D2.2 System Architecture

<table>
<thead>
<tr>
<th>Grant Agreement nr</th>
<th>611057</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project acronym</td>
<td>EUMSSI</td>
</tr>
<tr>
<td>Start date of project (dur.)</td>
<td>December 1st 2013 (36 months)</td>
</tr>
<tr>
<td>Document due Date :</td>
<td>31 May 2014</td>
</tr>
<tr>
<td>Actual date of delivery</td>
<td>6 June 2014</td>
</tr>
<tr>
<td>Leader</td>
<td>LUH</td>
</tr>
<tr>
<td>Reply to</td>
<td><a href="mailto:herder@l3s.de">herder@l3s.de</a></td>
</tr>
<tr>
<td>Document status</td>
<td>Submitted</td>
</tr>
</tbody>
</table>

Project co-funded by ICT-7th Framework Programme from the European Commission
Abstract
This task will define the overall architecture of the system based on the requirements as well as the individual components to be developed in WP3 and WP4. The architecture description will comply with ISO/IEC/IEEE 42010 Systems and software engineering — Architecture description standard. The architecture will describe all functional, logical, structural, physical, process, user action and data model of the system (system components, clients, delivery methods, etc.). UML diagrams will be used to specify, visualize, modify, construct and document the components of the system such as: activities, actors, business processes, and database schemas.

Keywords
Circulated to partners
Yes
Peer review completed
Yes
Peer-reviewed by
Jens Grivolla (UPF)
Coordinator approval
Yes
Table of Contents

1. BACKGROUND ................................................................. 5

2. INTRODUCTION ............................................................... 6
   2.1. Main Objectives and Goals ........................................... 6
   2.2. Methodology ........................................................... 6
   2.3. Overview ............................................................... 7

3. REQUIREMENTS AND STAKEHOLDERS .................................. 8
   3.1. Requirements from the Description of Work ....................... 8
   3.2. Summary of user requirements ..................................... 9
       3.2.1. Use case 1: Contextualizing Tool ............................. 9
       3.2.2. Use case 2: Second Screen .................................. 10
   3.3. Non-functional requirements ....................................... 12
       3.3.1. Maintainability .................................................. 12
       3.3.2. Robustness ...................................................... 13
       3.3.3. Security .......................................................... 13
       3.3.4. Easy to use ..................................................... 13
       3.3.5. Trustworthiness ................................................ 13
       3.3.6. Portability ....................................................... 13
       3.3.7. Development costs .......................................... 14
       3.3.8. Operational costs ............................................ 14
       3.3.9. Functional scalability ....................................... 14
   3.4. Stakeholders ........................................................... 14
       3.4.1. Roles of consortium partners ................................. 14
       3.4.2. Use-case stakeholders ....................................... 15
   3.5. Concluding remarks ................................................ 17

4. ARCHITECTURE DESCRIPTION ........................................... 18
   4.1. Initial sketches and selection of components ..................... 18
       4.1.1. First iteration (proposed by L3S) ......................... 18
       4.1.2. Second iteration: integration of EUMSSI with VSN .......... 19
       4.1.3. Third iteration: Processing architecture (proposed by UPF) 21
   4.2. Overview of the current architecture ............................... 22
       4.2.1. Diagram of components ..................................... 22
       4.2.2. Process flows ................................................ 26

5. ARCHITECTURE VIEWS – SYSTEM FUNCTIONALITY .................. 34

EUMSSI_D2.2_System Architecture
5.1. Selection of viewpoints .................................................................34
5.2. The context viewpoint .................................................................34
5.3. The information viewpoint ..........................................................35
5.4. The development viewpoint .........................................................36
5.5. The deployment viewpoint ..........................................................37
5.6. The operational viewpoint ...........................................................38
5.7. Concluding remarks ....................................................................38
6. CONCLUSION: REQUIREMENTS, COMPONENTS, PLATFORMS AND NEXT STEPS.39
6.1. Requirements ..............................................................................39
6.2. Individual components ...............................................................39
6.3. Platforms used and hardware requirements ...................................40
6.4. Next steps ..................................................................................40
7. References ....................................................................................41
8. LIST OF FIGURES .........................................................................42
9. GLOSSARY ..................................................................................43
1. BACKGROUND

This deliverable describes the current view on the system architecture design for the EUMSSI framework. The corresponding task in the DoW is described as follows: this task will define the overall architecture of the system based on the requirements as well as the individual components to be developed in WP3 and WP4. The architecture description will comply with ISO/IEC/IEEE 42010 Systems and software engineering — Architecture description standard. The architecture will describe all functional, logical, structural, physical, process, user action and data model of the system (system components, clients, delivery methods, etc.). UML diagrams will be used to specify, visualize, modify, construct and document the components of the system such as: activities, actors, business processes, and database schemas.
2. INTRODUCTION

The main goal of this deliverable is to describe the overall system architecture of the EUMSSI system and its individual components.

2.1. Main Objectives and Goals

The architecture description provided in this document serves several goals, including:

- Offering a basis for (formal) agreement about the form of the system to be built. The architecture description is the result of various iterations and discussions with consortium partners, either plenary or on a one-to-one basis. This document should serve as a reference point for mutual understanding.
- Validating assumptions on expected functionality and main priorities, as well as on the responsibilities of each partner with respect to the modules to be developed.
- Providing management decision points by making important choices and their implications explicit.

This deliverable is the result of the activities defined in Task 2.2 of the Description of Work:

```
This task will define the overall architecture of the system based on the requirements as well as the individual components to be developed in WP3 and WP4. The architecture description will comply with ISO/IEC/IEEE 42010 Systems and software engineering — Architecture description standard. The architecture will describe all functional, logical, structural, physical, process, user action and data model of the system (system components, clients, delivery methods, etc.). UML diagrams will be used to specify, visualize, modify, construct and document the components of the system such as: activities, actors, business processes, and database schemas.
```

2.2. Methodology

In line with the Description of Work, this document follows directions of the ISO 42010 standard, which aims to enable the expression, communication and review of architectures of systems by defining standard terminology and a number of aspects that need to be covered

1. Architecture description identification and overview
2. Identification of stakeholders and concerns
3. Selection of architecture viewpoints
4. Architecture views
5. Consistency and correspondences among architectural views
6. Architectural rationale

Our approach and the selection of viewpoints is largely inspired by (Rozanski & Woods, 2012), who discuss in detail the process of software architecture design and the use of viewpoints, perspectives, styles and patterns.

---


EUMSSI_D2.2_System Architecture 6
By following a formal approach, we aim to motivate the main design decisions by relating them to the stakeholder concerns, and to ensure that functional and non-functional requirements are being fulfilled.

However, at the same time, this document is kept concise on purpose. As pointed out by (Rozanski & Woods, 2012) in research projects like EUMSSI, architecture work is often seen as unnecessary overhead – with as a consequence that the consortium partners might take the architecture description for granted and ignore the directions given and concerns raised.

For this reason, we aim for a minimal architecture description for the most important aspects of the architecture, focusing on a small number of key views to address the key risks that the project might face. In addition to the formal UML diagrams, more informal diagrams will be provided to better communicate the ideas to our consortium partners. Further, wherever possible, we will ‘translate’ the architecture view into concrete decisions or proposals with respect to the chosen components, clients and delivery methods.

