EFPF: European Connected Factory Platform for Agile Manufacturing

European Factory Platform

WP2: Requirements Elicitation and Pilot Scenarios

D2.2: Initial Platform Interoperation Challenge Vs: 1.0

Deliverable Lead and Editor: Violeta Damjanovic-Behrendt, SRFG

Contributing Partners: SRFG, NXW, CNET, C2K, ICE, ASC, FOR, FIT

Date: 2019-05

Dissemination: Public

Status: <~~Draft~~ ¦ ~~Consortium Approved~~ ¦ EU Approved>

Short Abstract

This deliverable provides an overview of the platform development methodology devised in the EFPF project. Based on this methodology the development of the EFPF platform is kick-started through a small set of business challenges that involve implementing different usage scenarios across the 4 base platforms in the EFPF federation. The specifications of these scenarios, implementations and outcomes are described in this deliverable.

Grant Agreement: 825075

Document Status

Deliverable Lead	Violeta Damjanovic-Behrendt, SRFG
Internal Reviewer 1	Sarah Suleri, FIT
Internal Reviewer 2	Ingo Martens, HAW
Type	Deliverable
Work Package	WP2: Requirements Elicitation and Pilot Scenarios
ID	D2.2: Initial Platform Interoperation Challenge
Due Date	2019-05
Delivery Date	2019-05
Status	<~~Draft~~ ¦ ~~Consortium Approved~~ ¦ EU Approved>

History

See Annex A.

Status

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

Further Information

www.efpf.org

Disclaimer

Project Partners:

Executive Summary

The task “Initial Platform Interoperation Challenge” kick-starts the development of the EFPF platform and its federated digital manufacturing ecosystem. Platform interoperation challenges are a small set of focused experiments/challenges performed between two or more participating platforms in the EFPF federation.

The initial interoperation challenges are defined during consultations between the EFPF partners. The definition of the challenges is followed by undertaking the shortest possible implementation path that is capable to realise the defined usage scenarios and inform the development of EFPF federation.

The implementation of the initial interoperation challenges is performed by distributed development teams, primarily from the partners who developed the services of the base platforms, or those responsible for their functioning and availability over the course of the EFPF project. The implementation of the initial interoperation challenges represents a hands-on technology evaluation approach, with the purpose to inform the requirements capturing, technology development and integration tasks in the EFPF project.

The definition and implementation of the initial interoperation challenges also complements the architecture definition of the EFPF Data Spine by giving practical insights into the different data types, processes, security procedures, and communication protocols that need to be supported by the Data Spine.

In this way, this approach and the activities carried out as part of accomplishing this task, serve as a way of mitigating project risks through exploratory prototype implementations. This outcomes of the initial platform interoperation challenges and lessons learned will deliver several benefits including but not limited to providing the necessary insight into the existing or base platforms and easing the complexity of developing a platform federation in the EFPF project.

Table of Contents

0 Introduction

0.1 EFPF Project Overview

EFPF – European Connected Factory Platform for Agile Manufacturing – is a project funded by the H2020 Framework Programme of the European Commission under Grant Agreement 825075 and conducted from January 2019 until December 2022. It engages 30 partners (Users, Technology Providers, Consultants and Research Institutes) from 11 countries with a total budget of circa 16M€. Further information: efpf.org

In order to foster the growth of a pan-European platform ecosystem that enables the transition from “analogue-first” mass production, to “digital twins” and lot-size-one manufacturing, the EFPF project will design, build and operate a federated digital manufacturing platform. The Platform will be bootstrapped by interlinking the four base platforms from FoF-11-2016 cluster funded by the European Commission, earlier on. This will set the foundation for the development of the EFPF Data Spine and the associated toolsets to connect the existing platforms, toolsets and end users of the four base platforms. The federated EFPF platform will also be offered to new users through a unified portal with value-added features such as single sign-on (SSO), user access management functionalities to hide the complexity of dealing with different platform and solution providers.

0.2 Deliverable Purpose and Scope

The purpose of this document “D2.2 Initial Platform Interoperation Challenge” is to define and implement a set of initial interoperation challenges in EFPF, as an approach taken early in the project to enforce the establishment of a federated EFPF platform. The document corresponds to task T2.2 that starts in month 1 of EFPF project, with the goal to carry out prototypical experimentation on a set of platform interoperation challenges over a short (5 months) period of time. The document introduces Platform Interoperation Challenges as a new experimentation-based approach towards digital platform development, and particularly towards establishing a federation of distributed tools, systems and platforms. The document further informs the design and development of vital platform component such as the EFPF Data Spine and Marketplace Framework. It also serves as a case study on breaking down complex technological tasks into smaller challenges, where each of them addresses a specific aspect or user need.

The document is public in nature, therefore some of the implementation details are not made available to the reader. However, the provided information should allow the reader to understand the potential of interoperation, complexity of technical issues that are faced in the EFPF project and required technical capabilities to be addressed in the course of the EFPF project.

0.3 Target Audience

This deliverable provides an overview of the federation and platform development approach adopted in the EFPF project and is intended to be used by the technical teams working on platform development projects.

0.4 Deliverable Context

This document is one of the cornerstones for achieving the project results. Its relationship to other documents is as follows:

- **D2.3 – Requirements of Embedded Pilot Scenarios:** Captures pilot business needs as epics and user stories
- **D2.4 – EFPF Platform Requirements:** Captures the technological needs from technology providers in terms of user stories that are then translated into implementation tasks

0.5 Document Structure

This document includes the following sections:

- **Section 1: Motivation:** An introduction to this document addressing the motivation to adopt the Initial Platform Interoperation Challenge as an approach to develop the EFPF federation
- **Section 2: The Scope and Specification of Initial Platform Interoperation Challenges:** Defines the nature and scope of the initial interoperation challenges in the EFPF project
- **Section 3: Overview of Interoperation Challenges and their Relations with Pilots and Four Base Platforms:** Specifies the relationships between the initial interoperation challenges and the business needs coming out of the pilot cases in the EFPF project. The relation of each challenge with underlying base platform in EFPF is also elaborated
- **Section 4: Definition of Platform Interoperation Challenges:** Presents details of initial interoperation challenges with the four base platforms (and one external platform), defined in the project
- **Section 5: Preparation of Interoperation Challenges for the Experimentation:** Describes the way of selecting a set of challenges for implementation during the lifetime of the task T2.2 (M01 – M05)
- **Section 6: Implementation of Challenges – Lessons Learned:** Presents the outcome of implementing the interoperation challenges, performed by small technical teams in the project. The implementation details redact the sensitive information (e.g. API endpoints, security protocols, etc. by possibly replacing it with a string like *****) in the platform's security interest
- **Section 7: Concluding Remarks and Next Steps:** Provides a summary and an overview of next steps to be performed in the EFPF project

- **Annexes**:
  - **Annex A**: Document History
  - **Annex B**: EFPF Instance of the NIMBLE Platform
  - **Annex C:** COMPOSITION MQTT Broker
  - **Annex D:** Challenge 5 Implementation Details
  - **Annex E:** Challenge 9 Implementation Details

0.6 Document Status

This document is listed in the Description of Action as "Public" since it provides information to wider audience inside and outside the EFPF project.

0.7 Document Dependencies

None

0.8 Glossary and Abbreviations

A definition of common terms related to EFPF can be found at:

https://www.efpf.org/glossary

0.9 External Annexes and Supporting Documents

None

0.10 Reading Notes

None

1 Motivation

EFPF aims to become a digital collaborative manufacturing platform ecosystem in Europe. The project establishes a federation of four digital manufacturing platforms (from H2020 FoF-11-2016 cluster: NIMBLE[1], COMPOSITION[2], DIGICOR[3], and vf-OS[4]) that offer good complementarity of business functions, with a relatively common technology basis. In order to establish a federation of these four platforms (and beyond), the EFPF project must enable interoperation of functionalities offered by each platform joining the federation. This will enable the users in the federation, to utilise a variety of tools, services and other resources, outside the boundaries of their respective platforms.

The approach adopted in the project, to enable the initial interoperation between four available base platforms, is called “Initial Platform Interoperation Challenge”. This approach is the focus of task T2.2. The motivation to perform this task is to kick-start the development of the EFPF federation through implementing a set of small interoperation challenges that exploit the technical functionalities of the four base platforms.

The expected outcome of executing the initial interoperation challenges in T2.2 is to foster the understanding of the base platforms, to understand the process of their integration and interoperation in the EFPF federation, and to enforce the requirement elicitation and architecture design activities in the EFPF project.

At the time of submitting this document (M05 of the project), the EFPF Data Spine, a federation management mechanism (integration middleware) that enables interoperability at the level of processes, security protocols, APIs, data exchange, and other operations in EFPF, is not implemented yet. Thus, the initial platform interoperation challenges define and perform simplified interoperation solutions through *the direct platform interoperation* that enables any of the two platforms to perform defined functionality, in a controlled manner. This is different from *the platform interoperation via the EFPF Data Spine* that will be realised once the core EFPF federation services are established and available for use in the project.

Another motivation to adopt this experimental approach towards platform development is to gain technical know-how about interacting/interfacing practices for putting together currently available tools and services of the four base platforms. The valuable insights gained through the execution of challenges will be used in the design and development of the EFPF Data Spine that will facilitate the interoperation of diverse tools, systems and platforms.

Finally, in the later phase of the project, the tools and services of the four base platforms will be fully interlinked and controlled via the EFPF Data Spine.

[1] https://www.nimble-project.org/
[2] https://www.composition-project.eu/
[3] https://www.digicor-project.eu/
[4] https://www.vf-os.eu/

2 The Scope and Specification of Initial Platform Interoperation Challenges

The scope of the initial platform interoperation challenges is defined based on the underlying business needs of the pilot partners in the project, coming from the following scenarios:

- Connected Factories and Lot-Size-One Manufacturing: Ad-hoc Supplier Network in the Aviation Domain
- Lot-Size-1 Furniture Manufacturing in B2B Scenarios
- Agile Supply Networks in Circular Economy.

In the context of task T2.2, the definition of a set of initial platform interoperation challenges addresses specific needs of the pilot partners. At the high level, the mapping of the business needs of pilots, with identified technological challenges, is carried out to define the following three broad categories of interoperation challenges:

- **Business function level** interoperation challenges, targeting some of the core functionalities of platforms, e.g. login functionality, support for agile network creation, partner or product search across platforms, cross-platform marketplace utilisation, etc.
- **Platform level** interoperation challenges are about interoperability between various platform layers, i.e. devices, network, middleware, application, data and semantics. The interoperability challenges from this category require security, privacy and trust to be guaranteed. In addition, these interoperation challenges address interoperability of modules that cover device management, quality of services (QoS), external system services (e.g. cloud provider services), storage, virtualisation, etc. Examples of challenges in this category include cross-platform user identity and access management, cross-platform device access and cross-platform data-flow management.
- **Interoperation challenges with external platforms** support the idea of federation and extension of the EFPF ecosystem by enabling the 3rd party tools, systems and platform to interoperate through the EFPF Data Spine.

3 Overview of Interoperation Challenges and their Relations with Pilots and Four Base Platforms

At the beginning of the project, all three pilots in the EFPF project shared their business needs and expectations to be realised through the EFPF platform. For example, a specific business need identified through the pilots, is to enable single login functionality across multiple platforms coming into the EFPF federation. Hence, an interoperation challenge is defined to investigate single sign-on (SSO) functionality across multiple platforms. Similarly, based on the study of the user needs, a set of other challenges was drawn and mapped back to the user needs.
Figure 1 shows the interoperation challenges and their mapping with the business needs of all three pilots in the project:

- **Aerospace:** Ad-hoc supplier network in the aviation domain (Airbus, Hanse-Aerospace, 3D-ICOM, AM Allied Maintenance, Innovint Aircraft Interior, Walter Otto Müller)
- **Furniture:** Lot-Size-1 furniture manufacturing (LAGRAMA, AIDIMME)
- **Circular Economy:** Agile supply network in circular economy (KLEEMANN HELLAS, Eldia, MilOil)

