

# INTERACT – Interactive Manual Assembly Operations for the Human-Centered Workplaces of the Future

**Grant Agreement Number** : 611007  
: INTERACT  
**Project Start Date** : 1st October, 2013  
**Consortium** : DAIMLER AG (DAIMLER)- Project Coordinator  
ELECTROLUX ITALIA S.P.A. (ELECTROLUX)  
INTRASOFT INTERNATIONAL SA (INTRASOFT)  
IMK AUTOMOTIVE GMBH (IMK)  
EMPHASIS TELEMATICS AE (EMPHASIS)  
HADATAP SP ZOO (HADATAP)  
UNIVERSITY OF PATRAS (LMS)  
UNIVERSITAET ULM (IMI)  
DEUTSCHES FORSCHUNGSZENTRUM FUER KUENSTLICHE  
INTELLIGENZ GMBH (DFKI)



**Title** : Application requirements for the human centered manual assembly workplace  
**Reference** : D1.3.1  
**Availability** : Public (PU)  
**Date** : 31/03/2014  
**Author/s** : DAIMLER, ELECTROLUX, LMS, INTRASOFT, IMK, HADATAP, DFKI  
**Circulation** : EU, INTERACT consortium

## Summary:

Description of the requirement for the INTERACT applications (apps) and the INTERACT Enterprise Application Platform (EAP). This deliverable is the main outcome of Task 1.3 “Application requirements for the human centered manual assembly workplace”.



The INTERACT project (611007) is co-funded by the European Commission under the 7th Framework Programme.

This document reflects only authors' views. The European Commission is not liable for any use that may be done of the information contained therein.

## Table of Contents

1.	EXECUTIVE SUMMARY .....	3
2.	INTRODUCTION.....	4
3.	REQUIREMENTS ENGINEERING APPROACH.....	5
4.	3D SCENE VIEWER.....	6
	4.1. Vision - Application Description .....	6
	4.1.1. Problem and solution statement.....	6
	4.1.2. Key application features .....	6
	4.2. Use cases and scenarios.....	6
	4.3. System wide requirements list.....	7
5.	SENSOR NETWORK MANAGEMENT .....	8
	5.1. Vision - Application description .....	8
	5.1.1. Problem and solution statement.....	8
	5.1.2. Key application features .....	8
	5.2. Use cases and scenarios.....	8
	5.3. System wide requirements list.....	16
6.	BASIC ANALYSIS .....	17
	6.1. Vision – Application description.....	17
	6.1.1. Problem and solution statement.....	17
	6.1.2. Key application features .....	17
	6.2. Use cases and scenarios.....	17
	6.3. System wide requirements list.....	18
7.	COMPARISON ANALYSIS .....	19
	7.1. Vision – Application description.....	19
	7.1.1. Problem and solution statement.....	19
	7.1.2. Key application features .....	19
	7.2. Use cases and scenarios.....	19
8.	PROJECT EDITOR .....	21
	8.1. Vision – Application description.....	21
	8.1.1. Problem and solution statement.....	21
	8.1.2. Key application features .....	21
	8.2. Use cases and scenarios.....	22
	8.3. System wide requirements list.....	24
9.	COLLABORATION MANAGER.....	25
	9.1. Vision - Application description .....	25
	9.1.1. Problem and solution statement.....	25
	9.1.2. Key application features .....	25
	9.2. Use cases and scenarios.....	25
10.	ERGONOMICS ASSESSMENT .....	29
	10.1. Vision – Application description.....	29
	10.1.1. Problem and solution statement.....	29

---

10.1.2. Key application features .....	29
10.2. Use cases and scenarios.....	30
10.3. System wide requirements list.....	30
11. ENTERPRISE APPLICATION PLATFORM.....	32
11.1. Vision .....	32
11.1.1. Problem and solution statement.....	32
11.1.2. Key application features .....	32
11.2. Use cases and scenarios.....	33
12. CONCLUSIONS.....	34
13. GLOSSARY.....	35

## 1. EXECUTIVE SUMMARY

The main purpose of this document is to give an overview of the applications as well as the requirements that they address.

Application requirements were gathered from the industrial partners and analyzed by technological and research partners. From these requirements, a set of applications and specific use cases for each application are defined. The use cases provide a structural manner to ensure that end user expectations will be met.

The document is structured as follows:

- Chapter 1 provides an executive summary.
- In chapter 2, the relationship of this deliverable to other deliverables is described.
- Chapter 3 provides a description of the requirements engineering approach that was followed in order to define the requirements of the apps and Enterprise Application Platform (EAP).
- From chapter 4 to chapter 10, apps and their respective functionalities and requirements are presented:
  4. 3D scene viewer
  5. Sensor network management
  6. Basic analysis
  7. Comparison analysis
  8. Project editor
  9. Collaboration Manager
  10. Ergonomics Assessment
- Chapter 11 provides the requirements of the EAP.
- Chapter 12 describes the conclusions that were drawn from the analysis of requirements.
- Chapter 13 contains the glossary used for this document.

## 2. INTRODUCTION

The objective of this document is to describe the requirements for applications that will support industrial end-users as well as the requirements for the Enterprise Application Platform (EAP). The applications, of which the requirements are described hereafter, are strongly related to the definition of the pilot cases described in D1.4.1 “Industrial pilot cases definition”.

The applications will implement specific application logic that matches the industrial practices and objectives. However, in the “background”, the INTERACT applications will utilize major INTERACT modules of which their required functionality is defined in D1.1.1 “Requirements on efficient manual assembly model generation and interaction” and D1.2.1 “Requirements on monitoring of manual assembly operations”.

Last but not least this document specifies the requirements regarding EAP. EAP will provide the framework for application deployment as well as interfaces to different major system modules. The individual applications and EAP requirements will be implemented in the context of WP5 “Applications for the human-centered workplace for manual assembly operations”.

### 3. REQUIREMENTS ENGINEERING APPROACH

The overall process of defining the requirements will be based on the open source methodology called OpenUP (Open Unified Process)<sup>1</sup> part of the Eclipse Process Framework (EPF) which is in alignment with the *agile development aspects*; this is further discussed in subsequent paragraphs. The idea is to not use all the features of the methodology, but having regard to its principles, particularly the definition of the following key artifacts and the roles involved in identifying and outlining the requirements for INTERACT apps and EAP. The OpenUP approach drives the development based on use cases and scenarios, risk management and architecture-centric approach. This helps significantly to the integration in projects where not all requirements are defined at the beginning and some functionality are identified during the development of the solution. Thus the main reason to select the OpenUP methodology is because this methodology offers an iterative software development process that will be useful and a good approach for the INTERACT iterative approach, at the late stages in which there are concurrent demonstration (at the end user site) and development activities (at the developers site). In the requirements phase the following artifacts should be defined for each application and the EAP:

- **Vision:** Describes the view of the industrial end-users of the application to be developed. This definition is specified in terms of motivation, the key needs and features of the end-users. The vision contains an outline of the envisioned core requirements for the system.
- **Use cases and scenarios:** A use case describes what the system must do to provide value to the end-users. Define a main scenario and alternative flow of the scenario. Each use case should be accompanied with a main scenario (a scenario is an instantiation of the use case) and potential alternative scenarios.
- **System wide requirements list:** This contains functional and non-functional requirements for the system that does not apply to a specific use case.