2.3. Overview

In Chapter 3, we discuss the requirements with respect to the system, given by the main goals of the project and the use cases, as stated in the description of work. After a summary of the non-functional requirements, we end the chapter with an overview of the stakeholders that are related to the EUMSSI framework: the consortium partners and the end users, as described in the use cases.

We start Chapter 4 with a summary of the design iterations and discussions that led to the current system architecture design. In section 4.2, we describe the main components of the system, provide details of the constituent parts and the way they are related, and the process flows.

Chapter 5 provides a number of complementary viewpoints on the architecture: the context, the information (data), development, deployment and operational concerns. These additional viewpoints aim to further clarify the key decisions and their implications.

We end the deliverable with a brief summary of the main issues discussed in the preceding chapters.
3. REQUIREMENTS AND STAKEHOLDERS

User requirements and expected functionality for end users have been collected and described in detail in the corresponding deliverable D2.1, ‘User requirement analysis’. However, to facilitate reading, in this chapter we summarize requirements as stated in the description of work, the user requirements as extracted from the use cases, and the non-functional requirements. We conclude this chapter with an overview of the stakeholders involved.

3.1. Requirements from the Description of Work

In this section, we copy the relevant part regarding the main concept, as stated in the Description of Work. This description has been the starting point for the system architecture. The original text is displayed in regular font, comments or interpretations are given in separate paragraphs in italics.

A core idea is that the process of integrating content from different media sources is carried out in an interactive manner, so that the data resulting from one media helps reinforce the aggregation of information from other media, in a cross-modal interoperable semantic representation framework. Once all the available descriptive information has been collected, an interpretation component will dynamically reason over the semantic representation in order to derive hidden, or implicit, knowledge, following an event-centered structure, so as to answer such questions as: what are the themes or topics in the data stream?, what type of situation is being represented?, what kind of participants does it involve?, when and where is it happening?, what are the attitude and sentiment being expressed?

In the system architecture, we focus on the integration of content (or rather metadata) from different media sources. In line with the work package descriptions, this will be achieved by several components (called ‘workers’ later on in the document) that together enrich the (textual) content of the resource. This enriched information will be exploited by (Web) clients that target the visualization, explanation or recommendation of answers to questions as stated in the previous paragraph.

This will be accomplished thanks to the integration in a multimodal platform of state-of-the-art information extraction and analysis techniques from the different fields involved (image, audio and text analysis) plus the addition of the information generated by exploring the social context. Actually the mechanisms of social intelligence will be a core asset for interacting with the other modalities and reinforcing the aggregation and interpretation of the data.

The multimodal interpretation platform, in an optimized process chain, will analyze a vast amount of multimedia content and enrich the content with additional metadata layers [see Figure 1]. These metadata layers provide a multi-faceted access of comprehensive information and can be used for further analyses or e.g. content-based recommendation services for end-users. Moreover the metadata layers can be used to link the relevant entities within an event. The mechanisms of social intelligence will be a core asset both at the level of cross-media labelling as well as in the multimodal interpretation component.
The suggested output of the analysis in the figure above was taken as a starting point for the selection and refinement of analysis components in the context of WPs 3 and 4. Initial outcomes and the exact type of output are discussed in D2.3 – data infrastructure and data representation definition.

3.2. Summary of user requirements

The EUMSSI framework does not (yet) have actual users. The user requirements with respect to functionality are those that we expect that the end-users will have. The end-users are represented in the two use cases in the Description of Work. Below we copy the two use cases, along with comments or interpretations in italics. More information on the targeted end users as stakeholders is given in section 2.4. A further analysis of the use cases and the envisioned functionality of the end-user applications can be found in deliverable D2.1.

3.2.1. Use case 1: Contextualizing Tool

A journalist starts editing a news article about the unrest in Egypt. The multimodal interpretation platform identifies during the editing process which multimedia items might be interesting for the journalist and offers a list of items [see Figure 2] categorized by content type (e.g. video, audio, text), relevance, up-to-dateness etc. At the same time the platform considers the media and discussion agenda on the Social Web (e.g. Twitter trending topic). Through this additional analysis the journalist can consider more than one perspective in his article and contextualise the news article with the relevant multimodal information.
The EUMSSI platform, thanks to its multimodal analysis and interpretation capabilities, facilitates real-time curation of the digital assets through automatic semantic annotation, augmenting, at the same time, the knowledge database with social media documents, conveniently interrelated and annotated.

In addition, a multimodal content-based recommendation system built on top of this platform empowers the journalist to monitor and gather up-to-date multimedia and social media documents related with his investigation without the need of reviewing an enormous amount of insufficiently annotated records. With the help of the platform, he can perform his job in an optimized manner, thus increasing productivity and the quality of his work.

In the architecture design, we concentrate on the processes needed to perform multimodal analysis and interpretation, which will result in a rich (textual) representation of media content (metadata) and associated entities. The actual design of the content-based recommendation system will start in the context of WP6 in month 12. First discussions already emphasized the importance of effective information visualization mechanisms. However, the target of creating a multimodal content-based recommendation system clearly drives the choices regarding the types of analysis that will be performed and how the results will be represented and stored.

3.2.2. Use case 2: Second Screen

There is an increasing popularity of video on-demand. On the one hand, users want to choose the programs they like. On the other hand, they also like some guidance to find out ‘what's on, what's new, what's cool, what's for me’. Today's TV guide and program
should be personalized, based on user’s preferences, user’s interests, daily routines
(morning programs, evening programs) as well as importance or hype (such as
breaking news or national events). Moreover, in order for the old TV medium to remain
relevant, it should become more interactive. Many people already have an iPad or
tablet at hand when watching television programs. This device can effectively function
as a second screen, on which to exchange comments via Facebook or chat with a
friend about the program, but also the two screens could be connected in such a way
that the tablet provides background information on a given program, personalized
news tickers, suggest what to watch next, and so on.

The EUMSSI platform makes it possible the implementation of a personalised
multimodal content-based recommendation system able to make relevant suggestions
of multimedia content to the end-user based on what she has watched, what other
people have watched, or are watching, combined with multimedia contents related in
some ways to what she is actually watching and what are the people saying about
these contents in the social networks.

In line with the previous use case: the design of second-screen applications will start in
month 12, in the context of WP6.

Figure 3 - Multimodal platform catering both for the journalist and the end-user’s use-cases

Both user-cases have a common technical base in that they exploit (multimodal)
content-based recommendation algorithms but they differ in input/output aspects, in
addition to the type of user, as mentioned above. On the one hand, in the case of the
Contextualising tool, the input to the recommendation system is less “controlled”, as it
consists in incomplete textual input that is being created on the spot, but the output
result needs not be as precise, since it is “raw material” for the journalist to process.
On the other hand, in the case of the Second screen, the input is much more
controlled since it is an already semantically annotated document from the internal
database (DWs), but the output results, necessarily need to be more accurate for the end-user.

The above paragraph is a bit vague. What is meant is that both journalists and second-screen users will be provided with additional content and recommendations that match their current interests. For journalists, the current interest will be derived from the content of (the) article(s) that they are currently working on – text analysis and entity extraction will be performed on the fly to create starting points for matching with other resources and entities in the EUMSSI repository. For second-screen users, a finished resource (say, a video) is taken as a starting point. This video has already been analyzed – apart from text analysis, also video and audio analysis will have been performed.