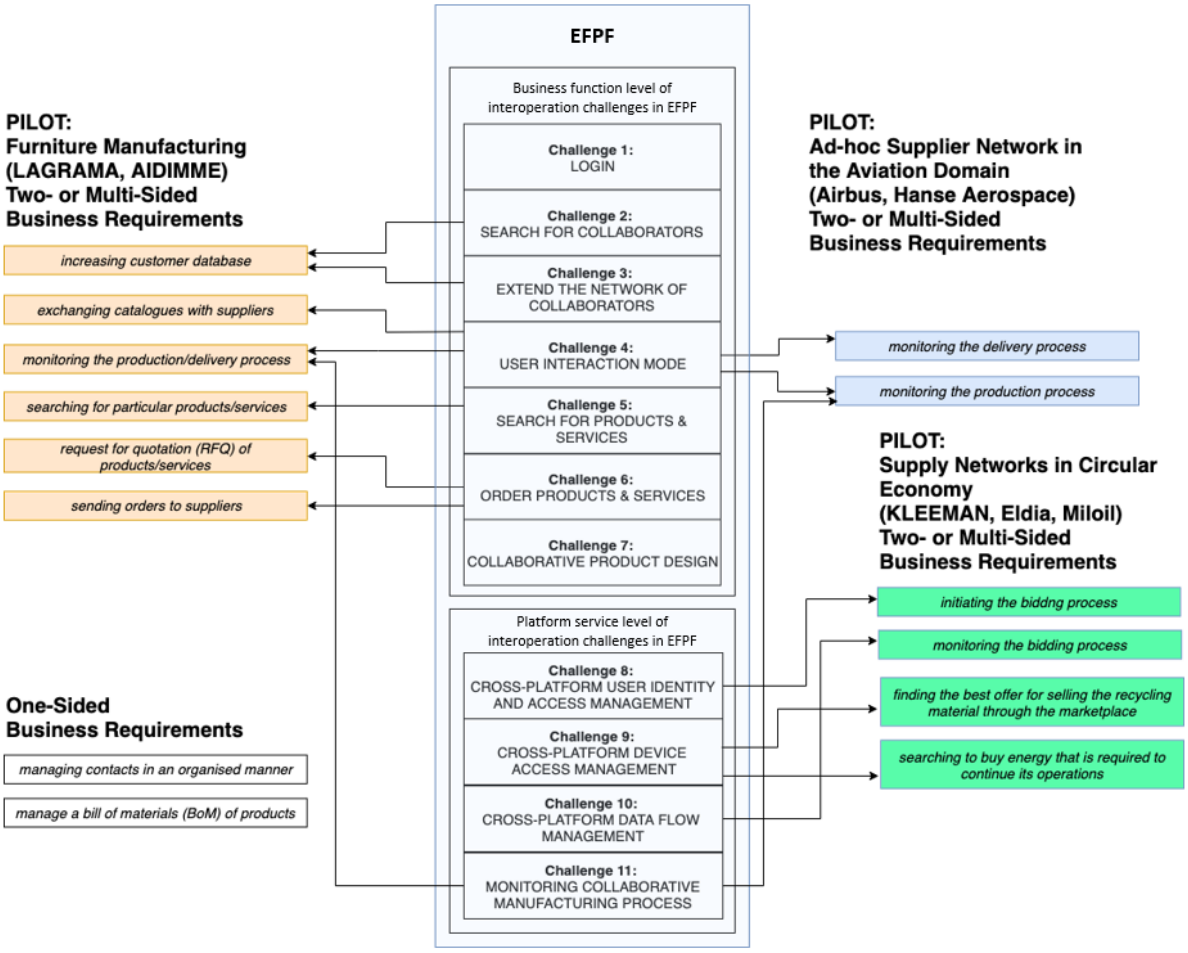


Figure 1: Relationships between the initial interoperation challenges in EFPF and the pilot requirements

The identified EFPF initial platform interoperation challenges shown in Figure 1 are listed below:

- **Challenge 1: Login**: Should be defined in a way to support user identification (login) at one platform, with the possibility to reuse the same user's credentials to access and use services of another platform in the federation. All three pilots require login (SSO) functionality.
- **Challenge 2: Search for collaborators**: Should support users to search for new collaborators within or across a particular domain in the platform federation. For example, the furniture manufacturing pilot partners want to "*increase customer database*" by searching for potential customers in the aviation pilot domain
- **Challenge 3: Extend the network of collaborators**: After finding new collaborators, the user wants to add them to the network of collaborators. This challenge can still relate to the furniture manufacturing pilot and its "*increase customer database*" requirement – as elaborated in D2.3[5]
- **Challenge 4: User interaction mode**: This challenge is defined in a generic fashion, covering various user interaction needs at the business platform level, e.g. monitoring of a production process at the shop floor. The complexity of performing any type of user interaction requires more efforts than it is available in task T2.2. Thus, Challenge 4 is elaborated at high level of abstraction. Some examples of EFPF pilot requirements that fit this challenge are: "*exchanging catalogues with suppliers*" as defined by the furniture manufacturing pilot, or "*monitoring the delivery process*" as defined by both the furniture manufacturing and ad-hoc supplier network in the aviation domain
- **Challenge 5: Search for products and services**: This challenge should enable search functionality in the platform federation. An example is "*searching for particular products/services*" by the furniture manufacturing and also circular economy pilot partners
- **Challenge 6: Order products and services**: After searching for product and services, this challenge is about placing the order, which can be further augmented to support negotiation and the contract creation between partner spread across multiple platforms. An example is "*request for quotation (RFQ) of products/services*" by the furniture manufacturing pilot
- **Challenge 7: Collaborative product design**: This challenge focuses on the support for collaborative designing of products
- **Challenge 8: Cross-platform user identity and access management**: This challenge is similar to Challenge 1, and includes user management through the platform federation, e.g. updates related to user management, changes of the access privileges, etc
- **Challenge 9: Cross-platform device access management**: This challenge focuses on the cross-platform access and data management aspects. An example is accessing an IoT gateway or factory connector from outside the platform
- **Challenge 10: Cross-platform data flow management**: This challenge focuses on the control of the data flow across multiple platforms. An example is "*monitoring the bidding process*" by the supplier network in circular economy pilot or porting sensor data from one platform to another platform for analytics

[5] EU-GID=D07 - EFPF-ID=D2.3 - Requirements of Embedded Pilot Scenarios (M5)

- **Challenge 11: Monitoring collaborative manufacturing process**: This challenge should enable monitoring of collaborative manufacturing processes between several platforms. The examples are similar to Challenge 4, but address more collaborative platform-related monitoring aspects

Note that the list of the identified EFPF initial platform interoperation challenges (Figure 1) includes 11 challenges with the four base platforms. The challenge to be performed with the external platform that is already available to the project (e.g. Symphony) is excluded from this task, and from the analyses illustrated in Figure 1 and Figure 2.
Note also that Figure 1 differentiate between business requirements of the pilots referring to the interaction between two or several platforms (two or multi-sided platform activities) and those requirements that can be performed on a single platform (one-sided platform activities). The interoperation of functionalities belonging to a single platform is out of the scope of T2.2.

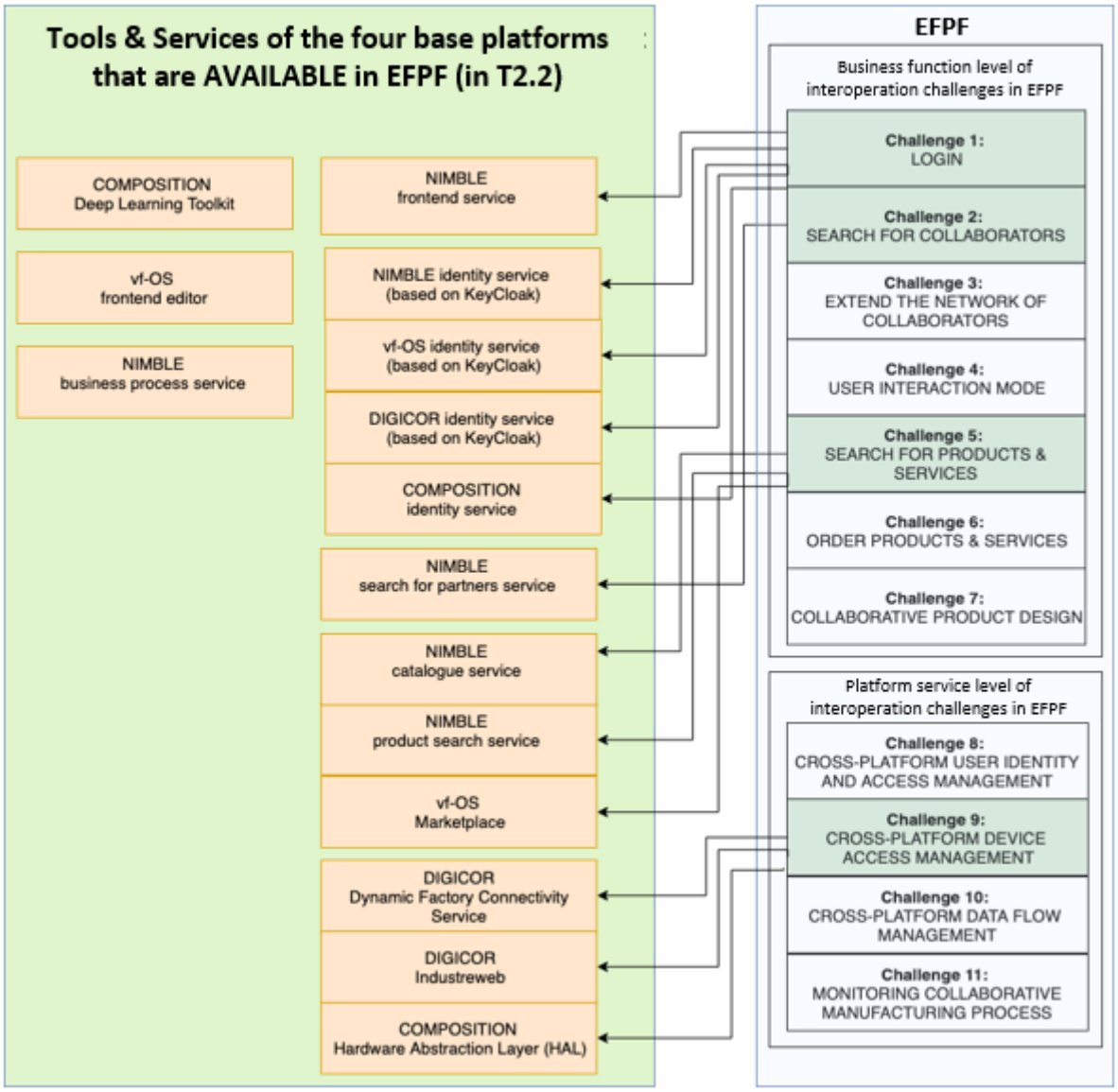


Figure 2: Links between the challenges and tools and services available in T2.2

Based on the mapping from Figure 1, the next step was to look for the tools and services of the base platforms that can be used to implement the above-identified 11 challenges. Performing this step required many consultations with technical teams of the base platforms, in order to understand services and APIs of the base platforms and to prepare platform instances for the experimentation. Figure 2 links initial interoperation challenges with the tools and services available for the experimentation in task T2.2 (before M05 of the project duration).
As Figure 2 shows, the majority of tools and services to support the implementation of the identified challenges are concentrated around: Challenge 1 (login), Challenge 5 (search for products & services), Challenge 9 (cross-platform device access management), while Challenge 2 (search for collaborators) can be supported by only one platform.

Therefore, those challenges that show the strongest technology support (e.g. Challenges 1, 5 and 9) are identified as the most feasible to implement in practice, in the course of task T2.2's lifetime (M1-M5 of EFPF project).

4 Definition of Platform Interoperation Challenges

The major objectives of the platform interoperation challenge in T2.2 can be summarised as follows:

- Kickstart the EFPF ecosystem formation
- Define a set of initial platform interoperation challenges that can be achieved in a short span of time to bring insight from complex platform development and integration tasks
- Involve different usage scenarios in a platform federation and prioritise their realisation through experimentation
- Make use of the tools and services of the four base platforms, which are available in task T2.2, in the period of M01-M05 of the EFPF project
- Define interoperation challenges with the external platforms brought in by EFPF partners, e.g. Symphony by NextWorks (NXW)

To define a set of initial platform interoperation challenges, a template was created with the following sections:

- **Challenge description**
- **Activity diagram** to visually describe the various steps in the challenge
- **Requirements to support the challenge** – technical teams of all four base platforms were asked to specify the conditions under which their tools and services can implement the challenge. In case of not having functional solutions to support the particular challenge, the technical teams were asked to clearly state it in order to balance their participation in another challenges' implementation
- **Collaborative design of the challenge** – to define potential contributions of technical teams towards the implementation of the challenge

The template is used for the uniform definition of challenges and to allow technical teams to individually contribute towards the definition of new challenges.

4.1 Interoperation Challenges: Business Function Level

4.1.1 Challenge 1: Login

This challenge focuses on user's identification and login on two or more platforms, using the same credentials (SSO). During this challenge, in case of the **direct platform interoperation** (without the EFPF Data Spine), after logging on the PLATFORM-1, the user should be able to access PLATFORM-2 using the same credentials (see Figure 3). Currently, the security controls of both PLATFORM-1 and PLATFORM-2 are designed to keep the integrity of their systems and business functionalities. In order to support Challenge 1, these controls need to be re-designed for each participating platform in the EFPF federation.