---

<sup>1</sup> <http://epf.eclipse.org/wikis/openup/>

## 4. 3D SCENE VIEWER

### 4.1. Vision - Application Description

#### 4.1.1. Problem and solution statement

The problem of	Visualizing the dynamic scene of the 3D model of the production line or the warehouse of the two use cases. The problem is hard because the models are potentially large and parts of the scene consist of product data which might be complex and modelled with great detail.
affects	The visualization app is foreseen for the people involved on the planning process of the production setup as well as for the people validating the result of the planning process.
the impact of which is	A high quality visualization app has a high impact on the efficiency of the planning and validation process.
a successful solution would be	An easy to use app based on standards that can visualize large 3D dynamic models in an interactive and possibly collaborative manner.

#### 4.1.2. Key application features

Need	Priority	Features
Sophisticated rendering	10	Accurate, efficient
Navigation (zoom, scroll, camera positioning)	10	User should be able to navigate intuitively through the scene
Interaction (delete, add)	7	User should be able to add and delete object in the scene dynamically
Animation	10	It must be possible to animate the scene, e.g. human model and assembly parts
Standards	8	Able to deal with established standards
Collaboration	3	Synchronized presentation of models across multiple clients
Annotation	2	Annotation of scene must be possible
Mobile access	1	Run on different devices: smart phones, phablets <sup>2</sup> , tablets, PCs etc.

### 4.2. Use cases and scenarios

The following modes of operation are expected:

1. *Navigation*: The user is able to navigate the scene in an intuitive manner which allows them to inspect every detail of the 3D scene. This includes “walking” around the scene, rotating

<sup>2</sup> <http://en.wikipedia.org/wiki/Phablet>, A phablet is a class of mobile device designed to combine or straddle the functions of a smartphone and tablet.

around a selected object or the coordinate system, zooming in and out, placing and rotating a “camera position” at will.

2. *Interaction*: We expect standard interactions like for example adding or deleting objects in the scene. Furthermore, it should be possible to implement application specific interactions for the user. For example it should be possible to define an interaction procedure to open a door.
3. *Collaboration*: Several users should be able to view and navigate through a scene cooperatively, i.e. users can view and navigate through the scene while using different computer devices which might be geographically distributed. Additionally, changes to the scene by one user should be synchronized with the scene of the other users. Also animations should be synchronized across connected viewers.

Three different application use cases are envisaged:

1. In the planning phase of a new production line or warehouse process, a planning expert checks a virtual setup for the new production line or warehouse process.
2. In an extension of use case 1: A team does the evaluation where the team members are potentially geographically distributed.
3. The visualization app is used in a workshop for the evaluation of the production line or warehouse process in a physical environment. The visualization app can run on a work station where the visualization is displayed on a large screen or on handheld devices of the different participants.

### 4.3. System wide requirements list

Usability	The user interface of the 3D scene viewer should be intuitive and easy to use.
Performance	The user should be able to navigate and interact with reasonably large scenes without any significant problems, e.g. the viewer hangs.
Consistency	The results of interactions should correspond to common sense knowledge about the action performed. CAD UI standards can be used, but production planners are neither CAD affine nor have the expertise, so other interaction paradigms are worth a try
Personalization	The user should be able to personalize the look and feel, e.g. by selecting themes or more importantly personalizing menus with the most used/relevant functions for their work



## 5. SENSOR NETWORK MANAGEMENT

### 5.1. Vision - Application description

#### 5.1.1. Problem and solution statement

The problem of	Many different sensors (MEMS and optical) will be applied in many different possible configurations and forms.
affects	Employees installing, configuring, managing and maintaining the whole sensor system.
the impact of which is	Proper, easily reconfigurable and maintainable configuration of sensor platform is mandatory for successful platform operation.
a successful solution would be	An application readily available in the shop-floor/warehouse environment that will enable the INTERACT sensor system installation, configuration, reconfiguration and management, as well as “raw” visualization of sensor data for installation testing purposes.

#### 5.1.2. Key application features

Need	Priority	Features
Deployment speed	10	Intuitive, coherent and concise user interface, providing relevant default or previously saved configuration if possible
Unification	10	Access to different sensors configuration from one standardized user interface
Visualization	8	Simple visualization of configured parameters and status of sensors (devices)
Mobile access	1	Run on different devices: smart phones, phablets, tablets, PCs etc.
Privacy	10	Sensor data captured from real working environment should be protected with regards to privacy (e.g. consider deleting them after been processed and used for motion generation purposes)

### 5.2. Use cases and scenarios

#### Use-Case 1: Detect available sensors

##### 1. Brief Description

Allows to search for physically installed INTERACT sensors available through platform network.

##### 2. Actor Brief Descriptions

- a. Servicing staff

##### 3. Preconditions

User has to be authenticated and have sufficient permissions.

##### 4. Basic Flow of Events

1. User clicks “Sensors list” button.
2. App displays “Sensors list” screen with two tables: “Currently used sensors” and “Sensors available on the network”, which contain sensor’s ID, address, type, name, etc.
3. User clicks “Refresh”.

4. Sensor Management Platform performs a search in the network for all accessible and serviceable sensors and “Sensors available on the network” table is updated on the screen.
5. The use case ends.

## 5. Alternative Flows

-

## 6. Subflows

-

## 7. Post-conditions

-

# Use-Case 2: Assign a sensor

## 1. Brief Description

Allows to assign a sensor for use in underlying platform algorithms.

## 2. Actor Brief Descriptions

- a. Servicing staff

## 3. Preconditions

User has to be authenticated and have sufficient permissions.

## 4. Basic Flow of Events

1. User clicks „Sensors list” button.
2. App displays “Sensors list” screen with two tables: “Currently used sensors” and “Sensors available on the network”.
3. User chooses a sensor from “Sensors available on the network” and clicks „Assign for use”.
4. Specified sensor is assigned for use in the system with default parameters and it is indicated on the screen by moving the entry from “Sensors available on the network” to “Currently used sensors”. *Editing/reconfiguration of default parameters is possible later by the „Configure sensor” use case.*
5. The use case ends.

