This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement n: 761999

EasyTV: Easing the access of Europeans with disabilities to converging media and content.

D5.5 - Integration and technical testing plan mid-term report

EasyTV Project

H2020. ICT-19-2017 Media and content convergence. – IA Innovation action.

Grant Agreement n°: 761999

Start date of project: 1 Oct. 2017

Duration: 30 months

Document. ref.: D5.5

Disclaimer

The EasyTV Consortium consists of the following partners:

	Partner Name	Short name	Country
1	Universidad Politécnica de Madrid	UPM	ES
2	Engineering Ingegneria Informatica S.P.A.	ENG	IT
3	Centre for Research and Technology Hellas/Information Technologies Institute	CERTH	GR
4	Mediavoice SRL	MV	IT
5	Universitat Autònoma Barcelona	UAB	ES
6	CorporacióCatalana de MitjansAudiovisuals SA	CCMA	ES
7	ARX.NET SA	ARX	GR
8	Fundación Confederación Nacional Sordos España para la supresión de barreras de comunicación	FCNSE	ES
9	Unione Italiana dei ciechi e degli ipovedenti	UICI	IT

PROGRAMME NAME:	**H2020. ICT-19-2017 Media and Content Convergence – IA Innovation Action**
PROJECT NUMBER:	761999
PROJECT TITLE:	EASYTV
RESPONSIBLE UNIT:	ENG
INVOLVED UNITS:	ENG, CERTH, MV, ARX
DOCUMENT NUMBER:	D5.5
DOCUMENT TITLE:	Integration and technical testing plan mid-term report
WORK-PACKAGE:	WP 5
DELIVERABLE TYPE:	Report
CONTRACTUAL DATE OF DELIVERY:	31-10-2018
LAST UPDATE:	29-10-2018
DISTRIBUTION LEVEL:	PU

Distribution level:

PU = *Public,*

RE = *Restricted to a group of the specified Consortium,*

PP = *Restricted to other program participants (including Commission Services),*

CO= *Confidential, only for members of the LASIE Consortium (including the Commission Services)*

Document History

VERSION	DATE	STATUS	AUTHORS, REVIEWER	DESCRIPTION
v. 0.1	20/07/2018	Draft	Giuseppe Vitolo ENG	Table of Contents definition and document structure
v. 0.2	26/07/2018	Draft	Giuseppe Vitolo ENG	Table of Contents changes
v. 0.3	02/08/2018	Draft	Giuseppe Vitolo ENG	Table of Contents changes
v. 0.4	11/10/2018	Draft	Giuseppe Vitolo ENG	Preliminary version
v. 0.5	23/10/2018	Draft	Giuseppe Vitolo ENG Kosmas Dimitropoulos CERTH	Crowdsourcing platform test integration
v. 0.6	25/10/2018	Draft	Giuseppe Vitolo ENG	Minor corrections
v. 0.7	26/10/2018	Preliminary version	Silvia Uribe UPM Francesc Mas Peinado CCMA	Review and comments
v. 0.8	29/10/2018	Final candidate	Giuseppe Vitolo ENG	Minor corrections

Definitions, Acronyms and Abbreviations

ACRONYMS / ABBREVIATIONS	DESCRIPTION
CD	Continuous Delivery
CI	Continuous Integration
GUI	Graphical User Interface
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
IEEE	Institute of Electrical and Electronics Engineers
REST	Representational State Transfer
SOAP	Simple Object Access Protocol
TTS	Text To Speech
V&V	Verification & Validation

Table of Contents

List of Figures

List of Tables

Executive Summary

This document is the fifth deliverable in WP5. It has been written by ENG with the collaboration of CERTH, MV and ARX. It focuses on integration testing, describing strategy and approaches aimed at testing the integration between the EasyTV components that make up the whole EasyTV multi terminal technical platform.

The first chapter introduces the concept and purposes of software testing and integration testing in particular.

The second chapter describes the elements to be integrated, the integration testing approaches and the integration testing sequence that will be used.

The third chapter gives a detailed description of the tests that will be performed when testing the integration between the different EasyTV components.

The fourth chapter highlights some useful tools that can be used to assist in the integration testing process.

Finally, the fifth chapter outlines conclusions and future work.

1. INTRODUCTION

Software testing is any activity aimed at evaluating an attribute or capability of a program or system and determining that it meets its required results [1]. It is an integral part in software development and it is deployed in every phase in the software development cycle.