C1	Steps	Business Function	Suitable Platform / Tool
C1a	Login to a base platform	• Secure login at the platform level • User management	• Keycloaks (for identity and role-based access controls) of the four base platforms

C1b	Use the same credentials from a base platform, to access another platform	• Secure login to another platform • User management	• NIMBLE identity service • vf-OS identity service • COMPOSITION identity service • DIGICOR identity service

4.1.1.1 Activity Diagram

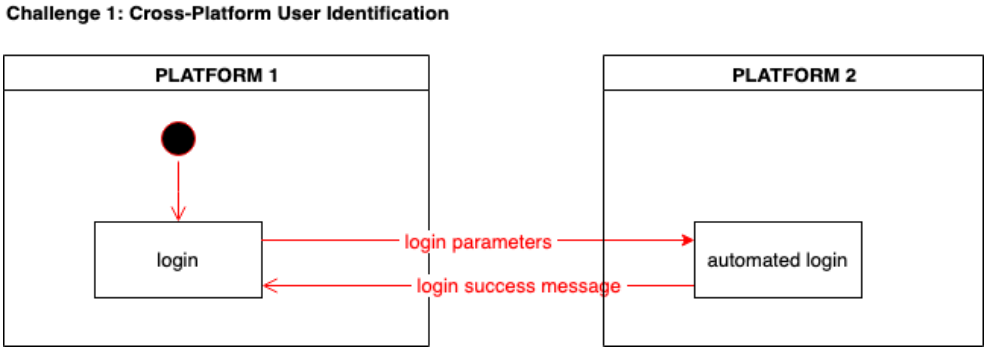


Figure 3: Cross platform user identification

4.1.1.2 Requirements to support the challenge

Table 1: Cross-Platform User Identification

Challenge Steps	Interoperation Requirements
Challenge-1.1	The registered user of PLATFORM-1 wants to access PLATFORM-2 using the same access credentials that can be exchanged between the platforms/ users
Challenge-1.2	The user of the PLATFORM-1 needs to have a mechanism to request the access to the PLATFORM-2, e.g. by selecting the name of the platform that s(he) would like to connect with. Such mechanism, through SSO (Single Sign On) or MFA (Multi factor Authentication) would enable access from PLATFORM-1 to PLATFORM-2.
Challenge-1.3	After the login is successfully performed, the user needs to be informed about the success of the login procedure, e.g. by receiving the login success message.

4.1.1.3 Collaborative design of the challenge

The following table describes the collaborative negotiation phase between the four base platform's development teams, during which all teams focused their attention on how to bring the best of their platform's tools and services to EFPF.

Table 2: Potential contributions of platforms towards implementation of Challenge 1

Platform	Potential Contributions of the Platform Towards the Challenge
NIMBLE	NIMBLE platform and APIs are tightly linked with security mechanism, authentication and role-based access controls and session management (via Keycloak). To support this challenge, NIMBLE needs to create a separated EFPF instance of the NIMBLE platform, a test user for EFPF and/or a test authorisation token to be used during the experimentation.

COMPOSITION	COMPOSITION platform uses OpenID Connect for authentication and an open, custom specification for authorisation. These security measures are enforced using software implemented as part of COMPOSITION and backed by Keycloak. To support this challenge, the COMPOSITION partners need to deploy the necessary components on a separate, sandboxed environment. The sandboxed setup would be sufficient to support this challenge. However, further implementation is needed to fully prepare COMPOSITION for use in EFPF in the absence of several development teams (outside of the consortium).
DIGICOR	Back-end services in the DIGICOR platform can be accessed through the AWS API Gateway, which checks for the presence of a valid access token. Tokens are obtained after successful authorization from AWS Cognito, which manages user identities. A test user for EFPF could be set up after permission from the DIGICOR consortium. In SMECluster, local authentication and authorisation is implemented, with cross platform support requiring development.
vf-OS	vf-OS identity service uses Keycloak as security and authentication management solution. To enable access to the vf-OS marketplace from a different platform, similar approach as the one defined in COMPOSITION will need to be adopted.

4.1.2 Challenge 2: Search for Collaborators

This challenge assumes that the user is successfully logged on both PLATFORM-1 and PLATFORM-2 (after performing Challenge 1). Both platforms belong to the EFPF federated ecosystem. The user of PLATFORM-1 wants to search PLATFORM-2 in order to find new partners/collaborators and extend the network of collaborators (see Figure 4). The search functionality can be designed without specifying search criterion or be based on the definition of specific search criterion. In the later, the user of PLATFORM-1 sends a list of query parameters to PLATFORM-2, e.g. “logistic company”, or ranking information, tags, proximity details, etc. to create a more specific search for collaborators.

C2	Steps	Business Function	Suitable Platform / Tool
C2a	User of PLATFORM-1 searches on PLATFORM-2 for partners with (or without) specific manufacturing capability(ies)	• Partner Search • Partner Matchmaking	• Search for collaborators/ partners • Partner matchmaking service

4.1.2.1 Activity Diagram

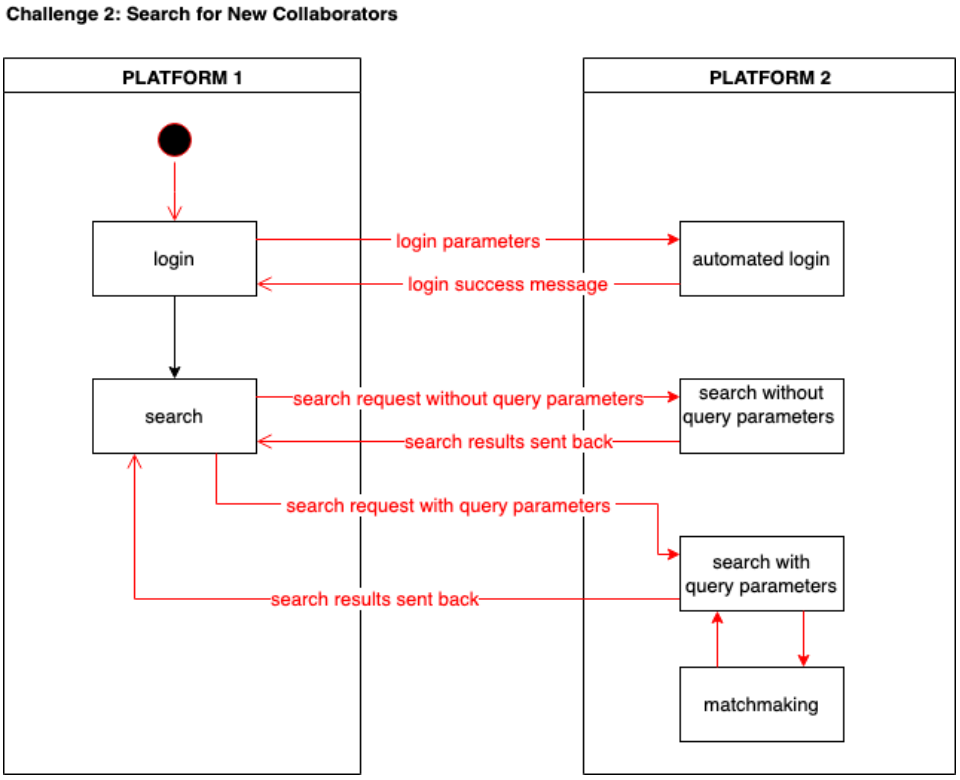


Figure 4: Search for collaborators

4.1.2.2 Requirements to support the challenge

Table 3: Interoperation Requirements, Challenge 2

Challenge Steps	Interoperation Requirements
Challenge-2.1	The user initiates search for collaborators, either by specifying the query parameters or without the query parameters.
Challenge-2.2	In case of searching with parameters, the user needs to have a way (UI) to specify query parameters.
Challenge-2.3	In case of searching with parameters, the search-for-partners service will additionally initiate matchmaking service, to offer the best search outcomes.
Challenge-2.4	After performing the search, the search results need to be presented to the user in a specific format (UI).

4.1.2.3 Collaborative design of the challenge

Table 4: Potential contributions of platforms towards implementation of Challenge 2

Platform	Potential Contributions of the Platform Towards the Challenge
NIMBLE	A new service for searching for companies is available in NIMBLE only from M4 of EFPF.
COMPOSITION	Participants (agents) in the COMPOSITION marketplace has access to the Collaborative Manufacturing Services Ontology using the COMPOSITION Ontology API. The interface has endpoints that can be queried for services and companies. There are currently limitations in the query parameters used in the

	API, but the underlying SPARQL queries has no such limitation. Results are returned as JSON in CXL (COMPOSITION eXchange Language) format. The details are available in the public deliverable D6.8 “Collaborative manufacturing services ontology and language II” of the COMPOSITION project.
DIGICOR	The DIGICOR platform provides the TDMS (Tender Decomposition and Matchmaking Service) component to search for companies/ partners based on business opportunity requirements. TDMS is currently being implemented in the SMECluster platform.
vf-OS	Partner/ collaborator search is not available in vf-OS.

4.1.3 Challenge 3: Extend the Network of Collaborators

In this challenge, the user of PLATFORM-1 initiates the search-for-partner service and identifies the partner organisation of PLATFORM-2 that the user would like to add to the portfolio of collaborators. The user of PLATFORM-1 sends the request to connect with the identified organisation on PLATFORM-2. The PLATFORM-1’ user receives the message that the request-to-connect is successfully performed.

In parallel, the PLATFORM-2 user receives message about the new request to connect. After accepting the request, both sides receive messages about their newly established connection. Figure 5 illustrates an example when the searching for the partners is done without configuring the search criteria. Figure 6 is the example when search for partners includes the query parameters.

C3	Steps	Business Function	Suitable Platform / Tool
C3a	Search for partners with specific manufacturing capability(ies)	• Partner Search • Partner Matchmaking	• Search for collaborators/ partners • Partner matchmaking service
C3b	Send request to the identified partner organisation, to connect with, for the future collaboration	• Request to connect	• Service for connecting with new collaborators

4.1.3.1 Activity Diagram

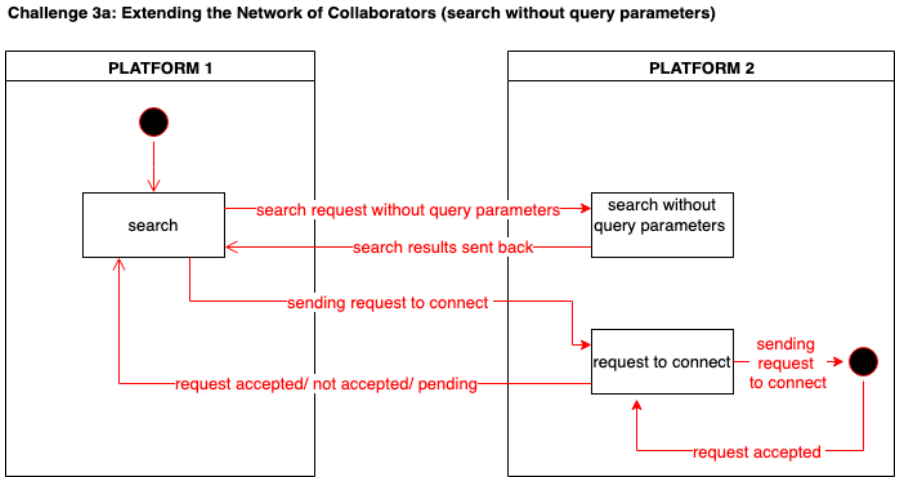


Figure 5: Extending the network of collaborators and sending the request to connect (search without query parameters)

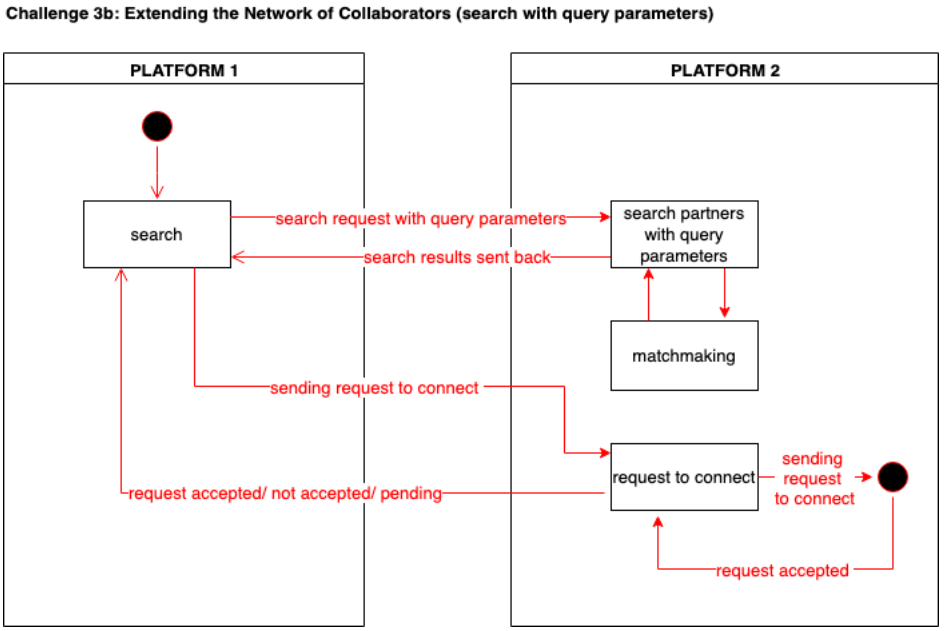


Figure 6: Extending the network of collaborators and sending the request to connect (search with query parameters and with partner' matchmaking)

4.1.3.2 Requirements to support the challenge

Table 5: Interoperation Requirements, Challenge 3

Challenge Steps	Interoperation Requirements
Challenge-3.1	Based on the received search results (see Challenge 2), the user identifies the partner organisations that s(he) would like to send the request to connect with. The user needs a way to write either specific message or to send the pre-generated message with the request to connect (UI). The user drafts the request-to-connect message.
Challenge-3.2	The user sends the request-to-connect message.
Challenge-3.3	The user receives the confirmation about the success of sending the request-to-connect message.
Challenge-3.4	The partner organisation receives the information about the new request-to-connect message.
Challenge-3.5	Depending on the partner's response, the status of the connection will be shown as "request accepted", or "request declined" or "pending".