## 5. Alternative Flows

- a. **No feasible default parameters for sensor**

If in step 4 of the basic flow it is not possible to determine default parameters, then

1. “Configure sensor” use case is automatically launched for given sensor.
2. The use case ends.

## 6. Subflows

-

## 7. Post-conditions

-

# Use-Case 3: Configure sensor

## 1. Brief Description

Allows to edit configuration parameters of a sensor, regarding its use in underlying platform algorithms.

## 2. Actor Brief Descriptions

### a. Servicing staff

## 3. Preconditions

User has to be authenticated and have sufficient permissions.

## 4. Basic Flow of Events

1. User clicks „Sensors list” button.
2. App displays “Sensors list” screen with two tables: “Currently used sensors” and “Sensors available on the network”.
3. User chooses a sensor from either list and clicks „Configure sensor”.
4. A table is displayed, containing all parameters relevant to the given sensor.
5. User edits chosen parameters. *Sensors of different types may have different parameter editors, which are specified below in “Subflows” section.*
6. The use case ends.

## 5. Alternative Flows

-

## 6. Subflows

### a. Assignment of sensor function

This subflow is relevant if there is more than one possible usage of the given sensor type by underlying platform algorithms, and a parameter must be configured to determine that.

1. User chooses to modify sensor function parameter.
2. A list of possible, pre-defined functions for a given type of sensor is displayed.
3. User selects desired sensor function.
4. User confirms parameter setting by clicking “OK”.

### b. Configuration of MEMS placement on human body (pre-defined lists)

This subflow is relevant if there is a parameter for a given sensor that specifies its placement on tracked user’s body, and the user wants to select it from pre-defined lists.

1. User chooses to define sensor position by selecting from pre-defined list.
2. A list of possible, pre-defined placement positions and orientations is displayed.
3. User selects desired sensor placement and orientation. Possible positions are visualized on an avatar for convenience.
4. User confirms parameter setting by clicking “OK”.

### c. Configuration of MEMS placement on human body (3D editor)

This subflow is relevant if there is a parameter for a given sensor that specifies its placement on tracked user’s body, and the user wants to define it by using 3D editor.

1. User chooses to define sensor position by 3D editor.
2. A 3D editor depicting an avatar is displayed.
3. User points to specific position on the avatar using his pointing device (e.g. mouse or touch screen).
4. System visualizes chosen placement position with default spatial orientation vectors on the avatar.
5. User adjusts orientation by dragging the vectors.
6. User confirms parameter setting by clicking “OK”.

### d. Configuration of simple numerical and textual parameters

This subflow is relevant if there are parameters that are simple numerical values or text fields.

1. User chooses to edit given textual or numerical parameter.
2. Editable input field containing current parameter value is displayed. In case of numerical parameters, unit of measure is also displayed. If numerical parameter has upper and lower bounds, they are also displayed along with a slider control for easy editing without using keyboard.

3. User inputs desired value using either keyboard or in some cases his pointing device (e.g. mouse or touch screen).
4. User confirms parameter setting by clicking “OK”.

**e. Configuration of parameters that are 2D or 3D vectors**

This subflow is relevant if there are parameters that are best represented as 2D or 3D vectors.

1. User chooses to edit given vector parameter.
2. Simple 2D or 3D editor is displayed, containing visualization of current parameter value.
3. User inputs desired coordinates of a vector in input fields using keyboard (visualization is immediately updated), or edits the vector directly on screen by using his pointing device (e.g. mouse or touch screen).
4. User confirms parameter setting by clicking “OK”.

**7. Post-conditions**

-

## Use-Case 4: Sensor calibration

**1. Brief Description**

Allows to calibrate sensors that need periodical, manual calibration (e.g. MEMS that loose precision over time) or one-time calibration (e.g. cameras after moving to another room).

**2. Actor Brief Descriptions**

- a. Servicing staff**

**3. Preconditions**

User has to be authenticated and have sufficient permissions.

**4. Basic Flow of Events**

1. User clicks „Sensors list” button.
2. App displays “Sensors list” screen with two tables: “Currently used sensors” and “Sensors available on the network”.
3. User chooses a sensor from either list and clicks „Calibrate sensor”.
4. App displays instructions for the current step of calibration (e.g. what to do physically with the sensor and when to click “Next”).
5. User approaches the sensor and executes displayed instructions (e.g. shows special visual marker to camera, levels a gyroscope, etc.).
6. When done, user clicks “Next”.
7. If further manual calibration steps are necessary, use case flow goes back to step 4.
8. The use case ends.

**5. Alternative Flows**

-

**6. Subflows**

-

**7. Post-conditions**

-

## Use-Case 5: Monitoring

**1. Brief Description**

Allows to view raw data stream from a sensor.

## 2. Actor Brief Descriptions

### a. Servicing staff

## 3. Preconditions

User has to be authenticated and have sufficient permissions.

## 4. Basic Flow of Events

1. User clicks „Sensors list” button.
2. App displays “Sensors list” screen with two tables: “Currently used sensors” and “Sensors available on the network”.
3. User chooses a sensor from either list and clicks „Receive data”.
4. App displays real-time stream of data from the sensor, which shows if the sensor is working or not. Different visualization modes for different sensor types are possible:
  - the simplest would be an endless text string with raw values (binary, hex, decimal),
  - more sophisticated (but maybe not necessary) are:
    - 2D graph,
    - 3D vector visualization,
    - or visualization of the whole avatar.
5. User clicks “Close”.
6. The use case ends.

## 5. Alternative Flows

-

## 6. Subflows

-

## 7. Post-conditions

-

# Use-Case 6: Save platform configuration profile

## 1. Brief Description

Allows to save current configuration profile of all sensors for later re-use.

## 2. Actor Brief Descriptions

### a. Servicing staff

## 3. Preconditions

User has to be authenticated and have sufficient permissions.

## 4. Basic Flow of Events

1. User clicks „Save configuration profile” button.
2. App asks for a name, destination folder and optional description for the new configuration profile.
3. User enters name and description.
4. User clicks “Save”.
5. System stores the configuration profile.
6. The use case ends.

## 5. Alternative Flows

### a. Configuration profile already exists

If in step 4 of the basic flow it turns out that a configuration profile with given name already exists, then

1. App asks if user wants to overwrite existing profile.

2. Depending on user choice, the use case resumes at step 5, or 2.

## 6. Subflows

-

## 7. Post-conditions

-

# Use-Case 7: Load platform configuration profile

## 1. Brief Description

Allows to load previously saved configuration profiles and set up parameters of all sensors accordingly.

## 2. Actor Brief Descriptions

- a. Servicing staff

## 3. Preconditions

User has to be authenticated and have sufficient permissions.

