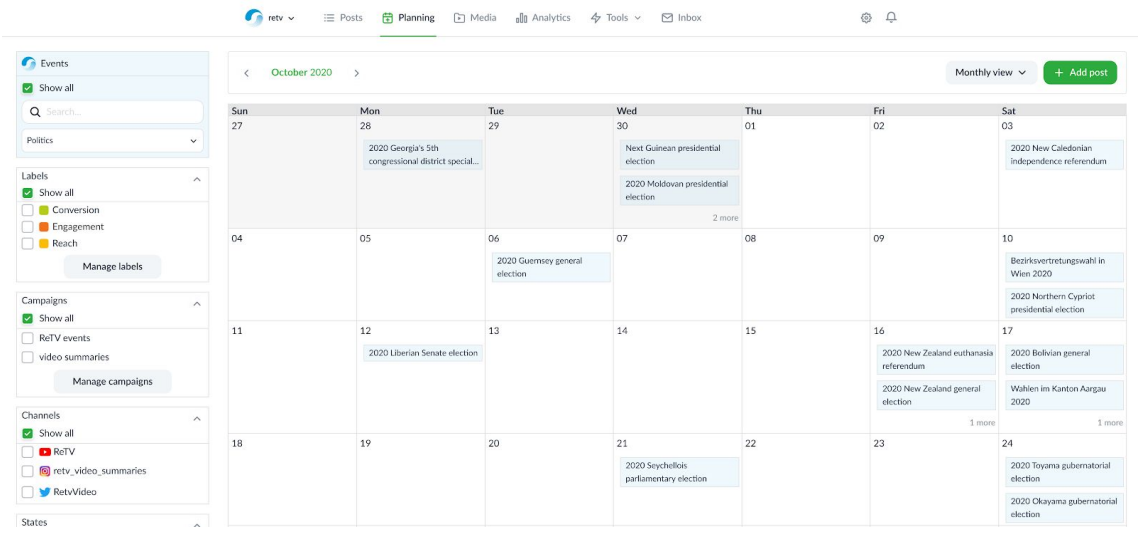


Figure 27: Events in the planning view of the Content Wizard.

The events are specific for either RBB or NISV: A "template" field indicates for which use case partner the events should be displayed. This template field is set on a project basis in the Content Wizard.

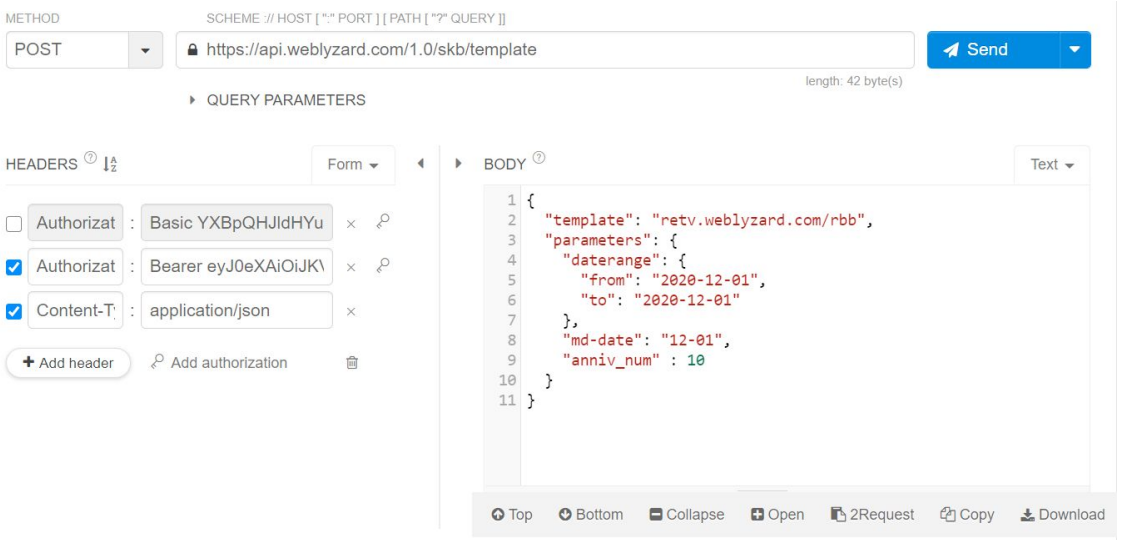


Figure 28: Example call to load events for RBB.

