vf-OS: virtual factory Operating System

WP8: vf Smart Application Piloting and Validation

D8.1a: Validation Scenarios - Vs: 1.0.3

Deliverable Lead and Editor: Raúl Poler, UPV

Contributing Partners: UPV, IKERLAN, UNINOVA, MASS, VS, CON, APR, TARDY, KBZ

Date: 2017-03

Dissemination: Public

Status : Draft

Short Abstract (Teaser)

This is the first iteration of the Validation Scenarios deliverables in which the methodology for the definition and the validation of vf-OS pilot applications is presented. It also includes the main goals and indicators used to evaluate pilot vApps and the development schedule. The methodology establishes a reference for the interactions between different vf-OS customer groups developing and using new vApps.

Grant Agreement: 723710

Document Status

Deliverable Lead	Raúl Poler, UPV
Internal Reviewer 1	Andries Stam, ALM
Internal Reviewer 2	Eduardo Saiz, IKERLAN
Internal Reviewer 2	Stuart Campbell, ICE
Type	Deliverable
Work Package	WP8: vf Smart Application Piloting and Validation
ID	D8.1a: Validation Scenarios
Due Date	2017-03
Delivery Date	2017-05
Status	Draft

History

See Annex B

Status

This deliverable is subject to final acceptance by the European Commission.

Further Information

www.vf-OS.eu

Disclaimer

Project Partners:

Executive Summary

Deliverable “D8.1a Validation Scenarios” provides the methodology used in the vf-OS project to support the implementation of validation scenarios. After the project, future vf-OS Platform users will interact to specify and develop new vApps. The methodology takes into account these post-project interactions, so that they can be analysed in the pilots in order to validate business hypothesis and find new business opportunities for vf-OS as a multi-sided platform. Thus, this deliverable enumerates the successive steps and interactions that occur in the methodology, during the following phases:

- **Initial Specification:** The starting phase that defines the functionality of vApps. The major actors involved in the process are the Manufacturing and Logistic Users (represented in vf-OS by pilot end users) and Software Developers (represented by pilot R&D partners).
- **Development and Test:** The phase in which the vApps are developed by Software Developers. The actors involved are the same as above although with more workload on the side of the Software Developers. Software releases are deployed inside pilots´ premises and tested by Manufacturing and Logistic Users. Together, they review the development plan with the feedback from the test results and plan the next release.
- **Evaluation:** During the evaluation phase, the vApps are validated to guarantee that they fulfil the needs of Manufacturing and Logistic Users. During this phase, another actor is involved in the process, the evaluator, which is represented in vf-OS by the partners that support the pilots in their implementation.

The deliverable also presents the templates and guidelines that both pilots and future users of vf-OS platform should fill to create a common understanding of what is desired. This includes a use case definition template, an application questionnaire, and guidelines for the creation of sketches, mockups, and workflow diagrams. These guidelines have been developed from the experience gained in the definition of the use cases described in D1.2.

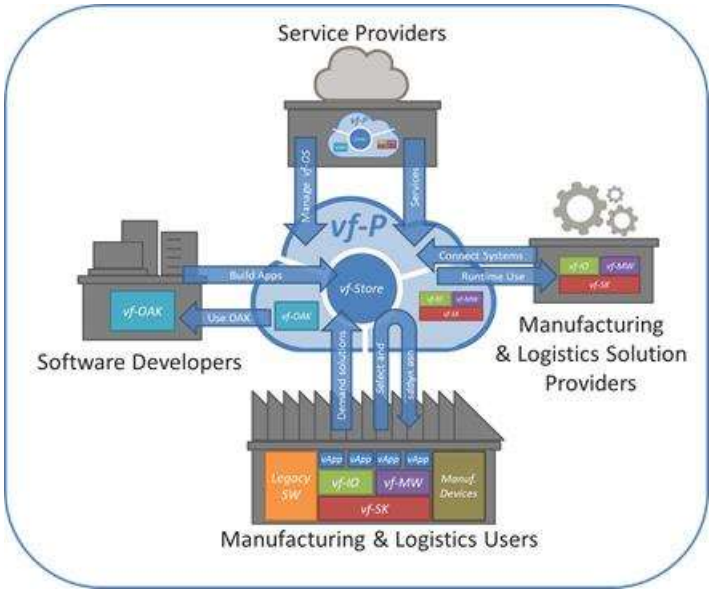
To guarantee the successful implementation of a vApp, the methodology also includes the definition of generic and pilot specific goals and indicators.

Finally, this document sets the implementation roadmap for the development and evaluation of the pilots specific vApps that were previously defined in D1.2.

Table of Contents

0 Introduction

0.1 vf-OS Project Overview



vf-OS – virtual factory Open Operating System – is a project funded by the H2020 Framework Programme of the European Commission under Grant Agreement 723710 and conducted in the period October 2016 until August 2019. It engages 14 partners (Users, Technology Providers, Consultants and Research Institutes) from 7 countries with a total budget of circa 7.5M€. Further information can be found at www.vf-OS.eu.

The World is facing the fourth industrial revolution based on ICT, specifically architectures and services, as key innovation drivers for manufacturing companies. Traditional factories will increasingly be transformed into smart digital manufacturing environments but currently the full potential for ICT in manufacturing is far from being fully exploited. Factories are complex systems of systems and there is a need to develop a platform on which future manufacturing applications can be built. Examples of platforms exist in some industrial sectors but there is a lack of cross cutting platforms based on open standards for creating an ecosystem for cooperative innovation. Innovative open platforms to attract talent from solution developers and to provide accessible manufacturing smart applications to European SMEs are examples of the kind of solutions being sought.

The goal of vf-OS is to develop an Open Operating System for Virtual Factories composed of a kernel, application programming interface, and middleware specifically designed for the factory of the future. An Open Applications Development Kit (OAK) will be provided to software developers for deploying Manufacturing Smart Applications for industrial users, using the vf-OS Manufacturing Applications Store all operated through a Virtual Factory Platform.

The Virtual Factory Platform is an economical multi-sided market platform with the aim of creating value by enabling interactions between four customer groups:

- **Software Developers (independent or within individual manufacturers)** which will build Manufacturing Apps either through innovation or from Manufacturing and Logistic User demand
- **Manufacturing and Logistic Users** which will explore the marketplace for already created solutions, ready to be run on the vf-OS
- **Manufacturing and Logistics Solutions Providers** which will provide ICT interfaces and manufacturing connections

- **Service Providers (vf-OS innovators and third parties)** will make available services (hosting, storage, connected cloud services, etc.) including those based on developed solutions

The Virtual Factory Platform will provide a range of services to the connected factory of the future to integrate better manufacturing and logistics processes. Manufacturing Applications Store will be open to software developers who, using the free Open Applications Development Kit provided, will be able to quickly develop and deploy smart applications to enable and optimise communication and collaboration among supply networks of all manufacturing sectors in all the manufacturing stages and logistic processes.

vf-OS aims to become the reference system software for managing factory related computer hardware and software resources and providing common services for factory computational programs. This operating system will be the component of the system software in a real factory system where all factory application programs will run.

0.2 Deliverable Purpose and Scope

Deliverable D8.1a is a technical report with the description of the methodology to be used by Manufacturing and Logistic Users, represented by the three project pilots domains to specify their needs and for Software Developers to provide solutions in the forms of vApps.

According to this methodology, D8.1a also provides the definition of the metrics needed to evaluate all the processes (specification, implementation, and use). This way, the scope of D8.1a is to define the methodology to collect, assess, and formalise the results of the use cases, with the aim of reporting the performance of the vf-OS Platform and tools in operational scenarios.

The definition of general and detailed specifications of the pilots is out of the scope of D8.1a. As a future iteration of this deliverable, D8.1b will include detailed pilot applications definitions and an assessment of the current (AS-IS) value of the indicators which will be used to validate the use cases. In fact this work has already started in parallel which has assisted in contributing to the methodology of this document.

The involvement of the pilots in this task is crucial for validating this methodology. Through the adoption of an incremental development model for the pilot vApps, they will continuously provide fundamental knowledge into other WPs. This feedback will be reported in D8.1c and D8.1d. In this way, the consortium will be able to immediately address needs coming from the Users' sites.

In summary, the deliverable describes the methods, support documents and reporting mechanisms used to validate the feasibility and functionality of vApps, the vf-OS platform, and in extension, the project pilots.

The relationship between the D8.1x series of documents is detailed in Annex C.

0.3 Target Audience

Whilst primarily aimed at the project partners, this public deliverable can be useful for the wider scientific and industrial community. This includes other publicly funded projects which may be interested in collaboration activities.

0.4 Deliverable Context

This document provides the methodological framework for the project pilots in WP8. Subsequent iterations of this deliverable and the development of applications for the project pilots are based on the results presented in this document. The methodology is mainly based on the experience and documents from WP1. The relationship to other deliverables is as follows:

- **Validation Scenarios (D8.1b):** A report defining the functionality for the different versions of the pilot applications as agreed by Manufacturing and Logistic Users and Software Providers according to the methodology to define vf-OS applications presented in this document
- **Validation Scenarios (D8.1cd):** A series of reports providing the results of the evaluation of pilot applications at different stages of the project following the methodology to evaluate vf-OS applications presented in this document
- **Pilot 1: Manufacturing & Logistic – Automation (D8.2abc):** A series of demonstrators of the applications developed for pilot domain 1
- **Pilot 2: Construction – Industrialisation (D8.3abc):** A series of demonstrators of the applications developed for pilot domain 2
- **Pilot 3: Manufacturing Assembly – Collaboration (D8.4abc):** A series of demonstrators of the applications developed for pilot domain 3
- **Vision Consensus (D1.1):** A report to strengthen the common understanding of the project vision and the overall aims and objectives of the project
- **User Scenarios Characterisation (D1.2):** A report providing the characterisation of the main industrial scenarios and the initial pilot scenarios description
- **Providers Scenarios Characterisation (D1.3):** A report providing the characterisation of the main developer scenarios that will use the vf-OS Platform for developing Manufacturing Smart Applications

0.5 Document Structure

This deliverable is broken down into the following sections:

- **Section 1:** **Methodology for the Implementation of Validation Scenarios**Error! Reference source not found. describes a methodology to define vf-OS use cases and to evaluate the level of satisfaction perceived by the different stakeholders. The definition of standard methods and procedures to define vApps will help Manufacturing and Logistic Users to express what they want. It will also encourage an active participation of Manufacturing and Logistic Users and Software Developers in the definition process, and will simplify the mapping of application features to functional requirements. In addition to this, the methodology will define the indicators used to evaluate pilot use cases.
- **Section 2:** **Use Case Definition Templates and Guidelines** contains the document templates, questionnaires, and guidelines used to define use cases including their corresponding workflows. Defined use cases are used to provide sample content and serve as examples for future use cases.
- **Section 3:** **Monitoring, Reporting, and Evaluation Methods** describes the procedures and indicators used to evaluate the applications development, to validate the use cases of the project pilots and to

measure Manufacturing & Logistic Users satisfaction. Also, indicators are mapped with applications to determine to what extent validation scenarios cover the expected impact of vf-OS.

- **Section 4:** **Pilot Scheduling** contains the planning for the implementation of the pilots. WP8 will make sure that the methodology is shared with all partners and that it facilitates their participation in the implementation phase. Furthermore, Section 4 also contains the implementation plan and the roadmap for the deliverables 8.1b, 8.1c, and 8.1d in which the methodology will be evaluated and improved.

Annexes:

- **Annex A: Document History**
- **Annex B: Reference**
- **Annex C: WP8 Deliverables**

0.6 Document Status

This document is listed in the Description of Action as “public” since it provides general information about the goals and scope of vf-OS and can therefore be used by external parties to receive insight into the project activities.

0.7 Document Dependencies

This document has no preceding documents. The document has three further interactions: D8.1b, D8.1c, and D8.1d. This deliverable defines the methodology used to validate the user scenarios in the project. According to this methodology, deliverable D8.1b will describe how vf-OS Partners have applied this methodology in the definition of the pilots. D8.1c, and D8.1d will monitor the development of applications and will verify that the applications respond to user needs and expectations. In addition to this, D8.1d will provide improvements to the validation methodology from the experience of the pilots. These relationships are described in detail in Annex C.

0.8 Glossary and Abbreviations

A definition of common terms related to vf-OS, as well as a list of abbreviations, is available in the supplementary and separate document “vf-OS Glossary and Abbreviations”.

Further information can be found at http://www.vf-OS.eu/glossary

0.9 External Annexes and Supporting Documents

None.

0.10 Reading Notes

None.

1 Methodology for the Implementation of Validation Scenarios

This section presents the methodology that will be used for the definition and implementation of validation scenarios. Section 1.1 describes the methodology and the rationale for its adoption. Section 1.2 Story Mapping and Section 1.3 Goals Questions Metrics (GQM) describe the core components of the methodology. Finally Section 1.4 describes its main steps and interactions.

1.1 Methodology Description

In this section, a common methodology is presented to support vf-OS Manufacturing and Logistic Users and Software Developers to define use cases and to evaluate and validate the functionalities of vApps. The methodology is used to define and evaluate validation scenarios for the project pilots, although the long-term objective is to facilitate the interaction between different vf-OS customer groups. As mentioned in the DOA, although there are platforms for connectivity of users and providers, few or none support the connection between needs and solutions. In vf-OS, Manufacturing and Logistic Users will use the platform to request functionalities and Software Developers will provide new applications based on these requests. Therefore, interaction between both customer groups is one important aspect of the vf-OS Platform.

In this sense, both customer groups need to define and agree on the functionality, procurement and planning of new vf-OS applications. It is therefore convenient to implement methods to:

- **Guide the definition of use cases:** One objective of the methodology is to establish standard procedures to capture the needs and expectations of Manufacturing and Logistic Users. This standardisation is beneficial for both customer groups. For Manufacturing and Logistic Users, standard templates and guidelines can ease the use case definition process and improve the quality of application requests. On the other hand, standardisation can reduce efforts in the selection and preparation of proposals by Software Developers.
- **Monitor the development of new applications:** It is important to establish mechanisms to track the development of new applications and to assess the performance of development activities. With these mechanisms, Software Developers can plan development activities more accurately and build a reputation according to their performance and conformance to agreed deadlines.
- **Evaluate and validate the functionality of new vf-OS applications:** In addition to this, it is important to account for the quality of development activities with regards to the accomplishment of negotiated objectives and requirements for applications. Evaluation is also relevant to certificate compliance with (industry) standards. It is thus necessary to facilitate the evaluation and validation of the functionality of new applications in the vf-OS platform.