## 4. Basic Flow of Events

1. User clicks „Load configuration profile” button.
2. App displays list of all available (previously saved) configuration profiles.
3. User selects profile.
4. User clicks “Load”.
5. System loads the configuration profile, and sets all sensors’ parameters accordingly.
6. The use case ends.

## 5. Alternative Flows

- a. Different sensors available on the network

If in step 4 of the basic flow it turns out that the set of sensors currently available on the network is different than the set for which the configuration profile was saved, then

1. App asks if user wants to load settings only into the subset of sensors that are common between the two sets. App displays detailed list of available and unavailable sensors.
2. If user chooses to cancel, the use case ends.
3. If user chooses to load the profile anyway, system partially loads the configuration profile, and sets parameters only for the sensors that are available at the time.
4. App asks if user wants to set parameters for the rest of sensors available on the network.
5. If user chooses to set parameters for the rest of sensors available on the network, list of those sensors and checkboxes is displayed.
6. Use-case 3 is automatically launched and repeated for all chosen sensors.
7. The use case ends.

## 6. Subflows

-

## 7. Post-conditions

-

# Use-Case 8: Save individual sensor’s configuration

## 1. Brief Description

Allows to save current configuration profile of a single sensor for later re-use.

## 2. Actor Brief Descriptions

- a. **Servicing staff**

## 3. Preconditions

User has to be authenticated and have sufficient permissions.

## 4. Basic Flow of Events

1. User clicks „Sensors list” button.
2. App displays “Sensors list” screen with two tables: “Currently used sensors” and “Sensors available on the network”.
3. User chooses a sensor from either list and clicks „Save configuration profile” button.
4. App asks for a name, destination folder and optional description for the new configuration profile.
5. User enters name and description.
6. User clicks “Save”.
7. System stores the configuration profile.
8. The use case ends.

## 5. Alternative Flows

- a. **Configuration profile already exists**

If in step 4 of the basic flow it turns out that a single sensor configuration profile with given name already exists, then

1. App asks if user wants to overwrite existing profile.
2. Depending on user choice, the use case resumes at step 7, or 4.

## 6. Subflows

-

## 7. Post-conditions

-

# Use-Case 9: Load individual sensor’s configuration

## 1. Brief Description

Allows to load previously saved configuration profiles of a single sensor and set up its parameters accordingly.

## 2. Actor Brief Descriptions

- a. **Servicing staff**

## 3. Preconditions

User has to be authenticated and have sufficient permissions.

## 4. Basic Flow of Events

1. User clicks „Sensors list” button.
2. App displays “Sensors list” screen with two tables: “Currently used sensors” and “Sensors available on the network”.
3. User chooses a sensor from either list and clicks „Load configuration profile” button.
4. App displays list of all available (previously saved) single sensor configuration profiles for a given type of sensor.
5. User selects profile.
6. User clicks “Load”.
7. System loads the configuration profile, and sets all sensor’s parameters accordingly.
8. The use case ends.

## 5. Alternative Flows

-

## 6. Subflows

-

## 7. Post-conditions

-

# Use-Case 10: Multi-sensor calibration

## 1. Brief Description

Allows to calibrate groups of related sensors. Specifically, this includes calibration of the system as a whole, if needed.

## 2. Actor Brief Descriptions

### a. Servicing staff

## 3. Preconditions

User has to be authenticated and have sufficient permissions.

## 4. Basic Flow of Events

1. User clicks „System-wide calibration” button.
2. App displays possible calibration options, which comprise of pre-defined sensor groups (“all sensors” possibly being one of them).
3. User chooses an option and clicks „Calibrate”.
4. App displays instructions for the current step of calibration (e.g. what to do in regard to physical interaction with the given sensor group and when to click “Next”).
5. User approaches specified physical location and executes displayed instructions (e.g. shows special visual marker simultaneously to a set of related cameras, positions body movement sensors in a way that the subject is simultaneously visible to cameras, etc.).
6. When done, user clicks “Next”.
7. If further manual calibration steps are necessary, use case flow goes back to step 4.
8. The use case ends.

## 5. Alternative Flows

### a. Individual sensors were not calibrated first

If in step 3 of the basic flow it turns out that for the chosen sensor group a necessary, individual calibration of single sensors was not performed first, then

1. App displays a warning saying that for the chosen group, individual sensor calibration should be done prior to system-wide calibration.
2. Depending on user choice, the use case resumes at step 4, ends, or a chain of “Use-Case 4: Sensor calibration” use cases is performed prior to resuming at step 4.

## 6. Subflows

-

## 7. Post-conditions

-



## Use-Case 11: System-wide configuration

### 1. Brief Description

Allows editing system-wide configuration parameters of the underlying platform algorithms (e.g. sensor fusion algorithm).

### 2. Actor Brief Descriptions

#### a. Servicing staff

### 3. Preconditions

User has to be authenticated and have sufficient permissions.

### 4. Basic Flow of Events

1. User clicks „System-wide configuration options” button.
2. A table is displayed, containing all general, system-wide parameters (not tied to any particular sensor).
3. User edits chosen parameters. *Parameters of different types may have different editors, which will be much like those specified in “Subflows” section of Use-Case 3 above. The actual fusion configuration options depend on the sensor systems implementation and will be defined along WP3 development.*
4. The use case ends.

### 5. Alternative Flows

-

### 6. Subflows

-

### 7. Post-conditions

-

## 5.3. System wide requirements list

Detailed, measurable parameters regarding qualities such as reliability and performance can be provided after needed analysis of requirements for tracking of human motion and deciding on specific sensing technologies and overall architecture that will be used.

Special attention should be put on performance of operations such as sensor network discovery or uploading/downloading of configuration parameters to and from sensors (for example all at once when loading/saving configuration profiles for the whole platform). From user experience point of view, these operations should be performed in reasonably short time by the underlying platform.

Most probably, sensor network will be built on readily available wireless and wired transmission protocols (Ethernet, Wi-Fi, Bluetooth, ZigBee, DASH7, etc.) and sensor discovery will use existing broadcast mechanisms of those protocols.

## 6. BASIC ANALYSIS

### 6.1. Vision – Application description

#### 6.1.1. Problem and solution statement

The basic analysis application is required for various reasons. Since the software, which is going to be developed in the project is not only for visualization and ergonomic assessment, but also for performance, time and space evaluation and optimization, all features regarding these issues are combined in this application.

The problem of	Analysis of: <ul style="list-style-type: none"> <li>- Times (tasks, cycles, process)</li> <li>- Space requirements (work station, working tasks, walking)</li> <li>- Walking paths (spaghetti chart)</li> <li>- Collision risk</li> </ul>
a successful solution would be	An application, which is capable of assessing, displaying and documenting all relevant parameters of the required basic analysis as described above.