Testing is performed for the following purposes:

- to identify defects (bugs or technical errors) and to subsequent correct those identified defects;
- for verification & validation (V&V). According to the IEEE Standard Glossary of Software Engineering Terminology, verification is defined as "The process of evaluating a system or component to determine whether the products of a given development phase satisfy the conditions imposed at the start of that phase" [2]. Validation, on the other hand, is defined as "The process of evaluating a system or component during or at the end of the development process to determine whether it satisfies specified requirements" [2]. Testing is therefore used as a tool in the V&V process;
- for reliability estimation, where reliability is defined as “the probability of failure-free operation for a specified period of time in a specified environment” [2].

There are many ways to categorize testing methods and testing techniques. By scope, software testing can be categorized as follows: unit testing, component testing, integration testing and system testing. The present document deals with **integration testing**, whose goal is to determine if the independently developed EasyTV components work correctly when they are connected and interacting with each other.

Integration testing assumes that the EasyTV components work as expected if taken individually (thus they have already been unit tested) and then groups them on the basis of the system architecture, executing tests described in chapter 3 “Test Description”, which covers all the tests identified so far. Further additions to the testing plan will be made as the development of the components and the integration process goes on: the integration and technical test plan final report will be available in D5.10.

2. INTEGRATION STRATEGY

This chapter describes the EasyTV components (modules) that will be integrated into a complete system (with the exclusion of the EasyTV Service Catalogue, of the EasyTV Service Registry and of the EasyTV Service Development Kit, that will be introduced in a subsequent phase and whose integration testing details will be found in the final integration testing report) and the integration testing approaches that can be used to accomplish the desired result. Part of the information regarding the single modules is drawn from D1.4 ("Final release of the EasyTV System Architecture"). Finally, the whole module integration sequence is outlined.

2.1. Elements to be integrated

2.1.1. Service Manager

The Service Manager will be the main communication hub between the broadcasters' premises, and the different components, modules, and services at the internal of the EasyTV platform. It will act as a gateway and orchestrator of the full platform and it will allow an abstraction of the work and processes that can involve multiple modules.

The Service Manager will have two main components:

- A **web graphical user interface** (GUI), which will allow the broadcaster to request content from the EasyTV platform in a centralized and unified way. This web GUI will keep track of their request, status, and serve the contents to the broadcaster premises in a user-friendly way without the need of knowing the underlying structure of the EasyTV modules, services, and components.
- A **REST API**, acting as a middleware between the user interface (and other possible services that the broadcasters may deploy in the future) and the EasyTV services.

The Service Manager will be able to address the specific content (accessibility and audiovisual contents) request to the appropriate EasyTV component where the repository for that content resides. If the requested content is available, it will be directly served to the broadcaster. If the requested content is not available (or doesn't exists yet), the Service Manager will generate the needed tasks in the EasyTV modules, allowing the broadcaster to know in every moment the status of the request.

As some of the tasks could be dependant of previous tasks, the Service Manager will take care of the correct workflow progress and monitoring.

2.1.2. Automatic Descriptive Narratives Module

This module generates a textual file similar to a subtitles file that contains valuable automatically generated data obtained from the intrinsic information of the multimedia contents and from different external information sources, such as the metadata associated to the file and the television guides. This information is related to the preferences of the user and it is suitable for obtaining real-time data about the contents that the user is watching. Moreover, this process will be completed with the textual information extraction from the video in order to provide more data that can improve the experience of the users, especially if they are visually impaired.

Once the audionarrative file is generated, the file is stored in the accessibility contents database through the service manager.

2.1.3. Automatic Voice Synthesis of Subtitles Module

The aim of this module is to provide spoken subtitles by using a TTS (Text-To-Speech) service available on the server side and another TTS service on the cloud for a more flexible and effective solution. Subtitles will be provided by the subtitling production platform available in EasyTV or

directly through a Web Based application that can be used by professional users. The Automatic voice synthesis of subtitles will include a set of functionalities available through two different interfaces: the Web API interface and the Web GUI interface. The Text To Speech Service (TTS) (local or on the cloud) will be also used for translating any text in an audio track to be included or mixed to an audio video streaming content, for example for audio narratives.

2.1.4. **Image Enhancement Module**

The main objective of image enhancement is the processing of an image for obtaining a more suitable result that the original image/video, which is adapted to the requirements of the user, especially when this user has impaired vision. Digital image enhancement techniques provide a multitude of choices for improving the visual quality of images. Appropriate choice of such techniques is greatly influenced by the imaging modality, task at hand and viewing conditions. The variety of algorithms commonly used for image enhancement is large, depending on the available processing resources and the real-time limitation. There are multiple techniques based on spatial domain techniques, with special reference to histogram processing and point processing methods.

On the EasyTV Cloud Platform this is the module that automatically detects texts and faces of an audiovisual content. This module generates a text file with the position where the content to be magnified is located. *Once* the text file is generated, the file is stored in the accessibility content database through the service manager. Then, the client application can obtain the text file and magnifies the content with the information obtained.

