ITEA Office
High Tech Campus 69 - 3
5656 AG Eindhoven
The Netherlands
T + 31 88 003 6136
E info@itea3.org
W www.itea3.org

ITEA 3 is a EUREKA strategic ICT cluster programme

D3.1.2 – Application Design – year 2

MOS2S

Media Orchestration from Screen to Screen

DRAFT VERSION

Edited by: J. Belpaire (Kiswe)
Contributions from:

- IMEC, KISWE, VRT(draft)
- Gerade
- Game ON
- ETRI

Version: v02
Date: 21-12-2018
Delivery date: 21-12-2018

Project key data

ACRONYM and full-length title

15022	MOS2S
Program Call	ITEA 3 Call 2
Full-length Title	Media Orchestration - Sensor to Screen
Roadmap Challenge	Urbanisation

Description

Novel and ubiquitous consumer-priced audiovisual sensors and data in particular, represent an important aspect of the Smart City environment, enabling a variety of applications for citizen information, participation, entertainment, experience, safety and security. Every user becomes a potential source of information, either directly or through social media buzz and its discovery. Audiovisual media provide citizens with Smart City data readily accessible to human senses. With the MOS2S project (Media Orchestration from Sensor to Screen), an international consortium of partners will develop and test audiovisual Smart City technologies and solutions in the context of citizen needs, and embed these solutions within the Smart City Playground.

Project duration & size

Size	Effort: 133.67 PY	Costs: 13.9 M€
Time frame	Start: 2016-10-1	End: 2019-09-30 (37 months)

Coordinator

Netherlands	TNO
Type	Research Institute
Contact Person	Gjalt Loots
Email Address	gjalt.loots@tno.nl

Consortium

Belgium	Nokia, iMinds, Kiswe Mobile, VRT*
Korea, Republic of	ETRI*, Mooovr
Netherlands	Amsterdam ArenA*, Bosch Security Systems B.V., Game On, Inmotio Object Tracking BV, Koninklijke KPN NV, TNO
Turkey	Bor Software inc.*, DİA Yazilim San. ve Tic. A.Ş., KoçSistem, TMOB BİLİŞİM

Table of Contents

List of Figures

Project acronyms

3DoF	3 degrees of freedom
BE	Belgium
CPA	Conformance Points A
CPB	Conformance Points B
DVB	Digital Video Broadcasting
IM	Instant Messaging
IT	Italy
KR	Republic of Korea
MOS2S	Media Orchestration Sensor To Screen
NL	(The) Netherlands
OB	Outside Broadcasting
PTZ	Pan-Tilt-Zoom
TR	Turkey
UHD	Ultra-High Definition
VR	Virtual Reality

1. Introduction

In Deliverable D3.1.2, the MOS2S consortium partners provide an overview of the application designs that have been designed, and of which some were demonstrated at the second year demonstration event (IBC2018). D3.1.2 is the updated deliverable from WP3 deliverable D3.1.1, in which we outline the design specifications for dedicated applications running on top of the platforms supporting the demonstrators. There have been some new application designs presented, but also some partners have updated their existing designs from year 1, in an attempt to take real life experiments feedback and market evolution into account.

In WP3, the platform functionality and generic components researched and prototyped in WP2 need to be combined with applications and use case specific functionality that will be designed and integrated in this WP. The resulting applications and the underlying platforms will be used to drive the yearly MOS2S demonstrators in the respective domains.

D3.1.2 is primarily a software deliverable. This document provides descriptions of the application software, as developed and being developed by partners as part of the MOS2S project. It is an outcome of T3.1 (Application design).

2. NL Application Design

2.1. Companion Screen application

Game On has extended its iPad application with a function that enables "screen mirroring" to RTMP endpoints. This function is similar to Apple's Airplay but supports "long distance". It allows the user to effortlessly share his or her iPad session with a remote audience. The mirroring function not only captures the user's screen but also the user's voice, which allows for a high level of interaction. Depending on the use case, the RTMP endpoint may be made available to a limited audience (such as team members) or to the public (such as fans). The user can turn on and off the mirroring with the touch of a switch. The stream is h264 encoded using the iOS hardware acceleration API, minimising resource usage during operation.

The new feature was tested in a live scenario during the Holland - Peru friendly match in the Amsterdam Arena. An iPad was made available to an “expert TV analist” who was able to make several on-the-spot analyses during the game. The session was streamed realtime to an RTMP endpoint provided by Kiswe, allowing a selected audience to access to the stream inside Kiwse’s application. This test proved successful.

3. BE Application Descriptions

3.1. Editorial tool for professional users (to be updated)

This is the tool where editors can interact with the end users and where they can select and publish content which are shared from the end users. It's also the place where they can initiate polls and set up automated bots which can send messages at a specific moment in time or who can respond to common questions.

The editorial dashboard is a web app, available at a public URL. It is also developed in Angular and Ionic. This enables us to build the software to a mobile app if it should be interesting for editors.

3.2. Mobile app for end users (to be updated)

The mobile app is the tool that's being used by the end users. It's the place where end users can share photos and videos with the editorial team. The end user can also interact with the editors and participate in polls. Bots can automatically help users with common questions, lowering the management load for editors.

