Pilot 2 Demostrator Architecture, Integration Plan and Evaluation Methodologies

Deliverable D5.3

CONTENT4ALL File ID:	CONTENT4ALL D5.3_Pilot 2 demonstrator architecture, integration plan and evaluation methodologies.docx
Version:	1.0
Deliverable number:	D5.3
Authors:	M. Giovanelli (FIN)
Contributors:	R. Bowden (UNIS), J. Fidacaro (FIN), C. Galkandage (UNIS), G. Meroni (FIN), V. Upadrasta (HFC), W. Paier (HHI)
Internal reviewers:	HFC, FIN
Work Package:	WP5
Task:	T5.6
Nature:	R – Report
Dissemination:	PU – Public
Status:	Final
Delivery date:	25.09.2019

Version and controls:

Version	Date	Reason for change	Editor
0.1	31/07/2019	First Draft	M. Giovanelli (FIN)
0.2	05/08/2019	Add G. Meroni (FIN) contributions	M. Giovanelli (FIN)
0.3	28/08/2019	Add J. Fidacaro (FIN) contributions	M. Giovanelli (FIN)
0.4	29/08/2019	Add V. Upadrasta (HFC) contributions	M. Giovanelli (FIN)
0.5	29/08/2019	Add R. Bowden (UNIS) contributions	M. Giovanelli (FIN)
0.6	30/08/2019	Update Timeline	M. Giovanelli (FIN)
0.7	02/09/2019	Add W. Paier (HHI) contributions	M. Giovanelli (FIN)
0.8	05/09/2019	Add C. Galkandage (UNIS) contributions	M. Giovanelli (FIN)
0.9	05/09/2019	Add R. Bowden (UNIS) contributions	M. Giovanelli (FIN)
0.10	06/09/2019	Add C. Galkandage (UNIS) contributions	M. Giovanelli (FIN)
0.11	10/09/2019	Finalization	M. Giovanelli (FIN)
0.12	10/09/2019	Internal reviewer submission	M. Giovanelli (FIN)
0.13	13/09/2019	Review	J. Fidacaro (FIN)
1.0	16/09/2019	Finalization	M. Giovanelli (FIN)

Acknowledgement: The research leading to these results has received funding from the European Union's Horizon 2020 Programme (H2020-ICT-2016-2, call ICT-19-2017) under grant agreement n° 762021.

TABLE OF CONTENT

LIST OF FIGURES

LIST OF TABLES

1. Introduction

1.1. Purpose

This document gives a general overview of the activities that will be executed as part of the *WP5 'System integration, Validation, Large pilot Set-up and User Testing'*.

Starting from a plan for integrating the components of the system, the deliverable describes the evaluation methodology that will be implemented in the large pilots and the main risks of the project.

1.2. Scope of the work

Scope of the document is to provide the details of integration and validation of the entire system corresponding to Pilot 2 Demonstrator (Phase 3). It describes the activities needed for integrating the different macro components into a complete and working system. It presents how the system will be deployed to support the Pilot 2 Demonstrator. It illustrates the methodology that will be applied for the system validation, so how the system will be tested into the pilots and how the outputs of the pilots will be assessed to produce an overall system evaluation.

1.3. Structure of the document

The structure of this deliverable consists of the following parts:

- Chapter 2 recalls the key components of the system architecture, described in D2.3, with a particular focus on the integration aspects and on data flows; it shows how it will be implemented and configured in the Pilot 2 Demonstrator in order to test the system capabilities.
- Chapter 3 presents the overall integration and validation plan, including all the activities needed for integrating the various components into a complete system, for testing the single features and finally for evaluating its capabilities
- Chapter 4 describes the methodology that will be applied to the system evaluation in the pilots, including the involvement of the potential users and the metrics for measuring their feedbacks
- Chapter 5 summarizes the outputs of the entire document

2. Architecture

The initial goal of the project is to enable the low-cost personalization of the content for deaf viewers, with no disruption to hearing viewers, while the long term goal is to achieve automatic sign-translation capabilities to a restricted discourse domain; the architecture will therefore reflect such different project goals. To this end, two different architectures have been defined: a "Phase 1" architecture that is needed to reach the short-term goal of the project and a "Phase 3" ("Final Phase") architecture that must satisfy the long-term goal of the project.

Within the Phase 3, the Pilot 2 Demonstrator will be developed and executed in order to satisfy the Requirements outlined into the deliverable "*D2.4 Final Requirements and Specifications for Pilot Demonstrator*". The main objective of the Pilot 2 Demonstrator (Phase 3) is to put in place a proof of concept system that enable automated sign-interpreted content creation in the form of an 3D virtual human (avatar) and send this content together with the main video so that users can choose which content to watch.

The infrastructure architecture that must be deployed is the one that has been designed in deliverable "*D2.3 Final Reference System Architecture*". In the following paragraph the architecture designed for Phase 3 will be described.

2.1. CONTENT4ALL Phase 3 Architecture

This high-level representation shows the logical composition and the logical interaction among the components: Broadcaster, Processing & Rendering Unit and User Terminals. The aforementioned components allow to achieve the project objectives, interacting with each other and following the flows described in Figure 1.

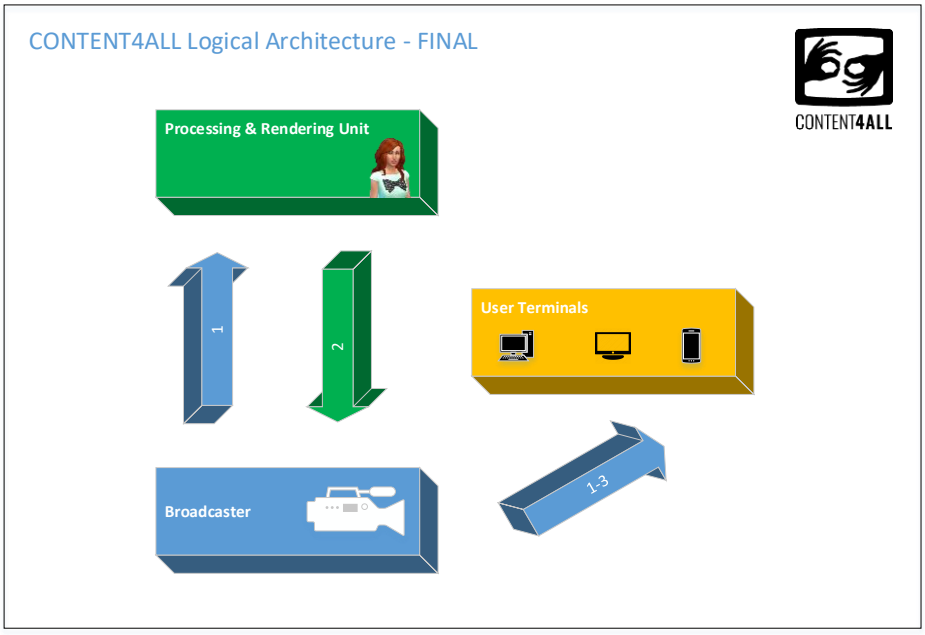

Figure 1 CONTENT4ALL Overall Logical Architecture Phase 3

1. The main broadcaster video is streamed towards the User Terminals
2. The Processing & Rendering Unit receives the main broadcaster video stream and reproduces the sign language through a 3D Photorealistic virtual human.
3. Both video streams (original and mixed one) are sent to the users' terminals (both for hearing and deaf people).

The Figure 2 describes the high-level architecture designed to satisfy the requirements of the Phase 3 of the CONTENT4ALL project. The reference architecture shows the different components and their interactions that will be implemented,

distinguishing between components provided by the project (blue) and components already available in broadcaster premises (orange) that will be used for the project piloting.

The main change respect to Phase 1 architecture is the removal of the Remote Studio and Handshape Analyzer components, replaced by the 3D Avatar Language Translator component that transmits data to the Photorealistic 3D model Renderer.

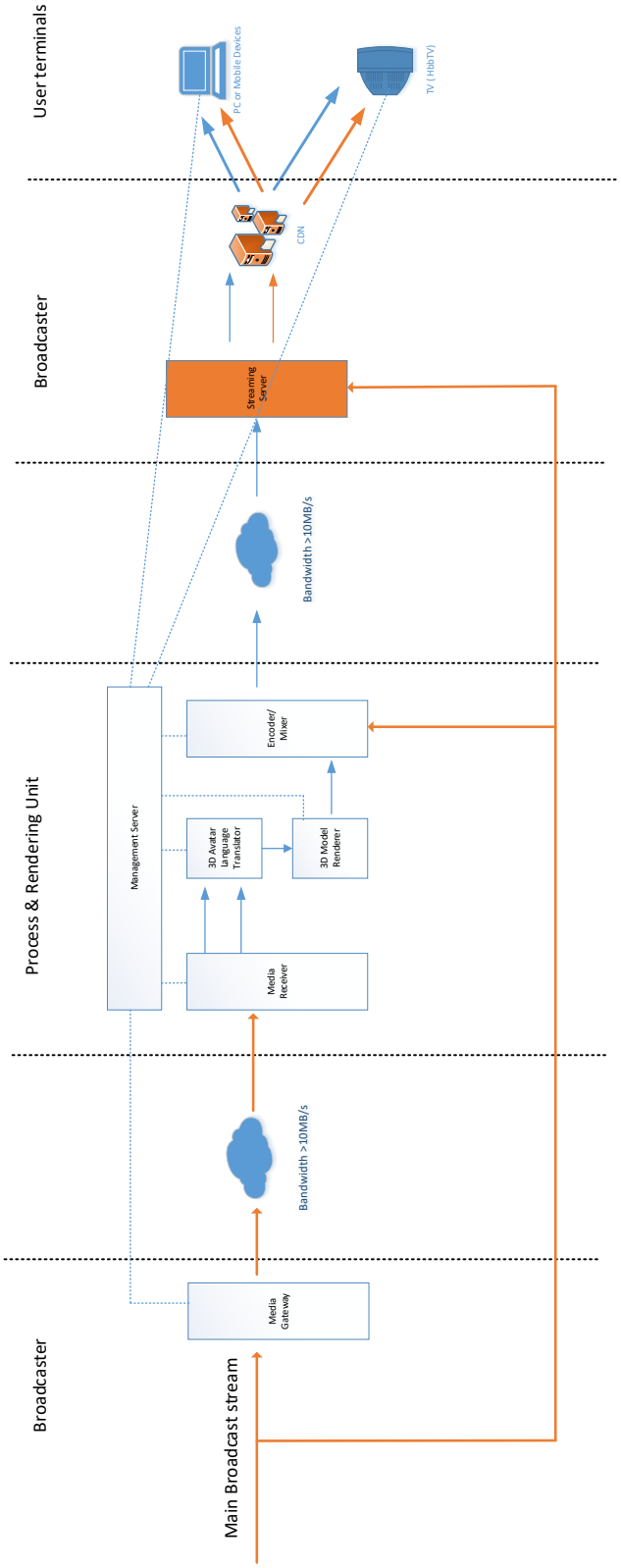

Figure 2 CONTENT4ALL Overall Architecture workflow

Logically the workflow can be divided in 3 components:

- Broadcaster
  - Main Broadcast stream
  - Origin Server
  - Media Gateway
- Processing & Rendering Unit
  - Management Server
  - Media Receiver
  - 3D Avatar Language Translator
  - 3D Model Renderer
  - Encoder/Mixer
- User Terminals
  - PC or Mobile Devices
  - TV (HbbTV)

2.2. Pilot 2 Demonstrator

During Phase 3 of the project, the CONTENT4ALL consortium will set up the architecture defined in Section 2.1.

The CONTENT4ALL frameworks will be tested in selected application contexts identified in the news-broadcasting domain (e.g. Weather News). The Main Broadcast stream will be sent to the Processing & Rendering Unit via the Media Gateway and forwarded to the 3D Avatar Language Translator. The 3D Avatar Language Translator will translate the input (i.e., video feeds) to a stream of data that will be the input to the Photorealistic 3D Model Renderer.

The model renderer component will render a 3D virtual human similar to the one provided during the ACTION-TV project[1] (see Figure 3 and Figure 4). As an improvement of this virtual human, the Photorealistic 3D Model Renderer component will add the rendering of the handshape. The Avatar's stream will be forwarded to the Mixer and Encoder component that in turn assembles together the main video content (provided by the broadcaster) and the photorealistic 3D virtual human into a single stream that will be further adapted (i.e. encoded in different formats) and delivered to HbbTV and mobile devices via broadband.