2.1.5. **Clean Audio Module**

This module allows the user to change the balance of the audio mix according to the listening environment or personal needs, obtaining important benefits for the hearing-impaired, including the people affected by hearing loss that increases gradually with age.

The clean-audio component interacts with the sound from the source multimedia content and develops a process for equalization of frequencies and advanced techniques of filtering for enhancing the intelligibility of the audio by identifying, processing and reinforcing the bandwidth where the most important audio information is located, especially the voices.

2.1.6. **Subtitle Production Module**

The aim of the Subtitle Production Module is to facilitate the translation jobs in order to achieve a good quality of translations in a reasonable term. With this objective in mind, two components have been included in the workflow of subtitle translation: the **Automatic Subtitles Translation Component** and the **Human Subtitling Production Tool**.

The Automatic Subtitles Translation Component provides a first translated version of the subtitles in the languages of interest indicated by the broadcaster/content owner. This is achieved by using free translation services available in the cloud. The translation process takes place as soon as a new job is mandated to the platform, and its results are also stored in the Data Storage Component and used as a starting point by the editorial users of the crowdsourcing platform in the Human Subtitling Production Tool.

The Human Subtitling Production Tool consists of a web-based interface that allows the collaborative users to perform the requested translation tasks. The users will be granted to the tool by means of the crowdsourcing platform.

2.1.7. **Sign Language Production Module**

The Sign Language Production module is responsible for the creation, validation and exploitation of the sign language content in different languages for easing the access of hearing impaired individuals to media content and services.

This module is made up of the following components:

- Sign Language Crowdsourcing: the purpose of this procedure is to allow users to contribute with sign languages and translations of available signs, creating a multilingual sign language repository. The administrators of the EasyTV Crowdsourcing platform will be responsible for defining Sign Language Tasks (SLTs) by proposing words/sentences that need to be translated in sign language and distribute these SLTs to expert sign language users. The users will then use a capturing setup in order to perform the requested signs, and by using a capturing component, record and upload the data to the Crowdsourcing Platform. The administrators will validate the users' contributions by stating an SLT as completed. Afterwards, the multilingual ontology will be responsible for annotating data in order to create a 1-to-1 mapping between the recorded signs, the text corresponding to the signing, and the concepts in the multilingual ontology. This will allow to automatically establish translation relations among signs in different languages.
  The enriched multilingual ontology can also be linked with existing repositories that contain signs in different languages (e.g. Spreadthesing), laying the foundations for the creation of a sub-cloud of resources that contain sign language data. Finally, the motion data along with the corresponding text and concept translation will be stored in a sign language repository for use by the other EasyTV services/modules. The Sign Language Crowdsourcing will also allow a user to search and view already recorded signs.
- Sign Language Capturing component: the purpose of this component is to allow users to record signs and upload them to the EasyTV Crowdsourcing Platform. Initially, the user will connect to the crowdsourcing platform and through his/her account, accept the assigned SLTs which recommend the signing of specific words or sentences. However, the user is not limited to completing the SLTs but he/she can perform his/her own signs given that he/she annotates them correctly. The Sign Language Capturing component will be responsible for processing the recorded data in order to extract valuable motion information that will be used for the correct differentiation among different signs. The recorded data will also accompany a text describing the signing sequence (word/sentence).
  At the next phase, there is an optional possibility of building a semi-automatic annotation algorithm that will accept the recorded data and the text describing them and output a data annotation that breaks down the video in frames and provides starting and ending frame numbers for each recorded sign. The semi-automatic annotation tool will be based on accurate and robust machine learning techniques applied on already learned signs. Finally, the expert user/signer will be able to view data annotation and perform corrections before uploading the data on the Sign Language Crowdsourcing component.
- Realistic Avatar: the purpose of this component is to allow the visualization of the recorded signs using an avatar. This component will communicate directly with the data repository of the Sign Language Crowdsourcing component and will acquire and play the sign representations using 3D motion data of face and hands.

2.1.8. **Hyper-personalization Module**

The EasyTV hyper-personalisation module will be built on top of profiling- user experience mining techniques; extraction, abstraction, homogenisation of dynamic user profiles and innovative interaction techniques for interface adaptation. This module will be able to handle dynamic and continuous changes in user experience in a user-transparent way in order to recommend new personalised services, dynamically adapted to the current context and device of the user.

The EasyTV hyper-personalisation module will consist of the following sub-components:

- EasyTV user models: the EasyTV user models will define user needs and preferences, including also the disabilities and functional limitations of the user. The structure of the EasyTV user models will be based on previous work conducted in previous projects, such as Cloud4All and VERITAS, as well as on the user models proposed by the VUMS cluster. Further extensions of the aforementioned user models will be developed to cover the EasyTV specific needs.