#### 6.1.2. Key application features

Need	Priority	Features
Performance measures of <ul style="list-style-type: none"> <li>- Cycle time</li> <li>- Time of adding value</li> <li>- Process times (tool usage, etc.)</li> </ul>	10	<ul style="list-style-type: none"> <li>- Evaluation</li> <li>- Visualization</li> <li>- Documentation</li> </ul>
3D Space requirement of <ul style="list-style-type: none"> <li>- Working space</li> <li>- Work place</li> <li>- Material supply areas</li> <li>- Walking paths</li> </ul>	6	<ul style="list-style-type: none"> <li>- Evaluation</li> <li>- Visualization including spaghetti chart</li> <li>- Documentation</li> </ul>
Collision risk	3	<ul style="list-style-type: none"> <li>- Detection</li> <li>- Visualization</li> <li>- Documentation</li> </ul>

### 6.2. Use cases and scenarios

#### 1 Assessment

The basic analysis app allows the ‘classic’ assessment of existing work places for evaluation, documentation and performance measures.

## **2 Comparison**

The second use case would be the comparison of different simulation alternatives in a workshop. The parallel display of different solutions requires an intuitive way of displaying the time and collision related information.

## **3 Coaching**

The third use case contains the basic analysis app as coaching tool on the shop floor level, where it could be used to display performance deficits of the worker and showing alternative ways of performing certain tasks.

### **6.3. System wide requirements list**

The requirements of the basic analysis application on the rest of the system lie in the access to the 3D environment for collision warning and 3D space assessment. Furthermore the motion synthesis has to transfer times regarding to the single tasks and the metadata about the type of process and the added value. The spaghetti chart requires the walking paths during the process.

Next to these requirement, the application requires an own library including parameters for the evaluation of space minima, safety distances to objects for walking, object handling and head movement.

## 7. COMPARISON ANALYSIS

### 7.1. Vision – Application description

This application will take as input two different motion scenes and will be able to analyze them and provide a delta based on existing standards of MTM analyses.

#### 7.1.1. Problem and solution statement

The problem of	Comparing a motion scene with another one and provide a list of differences between them.
Affects	The way that different motion scenes are compared and evaluated.
the impact of which is	Efficiency in comparing alternative motion scenes.
a successful solution would be	A system that associates simulated scenarios with MTM 1 bundle / C-values scenarios and compares them. Interactive comparison of planned MTM-1 bundles / C-values to the ones derived from the simulation as well as recognition and annotation of DHM basic operations (i.e. screwing, lifting etc.)

#### 7.1.2. Key application features

Need	Priority	Features
Make an MTM-1 or C-Value comparison between two different motion scenes.	10	Open and compare two different synthesized motions. Provide a list of delta in terms of time as well as operations between these two scenes. Synthesized motions that can result from constraints that are derived from sensor data from the EAP constraint generation service.
Ergonomic evaluation comparison	10	Open and compare two alternative process and provide a comparison of ergonomics analysis
Scene annotation	10	Annotate the motion scene by displaying in human readable format the current operation.

### 7.2. Use cases and scenarios

#### Use-Case 1: Compare two scenes

##### 1. Brief Description

This use case demonstrates the procedure to be followed in order to compare two different scenes as well as the output of the comparison.

##### 2. Actor Brief Descriptions

- a. **User.** Any end-user

##### 3. Preconditions

Download and install the application.

#### 4. Basic Flow of Events

1. The use case begins when user, opens the “Comparison Analysis” app.
2. “Comparison Analysis” provides a Project browser dialog showing all the projects the user created or has access to edit. User selects the desired one and clicks “Open” button.
3. A Project Variant browser dialog is displayed showing all the project’s variants and prompts the user to select the first variant to be compared. User selects the desired variant and clicks “Compare” button.
4. A Project Variant browser dialog is displayed again showing all the project’s variants and prompts the user to select the second variant to be compared. User selects the desired variant and clicks “Compare” button.
5. Two instances of “3D Scene Viewer” app open and relevant data are displayed inside.
6. A list of MTM-1 /C-values / building blocks differences occurs.
7. A single playback control is displayed. With this control the user can replay the two scenes simultaneously. An overlay of both alternatives would be helpful, i.e. two worker animations are superimposed to visualize differences.
8. User presses “Play” button on the playback control.
9. The two scenes start playback annotating the DHM actions.
10. The MTM delta list highlights the changes during playback.

## 8. PROJECT EDITOR

### 8.1. Vision – Application description

#### 8.1.1. Problem and solution statement

The problem of	<p>Quickly change a desired scene to reflect an alternative task execution. The process verification workshop starts with the resulting auto-generated model. In the workshop, the following changes may be required to made upon the simulation model</p> <ul style="list-style-type: none"> <li>• The order of tasks is changed</li> <li>• Tasks are modified by adding or deleting worker tasks</li> <li>• Tasks are modified by changing / adapting worker tasks</li> <li>• The (collision) environment is changed (e.g. storage bin positions, tools, plant layout)</li> <li>• The worker model is changed (e.g. anthropometric parameters such as height)</li> </ul>
Affects	The way a scene is manipulated.
the impact of which is	Considerable modelling time spent along with cost
a successful solution would be	<p>An application that enables the user to make model changes and create work task variants quickly. This will be supported by providing alternative means for scene manipulation:</p> <ul style="list-style-type: none"> <li>• Manipulation of a scene from captured data derived from the shopfloor.</li> <li>• Manipulation of a scene by Controlled Natural Language (CNL) editor.</li> </ul> <p>The application will be able to validate user input and identify problems (3D environment vs. process/work task). The outcome of this application will be a full 3D motion scene.</p>

#### 8.1.2. Key application features

Need	Priority	Features
Project management system.	10	Support of CRUD (Create, Read, Update, Delete) operations on projects.
Easy manipulation of a 3D scene.	10	<p>Data Model editors</p> <ul style="list-style-type: none"> <li>• 3D Scene editor (Product / Resource)</li> <li>• Interaction using CNL</li> <li>• Process/Work Task editor (Natural language support)</li> <li>• Model editing from sensor captured data</li> </ul>
Validation of scene data	10	Input data validation (3D environment vs. process/work task)
3D Motion Scene construction	10	3D scene construction from the data models (Product, Resources, CNL,

		captured data)
Management of variants	10	Project variant management (basic version control functionalities)
Object position editor	10	Define object position states e.g. open/close for doors, drawers, boxes Define paths followed by objects (e.g. lids, doors) when object position-state changes

## 8.2. Use cases and scenarios

### Use-Case 1: Project Management

#### 1. Brief Description

This use case demonstrates how to create, update save and delete a project as well as how the projects gets validated. Also demonstrates how the 3D environment can be edited.