Both the “Contextualizing tool” and the “Second screen” use case communicate some clear expectations regarding the types of analysis that will be provided by EUMSSI and the functionality of the end-user applications, which will be designed and developed starting month 12. The underlying architecture regarding the content flow and media analysis is largely independent from the final use case.

3.3. Non-functional requirements

The functional and non-functional requirements are discussed in D2.1. For the sake of completeness and to avoid unnecessary switching between deliverables, the common non-functional requirements are summarized below.

Note that D2.1 also contains non-functional requirements with respect to the text, audio and video processing tools. Adhering to these requirements is the concern of the developers of these tools, and will not be further discussed in this document.

3.3.1. Maintainability

In general, the system should be developed to be easy to maintain. A user guide should be prepared and it should include some specific chapters to find information to solve common problems. The software elements must produce log files to monitor possible problems. The logs must be easy to understand and they must point directly to any problem that may occur in order to find the root of the problem and the way to solve it as soon as possible.

Every element must be scalable after installation. There must be a procedure to do so and this procedure should be easy to follow. For example, there must be instructions to add a node to the data base cluster (if there is one). The data integrity should not be affected by these procedures and, as a desirable goal; these processes should be seamless for the user (meaning that the service availability is not affected during the process).

Automatic backups must be done of all the relevant information of the system periodically. This includes databases, Solr cores, logs, etc. A complete restore of all this data must be possible to do in case of general failure. The procedure to restore such data must be well documented and easy to follow.
3.3.2. Robustness
Optional redundancy must be possible for every element of the system. The adopted technologies must fulfil this requirement implicitly. The chosen database software and the server that will receive all the queries and commands must support redundancy. Budget of real use case adopters will decide whether to use redundancy or not, but the system must be designed to have the chance to do it.

The system functionality must not be affected by corrupted data input (invalid video or audio files, for example). This kind of input must be rejected if there isn’t any profitable data in them but the system availability cannot be compromised.

3.3.3. Security
All the internal elements must be hidden from external users. The only way to get data must be the public API. The servers must be correctly protected from any kind of attack. As a system that will receive queries and commands, there must be a special plan to avoid code injection on them.

3.3.4. Easy to use
The system API must be available for users (desirable: in different languages e.g. English, German, Spanish, French) and it must be clear enough to allow an easy integration and use.

The API should be supported by a well-established technology that the industry commonly uses (REST like) in order to make the integration process easier and faster.

3.3.5. Trustworthiness
This is, probably, the most difficult requirement to define. The event understanding must be reliable, and precise. For example: It is obvious that there should be a minimum set of famous people that the system should recognize (from all kind of sources). It must be able to recognize current and former presidents or prime ministers of every country but, where is the limit? How can be determined the set of famous people that should be automatically recognized?

It is clear that a system unable to recognize the presence of the British prime minister in a video is not meeting the requirements. The same reasoning applies to places, remarkable dates, monuments, etc.

In the online processing scenario, the suggested content must be related with the input of the journalist. If the system returns content that has nothing to do with the journalist text, the system will be useless and the users will lose confidence and be reluctant to use it.

3.3.6. Portability
As a general requirement, the client side must be able to run in any platform. The client side will perform queries and will issue processing requests. These two types of interactions can be initiated from any platform. This should not be a problem, as both interactions are planned to function over http (REST, SOAP, XML-RPC, etc.)
3.3.7. Development costs
The development costs must be adjusted to the current approved budget for the project. All the involved partners must respect the document of work approved by the consortium.

3.3.8. Operational costs
The system should be designed to be compatible with the most common hosting solutions (Azure, Amazon, etc.) so, in the case of a cloud installation, the hosting costs can be minimized. There must a special focus on minimize the bandwidth use in all the transactions with the clients because it has a direct impact on the overall hosting cost.

3.3.9. Functional scalability
The software must be designed from the beginning to allow further improvements in the future. For example, in the text analysis area, the supported languages will be English, German, French and Spanish. The software must be designed in a way that it should be easy to add support for more languages. The same principle applies to the input sources (it must be opened to new ones such as new social networks).

3.4. Stakeholders
Two main groups of stakeholders for the EUMSSI system can be identified.

On the one hand, there are the consortium partners, who entered the project with their individual goals and expectations that need to be fulfilled in the course of the project. For the research partners, this means that the development of the EUMSSI framework should provide opportunities for research and analysis in their field of expertise, as well as collaboration and knowledge exchange. The business partners, VSN and DW, are involved as they expect additional value of (parts of) the EUMSSI framework for their core activities.

During the lifetime of the project, the EUMSSI framework will function as a fully functional prototype that will be evaluated in two pilot experiments run in real-world environments. These pilot experiments will focus on the two use cases, as described earlier in this chapter, and address the stakeholders mentioned in these use cases. Even though these stakeholders are currently 'fictional', in the final year of the project they will 'materialize' as news editors (focus groups of DW journalists and VSN employees) and end-users, consisting of selected DW audience of different linguistic communities in Europe.

In this section, we describe these groups of stakeholders, and their expectations, in more detail.

3.4.1. Roles of consortium partners
The EUMSSI consortium consists of several partners who each have their specific goals and interests for participating in the project.

Most partners are research institutes, with a specific expertise in the analysis and enrichment of audio, video or text data - and are interested in extending their
knowledge and expertise by collaborating with partners with complementary fields of expertise. The EUMSSI framework provides many points of interaction and a common goal to make this possible.

The integration and application of the various technologies to be developed in EUMSSI is an important goal for all partners. For our research partners, the integrated system functions as a proof that their technologies can be deployed in real-world contexts - which is an increasingly important evaluation criterion for scientific publication.

In addition, our exploitation partners, expect EUMSSI to provide them with solutions that allow for enhanced (metadata) descriptions of audiovisual material, and specific search, analysis and recommendation functionality enabled by these enhanced descriptions.

- VSN, as a provider of multimedia content production and management solutions, is interested in EUMSSI's potential market and business possibilities, and will integrate EUMSSI's results into their own workflow tool. As a result, a real integration and strong validation of the project outcome is guaranteed.
- DW, as a public service broadcaster, provides the ideal setting for reaching to users, either professional users using the integrated tool, or to a large mass of TV end-users using the second-screen.

3.4.2. Use-case stakeholders

3.4.2.1. Contextualizing tool for journalists

During the EUMSSI kick-off meeting, Deutsche Welle presented their vision on how EUMSSI could improve the work of their journalists.

DW envisioned easy-to-use tools that provides new ways to harness data to change the way news stories are discovered and told. The EUMSSI framework should allow journalists to surface stories to investigate. Particularly, it should uncover hard-to-get information and relations.

DW expects that EUMSSI will develop data analysis and computer-assisted storytelling tools - all with the goal of supporting journalists and content producers in our data-driven world. For data analysis, they envisioned the combination of data mining and pattern recognition software that can generate story leads.

VSN provides a publication platform for (multimedia) content providers, which includes journalists. Similar to DW, they expect EUMSSI to automate the monitoring, gathering and filtering of relevant content from the Web - including social Web sources. They expect the system to function as a 'personal assistant' of the journalist that provides notifications of hot topics, related content and other suggestions.