The mixed stream will be then sent to users through the Video Streaming Origin Server component. The translated content will be streamed to different clients via the broadband connection. This content will be streamed directly to televisions or Set-top boxes that are HbbTV-enabled. Users can decide to switch from the original broadcast video to the broadband signed version directly using the remote control of their device.

Since VRT does not to date support the standard HbbTV 2[2], a web application will be implemented during the project, being able to receive and reproduce both the original and the translated contents. In doing so, the audience of possible users of the content is extended to anyone who owns a device with an internet browser supporting the selected streaming protocol, be it a PC or a mobile phone.

The HbbTV app and web application receive the URL of signed stream from the "CONTENT4ALL Management Server" component.

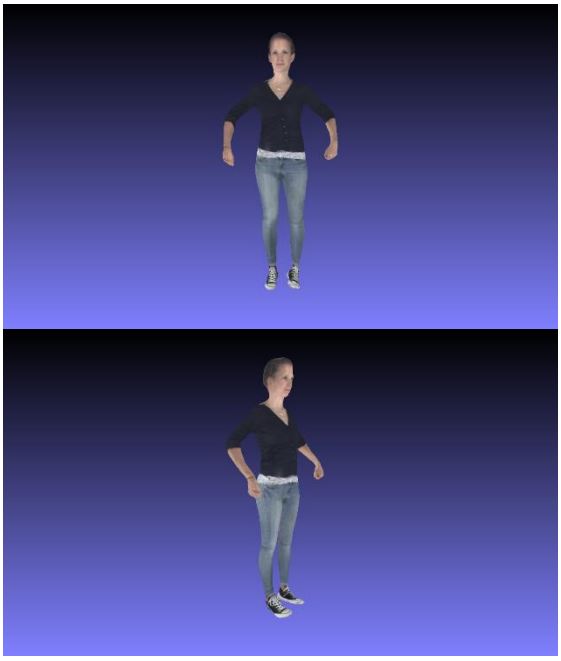

Figure 3 Example of female virtual human

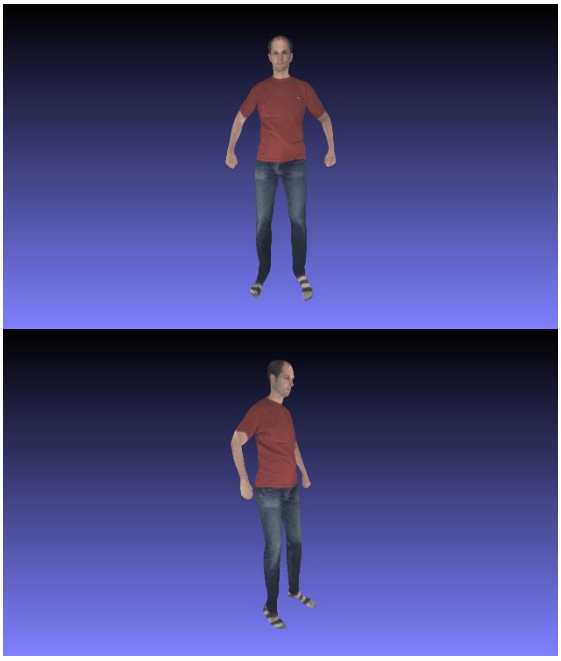

Figure 4 Example of male virtual human

1 https://cordis.europa.eu/project/rcn/191629_en.html

2 https://www.hbbtv.org/

2.2.1. Reference Scenario for Pilot 2 Demonstrator

The aforementioned Demonstrator will run on a local setup in order to simulate the workflow for the following reasons:

- the system is a proof-of-concept
- the Remote Studio is no longer necessary
- the technical conditions were already tested (Pilot 1 Demonstrator)

For this purpose, the system will be tested with the recording of Weather News and the final result (signing Avatar) will be evaluated. The current equipment and premises provided by the partners will be described in the following paragraphs.

Technical Requirements

This paragraph will define the technical requirements for running the Pilot 2 Demonstrator. As defined in paragraph 2.1, the different components of the overall CONTENT4ALL project architecture will be installed on different servers and devices (Ideally: 1 server for Broadcaster, 1 server for Processing & Rendering Unit and at least 1 TV running HbbTV and 1 PC running the Web App) meeting the minimum characteristics required by each component in order to correctly execute its job. To define the total characteristics that each server must comply with, the hardware and software requests required for each component will be specified below.

Media Gateway

Media gateway will change its behaviour with respect to the Pilot 1 Demonstrator (Phase 1) of the project.

Thanks to the experience obtained with the Pilot 1 Demonstrator, the Media Gateway for the Pilot 2 Demonstrator (Phase 3) has been remodelled with respect to the one proposed in D2.1. The new Media gateway will intercept the Main Broadcast Stream and stream subtitles to the Processing & Rendering Unit. In order to use the best possible audio transcription, the subtitles provided by the Broadcaster (obtained with re-speaking) will be used.

If there will be enough time and budget, automatic subtitle generation from audio will be experimented.

Software Requirements

Requirement	Type	Description
FFmpeg	Software	Open source software encoder, mixing, and media streaming library
Microsoft Visual Studio 2017 professional, Visual C++ -11	IDE	IDE for programming and software development
Microsoft .NET framework 4.5 or above	IDE/Software	Support libraries and programming environment
Microsoft Windows	Operating System	Version 8.1 – 64-bit

Hardware Requirements

Type	Item	Specification	description
Software decoding	CPU	8-cores of Intel 2.4GHz or above	Software decoding of compressed high-quality streams (CRF 20 configuration in FFmpeg libraries) video streams require at least 4 CPU cores per each 1080p stream.
Other hardware requirements	Memory	8-16 GB	Decoder, streaming and TCP buffers
	Ethernet NIC	10G and 100 Mbps NICs	Input streams: Main Broadcast Streams. Output streams: Raw transcription text data (subtitles).

Media Receiver

Media receiver is located within the **Processing & Rendering Unit** and is responsible for collecting the streams received from the **Media Gateway** and directing them to relevant components in the Processing & Rendering Unit.

Thanks to the experience obtained with the Pilot 1 Demonstrator (Phase 1), the Media Gateway for the Pilot 2 Demonstrator (Phase 3) has been remodelled with respect to the one proposed in D2.1. The new Media Gateway will intercept the Main Broadcast Stream and stream subtitles to the Processing & Rendering Unit. Due to this, also the Media Receiver will be remodelled. In fact, the Audio2Script component proposed in D2.1 is removed, so the Media Receiver will forward the subtitles directly to the 3D Avatar Language Translator.

Software Requirements

Requirement	Type	Description
Ubuntu OS	Operating System	Version 14.04, Linux kernel 3.19
Ubuntu build-essentials	gcc/g++ compilers and utils	Software compilation and packaging
dlib libraries	Software libraries	dlib is a modern C++ toolkit containing machine learning algorithms and tools for creating complex software in C++ to solve real world problems

Hardware Requirements

	Item	Specification	description
Software for subtitles forwarding	CPU	4-cores of Intel 2.4GHz or above	Software for subtitles forwarding has low requirements.
Other hardware requirements	Memory	8 GB	Streaming and TCP buffers
	Ethernet NIC	100 Mbps NICs	Input streams: Raw transcription text data (subtitles). Output streams: Raw transcription text data (subtitles).

3D Avatar Language Translator

The 3D Avatar Language Translator is the component responsible for converting a transcribed spoken audio track (subtitles) into the meta data suitable for animating a 3D signing virtual human. Translation between two written languages is usually accomplished in a single step. However, as sign is a visual language with no written form, we use sign glosses as intermediate representation. Translation can then be approached as a two-stage process where first the spoken language transcription is translated into sign gloss (a tokenised sign sequence) and then the sequence of sign glosses is converted into a sequence of animation parameters for the virtual human. Sign translation is an unsolved research problem so C4A will demonstrate this technology within a limited domain of discourse (e.g. Weather News), as a proof of concept demonstration.

Due to lack of data as a result of delays in the project, the 3D virtual human language translator is being trained using existing 2D sign data. Excellent progress has been made on the translation work using neural machine translation approaches and preliminary animation results in 2D are promising.

Software Requirements

Requirement	Type	Description
CUDA	Software	CUDA 10.0 parallel computing API
cuDNN	Software	cuDNN 7.3 deep neural network library
TensorFlow	Software	TensorFlow 1.14.0 Deep learning framework
PyTorch	Software	PyTorch 1.2.0 Deep Learning framework
NumPy	Software	Python scientific computation library
Python	Software	Python 3.7 Interpreter
OpenCV	Software	OpenCV 3.4.1 Python Library
SKLearn	Software	SKLearn 0.21.2 Library
Ubuntu or Windows	Operating System	Ubuntu 16.04, Linux Kernel 4.4 Windows 10 Build 18362

Hardware Requirements

	Item	Specification	description
3D Avatar Language Translator	CPU	Intel Core i-7 3.5GHz	Most processing is done via the GPU.
	GPU	NVIDIA GPU with CUDA compute capability of 7.5 and at least 11 GB of VRAM	Needed for GPU based deep learning computations of the handshape analyser.
Other hardware requirements	Memory	16 GB	Needed for buffering
	Ethernet NIC	10G and 100 Mbps NIC cards	Input streams: Raw transcription text data. Output streams: JSON stream.

Photorealistic 3D Model Renderer

The Content4All Photorealistic 3D Model Renderer is responsible for the creation, animation and rendering of the 3D Avatar.

In Phase 3, the virtual human rendering module will be based on a hybrid animation approach that combines geometry-based animation methods with image-based rendering techniques to achieve higher visual quality. Its input consists of animation control signals that will be generated by a sign language translation module. The output of the model renderer is a video of the animated virtual human. Both input and output data will be transmitted via network.

Software Requirements

Requirement	Type	Description
Boost libraries	Software libraries	Version 1.66 or higher
OpenCV	Software libraries	Version 3.3, for general image processing
CUDA	Software libraries	CUDA 10
FFmpeg	Software	Open source software encoder, mixing, and media streaming library

Hardware Requirements

Item	Minimum	Required
CPU	Intel Xeon E5-2697 or better	Intel Xeon E5-2697 or better
Memory	32 GB	32 GB
nVidia-GPU	DX11 compatible Equal or better to nVidia GeForce RTX 2070	DX11 compatible Equal or better to nVidia GeForce RTX 2070

Mixer and Encoder

The main purpose of the CONTENT4ALL Mixer and Encoder is to put together the main video content, provided by the Broadcaster, and the Photorealistic 3D virtual human, produced by the 3D Model Renderer, into a single video stream and then to adapt it for the delivery to HbbTV and mobile devices via broadband.

Going deeper into the two main logical functions separately, the mixing function will ingest two video sources coming via IP connections in a digital format that will be defined and agreed between the parties for each pilot application (typically, an MPEG-2 or MPEG-4 based one).

For Pilot 2 Demonstrator (Phase 3) the module should be able to deliver, beside the mixed stream already available for Pilot 1 Demonstrator (Phase 1), also a stream containing only the 3D virtual human properly packaged for HbbTV and mobile devices. This additional stream is needed to enable client-side mixing on devices that have this capability.

Software Requirements

Requirement	Type	Description
FFmpeg	Software	Open source software encoder, mixing, and media streaming library

Hardware Requirements

	Item	Specification	description
Software encoding	CPU	24-core Intel 2.4 GHz or above processors	*These estimated specs are for encoding of 2 HD, and 1 (512x424) video streams.* Software encoding with H.264 for high quality (CRF 20 configuration in FFmpeg) video streams require at least 8 CPU cores per each 1080p stream. Encoding, streaming, and video synchronization processes require at least 24 CPU cores for processing.
Software hardware accelerated encoding based	CPU	8-core Intel 2.4 GHz or above processors	The encoding is offloaded to the GPUs, hence the CPU is load is limited to frame synchronization, muxing and streaming
	GPU	Quadro P2000/Quadro M2000 or above	Hardware accelerated encoding with H.264 and H.265. Encoding of multiple HD and 4K streams.
Other hardware requirements	Memory	8-16 GB	Encoder buffers, and frame synchronization buffers
	Ethernet NIC	10G and 100 Mbps NIC cards	High quality streaming of muxed video streams require a minimum bandwidth of 75Mbps. The streaming of skeleton data requires a minimum bandwidth of 2Mbps.