The mobile app is built with web technology: Angular & Ionic as framework, combined with SASS based UI refinements which enables us to create a somewhat modified design for each event.
Another software stack we use is the open source project Cordova. With it we can build the web based software to mobile applications which are published to the Google Play Store and Apple App Store through the IDE's which are respectively Android Studio and Xcode.

3.3. Social capturing/discovery tool (to be updated)

The social capturing/discovery tool is a set of software components that enables the editor to search the social web for interesting content on a specific topic. The tools are being built on top of Firebase, Node.js, ElasticSearch, Kibana and the API's from Instagram and Twitter.

3.4. MOS2S Extensions (to be updated)

In this project we build on top of the results made from the toolset for crowd contribution and interaction that we have built in the ICoSOLE project, called "Wall of Moments". This toolset consists of an end-user app to contribute content and get updates from the editorial team ('Moments'), and an editorial toolset to request content from end-users, to segment and interact with them ('Trademark') and an app to display content in an interactive way during an event ('the Wall'). ICoSOLE was a -European Union's Seventh Framework Programme (FP7/2007-2013)

project under grant agreement n° 610370. More info about that project can be found at the website http://icosole.eu/.

Our goal in MOS2S is to enhance this toolset, and more specifically, expand the editorial app. We rebranded it to 'Switchboard' and plan to add location-based functionalities, integration with the Kiswe Cloud Clipping service and a more advanced tagging system to segment users and content. The refactored software is also built as a modular system. This enables us to prototype new insights faster and make pieces of software or services available to partners.

Short overview of the software platform

The platform consists of 4 core parts: a mobile app, an editorial dashboard, several backend services and a social capturing/discovery tool.

Those four core parts have been extended with

- a tool for the creation and publishing of a living and up-to-date long form article
- a digital signage tool, called "the Wall", to publish selected content on big screens at an event
- an automated service for the creation of personalized aftermovies

3.5. Iterative updates during test cases (to be updated)

In this section we want to outline software updates and refinements we did based on learnings during test cases.

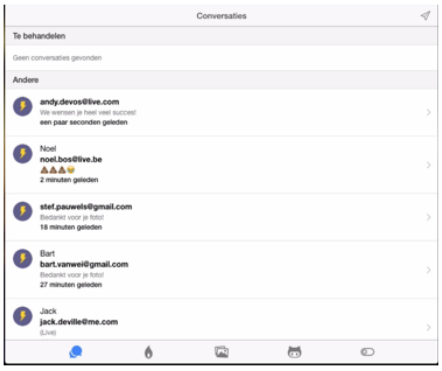
Music for Life (November/December 2016)

This was the first test case with the refactored version of the apps used in ICoSOLE. It included a modular design, tagging system and a first version of the new editorial tool 'Switchboard'.

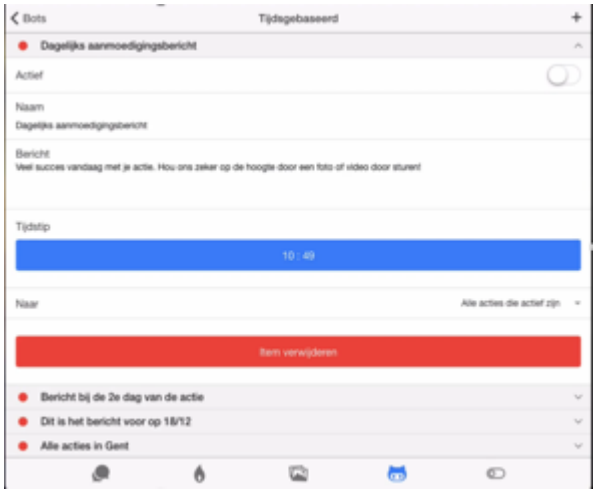
Most important new features include:

- Testing co-creation and interaction between production team and event organizers: contributed content was shared on third-party platforms for the first time (user generated photos & videos have been used on Facebook and on television as an overlay)

- First test of using a chat interface as the main interaction between the end user and the editors



- Introduction of the concept of chat bots: end-users were able to ask questions and got an automatic reply (if found). We also introduced time-based bots, which sent out personalized messages for every user on certain points in time.



- End-users received a personalized aftermovie following the event, which consisted of their sent-in photos. This was achieved using an After Effects plugin and a rendering engine.

Het Vooruitzicht 2017 (January 21, 2017)

During this event, several simultaneous presentations of VRT News journalists and experts were held. During these presentations, end-users were able to send their questions to the speaker, as well as participate in polls. Lastly, 'the Wall' app was revived and made more interactive by adding beacons.

Added features:

- Asking questions: Switchboard was updated to display separate feeds of messages, based on tagging; these feeds were used to filter incoming messages for each speaker; moreover, important messages could be starred and filtered out
- Interactive polls: the editorial team was able to send out polls (yes/no, multiple choice or score) to (a segment) of users, and follow the results live in Switchboard.

- Curated generated photos and videos were shown on 'the Wall' (on a big LED screen); the app driving the screen was connected with beacons near the screen, and adapted what was shown based on people nearby.

Ronde van Vlaanderen (April 2, 2017)

In partnership with Sporza, we created the app "Rondereporter", which people could use to send in their best moments of the cycling race by using the app. The main objective was to test a new way of publishing user-generated content by using a longform article.

- The editorial team was able to curate the best content, and to accommodate it with a title and/or description and publish it to the article live.

