VerSatilE plug-and-play platform enabling remote pREdictive mainteNANce

Grant Agreement No	:	767561
Project Acronym	:	SERENA
Project Start Date	:	1st October 2017

Consortium	:	COMAU S.p.A. Finn-Power Oyj VDL Weweler BV WHIRLPOOL EMEA SpA Kone Industrial Ltd Engineering Ingegneria Informatica S.p.A. OCULAVIS GmbH SynArea Consultants S.r.l. DELL EMC Laboratory for Manufacturing Systems & Automation Fraunhofer Gesellschaft zur Förderung der angewandten Forschung VTT Technical Research Centre of Finland Ltd TRIMEK S.A. Politecnico Di Torino

Title	:	Design of AR-based remote diagnostics platform and interfaces
Reference	:	D4.1
Dissemination Level	:	PU (public)
Date	:	2018-09-30
Author/s	:	OCULAVIS/LMS/VTT
Circulation	:	EU/Consortium

Summary:

Purpose of this document is to summarize activities on the design and development of a framework for AR-based human operator support, as a result of the SERENA project activities in the first year.

Contents

List of Abbreviations

DoA	Description of action
DB	Database

List of Figures

Executive Summary

The purpose of this document is to describe the basic design of AR-based remote diagnostics platform and interfaces in SERENA project, as developed within the first 12 months of the SRENA project.

The document starts with a short introduction followed by the system design in chapter 2. Chapter 3 focuses on file and data management of work package 4 followed by design descriptions of interfaces and integrity controls in chapter 4 and 5. Finally the operational scenarios based on the different use cases are provided in chapter 6.

All contents are presented with the support of illustrated figures in order to make it easier for the reader to follow.

1 Introduction

1.1 Motivation

The overall motivation of work package 4 is the overall system design for an augmented reality-based worker guidance system and its interfaces with other SERENA systems components.

1.2 Work package objectives

This WP will focus on the development and adaptation of Augmented Reality (AR) based technologies for providing assistance to the maintenance personnel and information about the status of the machinery and overall equipment within the factory, even from distance. The design of these solutions will be based on existing technologies and previous experience of the partners (**OCULAVIS**, **VTT**, **LMS**). Following the design, the development of the interfaces will take place as well as the generation of instructions based on the different kind of repairs to be done by the human operators. The integration of this platform and the communication interface with the cloud-based platform (link to WP5) will be developed as well.

1.3 Requirements

The selection of technologies and solutions will be done based on the requirements and the pilot cases defined within WP1. The whole architecture designed here should be compatible with the cloud based platform (link to WP5). Information about repairing activities and the machinery status should be provided to the human directly given the different devices that will be available (e.g. AR glasses, tablets, smartphones etc.). The human will have access in information provided by the remote cloud and should provide feedback back to the cloud about the status of repairing activities (e.g. completed, on-going, need for assistance etc.). Different kind of interaction between human and the remote platform have been defined (videos, instructions, live chat, visual instructions etc.) will be available. **OCULAVIS** will lead this task, working close with **LMS**, **SynArea, TRIMEK** and **IPT** as well as the technology providers (**COMAU**, **Finn-Power**).

The requirements of the use case owners were gathered by a survey about content requirements, context technology requirements as well as border conditions. The results can be found in the following figure (the entries in the yes and no columns represent the percentage of survey participants that answered the corresponding questions accordingly).

	Yes	No
Border conditions strongly drive hardware selection		
Safety helmet required in your use case / demonstrator?	0	100
Safety glasses required in your use case / demonstrator?	40	60
Noise protection required in your use case / demonstrator?	40	60
Explosion hazard existing in your use case / demonstrator?	0	100
High humidity in your use case / demonstrator environment?	0	100
Aggressive aerosol particles in your use case / demonstrator environment?	60	40
Constant and good light conditions in your use case / demonstrator environment?	60	40
Content requirements drive software design		
Text content required in your use case / demonstrator?	100	0
Image content required in your use case / demonstrator?	100	0
Audio content required in your use case / demonstrator?	40	60
Video content required in your use case / demonstrator?	40	60
3D content required in your use case / demonstrator?	80	20
Machine / sensor data required in your use case / demonstrator?	60	40
eedback data from AR apps required (meta data, cycle times, documentation etc)?	60	40
Which context technologies shall be used to trigger AR manuals?		
2D codes (e.g., Barcodes, QR Codes etc.)	60	40
Bluetooth Low Energy	20	80
NFC tags	0	100
Wifi network	80	20
GPS	20	80
Event driven push of AR manuals to smart wearables?	60	40

Figure 1: System requirements of work package 4

2 "System" design

2.1 Concept

The use-case of SERENA focuses on the development of a system for AR-based step-by-step instructions to support service technicians on-site.
The service technician in the SERENA use-case uses AR-based step-by-step instructions on his daily used mobile devices (tablets, phones, glasses) to perform maintenance tasks. He is performing scheduled maintenance cases or is trained on maintenance tasks for new machines. The tasks can even be performed without internet connection having the manuals stored locally on the device. The service technician typically works for a machine manufacturer and travels to the customer. The application the service technician uses will have advanced functionalities and a complex user interface because the technician will use the software in his daily work. Challenges and innovation of the research work are the context-related and automatic generation of AR step-by-step instructions based on the machine status and the technicians' skill level. All this information will be used as a feedback loop to improve the automatic and context-based creation of step-by-step manuals. The focus of the AR application is accuracy to detect the behaviour of the technicians on-site. So, depth tracking technologies will be in the focus of this research work. The technician gets an automatic and – based on the given machine status and skill level – context-related worker assistance system.