4.1.3.3 Collaborative design of the challenge

Table 6: Potential contributions of platform's developers/owners to implement Challenge 3

Platform	Potential Contributions of the Platform Towards the Challenge
NIMBLE	Under development in NIMBLE
COMPOSITION	Not available in COMPOSITION
DIGICOR	Extending the DIGICOR TDMS and Collaboration Service to cross-platform scenarios would require further investigation and development work. In SMECluster the connection request can be supported but handling of the

	response would need development. Handling business logic via the Workflow service would be an element of this solution.
vf-OS	Not available in vf-OS

4.1.4 Challenge 4: User Interaction Mode

In this challenge, two users of PLATFORM-1 and PLATFORM-2 want to interact and establish a common collaboration space. For example, they want to exchange messages and documents through a private repository, or to negotiate specific conditions using available negotiation templates, or to move their discussion to the forum and involve other users to comment on specific matters, etc. In order to perform this challenge, the users expect several interaction modes to be available through UIs.

Figure 7 illustrates the situation when the users of two platforms, select a specific interaction mode and establish the interaction between them.

C4	Steps	Business Function	Suitable Platform / Tool
C4a	Engage in private/secure interactions with partners	• Interaction channel • Negotiation	Depending on the interaction mode
C4b	Establish an interaction channel/ a common working space for collaboration	• Negotiation Templates • Private communication channels • Document Repository • Discussion Forum	Depending on the interaction mode

4.1.4.1 Activity Diagram

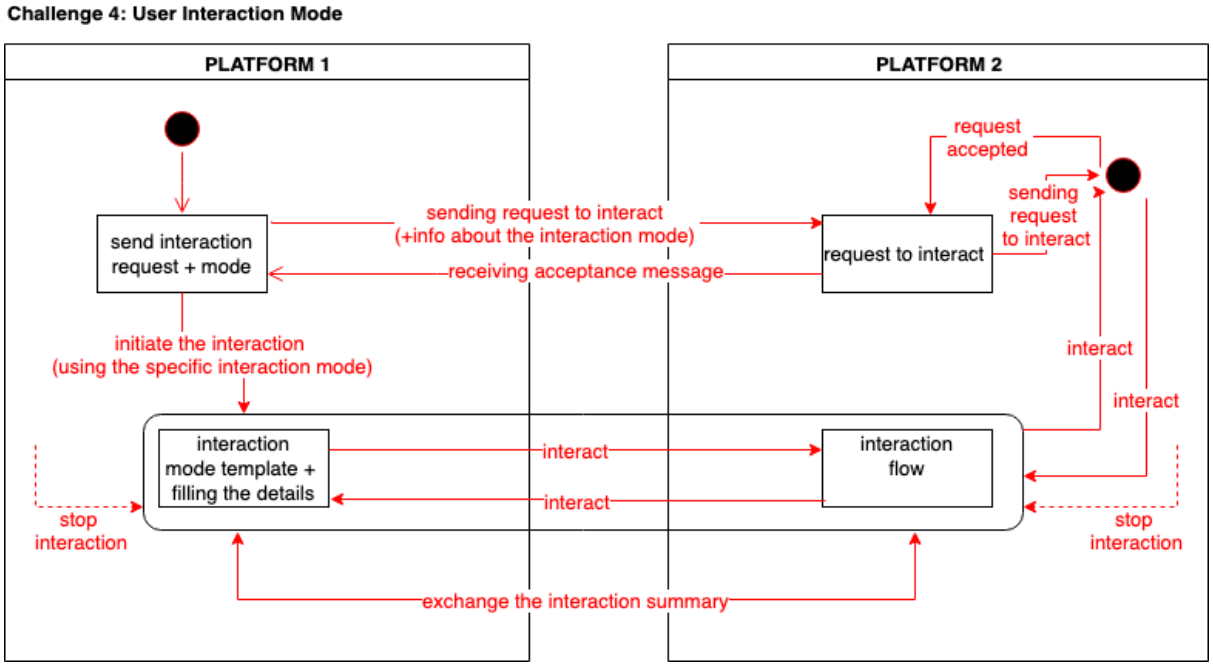


Figure 7: User interaction mode

4.1.4.2

Table 4.1. Interoperation Requirements, Challenge 4

Challenge Steps	Interoperation Requirements
Challenge-4.1	The user of the base platform sends the request for the interaction, to the user of another platform. The user needs a way to initiate the request-to-interact service, e.g. user of PLATFORM-1 can login to PLATFORM-2 and use common interaction medium (UI)
Challenge-4.2	The user sends the request-to-interact message.
Challenge-4.3	The user receives the confirmation about the success of sending the request-to-interact message.
Challenge-4.4	The partner organisation receives the information about the new request-to-interact message.
Challenge-4.5	Depending on the response, the status of the connection will be shown as “request accepted”, or “request declined” or “pending”.
Challenge-4.6	The user decides upon the type of the interaction mode to be used.
Challenge-4.7	The selected interaction mode is enabled by the system.
Challenge-4.8	The users enter into the interaction flow.
Challenge-4.9	The users stop the interaction flow, and exchange the outputs of their mutual interaction, e.g. summary report, contract, formulated decision, etc.

4.1.4.3 Collaborative design of the challenge

Table 4.2. The contributions of platform’s developers/owners to implement Challenge 4

Platform	Potential Contributions of the Platform Towards the Challenge
NIMBLE	NIMBLE fully supports the negotiation process prior to placing the final order.
COMPOSITION	COMPOSITION can support this challenge w.r.t. shop-floor data exchange. This can be done via the COMPOSITION Hardware Abstraction Layer (HAL) component, but additional development would be required to implement it as a standalone functionality. Moreover, no UI is available to support the user interaction in COMPOSITION.
DIGICOR	The DIGICOR Collaboration (Tender Workflow) Service supports the management and tracking of deliverables; the extent to which it could support this challenge would need to be examined more closely. SMECluster supports elements of this via Messaging Service and business logic handling via Workflow Service but an additional development would be required to implement unified functionality.
vf-OS	The user interaction is not possible in vf-OS.

4.1.5 Challenge 5: Search for Products and Services

In Challenge 5, the user of PLATFORM-1 is searching the products and services available from the marketplace of PLATFORM-2. The search functionality could be either based on the list of specific features of products and services, or generic, without

specifying the search details. For example, the user of PLATFORM-1 logins into the PLATFORM-2 and (1) starts searching the content of PLATFORM-2's marketplace without specifying the query parameters, or (2) defines searching parameters as a basis of searching for products and services.

Figure 8 illustrates the Challenge 5.

C5	Steps	Business Function	Suitable Platform / Tool
C5a	User of PLATFORM-1 can search for products & services on the marketplace of PLATFORM-2 with/ without specifying search parameters	• Products & Service Search • Product & Service Matchmaking • Partners Matchmaking	NIMBLE catalogue service can support exemplary call for querying products

4.1.5.1 Activity Diagram

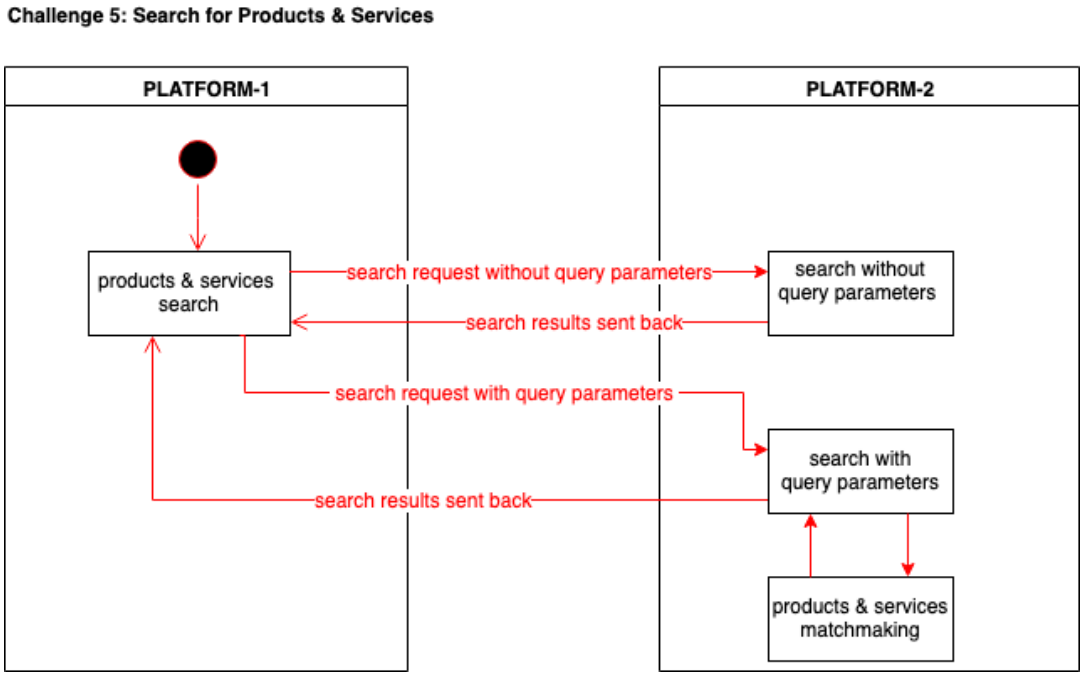


Figure 8: Search for products & services

4.1.5.2 Requirements to support the challenge

Table 5.1. Interoperation Requirements, Challenge 5

Challenge Steps	Interoperation Requirements
Challenge-5.1	The user needs to have a way to search the Marketplace for products & services, with or without specifying the query parameters.
Challenge-5.2	In case of searching without specifying the query parameters, the user sends a search request to the Marketplace.
Challenge-5.3	In case of searching with parameters, the user needs a way to specify the search parameters request (UI). The search-for-products-and-services service initiates (products & service) matchmaking, to find the best search outcomes.

Challenge-5.4	After performing the search, the results need to be presented to the user in a format that is previously agreed (UI).
Challenge-5.5	An addition to this challenge may be the requirement to perform matchmaking for both products & services and partners.

4.1.5.3 Collaborative design of the challenge

Table 5.2. The contributions of platform's developers/owners to implement Challenge 5

Platform	Potential Contributions of the Platform Towards the Challenge
NIMBLE	NIMBLE catalogue service is available
COMPOSITION	Currently, CXL (Composition eXchange Language) is needed in order to interact with the agent on the marketplace. To search for new solutions (products, services,) a message with the communication act (action) 'search' is needed. Agents in a COMPOSITION marketplace have access to the Collaborative Manufacturing Services Ontology using the COMPOSITION Ontology API. The interface has endpoints that can be queried for products and services. There are currently limitations in the query parameters used in the API, but the underlying SPARQL queries has no such limitation. Results are returned as JSON in CXL (COMPOSITION eXchange Language) format.
DIGICOR	With respect to the DIGICOR platform, the situation is similar to Challenge 2. SMECluster Marketplace implements Lucene as its search and index engine allowing Products and Services to be searched based on properties of the listing within the Marketplace. The Search Service API could support this requirement for search from an external platform. Population and development of marketplace is ongoing.
vf-OS	vf-OS provides existing API to request information about all assets listed in the Marketplace.

4.1.6 Challenge 6: Order Products and Services

After searching through the marketplace of PLATFORM-2 for products and services, the user of PLATFORM-1 would like to order some products & services. The PLATFORM-1 user creates the order and receive the contract from the PLATFORM-2 marketplace.

Figure 9 illustrates this challenge.

C6	Steps	Business Function	Suitable Platform / Tool
C6a	Search for products and services with/ without specific features	• Products & Service Search • Product & Service Matchmaking • Partners Matchmaking	NIMBLE catalogue service
C6b	Add products to the "basket"	• Put-in-the-basket service	NIMBLE Business Process service
C6c	Place the order (contract-based order)	• Ordering Service (Submit the Order service)	NIMBLE Business Process service

4.1.6.1 Activity Diagram

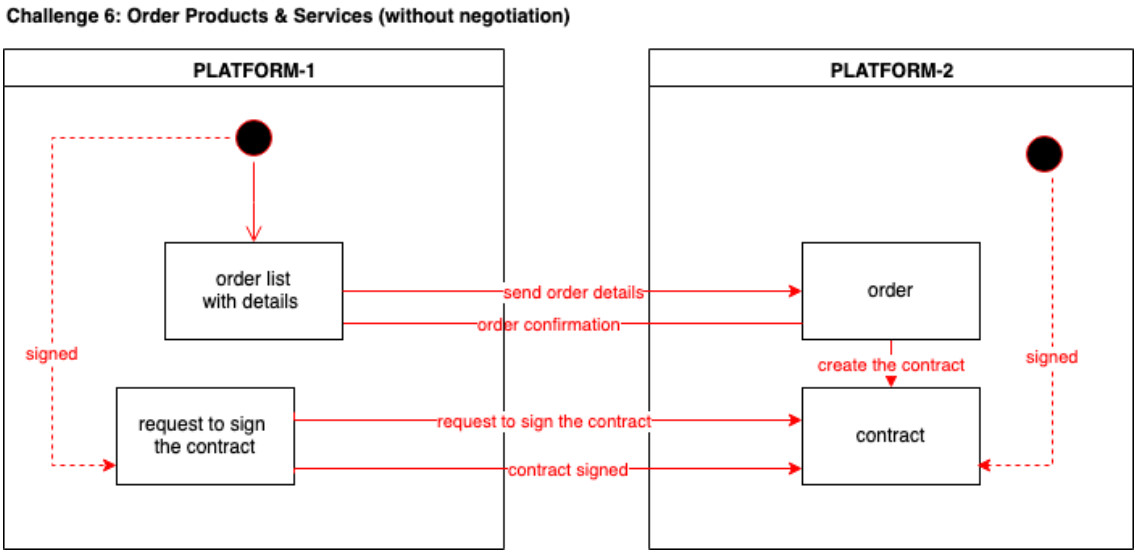


Figure 9: Order products & services

4.1.6.2 Requirements to support the challenge

Table 6.1. Interoperation Requirements, Challenge 6

Challenge Steps	Interoperation Requirements
Challenge-6.1	The user of PLATFORM-1 needs to have a way to search through products & services in the marketplace of PLATFORM-2, with or without specifying the query parameters.
Challenge-6.2	After performing the search functionality, the results need to be presented to the user in a specific format.
Challenge-6.3	After finalising, the user creates the order list with the order details.
Challenge-6.4	The user places the order, and the manufacturer receives the message about the new order.
Challenge-6.5	The marketplace accepts the order, which automatically trigger the contract creation.
Challenge-6.6	The contract needs to be signed by both sides.