Integration with the Text2Video search

When clicking on an event in the planning view or a topic in the TopicsCompass integration the text to video search is triggered using the calls described in Section 3.3.1 of D3.3. Videos matching the event or topic are shown to the user in order of relevance. Clicking on one of the videos allows the user to directly post the video on a content vector or automatically summarize it before doing so. The vector embeddings of the Text2Video service are stored in an optimized[21] Elasticsearch index, allowing for fast querying of the high-dimensional data.

Extension of the Scheduler to upload and delete videos into the Topics Compass

[21] https://www.elastic.co/guide/en/elasticsearch/reference/current/dense-vector.html

The Scheduler component now also uploads videos from RBB and NISV to the Topics Compass. The upload process also entails running the videos through the Video Adaptation & Repurposing service and storing the results in the feature storage. In the case of RBB, the videos are deleted automatically after seven days due to legal requirements.

5 TVP for Personalisation

Similar to the professional user scenarios, ReTV has also developed specific user scenarios for the prosumer user in the form of personalisation prototypes. In particular, the two use cases *4u2 Messenger* and *Abendgruss* address the need of the consumer to optimise their production workflows within a fast-paced environment of vectors and formats, by easing the process of delivering personalised, on-time media content to consumers. In Year 3 we added the "4u2 Messenger" use case which replaced the Content sWitch.

5.1 4u2 Messenger

The goal of the 4u2 Messenger use case is to surface video content relevant to each unique viewer. Based on the results of user research, we decided on chatbots and smart speakers as a publication vector. The first version of the messenger was built using the *Rasa* chatbot framework to abstract from different types of messengers (e.g. Telegram and WhatsApp). In the first iteration, we built two chatbots, one for NISV and one for RBB. They shared the majority of their code, but were configured differently.

Year 3 saw a complete reimplementation of the 4u2 Messenger for NISV. The Rasa chatbot framework proved to not be up to our specific requirements and proved too time-consuming to develop for. We chose to replace Rasa with botkit,[22] which allowed us more fine-grained control. Figure 29 shows the final architecture of the 4u2 Messenger. Users chat in the Facebook Messenger application. However, this could easily be replaced with other messaging applications like Telegram.

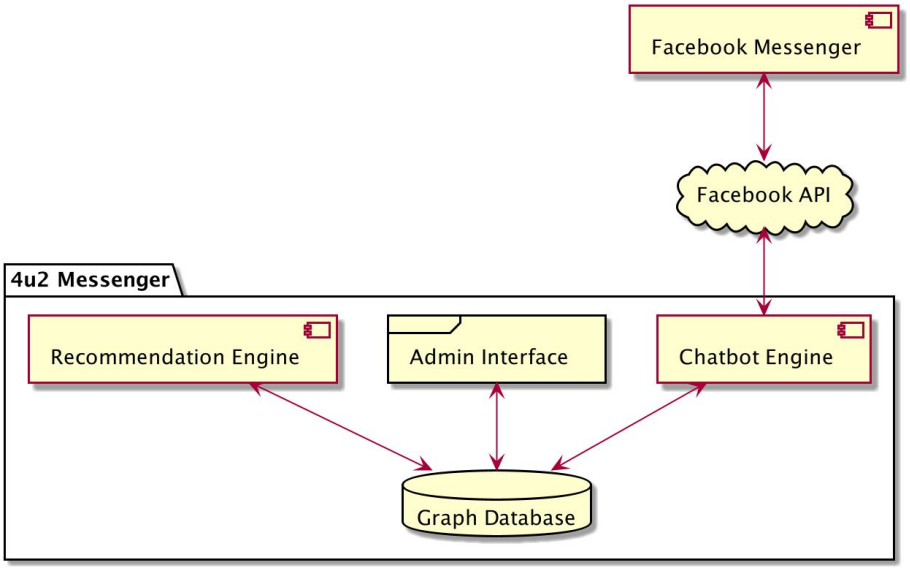


Figure 29: Architecture of *4u2 Messenger*.

[22] https://botkit.ai/

We can send users messages through the Facebook API. We receive their messages and events through the API as well. The chatbot engine deals with parsing and sending the messages and is based on botkit. We store messages and videos in a graph database. This graph database is also accessed by the admin interface, allowing editors to manually create messages and monitor the automated messages sent by the chatbot recommendation engine. The recommendation engine learns from user behaviour and balances delivering popular videos and serendipity. We extended the Scheduler component to be responsible for sending the personalised messages that Recommendation Engine of the 4u2 Messenger creates.

In terms of scalability we started using the Faiss[23] library, which allows us to produce the recommendations for thousands of users very quickly.