2.2 AR based human operator support in SERENA

The high-level system design (task 4.1) of work package 4 can be found in the following figure.

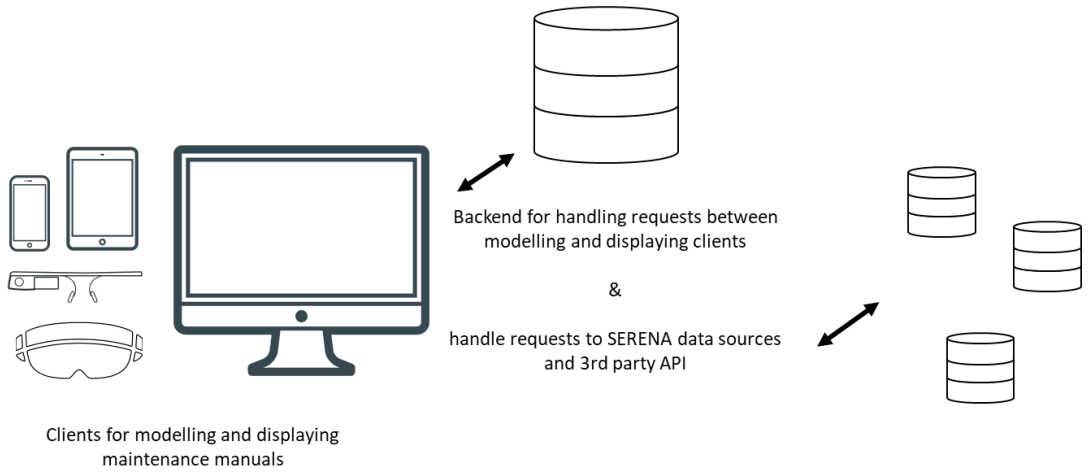


Figure 2: High level system architecture of WP4

Task 4.2 will focus on application development (backend, frontends, interfaces) that enables the AR based human operator support for in site maintenance activities of task 4.3. Integrating with other SERENA work package implementations will be subject to task 4.4 (versatile assistance platform integration).

2.3 Subsystems description

2.3.1 Maintenance instruction generator (MIG)

Name	Maintenance instruction generator – MIG
Description	The development of this component has taken place in accordance to the requirements of T4.3 of the SERENA DoA. Its purpose is the effective maintenance instruction generation towards providing onsite support to maintenance operators. As such a web application has been designed and partially developed for providing

support information on maintenance operations. A client-server architecture has been adopted aiming to support multiple client host devices such as wearables, industrial monitors, and portable devices.

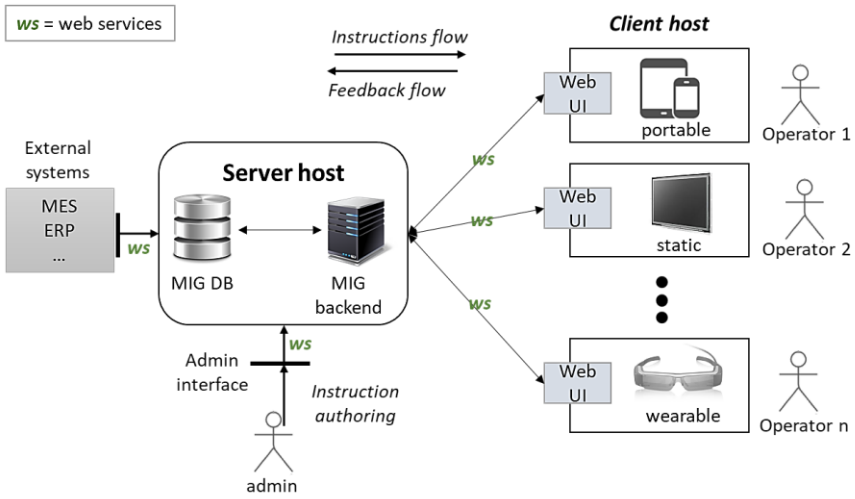


Figure 3: MIG architecture

The main functionalities of this component at the time of composing this report are as follows:

- Multi-media instruction authoring and storing
- User authorization
- Registering of devices (portable/static)
- Instruction set evaluation and selection for a given task, to a register to the user device and his/her experience level
- Registering of devices (portable/static)
- Instructions efficiency evaluation mechanism
- Operators feedback assessment engine
- Instructions enhancement based the two evaluation components and enrichment of the instruction set per operator/task/device.

Design constraints

The constraints of the systems are based on the interaction required to other systems. Hence, its I/O as well as its functionalities constitute its design constraints. As such, a high-level representation of its I/O leading to a first prototype design are presented in the following figure (Figure 4).