The proposed methodology addresses the three objectives mentioned above and is based on common practice. This methodology adopts a top-down approach towards requirement elicitation, from business requirements defined by Manufacturing and Logistic Users to application features for software development agreed with Software Developers.

Additionally, the methodology has been aligned with the agile management methods adopted by the vf-OS project.

The core components of the methodology are:

- **Story mapping:** Story mapping [PAT+14] is a strategy used in agile software development to build shared understanding of the requested software. Story mapping is described in Section 1.2.
- **Goal Questions & Metrics (GQM) method:** GQM [GQM+14] intends to define a measurement model for software composed by three levels: the conceptual (goals) level, the operational (questions) level, and the quantitative (metrics) level. The GCM method is described in Section 1.3.
- **Use Case Definition:** Gathers the information that defines the use case and that represents the baseline for the story mapping. Use Case Definition Templates and Guidelines, described in Section 2, provide the supporting documents needed by Manufacturing and Logistic Users to describe their needs and submit application requests.

Error! Reference source not found. identifies how the core components of the methodology are to be used by the different actors:

- **Manufacturing and Logistic Users:** The customers and consumers of vApps
- **Software Developers:** vApps developers with the necessary capabilities to implement Manufacturing and Logistic Users' needs
- **Evaluators:** The neutral party responsible for evaluation based on data gathered from Manufacturing and Logistic Users and Software Developers

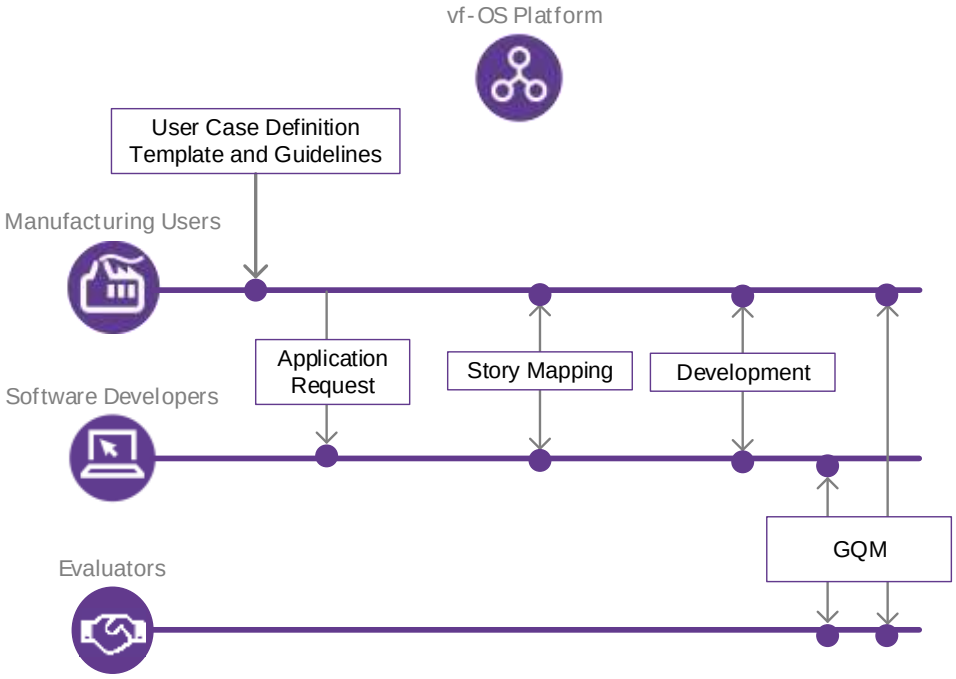


Figure 1. Methodology workflow and interactions

As depicted in **Error! Reference source not found.**, Manufacturing and Logistic Users make use of guidelines and templates to define their use case. According to these guidelines, they submit an Application Request to request a new vApp to be developed. Software Developers and Manufacturing and Logistic Users then interact to write the story map and to develop the application. Finally, evaluators interact with both consumer groups

using the GCM method to evaluate the software delivered. Section 1.4 describes the different steps and interactions in detail.

Besides the two main customer groups involved in the definition and implementation of use cases, the methodology introduces the evaluators as a neutral third party to conduct an independent evaluation using standard methods. The evaluation of software by independent evaluators is beneficial for both customers and developers, since it allows the identification of the quality of a product in a standardised and neutral way. This is a common procedure in software certification and a fundamental part to ensure compliance with Industry 4.0[1] standards and software quality standards like ISO/IEC SQuaRe [ISO+11]. Independent evaluation can also be beneficial for the evolution of the vf-OS Platform and as a general means to provide support to the different customer groups.

Consequently, it must be taken into consideration by the methodology and by the vf-OS Platform after the project. Prior to this, in the vf-OS project vision (D1.1), evaluators are regarded as providers hired by Manufacturing and Logistic Users to perform the evaluation tasks and there are no specific features in regards to automate the interactions with evaluators from vf-OS. However, after the project, evaluators can be regarded as Service Providers offering consultancy services on the vf-OS platform (ie another customer group in the vf-OS multi-sided platform). Obviously, in many cases Software Developers and/or Manufacturing and Logistic Users might not want to purchase such a service. Therefore, although in some cases the evaluation is an important part of the methodology (for instance, the validation of the pilots), it is not regarded as a mandatory process for all vApps in vf-OS and the vf-OS platform is not presented as supporting any interactions with evaluators.

On the other hand, the vf-OS Platform in **Error! Reference source not found.** represents the vf-OS collaborative environment supporting the interactions between the different actors highlighted above. The definition of the level of support provided by the vf-OS Platform and the methods used to implement this support in the long-term are out of the scope of D8.1a. Due to the agile approach adopted in the vf-OS project, it is expected that the requirements for these interactions evolve during the project, especially those related to the input from the pilots. Instead, one of the objectives of D8.1a is to highlight and describe these interactions among customer groups, in order to take advantage of the pilots to identify business opportunities for vf-OS as a multi-sided platform supporting these interactions. Following the lean start-up methodology[2], the vApp piloting and validation allows the experimentation of these interactions in a realistic environment. This way, it is possible to elicit business opportunities and test value proposition hypotheses around the interactions of involved customer groups even if the vf-OS Platform is not ready.

The following subsections describe the main core components of the methodology and the corresponding steps and interactions.

1.2 Story Mapping

The first step towards the definition of use cases is to facilitate the Manufacturing and Logistic Users context information in formats that help Software Developers understand user needs and transform them into development specifications. User stories and story mapping are common tools used in agile software development methodologies to

[1] http://www.plattform-i40.de/I40/Navigation/EN/Home/home.html
[2] https://es.wikipedia.org/wiki/Lean_startup

accomplish this. User stories describe features of an application from the point of view of the subject who expects the new feature. User stories normally follow a standard format: as *subject*, I want *what* so that *why*. Thus, the syntactic components of the description of a user story are the subject expecting the new functionality, a brief description of the expected functionality and a brief description how will the subject benefit from the new feature. Figure 2 lists the main elements of a user story.

User Story	
Description As *subject*, I want *what* so that *why*	
Acceptance criteria (List acceptance criteria)	
User priority (Essential, Important, Nice to Have, cosmetic)	
Unit tests (List of unit tests for development)	**Related requirements** (List of related requirements)

Figure 2: Elements of user stories

In user stories, the subsect is not restricted to the final user of the system being described and can be any entity with a behaviour, even the system being described when it calls other services or systems. Also, there is no strict definition for the level of detail of a user story. Normally, user stories start with broad descriptions that cover a lot of functionality and are known as epics, or user tasks if they imply user interactions. Epics or user tasks are broken down into more concise user stories in an iterative process until each user story describes a software development task and a group of user stories can be scheduled for a development sprint.

The acceptance criteria – a checklist that determines when the user story is considered as “done” – is an important element of user stories. The acceptance criteria are also expressed from the point of view of the subject that formulates the user story and provides a detailed description of the criteria by which user stories should be evaluated and validated. The acceptance criteria are useful to define granular tests (unit tests) to evaluate and validate the functionalities of the application. In this sense, it is important to note that a user story is not a replacement for a requirement, since stories only describe the features wanted by users. Instead, they should be regarded as pointers to requirements, when the functionality of the application is described from the point of view of the user.

In story mapping, user stories are organised following a hierarchical structure. The first step in story mapping is to understand the problem and define the general objectives of the application. Each overall objective is referred to as a goal and will represent a different branch of the story map, which is explained below:

- **Steps:** In user driven approaches, Goals are divided into the steps needed to achieve these goals.
- **Activities:** Activities provide a coarse definition of the behaviour of the system. Activities are organised from left to right describing the subject workflow (what will the

subject perform with the system) in the sequence the behaviour of the system is explained.

- **User tasks:** Activities are further split into user tasks, which are specific interactions between the subject and the system to complete an activity. User tasks are organised in the horizontal dimension following a logical sequence to complete the activity.
- **Stories:** Finally, user stories are defined for each user task. The user establishes priorities for each user story and the user stories for each task are organised in the vertical dimension by priority.

Once the story maps are defined, stakeholders determine that are the most important stories to achieve the (set) goals. They then define the Minimum Viable Product (MVP) version of the application that can be used to start testing. The development of the MVP is scheduled into the first release. Future releases implement the rest of the stories, yielding an incremental development plan for the application, which is reviewed and adapted in each iteration. Figure 3 shows the organisation of story maps.

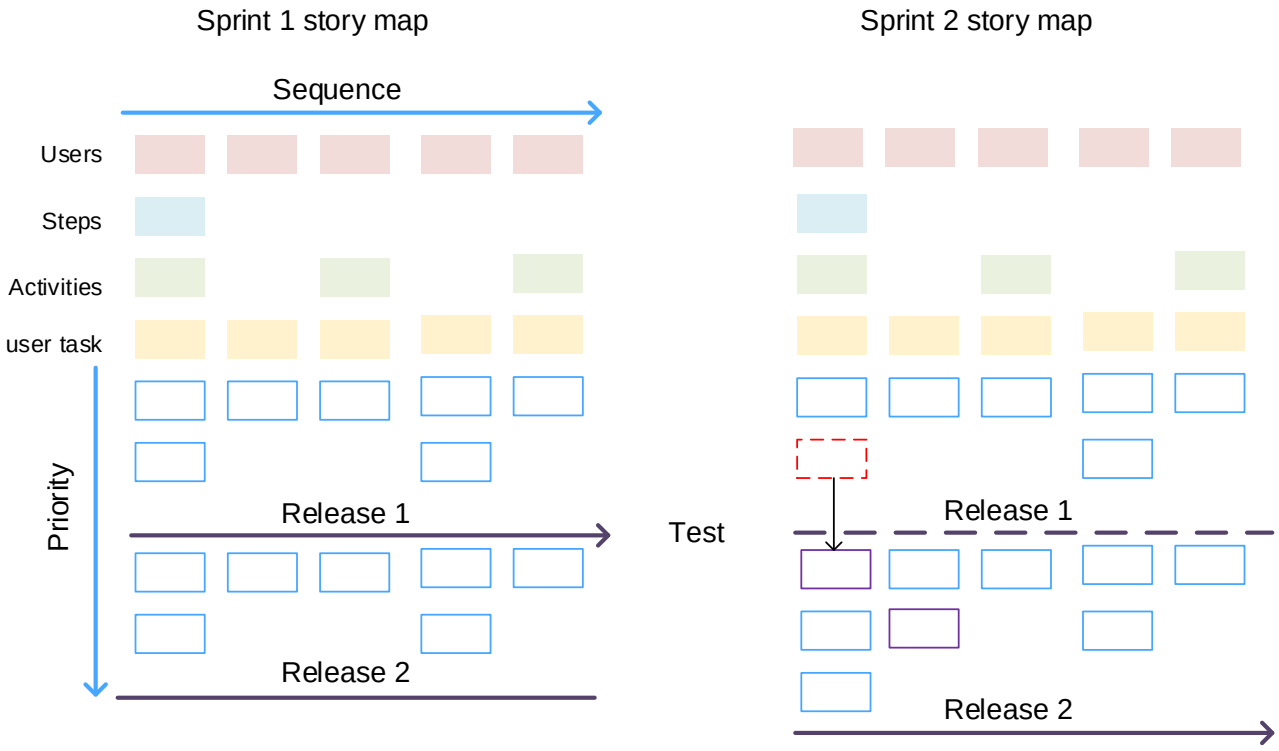


Figure 3. Story mapping organisation

This way, through story maps, Manufacturing and Logistic Users and Software Developers have a tool to define the context information needed by Software Developers to understand the needs and expectations of Manufacturing and Logistic Users. Furthermore, they can both agree on the features implemented on every Sprint and can define unit tests that need to be conducted to evaluate each release.

In line with story mapping, the methodology provides Manufacturing and Logistic Users with templates and guidelines to define goals, steps, and activities for their use cases when they submit an application request. These take into account the particularities of the vf-OS environment: Manufacturing sector, collaboration in supply networks, and the vf-OS multi-sided platform. These forms represent the starting point to start building story maps, in collaboration with Software Developers. In this context and for the definition of story

maps in this context, it is important to identify interactions with existing applications (eg ERP) or devices (eg sensors) that need to be connected to the vf-OS platform.

Moreover, Software Developers need to know details about the Manufacturing and Logistic Users and how they want to interact with the application in order to optimise user experience. Normally, user stories are accompanied with representations of users, ie Actors that help understand the context of use of the application. In vf-OS, it is important to bear in mind that, most likely, Manufacturing and Logistic Users belong to different organisations among the supply chain, or are involved in different supply chain processes. In addition to this, further information may be required to guarantee that Manufacturing and Logistic Users and Software Developers understand each other properly. Forms are provided to specify the terminology and clarifying concepts for developers to better understand the use case.

1.3 Goals Questions Metrics (GQM)

Evaluation is a complex procedure. In general terms, it can be described as a process of analysing completed or ongoing activities. As defined in [STE+70]: “Evaluation is a process of assessing the worth, desirability, effectiveness, and adequacy of a process, a product, or an activity according to certain criteria and with a certain purpose in mind.” In this context, assessment is about comparing achieved results with some reference criteria or in other words, the extent to which the set of goals can be achieved.