From the above statements it can be observed that journalists and content providers expect EUMSSI to provide meaningful summaries, interpretations, visualizations and recommendations while working on a particular topic, without interrupting their workflow. This means that the contextualizing tool(s) should be (loosely) integrated in the platforms that they use.

3.4.2.2. Second-screen users
In the description of work, it is stated that the EUMSSI platform will provide ‘personalized multimodal content-based recommendations of multimedia content to the end-user based on what she has watched’ in second-screen situations. However, it has not been clearly defined what exactly second-screen situations are and what kind of recommendations are expected. During the kick-off meeting, DW mainly suggested to provide users with relevant social media streams, including tweets involving Deutsche Welle.

According to Wikipedia², “The second screen refers to the use of a computing device (commonly a mobile device, such as a tablet or smartphone) to provide an enhanced viewing experience for content on another device, such as a television. In particular, the term commonly refers to the use of such devices to provide interactive features during "linear" content, such as a television program, served within a special app.”

The second screen is still a largely unexploited area, but several companies started experimenting with apps dedicated to particular programs - mainly in the field of entertainment. Functionality of these apps includes³:

- share the experience of live episodes together
- keep viewers engaged with polls, trivia, still photos, quotes and flashbacks
- follow the social media buzz
- capture screenshots, write own captions and share it with friends
- search for shows and watch full episodes on your cell phone
- access extra content
- tag television programs
- share opinions and thoughts

Disney recently launched its second screen⁴ to accompany their films and other broadcasts. Synchronized with the film, users can "explore interactive galleries, check out activities, learn interesting facts about the scenes you’re watching, connect to social media, and more."

These second-screen functionalities all provide additional information with (at least partially) the aim to entertain the user. For this type of interaction, the term infotainment has been coined⁵:

*Infotainment is "information-based media content or programming that also includes entertainment content in an effort to enhance popularity with audiences and consumers." The term can also refer to the hardware/software products and systems which are built into, or can be added to vehicles in order to enhance driver and/or passenger experience. It is a neologistic portmanteau of information and entertainment, referring to a type of media which provides a combination of*

---

³ [http://mashable.com/2014/02/03/second-screen-tv-apps/](http://mashable.com/2014/02/03/second-screen-tv-apps/)
⁴ [http://disneysecondscreenscreen.go.com](http://disneysecondscreenscreen.go.com)
information and entertainment. According to many dictionaries, infotainment is always television, and the term is "mainly disapproving." However, many self-described infotainment websites exist, which provide a variety of functions and services, many of which include the several increasingly popular social media websites and applications being used daily by billions of users world-wide.

3.5. Concluding remarks

In this chapter, we summarized the functional requirements, as derived from the Description of Work. Many of the functional requirements involve applications and tools that the targeted end users – as described in the use cases – will most likely expect. Even though the design and development of these applications and tools will start in M12, these requirements should be taken into account during the design of the EUMSSI framework and the data representation format.

As EUMSSI is a research project, the exact nature of the end-user applications has not been described in detail. Part of the (novel) functionality will also be driven by the (expectedly innovative) results of the audio, video and text analysis components. Therefore, apart from the end result, work on the EUMSSI framework should also be evaluated on the number of publications and other scientific output produced during the project.
4. ARCHITECTURE DESCRIPTION

From the requirement analysis in the previous chapter it becomes clear that the EUMSSI system should fulfil three main functions:

- Collect and store (meta-)data of multimedia resources from several sources in a unified format.
- Enrich the existing data with the (combined) outcomes of audio, video and text analysis.
- Provide end-users (journalist and second-screen users) with useful and/or entertaining suggestions, background information and recommendations.

Along with discussions on the system’s functionality, several iterations of architecture sketches have been created with the intention to communicate the envisaged functionality to the consortium partner, and to facilitate the selection of components. In the next section, we summarize the different iterations, to provide a historical background on the design decisions, as reflected in the current architecture.

The current software architecture design is discussed in Section 4.2. We will first provide an overview of the four main components, followed by a more detailed description of the main functionality and – where applicable – the framework or tools that will be used for implementing the component. The second part of the architecture design involves the process flows, which are illustrated in section 4.2.2.

4.1. Initial sketches and selection of components

The main partners involved in the architecture discussion are L3S, VSN and UPF. All three partners provided sketches and commented upon sketches provided by the others, during the monthly videoconferences, and via the EUMSSI forum and Google Doc commenting functionality.

4.1.1. First iteration (proposed by L3S)

The goal of the first architecture diagram, as displayed in Figure 4, was to emphasize that the data collection and data processing activities are completely independent from the applications that the end-users (journalists and second-screen users) will work with. The connection between the two parts of the system is the data store.

1. in the data collection, several sources will be considered. DW is the main source of videos and textual articles, but other sources such as Social Media sites (twitter, YouTube) will be included as promised in the DoW, the Guardian is a candidate but no decision is still made. The metadata for the different objects will be collected in a raw format.
2. Every object (its collected metadata) will be annotated with UIMA to further enrich it and provide better quality objects as input to the different analyses.
3. These enriched objects will be stored in a database (MongoDB, MySQL or Cassandra) from where the different analyses will take their input to further
4. All the metadata stored in the database will be, besides, indexed in Solr to provide free text search to both, EUMSSI partners and end-users (through a tomcat server).

4.1.2. Second iteration: integration of EUMSSI with VSN

As stated in the Description of Work, consortium partner VSN will be responsible for the election, installation and maintenance of a central platform for data storage (task 5.2). A particular concern of VSN is the availability of the contextualizing tools (audio, text and video analysis) to enrich their own repository. To quote Section 3.2.2 from the DoW:

VSN develops a powerful, entirely integrated solution for digital newscasts production, called VSNNEWS. It covers all issues in a uniform environment, including rundown planning and creation, resource management, word processing, video feed recordings, archive storage, organization and cataloguing. [...] VSN’s business model of the EUMSSI solution for journalists would be a pay-per-use service. In order to contribute to the results uptake, EUMSSI project will reach out to VSN international clients for creating a user group that could contribute to testing and evaluating of EUMSSI’s results, in particular, the contextualizing tools.

---

Note that at that point, following the DoW, we still considered the use of a SQL or NoSQL database. As will be explained later, the NoSQL functionality in the latest version of Solr made us decide to exclusively make use of Solr – which reduces any possible issues regarding synchronization.
As a consequence, VSN indicated that it would be important for the architecture to be agnostic with respect to the data storage used in the analysis process. More concretely: VSN proposed to split the content/data management tools from the unstructured media analysis tools, as illustrated in Figure 5.

![Figure 5: Split between unstructured media analyzer and MAM](image)

The VSN MAM is not open-source and therefore cannot be considered as a central repository for EUMSSI (as mentioned earlier, we chose to use Solr for this purpose). However, particularly for use case 1 (the contextualising tool, led by VSN) connectivity to the VSN MAM would be a great plus, as it provides functionality for auto-cataloguing, advanced search, recommendation for related assets and computer-aided story writing.

Apart from using the VSN MAM for the contextualising use case in the context of WP6, VSN expressed their expectation to adopt EUMSSI components for enriching the multimedia data for their own clients. They pointed out that they expect some of the analysis components (particularly those who provide text analysis and named entity recognition) to function real-time (on-demand). The video and audio analysis partners IDIAP and LIUM indicated that they can only provide their components as offline (batch) processors. On-demand text processing is considered particularly important for quick analysis of working documents (such as articles that journalists are currently writing on), in order to provide relevant recommendations. This split is illustrated in Figure 6.