"Meld Nieuws" app & Trump's visit to Belgium (May, 2017)

For one month, users were able to report news to the VRT news department, using a dedicated app. At the end of the testing period, Trump's visit to Belgium was planned, and the (location based feature of the) app was used to locate people nearby the places Trump visited.

Enhanced and new features:

- Push notifications and polls: we wanted to test how to best handle co-creation by setting up several small experiments. These were led by a small professional editorial team inside the news department
  - Collecting local news, proposed by the test users (example: a raid happening in a city)
  - Opinions on news events (example: second hand sale successes)
  - Doing an experiment and submitting feedback (example: traffic experiment)
- Location based services: we included a map in 'Switchboard' which showed the location of the users. Based on their location, the team was able to coordinate the people that were using the app. Especially during president Trump's visit we tested this by sending out 16 reporters in the field guided by an editorial team located at VRT.

3.6. KISWE CrowdStreaming

UGC contributions moderation

In our effort to seamlessly allow UGC live video contributions into professional productions, dialogues with VRT producers required us to introduce a moderation layer. This layer would allow to omit false positives, bad intent ànd bad quality contributions. The first approach to improving the quality of contributions was to implement not just moderation, but also a direct communication channel 'off air', so that the contributors could be validated but also coached by the moderator. This live video protocol was also then used to establish the connection between the user contributor ànd an eventual host of a live TV broadcast production. This is how we raised immediate interest for embedding the CrowdStreaming layer in the 2017 Music for Life TV broadcast production.

The design features of the CrowdStreaming application are:

- The selected implementation allows for app-independent dial in and provides huge potential for easy web platform single-button-dial-in integrations
- It supports both Android and (partially) iOS
- It provides an easy path forward to integrate into TV broadcast workflows
- It supports multiple simultaneous live events on same backend

- Security via moderator control authorization

The KISWE Application Architecture

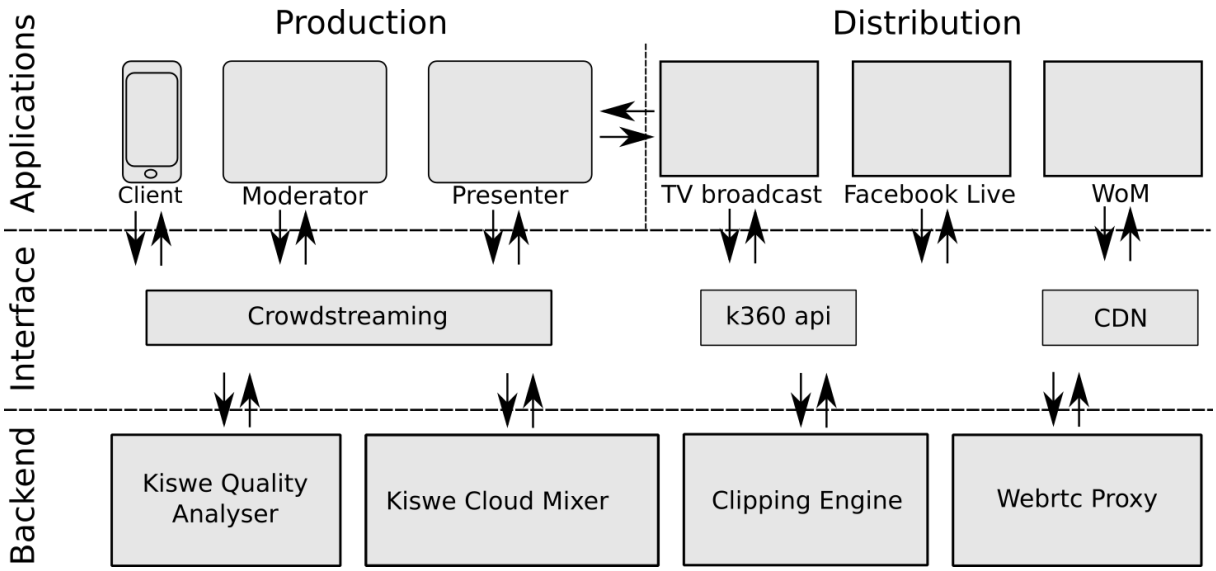


Figure 1: KISWE Application Architecture

Production Applications:

- The moderator can communicate with each of the callers for manual coaching and gets an overview of the callers' network and battery. He can then choose to push the caller forward into the live show and connect him with the presenter.
- The presenter does a one-on-one video call with one of the callers and can switch to the next caller in the queue without any frame drops.
- The client is a javascript api, which other 3th party customers can use to integrate in their own app like the "Warmste week" app, as the VRT team did for Music for Life.

Cloud Interface and Backend:

- The Crowdstreaming backend handles the communication and orchestration between clients and the Kiswe Cloud Mixers
- WebRTC Proxy (wowza): all webrtc calls are relayed through these servers to allow one-to-many webrtc streaming.
- The active Kiswe Cloud mixer listens in on webrtc calls through the webrtc proxy and mixes them into a live show. This can then be distributed to the Kiswe app, facebook live or any other channel over RTMP. The mixer also allows other types of input streams like RTMP, which we use for instance to replace the presenter webrtc video with a higher resolution RTMP stream from a broadcast camera. These streams were then frame accurately synced and allowed the caller to see the presenter properly.