Management Server

The CONTENT4ALL Management Server will be responsible for initializing and monitoring other components of the solution.

The Management Server will provide a UI for enabling users to configure the entire system and to monitor the status of the various components. Therefore, the first feature of the component will be to manage users and roles; it will allow creating new users, modifying details, including password, or deleting them. It will also allow assigning users with one or more roles.

The second feature is the ability to configure the various system components; the UI will allow defining and scheduling programs that will have a corresponding sign language translation (via Remote Studio or via automatic translation in Phase 3) and inserting the needed configuration parameters and it will be able to start the components with their relative configurations.

The third feature of the management server is the monitoring of the other solution components. It will collect info on the status of all the components and show them in a dashboard. In order to monitor the various components both at the infrastructure and at the application level. A monitoring agent needs to be installed in each component, and it will act as an application probe and send the application data to the management server.

Software Requirements

Requirement	Type	Description
Java SDK 1.8	Software	Java VM for Java Language
Telegraf	Software	Monitoring agent
InfluxDB	Software	Time series database (monitoring data store)
Grafana	Software	Monitoring dashboard
Node 6+	Software	Runtime Javascript
Angular 5	Software	Frontend development framework
MySQL	Database	Relational Database Management System

Hardware Requirements

Item	Minimum	Recommended
CPU	2 Cores	4 Cores
Memory	2 GB	4 GB
Storage	10 GB	50GB
Bandwidth	512 Kb/s	2 Mb/s

Users' TV – Set Top Box

For rendering the signed video stream onto HbbTV-enabled TV sets or Set top boxes, an HbbTV application is needed running on the device that allows the user to switch from the broadcast original video to the broadband signed one.

HbbTV applications are based on HTML and related standards (CSS, JS, etc.). They are signalled on the broadcast stream and can be launched automatically by the device or manually on user request. Applications usually have a UI showed on top of the broadcast video which allows users to interact with them. The apps can access TV specific features such as video resize, channel change, alternative audio streams, broadband video streaming and others via specific APIs defined by HbbTV standard.

The HbbTV app of CONTENT4ALL project should be able to retrieve from the Management Server component the info regarding currently available signed streams and, on users' request, stream from the broadband connection and render the signed video instead of the original broadcast content.

For Pilot 2 Demonstrator (Phase 3) the addition of client-side mixing feature will require an update of the application to be compliant to HbbTV 2.0.1 version and a device with two decoders having to decode concurrently the broadcast stream (main video content) and the broadband stream (3D virtual human).

Software Requirements

Requirement	Type	Description
HbbTV 2.0.1+	Software	TV Firmware

Hardware Requirements

A TV/STB running HbbTV v2.0.1 middleware and supporting dual decoding is needed to show application capabilities.

	Item	Specification	Description
TV/Set Top Box	HW	2 decoders	
	Firmware	HbbTV 2.0.1	

Users' Pc or Mobile Devices (WebApp)

Next to the regular TV channels, VRT also owns different online portals. The VRT Nu portal allows the user to livestream VRTs TV channels and catch-up via video on demand. This portal will be extended in order to receive and show the mixed video where the 3D Photorealistic virtual human is overlaid on the original video.

Software Requirements

Requirement	Type	Description
Browser	Software	A browser supporting HTML5, Javascript and MPEG-DASH
MediaElement player	Software	Online Video Player

Hardware Requirements

	Item	Specification	description
PC	Web	Browsers, all platforms	Subscription needed
Mobile	IOS	Safari	Subscription needed
Mobile	Android	Chrome	Subscription needed

3. Integration Plan

The CONTENT4ALL project requires a solid integration, evaluation and demonstrator planning which involves the concepts and technical components from WP2, WP3 and WP4. The objective of WP5 is to integrate the components defined in WP2 and developed during the WP3 and WP4 execution. During the Phase 3, the Pilot 2 Demonstrator will be set up in order to demonstrate the implemented functionalities. WP5's activities can be carried out in several phases (Figure 5):

- Planning
- Integration
  - Stub testing phase
  - Remote integration phase
  - Face-to-face integration
- Pilot 2 Demonstrator Setup
- Evaluation

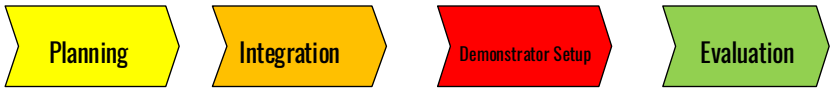

Figure 5 Pilot 2 Demonstrator Setup Phases

The Planning activity includes all the steps necessary for the realization of the Pilot 2 Demonstrator stage, describing the timing, the components that must be integrated, the evaluation methods. The result of this activity is the following deliverable that explains the plan for the integration of all the components in order to implement the Pilot 2 Demonstrator.

The Integration phase aims to combine individually tested software components into an integrated system. This phase can be divided further into three sub-phases: stub testing phase, remote integration phase and face-to-face integration phase.

The face-to-face integration phase can be performed during the Pilot 2 Demonstrator Setup, where partners meet each other at the designated places in order to set up the hardware and software components and to integrate each other to provide a fully integrated and working system.

The Evaluation phase aims to assess the fully integrated systems both functionally and in end-to-end manner. The Evaluation methodologies adopted for the end-to-end evaluation of the Pilot 2 Demonstrator are described on chapter 4.

3.1. Timeline

Figure 6 shows the timeline planned for the Pilot 2 Demonstrator Setup. Due to its proof-of-concept nature, there will be a continuous evaluation during the whole time. In the first 16 months (M18 to M33) all planning and development activities related to Pilot 2 Demonstrator are carried out. From the M23, in parallel, component integration activities will start. This phase will overlap with the development in order to fine-tune integration flows and eventually modify something where needed. The integration phase can be further divided into three phases: an initial phase of stub testing where each component developer will locally test his component; when the stub tests are completed it will begin the remote integration, where each component will start interacting with the others; the last integration phase is the face-to-face integration, where developer meet each other in order to complete the latest integration activities. The Demonstrator installation activities will start at M30 up to M35. During months M35 and M36 of the project, the Pilot 2 Demonstrator will run. The successful completion of all system component tests described in 3.5 means that Pilot 2 Demonstrator runs solidly.

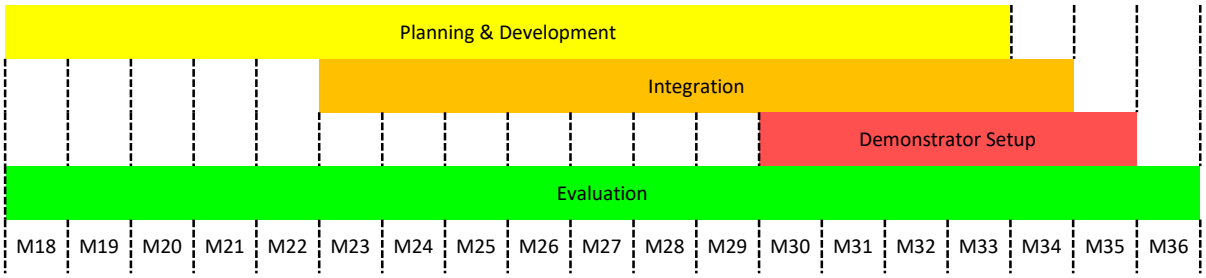

Figure 6 Timeline of Pilot 2 Demonstrator

The general plan for all WP5 activities related to the Pilot 2 Demonstrator is illustrated in Table 1.

Table 1 General time plan for integration, evaluation and demonstration activities for Pilot 2 Demonstrator

Phase	Date	Month
Stub testing phase	July 2019 – January 2020	M23 – M29
Remote testing phase	February 2020 – May 2020	M30 – M33
Face-to-face integration phase	June 2020	M34
Demo setup	February 2020 – July 2020	M30 – M35
Evaluation & corrective actions	February 2019 – August 2020	M18 – M36

3.2. Integration Strategies

The different components composing the CONTENT4ALL infrastructure will be integrated continuously and incrementally, when possible. The project's continuous integration approach is composed of a cycle of six steps as showed in Figure 7.

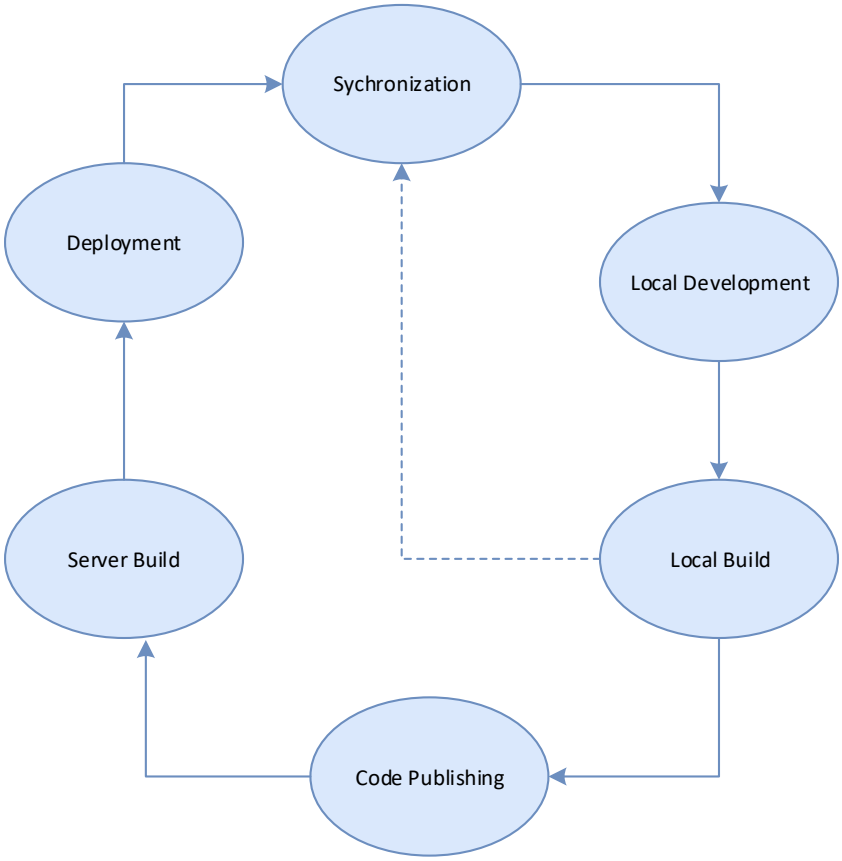

Figure 7 Continuous Integration Process

The next paragraphs describe more in detail each step of the cycle.

3.2.1. Synchronization

The first step of the continuous integration process is the synchronisation of the sources and related files on the local developer machine. The developer is checking out a working copy of the source code repository from the revision control system. Thus, the developer obtains a copy of the currently integrated source onto his local development machine. This local copy of the developer will then be linked to the remote revision control system.

3.2.2. Local Development

During this step, developers will develop the new features required and also define the test cases that their software will have to meet. These are unit tests that will verify the individual features. In case there are already local changes in a file which the developer wants to keep, but there are also changes in repository, the developer has to merge these changes.

3.2.3. Local Build

The developer who works on a software component and who changes the system makes every effort to ensure that the new feature does not introduce a bug into the overall system: once this has been done (and usually at various points while during the development) the developer triggers an automated build on his development machine. This takes the currently developed source code, compiles and links it into an executable program, and runs automated or manual tests. These tests may comprise functional, security, load and performance tests. Only if the entire code builds correctly and runs the tests without errors the overall build is considered to be valid. The unit can be functionally accepted.

3.2.4. Code Publishing

After the local build succeeded the developer can publish the local changes of the source code. This is done by committing the local sources to the central software revision system. In case there are already committed changes to the source code in the software revision system, the source code has to be synchronized again before it can be published.

3.2.5. Server Build

Whenever a new revision of a software unit is made available and committed to the repository, the automated continuous integration system is notified. The continuous integration server performs a checkout of the most recent version of the software system from the repository. The code is built on the integration server and the automated tests are executed.

With certain plugins the continuous integration server can also perform static or dynamic analysis checks that address the test coverage of the source code, possible duplication of code and dependencies among classes.

3.2.6. Deployment

Once the build completes correctly and there are no test failures, the artefacts can be deployed automatically, or manually installed to the designated instances.