#### 2. Actor Brief Descriptions

- a. **User. Any end-user**

#### 3. Preconditions

Download and install the application.

#### 4. Basic Flow of Events

1. The use case begins when user, opens the "Project Editor".
2. "Project Editor" provides a Project browser dialog showing all the projects the user created or has access to edit. User does not select any project but clicks on the "Create" button. "Open" button remains disabled as long as the user does not select an existing project. "Delete" button remains disabled as long as the user does not select an existing project.
3. A dialog prompts the user to provide a Project name. User types a project name and clicks on "Create".
4. Project Editor's desktop appears.
5. User navigates on "File->Import->Geometries" to insert the 3D environment (scene). A file browser opens that is used by the user to navigate to 3D data physical file. He selects it and presses the "Import" button.
6. The "3D Scene Viewer" app open and relevant data are displayed inside.
7. User repeats step 5 for inserting product, resources (tools, containers etc.) and DHM in the 3D scene.
8. User moves the various components imported inside the scene by using the "3D Scene Viewer" functionalities.
9. User navigates on "File->Import->Process/Work Task" to insert the process. A file browser opens that is used by the user to navigate to "Process/Work Task" physical file. He selects it and presses the "Import" button.
10. The user navigates to "File->Save".
11. A dialog box is displayed that "Project is not validated. Save the project anyway?". User clicks on "Save".
12. The use case ends.

#### 5. Alternative Flows

- a. **Open a Project**

If in step 2 of the basic flow the user selects a Project and clicks on "Open", then

1. A Project Variant browser dialog is displayed showing all the project's variants and prompts the user to select one. User selects the desired variant and clicks "Open" button.
2. The "3D Scene Viewer" app opens and relevant data are displayed inside.
3. The use case resumes at step 6

**b. Delete a Project**

If in step 2 of the basic flow the user selects a Project and clicks on "Delete", then

1. A dialog box is displayed that "Project will be permanent deleted. Are you sure?". User clicks on "Yes".
2. The use case resumes at step 2

**c. Validate a Project**

If in step 11 of the basic flow the user clicks on "Cancel", then

1. User clicks on "File->Validate".
2. An "Error Console" opens with the list of errors (i.e. A tool specified by the Work Task is not imported on the Project).
3. User performs corrective actions (i.e. imports the tool that is used by the Work Task or change the Work Task to use another existing tool)
4. User clicks on "File->Validate".
5. The "Error Console" list is updated informing the user that there are no errors.
6. The user navigates to "File->Save".
7. A dialog box is displayed that "Project is validated. Save?". User clicks on "Save".
8. The use case ends.

## Use-Case 2: Process/Work Task Editor

### 1. Brief Description

This use case demonstrates the Process/Work Task editing with the use of CNL.

### 2. Actor Brief Descriptions

**a. User. Any end-user**

### 3. Preconditions

Project Management app is open.

### 4. Basic Flow of Events

1. The use case begins when user, loads/creates a project on the "Project Management" app.
2. User navigates to "Tools-> Process/Work Task editor".
3. "Process/Work Task" is already imported.
4. A panel opens containing a selectable/movable list of Work Tasks. The panel provides the following buttons: "Delete", "Edit", and "Insert". The buttons are disabled unless a specific Work Task is selected.
5. User selects a Work Task.
6. User clicks on "Edit" button.
7. A window appears containing the description of the task (CNL) in a special text area.
8. The user edits the description. The text area provides masks for CNL editing. The text area provides interaction with the "3D Scene Viewer" when the mask requires to pinpoint a 3D object, further more relevant 3DObjects existing in Work Task are appropriate annotated. Text area provides the buttons: "Save" and "Cancel".
9. User writes verb "Pick" and points to an object in the "3D Scene Viewer".
10. Text area is updated with the clicked object's identifier.
11. User enters his credentials and presses the "Login" button
12. User clicks on "Save" and text area disappears.
13. The user navigates to "File->Save".
14. A dialog box is displayed that "Project is not validated. Save the project anyway?". User clicks on "Save".



15. The use case ends.

## 5. Alternative Flows

### a. Insert a new Work Task

If in step 4 of the basic flow the user clicks on “Insert”, then

1. Below the selected Work Task a new one is created.
2. A window appears containing an empty special text area.
3. The use case resumes at step 9.

### b. Generate a new Process/Work Task from recorded data (Editing from sensor data)

If in step 3 of the basic flow there is no “Process/Work Task” imported but an adequate set or recorded data exist, then

1. User navigates “Tools->Create “Process/Work Task” from simulated data”.
2. A window appears containing data sets from shop floor sensors.
3. User selects a data set and presses the “Create” button.
4. The use case resumes at step 4.

## Use-Case 3: Project variant management

### 1. Brief Description

This use case demonstrates the Project variant management functionality.

### 2. Actor Brief Descriptions

#### a. User. Any end-user

### 3. Preconditions

Project Management app is open.

### 4. Basic Flow of Events

1. The use case begins when user, navigates to “Tools->Project Variants”.
2. A dialog is displayed showing all the project’s variants highlighting the “current” one. The dialog contains the following buttons: “Delete”, “Load” and “New”.
3. The user clicks on “New”. A dialog box prompts for a name. Dialog contains “Create” and “Cancel” buttons.
4. User types the new variant name and presses the “Create” button.
5. A new variant is created which is a copy of the “current” one. The new variant becomes the new “current”. The project can have only one variant as “current” and each consequent changes to the project applies only to the “current”. Creating an project from scratch if you omit the creation of new variant all project data are stored under a default variant by the name default.
6. The use case ends.

## 8.3. System wide requirements list

Project validation rules:

1. All real world objects existing in the Process/Work Task must exist in the 3D scene.
2. Space requirements computation on 3D scene (no collision between objects)
3. Each 3D object is linked to a work task.
4. Basic kinematics for movement, e.g. drag position while defining the trajectory a part follows or kinematic constraints definition of parts

## 9. COLLABORATION MANAGER

### 9.1. Vision - Application description

#### 9.1.1. Problem and solution statement

The problem of	Collaboration between research, development and engineering (RD&E) and management for the verification of the planned production processes.
Affects	The way current collaborative workshops are performed with physical prototypes.
The impact of which is	<ul style="list-style-type: none"> <li>• Development time and cost. Especially with the increasing numbers of product variants.</li> <li>• Time to market</li> </ul>
A successful solution would be	<p>A digital environment providing means for synchronous and asynchronous communication as well as shared documentation (i.e. a shared view of a 3D environment). This application is required for performing collaborative workshops. The main user of the application is the Workshop Manager and Moderator.</p> <p>Functions allowed by the moderator include:</p> <ul style="list-style-type: none"> <li>- 3D model variants management. Decide on the 3D model version to be displayed on the shared 3D view.</li> </ul> <p>Each participant can:</p> <ul style="list-style-type: none"> <li>- Communicate with synchronous and asynchronous means.</li> </ul>

#### 9.1.2. Key application features

Need	Priority	Features
Creation of a digital workplace.	10	Open sessions and browse opened sessions for joining in.
Communication between participants of the digital workshop.	10	Synchronous (voice) and asynchronous (text messages) communication.
Show different variants of the same 3D model (i.e. a process variant or a different assembly sequence). Note: Variants are to be created from a separate application.	10	Project variant management. Select a Project (3D model aligned with a process) and a project's variant (a "tagged" instance of the model)

### 9.2. Use cases and scenarios

#### Use-Case 1: Creating/Joining a collaborative session

##### 8. Brief Description

The user opens up the "Collaboration Management" app and either accesses an already existing session or creates a new session for other participants to join.