- The Clipping engine allows for fast clipping in the live stream to integrate with 3th party applications such as Wall of Moments (WoM)
- The Kiswe quality analyser keeps track of the network stats of each client connected with the CrowdStreaming backend. It measures network bandwidth, latency and jitter and calculates a score to indicate the predictability of the users' network. This Network score is based on the Mean Opinion Score (MOS): https://www.voipfuture.com/wordpress/wp-content/uploads/2015/10/VPF123_WP_MOS-Calculation-And-Aggregation.pdf

The CrowdStreaming Moderator interface

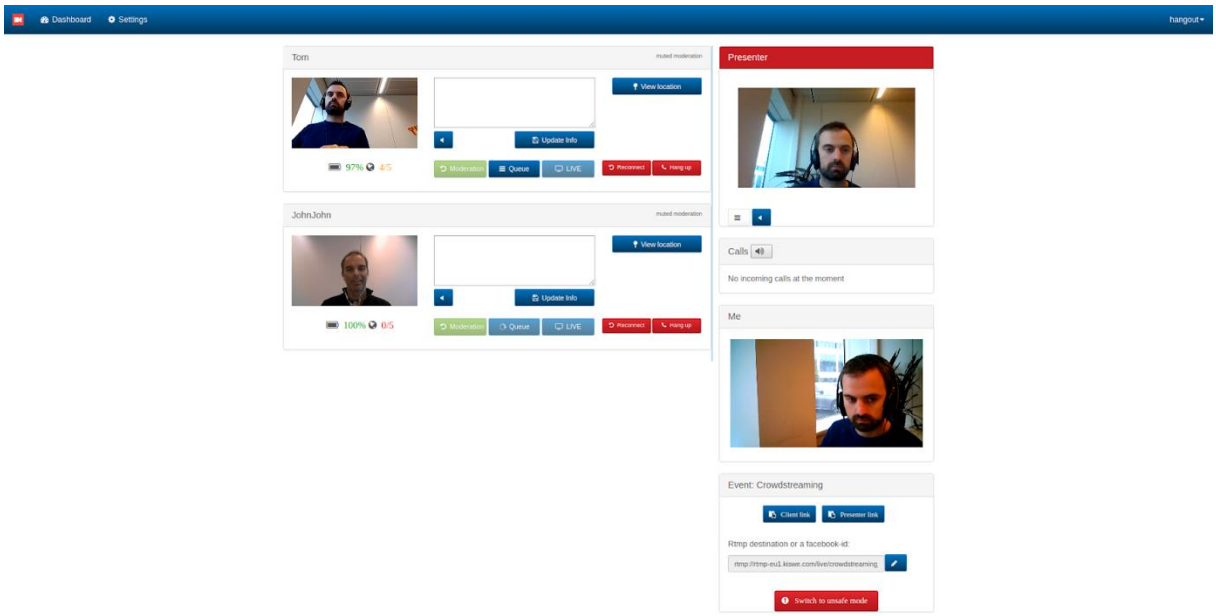


Figure 2: CrowdStreaming Moderator Interface

The CrowdStreaming moderator interface allows the moderator to manage all incoming callers, push them through a workflow into the queue, before pushing them into the live show. Options for direct chat with the presenter in the studio, pushing the output to social network livestreams ànd fast inspecting the caller's connection quality are available to the moderator.

Live Hangouts @ Music for Live 2017

Below an impression of the Facebook Livestreams we performed in the lead up to Music for Life 2017 the week before. Multiple charity fund raisers called into the Studio Brussel liveshow to talk about their actions and good causes.

Figure 3: Music for Life Facebook Live events

https://www.facebook.com/stubrumusicforlife/videos/10159701889145176/

https://www.facebook.com/dewarmsteweek/videos/1616197641774294/

During the event itself, a break-in production concept was developed to let two callers bring their story and show how they were raising funds for their good causes. The setup here had a more physical connection between the presenter on stage ànd the caller via a TV screen put on stage. The impression of a real conversation was triggered this way and added to enhancing the TV production experience.

Figure 4: Music for Life CrowdStreaming On Stage

Live Hangouts @ Villa Sporza covering World Cup Soccer Russia 2018

During the World Cup Soccer in Russia 2018, we repeated a similar live TV broadcast production item in Villa Sporza, the daily pre- and post-game studio shows. A big screen in studio would

show the contributions, and callers from all-over the world would make their contributions. We had multiple contributions from a cyclist going from Belgium to Russia, a person in Panama travelling to the embassy for the game, a referee, a coach, a youth-friend of a Belgian team player, etcetera...

Figure 5: Villa Sporza WorldCup CrowdStreaming

3.7. IMEC RMLStreamer and DS Visualizations

Imec contributes to the project with 2 tools: the RMLStreamer and the DS visualizations dashboard. The two tools are aligned to aligned to each other offering a complete workflow from the media (sensors in this case) to the screen (in the form of visualizations). The two tools are presented in detail.

In this project, we originally planned to build upon the RMLMapper, a tool designed for generating Linked Data from heterogeneous data sources. However, MOS2S use cases that we are involved in, are related to streaming data whose processing is fundamentally different compared to static data that the RMLMapper is optimized for. In principle, generators for static data (files), like the RMLMapper, are limited by memory constraints, and generators for dynamic data (streams) are limited by the CPU speed of an individual node. Therefore, after thorough investigation, we concluded that we would achieve better results if we aim for a new solution compared to extending the existing solution.