The GQM method has been chosen in vf-OS as the proper approach for evaluation. The GQM method, as it follows from the definition, includes three main constituents [BCR+94], Goals (conceptual level), Questions (operational level), and Metrics (quantitative level):

- **Goals:** Identify the desired outcome or functionality and are defined for an object which can represent products, processes, or resources. In the context of vf-OS, GQM goals is the same concept as story mapping goals defined in Section 1.2.
- **Questions:** Are used for the characterisation of defined goals. Questions identify key points that are critical for the assessment of whether the achieved goal meets the requirements.
- **Metrics:** Provide answers to questions in a qualitative or quantitative way. Thus, Metrics can be objective (independent of a certain viewpoint) or subjective (combining measurements and a certain viewpoint).

The first step of the GQM is to identify goals, since the questions and metrics are using them as a basis. In this sense, the methodology establishes a set of generic goals applicable to all vApps and a set of specific goals for vApps that correspond to the different goals of the story mapping. From these goals, a set of questions is obtained and the corresponding metrics to evaluate them are defined. Key stakeholders fill in the corresponding evaluation forms to obtain the metrics defined. Metrics must cover all software quality evaluation requirements. In this sense, ISO/IEC 25040 [ISO+11] defines seven Evaluation Modules (EMs): EM1 Functionality, EM2 Functionality (handle errors), EM3 Reliability, EM4 Usability, EM5 Efficiency, EM6 Maintainability, and EM7 Portability. The methodology regards the eight ISO/IEC EMs. Additionally, specific questions and metrics are used to integrate other functional or non-functional requirements in the scope of each use case scenario. In this sense, the definition of acceptance criteria and unit tests in story maps provide evaluators with a baseline to define the metrics used to assess the functionality of the applications, as explained in detail below in Section 3.3.

1.4 Steps and Interactions

The following subsections describe the main steps and interactions between the different actors (Manufacturing and Logistic Users, Software Developers and Evaluators). The vf-OS Platform is represented as an intermediate element supporting the different interactions. Steps are grouped into three phases which are further described below:

- Initial Specification phase
- Development and Test phase
- Evaluation phase

1.4.1 Initial Specification Phase

The initial specification phase consists of the following steps, represented in Figure 4.**Error! Reference source not found.**

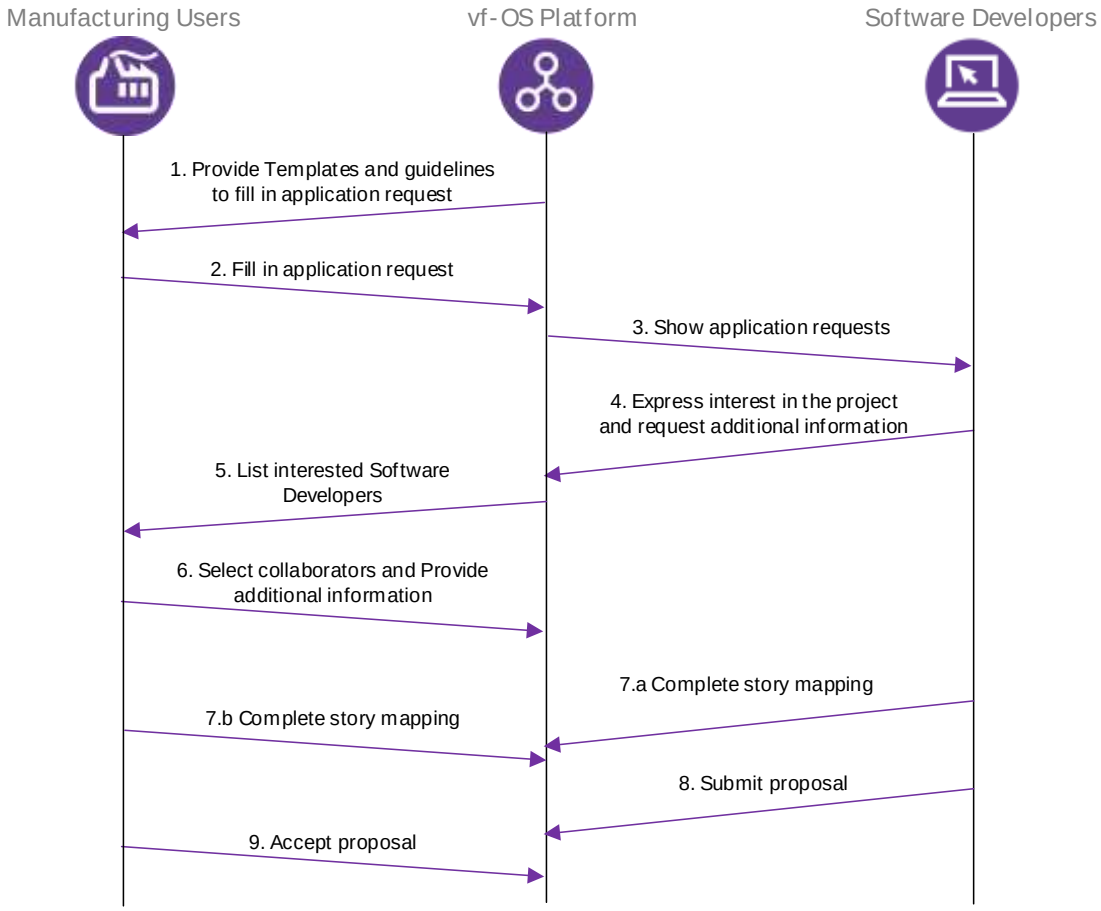


Figure 4. Initial Specification phase steps

1. **Provide templates and guidelines to fill in application request:** Users receive the Use Case Definition Templates and Guidelines to prepare an application request. This stage is needed to set the purpose and structure for upcoming tasks in order to ease the communication process between Manufacturing and Logistic Users and Software Developers.
2. **Fill in application request:** The forms and templates received during the previous step are used to fill in the application request by the Manufacturing and Logistic Users according to the guidelines. They also specify the time windows for Software Developers to express interest and submit proposals.

3. **Show application requests:** Software Developers are informed of the open application requests
4. **Express interest in the project and request additional information:** Software Developers who possess the necessary capabilities respond to the request to express interest and request additional information.
5. **List interested Software Developers:** Manufacturing and Logistic Users are informed of the interest in the request from different Software Developers
6. **Select Collaborators and Provide additional information:** Manufacturing and Logistic Users respond to the Software Developers they want to collaborate with. In this response, additional information, which is sufficient for Software Developers to formalise an offer, is provided by Manufacturing and Logistic Users.
7. **Complete story mapping:** Manufacturing and Logistic Users and Software Developers interact to complete the story mapping. During this phase, both customer groups elaborate ideas and develop supporting material like workflow diagrams and mockups to build shared understanding.
8. **Submit proposal:** The joint proposal based on mutual interests is formed and submitted. The proposal specifies the expected release plan for the different versions of the vApp, development costs, and the acceptance criteria obtained from the story mapping.
9. **Accept proposal:** After final revision of the offer, the user accepts the joint proposal

1.4.2 Development and Test Phase

As highlighted in Figure 5, the Development and Test phase is an iterative phase where Software Developers release new versions of the vApp and Manufacturing and Logistic Users test the provided software.

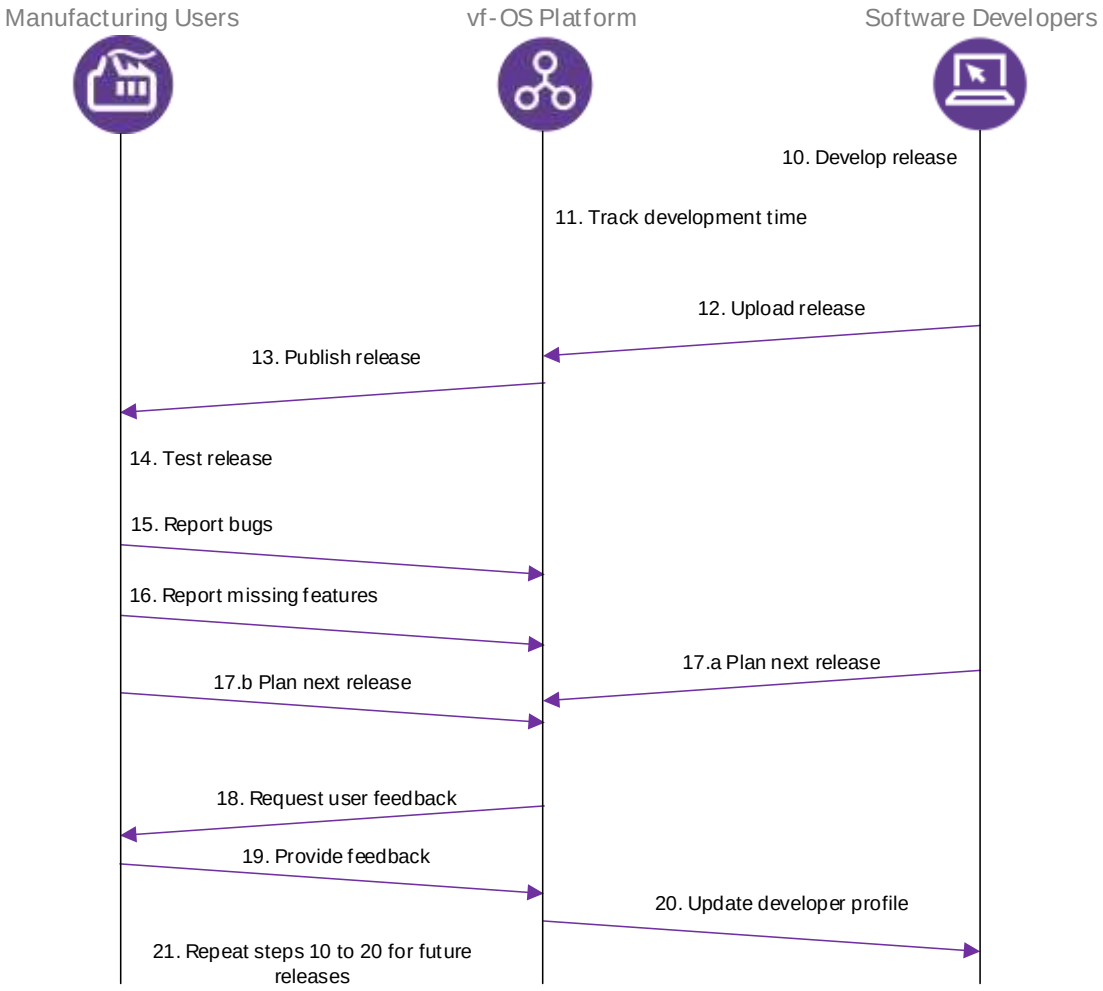


Figure 5. Development and Test phase interactions

The steps are described as follows:

10. **Develop release:** Software Developers develop a release of the vApp
11. **Track development time:** The time invested in software development is tracked. In the vApp piloting and validation, this is necessary to assess whether the vf-OS tools provide enough support for rapid vApp development. In the long-term, the Development Engagement Hub in the vf-OS platform will support development projects and development time tracking is a common feature to help developers plan and track software releases.
12. **Upload release:** The release of the vApp is provided
13. **Publish release:** The new release is published
14. **Test release:** Test iteration of received release
15. **Report bugs:** Technical gaps are reported
16. **Report missing features:** Missing features that were planned for this release are identified and reported
17. **Plan next release:** The plan for the next releases is adjusted according to the results of the previous iteration

18. **Request user feedback:** Feedback on the release is requested from users
19. **Provide feedback:** Manufacturing and Logistic Users provide feedback on the release
20. **Update developer profile:** The Software Developer profile is updated according to the feedback provided by Manufacturing and Logistic Users
21. **Repeat steps 10 to 20 for future releases:** The steps are repeated in an incremental development plan to improve the vApp

1.4.3 Evaluation Phase

The following steps are involved in the evaluation phase, depicted in Figure 6:

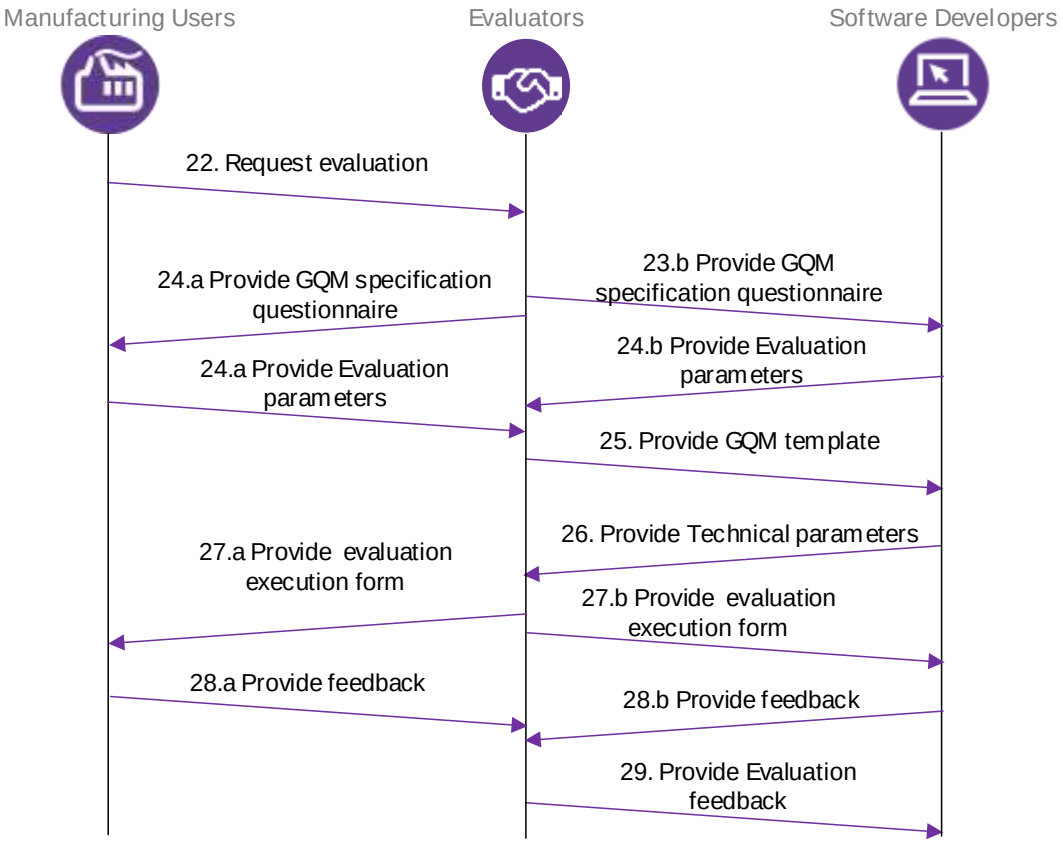


Figure 6. Evaluation phase interactions

22. **Request evaluation:** The Manufacturing and Logistic User makes a request for evaluation of the vApp release to a Service Provider based on predefined parameters and functionalities.
23. **Provide GQM specification questionnaire:** According to the information of the vApp received, evaluators prepare a questionnaire to prepare the specifications of the GQM method.
24. **Provide evaluation parameters:** Manufacturing and Logistic Users and Software Developers fill in the GQM specification questionnaire. This way, they identify parameters, which are critical from their point of view in the context of a certain pilot regarding to all components being evaluated.
25. **Provide GQM template:** A template is being prepared by the evaluators according to GQM approach and the GQM specification questionnaire results (parameters identified by Manufacturing and Logistic Users and Software Developers).