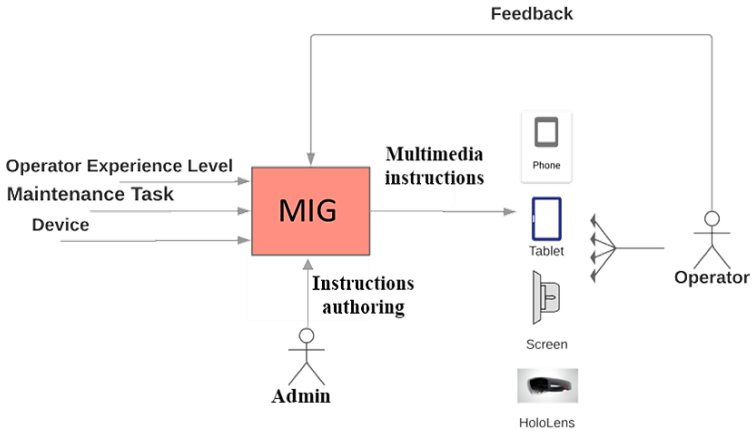


Figure 4: System I/O

	The system is expected to generate instructions for a specific maintenance activity, for an operator of a certain experience level to the device that is assigned to him/her, or another of his/her selection. As such its I/O requirements are as follows: • Input - Operator experience level, Maintenance task parameters (description, duration, location, ..) and device to display the instructions • Output - Multimedia instruction set In the future it is expected to add in a closed-loop the evaluation of the operator's feedback towards providing him/her with a more personalised support.
Hardware architecture	There is no specific hardware architecture, since the component is a web application. However, as such, its functionality depends on web services requiring the existence of networking and connectivity infrastructure.
Software architecture	The software architecture follows a client-server approach. As such the modules can be separated based on their deployment either on the server side (backend) or on the client side (frontend). The frontend includes the components and services facilitating the interaction with the user, supporting multiple language preferences. The development of the frontend has been carried out using HTML [1] , CSS [2] and AngularJS [3] . The backend has been developed using the Spring Framework supporting the key functionalities of the tool, namely: - Data management - User authorization - Instructions authoring/editing/enriching - A multi-criteria decision-making module responsible for the business logic of the component - Multichannel communication interfaces It is planned to enrich the functionalities of the existing design with the following additions: • Bot functionality for advanced interaction with the operators • Operator feedback evaluation mechanism and instruction set modification • The creation of a context engine for transforming raw data to maintenance context information, intended for more efficient operator support and more personalized interaction with the component.

2.4 Internal communications

The MIG component consists of several smaller components responsible for certain set of functionalities. As such, a high-level representation of their interaction and data exchange enabling the MIG functionality is presented in Figure 5.

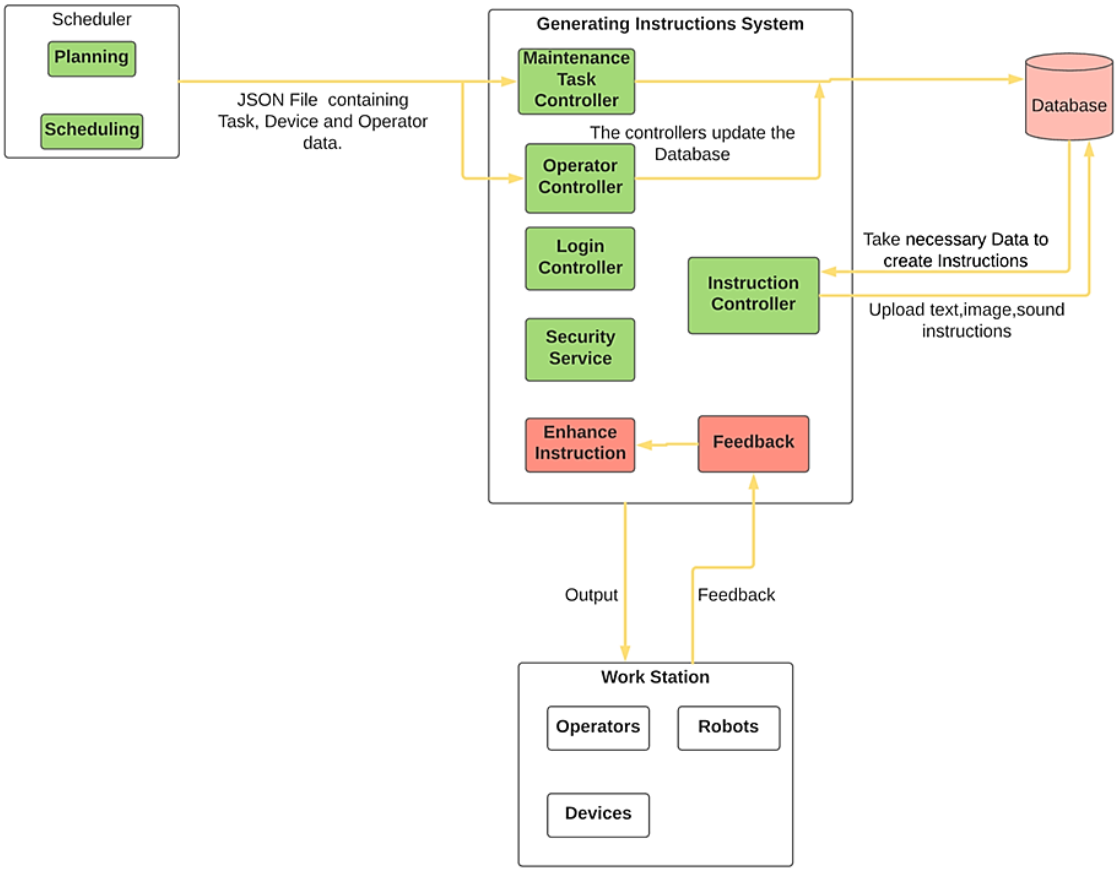


Figure 5: MIG modules and interaction

3 File and data management

3.1 Description and Purpose

The MIG component is closely couple to a database structure preserving the maintenance instruction sets. As such it supports the in-time retrieval of the stored information in runtime for onsite operator support.