To achieve that, we looked into the use cases and defined the requirements that drove the design of the new generator, the so-called RMLStreamer. The RMLStreamer is a generator that parallelizes the ingestion and mapping stages of Linked Data generation across multiple instances. The RMLStreamer handles dynamic data of any velocity by scaling the mapping process to multiple nodes. Our newly proposed approach is driven by observations of workloads from our existing generator, i.e., the RMLMapper. Firstly, we investigated and identified aspects of the Linked Data generation process that can be parallelized. Then, we came up with a solution that parallelizes the Linked Data generation process following the analysis of the different identified aspects.

The distributed and parallelized Linked Data generation process, which is designed for both static and dynamic data, can be divided in three main subtasks: (i) ingestion, (ii) mapping, and (iii) combination. These subtasks are linked to each other following the producer–consumer paradigm. The producer and consumer are two concurrent processes which use a common buffer as a queue. The producer generates data into the buffer and the consumer takes data out of the buffer. Multiple instances of each subtask can exist in parallel. Each instance of a subtask produces data in buffers in memory for consumption for the next subtask. The ingestion subtask consumes data from data sources and produces data records. The mapping subtask then consumes these data records and maps them to Linked Data according to the available rules. The combination subtask consumes the results from all mapping subtasks that are available and reduces this to a single Linked Data set as shown in the figure.

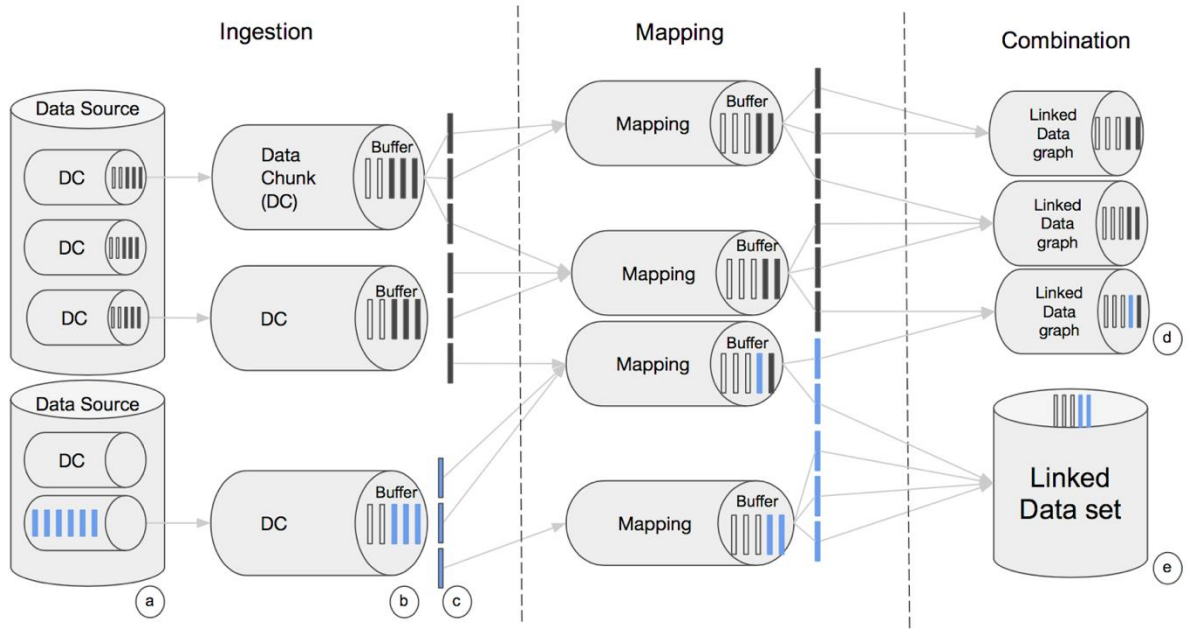


Figure 6: RMLStreamer Linked Data generation

Parallel Linked Data generation from dynamic data. In parallel, multiple Data Sources [DS] can be ingested, its data can be split in Data Chunks [DC], their Data Records [DR] can be processed, and (iv) their mapping can be performed to generate Linked Data.

The RMLStreamer is a Scala implementation built on top of the distributed processing framework Apache Flink, for handling the parallel execution of each subtask of the generation process over multiple (distributed) instances. The subtasks of our proposed approach are implemented and put together as part of a Flink pipeline. A job is executed by a running instance, a task manager that orchestrates given tasks on a local node or a cluster of multiple nodes. Flink pipelines are defined by several operators that handle input, transformations and output. If an operator is parallelizable, Flink's task managers distribute the execution of this operator over multiple

(distributed) instances. The RMLStreamer workflow consists of three steps: Flink runtime setup, Mapping configuration, and RMLStreamer execution. At first, a Flink runtime is configured and start running. Hereafter, a set of RML rules are provided which is used by the RMLStreamer for execution on the Flink runtime.

The DSLab dynamic dashboard uses semantic reasoning to suggest visualizations for sensor data. It automatically suggests appropriate visualization options for specific types of data and can be applied in different use cases by changing the underlying domain model. The visualizations dashboard consists of 3 core components: (i) sensor gateway, (ii) broker, and (iii) client. The sensor gateway is used as a proxy for the available data services, visualization services and aggregation services. The broker is the core of the decision framework and has two core functionalities: (i) discovering sensor services and (ii) reasoning about interesting aggregations and visualizations matching with the available sensor services. The client component is able to visualize sensors that the user has selected.