4.1.6.3 Collaborative design of the challenge

Table 6.2. The contributions of platform's developers/owners to implement Challenge 6

Platform	Potential Contributions of the Platform Towards the Challenge
NIMBLE	NIMBLE Business Process service supports this functionality
COMPOSITION	Currently, CXL (Composition eXchange Language) is needed in order to interact with the agent on the marketplace. To start a new bidding process, a message with the communicative act (action) 'cfp' is needed. Also, agents on the marketplace currently operate with either 'Contract-net' or 'English auction' protocols.

DIGICOR	The tools currently available in the DIGICOR platform do not support the process of ordering products and services as such. SMECluster implements an enterprise e-commerce framework called Flexeweb Commerce. This provides API for Orders (Basket), Customers and Catalogue, with basket and checkout logic managed by the Workflow service.
vf-OS	Ordering of products and services is currently not supported in vf-OS at the time of submitting this deliverable

4.1.7 Challenge 7: Collaborative Product Design

In this challenge, the user of PLATFORM-1 searches for partners on PLATFORM-2 that can support collaborative product design. After identifying suitable partners, the collaborative process design environment of PLATFORM-2 is used to design a collaborative process that is accessible/visible to all partners from different platforms. Figure 10 illustrates this case.

C7	Steps	Business Function	Suitable Platform / Tool
C7a	Search for partners with specific manufacturing capability(ies)	• Partner Search • Partner Matchmaking	NIMBLE search for partners service
C7b	Design a collaborative/ distributed process where partners from different platforms can contribute towards different activities	• Private Channel for Collaborative production • Process Design Validation	DIGICOR tools for the collaborative design of products

4.1.7.1 Activity Diagram

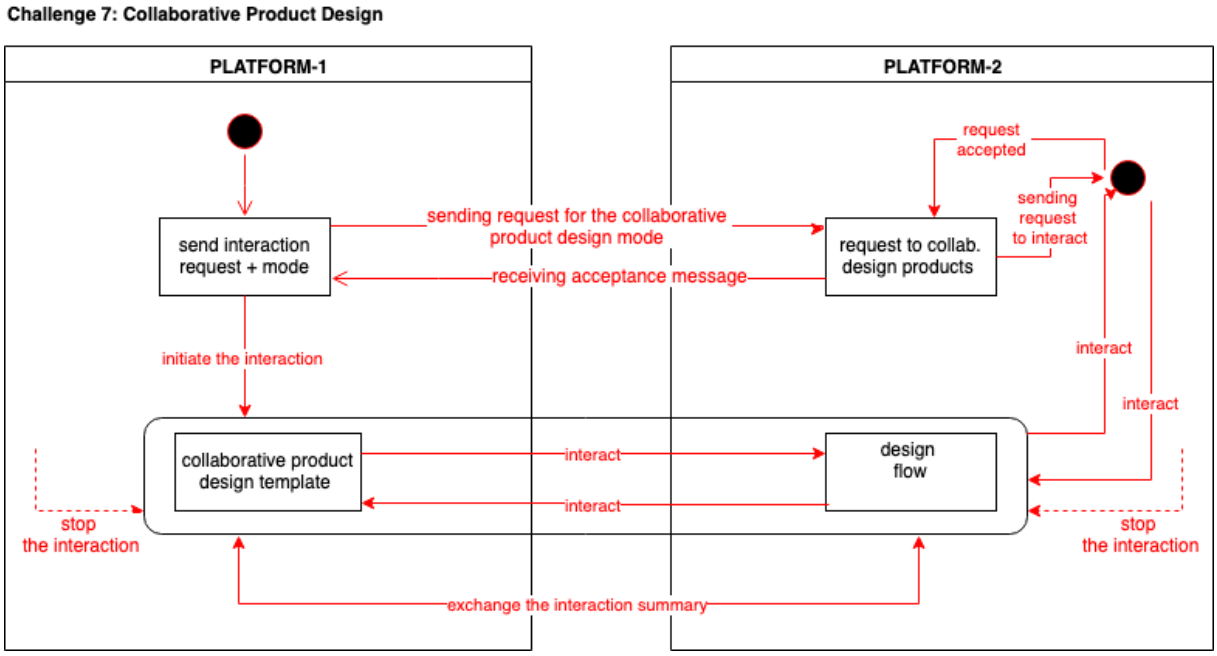


Figure 10: Collaborative product design service

4.1.7.2 Requirements to support the challenge

Table 7.1. Interoperation Requirements, Challenge 7

Challenge Steps	Interoperation Requirements
Challenge-7.1	The user is searching for partners with specific manufacturing capabilities, in order to realise collaborative production scenario
Challenge-7.2	After identifying partners on PLATFORM-2, the partners initiate a specific process designer that allow them to develop distributed processes
Challenge-7.3	They enter the collaborative design mode on the process designer
Challenge-7.4	The design process can be securely saved in the Cloud
Challenge-7.5	They can invite more collaborators to join the designed process
Challenge-7.6	After finalising the collaborative design, the summary reports and specifications are created

4.1.7.3 Collaborative design of the challenge

Table 7.2. Contributions of platform's developers/owners to implement Challenge 7

Platform	Potential Contributions of the Platform Towards the Challenge
NIMBLE	Not available in NIMBLE
COMPOSITION	Not available in COMPOSITION
DIGICOR	The DIGICOR Collaboration (Tender Workflow) Service supports the management and tracking of deliverables; the extent to which it could support this challenge would need to be examined more closely. Moreover, the Workflow and Service Automation tool can support collaborative design of processes (though not real-time for multiple users). Processes can be saved and made accessible to specific partners and roles, though this functionality is currently being developed. Once a process has been agreed, it is saved and executed. Activities (so far) are REST calls to services exiting in any other platform.
vf-OS	No partner/collaborator search available in vf-OS

4.2 Interoperation Challenges: Platform Service Level

Platform service level interoperation challenges address interoperability of modules that cover device management, quality of services (QoS), quality of data, external system services (e.g. Cloud provider services), storage, virtualisation, etc. These challenges are also about security, privacy and trust mechanisms at the level of the platform(s).

4.2.1 Challenge 8: Cross-Platform User Identity and Access Management

To access multiple platforms in the EFPF ecosystem, the user wants to use a single set of credentials. From the perspective of the platform, to effectively secure the data of users and their companies in the Cloud, and to ease the information exchange between the platforms, several controls would need to be enabled. Examples of such

controls include user identity and access control management, event management, policy management, authentication (to enable the use of various applications), command execution to enable password management, etc. Authentication could be based on MFA (Multi Factor Authentication) or SSO (Single Sign On), or their combination.

Figure 11 illustrates steps (C8a – C8c) are with a focus on user identity. The remaining steps (C8d – C8e) require the specific security control to be designed and implemented at the level of the EFPF platform, as shown in Figure 11. For example, if the user of PLATFORM-1 wants to perform C8e, which is about password management across platforms, such functionality shouldn't be allowed from any other platform except from the EFPF platform.

The challenge 8 follows the flow shown in Figure 11.

C8	Steps	Business Function	Suitable Platform / Tool
C8a	Platform enables user identity management	• Identity Management Service	User login
C8b	It enables user access control management between platforms	• Access Control Service	User authorisation and access controls (role-based access controls)
C8c	It enables user authentication	• Authentication Service	User authentication
C8d	It supports event log management	• Event Log Service	History of access logs
C8e	It enables command execution for security (and system) cloud based remote updates (e.g. password mngm.)	• Command Execution Service	• Changing the user's roles in the system • History of activities • Log files for security analysis

4.2.1.1 Activity Diagram

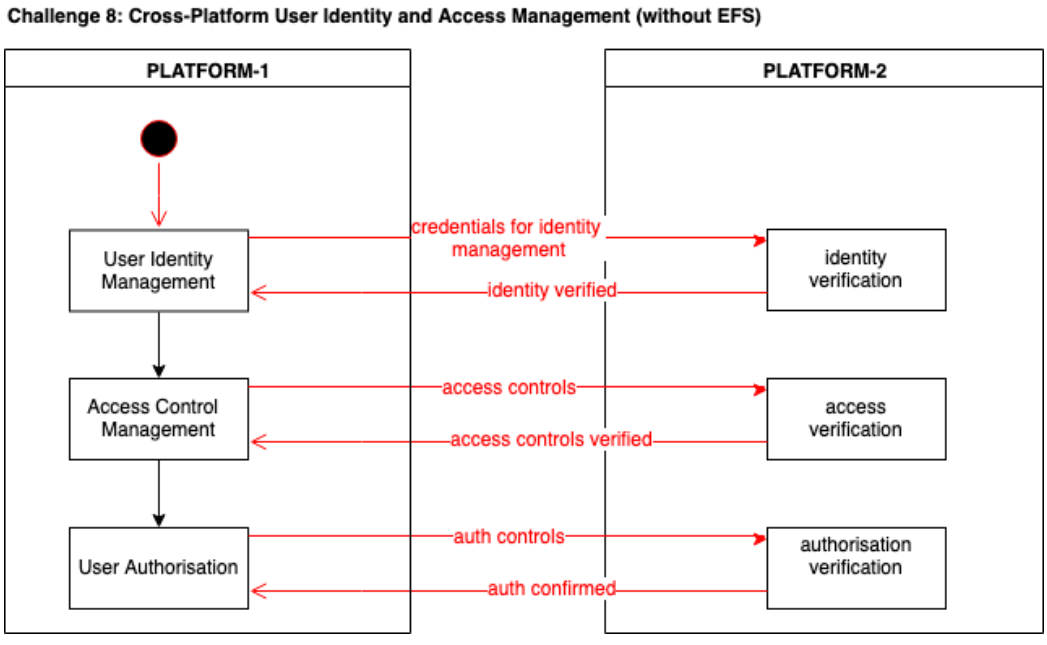


Figure 11: Cross-platform user identity and access management

An example of the multi-tenant identity and access management, via the EFPF platform (and the EFPF Security (EFS) component, to be part of the EFPF Data Spine) is illustrated in Figure 12. To perform the multi-tenant identity, the EFPF platform requires the federated identity management (through the EFPF Keycloak) to be designed and implemented, in order to efficiently govern the security management for different platforms. The EFS enables a class of SuperAdministrator which role is to provide secure authentication of any tenant platform in the ecosystem (e.g. multi identities to be managed across company's accounts).

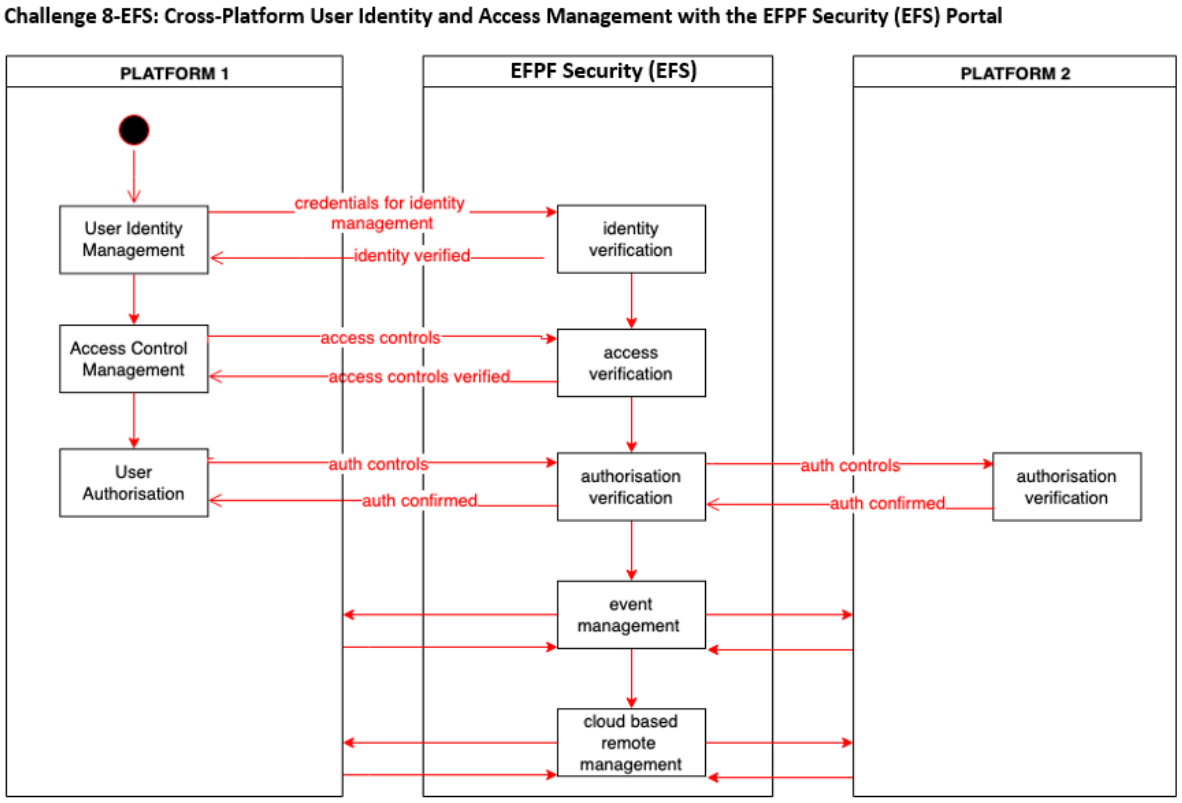


Figure 12: Cross-platform user identity and access management using the EFPF Security portal

4.2.1.2 Requirements to support the challenge

Table 8.1. Interoperation Requirements, Challenge 8

Challenge Steps	Interoperation Requirements
Challenge-8.1	The user of the PLATFORM-1 needs to be identified using credential (e.g. user name, password) and his/her identity need to be verified by the PLATFORM-2.
Challenge-8.2	The access controls rights of the user of the PLATFORM-1 need to be verified by the PLATFORM-2.
Challenge-8.3	The user needs to be authorised by the other platform in order to be enabled to access it and perform any action.