![Figure 6: Split between on-demand workers and batch workers](image)
4.1.3. Third iteration: Processing architecture (proposed by UPF)

The third iteration of the architecture design, as illustrated in Figure 7, integrates the initial architecture design with the observations from the second design.

An important difference is the division between the preprocessing stage and the processing stage.

- Preprocessing is now understood as the collection of multimedia and the alignment of metadata into a common scheme. The raw, unaligned material will be stored in the MAM (which we will call ‘Unstructured Data Store’ later). Aligned material will be stored in the EUMSSI CAS.
- Text, video and audio analysis will make use of the aligned EUMSSI CAS. The analysis results will be fed back into the CAS as either updated (annotated) fields or as extra fields (such as identified persons in video fragments).

Figure 7: Architecture design, third iteration

The envisaged process flow can be summarized as follows:

1. new data arrives (or gets imported)
2. preprocessing stage
   a. make content available through unique URI (from central MAM)
   b. create initial CAS with aligned metadata / text content and content URI
   c. add content to processing queues
3. processing / content analysis
   a. distributed analysis systems query queue when they have processing capacity
   b. retrieve CAS with existing data (or get relevant metadata from wrapper API)
   c. retrieve raw content based on content URI
   d. process
e. update CAS (possibly through wrapper API)

f. update queues
   i. mark as processed
   ii. add to queues for other processes that depend on previous analysis results

4. indexing when processing is complete for a content item (e.g. with Solr)

Note that this architecture design mainly depicts the data analysis part of the EUMSSI system – the deployment by Web applications is not visible in the figure. Still, it is clear that the functionality of the analysis part has direct consequences for the deployment (even though the components will be strictly separated):

- Reasoning and cross-modal recommendations and visualizations for journalists and second-screen users will depend on the outcomes of the analysis workflow (i.e. the richness of the data and the metadata).
- On-the-fly text analysis will be needed to extract key concepts from working documents, which function as an implicit query for recommendation.

The CAS will only store the objects’ metadata with pointers to the corresponding sources (i.e. video or audio material). For the video material of Deutsche Welle, we will have guaranteed access to their media center. For video material from other sources it might be desirable to store a local copy (as far as allowed by copyright regulations).

4.2. Overview of the current architecture

From the user requirements as well as the functional and non-functional requirements, it has become clear that the EUMSSI system needs to be highly modular and flexible, in order to cope with different types of data sources, multiple representation formats, an evolving set of analysis methods and various types of deployment for demonstrating or testing purposes.

For this reason, the system has been divided into four components, which each provide a clearly defined function.

Essentially, the EUMSSI system is a workflow for obtaining, preprocessing, analysing and processing (meta-)data of multimedia resources. As explained in more detail in D2.3, we have chosen the open source enterprise search platform Solr7 as our data storage platform. Solr provides advanced search functionality as well as NoSQL features. Further, we make use of the UIMA framework8, which provides a solid basis for the development and deployment of annotators and other tools. For this reason, the EUMSSI framework will be implemented in Java.

4.2.1. Diagram of components

---

7 http://lucene.apache.org/solr/
8 http://uima.apache.org/index.html
Figure 8: Component overview

Figure 8 shows the four main components and the functionality that they provide. All components are independent from one another (the only dependence is that the data that the components work with is available and that the relevant data stores are defined).

The components will be implemented as Java classes, which will reside in the package corresponding to the main component, i.e.: eu.eumssi.datacollection.*, eu.euimssi.datapreprocessing.*, eu.eumssi.dataanalysis.*, eu.eumssi.deployment.*.

The components and their functionality are described in more detail in the following subsections.

4.2.1.1. Collection of data from data sources

Data is either provided as data dumps or automatically collected via crawlers. The task of the data collection component is to consume this data and to prepare it for storage in the unstructured data store.

Apart from differences in syntax, the unstructured data store contains the original, unmodified data, including provenance metadata (metadata that indicates from which source the data comes, and the time it was crawled).

The data collection components will each follow their own procedure, dependent on the type of data and the way it is collected (which varies from manual upload of data dumps via daily dedicated crawls to continuous crawling). All components will store data in the Unstructured Data Store, either by using the Solr UpdateRequestHandler, which directly stores data into the Solr core as specified by a URL, or by making use of a library - for Java the best candidate is SolrJ.

4.2.1.2. Data preprocessing and alignment
The main goal of data preprocessing is to ensure that the metadata of the different sources is aligned. For this purpose, in D2.3 – Data Infrastructure and Representation – a metadata scheme is proposed that is a subset of the widely adopted standard schema.org is proposed, along with a number of fields that are important within EUMSSI but that do not have a suitable field in schema.org.

The conceptual mapping from the data sources (in D2.3 the mapping of the datasets that we currently consider is presented) needs to be implemented by a Data Preprocessor. The Data Preprocessor can be considered as a simple pipeline that consumes the newly arrived data in the Unstructured Data Store, maps the metadata names to the agreed format, and stores this in the EUMSSI CAS Store.

In principle, the Data Preprocessor collects the relevant input from the Unstructured Data Store, which is a Solr instance, process the data, and then stores it into the EUMSSI CAS Store, which is another Solr instance (or a different core in the same Solr instance – this will be decided during the implementation phase, but this is just a minor detail from an architecture perspective). An interface implementation for the Data Preprocessor will be created early in the development process – most probably the Preprocessor will implement the UIMA Annotator Interface (similar to the Workers for data analysis, as explained immediately after this section).

4.2.1.3. Data processing and analysis

Data processing and analysis is performed by the audio-visual analysis and text analysis components, which will be developed in WP3 and WP4. How these components work internally, is beyond the scope of this document. However, we assume that these components consume and modify data from the EUMSSI CAS Store9. Further, we assume that there will be two different kinds of analysis components, which we will call workers from now on: batch workers, which analyse data offline, and on-demand workers, which return the results in an agreed-upon maximum time span.

The order in which the workers will perform their jobs (sequentially and/or in parallel) will be defined in the Queue Manager, a Java class that sends and receives messages to the worker in a (configurable) defined order.

Depending on the defined dependencies, after having sent a job to a worker, the Queue Manager will wait for an answer from the worker (in the case of on-demand workers) or continue with its activities (in the case of batch workers) - once the batch worker indicates that it has completed its job, follow-up jobs can be scheduled. The exact process flow is explained in the next section.

The Queue Manager and the Workers will communicate through a messaging broker. A state-of-the-art broker that fits our requirements is the open source server Apache

---

9 Note that our partner VSN is interested in using the EUMSSI annotation workflow for annotating the multimedia material in their CAS. This is foreseen in the architecture design and therefore pictured in grey.
ActiveMQ\textsuperscript{10}, which supports various programming platforms and various communication protocols, including OpenWire, Stomp and REST. The Queue Manager and the workers can take the role of message producers (sending messages) and message consumers (receiving messages)\textsuperscript{11}. These messages can be any kind of serializable object. In the data analysis context of EUMSSI, the following two messages will play the most important role:

- \textit{Jobs issued by the Queue Manager}: an array, consisting of a unique Job Identifier, followed by the identifiers of the documents to be processed
- \textit{Status messages from the workers}: a tuple, consisting of the Job Identifier, followed by a status code (at this point of time, we consider the following three as the most relevant ones: completed, pending, cancelled).