3.3. Version Control Systems and Tracking Tool

Managing code for a big and collaborative project such as CONTENT4ALL, where many different components need to be developed or enhanced and integrated, has many challenges that need to be tackled. Within CONTENT4ALL it was decided that a hybrid integration approach will be followed, respecting individual component specifications as well as integrated platform requirements. According to this approach:

- Some components will reside in one same server.
- Some components will reside on distinct servers (e.g. in order to address tight performance requirements that require special hardware such as servers with GPUs).
- Some components will reside on distinct devices (e.g. in order to extend the use of the framework as much as possible)

A centralized management system will be created to keep track of every part of the code from all the related partners and use it to integrate all modules into the final prototype. Another requirement is bug tracking and version control of the code so that we can identify any problems that may arise during development and testing, and update our system code when needed. It is common practice to immediately commit every change to a revision control system, no matter how small it is. The rationale behind is that other developers should always work with the latest version of the code base. For source code management a Git repository will be set up during the Planning phase and before the beginning of the Integration phase.

Git is a version control system that is used for software development and other version control tasks. As a distributed revision control system, it is aimed at speed, data integrity, and support for distributed, non-linear workflows. As with most other distributed version control systems, and unlike most client–server systems, every Git directory on every computer is a full-fledged repository with complete history and full version-tracking capabilities, independent of network access or a central server. It is also free software distributed under the terms of the GNU General Public License.

Git has many desirable features such as:

- Distributed development, with each developer working on a local copy
- Non-linear development, with branching and merging
- Cryptographic authentication of history, change or deletion of a commit is marked
- Pluggable merge strategies, merge auto-completion model is well defined and in case of a conflict the developer can manually merge the files

If necessary, the development tracking tool will be integrated with a planning and tracking tool (e.g. JIRA, Trello) to monitor the status of the development and have a visual interface where is possible to track issues and timeline for each implemented feature.

It is important to note that a revision repository should contain all files necessary to build the software system but should not contain any files that can be derived from other files in an automated fashion. Such redundancy only pollutes the repository and is considered bad practice (Richardson, J. & Gwaltney, W.A., 2005).

3.4. System components for integration, evaluation and demonstration

Table 2 provides the system components detailed in D2.3 which are referenced throughout the document.

Table 2 System components for integration, evaluation and demonstration

C.ID	Component name	Comp. ID	Sub-Component Name	Resp. Partners
MG	Media Gateway	MG.SE	Subtitles Extractor	UNIS
MR	Media Receiver	MR.SF	Subtitles Forwarder	UNIS
HA	3D Avatar Language Translator	LT.SUGL	Subtitles to Sign Glosses Converter	UNIS
		LT.GLSK	Sign Glosses to Skeletal Poses Converter	UNIS
PMR	Photorealistic 3D Model Renderer	PMR.HAE	Hand Animation Engine	HHI
		PMR.BAE	Body Animation Engine	HHI
		PMR.FAE	Facial Animation Engine	HHI
		PMR.AR	Avatar Renderer	HHI
		PMR.INTCPIF	Input TCP Interface	HHI
		PMR.OUTTCPIF	Output TCP Interface	HHI
ME	Mixer and Encoder	ME.SBD	Synchro Buffer + Decoder	FIN
		ME.M	Mixer	FIN
		ME.LE	Live Encoder	FIN
		ME.MA	Management Agent	FIN
		ME.INIF10G	Input 10Gps Interface	FIN
		ME.INIF100M	Input 100Mps Interface	FIN
		ME.OUTIF100M	Output 100Mps Interface	FIN
MS	Management Server	MS.C	Configuration	FIN
		MS.UI	User Interface	FIN
		MS.M	Monitoring	FIN
		MS.UM	User Management	FIN
		MS.CD	Command Dispatcher	FIN

		MS.MG	Monitoring Gateway	FIN
BC	Broadcaster	BC.DVBP	DVB Playout	
		BC.E	Encoder	
		BC.OS	Origin Server	
UT	User Terminals	UT.HBBTV	HbbTV APP	
		UT.WAPP	Web Application	

3.5. Software Integration Plan and Component Evaluation Metrics

System components, which are developed under WP3, WP4 and WP5, are evaluated independently as described in the following paragraphs. The assessment of the end-to-end Pilot 2 Demonstrator will be described on Chapter 4.

The evaluation is aligned with the integration timeline described in section 3.1 and performed under the Reference Scenario described in 2.2.1. The aspects presented below will be assessed.

- Evaluation of the delivery of data exchange messages

  Successful arrival of data exchange messages at each system component is assessed. The data exchange message should arrive at the correct system component and the response status/acknowledgement/content should be returned to the invoked component.

- Evaluation of the delivery of video streams

  Reception of Main Broadcast stream by the Processing & Rendering Unit is assessed

- Evaluation of the delivery of mixed (broadcaster video with virtual human overlay) video stream

  Reception of mixed video stream from the Processing & Rendering Unit to the User Terminals is assessed

3.5.1. Components integration types

CONTENT4ALL integration activities can be categorized mainly in three types. Each integration action will be carried out according to the specified types, which are explained in detail in following subsections.

1. *Data exchange messages*

The CONTENT4ALL system requires various data messages that need to be exchanged. The data interchange format for all data exchange messages will be JSON, which is a text format that is completely language independent but uses conventions that are familiar to programmers of the most used programming languages. These properties make JSON an ideal data-interchange language. JSON is built on a collection of name/value pairs and an ordered list of values. JSON data will be transmitted over TCP stream sockets. For this purpose, socket addresses and port numbers will be defined and shared among partners.

2. *Media delivery*

The delivery of media content to the different components is defined as a media delivery. In this context, two types of media exchanges are foreseen in the CONTENT4ALL demonstrator. These are briefly described below together with the format of the exchange.

a) Raw domain exchanges: Delivery of decompressed visual data for content dependent processing activities. Example: Delivery of the video content from the Photorealistic 3D Model Renderer to the Mixer and Encoder component.
Format: Framed RAW information (e.g., YUV) over TCP/IP transmissions.

b) Compressed domain exchanges: Delivery of compressed visual data for content independent processing activities. Example: Delivery of the video content from the Broadcaster Origin Server to the HbbTV Application. Format: H.264/AVC, Mpeg-TS over TCP/IP transmissions.

In the Pilot 2 Demonstrator, the video stream needs to be transmitted at a certain transmission rate between the Processing & Rendering Unit and the User Terminals. For the transmission of the multimedia stream we plan to use one of the following transport protocols: UDP and TCP. Both protocols have positive and negative aspects that it is worth to analyze. The User Datagram Protocol (UDP) is a simple and efficient transmission mechanism, which sends the media stream as a series of small packets. However, there is no mechanism within the protocol to guarantee delivery. It is up to the receiving application to detect loss or corruption and recover data using error correction techniques. If data is lost, the stream may suffer a dropout. Reliable protocols, such as the Transmission Control Protocol (TCP), guarantee correct delivery of each bit in the media stream. However, they accomplish this with a system of timeouts and retries, which makes them more complex to implement. It also means that when there is data loss on the network, the media stream stalls while the protocol handlers detect the loss and retransmit the missing data. In the CONTENT4ALL demonstrator, due to susceptibility of media components to packet loss, TCP/IP will be employed first, which is the fall back solution. In parallel, UDP/IP is being implemented which will provide the flexibility to switch in between if there will be too much media stream stalls and performance issues.

3.5.2. Intra-component integration

This section describes the integration actions required among the sub-components within a particular component and the evaluation plan for verifying a successful integration.

Media Gateway

The CONTENT4ALL Media Gateway consists of one sub-component. The Media Gateway architecture and functional description are described in detail in Deliverable *D2.3 – Final Reference System Architecture*.

Due to the fact that the Media Receiver is composed of only one sub-component dedicated to extract subtitles, it does not need integration actions.

Media Receiver

The CONTENT4ALL Media Receiver consists of one sub-component. The Media Receiver architecture and functional description are described in detail in Deliverable *D2.3 – Final Reference System Architecture*.

Due to the fact that the Media Receiver is composed of only one sub-component dedicated to forward subtitles, it does not need integration actions.

3D Avatar Language Translator

The CONTENT4ALL 3D Avatar Language Translator consists of 2 sub-components:

- Subtitles to sign gloss translation
- Sign gloss to animation parameters

The following section briefly describes the integration between individual sub-components of the language translator.

The incoming time aligned subtitle stream is first converted to a sign gloss representation using neural machine translation techniques. The sign gloss is then converted to skeletal poses which are passed to the 3D virtual human as animation parameters.

Integration Plan

The information exchanged between sub-components for each integration action, including input and output parameters, is described in the Table 3.

Table 3 Integration actions for 3D Avatar Language Translator

Action #	From	To	Expected input/output	Integ. type	Responsible partner
LT-IA1	Subtitles	Subtitles to Sign Glosses Converter	Input: Raw transcription text data (subtitles) Output: Sequence of Sign Glosses	1	Input: UNIS Output: UNIS
LT-IA2	Subtitles to Sign Glosses Converter	Sign Glosses to Skeletal Poses Converter	Input: Sequence of Sign Glosses Output: JSON stream of Avatar Animation Parameters	1	Input: UNIS Output: UNIS

Evaluation Plan

The Evaluation Plan for testing the success or failure of the integration actions described in Table 3 is listed below.

- Subtitles to Sign Glosses test: This test verifies that the sign gloss will reflect ground truth annotation and timing information. This test will evaluate the success or failure of the integrated action #LT-IA 1.
- Sign Glosses Sequence to Skeletal Poses test: This test verifies that the Skeletal Poses will reflect sign glosses. This test will evaluate the success or failure of the integrated action #LT-IA 2.

Photorealistic 3D Model Renderer

The CONTENT4ALL Photorealistic 3D Model Renderer consists of six sub-components. The Photorealistic 3D Model Renderer architecture and functional description are described in detail in Deliverable *D2.3 – Final Reference System Architecture*. The following section briefly describes the integration between individual sub-components of the Photorealistic 3D Model Renderer with respect to the required input and expected output for each integration action.

Integration Plan

The information exchanged between sub-components for each integration action, including input and output parameters, is described in the Table 4.

Table 4 Integration actions for Photorealistic 3D Model Renderer

Action #	From	To	Expected input/output	Integ. type	Responsible partner
PMR-IA1	Facial Animation Engine	Avatar Renderer	Input: FACS weights, visemes Output: Synthesized face texture	2a	Input: HHI Output: HHI
PMR-IA2	Body Animation Engine	Avatar Renderer	Input: OpenPose skeleton parameters Output: skeleton joint angles	1	Input: HHI Output: HHI
PMR-IA3	Hand Animation Engine	Avatar Renderer	Input: Handshape label stream or hand skeleton parameters Output: Joint angles	1	Input: HHI Output: HHI
PMR-IA4	Avatar Renderer	TCP Socket Interface	Input: Face Texture, Avatar Skeleton, Joint angles Output: Avatar Image	1/2a	Input: UNIS Output: UNIS

Evaluation Plan

The Evaluation Plan for testing the success or failure of the integration actions described in Table 4 are listed below.

- Face texture synthesis test: This test verifies that the face texture is correctly synthesized according to the given animation parameters. This can be tested by generating the animation parameters from real data and comparing the displayed facial expression.
  This test will evaluate the success or failure of the integrated action #PMR-IA1.
- Avatar skeleton matching test: This test verifies that the virtual human skeleton parameters are extracted correctly from the input skeleton data streams.
  This test will evaluate the success or failure of the integrated action #PMR-IA2.
- Hand skeleton extraction test: This test verifies that the hand joint parameters are correctly extracted from the handshape label / hand skeleton data.
  This test will evaluate the success or failure of the integrated action #PMR-IA3.
- Avatar frame generation test: This test verifies that the virtual human image is correctly rendered starting from the face texture, virtual human skeleton and joint angles streams
  This test will evaluate the success or failure of the integrated action #PMR-IA4.

Mixer and Encoder

The CONTENT4ALL Mixer and Encoder consist of seven sub-components. The Mixer and Encoder architecture and functional description are described in detail in Deliverable *D2.3 – Final Reference System Architecture*. The following section briefly describes the integration between individual sub-components of the Mixer and Encoder with respect to the required input and expected output for each integration action.

Integration Plan

The information exchanged between sub-components for each integration action, including input and output parameters, is described in the Table 5.