3.2 Database management system files

Purpose of the MIG system is the generation of maintenance related information to support the human operators during such activities on the shopfloor. As such, the web application provides an instruction authoring interface for editing instruction sets to a database. Afterwards, stored information is transmitted to the client devices evaluating several parameters, such as the operator name, experience level, task, etc. The efficient storage and manipulation of stored instructions and content enables better operator support services in runtime. The presented web app used a MySQL database with the following schema at the moment of composing this report (Figure 6).

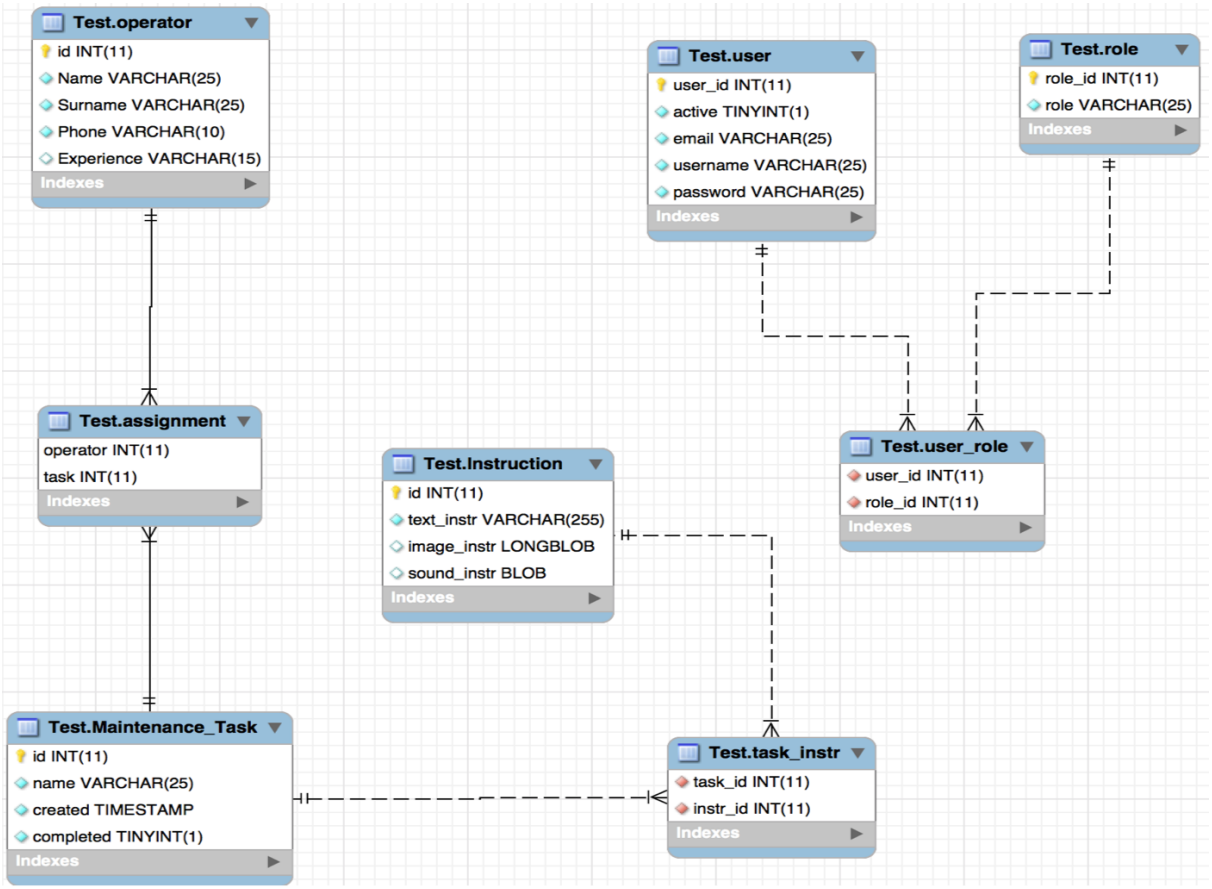


Figure 6: MIG database schema

3.3 Non-database management system files

Except for database files, the web application uses non-database files in order to generate the single page application. To this end, the AngularJS [1] functionalities are adopted.

4 System interfaces

4.1 Human – machine interfaces

4.1.1 MIG user interface

4.1.1.1 Purpose

It constitutes a user web interface accessible through a web browser. The purpose of this interface is to create multi-media instructions , correlating to experience levels and maintenance operations as well as display devices. An administrator or a maintenance expert inserts text, image, audio instruction and stored them into the database of the MIG component. There are currently 2 parallel systems in development, one from OCULAVIS and one from LMS. Within scheduled user studies the decision will be made, which system to use.