26. **Provide technical parameters:** Technical details are provided by Software Developers in addition to the evaluation parameters
27. **Provide evaluation execution form:** This step includes both functional and non-functional parameters in the form of a questionnaire to be answered by Manufacturing and Logistic Users as well as Software Developers [MSB+17]. The Execution form encapsulates technical parameters from Software Developers and evaluation parameters from Manufacturing and Logistic Users.
28. **Provide feedback:** The results are collected
29. **Provide evaluation feedback:** Analysis of achieved results and recommendations provided to Software Developers for effective improvements

Following the evaluation feedback, Software Developers may decide to develop new releases considering the gaps identified during the evaluation stage, thus starting a new Development and Test phase.

2 Use Case Definition Templates and Guidelines

This section contains the templates, questionnaires and guidelines used by Manufacturing and Logistic Users to fill in application requests.

2.1 Rationale

To facilitate the use case definition, a user should fill in an application request, ie a document containing several forms that capture significant information to define the use case that serves as the starting point to start preparing the story maps. This information is not used to elicit all user needs since, as explained in Section 1.3 this is performed in cooperation with Software Developers. Instead, the objective is to provide a ballpark estimation of the project dimension, technical requirements, and development costs that can help Software Developers filter and sort project proposals at an early stage. Later, Software Developers and Manufacturing and Logistic Users interact to build a shared understanding of the use case, from the start point provided by the application request.

Together with the forms, Manufacturing and Logistic Users should have a guide explaining how the application request should be filled in and how they can be complemented with supplementary material, eg UI sketches, mockups and workflow diagrams. All this information will be compiled into a document called “Use Case Definition Templates and Guidelines” that will be developed in this Work Package.

The different versions of the guide will be annexed to future versions of this deliverable. Section 2.2 describes and motivates the set of forms – Organisation, Application, Goals, Actors, Activities, and Terminology – that will be included in the application request and described in the guide. In addition to these forms, Section 2.3 provides a questionnaire filled in by Manufacturing and Logistic Users to help Software Developers understand the vf-OS components involved in the application feature set. This questionnaire is also part of the application request forms and contains basic questions that can be easily understood by final Users, without in-depth knowledge of vf-OS platform components.

Finally, Section 2.4 and Section 2.5 introduce and provide guidelines for the preparation of supplementary material for the application request: User Interface sketches, mockups, and workflows diagrams. The great benefit of creating UI sketches and mockups is that Manufacturing and Logistic Users can have a full view on what the application will look like, get a better understanding of its functionalities, and can easily make the necessary adjustments for it. On the other hand, it eases the work of the Software Developer in terms of understanding the frontend and backend functionalities of the vApp. Similarly, a workflow diagram is a graphical tool used to describe the steps taken by different actors when using applications and to depict other relevant aspects like interactions with devices. Application workflow diagrams emphasise the most important concepts of an application and are helpful to reinforce the description of goals and activities for applications.

With these templates and guidelines, Manufacturing and Logistic Users and Software Developers will have a standard method to define vf-OS use cases and grow a common understanding on the user needs and expectations for new vApps.

2.2 Application Request Forms

First, Manufacturing and Logistic Users can provide a preliminary name for the application and a brief, high level, description of its functionality. This information is provided in the application information form in Figure 7**Error! Reference source not found.**.

Application	
Name	Provide a preliminary name for the application
Description	Provide a short description of the main application functionalities
Inspiration [Optional]	Add URLs of similar applications

Figure 7: Application information form

Also, Manufacturing and Logistic Users need to provide a brief description of their organisations. In case that a consortium wants to request an application, Manufacturing and Logistic Users need to provide information of every organisation in the consortium. If the requested application must support interactions across supply networks, Manufacturing and Logistic Users need to define these interactions, providing information on the organisations they collaborate with and their role in the collaboration. Thus, the Organisation information form in Figure 8 needs to be filled in by every organisation member of the consortium requesting a new (single) vf-OS application.

Organisation	
Name	Provide the name of your organisation
Description	Provide a short description of your organisation, sector, activity, main products, and services
URL [Optional]	Provide a URL of your website, if available
Supply chain collaborators [Optional]	Name other organisations you need to interact with in the application
Supply chain roles [Optional]	Define your role in the interactions (eg supplier, customer)

Figure 8: Organisation information form

Manufacturing and Logistic Users need to define Actors: persons that interact with the application as described in Section 1. Actors should help developers to better understand the users and identify functional and non-functional requirements related to the user experience. Actors should also help in identifying existing applications or devices that need to be integrated with the vf-OS platform. Figure 9 describes the form used to define Actors.

Actor				
Name		**Device preference**	Laptop	(None, Low, High)
Age [Optional]			Tablet	(None, Low, High)
Occupation			Smartphone	(None, Low, High)
Company			Other	(Which one)
Location		**Application use**	(Use of legacy software in daily jobs) (None, Low, High)	
Needs (Describe in a couple of paragraphs the current job of this Actor and highlight why he needs the application)				
Top goal Define in a sentence the main goal for the application according to this Actor				
Prioritised key features Numbered list of features the Actor expects ordered by priority				
Applications or devices owned and used Add names and logos of applications or devices that the Actor uses and is currently used to manage data related to the objectives of the requested vf-OS application				

Figure 9: Actor definition form

As described in Section 1, the first step in the proposed methodology is the definition of the Goals: High level objectives for the application. To that end, Manufacturing and Logistic Users need to define the technical objectives, expected benefits, and impact of the application. Thus, each application consists of several Goals. Figure 10 shows the form used to define them.

Goal	
(Describe the Goal with a brief sentence)	
Who is it for?	Name the Actors this Goal is directed to
What problem does it solve?	Describe in a couple of sentences the problem in your organisation or supply chain that you want to solve with this application
Indicators [Optional]	Optionally define some measurable indicators that can be used to determine whether you have reached the Goal or not
Inspiration [Optional]	Add UI sketches that can help developers understand what you expect from this application

Figure 10: Goal definition form

Goals are divided into Activities, which define the steps taken by Actors using the application to complete the Goal. Activities need to show interactions with other applications, devices, or external services. The Activity form is depicted in Figure 11.

Activities	
(Describe the activity with a brief sentence)	
Actor	Name the actors that conduct this activity
Description	Describe in a couple of sentences the activity and how actors interact with the application as well as with other applications and devices
Details	Provide additional information, needs, or expectations related to this activity
Interfaces [Optional]	Describe briefly data Inputs and Outputs related to this activity
Inspiration [Optional]	Provide vf-OS workflow diagrams or UML activity Diagrams describing the activity

Figure 11: Activities definition form

Finally, Manufacturing and Logistic Users need to clarify terms used throughout the forms that are specific to their activity and Software Developers may not understand. These terms are included in the Terminology form. Obviously, Software Developers may require further clarifications and this is covered by future steps of the methodology. Nevertheless, an initial effort to clarify the terminology used is required at this point to help Software Developers understand the Use Case, using the form in Figure 12.

Terminology	
Term **[Optional]**	Definition of the term

Figure 12: Terminology definition form

2.3 Application Questionnaire

The questionnaire in Figure 13 consists of ten questions designed to reinforce understanding of the application needs and requirements. Basically, the questionnaire explores which high level vf-OS architecture components and subcomponents are needed (eg input-output, middleware, data storage, data analytics), interactions with common external services (user authentication, document management, etc.) and important non-functional requirements (security and UI quality). This information can be useful for Software Developers to filter requests according to their expertise and to get a rough estimate of the requirements, enough to express interest in the application request. The requested information is translated into questions that make sense to users.

Questionnaire			
1. What kind of login do you want for your application? (Login is required to identify who is interacting with your application)			
Email	**Other service (eg Google or ActiveX)**	**No login**	**I don´t know**
2. Do users need personal profiles? (The application needs to behave in different ways depending on who is using it)			
Yes		**No**	**I don´t know**
If yes, can you provide some details?			
3. Do you need support for collaboration? (Partners from different organisations are going to collaborate through the application)			
Yes		**No**	**I don´t know**
If yes, can you provide some details?			
4. Does your application need to connect with legacy devices? (This means you will need Drivers to connect to devices like production machines)			
Yes		**No**	**I don´t know**
If yes, can you provide some details?			
5. Does your application need to connect with legacy services? (This means you will need APIs to connect to systems and applications like ERP, MES, CRM)			
Yes		**No**	**I don´t know**
If yes, can you provide some details?			
6. Are connected services or devices distributed among collaborators? (This means your application will need to be connected to systems in different organisations)			
Yes		**No**	**I don´t know**
If yes, can you provide some details?			
7. Does your application need to manage large amounts of data? (The application needs infrastructure to securely manage data from connected devices and applications)			

Yes	No	I don´t know
If yes, can you provide some details?		
8. Does your application need to analyse data? (You want your application to help you discover useful information)		
Yes	**No**	**I don´t know**
If yes, can you provide some details?		
9. Does your application need to manage documents? (The application needs to handle different text or multimedia documents)		
Yes	**No**	**I don´t know**
10. How should your application look? (Provide an indication of the quality expected for your user interface)		
Bare-bone	**Stock**	**Beautiful**

Figure 13: Application questionnaire

2.4 Sketches and Mockups

This section describes the methods and guidelines used to prepare UI mockups and how they will be used to further develop the UI of pilot vApps and, by extension, how mockups can be helpful in the development of vApps after the project. The use of UI sketches and mockups has been very helpful to develop the concepts and expected functionalities of the vApps in the characterisation of user scenarios in WP1. These guidelines are based on the experience gained in the preparation of the sketches and mockups prepared for the pilots vApps and included in D1.2 Annex.

Mockups are design tools that help developers test UI design and usability performance before starting the development of vApps. As explained in Section 1.4, in vf-OS, Manufacturing and Logistic Users attach UI sketches to the application request to indicate expectations on the vApp user interface. In the Specification Phase, Software Developers can interact with Manufacturing and Logistic Users to elaborate and validate these concepts and prepare more accurate UI mockups, which in turn will be the starting point for UI prototyping. This process is highlighted in Figure 14.

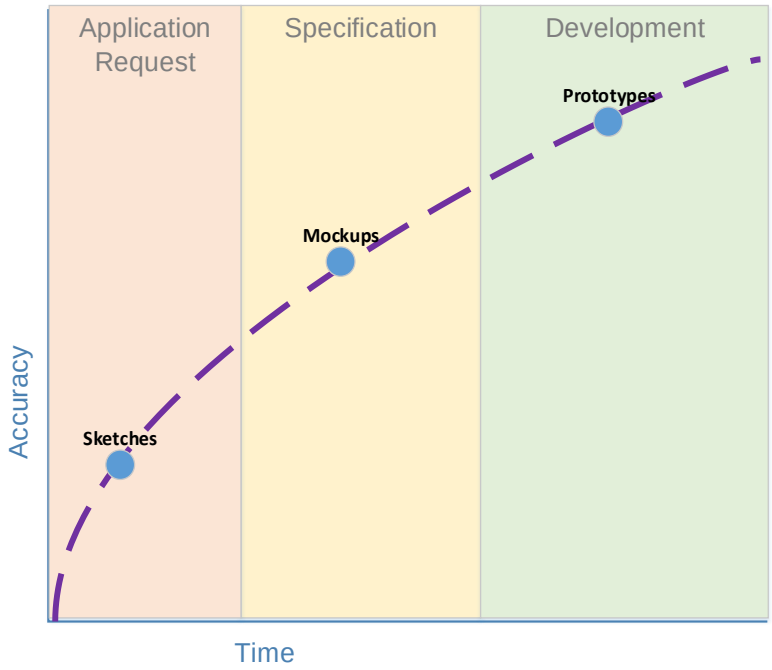


Figure 14. UI Development and design with mockups in vf-OS

Thus, in vf-OS, the main objective of UI sketches and mockups is to capture ideas and expectations on the UI from Manufacturing and Logistic Users and get fast and meaningful feedback from Software Developers in the same format. With the guidelines provided, it is expected that users can easily prepare initial sketches to present their ideas on functionality and interfacing in a graphic format. At this point, it is important to give Manufacturing and Logistic Users freedom to express themselfs. Thus the guidelines for the preparation of sketches only contain basic indications on how the UI sketches can be structured and what kind of additional information should be prepared. The main guidelines for Manufacturing and Logistic Users to prepare mockups are:

- **Use any drawing tool you are comfortable with:** Users should be free to use any tool of their preference to prepare UI sketches or mockups. UI sketches and early prototypes can be prepared with any drawing tool, but there are different free prototyping tools to create user interface mockups that are very easy to use and do not require expert knowledge on UI design. For instance, Marvelapp[3] and Balsamiq[4] are very user-friendly, multi-platform, free tools that can be used to draw user interfaces on demand. Marvelapp allows users to add links, gestures, or transitions to the mockups. Additionally, mockups can be done for desktop, mobile, or other devices like the Apple Watch. On the other hand, Balsamiq provides a palette where Users can find common UI elements like buttons, icons, and data tables. Users can browse the palette to select an element and drop it to the canvas to place it in a mock-up. Users can create links from elements to other wireframe, to define and test the main navigation properties of the application. Balsamiq elements look like handmade sketches by default. This reinforces the notion that the mock-up is still a concept and not a real application. Figure 15**Error! Reference source not found.** shows a screenshot of a vf-OS prototyping project with Balsamiq.