Both on-demand text processing and batch audio and video processing components will be implemented as (or \textit{wrapped} as) UIMA Annotators. The Apache UIMA project\textsuperscript{12} is a framework for the analysis of unstructured content such as text, audio and video.

UIMA annotators all implement a standard interface\textsuperscript{13}, which is defined in the JCasAnnotator_ImplBase class. The main method to be implemented is \textit{process}, which takes a JCAS object (the java implementation of the Common Analysis System API) as input. Analysis results will be stored in the same object.

After analysis, the CAS object needs to be stored in the EUMSSI CAS Store. For this purpose, the UIMA add-on Solr CAS Consumer\textsuperscript{14} can be used, which needs to be configured with a mapping file for the relevant Solr core and the address of the Solr instance.

We probably might want to use two different CAS Stores: one CAS Store that stores the complete annotations in XML or binary format, and one that contains the data in a way that can be easily queried for the end applications (following the metadata scheme defined in D2.3).

\textbf{4.2.1.4. \textit{Deployment and demonstration}}

Deployment and demonstration will take place via (Web) clients that query the data available in the EUMSSI CAS Store. As explained in more detail in D2.3, Solr provides a Web API for querying the data, which the clients will use.

\begin{itemize}
\item \textsuperscript{10} http://activemq.apache.org/
\item \textsuperscript{11} See http://www.javablogging.com/simple-guide-to-java-message-service-jms-using-activemq/ for a basic example
\item \textsuperscript{12} http://uima.apache.org/index.html
\item \textsuperscript{13} Documentation can be found here: http://uima.apache.org/downloads/releaseDocs/2.2.2-incubating/docs/html/tutorials_and_users_guides/tutorials_and_users_guides.html#ugr.tug.ae
\item \textsuperscript{14} https://uima.apache.org/d/uima-addons-current/Solrcas/SolrcasUserGuide.html
\end{itemize}
These clients will be designed and developed in the context of WP62 (multimodal content-based recommendation), in which a contextualising tool for news editors and a second-screen demonstrator will be built, following the use-cases. The internal working of these clients is out-of-scope for this document and will be described in the upcoming deliverables D6.2 and D6.3. Based on first discussions, it is likely that the contextualising tool will be a combination of a regular Web application and an add-on/plugin for either a Web browser or a CMS.

4.2.2. Process flows

As explained in the previous section, the EUMSSI system supports four main activities by components and associated processes that work more or less independently. In this section, we explain the process flows in more detail.

4.2.2.1. Data collection

The data that will be collected by the EUMSSI system involves articles, (links to) video and audio, and the associated metadata. The goal of the data collection process is to ensure that the data:

- will be converted to a format suitable for the subsequent processing step
- will be stored in its original format, without alignment or changes to the metadata.

Data will be either offered as regular *dumps* (as is the case with data from Deutsche Welle) or collected using dedicated *crawlers*, which provide new data on a regular basis.

The process of collecting data dumps is rather straightforward, as illustrated in Figure 9.
Once the data arrives, the process is started. The input file is read and the metadata will be stored in unaltered form in the unstructured data store. *Unaltered* means that field names and data values will not be aligned in this phase – the actual syntax may be changed, though, to facilitate the data preprocessing. Once the collection process ended, a notification is sent to the Data Alignment.

The collection of data retrieved via crawlers follows the same procedure. The only difference is that the process is not started manually (as with the data dumps) but is initiated once the crawlers provide new data – see Figure 10.
The data is processed as described earlier. Once the data alignment process is notified, the component returns to its waiting state until new data arrives.

4.2.2.2. Data preprocessing and alignment

Data preprocessing involves the alignment of the metadata to a unified metadata scheme, as chosen and agreed upon by the consortium. As explained in D2.3, the current unified scheme is a subset of schema.org, with a number of additional fields.

As the preprocessing steps are the same for each data source, the preprocessing will be implemented using a standard interface. However, as the schemes of the data sources differ between data sources – and in order to keep the system open for additional data sources that are currently not foreseen – the interface will be implemented by dedicated components for each individual data source.

The alignment components wait until new data arrives in the unstructured data store. Based on the detected scheme, a specific pre-processor/alignment component is chosen. This component maps the data and metadata to the common scheme and creates a CAS structure for each document, which is saved in the EUMSSI CAS store. This process is depicted in Figure 11.

Figure 10: Collection of crawler data
4.2.2.3. Data processing: audio, video and text analysis

The data in the EUMSSI CAS initially contains just the metadata as available in the original sources. The goal of the data processing step is to add additional metadata, which are the outcomes of the audio, video and text analysis.

The preprocessing is controlled by a queue manager, which activates individual workers once data arrives. There are two types of workers:

- **Batch workers**: these workers analyse data offline and do not provide any guarantees when the analysis will be complete.
- **On-demand workers**: these workers analyse (textual) data and return the results in an agreed-upon maximum time span.

Batch workers perform time-intensive or experimental processing – within EUMSSI, this involves the audio and video analysis. On-demand workers perform text analysis tasks that can be achieved in a reasonable amount of time.

The queue manager has a configuration of workers to activate when new data arrives. A typical configuration would be: first it activates the on-demand text analysis for extracting persons, places and other entities and waits for the results. Then it activates one or more batch workers (batch workers might work in parallel if there are no dependencies). As a final step, another round of text processing is applied to the extracted metadata.

**Figure 11: Data preprocessing**
A typical process for on-demand processing would be (see Figure 12): once the data arrives, the relevant workers are selected and added to the processing queue. From the processing queue, the joblists are sent to the workers and the queue manager waits until the processing is complete. The EUMSSI CAS is updated and the queue manager returns to its waiting state.

Batch processing follows the same procedure, with a few small adaptations.
The queue manager is activated when the data arrives, selects the workers and sends the job lists to the individual workers (making use of the ActiveMQ message broker, as explained in the previous section). However, it will not specifically wait for the results to be ready, but wait for new data in the meantime – or a notification that a worker process has been completed, upon which the processing queue will be updated.

The current status of the documents with respect to the analysis (i.e. done, in progress, pending, cancelled, failed) will be stored either together with the CAS or in a separate store. This will allow the rebuilding of queues in case of system failure.

The individual workers’ only responsibility is to carry out a specific type of analysis, when requested via the processing queue. The workers are not aware of the larger picture and could serve as stand-alone components. However, they need to adhere to a specific communication protocol (as explained in the previous section, we will use the UIMA framework for this purpose).

In response to stakeholder concerns from the research partners, some components may communicate using a simplified API that does not expose the underlying CAS structure of the document. This concerns in particular components such as ASR which need little information about the document beyond the audio stream that needs to be analysed and for which the CAS would present unnecessary overhead. In those cases the dispatching server will take care of merging the analysis results with the existing information in the CAS.

Figure 13: Queue managing of batch processes
As illustrated in Figure 14, batch workers check the processing queue on a semi-regular (or even irregular) basis. When the processing queue is not empty, the batch worker consumes the requests, process these requests (in other words: analyses and enriches the data) and stores the additional data in the EUMSSI CAS. Finally it sends a notification to the queue manager that the processing has been done.