- User model editor**:** the EasyTV User Model Editor will be a web-based tool that will enable the easy creation and editing of EasyTV end user models through intuitive web forms.
- Hybrid matchmaker: the EasyTV hybrid matchmaker will perform matchmaking between user needs and preferences defined in user profiles and UI capabilities, accessibility features specifications and DASH streaming services specifications.
- UI adaptation and accessibility features configuration module: this module will take as input the results of the hybrid matchmaker and will perform automatic turn on and configuration of accessibility features (e.g. volume, rate, pitch, colour preferences, etc.) that are built into different TV operating systems, applications and embedded devices that will be supported.
- Content adaptation module: the EasyTV content adaptation module will perform content adaptation based on the results of the hybrid matchmaker in order to offer streaming content in the best possible form for a specific user (e.g. by selecting the audio that corresponds to user's language).

2.1.9. Crowdsourcing Platform

The Crowdsourcing Platform provides an infrastructure for implementing the sign language production and subtitle production procedures. More specifically, it includes functionality for task definition, distribution and validation, and its purpose is to allow the creation and management of professional user profiles and the access of users in the sign language and subtitle production procedures through web Graphical User Interfaces (GUIs).

2.2. Integration testing approaches

The most common software integration testing approaches are big bang (non-incremental), bottom-up, top-down and sandwich (also called mixed or hybrid). The following subsections describe each testing approach.

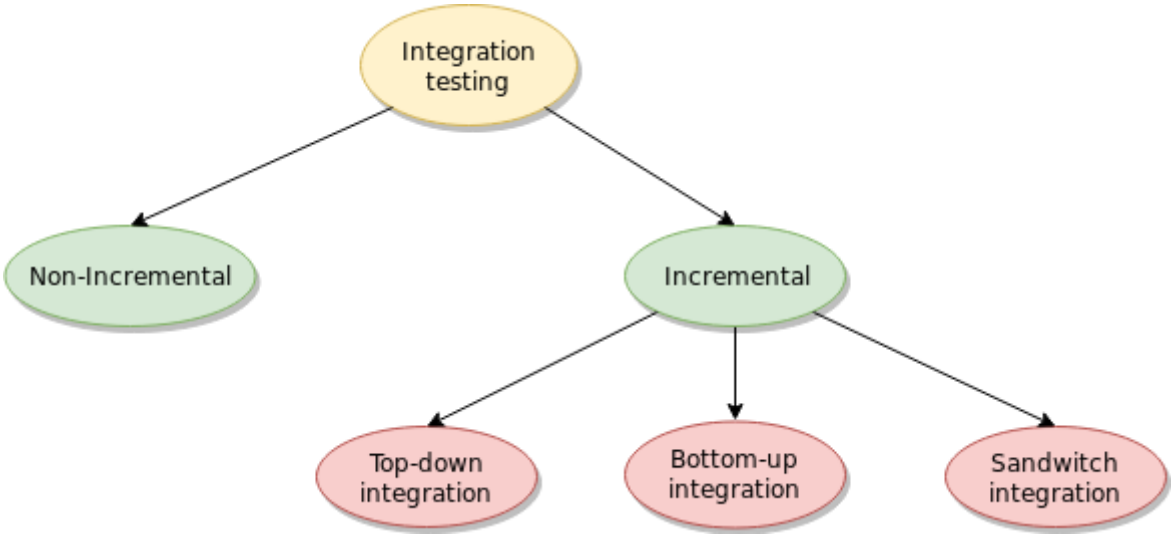

Figure 1 - Integration testing approaches

2.2.1. Big Bang (non-incremental)

Big Bang is a type of integration testing in which software elements are combined all at once into an overall system, rather than in stages [2]. It is important to avoid confusion between this approach and system testing: the former only tests the interactions between the modules, while the latter tests the entire system. This kind of integration testing strategy is convenient for small

systems but has major disadvantages, such as difficulties in precise defect localisation and the higher chance of critical failures because all the modules are integrated and tested together at the same time.

2.2.2. **Bottom-up**

In bottom-up integration testing, the starting point are modules that have no dependency (lowest level modules): each module is integrated with another module that uses it and the pair is tested. The two integrated modules are considered as one single module and the process is repeated until the top of the component hierarchy is reached. Upper level modules not yet integrated or not available can be replaced by a “driver”, which is a software module (caller module) used to invoke a module under test [2]. One of the main advantages of this kind of integration testing approach are that lower level modules receive more testing and fault localisation is easier. This is going to be the prevalent approach to integration testing in the context of the EasyTV project, since more emphasis should be given to the low level modules which expose the main EasyTV functionalities.