In the frame of MOS2S, the DSLab dynamic dashboard is extended to accommodate visualizations suitable for the MOS2S use cases. In particular, the demonstration will take place at schools, therefore the visualizations were needed to be adjusted for children. In the figure below, we show a mockup of the visualisations which are being designed. In more details, we developed a colored score widget, that indicates if the data values are in the desired threshold. To achieve that an extra option in the dashboard is required that allows a user to specify “observation ranges” (e.g. number from 15 to 25 to be shown in green as indicated above) when creating a visualization. Moreover, we developed smiley visualizations next to the numeric values which allow children to easily identify if the data value is within the desired threshold or not. Last, we translated parts of the interface in Dutch.

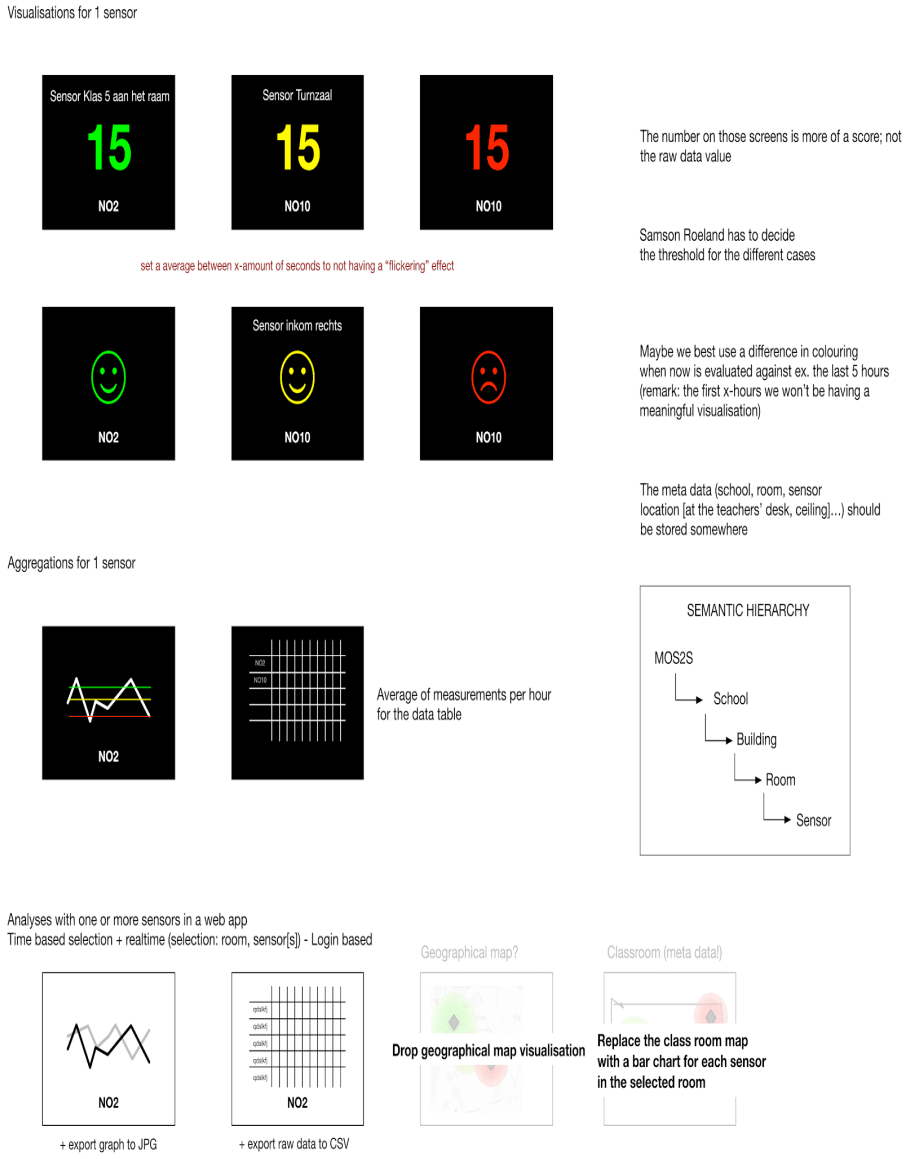


Figure 7: DSLab Dynamic Dashboard

4. KR Application Descriptions

4.1. Wide Field of View Video Experience

Figure 8: 12Kx2K capturing, stitching transmission and rendering experience in Korea.

Normally, wide field of view videos providing a viewing angle of more than 100 degrees can cover a person's viewing angle. If MOS2S provide such wide field of view without losing any quality degradation, users can enjoy immersive experiences like being in the field. Besides MOS2S project, Korea consortium tested a long-distant transmission with 12Kx2K resolution video which covers about 100 degree viewing angle shown in Figure 8. Based on this experience in Korea, MOS2S is going to have an integrated and cooperative field test in Johan Cruyff Arena which holds many valuable professional soccer games and public events.