On-demand workers work in a similar manner, with the difference that upon completion of a job, the process does not end – instead, it returns to a waiting state until new data arrives in the processing queue.
4.2.2.4. Deployment and demonstration

In this section, we provided a detailed overview of the current thinking on the architecture for the EUMSSI platform. As a result of several iterations of sketches and discussions, the functionality is divided into four separate components:

- Data collection
- Data preprocessing
- Data processing/analysis
- Deployment/demonstration

For each main component, we discussed the main (expected) functionality, the format of the input and output, the communication protocols to be used, and the frameworks that will be used for implementing the components. In addition, the process flows for each component have been illustrated by UML diagrams.

Central to the EUMSSI framework will be the data repositories that will be used for the interim storage of unstructured data, the aligned metadata together with the analysis results, and the repository that will be used by the end-user applications (contextualizing tools and second-screen applications). At the time of writing, the data repositories are available as separate cores in a single Solr installation.

The architecture design, as discussed in this chapter, will follow the basis for the implementation activities in the context of task 5.2 (data infrastrucutre) and task 5.3 (bundling of processing tools and APIs). We expect that during the implementation phase several details will be refined or slightly changed, due to progressing insight.
5. ARCHITECTURE VIEWS – SYSTEM FUNCTIONALITY

It is close to impossible to describe a software architecture using a single model. Moreover, architecture designs that have been described in a single model are typically heavily overloaded and therefore not suitable for communicating the key ideas.

For this reason, the ISO 42010 standard recommends to break the architecture down in several interrelated views, where a view is defined as “a representation of one or more structural aspects of an architecture that illustrates how the architecture addresses one or more concerns held by one or more of its stakeholders”. A view is described as a viewpoint, a “collection of patterns, templates and conventions for constructing one type of view.”

Commonly used views are Logical, Process, Physical and Development. (Rozanski & Woods, 2012) translated these views into seven viewpoints.

5.1. Selection of viewpoints

From the seven viewpoints, we selected the six that are most relevant to the EUMSSI system, leaving out the concurrency aspect. These viewpoints concentrate on the context, the information (data), development, deployment and operational concerns. We omitted the functional viewpoint from this chapter, as functionality has been covered extensively in the previous chapter.

5.2. The context viewpoint

The context viewpoint describes the relationships, dependencies and interactions between the system and its environment. The context view plays an important role in helping stakeholders to understand the system’s responsibilities and how it relates to their organization.

Figure 16: Stakeholders of the EUMSSI Framework
Building upon the observations in Section 2.3, Figure 16 summaries the main stakeholders that are directly related to the EUMSSI framework.

The main stakeholders during the project are the consortium partners, which can be divided into research partners and exploitation partners.

The research partners IDIAP, LIUM, LUH, GFAI and UPF will use the framework as a basis for their research and development activities. It is their responsibility to ensure that the EUMSSI analysis components will provide significant enhancements compared to the existing metadata of multimedia material. In return, the partners expect that the activities form a basis for scientific publications.

The exploitation partners DW and VSN, in cooperation with UPF, are responsible for the demonstrators, which are summarized as ‘Web client’ in the picture. It is their responsibility to showcase the results of the metadata enrichment by the research partners in terms of recommendation, analysis and visualization sites, add-ons or widgets to be used in the demonstration activities. In return, DW expects a number of tools/add-ons that provide additional value to journalists and that can be considered for take-up by Deutsche Welle. Deutsche Welle also plays the role of data provider. VSN’s expectation mainly concerns the analysis modules, which are expected to enrich the multimedia metadata in their existing repository.

Journalists and second-screen users are the envisaged end-users of the EUMSSI framework. The demonstrators will be developed with the requirements from these end-users in mind.

5.3. The information viewpoint

The information viewpoint describes the way that the system stores, manipulates, manages and distributes information. This viewpoint develops a complete but high-level view of static data structure and information flow.

As the EUMSSI project focuses on the collection, transformation and enrichment, and deployment of multimedia social information streams, the data infrastructure and
representation has been described in more detail in deliverable D2.3. Figure 17 provides a high-level overview.

Data is collected from a number of data sources. This data may be in different (file) formats and may make use of different metadata schemes. Preserving the original metadata scheme, the data is stored into the Unstructured Data Store, which is a Solr instance with a separate core (and associated schema) for each dataset. The data is then aligned to a common scheme, based on schema.org. This data is stored in a separate Solr instance (or core) and used for creating the UIMA representation (CAS).

The data in the EUMSSI CAS is enriched, making use of various analysis components for audio, text and video. The exact order of this enrichment is controlled by the processing queue. For the enrichment process, external knowledge bases (such as DBpedia) may be consulted.

For the deployment of the data, in the form of demonstrators and prototypes targeted at journalists and second-screen users, it may be needed or desirable to optimize the data to a deployable EUMSSI CAS, with optimized indices.

Note that when speaking about multimedia data, we mean the metadata of the multimedia, which includes a reference (URL) to the actual multimedia, which may or may not reside on the EUMSSI server (depending on availability and copyright restrictions).

5.4. The development viewpoint

Development views communicate the aspects of the architecture of interest to those stakeholders involved in building, testing, maintaining and enhancing the system.

The most important aspect of the development viewpoint is the development plan, which is given by the Description of Work. A component-oriented summary is given in Figure 18.
The first six months of the project were dedicated to the collection of user requirements, and the definition of the software architecture and the data infrastructure. In the meantime, the data from the Deutsche Welle has been imported into the Unstructured Data Repository.

The second milestone is planned in Month 12, the Preliminary Version. The core of the Preliminary Version is formed by the Data Infrastructure and the developed Tools and APIs. These tools and APIs will be used by the various analysis components, to be developed in WP3 and WP4. These components will be developed more or less independently. The preliminary version loosely connects all components and will show that they can work with the same data base.

The preliminary version as a whole, and the individual components will be tested, refined and further integrated in the period Month 12 – Month 24. In this period, the functionality will be extended with Metadata Enrichment, and the demonstrators (Contextualizing Tool and Second Screen) will be developed. This will result in the second milestone, the First Integrated Version, in Month 24.

Evaluation with the demonstrators, further testing and refinement will lead to the availability of the Final Version in M32. We planned four additional months for further validation and bug fixing.

5.5. The deployment viewpoint

The deployment viewpoint describes the dependencies that the system has on elements of it. This view captures the hardware environment that the system needs and the technical environment requirements.
The EUMSSI platform will be implemented in JAVA Version 7. Apart from Java, the system will depend on the availability of a number of packages: Solr, a Tomcat server, ActiveMQ, UIMA and probably a database.

For reasonable performance, a system with sufficient runtime memory, processing speed and hard disk speed is needed. At this point in time, it is hard to estimate the exact hardware requirements. Currently, we have an Ubuntu 13.10 server installation with 30 GB memory and ‘lots of’ disk space. As the current use of physical memory by Solr, with all data from Deutsche Welle, is 2.42 GB, we assume that this configuration is more than sufficient for our requirements. Memory, speed and disk space can be extended when needed. For additional scalability, multiple Solr instances on a cluster can be used.

To support the development process, we will use the Eclipse IDE. Code will be shared through Github. Documentation and other communication will take place via Google Docs and Google Forums.