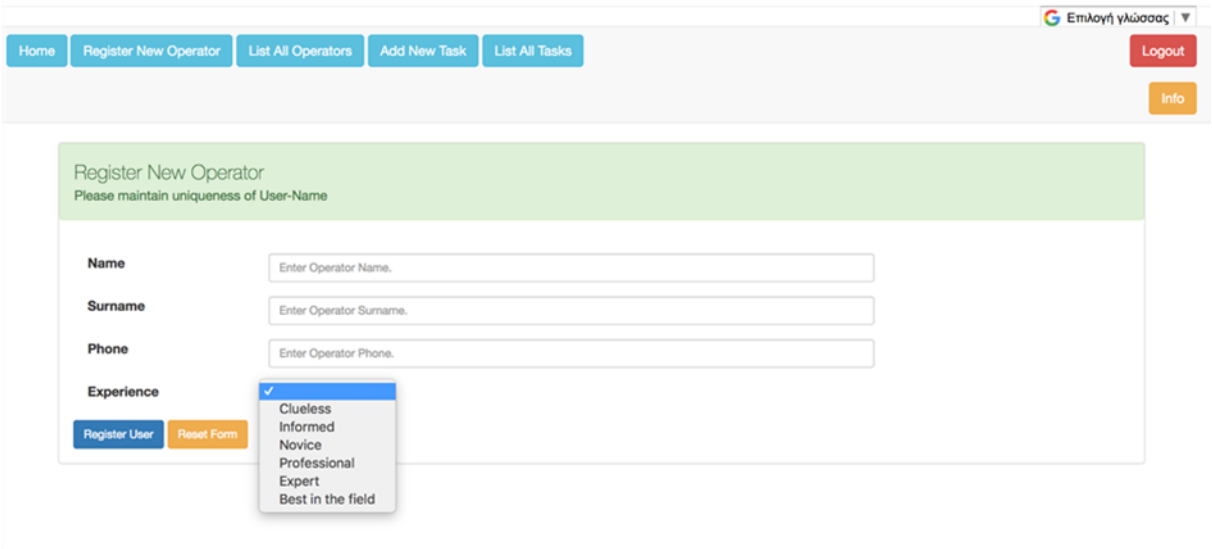


Figure 7: Instructions authoring interface (LMS)

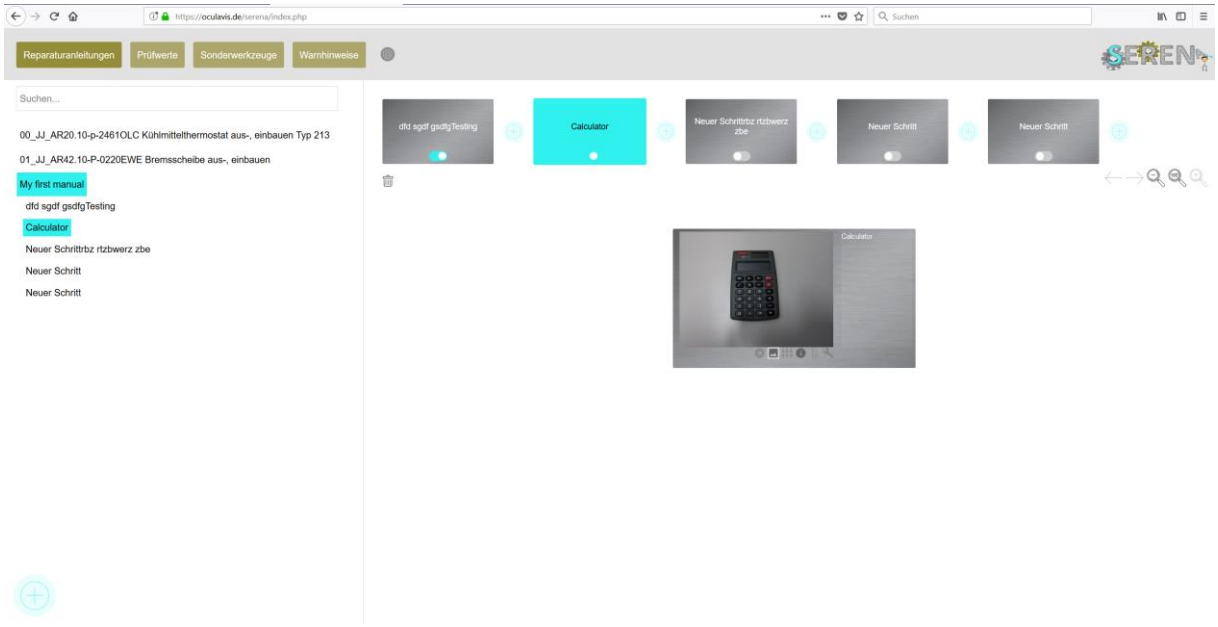


Figure 8: Instructions authoring interface (OCULAVIS)

In addition to these authoring tools there are mobile apps for smart glasses, tablets and phones under development to display the authored information to the maintenance staff (see following figure for an app flow on Google Glass).