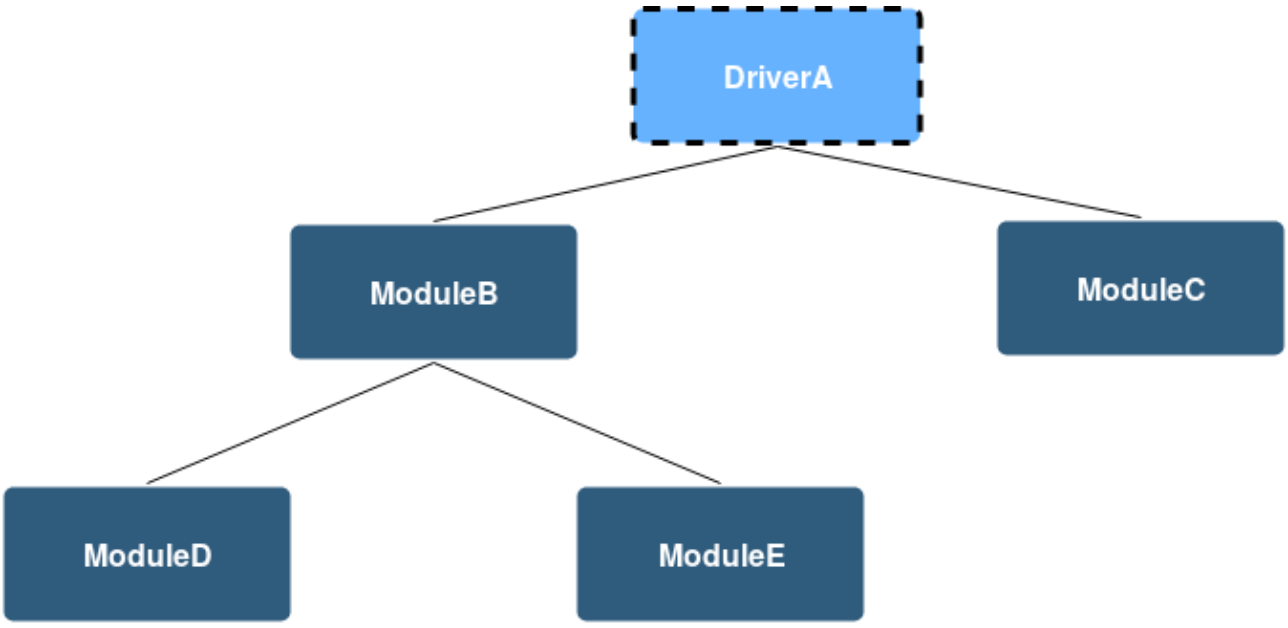

Figure 2 - Bottom-up integration testing

2.2.3. **Top-down**

In top-down integration testing, the starting point are modules that do not depend on any other module (highest level modules): each module is integrated with a called module and the pair is tested. The two integrated modules are considered as one single module and the process is repeated until the bottom of the component hierarchy is reached. If a called module is not available it is necessary to use a “stub”, which is skeletal or special-purpose implementation of a software module, used to develop or test a module that calls it or is otherwise dependent on it [2]. The main advantage of this kind of approach is that it is generally possible to obtain an early prototype and verify major decisions early in the test process, but the basic functionality of the software is usually tested at the end of the cycle.