[3] https://marvelapp.com/
[4] https://balsamiq.com/products/mockups/

Figure 15 – Balsamiq tool screenshot

- **Focus on the concept you want to explain:** UI sketches are meant for helping Manufacturing and Logistic Users to articulate their main concerns with interaction but they do not need to produce sketches for every screen or dialog to be implemented in the vApp. Also, sketches are very early design concepts, so Manufacturing and Logistic Users should focus on the concepts they see critical and not provide too much detail.
- **Use existing UI toolkits and standard elements:** In the preparation of UI sketches and mockups, it is important to search for premade toolkits with UI elements for the tool being used. Most prototyping tools like Balsamiq or Mainframe provide standard UI elements such as buttons or icons by default. If Manufacturing and Logistic Users want to use other drawing tools, then it is important to search for a library with premade elements since it will save a considerable amount of work and will help Software Developers understand better the design concepts.
- **Take advantage of comments:** Manufacturing and Logistic Users can add comments to the sketches to provide additional information about the functionality or the usability of UI elements. Most prototyping and drawing tools provide graphic elements to add comments, so Manufacturing and Logistic Users should use it as they see fit.
- **Capture navigation aspects:** UIs are not static and in this sense, navigation is an important aspect of UI design. In prototyping tools such as Balsamiq, you can create several mockups or wireframes in the same project and easily set links between them. Otherwise, if Manufacturing and Logistic Users are using drawing tools that do not implement this feature, they can insert arrows and comments to capture navigation aspects.
- **Use device elements:** Device elements represent devices like smart phones and tablets and can be used and combined in sketches to indicate device preference and to represent preference on relevant design aspects like scale of components or font sizes. They also provide a framework to organise elements that is helpful in the preparation of UI sketches.

Figure 16**Error! Reference source not found.** shows an UI sketch from one of the pilot use cases following these guidelines.

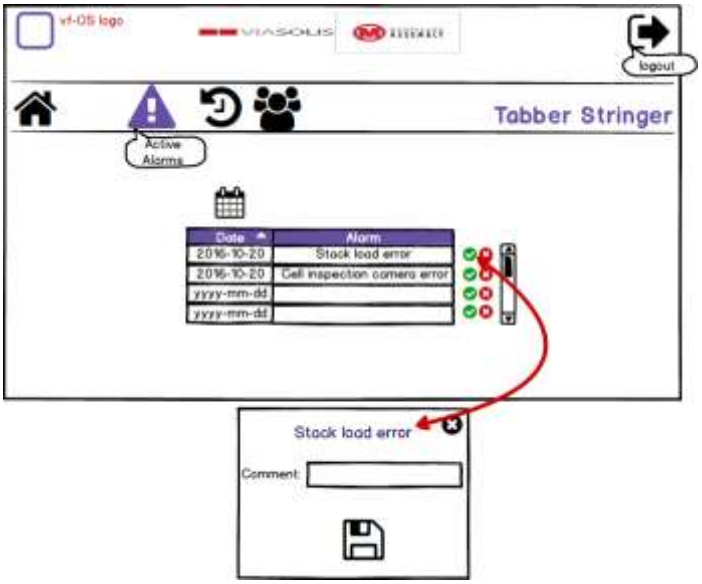


Figure 16. Sample pilot 1 mock-up

Software Developers, on the other hand, must have total freedom to prepare the final version of the mockups to be included in proposals. Since they are familiar with the vf-OS Platform, they can use sketches from Manufacturing and Logistic Users as input to their designs and apply their knowledge of the vf-OS Platform to properly adjust them to the design constraints imposed by vf-OS components functionalities or to harmonise the design of the vApp with the vf-OS Frontend Environment. Internally, Software Developers can use any design process and any design tool they want to use to work from sketches to mockups to prototypes. It is possible to define other intermediate stages and there is no requirement on the required fidelity or accuracy for mockups specified in this methodology, if proposals are accepted by Manufacturing and Logistic Users.

2.5 Workflow Diagrams

The following guidelines provide a standard format for Manufacturing and Logistic Users and Software Developers to prepare workflow diagrams for vf-OS. In the preparation of workflow diagrams for the use cases characterisation described in D1.2, Manufacturing and Logistic Users tended to use different graphic elements to describe the same components. Even if there were only 5 different users involved in the use case characterisation, this made it difficult for readers to understand and follow the different workflow diagrams, so it was decided to use a standard format for all user diagrams. If vf-OS Software Developers need to process many different application requests from several Manufacturing and Logistic Users, they will face similar problems and establishing a standard format for workflow diagrams would be beneficial for them. Thus, the main objective of this standard format is to help Software Developers interpret many separate application requests faster. If separate Manufacturing and Logistic Users use the same formats and guidelines to prepare the workflows, the information is more uniform across application requests and thus easier to process.

Figure 17 lists the different icons defined for each high-level component of the application, as well as for other environment elements (eg Actors, Applications, Factories, or Logistics) that can be useful for the preparation of workflow diagrams. The icon set uses the free

Font Awesome[5] open font and toolkit, so Manufacturing and Logistic Users and Software Developers can extend it if necessary for a specific use case.

Icon	Component
	**Analytics** Analyses data to guide decisions
	**Storage** Stores data for later use
	**Transformation** Transforms data
	**Messaging** Facilitates communication between services and applications
	**Security** Provides secure communications and operations
	**System Dashboard** Manages applications and data
	**IO** Connects legacy services, software, and devices to vf-OS
	**Machine** Machine to be connected to vf-OS
	**Application** Application to be connected to vf-OS
	**Transport** Transport of goods or materials
	**Warehouse** Storage facility
	**Actor** vApp user
	**Factory** vApp manufacturing facility
	**vf-OS Platform** **Encapsulates all vf-OS services**

Figure 17: Component icons for workflows

It is worth noting that the formats and icons used in these guidelines are somewhat different to the formats used in the workflow diagrams in D1.2. In WP8 it was found that the workflow diagrams in D1.2 were very (too) detailed for an initial application request.

[5] https://fontawesome.io/

Manufacturing and Logistic Users are expected to generate simpler workflow diagrams to support the information provided in the application request forms before they interact with Software Developers. Additionally, the format needs to support any vf-OS use case and be applicable outside the scope of the project pilots and domains.

Figure 18 – Workflow diagramFigure 18 shows the guidelines for the preparation of workflow diagrams. All workflow diagrams in application requests and proposals should follow these guidelines:

- **Standard icons:** Workflow diagrams use the standardised vf-OS icon set (presented in Figure 17) to describe the different elements in the workflow
- **Element labels:** Elements have labels to be able to refer to them
- **Numbered arrows**: Elements are connected to each other with numbered arrows. An arrow represents an interaction between elements of the diagram. The direction of the arrow represents the direction of the interaction from data producer to data receiver
- **Caption text:** A brief caption above the arrow describes the interaction
- **Legend:** If needed, diagrams can incorporate a legend to provide more detailed descriptions of interactions
- **Labelled containers:** Boxes are labelled with an acronym and an icon that represents the owner organisation of the container
- **Boxed elements:** Elements can be placed into Labelled containers to indicate relationships like location (eg a machine inside a Factory or a user related to a logistics operation).
- **Additional notes:** Users can include additional text information in notes

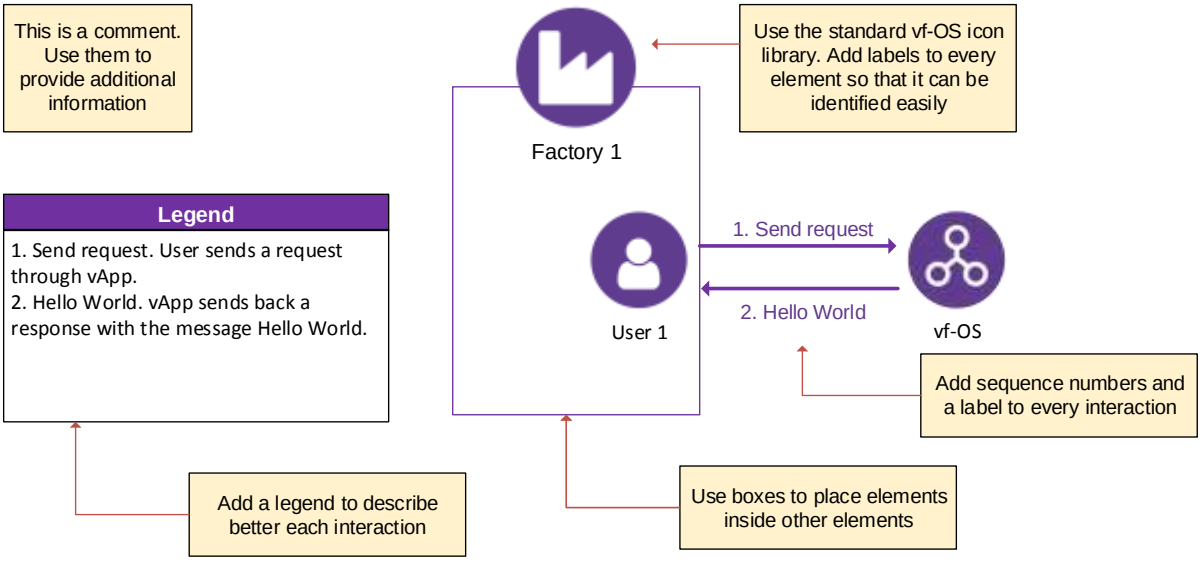


Figure 18 – Workflow diagram guidelines

Following the guidelines, Figure 19 shows a sample workflow diagram based on one of the use cases defined for pilot 1.

Legend
1. Collect Data. Tabber Stringer machine data is uploaded to the platform. 2. Generate alarm. Data analytics detects an event that could cause a failure and generates an alarm 3. Send Alarm. The application generates a notification for the user 4. Preventive intervention. The Maintenance Manager checks the machine and collects more information. 5. The user enters the information into the system.

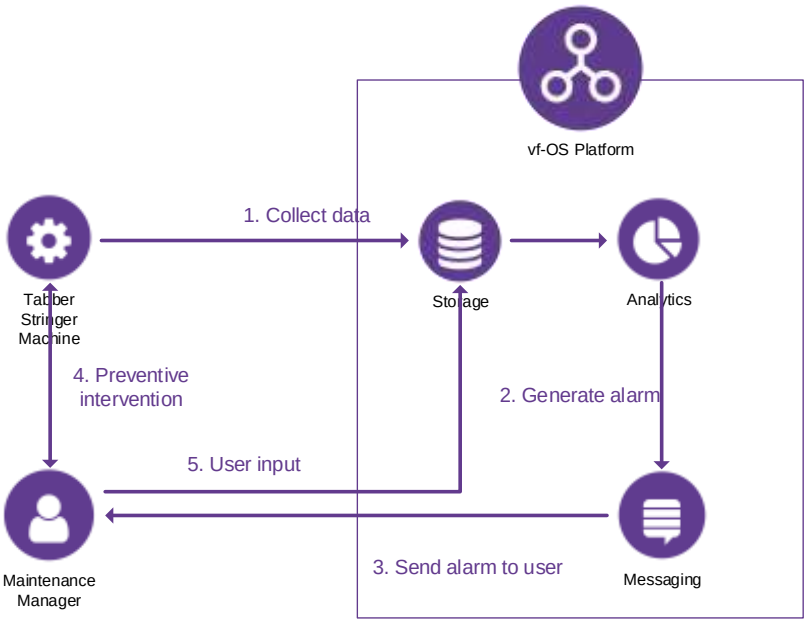


Figure 19 – Sample pilot 1 workflow diagram

3 Monitoring, Reporting, and Evaluation Methods

This section describes the Monitoring, Reporting, and Evaluation Methods. Section 3.1 describes the GQM Goals in the context of vf-OS and the pilots. Section 3.2 defines the Generic GQM Specification Questions. Section 3.3 contains the Key Performance Indicators (KPIs) and Section 3.4 describes pilot specific indicators and assessment criteria.

3.1 Goals

3.1.1 General Goals

Figure 20 lists general objectives common to any vApp. This list has been obtained from [YAA+13] and adapted to vf-OS, covering the eight Evaluation Modules (EMs) defined in ISO/IEC 25040.

ID	vApp General Goals
GA1	Time frame needed to develop the vApp
GA2	Expected functionalities of the vApp
GA3	Advantage of the vApp compared to existing alternatives
GA4	Rationale of having the vApp (Cost/Popularity/Technology)
GA5	Education level of the vApp Users and support staff
GA6	Level of training given related to the software
GA7	Level of adaptation to current trends (technology/devices/user interfaces)
GA8	User satisfaction level in using the software
GA9	Level of software portability
GA10	Reliability of the software

Figure 20: General vApp objectives

3.1.2 Pilot Specific Goals

Besides these general objectives, every vApp will have a set of specific application level objectives that will be identified in the Initial Specification and Development phase using the Use Case Description Template and specification described in Section 2. The following subsections list the different high-level functional objectives defined for the different pilot use cases. These high-level objectives will determine the acceptance criteria for the use cases. The list of specific objectives for each pilot will be completed at the end of the use case definition phase, as will be explained in Section 4.