## 9. Actor Brief Descriptions

- a. Workshop Manager
- b. Workshop Participants

## 10. Preconditions

The user is logged in in the “Enterprise Application Platform” and is able to access the “Collaboration Manager” app.

## 11. Basic Flow of Events

6. The use case begins when user, opens the “Collaboration Manager” app.
7. “Collaboration Manager” app provides a dialog box asking the user if he wants to create a collaborative session or join an existing session.
8. User selects the “create new collaborative session”.
9. A dialog is displayed to the user for input the session name.
10. User click on create button.
11. User is navigated to the “Collaboration Manager” desktop.
12. The use case ends.

## 12. Alternative Flows

### a. Join a session

If in step 2 of the basic flow the user selects the “join an existing session”, then

3. A dialog appears with the available sessions. A selectable list of session names with a “Join” button
4. The user select a desired session name
5. The user presses the “Join” button.
6. The use case resumes at step 6

## Use-Case 2: Communication

### 1. Brief Description

The user communicates synchronously and asynchronously with workshop participants

### 2. Actor Brief Descriptions

- a. **Workshop participant**

### 3. Preconditions

The user should participate in a collaborative session. For the synchronous communication a microphone and speakers are required to be connected to the user's computer.

### 4. Basic Flow of Events

1. The use case begins when user, participate in a collaborative session.
2. The system automatically tries to access user's microphone and speakers. No need for a particular action from the end user.
3. The use case ends.

### 5. Alternative Flows

- a. **Asynchronous communication**

If in step 2 of the basic flow the user wants to submit a message, then

1. In the "Collaboration Manager" desktop on the right corner there is a list of participants and below there is a text box. In the text box the user fills in the message that wants to be submitted.
2. The user selects from the participants list the applicable recipients of the message or selects NONE and then the message will be delivered to all participants.
3. The user presses the send button.
4. The use case ends.

## Use-Case 3: Open Project for collaborative session

### 1. Brief Description

The moderator opens a project to be discussed during the workshop.

### 2. Actor Brief Descriptions

- a. **Moderator**

### 3. Preconditions

The moderator should create a collaborative session.

### 4. Basic Flow of Events

1. The use case begins when moderator, created a collaborative session and navigated to the "Collaboration Manager" desktop.
2. Moderator navigates "File->Open Project".
3. A Project browser dialog is displayed showing all the projects he created or has access to view. Moderator selects the desired one and clicks "Open" button.
4. The "3D Scene Viewer" app opens and Project data is displayed inside.
5. The use case ends.

### 5. Subflows

- a. **Open different Project variant**

1. The moderator does not want to show the default Project variant but another one. He navigates on "File->Open Variant".

2. A Project Variant browser dialog is displayed showing all the project's variants. Moderator selects the desired one and clicks "Open" button.
3. The "3D Scene Viewer" updates its contents appropriately.
4. The use case ends.

**10.ERGONOMICS ASSESSMENT**

**10.1. Vision – Application description**

**10.1.1. Problem and solution statement**

Ergonomic assessment is a key feature of digital human models. Next to the correct evaluation of simulated processes, ergonomics can be an important parameter for the optimization of the simulation-process itself. The status quo of ergonomic assessment and optimization is an iterative process through which the user of DHM software has to find the best ergonomic solutions himself. Therewith the responsibility and estimation of the importance of ergonomics for certain work activities is in the users hands. The vision for future software products and thus in the INTERACT prototype is an automatic or at least interactive optimization of simulated movements and the ergonomic correctness and validation that comes with it. Interactive in this case implies the proposal of ergonomically better solutions for a movement execution by the software during planning workshops, as described in D1.1.1. The main objective of the ergonomic assessment application is the assessment of simulations within the scope of standards like EAWS, NIOSH, OCCRA or others. The outcome of these assessments has to be visualized as intuitively as possible.

The problem of	<ul style="list-style-type: none"> <li>• Necessity of ergonomically optimized work places and processes</li> <li>• Ergonomic conditions are not verified during design and operation.</li> </ul>
affects	<ul style="list-style-type: none"> <li>• Musculoskeletal disorders of workers with manual activities.</li> <li>• Waste due to improper handling of components.</li> </ul>
the impact of which is	<ul style="list-style-type: none"> <li>• Costs</li> <li>• Personnel health</li> <li>• Personnel satisfaction</li> <li>• Process and Product quality</li> </ul>
a successful solution would be	<ul style="list-style-type: none"> <li>• Valid assessment tool for the EAWS, NIOSH and OCCRA standards</li> <li>• A tool for interactive proposals for ergonomic optimized processes (large object handling) and object-designs (carrier design)</li> <li>• movement optimization based on ergonomic principles → back coupling to motion generation</li> </ul>

**10.1.2. Key application features**

Need	Priority	Features
Ergonomic assessment of existing simulations	10	EAWS, NIOSH standard
Visualization of ergonomic risks	10	real-time coloring of affected body parts
creation of ergonomic landscapes	8	analyses and visualization of ergonomic assessment of multiple work stations
identification of 'point-booster'	4	Analyzing tool for ergonomic assessment results
evaluation of non-human related working conditions	2	Assessment tool for different industry standards or/and guidelines
Constraint generation for sampling motions	10	Motions which severely affect ergonomic ratings should be identified, so that the motion generation tries to avoid them whenever possible

## 10.2. Use cases and scenarios

### 1 Assessment

The Ergonomic assessment app allows the 'classic' assessment existing work places for evaluation, documentation and ergonomic landscapes. Therefor the application doesn't differentiate between captured and synthesized motions.

### 2 Comparison

The second use case would be the comparison of different simulation alternatives in a workshop. The parallel display of ergonomic assessment of different solutions requires a user-friendly and intuitive way of displaying and comparing the ergonomic information.