Table 5 Integration actions for Mixer and Encoder

Action #	From	To	Expected input/output	Integ. type	Responsible partner
ME-IA1	Mixer	Live Encoder	Input: Broadcaster main video stream and Raw Avatar Image Stream Output: Mixed video stream 3D Avatar stream	2a	Input: FIN Output: FIN
ME-IA2	Live Encoder	Network interface	Input: Mixed video stream Output: Compressed mixed video stream Compressed 3D Avatar video stream	2a	Input: FIN Output: FIN

Evaluation Plan

The Evaluation Plan for testing the success or failure of the integration actions described in Table 5 are listed below.

- Mixing video test: This test verifies that the Mixer component mix in one stream two different sources: the main broadcaster video stream and the virtual human stream.
  This test will evaluate the success or failure of the integrated action #ME-IA1.
- Live Encoding test: This test verifies that the mixed video stream is compressed in the different profiles with increasing quality (resolution and bitrate).
  This test will evaluate the success or failure of the integrated action #ME-IA2.

Management Server

The CONTENT4ALL Management Server consists of six sub-components. The Management Server architecture and functional description are described in detail in Deliverable *D2.3– Final Reference System Architecture*. The following section briefly describes the integration between individual sub-components of the Management Server with respect to the required input and expected output for each integration action.

Integration Plan

The information exchanged between sub-components for each integration action, including input and output parameters, is described in the Table 6.

Table 6 Integration actions for Management Server

Action #	From	To	Expected input/output	Integ. type	Responsible partner
MS-IA1	Monitoring Gateway	Monitoring	Input: JSON raw stream of monitoring data Output: -	1	Input: FIN Output: FIN
MS-IA2	Configuration	Command Dispatcher	Input: JSON raw stream of command and Component ID Output: SSH encrypted component execution command	1	Input: FIN Output: FIN

Evaluation Plan

The Evaluation Plan for testing the success or failure of the integration actions described in Table 6 are listed below.

- Monitoring stream receiver test: This test verifies that the Monitoring Gateway component is able to receive a raw JSON stream of data.
  This test will evaluate the success or failure of the integrated action #MS-IA1.
- Command dispatcher test: This test verifies that the Command Dispatcher correctly receives in input a command in the form of a JSON object and forward this command to the right component.
  This test will evaluate the success or failure of the integrated action #MS-IA2.

3.5.3. Inter-component integration

This section describes the integration actions required between the different components of the CONTENT4ALL Pilot 1 Demonstrator and the evaluation plans for verifying a successful integration.

Component X- Management Server

A particular integration action that is common among the different components that make up the architecture is the one between each component and the CONTENT4ALL Management Server. Since the Management Server is the component in charge of the management of the overall system configuration, each component should be handled and configured via the Management Server.

Integration Plan

The following action described in Table 7 is the general integration that is requested for each component and that requires evaluation.

Table 7 General integration action between Component X and Management Server

Action #	From	To	Expected input/output	Integ. type	Responsible partner
CX-MS-IA1	Management Server	Component X	Input: Configuration Parameters Output: -	1	Input: FIN Output: Component Owner

Evaluation Plan

The Evaluation Plan for testing the success or failure of the integration actions described in Table 7 are listed below.

- Component Initialization Parameter test: This test verifies that the configuration parameters are correctly sent to the corresponding component.
  This test will evaluate the success or failure of the integrated action #CX-MS-IA1.

Broadcaster – Media Gateway

There is one integration activity that exists between the Broadcaster and the Media Gateway. The integration action below briefly describes the integration between individual components with respect to the required input and expected output for each integration action.

Integration Plan

Table 8 describes the Inter-component integration actions between the Broadcaster and the Media Gateway.

Table 8 Inter-component integration actions between the Broadcaster and the Mixer and Encoder

Action #	From	To	Expected input/output	Integ. type	Responsible partner
BC-MG-IA1	Broadcaster	Media Gateway	Input: Main Broadcast Stream Output: -	1	Input: STXT/VRT Output: UNIS

Evaluation Plan

The Evaluation Plan for testing the success or failure of the integration actions described in Table 8 are listed below.

- Broadcaster main stream transmission test: This test verifies that the main stream produced by the Broadcaster is correctly received by the Media Gateway.
  This test will evaluate the success or failure of the integration action # BC-MG-IA1.

Media Gateway-Media Receiver

There is one integration activity that exist between the Media Gateway and the Media Receiver. The integration action below briefly describes the integration between individual components with respect to the required input and expected output for each integration action.

Integration Plan

Table 9 describes the Inter-component integration actions between the Media Gateway and the Media Receiver.

Table 9 Inter-component integration actions between the Media Gateway and the Media Receiver

Action #	From	To	Expected input/output	Integ. type	Responsible partner
MG-MR-IA1	Media Gateway	Media Receiver	Input: Raw transcription text data (subtitles) Output: -	1	Input: UNIS Output: UNIS

Evaluation Plan

The Evaluation Plan for testing the success or failure of the integration actions described in Table 9 are listed below.

- Raw transcription text data (subtitles) stream transmission test: This test verifies that the Raw transcription text data (subtitles) stream produced by the Media Gateway is correctly received by the Media Receiver.
  This test will evaluate the success or failure of the integration action # MG-MR-IA1.

Media Receiver – 3D Avatar Language Translator

There is one integration activity that exist between the Media Receiver and the 3D Avatar Language Translator. The integration action below briefly describes the integration between individual components with respect to the required input and expected output for each integration action.

Integration Plan

Table 10 describes the Inter-component integration actions between the Media Receiver and the 3D Avatar Language Translator.

Table 10 Inter-component integration actions between the Media Receiver and the 3D Avatar Language Translator

Action #	From	To	Expected input/output	Integ. type	Responsible partner
MR-LT-IA1	Media Receiver	3D Avatar Language Translator	Input: Raw transcription text data (subtitles) Output: -	1	Input: UNIS Output: UNIS

Evaluation Plan

The Evaluation Plan for testing the success or failure of the integration actions described in Table 10 are listed below.

- Raw transcription text data (subtitles) stream transmission test: This test verifies that the Raw transcription text data (subtitles) stream produced by the Media Receiver is correctly received by the 3D Avatar Language Translator. This test will evaluate the success or failure of the integration action # MR-LT-IA1.

3D Avatar Language Translator – Photorealistic 3D Model Renderer

There is one integration activity that exists between the 3D Avatar Language Translator and the Photorealistic 3D Model Renderer. The integration action below briefly describes the integration between individual components with respect to the required input and expected output for each integration action.

Integration Plan

Table 11 describes the Inter-component integration actions between the 3D Avatar Language Translator and the Photorealistic 3D Model Renderer.

Table 11 Inter-component integration actions between the 3D Avatar Language Translator and the Photorealistic 3D Model Renderer

Action #	From	To	Expected input/output	Integ. type	Responsible partner
LT-PMR-IA1	3D Avatar Language Translator	Photorealistic 3D Model Renderer	Input: JSON stream of Avatar Animation Parameters (Handshape-Id, Skeleton, FACS, Viseme-Id, Eyegaze) Output: -	1	Input: UNIS Output: HHI

Evaluation Plan

The Evaluation Plan for testing the success or failure of the integration actions described in Table 11 are listed below.

- JSON byte stream transmission test: This test verifies that the JSON byte stream produced by the Avatar Language Translator representing the handshape, body pose and facial expression is correctly received by the Photorealistic 3D Model Renderer. This test will evaluate the success or failure of the integration action # LT-PMR-IA1.

Photorealistic 3D Model Renderer – Mixer and Encoder

There is one integration activity that exists between the Photorealistic 3D Model Renderer and the Mixer and Encoder. The integration action below briefly describes the integration between individual components with respect to the required input and expected output for each integration action.

Integration Plan

Table 12 describes the Inter-component integration actions between the Photorealistic 3D Model Renderer and the Mixer and Encoder.

Table 12 Inter-component integration actions between the Photorealistic 3D Model Renderer and the Mixer and Encoder

Action #	From	To	Expected input/output	Integ. type	Responsible partner
PMR-ME-IA1	Photorealistic 3D Model Renderer	Mixer and Encoder	Input: 24-bit Raw RGB image data stream Output: -	2a	Input: HHI Output: FIN

Evaluation Plan

The Evaluation Plan for testing the success or failure of the integration actions described in Table 12 are listed below.

- Avatar Raw image stream transmission test: This test verifies that the virtual human raw image stream produced by the Photorealistic 3D Model Renderer is correctly received by the Mixer and Encoder.
  This test will evaluate the success or failure of the integration action # PMR-ME-IA1.

Broadcaster – Mixer and Encoder

There is one integration activity that exists between the Broadcaster and the Mixer and Encoder. The integration action below briefly describes the integration between individual components with respect to the required input and expected output for each integration action.

Integration Plan

Table 13 describes the Inter-component integration actions between the Broadcaster and the Mixer and Encoder.

Table 13 Inter-component integration actions between the Broadcaster and the Mixer and Encoder

Action #	From	To	Expected input/output	Integ. type	Responsible partner
BC-ME-IA1	Broadcaster	Mixer and Encoder	Input: Uncompressed main video stream Output: -	2a	Input: STXT/VRT Output: FIN

Evaluation Plan

The Evaluation Plan for testing the success or failure of the integration actions described in Table 13 are listed below.

- Broadcaster main video stream transmission test: This test verifies that the main video stream produced by the Broadcaster is correctly received by the Media and Encoder.
  This test will evaluate the success or failure of the integration action # BC-ME-IA1.

Mixer and Encoder – Broadcaster Origin Server

There are two integration activity that exist between the Mixer and Encoder and the Broadcaster Origin Server. The integration action below briefly describes the integration between individual components with respect to the required input and expected output for each integration action.

Integration Plan

Table 14 describes the Inter-component integration actions between the Mixer and Encoder and the Broadcaster Origin Server.

Table 14 Inter-component integration actions between the Mixer and Encoder and the Broadcaster Origin Server

Action #	From	To	Expected input/output	Integ. type	Responsible partner
ME-BCOS-IA1	Mixer and Encoder	Broadcaster Origin Server	Input: Mixed compressed video stream Output: -	2b	Input: FIN Output: STXT/VRT
ME-BCOS-IA2	Mixer and Encoder	Broadcaster Origin Server	Input: 3D Avatar compressed video stream Output: -	2b	Input: FIN Output: STXT/VRT

Evaluation Plan

The Evaluation Plan for testing the success or failure of the integration actions described in Table 14 are listed below.

- Mixed video stream transmission test: This test verifies that the mixed video stream produced by the Mixer and Encoder is correctly received by the Broadcaster Origin server.
  This test will evaluate the success or failure of the integration action # ME-BCOS-IA1.
- 3D Avatar video stream transmission test: This test verifies that the 3D Avatar video stream produced by the Mixer and Encoder is correctly received by the Broadcaster Origin server.
  This test will evaluate the success or failure of the integration action # ME-BCOS-IA2.

Broadcaster Origin Server – HbbTV App

There are two integration activities that exists between the Broadcaster Origin Server and the HbbTV Application running on TVs or Set Top Box. The integration action below briefly describes the integration between individual components with respect to the required input and expected output for each integration action.

Integration Plan

Table 15 describes the Inter-component integration actions between the Broadcaster Origin Server and the HbbTV Application.

Table 15 Inter-component integration actions between the Mixer and Encoder and the Broadcaster Origin Server

Action #	From	To	Expected input/output	Integ. type	Responsible partner
BCOS-HBBTV-IA1	Broadcaster Origin Server	HbbTV Application	Input: Mixed compressed video stream Output: -	2b	Input: STXT Output: STXT
BCOS-HBBTV-IA2	Broadcaster Origin Server	HbbTV Application	Input: 3D Avatar compressed video stream Output: -	2b	Input: STXT Output: STXT

Evaluation Plan

The Evaluation Plan for testing the success or failure of the integration actions described in Table 15 are listed below.

- Mixed video stream transmission test: This test verifies that the mixed video stream coming from the Broadcaster Origin Server is correctly received by the HbbTV Application running on TV or Set-Top-Box.
  This test will evaluate the success or failure of the integration action # BCOS-HBBTV-IA1.
- 3D Avatar video stream transmission test: This test verifies that the 3D Avatar video stream coming from the Broadcaster Origin Server is correctly received by the HbbTV Application running on TV or Set-Top-Box.
  This test will evaluate the success or failure of the integration action # BCOS-HBBTV-IA2.

Broadcaster Origin Server – PC's Web Application

There is one integration activity that exists between the Broadcaster Origin Server and the Web Application running on Browsers on PCs or Mobile Devices. The integration action below briefly describes the integration between individual components with respect to the required input and expected output for each integration action.