3.1.2.1 Pilot 1 specific high level objectives

Figure 21 gathers the specific high-level objectives for the use cases in pilot 1.

vApp	ID	vApp Goal
vfFailurePrevention (vApp P11)	**GA111**	Automatic detection of equipment maintenance
	GA112	Automatic detection and notification of failure events
	GA113	Automatic trigger of monitoring processes
vfFailureManager (vApp P12)	**GA121**	Automatic registration of the failures of the Tabber Stringer
	GA122	Manual fault creation from the equipment owner
	GA123	Historical record of registered faults and applied solutions
vfStockPolicies (vApp P13)	**GA131**	Manage spare-parts needs
	GA132	Monitor spare-parts stocks
	GA133	Alert when the stock is off limits for each spare-part
vfProductionFollowUp (vApp P14)	**GA141**	Collect selected production PLC data from Tabber Stringer machine in real time
	GA142	Store production data
	GA143	Calculate production KPI's such as productivity, availability, performance, or quality
vfMaintenanceCalendar (vApp P15)	**GA151**	Schedule a maintenance calendar for the Tabber Stringer machine
	GA152	Support the communication of maintenance operations between MASS and VS
	GA153	Update the schedule maintenance calendar accordingly

Figure 21: Pilot 1 vApp objectives

3.1.2.2 Pilot 2 specific high level objectives

Figure 22 lists the specific high-level objectives for pilot 2.

vApp	ID	vApp Objective
vfDocumentPortal (vfApp P21)	**GA211**	Edit internal documentation related to a construction site
	GA212	Store internal and external documentation related to a construction site
	GA213	Access all the relevant documentation regarding the construction site lifetime operations
vfSteelValidation (vApp P22)	**GA221**	Visually identify steel bars ID by inspection of the bar
	GA222	Visually identify steel bars ID by inspection of the tag
	GA223	Validate the steel bar
vfOnSiteManager (vApp P23)	**GA231**	Allow the Constructor to report internal and external delays
	GA232	Assist the Supervisor in the rescheduling of his own supervision

		plan
	GA233	Define the impact level of the delay
vfConcreteFeedback (vApp P24)	**GA241**	Provide concrete slump[6] test results to Constructor
	GA242	Allow the definition of concrete slump test thresholds
	GA243	Alert Constructor to reject concrete accordingly
vfProductValidation (vApp P25)	**GA251**	Control the reception of construction products at construction site
	GA252	Approve the reception of construction validated products
	GA253	Submit the reception of construction products

Figure 22: Pilot 2 vApp objectives

3.1.2.3 Pilot 3 specific high level objectives

The specific high-level objectives for pilot 3 are detailed in Figure 23.

vApp	ID	vApp Objective
vfCollaborationAnalyser (vApp P31)	**GA311**	Process project submission (STEP) files
	GA312	Define a classification of business opportunities
	G3213	Automatically classify business opportunities in STEP files
vfIndusEnabler (vApp P32)	**GA321**	Process project submission (STEP) files
	GA322	Create user-friendly communication channels
	GA323	Automatically synthesize project features based on collected data
vfProductionPlanner (vApp P33)	**GA331**	Access production plans
	GA332	Generate coordination actions between companies sharing production of an item
vfQualityAssurance (vApp P34)	**GA341**	Access production plans
	GA342	Analyse production environment to detect anomalies
	GA343	Inform production managers accordingly
vfProductionTracker (vApp P35)	**GA351**	Access production plans and status
	GA352	Alert production managers about deviations
	GA353	Suggest actions to mitigate the unfavourable forecast

Figure 23: Pilot 3 vApp objectives

3.2 Generic GQM Specification Questions

As explained in Section 1.4.3, in the evaluation phase, a questionnaire related to the different objectives of the vApp should be filled in by Manufacturing and Logistic Users and Software Developers when the development of a vApp is completed. Figure 24 shows the base user questionnaire containing the questions defined to evaluate the general objectives of vApps, their relationship to the different objectives, and an indication of

[6] https://en.wikipedia.org/wiki/Concrete_slump_test

whether the requested information may be available in the vf-OS Platform (Platform) or needs to be provided by Manufacturing and Logistic Users or Software Developers.

Generic GQM Specification questions		
Q1. **Name of the vApp that you required** (**Platform**)		
Q2. **Which software company developed your App?** (Platform)		
OA1: Time frame needed to develop the vApp		
Q3. **How long did it take to develop the vApp?** (Platform)		
Q4. **How long did it take to develop each vApp component? (list)** (Developer)		
Q5. **How long did it take from the first delivery to the final release?** (Platform)		
Q6. **How many iterations took place between the first release and the final release?** (Platform)		
OA2: Expected functionalities for the vApp		
Q7. **Were requirements revised (changed/removed/improved) during the development?** (User)		
Q8. **Did the vApp integrate the required functionalities?** (User)		
Yes	**If no, why?**	**I don´t know**
OA3: Advantage of the vApp compared to existing alternatives		
Q9. **Have you already used similar products from other suppliers?** Name advantages/disadvantages compared to the proposed solutions (User)		
OA4: Rationale of having the software vApp (Cost/Popularity/Technology)		
Q10. **Are additional expenses required to use the application (staff training, new equipment etc.)?** (User)		
Q11. **Do you think the vApp helped the company?** (User/Developer)		
Yes, in which sense?	**If no, why?**	**I don´t know**
OA5: Education level of Users and support staff		
Q12. **Does the vApp require a specific education level from Users and support staff?** (User)		
OA6: Training given related to the software		

Q13. **Is the current staff able to use the new product without additional training?** (User)
OA7: Level of adaptation to current trends (technology/devices/user interfaces)
Q14. **Are the technologies used complying with modern trends and standards?** (Developer)
OA8: Level of satisfaction on using the software
Q15. **Would you recommend the vApp to other companies, like your customers or suppliers?** (User)

Yes	If no, why?	I don´t know

Q16. **On a scale of one to ten, how would you rate your satisfaction with the vApp?** (Platform)

1	2	3	4	5	6	7	8	9	10

Q17. **On a scale of one to ten, how would you rate your overall satisfaction with vf-OS?** (Platform)

1	2	3	4	5	6	7	8	9	10

OA9: Level of portability
Q18. **Is the vApp platform independent?** (Developer)

Yes	If no, why?	I don´t know

OA10 Level of reliability
Q19. **Does the vApp check whether the data provided by users is correct?** (Developer)
Q20. **Does the vApp provide mechanisms provided in case of application failure?** (Developer)

Figure 24: Base evaluation questionnaire

The base evaluation questionnaire will be used by evaluators to conform the evaluation template delivered to Manufacturing and Logistic Users and Software Developers in the evaluation phase, as described in Section 1. The final questionnaire will contain use case specific questions to evaluate the specific goals of each application.

3.3 Key Performance Indicators

This section presents the methodology to evaluate the development and the performance of vApps. The ultimate goal of this evaluation procedure will be to generate Key Performance Indicators (KPIs) that define the success rate or the approval or rejection of the different steps defined in the use sequence of the vApp (eg story map steps) and the vApp as a whole.

By definition, KPIs are high-level indicators of the evaluation of selected instances participating in the process. Those components can include user or developer requirements along with metrics from software components' evaluation, in this case the vApps used in the execution of every step. Generally, KPIs are evaluated in several executions of a process, based on a predefined expected value for the metric and a certain deviation from the average for the vApp to be accepted.

Additionally, it is important to note that KPIs can be weighted and merged into other indicators encapsulating the outcome of several indicators. KPIs, as explained below, can have different natures and temporal enrolment but, in the end, provide metrics to evaluate process steps and characterise the overall process.

KPIs for software development processes can be classified as follows:

- **Process KPIs:** Measure the efficiency or productivity of a business process. They can be further classified into:
  - Long-term KPIs:
    - Average time KPI. Average time for each process
    - Number of steps required for each process (decomposition)
  - Short term KPIs:
    - Average time used for each step in the process
- **Input KPIs:** Measure assets and resources invested in or used to generate business results
- **Outcome KPIs:** Measure the overall results or impact of the business activity in terms of generated benefits, as a quantification of performance
- **Qualitative KPIs:** Measure the customer satisfaction
- **Quantitative KPIs:** Measure a set of characteristics essential to vApp success

The different indicators used in the evaluation of pilots and in general vApps will be defined in their corresponding evaluation phase. However, a set of indicators encapsulating the overall outcome of each ISO EM are provided in Figure 25. As described in the Figure, the steps and unit tests defined in story maps can provide useful input in certain situations for the definition of KPIs to evaluate the functionality of vApps. In this sense, automated unit testing, a common practice in agile software development, can reduce significantly the effort required to collect data for the calculation of KPIs. By monitoring KPIs, it is possible to take immediate action to adjust the agreed development plan in order to guarantee the success of the vApp.

EM1 Functionality		
Name	Description	Expression
Functionality Metric of each Scenario Step **(FMSS)**	Number of successful step executions ($TSS_{success}$) divided by the Total number of Executions (TSS_n)	$FMSS = \frac{TSS_{success}}{TSS_n}$
EM1 Functionality Metric of the Pilot **(FMT1)**	Average of the FMSS over the total number of scenario steps (SS_n)	$FMT1 = \frac{\sum_{i=1}^{n} FMSS_i}{SS_n}$
EM2 Functionality (handle errors)		
Name	Description	Expression
EM2 Functionality Metric of each Scenario Step (handle errors) **(FMSS2)**	Number of successful executions of unit tests handling specific, pre-established errors ($UTSS_{success}$) divided by the total number of unit test executions ($UTSS_n$)	$FMSS = \frac{UTSS_{success}}{UTSS_n}$
EM2 Functionality Metric of the Pilot **(FMT2)**	Average of FMSS2 over the total number of scenario steps (SS_n)	$FMT2 = \frac{\sum_{i=1}^{n} FMSS2_i}{SS_n}$
EM3 Reliability		
Name	Description	Expression
EM3 Reliability for each Scenario Step **(RMSS)**	Number of faults (F) detected when performing pilot scenario step executions divided by the total number of executions (TSS_n)	$RMSS = \frac{F}{TSS_n}$
EM3 Reliability Metric of the Entire pilot **(RMT)**	Average of RMSS over the total number of scenario steps (SS_n)	$RMT = \frac{\sum_{i=1}^{n} RMSS_i}{SS_n}$
EM4 Usability		
Name	Description	Expression
EM4 Usability Metric for each Scenario Step **(UMSS)**	The user interface of the software is evaluated in accordance with the software related parts 12 to 17 of IS0 9241 [ISO+01], which are respectively related to:1) Presentation of information; 2) User guidance; 3) Menu dialogues; 4) Command dialogues; 5) Direct manipulation dialogs; 6) Form filling dialogues. The usability metric (UMSS) for each scenario step will be the number of successful Answers ($A_{success}$) divided by the number of Check Lists (CL_n)	$UMSS = \frac{A_{success}}{CL_n}$

EM4 Usability Metric of the Pilot **(UMT)**	Average of UMSS over the total number of scenario steps (SS_n)	$UMT = \frac{\sum_{i=1}^{n} UMSS_i}{SS_n}$
EM5 Efficiency		
Name	Description	Expression
EM5 Efficiency Metric for each Scenario Step **(EMSS)**	Sum of Efficiency Answers scores (AU) to the Efficiency questions given to classify the efficiency of the scenario step (U_n is the total number of questions)	$EMSS = \frac{\sum_{i=1}^{n} AU_i}{U_n}$
EM5 Efficiency Metric of the Pilot **(EMT)**	EMSS of each scenario step divided by the total of scenario steps (SS_n)	$EMT = \frac{\sum_{i=1}^{n} EMSS_i}{SS_n}$
EM6 Maintainability		
Name	Description	Expression
EM6 Maintainability Metric of each Scenario Step **(MMSS)**	Sum of Maintainability Answers scores (MA) to the Maintainability questions given to classify the efficiency of the scenario step (MQ_n is the total number of questions)	$MMSS = \frac{\sum_{i=1}^{n} MA_i}{MQ_n}$
EM6 Maintainability Metric of the Pilot **(MMT)**	Average of MMSS over the total number of scenario steps (SS_n)	$MMT = \frac{\sum_{i=1}^{n} MMSS_i}{SS_n}$
EM7 Portability		
Name	Description	Expression
EM7 Portability Metric of each Scenario Step **(PMSS)**	Average of the component portability metric (PMC) over the total number of step components (C_n)	$PMSS = \frac{\sum_{i=1}^{n} PMC_i}{C_n}$
EM7 Portability Metric of the Pilot **(PMT)**	Average of PMSS over the total number of scenario steps (SS_n)	$PMT = \frac{\sum_{i=1}^{n} PMSS_i}{SS_n}$
EMIV Scenario Step Information Validation		
Name	Description	Expression
EMIV Efficiency Metric of the Pilot **(EMIV)**	This evaluation process takes into account the different vApps in every pilot. The measurement of the information validation of each pilot (EMP) will be the number of successful addressed characteristics (project scenario requirements) divided by the total number of such characteristics (CH_n)	$EMIV = \frac{\sum_{i=1}^{n} EMP_i}{CH_n}$
EMIV Functionality Metric of the Pilot **(EMIVP)**	Average of EMIV over the total number of scenario steps (SS_n)	$EMIVP = \frac{\sum_{i=1}^{n} EMIV_i}{SS_n}$

Figure 25: Generic indicators

3.4 Pilot Specific Indicators

The project DOA already established some metrics and target values for each pilot that need to be included and assessed in the evaluation of the different use cases. The following subsections include a definition and a brief description of the calculation of each indicator, together with the tables that will be used to report these indicators.

3.4.1 Pilot 1: Manufacturing & Logistic – Automation

The evaluation of use cases in pilot 1 must include the following list of indicators:

- **Reduction of the average spare-parts replacement time after failure (%):** To calculate this indicator, the time passed from failure detection until the validation of the failure solution will be measured and specified before the pilots and during the pilots. The indicator is calculated as the difference relative to the AS-IS replacement time.
- **Average downtime loss due to spare-parts failure (%):** This indicator measures the loss in production time due to spare-part failures. The indicator is calculated as follows:

  $$t_{spare\text{-}part\text{-}failure}/(t_{production} + t_{spare\text{-}part\text{-}failure})$$

  where $t_{spare\text{-}part\text{-}failure}$ is the overall downtime due to spare part failure in the evaluation period and $t_{production}$ is the overall production time in the evaluation period.
- **Reduction of spare-parts stock levels (%):** The stock level will be measured in terms of cost of materials and compared to the initial value before the pilot starts
- **Reduction of spare-parts sales (%):** Measurement of the total amount of spare parts sales to the customer before and during the pilot development
- **Replenishment costs (shipping and logistics):** This indicator measures the total cost of shipping and logistics. In some situations, especially when there are uncertainties about the exact cause of failure, some extra parts are sent just in case directly from the spare part provider to the customer.
- **Overtime costs reduction due to spare-parts failures (%):** The measurement of this indicator will be performed by calculating the cost of production overtime after machine failure. The cost of production overtime is defined as the costs associated to the production overtime needed to recover the production lost due to failures.