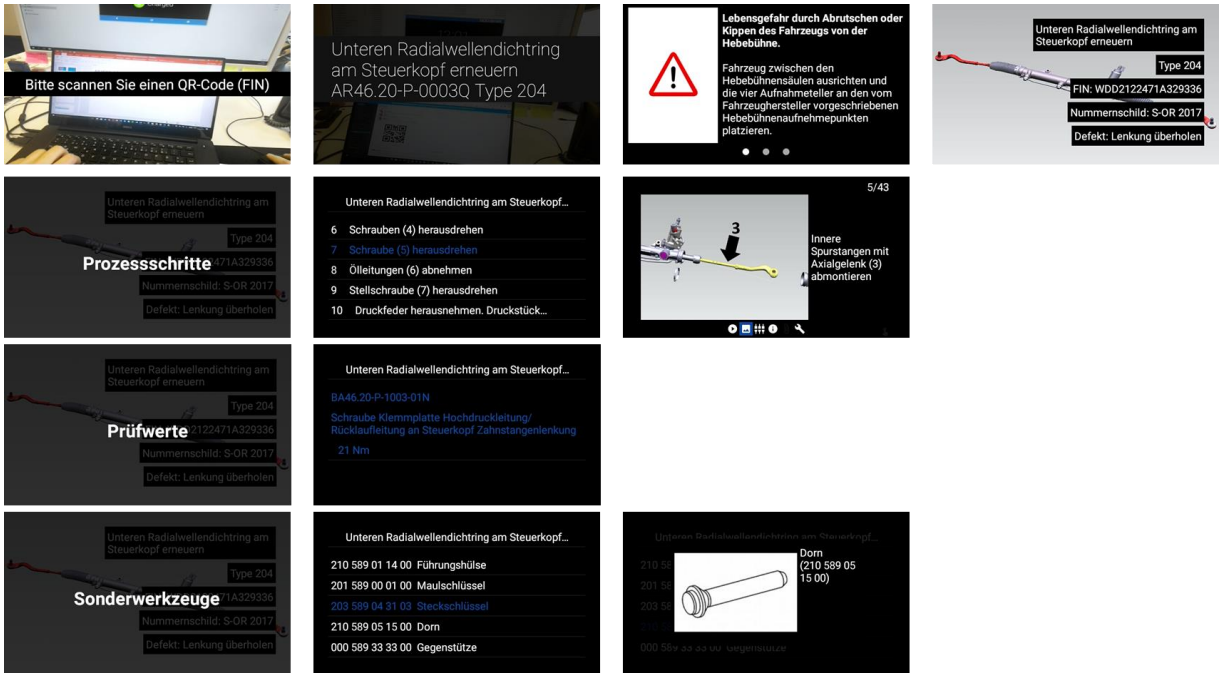


Figure 9: Instructions app flow on Google Glass

4.1.1.2 Inputs

The instruction authoring interface enables the creation of instruction sets from a maintenance expert based on (a) operator's experience level, instruction display devices and maintenance task. Then various multimedia files (sound, text, video, mix) can be associated to this instruction set by uploading the appropriate files or text cab be added through the web interface on the fly.

4.1.1.3 Outputs

The output is the outcome of the user interaction with this interface constituting one or more maintenance instruction sets. These sets are stored in the local MySQl database to support their retrieval at a later point and dispatchment to the operator's client device at runtime.

4.2 External interfaces

4.2.1 Maintenance aware scheduling tool

4.2.1.1 Purpose

The main purpose of this external interface is to retrieve maintenance task assignments during runtime for selecting the appropriate set of instructions, based on the task, experience level and associated display device. It is assumed that all planning time information at this moment will be retrieved by the scheduling tool of SERENA, to be developed in WP3. This includes information such as the complete maintenance task list, pool of operators and devices and their experience levels required for the authoring of the instruction sets. In the future, that information may be coming as well by any other legacy system.

4.2.1.2 Inputs

The parameters that are provided as inputs to the MIG components from the scheduling tool are the following:

- List of maintenance operations/tasks
- List of operators
- Operators' experience level
- List of display devices
- Device associated to the task assigned operator

4.2.1.3 Outputs

The instructions generation tool at the moment does not provide any output to the maintenance aware scheduling. In the future, there will be assessed if the execution status should be provided through this tool to the scheduling component.

5 Integrity controls

Security:
Authentication Customization Against a Relational Database and Method-Level Security from Spring Boot is used. In the back-end, the REST endpoints have been secured. In the front-end, the login and logout authentication process has been secured with CRSF cookies. It is planned for the near future to customize the security behaviour with authentication against a database so that it uses the application's users. The Method-level security secures URI access. Hence, it is envisioned to secure that only users with administrative privileges can perform a delete action.

Handle Errors in a RESTful API:
The MIG provides error handling in RESTful API.

Data validation:
The system provides an advice controller that handles all the exceptions from MethodArgumentNotValidException class for all the REST controllers. Specifically, the advice controller will validate the request body of the HTTP POST request for the length and the NULL value.

Logger:
In the future, a logger that keeps tracks of users' verification, date/time logins and database queries, will be integrated with the system.

6 Operational scenario

A high-level representation of the operational use cases for the MIG system is provided in the following figure.