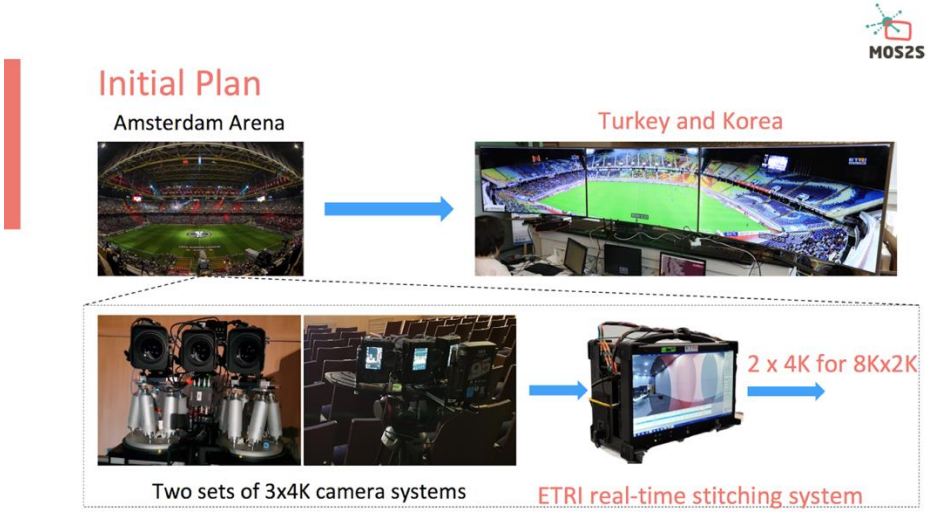


Figure 9: Initial Application plan for the 2nd use event

Figure 9 presents the initial application design of the 2nd year use case. At Johan Cruyff Arena in Amsterdam, Netherlands, MOS2S will capture a wide field of view video with multiple 4K-grade cameras and a real-time stitching system will manipulate the multiple camera sources into a seamless stitched video. The stitched video may be transmitted to Korea and Turkey if possible and users can enjoy the stitched stream with multiple displays. In order to achieve an ene-to-end real-time consuming, a special network environment with low latency and almost lossless must be required.

5. TR Application Descriptions

5.1. Gerade Online Debate Application

Online debate application is venue stream based online entertainment platform where a member can create a debate, invite other users and exchange their ideas, share their comments, publish streams or videos relevant to debate topics, or only watch the debates and vote the debater or moderator by sending notifications or emojis . They also can rate the surveys prepared by moderator. Adding resources, publishing surveys, banning or kicking off users are some main features included online debate application.

To develop this application, software tools and technologies will be used as mentioned below:

1. Nginx 1.10.x
2. Node.js 8.11.x (for backend)
3. Sails.js Framework 0.12.14 (for backend)
4. Kurento Media Server 6.7.1 (for webrtc)
5. Wowza Media Server 4.7.5 (transcoder)
6. Redis 4 (for data storage)
7. MYSQL 5.7 (data storage)
8. Angular.js 1.6.6 (front end)

Upon user requirements and current technologies, three layered landscape architecture software will consist of one client (angular), one signaller(Kurento to handle media streaming and videos), one node server(to handle static web content at backend). General architecture will be provided according with the steps mentioned below:

1. Only a presenter can connect to the system and create a session at client side. If more than one attempts to create session occur, an error message will be sent to the users.
2. Multiple viewers can join sessions and there is no limitation with respect to how many viewers will be added.
3. Viewers can leave the communication at any time.
4. Only present can close the session and termination message will be delivered to viewers.

At system design level, user login and registration will be handled at application server. The tasks related to adding and publishing a stream, web urls or medias will be executed via Kurento media server. Communication protocol between application server and kurento media server will be provided as JSON format.