5.2 Abendgruss

The Abendgruss use case was developed in Year 3, fitted into the existing TVP architecture without the need to develop new services, demonstrating the flexibility and extendability of our API framework. It uses the CERTH VA service to analyze and segment episodes of the "Abendgruss" TV show for children.

While the use case will be described in detail in D6.3, we would like to highlight a particular innovation with regards to scalability. The Abendgruss use case creates personalised videos for every user, combining different intros and main segments. Encoding this as a new video is time-consuming, often taking as long as the actual video itself. This is not an option for an app that should start playing a video as soon as the user has finished their choice.

We therefore deliver the Abendgruss episodes as HLS streams in M3U8 playlists, which list the different segments that need to be displayed, but leave the correct stitching to the video player. The stitching happens seamlessly thus removing the need for re-encoding the video.

6 Conclusion And Outlook

This deliverable presents the final version of the TVP, following the initial description in M12 (D4.1) and the revised prototype in M24 (D4.2). At M36, the TVP is fully functional. All components of the target architecture have been integrated, optimized and thoroughly tested.

The focus of the WP4 work in Year 3 of the project, as summarized in Section 1.1, elevated the TVP from early proof-of-concept configurations into a stable and commercially viable technical platform, with particular focus on high-availability and rapid scaling out capabilities. Platform evaluations, both from a technical perspective and a usability perspective (in collaboration with WP5 and WP6) have helped us to refine the downstream subsystems *Prediction API* and *Recommendation / Scheduling* - which are available as a stand-alone version via REST APIs, as well as integrated into the use case applications.

As the ReTV project concludes and the technical integration has been completed successfully, the focus will now shift to the project partners' joint and individual exploitation plans. Outlined in Section 4 of Deliverable D7.4, the partners are committed to the continued maintenance and evolution of the platform and anticipate a strong demand for TVP-enabled services and applications. The achieved modularity and flexibility of the platform will be a major asset when adapting the products to changing customer needs, or when reconfiguring the workflow to rapidly deploy new services for yet unforeseen business opportunities.

[23] https://github.com/facebookresearch/faiss

Appendix A: WP1 Components Overview

Component	Final Version
EPG	The current version loads EPG data from Zattoo and stores it in an internal database of Genistat. The top 10 TV channels in Germany and Switzerland are currently loaded. Those are ARD, ZDF, Prosieben, RTL 2, Sat 1, SRF 1, SRF zwei, VOX, ZDF. Additionally, we also push the EPG data of consortium partner RBB. We decided to focus on those channels, as their popularity means that there will be a significant amount of Zattoo audience data to match to the content.
Binary repository	The binary repository is being fed with the video data from ARD, ZDF, RBB, Prosieben, SRF 1 and SRF zwei.
Scheduler	The scheduler is *EPG* data to the *Metadata Repository.* The scheduler can also push audience data from *Audience Metrics* to the *Metadata Repository.* This feature is currently turned off.
Video Fragmentation and Annotation	More accurate and compact DNNs based on network pruning techniques, to improve the accuracy and computational efficiency of video annotation, as reported in Section 4.2.2 of D1.2. New set of video analysis methods for Sandmannchen episodes for the support of Abendgruss use case scenario. More specifically, domain-specific fragmentation and annotation techniques were developed, as reported in Section 4.2.2 of D1.3. Software optimization for enabling ingesting large video files of over 2 hours duration, as reported in Section 4.1.1 of D1.3.
Brand Detection	New car brand detection method, as reported in Section 5.4 of D1.3. This module was implemented in order to support a specific use case scenario. Further expansion of the set of detectable Brands, as reported in Section 5.2 of D1.3. Introduction of a merged objection detection model to improve processing time and memory footprint, as reported in Section 5.3 of D1.3
Crawler	Crawling has continued as planned. An extended list of Web sites, particularly TV/Radio Web sites as well as 'hybrid' (both news and TV/radio related content), is now supported. Dutch content was added and the Dutch language now supported in the NLP part of the pipeline. Our full end-to-end data ingestion pipeline is now customizable to new domains as required by stakeholders. Data source collection is supported by an online spreadsheet. Source management is facilitated by a new source categorization (e.g. marking sources of interest to RBB's Radio Fritz as "youth").