4.2.1.3 Collaborative design of the challenge

Table 8.2. Contributions of platform's developers/owners to implement Challenge 8

Platform	Potential Contributions of the Platform Towards the Challenge
NIMBLE	NIMBLE team will set up the EFPF Keycloak (EFS in this document)
COMPOSITION	COMPOSITION platform is uses Keycload based security and access management solution that can be tuned to operate with EFS
DIGICOR	The DIGICOR platform uses Keycloak as security and access management solution. SMECluster is currently implementing Keycloak based access mechanism that can be interconnected through the EFS
vf-OS	The vf-OS instance setup for the EFPF project has been protected by Keycloak based security, authorisation and user access management system. Interlinking with EFS will be investigated in due course

4.2.2 Challenge 9: Cross-Platform Device Access

This challenge can be initially performed directly between two or more platforms, as illustrated in Figure 13.

C9	Steps	Business Function	Suitable Platform / Tool
C9a	Access to shop floor devices across the platforms	• Shop floor device connector	

4.2.2.1 Activity Diagram

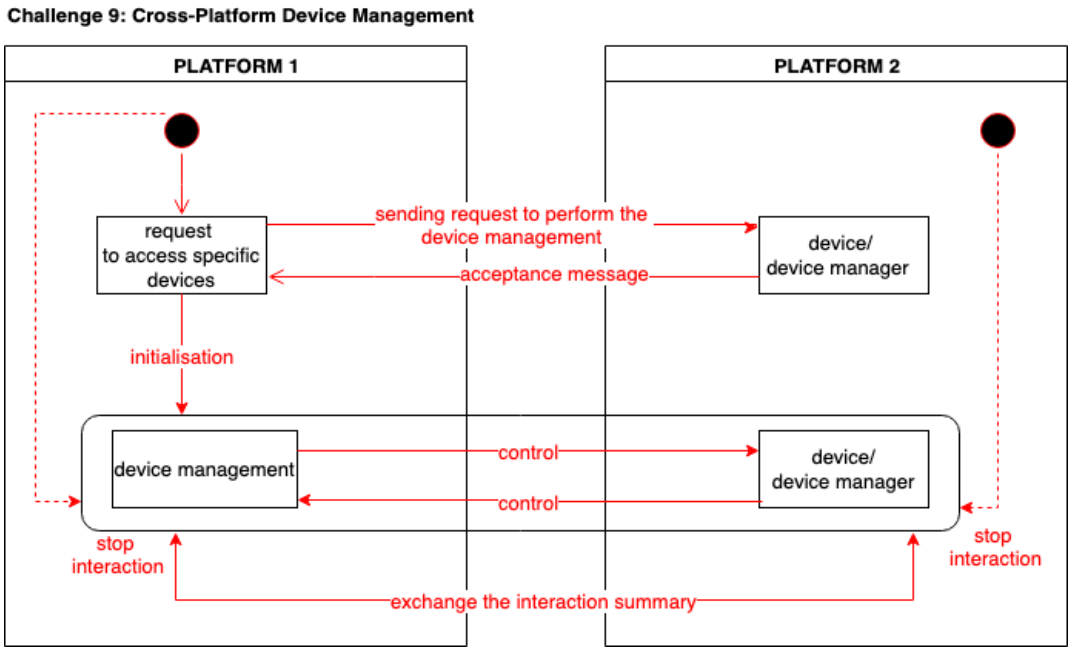


Figure 13: Cross-platform device management access

In the later phase of the project realisation, the device management functionality will be performed via the EFPF platform, involving the EFPF Device Management Access Service and the EFPF Event Management Service, as illustrated in Figure 14.

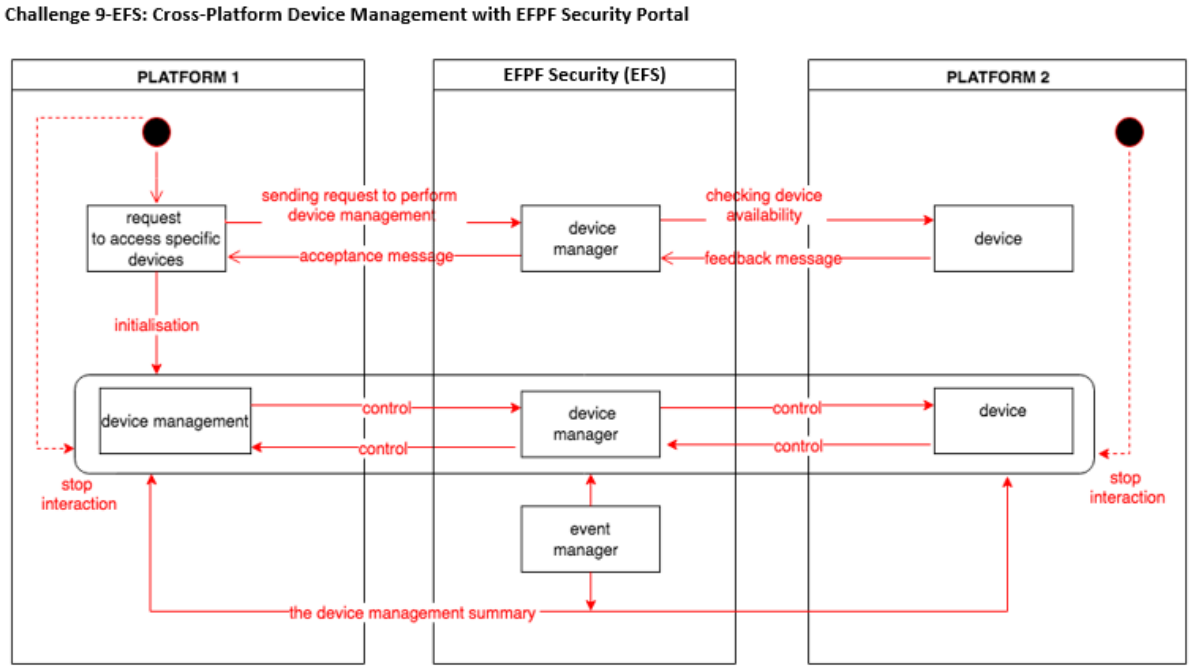


Figure 14: Cross-platform device management access using the EFPF Security portal

4.2.2.2 Requirements to support the challenge

Table 9.1. Interoperation Requirements, Challenge 9

Challenge Steps	Interoperation Requirements
Challenge-9.1	The user of the base platform sends the request to access (and manage, e.g. monitor) metadata (e.g. instrument ranges) and live data (e.g. temperature measurements) of specific devices from the other platform: a group of devices, or a type of devices, e.g. manufacturing machines, sensors, or infrastructure devices, or hardware pieces networked together as part of a communications infrastructure.
Challenge-9.2	After acceptance of the request, the user receives access to data from the selected devices for a specific time and for a specific, prearranged purpose of monitoring, e.g. to monitor the progress of manufacturing processes.
Challenge-9.3	Access to device data is limited to a specific, prearranged purpose.

4.2.2.3 Collaborative design of the challenge

Table 9.2. Contributions of platform's developers/owners to implement Challenge 9

Platform	Potential Contributions of the Platform Towards the Challenge
NIMBLE	In the development.
COMPOSITION	In the development.
DIGICOR	Factory connector framework enables two-way communication with shop-floor devices including broad library of common systems and devices. Protocols supported to the device including MQTT, AMQP, REST, selected profiles of OPC UA and OPC UA PubSub. Two implementations exist: Industreweb and the DIGICOR Dynamic Factory Connectivity service, with the latter especially

	addressing the subprocess of requesting access to an external data source and mapping the data provided by this source to match what is required by the tool on the platform.
vf-OS	Access to shop-floor is provided by devices drivers (connection to IoT devices) and API connectors (connection to shop-floor software). Both types are provided as assets in the marketplace, which means, they have to be ordered. There is no static public access available to other shop-floors in vf-OS.

4.2.3 Challenge 10: Cross-Platform Data Flow Management

In this challenge, shop floor data collected from machines that are connected to PLATFORM-1, needs to be exposed to data analytics services of PLATFORM-2. Firstly, the appropriate data analytics services need to be identified within the platform ecosystem. Secondly, secure data channels need to be established between the platforms, before starting new data analytics task. In order to identify appropriate data analytics services in the ecosystem, similar approach to Challenge 5 (C5a - search for products & services with specifying search parameters) is adopted. Figure 15 illustrates this challenge.

C10	Steps	Business Function	Suitable Platform / Tool
C10a	Identify a cluster of shop floor data to be analysed	• Shop floor device connector	DIGICOR shop floor device connectors COMPOSITION service for shop floor connection
C10b	Expose shop floor data to the data analytic tool of another platform	• Data Analytic	COMPOSITION Data analytics

4.2.3.1 Activity Diagram

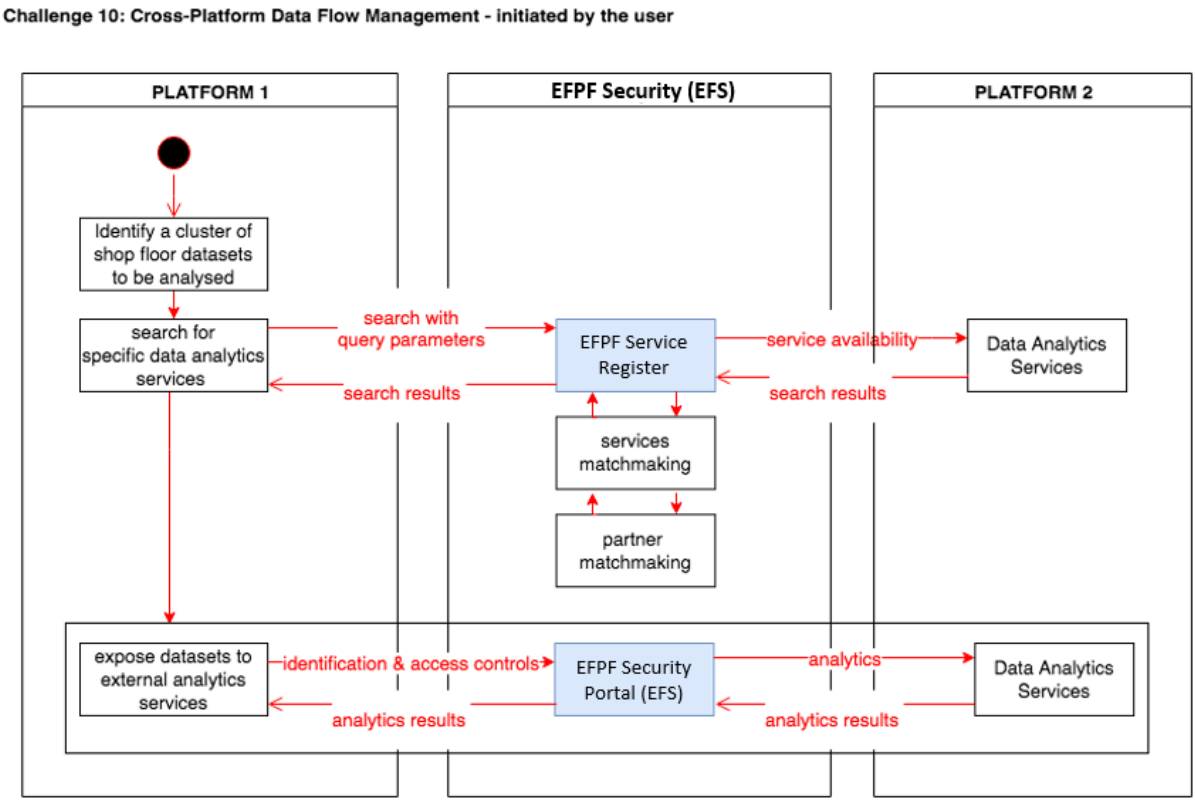


Figure 15: Cross-platform data flow management

4.2.3.2 Requirements to support the challenge

Table 10.1. Interoperation Requirements, Challenge 10

Challenge Steps	Interoperation Requirements
Challenge-10.1	Shop-floor data is being collected by a tool of PLATFORM-1.
Challenge-10.2	A pre-selected data analytic service of PLATFORM-2 can analyse shop-floor data and return analysis report
Challenge-10.3	The user directs the shop-floor data from PLATFORM-1 to the data analytic service in PLATFORM-2.
Challenge-10.4	The analytic service in PLATFORM-2 returns the analytic report to the user in some shape-or-form.