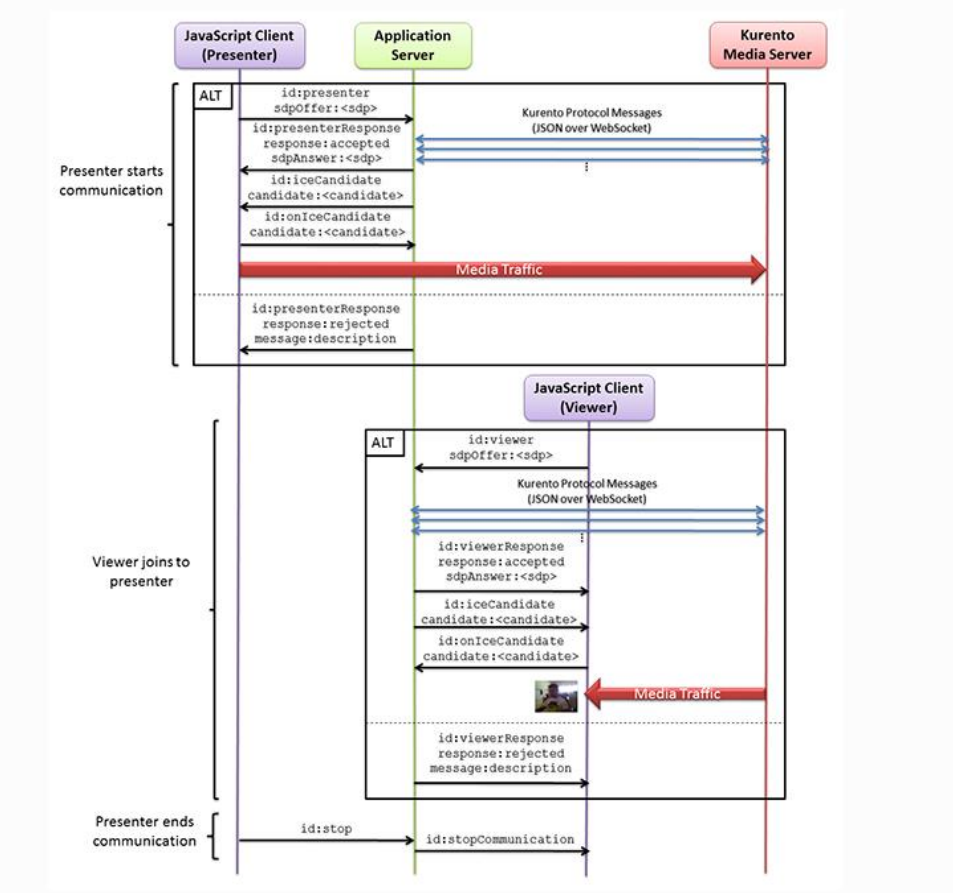
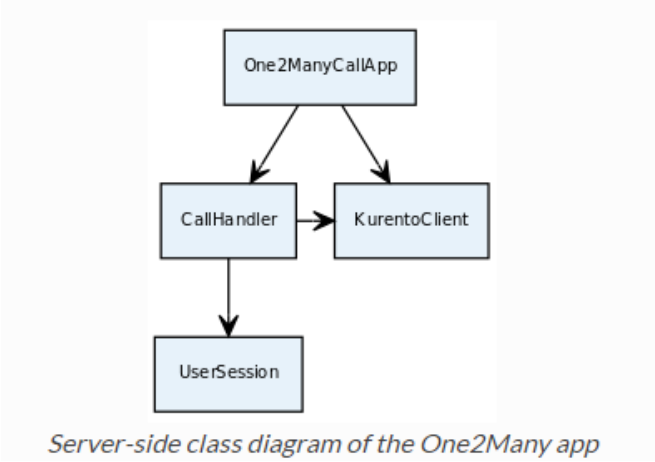


Figure 10: Gerade 3 Layer landscape architecture

Sequence Diagram of Online Debate Application

At server side, Java Spring Framework were used o to create one2many application shown below.



Server-side class diagram of the One2Many app

Figure 11: One2Many app Serverside class Diagram

Based on One2Many Application after session is initiated by presenter by evoking Spring Bean, presenter connects all viewer to create peer to peer network. At server side, presenter will create Kurento pipeline to act on media streaming and data exchange. Also this architecture allows user session storage, media exchange and streaming with ICE-on based protocol, data brokerage between participants, easy configuration.

A few screenshots taken from online debating event taken place 6th september during Holland-Peru match provide some insight into how our program look like and how to integrate with external applications and tools.

Figure 12: The Debate Screen

Figure 13: Voting Debaters

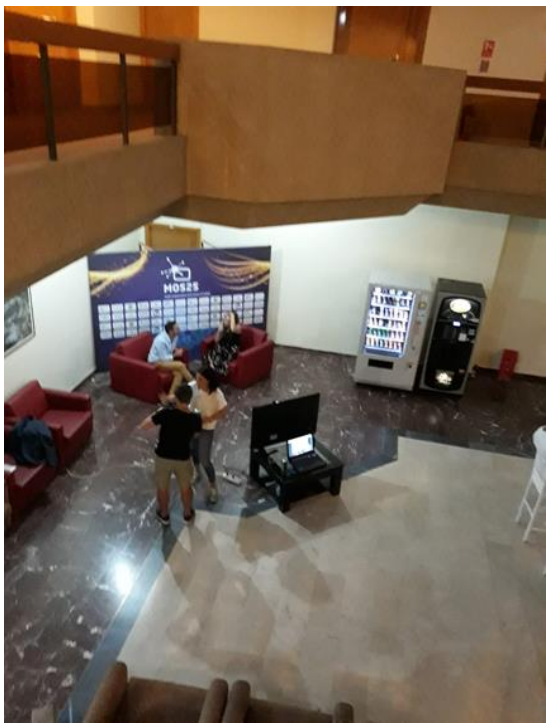


Figure 14: Photos from Netherland-Peru match

Feature extension of MOS2S application developed within second term are mentioned as following items:

1. **Adding IPTV camera**: Beyond adding web url, youtube videos or HLS streams , the feature all the records taken by IPTV camera can be added and published during debate event is extended in MOS2S application. Thereby, live streams around the place is managed to publish without need any external tools.

2. **Publishing RTMP**: Before all RTMPs coming from different resources had to be converted in HLS format via WOWZA resulting in synchronization and lag problems. Thus, instead of using extra tools like Wowza, a plugin is developed to publish RTMPS directly to the online debate, provided as an enhancement. This extension reduced synchronization and latency related concerns during online debate, accelerated media streaming and data sharing, and bolstered media orchestration in the application.

6. Concluding Remarks

This Deliverable D3.1.2 reports on the application designs for the 2nd year milestone. The document outlines the various applications that the MOS2S partners develop to showcase the innovations at the second year demonstrator. With respect to the initial Full Project Proposal, partners have developed application software that focuses on their interests and market developments, and continued concentrating more on contributions to live (sports) events. A variety of applications have been developed, in which several form of integration between data and video have been explored.

These applications will be used in upcoming year 3 demonstrator events to test-drive the made developments in platform functionality.