	Our NER/NEL annotation capability is iteratively improved and remains state of the art, supported by an error classification approach and a visual error correction interface called ORBIS. We have improved annotation accuracy for the entity type Works (e.g. TV program titles) and introduced a new entity type TermEntity which supports the correct entity-keyword labelling of other entity types (e.g. events such as music festivals) while avoiding incorrect translations (like the "Tomorrowland" festival in English being translated to "Morgenland" in a German interface)
Semantic Knowledge Base	The knowledge base has been extended by new entities as required by the use cases, e.g. for specific holidays or events. Stakeholders can indicate specific entities to add via an online spreadsheet. The scope of Person entities was extended significantly (births before 1900). Event entity metadata was improved with the addition of a categorization (e.g. sports) and short descriptions (based on Wikipedia abstracts). Location entities were also expanded significantly by including lower granularity locations such as streets (in the DACH region). We released the Web interface to the KB that allows adding, checking and correcting entity descriptions, allowing expert end users to curate available entities. It includes an advanced search interface and a means to suggest new entities for the SKB.
Video Feature Storage	Features extracted from *Video Annotation and Fragmentation, Text-to-Video embeddings* and *Brand Detection* are stored as binary files to speed up the creation of video summaries.

Appendix B: WP2 Components Overview

Component	Final Version
Event Extraction	Events from WikiData and selected iCals are extracted and stored in the *Semantic Knowledge Base* on a regular basis. Event descriptions are now added and all metadata is also updated from the same sources at regular intervals. The component can be adapted to extract events from other knowledge graphs, calendars or other sources on request from a stakeholder.
Events and Anniversaries API	Events as well as anniversaries matching a certain date are returned, using the SKB as data source. The final version includes support for templates which can be defined on a case by case basis to adapt the returned events or anniversaries to the interests of different stakeholders. Also the scope of metadata returned has been expanded. The component can be customised to stakeholders through the definition of new templates to match events and anniversaries.
Temporal Annotation	Temporal reference detection in documents was evaluated, refined and re- implemented (both absolute and relative references - it is now connected to the SKB to associate events to dates). It is available to the *Prediction Component.* It has also been expanded to use the entire multilingual News document corpus in the *Metadata Repository*. The component can process other textual document repositories as required.
Content-based Success Metrics	Trend detection for the success metrics based on historical audience and viewer data. Improved keyword extraction with a special focus on compound nouns and part-of-speech validity checks. Daily intervals replaced by a shorter timespan granular enough to track the impact of short-term interventions, e.g. changes in advertising strategies. Success factors calculated based on multiple emotional categories (in addition to bipolar sentiment annotations) and measures of disagreement calculated for arbitrary time intervals and with an on-the-fly reconfiguration of weights.
Genistat Audience Metrics	The Zattoo audience data for Switzerland and Germany is being aggregated into 5-minute slices and then pushed to the *Metadata Repository*
Prediction Service	A Prediction API makes the hybrid prediction model available to stakeholders, as well as part of the Content Wizard. The final version incorporates the following as separate prediction models: 1. prediction of audience metrics based on past audience and including content and event features for improved accuracy

	2. prediction based on historical data (i.e., past success metrics) and including temporal reference (3, below) and event (4, below) features for improved accuracy 3. prediction of future keyword popularity based on the *Temporal Annotation* component 4. prediction of future events of relevance based on the *Event Extraction* component

Appendix C: WP3 Components Overview

Component	Final Version
Metadata Model	The metadata model is described in *D3.2,* encompassing datasets collected and extracted in WP1 and WP2. The Elasticsearch cluster of WP4 is used to store the data.
Recommendation & Scheduling	Recommendation engine using the Text-to-Video embeddings used in the 4u2 Messenger. Described in Section 3.2 of D3.3. Recommendation and scheduling of videos in the Content Wizard. Described in Section 3.3 of D3.3. Recommendation of ideal texts in the Content Wizard described in Section 3.4 of D3.3.
Video Adaptation and Repurposing (aka Video Summarization)	Introduction of learning-based video summarization, using Generative Adversarial Networks, as reported in Section 2.1.2 of D3.3. Joint consideration of editor-specific rules (supported by video annotation results generated by the WP1 components) and learning-based summarization for generating the final video summaries, as reported in Section 2.1.2 Section of D3.3. Smart-cropping for retargeting video summaries to different aspect ratios, as reported in Section 2.2 of D3.3. Support for different video summarization profiles, as reported in Section 2.4.1 of D3.3. Support of text-query-driven video summarization utilizing the Text-to-Video embeddings. Software optimization for speeding-up the video summary generation process, specifically a multi-threaded video read and decode workflow, as reported in Section 2.4.1 of D3.3.
Text-to-Video Embeddings Extraction	Extraction of Text-to-Video embeddings from video and text strings, using a dual encoder attention network, as reported in Section 2.3 of D3.3. Introduction of a new REST endpoint on a separate server for real-time extraction of Text-to-Video embeddings from text strings, as reported in Section 2.4.1 of D3.3.