5.6. The operational viewpoint

The operational viewpoint describes how the system will be operated, administered and supported when it is running: it describes concerns regarding installing, managing and operating the system.

The Ubuntu server resides at VSN. VSN is responsible for the physical running of the server, and will provide support in case of hardware failures or power outages.

Installing, managing and operating the system is the responsibility of the super users. Currently, there is a group of four super users, belonging to UPF, VSN and L3S.

The batch workers for audio and video analysis (audio segmentation, speech recognition, person identification and video analysis) will initially not be deployed on the EUMSSI server, but they will run on the computers of the involved partners (IDIAP, LIUM). It will be their responsibility to access the Queue Manager (via the ActiveMQ messaging system) and to update the Solr installation that serves as the EUMSSI CAS.

5.7. Concluding remarks

In this chapter, we discussed the EUMSSI framework from five views. We placed the framework in the context of the involved stakeholders (consortium partners and use case stakeholders). We described the data flow from an information viewpoint. We looked at dependencies in the development process, and we described requirements with respect to deployment and operation of the system.

Most information in this chapter could already have been derived from the system architecture design, as described in Chapter 3. However, the additional viewpoints describe the system from a slightly different perspective and may be helpful as a reference during the development and testing of the associated aspects of the system.
6. CONCLUSION: REQUIREMENTS, COMPONENTS, PLATFORMS AND NEXT STEPS

In this chapter, we provide a brief summary of the most important aspects regarding the user requirements, the individual components and the platform used. We end the chapter with an overview of the next steps that will be taken.

6.1. Requirements

The functional requirements of the EUMSSI framework stem from the stated goals in the Description of Work.

The main technological goal is that the framework will integrate content from different media sources, so that data stemming from one modality (text, audio, video) will help in extracting data from other modalities. The integrated analysis, which will be implemented as a Processing Queue, will lead to significantly enhanced (metadata) descriptions of audiovisual content, which allows for exploitation by (visualization and recommendation) tools that support two use cases: a contextualizing tool that empowers journalists to monitor and gather up-to-date multimedia and social media documents related to his current work, and a second screen that provides end-users with relevant (entertaining) content related to the programs they are watching.

Journalists and second-screen users are the target end users of the system. However, at the moment, they mainly exist on paper. They will be represented by participants in the demonstration and validation activities in the second part of the project. In addition to these ‘fictional’ stakeholders, the consortium partners are also stakeholders in the system: as data providers, as research partners, and as exploitation partners.

6.2. Individual components

The EUMSSI framework will have to cope with different types of data sources, multiple representation formats, an evolving set of analysis methods and various types of deployment for demonstrating or testing purposes. Therefore, the system has been divided into four independent components, which each provide a clearly defined function. The components are connected by means of the data (sources) that they consume or produce.

- **Collection of data from data sources**: data stemming from dumps or crawls will be stored in its original form in an Unstructured Data Store
- **Data preprocessing and alignment**: the alignment of unstructured data from the previous step into a common metadata format, which will be stored in the EUMSSI CAS.
- **Data processing and analysis**: a configuration of on-demand text processing and batch audio and video processing, which leads to enhanced and additional metadata of the resources stored in the EUMSSI CAS.
- **Deployment and demonstration**: The development of multimodal content-based recommendation mechanisms that exploit the enriched content in the EUMSSI CAS, and the design of tools that address the two use cases: contextualizing tool and second-screen.
The exact functionality of these main components, their relations, and the process flows have been described in detail in Chapter 3. Chapter 4 provided additional viewpoints with respect to the context, the information (data), development, deployment and operational concerns.

6.3. Platforms used and hardware requirements

As stated in section 4, the EUMSSI platform will be implemented in JAVA Version 7. Apart from Java, the system will depend on the availability of a number of packages: Solr, a TomCat server, ActiveMQ and UIMA.

For reasonable performance, a system with sufficient runtime memory, processing speed and hard disk speed is needed. At this point in time, it is hard to estimate the exact hardware requirements. Currently, we have an Ubuntu server installation with 30 GB memory and ‘lots of’ disk space.

6.4. Next steps

At the time of writing, the data available from Deutsche Welle is available from the Unstructured Data Repository, a Solr installation that runs at the EUMSSI server hosted by VSN. The mapping of the metadata of the Deutsche Welle resources – as well as selected other resources – is documented in deliverable D2.3. Next steps to be taken during the next few months include:

- Implementation of the defined mappings and the creation of the EUMSSI CAS with the aligned data.
- Implementation of crawlers to obtain additional data from the sources specified in D2.3.

The partners involved in WP3 and WP4 already specified the expected output, and its format, of the audio, video and text analysis. This has been taken into account by the design of the uniform representation format in the EUMSSI CAS. Simultaneously, the partners already started implementing and testing the tools. In order to integrate the components into the EUMSSI framework, the next steps to be taken are:

- Installing the ActiveMQ message broker service and implementation of the messages regarding job scheduling.
- Providing a reference implementation of an annotation component, to serve as a basis for the (wrappers of the) analysis components.

These activities will result in the preliminary version of the multimodal data management platform, which will be presented in the corresponding deliverable D5.2.
7. References


Web references have been listed as footnotes throughout the document.
8. LIST OF FIGURES

Figure 1 - Video Mining Analysis ................................................................. 9
Figure 2 - Composing Desk enhanced with Contextualising Tool .......................... 10
Figure 3 - Multimodal platform catering both for the journalist and the end-user's use-cases .................................................................................................................. 11
Figure 4: Architecture design, first iteration ...................................................... 19
Figure 5: Split between unstructured media analyzer and MAM ....................... 20
Figure 6: Split between on-demand workers and batch workers ....................... 20
Figure 7: Architecture design, third iteration .................................................... 21
Figure 8: Component overview ......................................................................... 23
Figure 9: Collecting data from a dump file ...................................................... 27
Figure 10: Collection of crawler data ................................................................. 28
Figure 11: Data preprocessing .......................................................................... 29
Figure 12: Queue managing of on-demand processes ....................................... 30
Figure 13: Queue managing of batch processes .............................................. 31
Figure 14: Batch worker .................................................................................. 32
Figure 15: On-demand workers ....................................................................... 33
Figure 16: Stakeholders of the EUMSSI Framework ....................................... 34
Figure 17: EUMSSI multimedia data flow ....................................................... 35
Figure 18: Development plan .......................................................................... 37
9. GLOSSARY

API: an application programming interface (API) specifies how some software components should interact with each other.

CAS: Content Addressable Storage. In this document, we use the term EUMSSI CAS for the Solr core that contains the aligned and enriched multimedia data. Not to be confused with CAS Object.

CAS Object. An UIMA document (more precise, a Subject of Analysis) that adheres to the Common Analysis System (CAS) API.

MAM: Multimedia Asset Management. In this document, the term EUMSSI MAM refers to the Unstructured Data Repository.

REST: Representational state transfer, a protocol for Web service APIs.

Solr: An open source enterprise search platform from the Apache Lucene project.

UIMA (Unstructured Information Management Architecture) is a component software architecture for the development, discovery, composition, and deployment of multimodal analytics for the analysis of unstructured information and its integration with search technologies

UML: Unified Modelling Language, a general-purpose modelling language in the field of software engineering, which is designed to provide a standard way to visualize the design of a system.