4.2.3.3 Call for collaborative development

Table 10.2. The contributions of platform's developers/owners to implement Challenge 10

Platform	Potential Contributions of the Platform Towards the Challenge
NIMBLE	In the development.
COMPOSITION	The Deep Learning Toolkit (DLT) can be used for predictive maintenance scenarios. Prerequisites: 1) Use case must be feasible with a deep learning approach 2) Large dataset of historical data 3) Preliminary data analysis 4) Training phase offline 5) Clear output expected Note: At this point of the project the predictions of the DLT are exposed through the FIT component called Learning Agent (LA).
DIGICOR	Workflow and Service Automation tool has a portlet being develop for data analytics related processes regardless of the type of activities done as part of processes. That is, if a process controls machines then the analytics will be about that; if the process runs at the business level then the analytics will be about that. If this challenge – which focuses on exposing data-at-rest – was extended to cover data-in-motion (stream analytics), the Factory Connector framework could support two-way communication between local factory systems (e.g. through OPC UA) and the collaborative platform (e.g. through OPC UA PubSub over AMQP).
vf-OS	Not supported in vf-OS

4.2.4 Challenge 11: Monitoring Collaborative Manufacturing Process

In this challenge, a user of PLATFORM-1 and a user of PLATFORM-2 (who previously agreed on collaborating together on manufacturing a specific product) work in a collaborative manufacturing environment synchronised between both platforms to (1) design and model the collaborative manufacturing process they will follow, in order to, (2) monitor the progress this distributed manufacturing process. The users of

PLATFORM -1 and PLATFORM -2 have the option to share status update data about the manufacturing process directly from their factories (see Figure 16).

C11	Steps	Business Function	Suitable Platform / Tool
C11a	Design/model a collaborative/ distributed manufacturing process where partners from different platforms contribute towards different activities	• Private Channel for Collaborative manufacturing • Process design (modelling) and (peer) validation	• DIGICOR services for collaborative design and production • DIGICOR services for the validation of the design
C11b	Monitor a collaborative/ distributed manufacturing process where partners from different platforms send status updates regarding their manufacturing activities	• Channels for sending manufacturing status updates • Monitoring collaborative manufacturing	• DIGICOR services for collaborative design and production • DIGICOR monitoring services

4.2.4.1 Activity Diagram

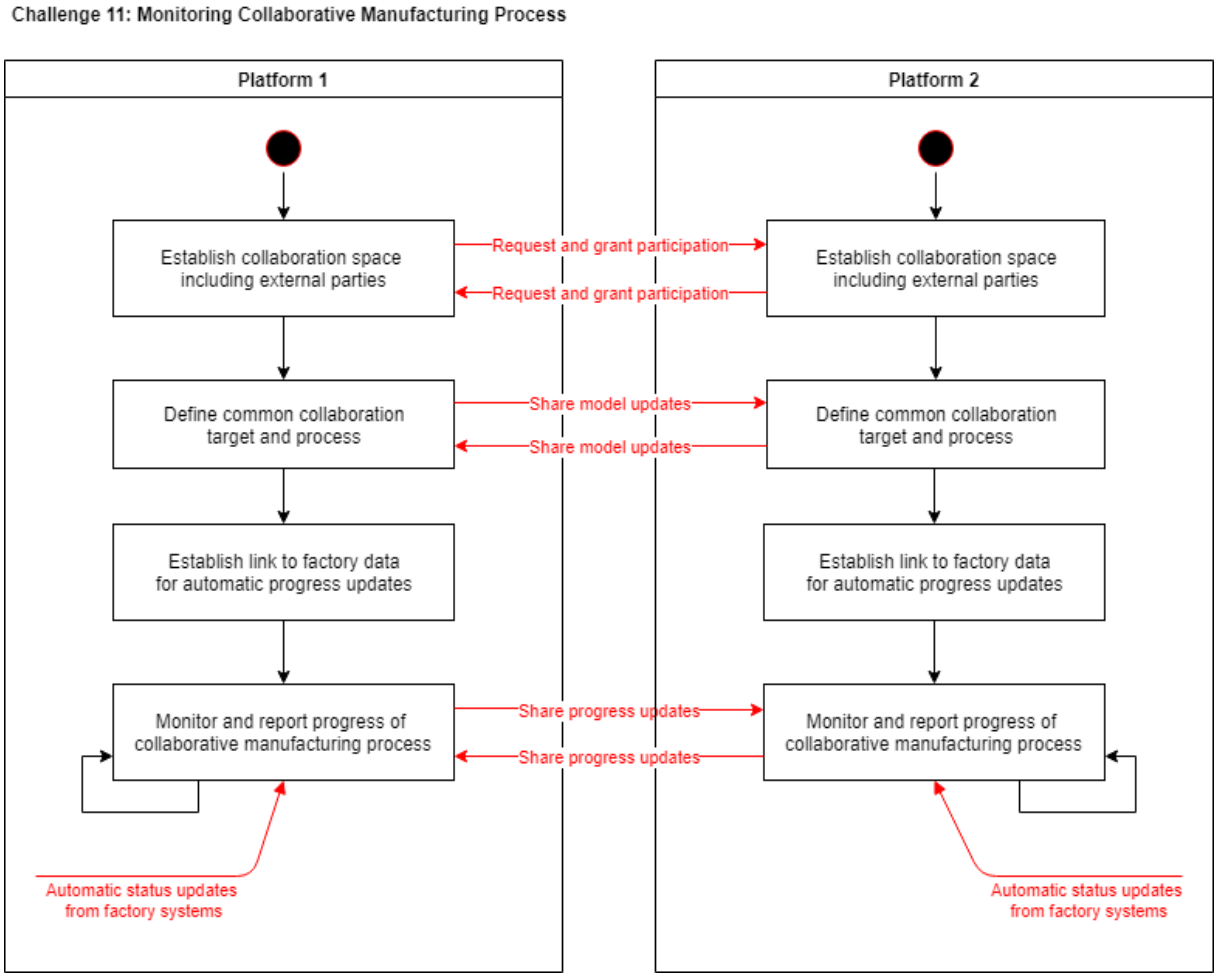


Figure 16: Monitoring collaborative manufacturing process

4.2.4.2 Requirements to support the challenge

Table 11.1. Interoperation Requirements, Challenge 11

Challenge Steps	Interoperation Requirements
Challenge-11.1	Partners from PLATFORM -1 and PLATFORM -2 initiate a collaborative environment that allows them to design/model and monitor a manufacturing process
Challenge-11.2	They enter the collaborative manufacturing process design mode on the collaborative manufacturing environment on PLATFORM -1 and PLATFORM -2 respectively. Design changes made by partners on PLATFORM -2 are propagated to the model on PLATFORM -1 and vice versa.
Challenge-11.3	They specify the process and the progress updates that they are willing to provide to each other. The process model is securely saved in the cloud
Challenge-11.4	If progress updates can be obtained from digital systems in partners' factories directly, partners set up the necessary data links using the infrastructure of the platform that they use.
Challenge-11.5	During execution of the collaborative manufacturing process, the involved partners provide and obtain progress information via the respective tool on their platform. This information is synchronized with the other platform in real time.

4.2.4.3 Collaborative design of the challenge

Table 11.2. The contributions of platform's developers/owners to implement Challenge 11

Platform	Potential Contributions of the Platform Towards the Challenge
NIMBLE	Not supported in NIMBLE
COMPOSITION	Not supported in COMPOSITION
DIGICOR	Workflow and Service Automation tool can support collaborative design of processes (though not real-time for multiple users). Processes can be saved a made accessible to specific partners and roles, though this functionality needs development. Once a process has been agreed, it is saved and executed. Activities (so far) are REST calls to services exiting in any other platform. The Factory Connector framework can support two-way communication between the local factory data sources and the collaborative service (see Challenge "Cross-Platform Device Access").
vf-OS	Not supported in vf-OS

4.3 Interoperation Challenges with External Platforms

4.3.1 Challenge 12: Symphony Platform

Symphony is a commercial automation/IoT platform that has been on the market since 2007. Symphony targets luxury mega yachts and villas, offices and resorts with solutions to hotels and factory automation.
Link: [http://www.nextworks.it/en/products/brands/symphony]

This challenge focuses on enabling the interaction of Symphony with other tools/systems/platforms in the EFPF ecosystem. The insight and experience gained through this challenge will be helpful in establishing the interlinking of other 3rd party platforms in the EFPF ecosystem.

A. Challenge description

C12	Steps	Business Function	Suitable Platform/ Tool
C12a	Enable access to shop-floor data exposed by Symphony	Shop floor device connector	Symphony, EFPF Data Spine
C12b	Compose Symphony services with third-party services	Composable services	Symphony, EFPF Data Spine, EFPF compliant / connected services
C12c	Enable access to Symphony services for an already EFPF-enabled customer	Multi-provider service access	Symphony, EFPF Data Spine, third-party EFPF compliant platform
C12d	Enable EFPF shop-floor connectors / drivers to be used in Symphony installations	• On-premise Data Spine • Composable shop-floor connectors / drivers	Symphony, EFPF Data Spine, third-party shop-floor connectors / drivers

This is a list of possible challenges for showcasing interoperations between Symphony platform and other EFPF-compliant services / tools. The implementation of a subset of such challenges is under evaluation.

- **Scenario 1:** Symphony is deployed in the customer's premises and provides both building management functions (e.g. energy, lighting, HVAC, CCTV, access control) and shop floor interconnectivity. An EFPF connector / gateway exposes data from the local shop floor to the EFPF Data Spine. Local services (e.g. data analytics, dashboards) and end-user applications directly interface with Symphony, whereas external, third-party services can access shop floor information and perform actions through the Data Spine. If the BMS information model is mapped to EFPF, too, external services might also access building-related functions in addition to the factory-related ones
- **Scenario 2**: Symphony BMS acts as a client of the EFPF Data Spine to access services provided by third parties (e.g. data analytics, feature extraction, prediction models) and provides to the customer a new service composed of the EFPF-provided ones
- **Scenario 3:** A third-party system integrator who is already selling products and services to an EFPF-enabled customer wants to offer a new service, which is provided by a Cloud-based instance of Symphony BMS. The integrator is able to connect the new service through the Data Spine, which takes care of data format and interface interoperability. For example, Symphony provides Cloud-based multi-tenant multi-site dashboard and reporting interfaces. A single customer with multiple factories would be able to see information from all of his plants on a single Symphony dashboard, provided that each plant has an EFPF compatible

gateway sending data to the Data Spine. Without EFPF, this would be possible only if Symphony were installed in each plant. This scenario is symmetrical with Scenario 2, with Symphony becoming a “third party service” and any other EFPF gateway providing access to the shop floor

- **Scenario 4:** The EFPF Data Spine is used as an interoperability layer between Symphony BMS and a third-party tool / service. This would allow an existing installation, Symphony in this case, to benefit from tools and connectors provided by the other EFPF compatible systems. For example, this might be the case for accessing existing SCADA systems for which Symphony has no driver, but an EFPF connector exists. Depending on the actual use case, though, it might be necessary to have a local deployment of the Data Spine in order to minimise latency and dependency on Cloud connectivity. Pros and cons of this hypothesis need to be carefully evaluated