### 3 Coaching

The third use case contains the ergonomic assessment as coaching tool on the shop floor level, where it could be used to make the ergonomic risk of certain behaviors more transparent for the worker.

## 10.3. System wide requirements list

### 1 Usability

The ergonomic assessment app as a tool for either a quick assessment or a comparison between different simulation variations, demands a clear and intuitive visualization. Through this requirement, the ergonomics application demands further requirements to other applications, like the 3D viewer, which should be capable of showing several simulations at once.

### 2 Performance

In comparison to other aspects of the INTERACT project, the ergonomic assessment should not be a critical component in computational costs.

### 3 Supportability

The ergonomic assessment will relay on standards. A possibility of the application should be the extension with further standards, if required by new user groups.

### 4 System Interfaces

The ergonomic assessment as analyzing tool depends strongly on informations generated by other system components. Interfaces are required for manual user input (changing of cycle times, etc.), the motion synthesis (angles, times, etc.) and the external sensors (loads ad forces).

#### **4.1 User Interfaces**

Even if most of the necessary information comes directly from other software components, there is a need for several manual inputs:

- Selection of ergonomic standard
- Cycle time (for quick comparison)
- Extra points in assessment tools (vibration, hot objects, etc.)
- User related adaptations of individual standards

#### **4.2 Layout and Navigation Requirements**

- 3D Viewer
- Alpha-numerical area for results, details (loads, postures, etc.), comparison

#### **4.3 Interfaces to External Systems or Devices**

Due to the requirements given by the ergonomic assessment standards, there is a need for an interface to the additional sensors like force sensors, or pressure distribution. These sensors are necessary as trigger, especially in captured motion sequences to determine the start and end of certain tasks or sub tasks. Furthermore sensor information will be used to evaluate occurring loads and forces, which are essential for the assessment.

#### **4.4 Software Interfaces**

The ergonomic assessment app will have minimal influence on other components of the software, since it works exclusively as an assessment tool. If an ergonomic optimization of synthesized motions is required, this won't be accomplished by the ergonomic app, but by the motion generation itself. The ergonomic assessment tool won't influence other apps directly; it gets input and provides Ergonomics results.

### **5 Applicable Standards**

Applied standards for ergonomic assessment will be:

- EAWS
- NIOSH lifting index
- OCRA
- others

### **6 System Documentation**

A documentation of the assessments and the changes in scores throughout changes in proposed process variations should be available in form of excel, pdf or csv files.



## 11. ENTERPRISE APPLICATION PLATFORM

### 11.1. Vision

#### 11.1.1. Problem and solution statement

The problem of	Design of a holistic framework for supporting application development, application management, access rights. Provide a unified way to data access. Provide generic services needed by the majority of app functionality (motion generation, motion constraints generation)
Affects	The development and deployment of INTERACT applications as well as provide guidelines for third party application development.
The impact of which is	<ul style="list-style-type: none"> <li>• Low cost third party application development</li> </ul>
A successful solution would be	<p>An “Enterprise Application Platform” with the following responsibilities:</p> <ul style="list-style-type: none"> <li>• Provide data access to applications in a seamless way</li> <li>• Authentication and Authorization management</li> <li>• Application hosting. Application Store paradigm.</li> <li>• Services for message/data exchange between applications</li> <li>• Services for constraints and motion generation</li> </ul>

#### 11.1.2. Key application features

Need	Priority	Features
Authentication and Authorization management	10	Provide a “Single Sign On” for all applications.
Application hosting	10	Application Store paradigm. Browse, download and use available apps.
Unified Data Access	10	An API for CRUD (Create Read Update Delete) operation on data sets
Message/Data exchange between applications	10	An API for inter-application communication.
Motion synthesis	10	<p>Provide services (API) that will generate:</p> <ul style="list-style-type: none"> <li>• motion constraints from “project” variant (3D scene and CNL or 3D scene and sensor dataset)</li> <li>• motions based on specific constraints</li> </ul>

## 11.2. Use cases and scenarios

### Use-Case 1: Single Sign On

#### 1. Brief Description

This use case demonstrates the single sign on procedure for accessing the platform and its functionalities.

#### 2. Actor Brief Descriptions

- a. **User. Any end-user**

#### 3. Preconditions

None.

#### 4. Basic Flow of Events

1. The use case begins when user, opens the “Enterprise Application Platform”.
2. “Enterprise Application Platform” provides a dialog box asking the user credentials, username and password.
3. User enters his credentials and presses the “Login” button.
4. User is navigated to the “Enterprise Application Platform” desktop.
5. The use case ends.

#### 5. Alternative Flows

- a. **Failed Authentication**

If in step 3 of the basic flow the user inputs wrong credential, then

1. A dialog appears with a warning message and an “OK” button
2. The user presses the “OK” button.
3. The use case resumes at step 2

### Use-Case 2: Apps access

#### 1. Brief Description

This use case demonstrates how the user browses, download and access an app from the integrated application store.

#### 2. Actor Brief Descriptions

- a. **User. Any end-user**

#### 3. Preconditions

User is successfully logged in into the “Enterprise Application Platform”.

#### 4. Basic Flow of Events

1. The use case begins when user is navigated to the “Enterprise Application Platform” desktop.
2. User navigates “Apps->Marketplace”.
3. A dialog displays the available apps with an install button under each app.
4. The user click on an install button.
5. The app is downloaded and installed.
6. The user navigates “Apps->Installed”
7. A dialog displays the installed apps.
8. User clicks on an App.
9. The “App” window is presented inside the platform.
10. The use case ends.

## 12. CONCLUSIONS

The deliverable on hand is the result of the requirements analysis phase of the INTERACT project. It was created based on the requirements from the specific pilot applications in the white goods, and automotive industries, but also considered more general application fields that have been analyzed by literature research. It serves as a guideline for the developers and enables them to address the specific business demands of the end users.

Therefore it describes a requirements catalogue containing

- use cases for each application
- functional and non-functional requirements for each application
- a first concept for the Enterprise Application Platform

The definition of the project Apps and their requirements will be used as input for WP5 “Applications for the human-centered workplace for manual assembly operations” in which the development of the Apps and EAP takes place.

**13.GLOSSARY**

DHM	Digital Human Modelling
EAP	Enterprise Application Platform
MEMS	MicroElectroMechanical Systems
MTM	Methods-Time Measurement
CNL	Controlled Natural Language
C-Values	C-Values are a Daimler in-house system that has been developed from MTM. Objectives are reduction of analysis time and application throughout the planning phase.
EAWS	Ergonomic Assessment Worksheet
NIOSH	National Institute for Occupational Safety and Health
OCCRA	Occupational Repetitive Actions