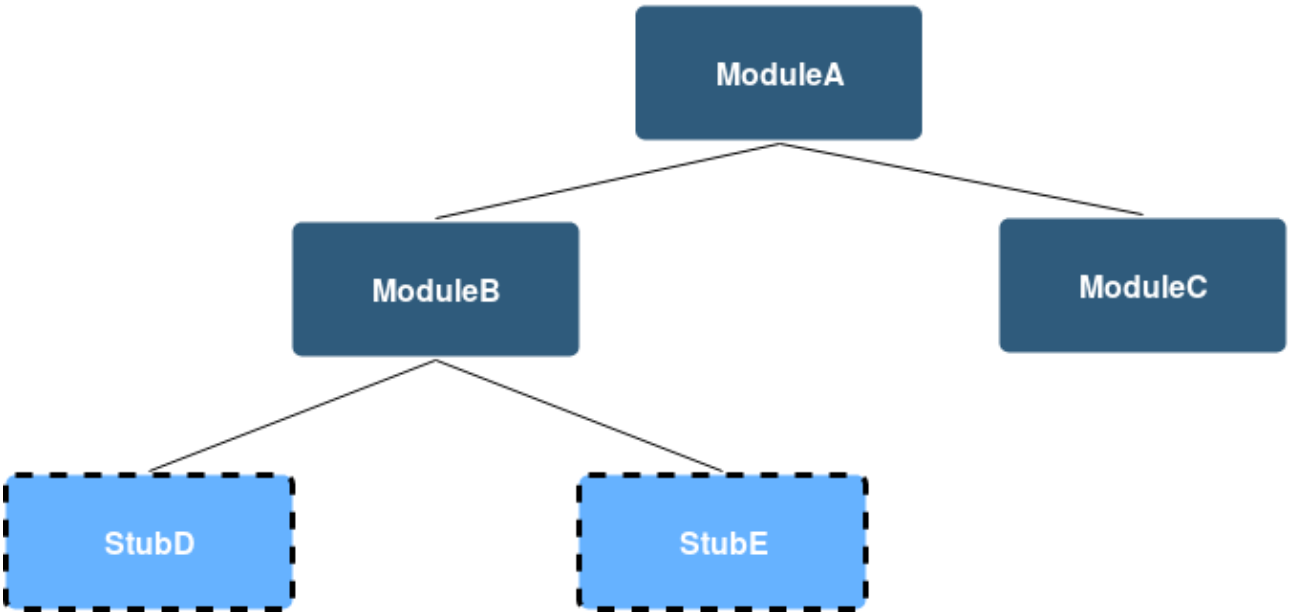

Figure 3 - Top-down integration testing

2.2.4. Sandwich (mixed)

Sandwich (also called “mixed” or “hybrid”) integration testing is an approach where the bottom-up and the top-down approaches are combined. Since this approach involves both forms of testing, the system to be tested is viewed as having three layers: a target layer in the middle, a top layer (above the middle layer) and a bottom layer (below the middle layer). Tests can be conducted in parallel on each layer by using “drivers” and “stubs” appropriately.

It has the advantages and disadvantages of both integration testing approaches. It is useful to integrate a large number of components in a very large project, but it often requires more resources and large teams to be correctly performed.

2.3. Integration sequence

In this section the sequence in which the EasyTV modules will be integrated is identified. As mentioned in the previous section, the prevalent approach used in integration testing strategy will be the bottom-up approach, since it best fits the project development workflow.

As noted in D1.4 (“Final release of the EasyTV system architecture”) the component named “Service Manager” will act as the main communication hub between the broadcasters’ premises and the different components, modules and services which will run on the EasyTV platform, so it is advisable to start integrating the lowest level modules with it. Sequential steps are outlined below:

1. Service Manager – Automatic Descriptive Narratives Module
2. Service Manager – Automatic Voice Synthesis of Subtitles Module
3. Service Manager – Clean Audio Module
4. Service Manager – Image Enhancement Module
5. Crowdsourcing Platform – Sign Language Production Module
6. Crowdsourcing Platform – Subtitles Production Module
7. Service Manager – Sign Language Production Module
8. Service Manager – Subtitles Production Module

There are three more component which are going to interact with the EasyTV platform, namely the Service Registry, the Service Catalogue and the Service Development Kit. Integration testing of the

aforementioned components will be outlined in the final report (D5.10).

3. TEST DESCRIPTION

This chapter covers the integration tests that have been identified so far. Note that not every element outlined in section 2.1 (“Elements to be integrated”) is present: the complete integration testing plan will be available in D5.10 (“Integration and technical test plan final report”).

3.1. Service Manager – Automatic Descriptive Narratives integration

Test ID	SM-ADN-001
Test Description	The Service Manager should send (upload) the target multimedia content (along with metadata associated with the content itself) to the Automatic Descriptive Narratives Module.
Prerequisites	Multimedia content file and associated metadata are available and ready to be uploaded.
Expected result	The Automatic Descriptive Narrative Module successfully obtains the required data.

Table 1 - SM-ADN-001 integration test

Test ID	SM-ADN-002
Test Description	The Automatic Descriptive Narratives module should notify the Service Manager when the processing has been completed.
Prerequisites	The Automatic Descriptive Narrative module can read and process the input multimedia content file and its associated metadata.
Expected result	The Service Manager gets a “processing successfully completed” notification.

Table 2 - SM-ADN-002 integration test

Test ID	SM-ADN-003
Test Description	The Service Manager should get the output audionarrative file from the Automatic Descriptive Narratives module.
Prerequisites	The Automatic Descriptive Narrative module has completed its processing.

Expected result	The Service Manager gets an audionarrative file.

Table 3 - SM-ADN-003 integration test

3.2. Service Manager – Automatic Voice Synthesis of Subtitles integration

Test ID	SM-AVSS-001
Test Description	The Service Manager should send (upload) a subtitle file to the Automatic Voice Synthesis of Subtitles module.
Prerequisites	Subtitle file is available and ready to be uploaded.
Expected result	The Automatic Voice Synthesis of Subtitles module successfully obtains the required data.