Integration Plan

Table 16 describes the Inter-component integration actions between the Broadcaster Origin Server and the Web Application.

Table 16 Inter-component integration actions between the Mixer and Encoder and the Broadcaster Origin Server

Action #	From	To	Expected input/output	Integ. type	Responsible partner
BCOS-WAPP-IA1	Broadcaster Origin Server	Web Application	Input: Mixed compressed video stream Output: -	2b	Input: VRT Output: VRT

Evaluation Plan

The Evaluation Plan for testing the success or failure of the integration actions described in Table 16 are listed below.

- Mixed video stream transmission test: This test verifies that the mixed video stream coming from the Broadcaster Origin Server is correctly received by the Web Application running on Browser's PCs or Browser's Mobile Devices. This test will evaluate the success or failure of the integration action # BCOS-WAPP-IA1.

4. Evaluation Methodologies and Metrics

All user-related activities within CONTENT4ALL follow a user-centred design approach as represented in

Figure 8. Following this approach, user research has started with analysing the requirements of users and other stakeholders. Therefore, an initial user workshop, user interviews and an online questionnaire inquiry with deaf and hearing signers were conducted in 2017 and 2018. The outcomes have formed the basis for initial user requirements (D2.2), which were submitted to a three-stage inquiry (Delphi technique) within the CONTENT4ALL consortium and aggregated to final user requirements (D2.4).

There are two main approaches for the evaluation process: first, to support the decision-making during the development stages, by comparing the effects of different choices regarding components like the parts of GUI or the virtual signer on the UX of the system, i.e. formative testing in the phases "Iterative implementation" and "UX-Testing". The other goal is to evaluate the Pilot 2 Demonstrator (Phase 3) as a whole, i.e. summative testing, which constitutes an overall concept validation and addresses the phases "UX-Testing" and "Prototypes".

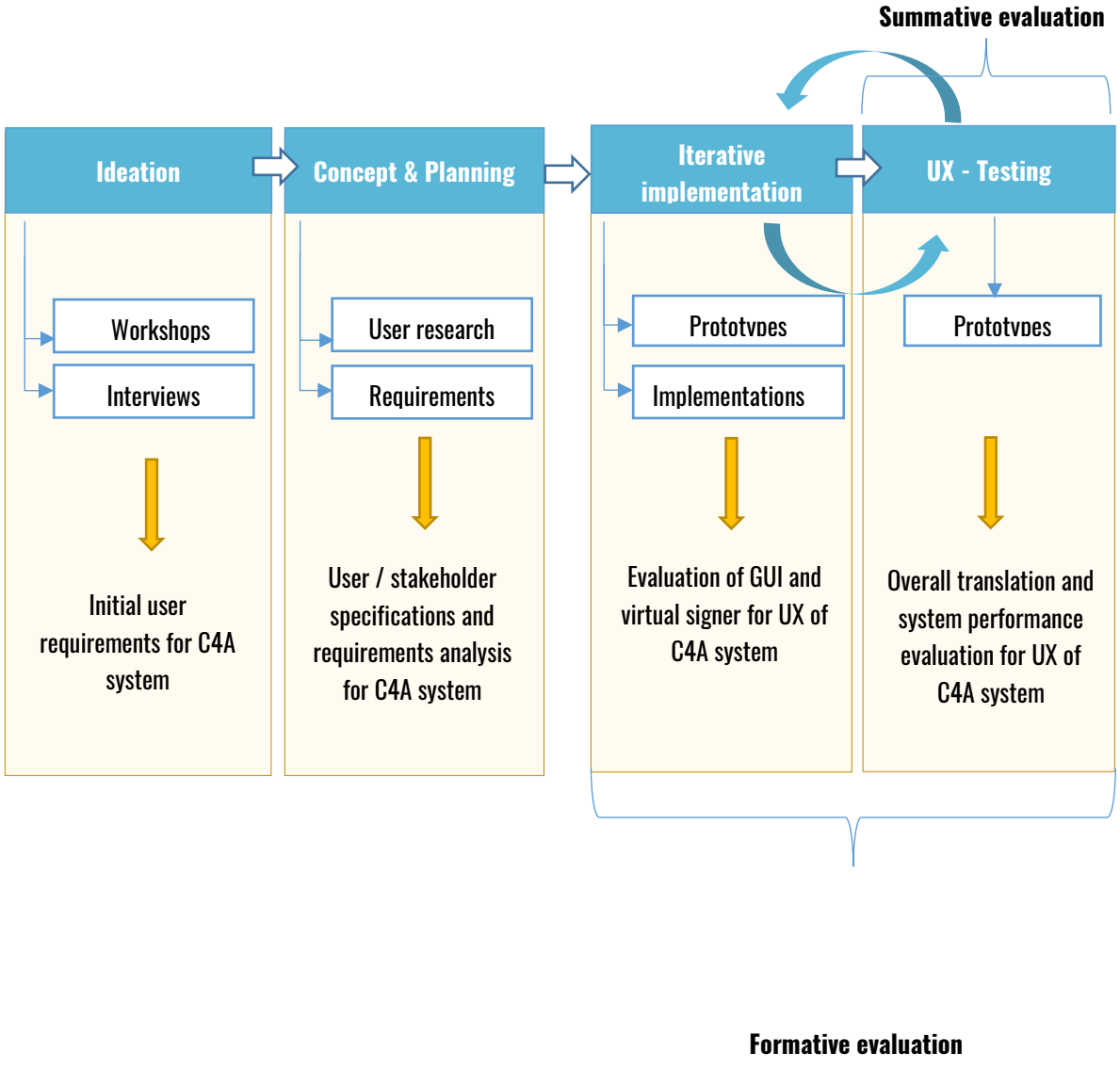

Figure 8 User-centred design process (adapted from Moser (2012), p. 14 & 15)

4.1.1. User Test Plan for System Evaluation

4.1.1.1. Methodological overview

An encyclopaedia of methods was developed with the aim of planning the user tests (both formative and summative testing) in WP5. A wide range of evaluation tools, methods and benchmarks that cover aspects of user experience, usability, quality of experience and comparable were collected, categorized and assessed regarding their suitability in the context of CONTENT4ALL user tests. These include methods such as questionnaires, physiological measurements, expert evaluations and various others.

The encyclopaedia of methods has different categories, according to which the methods are classified. These include classification into the type of method, the study type, time of evaluation and type of collected data as well as the methods' strengths and weaknesses, and their costs.

The categories are described in more detail in the following section. With regards to these categories all of the methods within the encyclopaedia were evaluated within the HFC research group in terms of their relevancy for CONTENT4ALL demonstrator evaluations. The selected methods are listed at the end of the document. As an example, for the categorization of methods, Table 17 shows how the methods can be divided according to the type of collected data into quantitative and qualitative methods. In the following sections the categorization will be demonstrated on the example of the method of Thinking Aloud.

4.1.1.2. Selected categories

Each method in the encyclopaedia of methods is categorized based on the following categories: UX, type of data, study type, collection method, and time of evaluation. Additionally, a description and the costs of the method are provided.

QoS-UX-QoE Framework: Assessing User Responses

The terms Quality of Experience, Quality of Service, Usability and User Experience are often used to describe the experiences of users before, during or after the interaction with products. Since these terms are often not uniformly defined and classified, a classification will be presented below according to which the encyclopaedia of methods is based. The model described below was developed in the ACTION-TV project (Oehme, Horn, Wieser, Waizenegger, & Fernando, 2016) and describes a user-oriented model, which includes the three variables environmental influences (Quality of Service), cognitive perception (User Experience) and effects on appraisal and behaviour (Quality of Experience). These variables were introduced by Wu et al. (2009) and Pallot et al. (2013) in the context of DIME research (Distributed Interactive Media Environment). In this model, the quality of a product or service (technical features, workmanship, performance, etc.) influences the cognitive perception of the user, i.e. the user experience. Depending on which aspect of perception is addressed, e.g. perceived usefulness, perceived aesthetics, etc., the degree of impact on the user's behaviour and assessment can be observed, i.e., the degree of satisfaction, anger or pleasure. The relationship is illustrated in the following graphic (Figure 9).

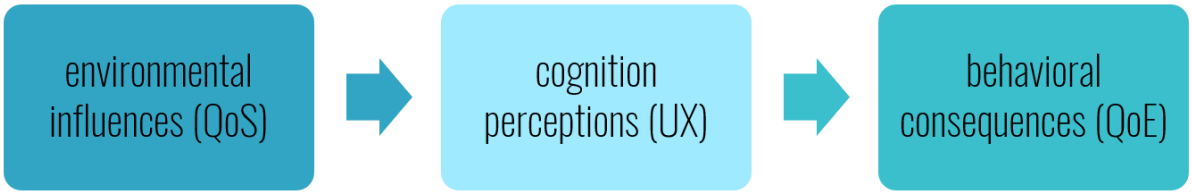

Figure 9. QoS-UX-QoE relationship (adapted from Wu et al., 2009; Pallot et. al, 2013)

Quality of Service (QoS) relates to measures of technical characteristics that affect service performance. It can be divided into QoS of network, system and application as well as content. QoS serves as an independent variable, i.e. it is measured only to provide a description of the objective features of a system or service.

The link to human perception is provided by selecting an appropriate research construct. With regard to UX this could emerge from pragmatic, hedonic, and social aspects. According to the 2010 ISO 9241-210, User Experience describes perception and response of a person resulting from an actual or expected usage of a product, system or service. Hassenzahl and Tractinsky (2006, p. 95) go a bit further in their definition and describe UX as "consequence of a user's internal state (predispositions, expectations, needs, motivation, mood, etc.), the characteristics of the designed system (e.g. complexity, purpose, usability, functionality, etc.) and the context (or the environment) within which the interaction occurs (e.g. organizational/social setting, meaningfulness of the activity, voluntariness of use, etc.)".

Within this interaction, pragmatic and hedonic qualities impact the user's experience. The division into pragmatic and hedonic is similar to the division into instrumental and non-instrumental qualities described by Thüring and Mahlke (2007). In DIME, these experiences are strongly linked to interaction with other persons, which is why the category social qualities was added following Pallot et. al, 2013.

In contrast to UX, which represents the perception and appraisal, QoE is the consequence of this perception and appraisal, i.e. the "perceivable" answer of the user. Diepold (2012, p. 7) describes it as "the degree of delight or annoyance of the user of an application or service. It results from the fulfillment of his or her expectations with respect to the utility and/or enjoyment of the application or service in the light of the user's personality and current state." It can be said that it describes "user perceptions of the performance of a service including

- how a user perceives the usability of a service when in use -
- how satisfied he or she is with a service, and
- the overall acceptability of an application or service, as perceived subjectively by the end user" (Brooks, France, & Hestnes, 2010).

An overview of the relationship between all influencing variables is depicted in Figure 10.