Figure 26 indicates whether the expected impact of each pilot vApp in each indicator is low, medium or high.

	vApp				
Indicator	**P11**	**P12**	**P13**	**P14**	**P15**
Average spare-parts replacement time after failure	Medium	High	Low	Low	Low
Average downtime loss due to spare-parts failure	High	Medium	Low	High	Low
Spare-part stock levels	Medium	Medium	High	Low	Medium
Spare-parts sales	Low	Low	Medium	Low	Medium
Replenishment costs (shipping and logistics)	Low	Low	Medium	Medium	High

Costs for production overtime due to spare-parts failure	High	Medium	Low	High	Low

Figure 26: Expected impact of pilot 1 vApps on indicators

3.4.2 Pilot 2: Construction – Industrialisation

The following indicators must be included in the evaluation of pilot 2:

- **Time reduction/increase for accessing construction project data (%):** This is a measure of efficiency gains from using the vApp as compared to the normal current procedure. This will be expressed as a percentage of the time required to access documents using the current process and will be calculated as the ratio between the time required to access documents in the vApp and the time to access physical documents in construction projects, times 100. Values over 100% mean that accessing documents using the vApp is more time consuming than through the current procedure. A significant time reduction for accessing documents is expected. This indicator evaluates the access to construction project data, the adaptation to various construction projects and the assessment of construction process through access to global project data.
- **Time reduction/increase spent on the set-up of a new construction supervision project (%):** This is a measure of efficiency gains from using the vApp. The setting up of a construction supervision service is time consuming and involves the mobilisation of various contributions; it is expected that this is improved using the vApp. This will be expressed as a percentage of the time required to set-up a supervision project using the current process and will be calculated as the ratio between the time required to set-up the services as is currently done and the time required for the same purpose using the vApp, times 100. Values over 100% will mean that setting up a new supervision project using the vApp is more time consuming than through the current procedure. This indicator evaluates the construction project development and the adaptation to various construction projects.
- **Time spent to identifying steel ratio (%):** Acceptance of steel bar lots involves the visual inspection of the bars to assess if the bar code engraved on each of them corresponds to that of the approved manufacturer. The vApp is expected to allow this reading to be automatic. Thus, the indicator will be the ratio between the time required to identify a steel bar lot using the visual tools of the vApp and the time needed to manually and visually identify a steel bar lot, expressed as a percentage. Values over 100% will mean that identifying steel bars using the vApp is more time consuming than through the current procedure.
- **Slump test result Variability (%):** The slump test is done upon a concrete truck's arrival at the work site and defines its acceptance or rejection, through measurement of concrete consistency. The indicator will be the ratio of a and b where a) is the variability of the Slump test results, using a suitable sample size group, in the pilot using the vApp, and b) is the variability of the Slump test results, using a suitable sample size group. This is relevant to the AS-IS procedure, expressed as a percentage. This will be a measure of efficiency on process control.
- **Time spent on rescheduling the supervision plan when delays or failures occur (%):** When delays or failures occur, the works supervision team and the works contractor may spend some considerable time in rescheduling the work to minimise the effects of the delay or failure. It is expected that the vApp can contribute to reducing this time. Thus, the indicator will be the ratio between the time spent to

reschedule the supervision plan, following the occurrence of a delay/failure, with the vApp during the project pilots, versus the AS-IS procedure, expressed as a percentage. This indicator evaluates the adaptation of construction project changes and the rescheduling time when delays or failures occur.

- **Time spent on data exchange between stakeholders in the product acquisition process (%):** The circulation of information concerning reception, approval or rejection of materials, requires some formality but its time consuming. It is expected that the use of the vApp can introduce some efficiency improvements, without losing the formalism. This is the ratio between the time spent to fill and exchange necessary documents between the different stakeholders, during the product acquisition process, using the vApp versus the AS-IS procedure, expressed as a percentage.

Figure 27 shows the expected impact of each pilot vApp in each indicator.

	vApp				
Indicator	**P21**	**P22**	**P23**	**P24**	**P25**
Time spent on accessing construction project data ratio	High	Low	Low	Low	Low
Time spent on the creation of construction work supervision projects ratio	High	Low	Low	Low	Low
Time spent on identifying steel ratio	Low	High	Low	Low	Low
Slump test result variability	Low	Low	Low	High	Low
Reduction of the time spent on rescheduling when delays or failures occur	Low	Low	High	Medium	Low
Time spent on data exchange between stakeholders in the product acquisition process ratio	Low	Low	Low	Low	High

Figure 27: Expected impact of pilot 2 vApps on indicators

3.4.3 Pilot 3: Manufacturing Assembly

The following list describes the different indicators that must be included in the evaluation of Pilot 3 use cases:

- **Time spent on identifying the typology of rejected projects (hours):** The difference between the time of project classification and the time of rejection. This time is specific for each product and an average per product will be used for the evaluation of the pilot.
- **Measurement of delivery period: additional delays for complex products (%):** The deviation of the actual delivery date of the project compared with the planned delivery date specified in the ERP. This metric is also specific for each product and thus, an average per product will be used for the evaluation of the pilot.
- **Time spent on confirming project acceptance for processing (days):** The time spent by the consortium evaluating a project proposal and accepting the project for processing or rejecting it.
- **Number of inter-sites bottlenecks in manufacturing (operations):** The number of inter-sites transfer operations in a manufacturing sequence. This indicator is also product-specific and an average will be used for the pilot.

- **Global costs (manufacturing + logistic) of multi-site products (€):** Costs of final product (from ERP) + costs of logistic (warehousing + transfer). This indicator is specific for each product and will be averaged for the pilot evaluation.

Figure 28 lists the expected impact of vApps on the different indicators.

	vApp				
Indicator	**P31**	**P32**	**P33**	**P34**	**P35**
Time spent on identifying the typology of rejected projects	High	Low	Low	Low	Low
Measurement of delivery period: additional delays for complex products	Low	High	Low	Low	Low
Time spent on confirming project acceptance for processing	Medium	High	Medium	Low	Low
Number of inter-sites bottlenecks in manufacturing	Low	Low	High	Low	Low
Global costs (manufacturing + logistic) of multi-site products	Low	Low	High	High	High

Figure 28: Expected impact of pilot 3 vApps on indicators

3.5 vf-OS Environment Evaluation Form Template

The following evaluation form is an example generated using the methodology and applied to the vf-OS environment. The version generated for this deliverable is a draft version that will be updated in the next versions of it. In line with the definitions above, the form also includes ISO Evaluation Modules (EMs) specifically to evaluate functionality, reliability, usability, effectiveness, maintainability, and portability of software systems. Additionally, the EM IV section in this form integrates functional and non-functional requirements specific of vf-OS.

A consideration about this form is that it is designed to have the support of evaluators, users, and developers. Each one has their own role. The evaluators coordinate the process and calculate the result of the evaluation using the existent formulas for each EM, which may be adjustable in weight within the GQM analysis. This means some characteristics may have more importance and, consequently, higher weight than others, even though all are present in the evaluation procedure. The Manufacturing and Logistic Users must answer questions regarding the usability of the tool; Software Developers have to answer specific questions related to the maintainability of the application.

The eight EMs defined are outlined below but without specifying any importance weight in any of the evaluation characteristics because, as stated before, these weights can be adjusted in certain situations or scenarios. As an example, the EMs of one specific scenario step can have more weight than the other scenario steps of the same pilot.

Figure 29 shows an example Evaluation form template for the vf-OS Platform performance which could, for example, be used to capture some of the elements above.

Pilot Scenario Step Identifier:		**Actors involved:**	End Users	**Components involved:**			**To be answered by**: E – evaluators; D – developers; U - Users ()
Short Description	Evaluation vf-OS environment performance						
Short summary / Goal							
In this scenario, the objective is to examine whether the vf-OS Platform fulfils the key requirements.							
EM1 – Functionality	$TSS_{success}$		TSS_n		FMSS		E
Various inputs	**Expected Results**					**Passed (Y/N)**	
Register or creation of a user account	By creating an account, the user gets access to a personal cabinet where all user activities are represented						U
Publishing of new events	Event published: It is visible only by the proprietary user						U
Subscribing to new events	Event subscribed: It is mapped in the personal account						U
Inconsistencies	Event updated: It is visible only by the proprietary user						U
Access to the marketplace	Available applications are displayed in the user's account						U
Booking of desired application	From their personal account, users are able to book and pay for the applications using payment systems						U
Interpret and visualise the results	An interpretation of the results is given. It is visible only by the proprietary user in their personal cabinet						U
EM2 – Functionality (handle errors)	$UTSS_{success}$		$UTSS_n$		$FMSS2_{(he)}$		E
Various inputs with expected errors	**Expected errors – Define dialog messages and/or recovery stages**					**Passed (Y/N)**	
Send error messages	Messages informing about various types of failures are displayed						U
EM3 – Reliability	F		TSS_n		RMSS		E
Questions						**Answer (*)**	
Is the delivered data checked for correctness?							D
Are report mechanisms provided for the case of failures?							D
EM4 – Usability	A_{sucess}		CL_n		UMSS		E
System User Interface							
Characteristics	**Questions**					**Passed (Y/N)**	
Presentation of Information	Is the displayed information clear and understandable?						U
User Guidance	Is specific information promptly displayed when required by the user?						U
Menu Dialogues	Are the menu dialogues distinctive and descriptive?						U

Command Dialogues	Are key assignments functional and logical to related tasks?				U
Direct Manipulation Dialogues	If appropriate to task, can the user redirect, interrupt, or stop input/output upon request?				U
Form Filling Dialogues	In the occurrence of form filling errors, is it needed to re-enter the related data?				U
EM5 – Efficiency	$\sum AU / U_n$		U_n		E
Questions				**Answer (*)**	
Is the new system more efficient, compared to your old/traditional process?					U
EM6 – Maintainability	$\sum MA / MQ_n$		MQ_n		E
Questions				**Answer (*)**	
Are any recovery options available?					D
Is it easy to add a GUI able to manage the Users' activities?					D
EM7 – Portability (**)	$\sum PMC / C_n$		C_n		E
Questions				**Answer (*)**	
Is the vf-OS platform platform-independent?					D
EMI IV–SS Information Validation	$\sum EMP / CH_n$		$\sum EMIV / SS_n$		E
Functional requirements				**Passed (Y/N)**	
Is vf-OS able to publish its product catalogue to external partners? [MSB+17]					E
Did production partners receive detailed information about vf-OS catalogue items?					E
Are production partners informed in real time about changes happening on catalogue items?					E
Non-functional requirements				**Passed (Y/N)**	
Reveal and quantify the importance of each metric (setting weights) relating to the estimation of vf-OS environment efficiency					E
Tools involved are:	**vf-OS Platform**				E
Preconditions				**Passed (Y/N)**	
Are the data represented in appropriate form?					E
Post-conditions				**Passed (Y/N)**	
Is the platform system information interpreted and visualised appropriately?					E
(*) – Answer is a value from {bad, poor, fair, good, excellent) (**) – Degree of independence that a component has in relation to the operating system platform					

Figure 29: Evaluation form example template

4 Pilot Scheduling

Figure 30**Error! Reference source not found.Error! Reference source not found.** presents the plan detailing the development of pilot vApps addressing pilots' needs. Note that this is a preliminary version that can be updated in the upcoming versions of deliverable D8.1a, namely D8.1b, D8.1c, and D8.1d. This plan is aligned with the overall project plan and, in extension, the development of the different vf-OS platform components.

Phase	Start	End
Specification phase	**M7**	**M12**
Use Case Definition Templates and Guidelines	7	7
As-Is Evaluation and Reporting	7	7
Pilot Application requests	8	10
Request for additional info	9	10
Pilot application proposal	9	12
Development and Test Phase 1	**M13**	**M24**
Development of vApp Group 1:	16	24
vApp P12		
vApp P24		
vApp P31		
Development of vApp Group 2:	25	30
vApp P11		
vApp P14		
vApp P21		
vApp P23		
vApp P32		
vApp P33		
Development of vApp Group 3:	31	36
vApp P13		
vApp P15		
vApp P22		
vApp P23		
vApp P34		
vApp P35		
Evaluation Phase	**M25**	**M30**
Evaluation of Group 1:		
GQM Specification	20	20
Definition of evaluation parameters	21	22
Evaluation template definition	23	23
R1 Evaluation and Reporting	24	24
Evaluation of Group 2:		
GQM Specification	20	20
Definition of evaluation parameters	21	22
Evaluation template definition	23	23
R1 Evaluation and Reporting	24	24
Evaluation of Group 3:		
GQM Specification	20	20
Definition of evaluation parameters	21	22
Evaluation template definition	23	23
R1 Evaluation and Reporting	24	24

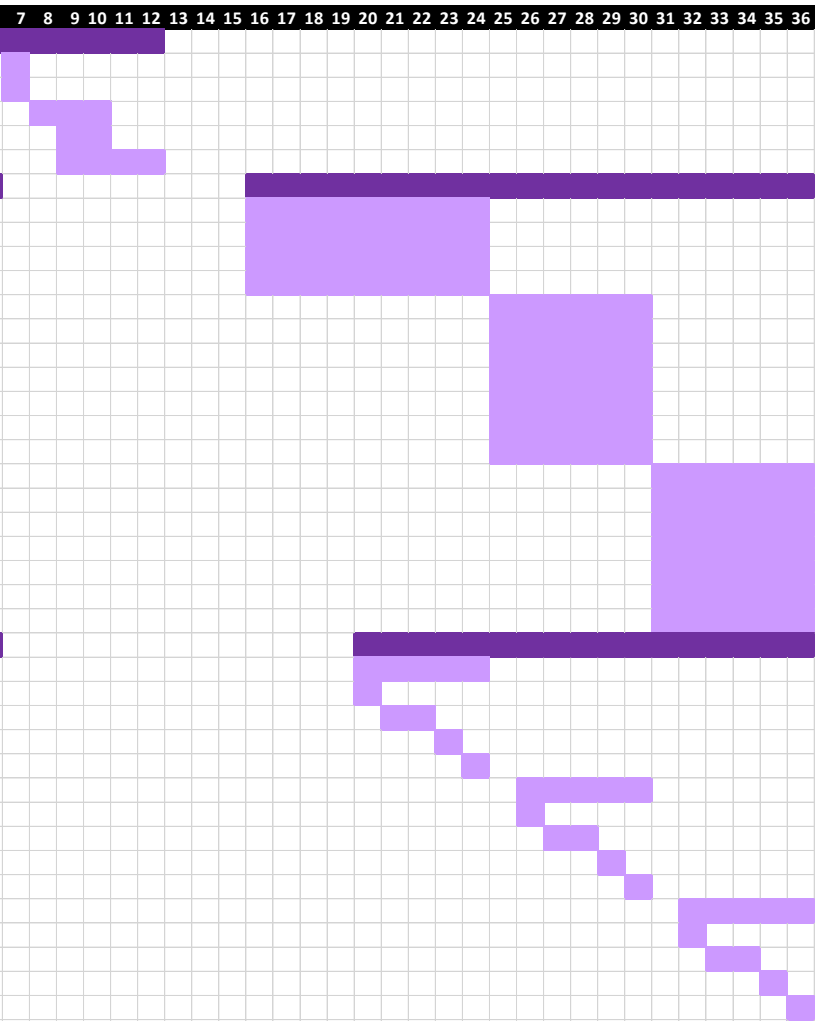


Figure 30: Pilot scheduling

Thus, M1 is the first month from the project start. This plan comprises all activities related to the pilots from the delivery of D8.1a to the end of the project. It also defines the contents and the expected reporting for the following versions of this deliverable. In accordance with the Methodology phases and steps defined in Section1.2, the plan is composed of the following phases:

Specification Phase	
Start/End	Start: M7 / End: M12
Objectives:	