Table 4 - SM-AVSS-001 integration test

Test ID	SM-AVSS-002
Test Description	The Automatic Voice Synthesis of Subtitles module should notify the Service Manager when the processing has been completed.
Prerequisites	The Automatic Voice Synthesis of Subtitles module can read and process the input subtitles file.
Expected result	The Service Manager gets a “processing successfully completed” notification.

Table 5 - SM-AVSS-002 integration test

Test ID	SM-AVSS-003
Test Description	The Service Manager should get the output audio track file from the Automatic Voice Synthesis of Subtitles module.
Prerequisites	The Automatic Voice Synthesis of Subtitles module input file processing has been completed.
Expected result	The Service Manager gets an audio track file.

Table 6 - SM-AVSS-003 integration test

3.3. Service Manager – Image Enhancement integration

Test ID	SM-IE-001
Test Description	The Service Manager should send (upload) a multimedia video file to the Image Enhancement module.
Prerequisites	Multimedia video file is available and ready to be uploaded.
Expected result	The Image Enhancement module successfully obtains the required data.

Table 7 - SM-IE-001 integration test

Test ID	SM-IE-002
Test Description	The Image Enhancement module should notify the Service Manager when the processing has been completed.
Prerequisites	The Image Enhancement module can read and process the input video file.
Expected result	The Service Manager gets a “processing successfully completed” notification.

Table 8 - SM-IE-002 integration test

Test ID	SM-IE-003
Test Description	The Service Manager should get the output position file (a text file with the position of the content to be magnified) from the Image Enhancement module.
Prerequisites	The Image Enhancement module input file processing has been completed.
Expected result	The Service Manager gets an output position text file.

Table 9 - SM-IE-003 integration test

3.4. Service Manager – Clean Audio integration

Test ID	SM-CA-001

Test Description	The Service Manager should send (upload) a multimedia video (and audio) file to the Image Enhancement module.
Prerequisites	Multimedia video file is available and ready to be uploaded.
Expected result	The Image Enhancement module successfully obtains the required data.

Table 10 - SM-CA-001 integration test

Test ID	SM-CA-002
Test Description	The Clean Audio module should notify the Service Manager when the processing has been completed.
Prerequisites	The Clean Audio module can read and process the input multimedia file.
Expected result	The Service Manager gets a “processing successfully completed” notification.

Table 11 - SM-CA-002 integration test

Test ID	SM-CA-003
Test Description	The Service Manager should get the output audio file from the Clean Audio module.
Prerequisites	The Clean Audio module input file processing has been completed.
Expected result	The Service Manager gets an audio track file.

Table 12 - SM-CA-003 integration test

3.5. Service Manager – Subtitle Production integration

Test ID	SM-SP-001
Test Description	The Service Manager should send (upload) the subtitle file in the original language to the Subtitle Production module. The Service Manager should provide the desired output language.
Prerequisites	Subtitle file in the original language is available

	and ready to be uploaded.
Expected result	The Subtitle Production module successfully obtains the required data.

Table 13 - SM-SP-001 integration test

Test ID	SM-SP-002
Test Description	The Subtitle Production module should notify the Service Manager when the processing has been completed.
Prerequisites	The Subtitle Production module can read and process the input subtitle file.
Expected result	The Service Manager gets a “processing successfully completed” notification.

Table 14 - SM-SP-002 integration test

Test ID	SM-SP-003
Test Description	The Service Manager should get the output subtitle file in the desired language.
Prerequisites	The Subtitle Production module input file processing has been completed.
Expected result	The Service Manager gets a translated subtitle file.

Table 15 - SM-SP-003 integration test

3.6. Service Manager – Sign Language Production integration

Test ID	SM-SLP-001
Test Description	The Service Manager should send (upload) the subtitle file in the original language to the Sign Language Production module. The Service Manager should send (upload) the multimedia video file to the Sign Language Production module.
Prerequisites	Subtitle file in the original language is available and ready to be uploaded. Multimedia video file is available and ready to be uploaded.

Expected result	The Sign Language Production module successfully obtains the required data.

Table 16 - SM-SLP-001 integration test

Test ID	SM-SLP-002
Test Description	The Sign Language Production module should notify the Service Manager when the processing has been completed.
Prerequisites	The Sign Language Production module can read and process the input subtitle file.
Expected result	The Service Manager gets a “processing successfully completed” notification.

Table 17 - SM-SLP-002 integration test

Test ID	SM-SLP-003
Test Description	The Service Manager should get the sign avatar video.
Prerequisites	The Sign Language Production module input file processing has been completed.
Expected result	The Service Manager gets an avatar video file.