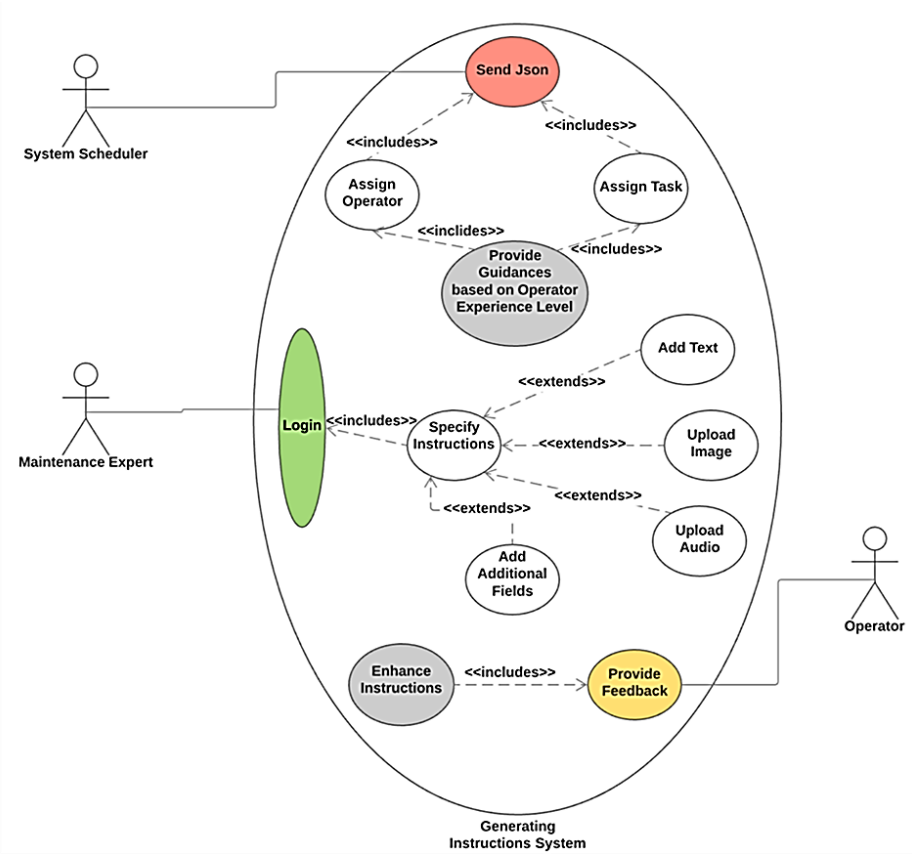
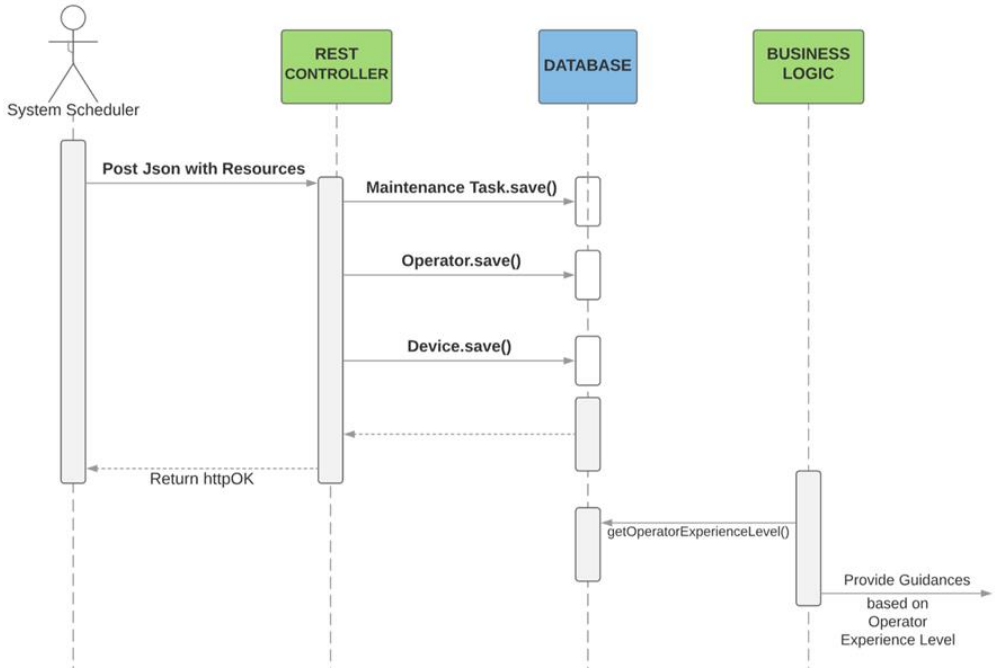


Figure 10: Instruction generation system use cases

- **Interaction with the scheduling component**



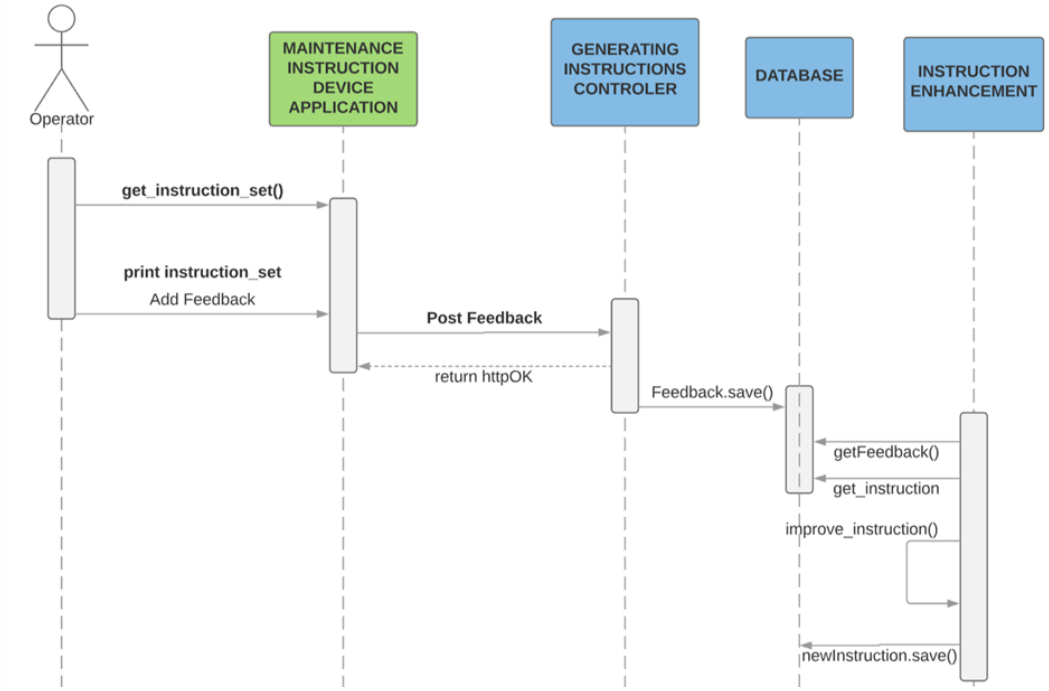
Basic Flow:

1. The scheduler sends the maintenance operations with HTTP POST in the MIG through a JSON message.
2. The Rest Controller of the MIG takes the JSON, creates objects from POJO entities, saves them in the database and returns HTTPOK.
3. The MIG takes the operator experience level and provides guidance on the maintenance expert about how to assign instructions and what types of instructions.

Alternative Flow:

1a. The system scheduler sends a wrong JSON file.
2a. The Rest Controller returns HTTPERROR.

- **Interaction with the operator**



Basic Flow:

1. The operator opens the external application to his device.
2. The operator receives an instruction.
3. The operator follows the instruction.
4. The operator adds a feedback for the instruction
5. The external application sends a JSON with the feedback with HTTP POST in the GIS.
6. The rest controller takes the feedback, saves it in the database and returns HTTP OK.
7. After the Instruction Enhance component takes the feedback from the database and improves the instruction set.

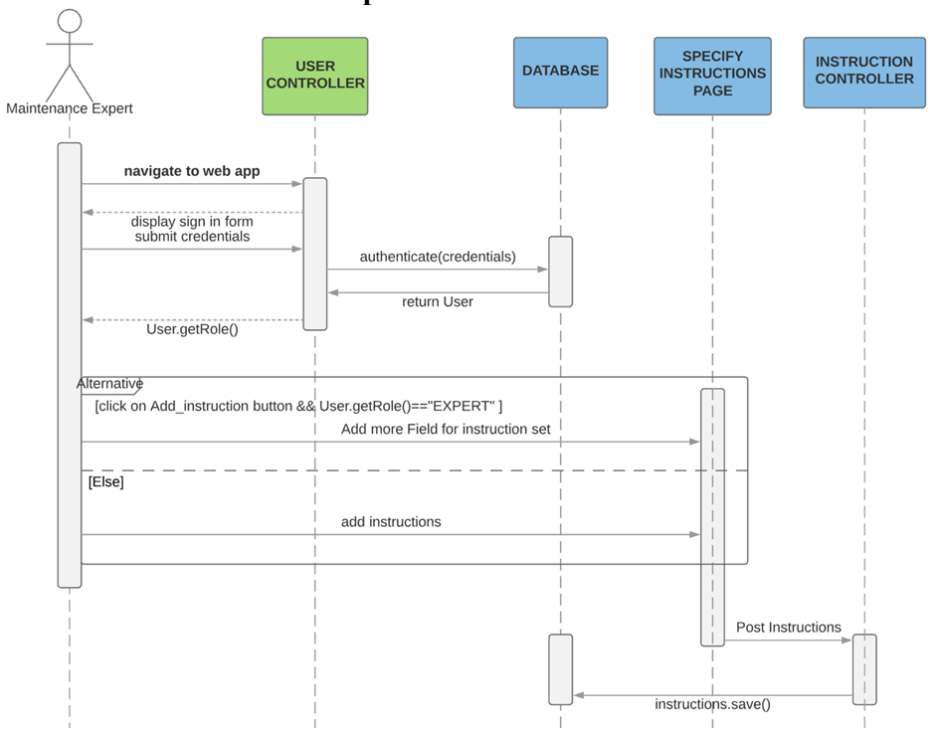
Alternative Flow 1:

7a. The instruction set doesn't need improvement because the feedback was high-grade.

Alternative Flow 2:

5a. The external application sends a wrong JSON with HTTP POST in the GIS.
6a. The Rest Controller returns HTTPERROR.

- **Interaction with the maintenance expert/admin**



Basic Flow:

1. The maintenance expert navigates to the web app.
2. The maintenance expert submits credentials in the sign in form.
3. The GIS authenticate credentials and returns the user and his privileges.
4. The maintenance expert goes to the Specify Instructions page.
5. The maintenance expert clicks the Add Instruction button to add the number of fields that the instruction set requires.
6. The maintenance expert specifies the instructions based on the guidance that has arisen from the interaction with the system scheduler.
7. The GIS saves the instruction set in the database.

Alternative Flow1:

3a. The maintenance expert submits wrong credentials.

3b. repeat step 2

7 Conclusion

The document has provided a design description of work package 4 objectives. In the next step the implementation of the different system components (backends, interfaces, clients, communication) will start respectively is on-going.

References

[1] https://www.w3.org/TR/html52/
[2] https://www.w3schools.com/w3css/
[3] https://angularjs.org/