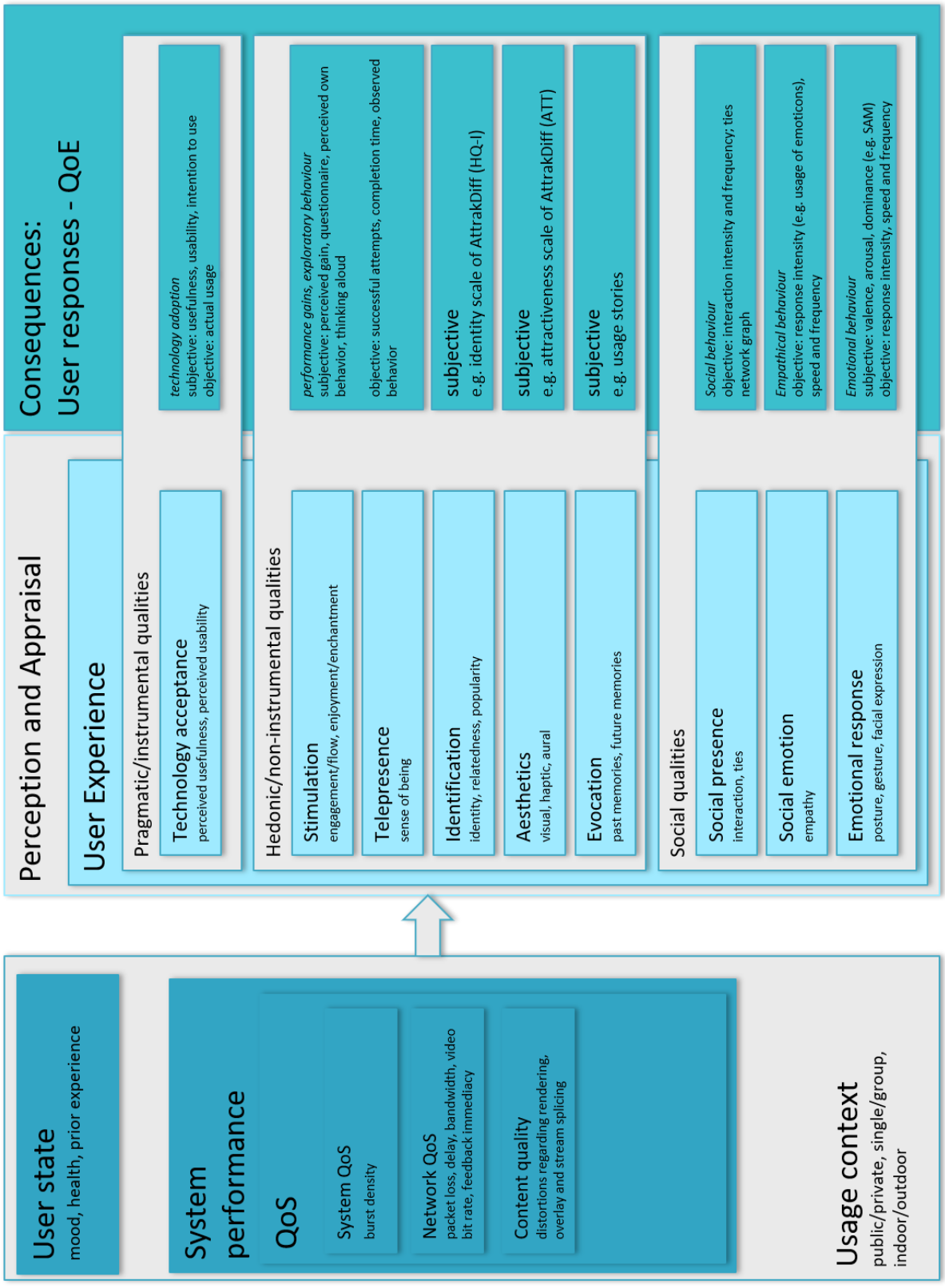

Figure 10. QoS-UX-QoE descriptive model (elaborated based on Oehme et al., 2016)

As already mentioned, UX can be described as perception and appraisal, and can be roughly divided into pragmatic/instrumental qualities, hedonic/non-instrumental and social qualities. Pragmatic/instrumental qualities refer to the technological acceptance of the product/service. This includes perceived usefulness and usability. The hedonic/non-instrumental qualities include the look and feel of a product/service. The user's feeling of so called "be-goals" (e.g. "being competent", "being related to others", "being special") and general needs (e.g. for novelty and change, personal growth, self-expression and/or relatedness) are addressed by these qualities (Hassenzahl, 2008, p. 2). Social qualities represent the user's social experience such as emotions that are elicited by the social aspects of the product/service or the context in which it is used. Responses to social qualities manifest in form of interaction, posture, gesture or facial expression. Social qualities are important in the context of interactive media and thus might not play a prominent role for CONTENT4ALL evaluation.

The three constituents result in the user's overall appraisal of the system and thus influence future decisions and behaviour, like the decisions to use the system on a more or less regular basis, if at all, intentions to migrate to another system with potentially similar capabilities, etc. The methods in the encyclopaedia are divided into the three UX constituents, i.e. as addressing either pragmatic, hedonic or social quality (or more than one of these aspects). An example of this classification is illustrated below in Figure 11.

User Experience (UX)		
Pragmatic qualities	Hedonic qualities	Social qualities

Figure 11. UX classification example for Thinking Aloud

Type of data collected

Depending on the study design and method type, data that is collected could be either quantitative or qualitative data. Quantitative data are everything that can be quantified or expressed as a number. For example, quantitative data are scores on tests, number of hours, weight of a subject etc. Such data is usually denoted by ordinal, interval or ratio scales, thus allowing the collected data to be statistically analyzed.

Qualitative data on the other hand cannot be expressed as a number. Rather it is a method which "(in general) generate words" (McCusker & Gunaydin, 2015, p. 1). Qualitative methods intent "to answer questions about the "what" or "why" of a phenomenon rather than "how many" or "how much", which are answered by quantitative methods" (McCusker & Gunaydin, 2015, p. 1). So qualitative methods refer to data that could provide insights and understanding about a problem. An example would be interviews with the use of open questions. However, as qualitative data is descriptive in nature, it is often difficult to analyze. Researchers make use of techniques such as content analysis, grounded theory (Glaser, Strauss & Strutzel 1967), or thematic analysis (Braun & Clarke, 2006). This kind of analysis is usually very time-consuming.

In Table 17, the categorization of methods according to the type of collected data into quantitative and qualitative methods is shown. For example, data collected using Thinking Aloud is qualitative in nature.

Table 17. Type of collected data classification example for all methods

Quantitative	**Qualitative**
Electrodermal activity	3E (Expressing, Experience & Emotion)
Heart rate (variability)	AXE (Anticipated eXperience Evaluation)
Secretory Immunoglobin A	Thinking Aloud
EMG (face muscles)	Experimental Contextual Inquiry
Self-Assessment Manikin (SAM)	Co-discovery
Emotion Slider	Valence method

Face Reader	Pluralistic Walkthrough
Geneva Emotion Wheel	Workshops
PANAS (Positive and Negative Affect Scale)	Open Interview
Eye Tracking	
Affect Grid	
Presence Questionnaire	
AttrakDiff	
Multiple Sorting Method	
Hedonic Utility Scale (HED/UT)	
Aesthetics Scale	
Video rating with UX scale	
SUMI	
TUMCAT	
GSA GQM	
Intrinsic Motivation Inventory (IMI)	
System Usability Scale (SUS)	
Completion Rate (Success Rate)	
Number of Errors	
NASA-TLX	
SUPR-Q	
QUIS	
TAM Questionnaire	
meCue	
Repertory Grid technique (RGT)	
User experience questionnaire	
iScale	
Card Sorting Method	
Behavioral Observation (measuring presence)	
Fun Toolkit	

Study type

In general, UX is very context-dependent, which is why the experience with the same design can vary significantly in different circumstances (Roto et al., 2018). Therefore, different types of studies need to be considered. For the encyclopaedia of methods, study types are divided into three categories:

Laboratory. Lab studies are conducted in a man-made experimental set up, where the experimenter is able to control the setting and intervene in the event. These types of studies are very popular for testing immature prototypes in early phases of

product development. During a traditional lab study, the participant is given a task with one or more user interface designs and is asked to perform Thinking Aloud throughout the process. The experimenter then observes the participant's actions and tries to make sense of their motivations and needs in order to identify usability problems (Roto et al. 2018).

Field. In contrast to laboratory studies, field studies are performed in real-life situations where the experimenter takes the role of an observer and cannot intervene in the event. Field studies have the advantage that they provide most realistic results due to the context-dependency of user experience. Field study evaluation methods can either be used for investigating user experience during a specific amount of time, for testing a prototype or for interviewing and observing participants, all in a realistic context (Roto et al., 2018).

Online. Online studies are conducted over the internet. Participants can be anonymous people from all over the world or selective participants can be chosen per invitation only. A variety of collection methods can be employed to conduct online studies such as interviews, questionnaires etc. Online surveys or questionnaires can be the most time and cost-efficient way to collect data from a big number and wide range of participants. For example, they can be used as an extension to website testing, e.g. in form of the AttrakDiff questionnaire. Nevertheless, it is important to keep in mind that online surveys can be more time-consuming and costly if it requires specific equipment to deliver the system to the participants (Roto et al., 2018). Moreover, some people e.g. those without internet access are automatically excluded of the online studies (Wright, 2005).

For Thinking Aloud two of the four study types apply which can be seen in Figure 12.

Lab	Field	Online

Figure 12. Study type classification example for Thinking Aloud

Collection method

Data can be collected in a variety of ways. Collection methods in UX can either be on the attitudinal dimension, i.e. self-reports, questionnaires and interviews or on the behavioural dimension, i.e. physical measurements, observations and creative methods. Also, collection can be either quantitative or qualitative (Rohrer, 2014). In the following, collection methods that are most commonly used in UX are described in detail.

Self-reports. Data can easily be collected by using self-reports. Users will then be asked about their level of satisfaction or frustration after using a system or about their preference for a system over another, e.g. in form of questionnaires. These measurements have the advantage that they consider the users' feelings and emotions while using the system. However, with this method, feelings and emotions are interpreted as a response to the system and therefore they fail to consider other motivations and goals of the user (Spink & Heinstrom, 2011).

Questionnaires. Questionnaires are a common instrument for UX evaluations in order to collect data from participants. In questionnaires, questions can be used either in the open-ended format or in the closed-ended format. Open-ended questions allow the participant to answer freely without response options. Closed-ended questions will provide multiple response options where the participant can choose one of them (Wei & Moyer, 2008). A form of closed-ended questions that is commonly used in UX research is a semantic differential scale or Likert scale, which consists of a range of values describing an attribute. The most extreme values on each end of the scale are called anchors and indicate the section of the scale that is again divided by points. Participants give ratings by choosing one value on the scale (Hartson & Pyla, 2012). Examples for questionnaires of this kind are the Self-Assessment-Manikin Scale (SAM) or the Positive and Negative Affect Scale (PANAS). Self-report questionnaires are usually easy to interpret and inexpensive (Kline, 1993). Moreover, they are an easy way to collect data from a wide range of participants, especially online (Paulhus und Vazire, 2007). Still, the questionnaire's format, wording and structure can easily lead to several response biases, e.g. responding in a socially desirable way (Schwarz, 1999).

Physiological measurements. Another possible data collection method is using physiological measurements or biometrics. Biometrics can detect and measure autonomic or involuntary bodily responses triggered by the nervous system in response to emotions during interactions, for example heart rate, perspiration and eye pupil dilation. While those physical changes can be correlated with emotional responses and can therefore give information about experiences and emotions while using a system,

the great disadvantage is that the specialised monitoring equipment needed for such measurements is difficult to access for most researchers (Hartson & Payla, 2012).

Interviews. Interviews are a useful method of data collection for UX research as well. The general aim of interviews is to gather data about a certain topic, while at the same time allowing exploration of new issues. They can be applied either as unstructured, semi-structured or structured interviews. Unstructured interviews contain a general topic and agenda but no predefined questions or format with the aim to gather data about the participants' experiences without giving any form of restrictions on their response. In a semi-structured interview predefined questions as well as open-ended questions can be used and therefore it combines elements of structured and unstructured interviews. Structured interviews are predefined questionnaires with closed-ended questions and therefore a limited and fixed set of questions and responses. Advantages of structured interviews are that the responses are reasonably comparable and that the interviews can also be conducted by untrained interviewers. On the other hand, unstructured interviews give more space to explore new issues (Wilson, 2014).

Creative UX methods. Creative UX methods are especially valuable in early stages of design and can be combined with evaluative methods. They require the user to provide design improvements, e.g. by using drawings and other creative techniques (Sluis-Theischeffer, Bekker & Eggen, 2009). While other data collection methods are useful to identify problems of a current system or product, creative UX methods therefore make it possible to involve the user in the process of design and therefore in the process of finding solutions (Müller & Kuhn, 1993). Creative UX methods are not based on user evaluation and thus do not require a certain level of analytical skills. Instead, they are based on problem solving skills and a creative attitude. This is why they are especially useful when the user's evaluation capability is limited or easily influenced or if the users are young children. Still, creative methods can only be a complementary approach, used in combination with evaluative methods (Sluis-Theischeffer et al., 2009).

Observations. User actions can be observed by experts in order to gather data about usability. This can be done in field studies, lab studies or mixed method studies as well. A very common combination is to do user observations performed by experts, followed by user interviews. This is an effective way to combine opinions with objective data. Especially in early phases of the system expert observations and evaluations are reasonable to save further costs regarding the development and design of the system. However, to fully understand UX, an observation needs to be combined with other methods and cannot give conclusive results on its own (Roto, Mariannna & Väänänen-Vainio-Mattila, 2018).

The categorization according to the collection method for the example of Thinking Aloud is shown in Figure 13.