Table 18 - SM-SLP-003 integration test

3.7. Crowdsourcing Platform – Subtitle Production integration

Test ID	CP-SP-001
Test Description	The Crowdsourcing platform should get a description of a new translating task.
Prerequisites	The Automatic Subtitles Translation Component has produced translation files which are stored in the Data Storage Component.
Expected result	The Crowdsourcing platform reads the stored files from the storage component.

Table 19 - CP-SP-001 integration test

3.8. Crowdsourcing Platform – Sign Language Production integration

Test ID	CP-SLP-001
Test Description	The files generated from the Sign Language Production module should be uploaded to the Crowdsourcing Platform.
Prerequisites	The Sign Language Production module has produced the expected files (video, annotation) in the correct format.
Expected result	The Crowdsourcing Platform confirms that files are uploaded and are of correct format. The files are saved in the platform and are accessible.

Table 20 - CP-SLP-001 integration test

Test ID	CP-SLP-002
Test Description	Crowdsourcing Platform should invoke a service on the Multilingual Ontology Module passing the files containing signs.
Prerequisites	All necessary linguistic information exists in the Crowdsourcing in the expected format.
Expected result	The Multilingual Ontology responds by returning a file containing the concepts corresponding to the given signs.

Table 21 - CP-SLP-002 integration test

4. TOOLS

Integration testing is a complex process made up of many tasks, so it is useful to have a certain number of tools which can support testers and system integrators to achieve the desired result. Some of these tools are useful to perform contract testing, which is a way to test that a specific service provides an API that its client expects, while other tools are useful to run performance and load tests. In the following sections a brief description of each tool is given.

4.1. SoapUI

SoapUI [3] is a tool to test SOAP Web Services and RESTful or HTTP based services. There is an open source version of this tool as well as a commercial one, called “SoapUI Pro”, which has extra functionalities such as live API traffic record, advanced scripting and CI/CD infrastructure integration.

It is possible to use this tool both for functional testing and for load testing. It is not necessary for the user to be a developer, since a powerful GUI enables the user to create and test complex scenarios by using a drag-and-drop interaction type. In this way it is easy to create test suites and test cases without learn a domain-specific language.

Indeed, this is one of the most powerful tool among various solutions that can be used to perform API testing.

4.2. Postman

Postman [4] is a complete RESTful API development environment. Like SoapUI, this tool has a nice GUI which allows the user to easily create and perform the desired number of tests, but it also has a command line extension called Newman, which allows the user to run and test a Postman collection (a set of tests) directly from the command line.

Among its features there is the “Collection Runner”, which enables a developer or a tester to run all requests within a collection and to use data files as parameters for each iteration, and the “Mock Servers” feature, which simulates a back-end service and makes possible to mock a single endpoint or an entire API.

4.3. Apache Jmeter

Apache Jmeter [5] is an open source application designed to load test functional behavior and measure performance. It can be used to perform load testing on many different applications and protocol types (e.g. HTTP, HTTPS, SOAP Web Services, REST Web Services). It includes a Test IDE that allows to record a test plan from browsers or native applications and a command line to perform load test from any Java compatible OS (it is a pure Java application and should run correctly on any system that has a compliant Java implementation).

A key feature of this product is its extensibility: plugins can be created and loaded to increase the number of functionalities and to improve personalisation (in particular regarding data analysis and visualization).

5. CONCLUSIONS AND FUTURE WORK

This document presented the concept of integration testing and the approaches that will be used to integrate the EasyTV platform modules. A list of elements to be integrated and an initial integration testing plan have been outlined, describing the integration sequence and specifying a set of integration tests to be performed. In addition, some tools which can aid in the integration process have been described.

Future work will be focused on providing a more accurate and complete testing plan, which will be available in D5.10 (“Integration and technical test plan final report”). Particular attention will be given to the integration testing planning and execution of the EasyTV Service Catalogue and the EasyTV Service Registry, along with the EasyTV Service Development Kit. Integration tests presented in Chapter 3 (“Test Description”) will be re-evaluated before being executed, in light of the fact that the EasyTV modules are in an on-going development phase.

6. REFERENCES

[1] William C. Hetzel, The Complete Guide to Software Testing, 2nd ed. (last accessed 25 Oct 2018)

[2] IEEE, IEEE Standard Glossary of Software Engineering Terminology (IEEE Std 610.12-1990). (last accessed 25 Oct 2018)

[3] SoapUI. https://www.soapui.org/. (last accessed 25 Oct 2018)

[4] Postman. [Online]. https://www.getpostman.com/ (last accessed 25 Oct 2018)

[5] Apache JMeter. https://jmeter.apache.org/. (last accessed 25 Oct 2018)