O1: To define the AS-IS value for the pilot specific indicators **O2:** To provide the Use Case Definition Templates **O3:** To submit the application request for each pilot vApp **O4:** To complete Story Maps
Description: The Use Case Definition Templates and Guidelines document will be completed and application request samples from every pilot vApp will be annexed to the deliverable. In this version, the Use Case Definition Templates and Guidelines will consist of a document which could be updated to web formats in future releases. Using this template, Manufacturing and Logistic Users will fill in an application request for applications that they want to see developed within the vf-OS project (specified in D1.2). With this information, Software Developers and Manufacturing and Logistic Users will interact to complete the Story Maps and a detailed functional description of every vApp, which is documented in the corresponding story map. The pilots will provide an AS-IS reporting of the status of the indicators, evaluating the baseline for the pilot specific indicators in Section 3.4. **Roles: UPV** leads the definition of the Use Case Definition and Templates. **VS** fills in application request for pilot 1. **CONSULGAL** fills in application request for pilot 2. **APR** and **Tardy** fill in application request for pilot 3. **MASS**, **IKERLAN** and **UPV** complete story maps for pilot 1. **CONSULGAL**, **UNINOVA** and **KBZ** complete story maps for pilot 2. **APR**, **Tardy** and **LYON2** complete story maps for pilot 3.
Deliverables: **D8.1b Annex (Use Case Definition Template) Validation:** Use Case Definition Template and Guidelines M9 **D8.1b Annex (Use Case Definitions) Validation Scenarios:** Describes the application requests and story maps for the different pilots (Annexed Use Case Definition Template and Guidelines) M9 **D8.1b Validation Scenarios:** Describes how the application proposals were prepared and provides the baseline for pilot indicators. M12

Figure 31: Specification Phase description

Development and Test Phase	
Start/End	Start: M16 / End: M36
Objectives: **O1:** Develop and test vApps	
Description: In this phase, the vApps are developed and tested. Note that the development phase is parallel to the development of the vf-OS platform. The schedule is defined in a way so that pilots can influence the development of the components. Software Developers need to apply agile methods and work together with Manufacturing and Logistic Users in short development sprints to implement an incremental, dynamic development plan. Since both components and vApps adopt agile software development methodologies, the user stories of pilot vApps and vf-OS components can be checked against each other on each iteration. Also, the functional specifications can be updated accordingly, also taking into consideration the allocation of development efforts between components (core functionality) and pilot vApps (pilot specific functionality). The vApps are organised into three groups to better focus efforts into the development of pilot vApps and optimise the feedback from the pilots to vf-OS components. • The first vApp group is consisted of vApps P12, P24, and P31 because they solve specific pilot needs and because they have a more straightforward with specific vf-OS components: Drivers, Data Analytics, Enablers, and APIs. Since they require basic component features and interactions, they can use the first release of components and a basic OAK integrating the needed libraries to specifically implement this. Therefore, the development of the first releases of the components will take this input into account. • The second vApp group contains vApps P11, P14, P21, P23, P32, and P33 because they have more complex interactions that need more advanced versions of vf-OS components and the OAK and are well	

suited for validation of vApps in relevant scenarios. This group will therefore use more advanced versions of the OAK and the platform components.

- The third vApp group consists of vApps P13, P15, P22, P23, P34, and P35. The main objective is to validate the usability of the platform to develop applications.

Roles: MASS and **VS** in pilot 1; **CONSULGAL** in pilot 2; **APR** and **Tardy** in pilot 3 act as Manufacturing and Logistic Users and are responsible to test and evaluate the software for every iteration. They also need to set priorities and make decisions on the rescheduling and/or redefinition of user stories. **IKERLAN** provides implementation guidelines for pilot 1 vApps and develops one vApp in the third group release in pilot 1. **UPV** develops one vApp in the first group release and one vApp in the second group release for pilot 1.**CMS** develops one vApp in the second group release and one vApp in the third groups release for pilot 1. **KBZ** develops vApps for pilot 2. **LYON2** develops vApps for pilot 3. **ICE** provides implementation guidelines for data management and data analytics. **ASC** provides implementation guidelines for the front end. **ALM** provides implementation guidelines for service integration.

Deliverables:

D8.2a Pilot 1: Manufacturing & Logistic (DEM) Group 1 vApps

D8.3a Pilot 2: Construction – Industrialisation (DEM) Group 1 vApps

D8.4a Pilot 3: Manufacturing Assembly: Collaboration (DEM) Group 1 vApps

D8.2b Pilot 1: Manufacturing & Logistic (DEM) Group 2 vApps

D8.3b Pilot 2: Construction – Industrialisation (DEM) Group 2 vApps

D8.4b Pilot 3: Manufacturing Assembly: Collaboration (DEM) Group 2 vApps

D8.2c Pilot 1: Manufacturing & Logistic (DEM) Group 3 vApps

D8.3c Pilot 2: Construction – Industrialisation (DEM) Group 3 vApps

D8.4c Pilot 3: Manufacturing Assembly: Collaboration (DEM) Group 3 vApps

Figure 32: Development Phase description

Evaluation Phase	
Start/End	Starts: M24 / End: M30
Objectives: **O1:** Complete the Evaluation **O2:** Evaluate releases in the Evaluation Phase	
Description: During the development of vApps, evaluators will have all the necessary information to prepare the different evaluation forms for the different pilot use cases, according to the GQM methodology described in this deliverable. When the development of a vApp is completed, Manufacturing and Logistic Users and Software Developers will receive the GQM forms and fill in the necessary information. Evaluators will include the submitted forms, the results and the status of the indicators in the different deliverables. **UNINOVA** will act as evaluator of pilot vApps.	
Deliverables: **D8.1c:** Reporting of vApps Group 1 and Group 2 **D8.1d:** Final evaluation report	

Figure 33: Evaluation Phase description

5 Concluding Remarks

The methodology to support the definition and evaluation of vf-OS use cases has been defined; linking story maps –an agile software development tool to define user stories – with the GCM method – a methodology to evaluate the development and functionality of software. The same methodology can be used post project to support the interactions between the different customer groups of the multi-sided platform.

The forms and guidelines supporting the definition of vApps have been motivated and presented. As a first conclusion, the methodology and supporting documents provide a basis to further analyse interactions in the multisided platform in the pilots and identify new business opportunities post project around them.

Based on this methodology, a set of generic and pilot specific goals and indicators have been defined. Another important conclusion is that unit tests associated to story maps allows Manufacturing and Logistic Users to define KPIs to effectively evaluate the development and the functionality of vApps and take action if the results are not optimal.

As a final conclusion, the methodology can be effectively applied to evaluate and validate the implementation of the pilot vApps defined in D1.2.

Annex A: History

Document History	
Versions	V0.1: ToC Draft and editorial responsibilities V0.2: Introduction V0.3: Use Case Definition and Templates Draft V0.4: Use Case Definition and Templates Revision and Contributions V0.5: Monitoring, Reporting and Evaluation Methods Draft V0.6: Monitoring, Reporting and Evaluation Methods Revision and Contributions. Pilot Scheduling Draft V0.7: Pilot Scheduling Revision and Contributions V0.8: First internal revision by authors V0.9: First internal independent revision V1.0: Changes and corrections from first internal independent revision V1.0.1: Document ready for second independent revision V1.0.2: Document ready for third independent revision V1.0.3: EU Submission
Contributions	UPV: • Raúl Poler • Francisco Fraile KBZ: • Raquel Melo UNINOVA: • João Sarraipa • Sudeep Ghimire MASS: • Jon Atulbe Tardy: • Ibrahim Benali APR: • Benjamin Menghini CONSULGAL: • Fernando Monteiro IKERLAN: • Eduardo Saiz ALM: • Andries Stam ICE: • Stuart Campbell • Rebecca Campbell

Annex B: References

[BCR+94] Basili, V et al. "The Goal Question Metric Approach," in *Encyclopedia of Software Engineering*, J. Marciniak. Hoboken, NJ: John Wiley & Sons, 1994.

[ISO+11] *ISO Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Evaluation process*, ISO/IEC Standard number 25040, 2011.

[ISO+01] ISO Ergonomic requirements for office work with visual display terminals (VDT)s - Part 14 Menu dialogues, ISO/IEC Standard number 9241-14, 2001.

[MSB+17] Marcelino-Jesus E., Sarraipa J., Beça M. and Jardim-Goncalves R. "A framework for technological research results assessment," International Journal of Computer Integrated Manufacturing, Vol. 30, pp. 44-62, 2017.

[PAT+14] J. Patton and P. Economy, *User story mapping: discover the whole story, build the right product*, Sebastopol, CA: O'Reilly Media, 2014.

[STE+70] S.M. Stele. "Program Evaluation - A Broader Definition," Journal of Extension, VII, 1970, pp. 5-17.

[YAA+13] J. H. Yahaya, Z.N.Z. Abidin, N. M. Ali and A. Deraman, "Software ageing measurement and classification using Goal Question Metric (GQM) approach," 2013 Science and Information Conference, London, 2013, pp. 160-165.

Annex C: WP8 Deliverables

This deliverable is the basis for the series of deliverables regarding WP8, organising and reporting the validation tasks conducted in the vf-OS pilots. The relationship of the different deliverables is represented in Figure 34.

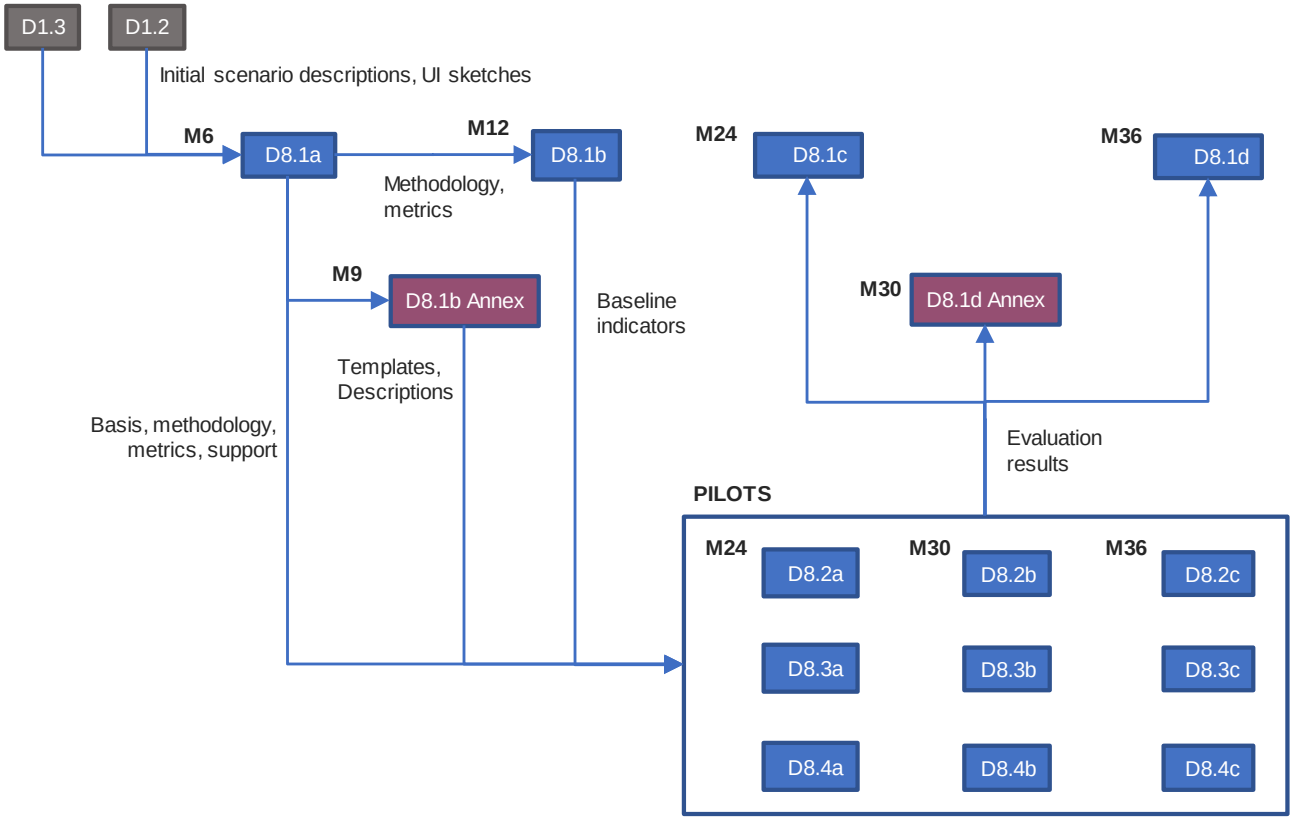


Figure 34: WP8 Deliverables

Considering the iterative and incremental development plan proposed for the pilot vApps, the reporting of the pilots is organised in the following deliverables:

- **D8.1 Validation Scenarios:** This series of deliverables detail the implementation plan and the results from the pilots.
  - **D8.1a:** This deliverable is based on the detailed vApp descriptions and UI sketches available in D1.2 that describe the initial pilot scenarios and builds on the experience gained to define a methodology to support the definition, evaluation and validation of vApps in the context of the pilots. D8.1a also provides a detailed implementation plan for the pilots.
  - **D8.1b:** This deliverable provides the baseline for the different indicators defined to evaluate the pilots
  - **D8.1b Annex:** The methodology allows pilot contributors to iteratively and interactively add detail to the vApps defined for the pilots. This methodology can be used post project to request new vApps. D8.1b provides the templates and guidelines used to request applications, together with the first samples collected from the project pilots and suggested improvements. The document also provides a detailed description of the features and functionality of the vApps in the form of story maps. This Annex is submitted prior to deliverable D8.1b in order to have this detailed description as soon as possible. After the

submission, the story maps will continue to be iteratively reviewed and developed following an agile software development methodology.

- **D8.1c:** This deliverable contains the conclusions from the evaluation of the first group of applications, aimed at solving specific pilot needs based on the technologies being delivered by the vf-OS Platform. The deliverable also includes an update on the templates and guidelines, a revision of the story maps and the implementation plan for the next development group of vApps.
- **D8.1d Annex:** This Annex is defined to report the main conclusions of the evaluation of the vApps grouped in the second release. As D8.1c, this Annex also provides a revision for the templates and guidelines, the implementation plan, and the story maps of the last group of vApps.
- **D8.1d:** This final deliverable includes the results of the evaluation of the vApps in the last group. This final evaluation report also includes the final conclusions of the evaluation and validation processes and the final version of the templates and guidelines that can be used post project.

Based on the methodology, WP8 will generate the demonstrators for the pilots that will be reported in D8.2, D8.3 and D8.4 in three different releases.

www.vf-OS.eu