Questionnaire	Psychophysiological measures	Self-report	Collage/Drawing	Interview	User observation

Figure 13. Collection method classification example for Thinking Aloud

Time of evaluation

Time of evaluation refers to when the method is administered. This is divided into "before", "in situ" and "after". In some studies, it is important to determine whether the user has expectations of the product or service, and hence it is sensible to conduct surveys before use, which can then be combined with surveys during or after use. In some usage scenarios it is more suitable to measure the user's experience parallel to when the product/service is being used, for example in the case of physiological measurements or behavior observations. The most common scenario, however, is that the data collection only takes place after use of the product/service.

In the case of Thinking Aloud the time of evaluation is in situ as shown in Figure 14.

Before	In situ	After

Figure 14. Time of evaluation classification example for Thinking Aloud

Description

This section briefly describes the method. Additionally, advantages and disadvantages are listed that would aid in the selection process. As an example, the description for Thinking Aloud is shown in Figure 15.

Participants are asked to use the system while continuously Thinking Aloud.

Benefits:	**Disadvantages:**
• Cheap • Robust • Easy to learn as examiner	• Unnatural situation • Biasing users' behaviour

Figure 15. Description classification example for Thinking Aloud

Costs

When selecting suitable methods, it is essential to check overall expenses. Many questionnaires and tests are free of cost and readily available, however some may require licenses or entail equipment costs. Each method in the encyclopaedia is indicated whether it is available free of costs or bare license or equipment costs.

Figure 16 shows the category of Costs for Thinking Aloud.

Free	License	Equipment

Figure 16. Cost classification example for Thinking Aloud

Relevance for C4A

All the above-mentioned categories for each method in the encyclopaedia was systematically considered and its relevance for Content4All was assessed by a group of HFC researchers belonging to the fields of psychology and Human Factors. Each method was classified as either high, medium or low relevance for Content4All. Based on this assessment, UX methods will be chosen in order to evaluate the Content4All system.

The evaluation of relevance for Content4All of Thinking Aloud is the following:

> Medium relevance for C4A. Method is more useful in interaction context. However, interaction is not a relevant factor in C4A system as no interaction between system and user is observable.

More information or references

Lastly, any additional information and reference of each method are listed.

For Thinking Aloud, the reference listed is: Krahmer E., Ummelen N. (2004). Thinking about thinking aloud: A comparison of two verbal protocols for usability testing. IEEE Transactions on Professional Communication, 47(2), 105-117.

4.1.1.3. Relevant methods: Formative and summative testing plans

The purpose of the encyclopaedia of methods is to aid the planning of the user tests in WP5. As mentioned before, two user evaluation stages are essential: the first stage, pilot evaluation stage 1 (T5.2), consisting of user evaluation with regards to interface design and virtual signer i.e. formative testing, and the second stage, pilot evaluation stage 2 (T5.7), which focuses on the overall translation and system performance of the C4A system i.e. summative testing. Formative testing regarding the quality of appearance and behaviour of the virtual signer was provided as non-formalized feedback from user experts and user organisations throughout the project so far in project meetings with technical partners. Feedback was based on the insights of the initial user research, and the preferences of the user group.

Formalized formative feedback on the TV layout will be gathered via online questionnaire regarding different designs of the TV picture layout during virtual signing with the goal to assess the effectiveness and satisfaction of potential users with

respect to different TV interface designs (qualitative & quantitative data). The questionnaire will consist of three parts of data collection: In the first part a combination of variables (size, type of background) resulting in different possible sample layouts is presented to the participants in random order. Participants are requested to rate each combination on a 5-point Likert scale using a star system. The second part of the questionnaire features open questions. In part three, participants have the opportunity to create their own TV layout with free positioning and sizing of the virtual signer and TV video, free tilting of TV video, etc.

Formalized summative feedback on the virtual signer within the reference scenario "weather forecast" will be gathered in focus groups with the involved user organisations in Switzerland and Belgium. For this purpose, videos are created featuring the project demonstrator, which are watched and discussed in a semi-structured group interview. These focus groups and the online questionnaire mentioned above are planned to be conducted in autumn and winter 2019.

The encyclopaedia of methods will assist in choosing the most applicable evaluation tools based on their relevance for both user evaluations, formative and summative, of the C4A system. Each method is given a relevance-rating based on certain important UX aspects that had been identified to be more significant with regards to user-testing of the C4A system: overall UX, acceptance, aesthetics, stressful/demanding and usefulness. Other factors such as collection method, study type, and costs are also taken into consideration during this assessment. The final list of methods is presented in Table 18. Some of these methods are redundant regarding the type and content of data collected. Thus, not all of the methods listed will be used for the above-mentioned evaluation phases. Methods will be selected in accordance with availability of demonstrator components and their maturity as well as technical set-up and feasibility of study conduction.

Table 18. Overview of relevant methods for pilot evaluation stage 1 (P1) and pilot evaluation stage 2 (P2).

Method	Description	Significance for C4A	P1	P2
SAM (QoE- social qualities, questionnaire, quantitative)	Measures pleasure, arousal and dominance	Independence of language		x
3E Expressing, Experience & Emotion (QoE - social qualities, collage/drawing, qualitative)	Users draw and write their experiences and emotions	Possibility of pictorial and verbal reporting		x
AXE (QoE – social qualities, self-report, qualitative)	Uses images as interview stimuli for metaphorical thinking and reflection	Valuations are freely defined by users	x	
Affect Grid (UX - hedonic qualities, self-report, quantitative)	Single item scale with three dimensions	Simple and effective		x
AttrakDiff (QoE - hedonic qualities, questionnaire, quantitative)	Aims to get a global positive-negative evaluation of a product	Fits the framework of C4A user evaluations		x
Hedonic Utility Scale (UX - hedonic qualities, questionnaire, quantitative)	Captures hedonic qualities, utility and usability	Usable in different settings		x
Aesthetics Scale (QoE – hedonic qualities, questionnaire, quantitative)	Especially used for websites	Two factors of users' perception are measured		
Co-discovery (UX - pragmatic qualities, self-report, qualitative)	Measures UX with interaction among a group of users	Allows discovery of new aspects through discussion	x	x

Pluralistic Walkthrough (UX – pragmatic qualities, self-report, qualitative)	Meeting of users, usability experts and product developers	Easy to identify number of UX problems in one sitting		
Workshops (UX - pragmatic qualities, self-report/interview, qualitative)	General discussions and collections of feedback	Reliable tool supporting the users' imagination	x	x
User Experience Questionnaire (UX- pragmatic qualities, questionnaire, quantitative)	Measure of subjective impression during interaction with a product	Comparable to AttrakDiff and Hedonic Utility Scale		x
meCUE (UX - pragmatic and hedonic qualities, questionnaire, quantitative)	Consists of four separately validated modules	Evaluation of holistic UX		

5. Summary

This document describes the project plans for the integration, pilot demonstration and evaluation activities.

Initially it briefly recalls the system architecture, thoughtfully described in D2.3, and then it describes in detail the requirements of each component of the Pilot 2 Demonstrator, including software and hardware needed to deploy the components in a real environment.

The focus then moves to the integration, starting from an initial integration plan of the system components and the description of integration strategies. The inter-component (between the main modules of the architecture) and intra-components (between sub-components of each module) integration actions are then exposed in detail, together with the description of related evaluation actions.

Finally, the deliverable describes the methodologies that will be applied for the Pilot 2 Demonstrator evaluation, which will include both objective and subjective evaluation in different steps involving an increasing number of users (from experts' evaluation to group discussion).

The outputs of the Pilot 2 Demonstrator integration and evaluation will be properly described in the deliverable *D.5.8 "Integrated Pilot 2 system evaluation report"*.

Bibliography

Braun, V. & Clarke, V. (2006). Using thematic analysis in psychology. *Qualitative Research in Psychology, 3*, 77–101.

Brooks, P., France, T. & Hestnes, B. (2010). User measures of quality of experience: why being objective and quantitative is important. *IEEE Network: The Magazine of Global Internetworking - Special issue on improving quality of experience for network services archive*, *24*(2), pp. 8-13. doi:10.1109/MNET.2010.5430138.

Diepold, K. (2012). The Quest for a Definition of Quality of Experience. *Qualinet Newslet, 2*, 2-8.

Glaser, B. G., Strauss, A. L., & Strutzel, E. (1968). The discovery of grounded theory; strategies for qualitative research. *Nursing research, 17*(4), 364.

Hartson, R. & Pyla, P. (2012). *The UX Book – Process and guidelines for ensuring a quality user experience*. USA: Elsevier.

Hassenzahl, M. (2008). User experience (UX): Towards an experiential perspective on product quality. In *Proceedings of the 20th International Conference of the Association Francophone d'Interaction Homme-Machine*, pp. 11-15. doi:10.1145/1512714.1512717.

Hassenzahl, M., & Tractinsky, N. (2006). User experience – a research agenda. *Behaviour & Information Technology, 25*, 91-97.

Kline, P. (1993). *Personality: The psychometric view*. London: Routledge.

McCusker, K., & Gunaydin, S. (2015). Research using qualitative, quantitative or mixed methods and choice based on the research. *Perfusion, 30*(7), 537–542. https://doi.org/10.1177/0267659114559116

Moser, C. (2012). *User Experience Design*. Berlin/Heidelberg : Springer-Verlag.

Muller, M. J. & Kuhn, S. (1993). Participatory Design. *Communications of the ACM - Special issue Participatory Design, 36* (4), 24-28.

Oehme, A., Horn, H.-P., Wieser, M., Waizenegger, W. & Fernando, W. A. C. (2016). User Experience in immersive TV – A research agenda. In *Proceedings of IEEE International Conference on Image Processing (ICIP)*. 25 – 28 Sept. 2016, Phoenix, USA. doi: 10.1109/ICIP.2016.7532376

Paulhus, D. L., & Vazire, S. (2007). The self-report method. In R. W. Robins, R.C. Fraley & R.F. Krueger (Eds.), *Handbook of research methods in personality psychology* (pp. 224-239). London: The Guilford Press.

Richardson, J. & Gwaltney, W.A. (2005). S*hip it. A Practical Guide to Successful Software Projects.* s.l. : The Pragmatic Programmer.

Rohrer, C., (2014). *When to Use Which User-Experience Research Methods*. Nielsen Norman Group.

Roto, V., Marianna, O. & Väänänen-Vainio-Mattila, K. (2018). *User experience evaluation methods in academic and industrial contexts*. Proceedings of UXEM 09 workshop.

Scholtz, J. (2004). Usability evaluation. *National Institute of Standards and Technology*, *1*.

Schwarz, N. (1999). Self-reports: How the questions shape the answers. *American Psychologist, 54,* 93-105.

Sluis-Theischeffer, W., Bekker, T., & Eggen, B. (2009). *Adding user creativity to the UX toolbox: Exploring the use of Creative UX methods*. Netherlands: Proc. CHI

Spink, A. & Heinstrom, J. (2011). *New Directions in Information Behaviour* (1st ed.). UK: Emerald Group Publishing Ltd.

Thüring, M. & Mahlke, S. (2007). Usability, aesthetics, and emotions in human-technology interaction. *International Journal of Psychology, 42* (4), 253-264.

Wei, L. & Moyer, M. G. (2008). *The Blackwell guide to research methods in bilingualism and multilingualism*. UK: Blackwell Publishing Ltd.

Wilson, C. (2014). *Interview techniques for UX practioners*. USA: Elsevier.

Wright, K. (2005). Researching internet-based populations: advantages and disadvantages of online survey research, online questionnaire authoring software packages, and web survey services. *Journal of Computer-Mediated Communication*, *10* (3).

Wu, W., Arefin, A., Rivas, R., Nahrstedt, K., Sheppard, R., & Yang, Z. (2009). Quality of experience in distributed interactive multimedia environments: toward a theoretical framework. In *Proceedings of the 17th ACM international conference on Multimedia (MM '09),* ACM, New York, NY, USA, pp. 481490. doi: http://doi.acm.org/10.1145/1631272.1631338

PARTNER SHORT NAMES

Short Name	Name
FIN	Fincons Group AG
UNIS	University of Surrey
HHI	Fraunhofer Institute for Telecommunications Heinrich Hertz Institute
HFC	HFC Human-Factors-Consult GmbH
TXT	Swiss TXT AG
VRT	Vlaamse Radio -en Televisieomroeporganisatie