B. Activity diagram

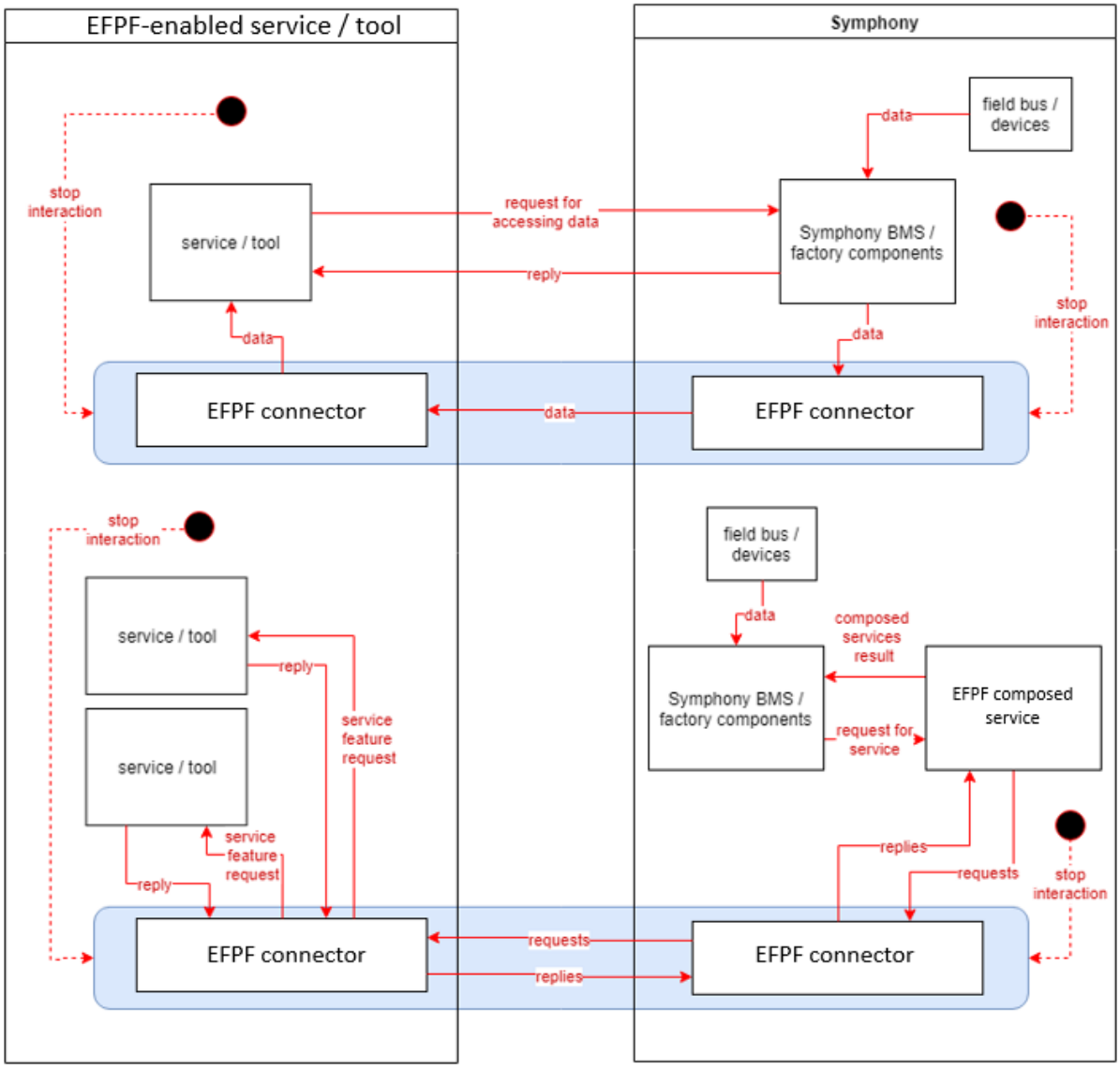


Figure 17: Symphony platform: from EFPF to Symphony

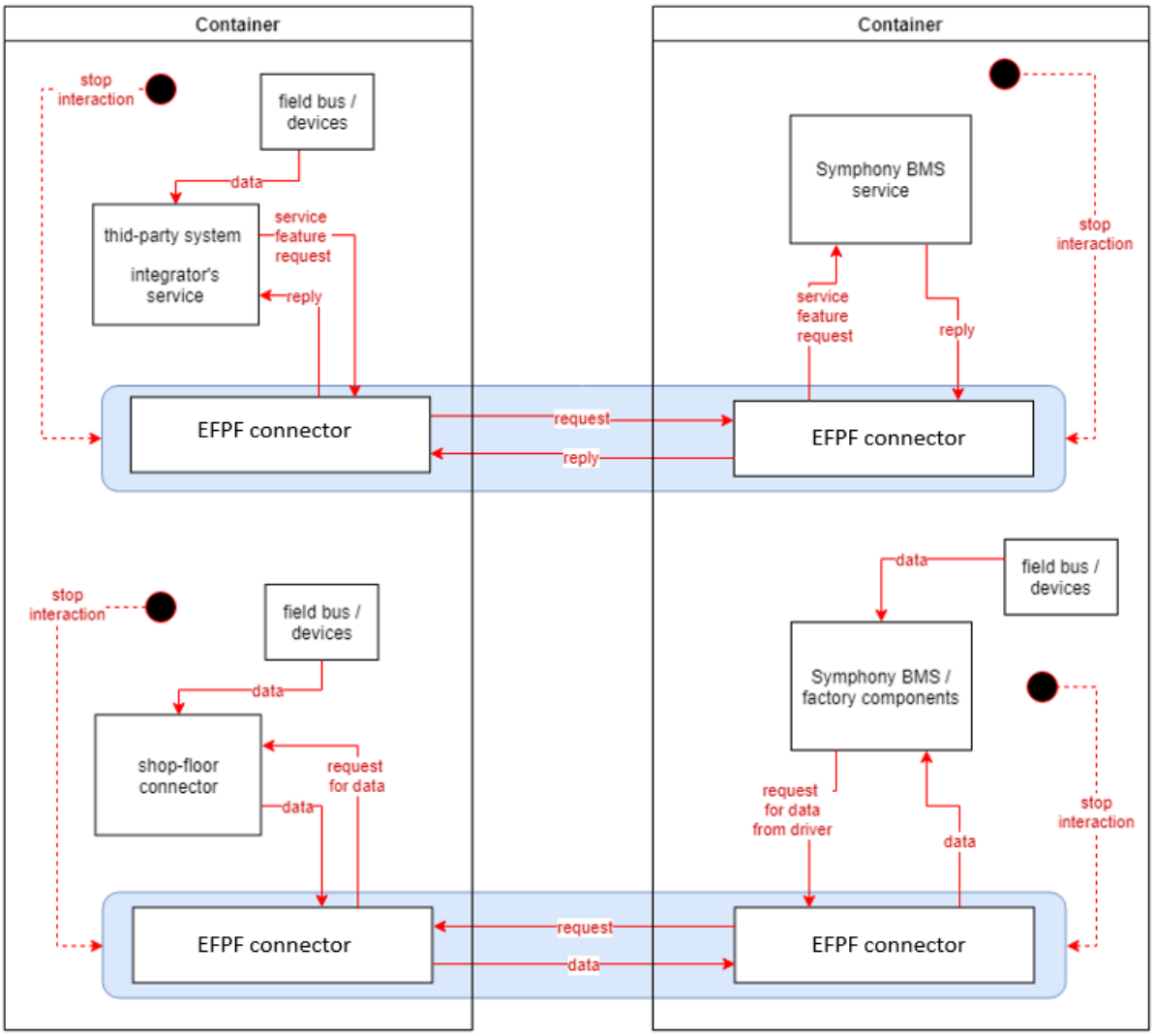


Figure 18: Symphony platform: among the containers

C. Requirements to support the challenge

A set of requirements to support Challenge 12 (based on Figure 17 and Figure 18) is presented in Table 12.1:

Table 12.1. Interoperation Requirements, Challenge 12

Challenge Steps	Interoperation Requirements
Challenge-12.1	The user of the base platform sends the request to access (i.e. monitor) specific data from Symphony platform (e.g. manufacturing shop-floor data)
Challenge-12.2	A platform is able to expose data to external services / tools through EFPF
Challenge-12.3	EFPF-enabled services / platforms have well-defined data formats and APIs
Challenge-12.4	EFPF-enabled services can be combined

5 Preparation of Interoperation Challenges for the Experimentation

The very first step to prepare for the experimentation with the initial platform interoperation challenges defined in tasks T2.2, was to select and prioritise their implementation.

Firstly, several technical consultations with the base platform's technical teams were organised to understand the specific conditions under which tools and services of a specific base platform can be used in the EFPF federation. In parallel, the definition of the initial interoperation challenges was completed and iteratively refined, as described in Section 4.

Secondly, the three challenges that has shown the strongest technology support (e.g. Challenge 1 (login), Challenge 5 (search for products & services), and Challenge 9 (cross-platform device access management) are identified as the most feasible to implement during the lifetime of the T2.2, as illustrated in Figure 2. The implementation of other challenge will be pursued during the mainstream technical developments in the EFPF project.

Thirdly, the availability of the base platforms' tools and services to provide the technology support to task T2.2 had to be declared by the technical teams.

The next step was to define the roles and responsibilities of the technical teams involved in implementation of each challenge. For example:

- **Challenge 1: Login:** SRFG defined the best suited workflow for user identification and sets up the EFPF federated user management. An EFPF Keycloak server was setup as a central Identity Provider (IDP) by SRFG, to be accessible through the EFPF portal. Technical teams from SRFG, ICE, CNET and FIT collaborated to define the direct way of establishing SSO across multiple platforms using the EFPF IDP
- **Challenge 5: Search for products & services:** C2K, CNET, ICE, and ASC performed Challenge 5 between COMPOSITION, vf-OS, NIMBLE, and DIGICOR/ SMECluster platforms
- **Challenge 9: Cross-platform device access management:** NXW and C2K performed Challenge 9 between COMPOSITION and DIGICOR/SMECluster platforms

One of the pre-requisites for the experimentation with each of the four base platforms was to create their separated EFPF platform instances with Keycloak providing the identity and access management functionalities. In this respect, the EFPF specific instances of vf-OS, NIMBLE and COMPOSITION platforms were setup during the duration of this task (T2.2). For DIGICOR, the SMECluster instance was tuned to participate in the Challenge 5 and Challenge 9.

An example of the EFPF instance of the NIMBLE platform is shown in Annex B.

6 Implementation of Challenges – Lessons Learned

6.1 Challenge 1 Implementation: Login

Partners involved in Challenge 1: SRFG, FIT, ICE, CNET

Challenge 1 turned to be not an easy one to be performed in a short span of T2.2 time. It required the following actions:

- To analyse the user identification tools, APIs and protocols of the four base platforms
- To discuss with the technical teams of the four base platforms, the required enhancements in the existing identity and access management solutions that would allow the base platforms to support user identification across multiple platforms
- To explore possible technology solutions and alternative approaches
- To experiment with difference solutions and analyse whether it is feasible to achieve the challenge in the short time span of T2.2
- To select the feasible solution that should not be abandoned once T2.2 is over and the EFPF Data Spine middleware is available in the project

The platform interoperation Challenge 1 focuses on the support for the identification of users across multiple platforms. In April 2019, during the EFPF plenary meeting, the partners decided to create the EFPF portal with its own Identity Provider (IDP) that can enable the user federation and SSO. All four base platforms in EFPF should comply with a common standard to achieve such federation. The partners decided to use the standard Oauth/OIDC protocol to implement the federation, because all of the four base platforms are already utilising this protocol. In addition, the following approach was agreed:

- All EFPF users should register on the EFPF portal (through EFPF IDP) in order to access the federated ecosystem
- All base platforms should add the EFPF IDP as a trusted IDP (in their base platform's IDPs)

The technical teams involved in the Challenge 1 implemented the following workflows:

- **Workflow 1:** In the first workflow, the user goes to the base platform and tries to login with his/her EFPF credentials. This workflow facilitates the user federation from EFPF to the base platforms
- **Workflow 2:** In the second workflow, the user needs to login to the EFPF portal first, and then to a base platform within the same browser session. When the user goes to another base platform, he/she can opt to automatically login with existing EFPF credentials.

6.1.1 Workflow 1 Implementation: User Federation

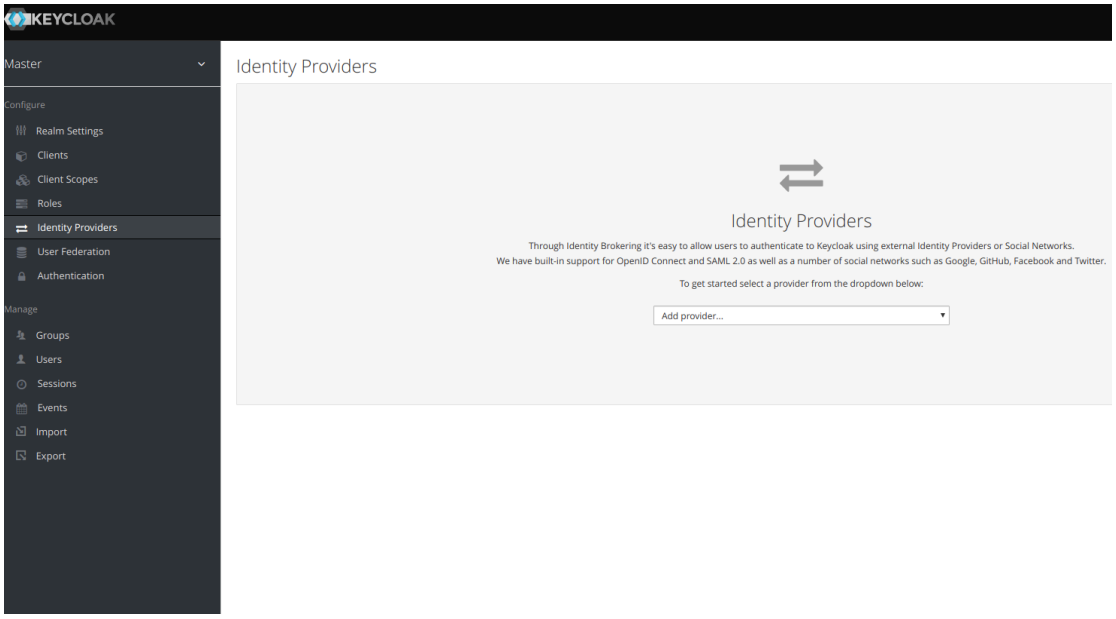
After consultations with partners, the technical teams in Challenge 1 decided to proceed further with the Workflow 1 through the following steps:

- User goes to the base platform (e.g. NIMBLE), and clicks "Login to EFPF"

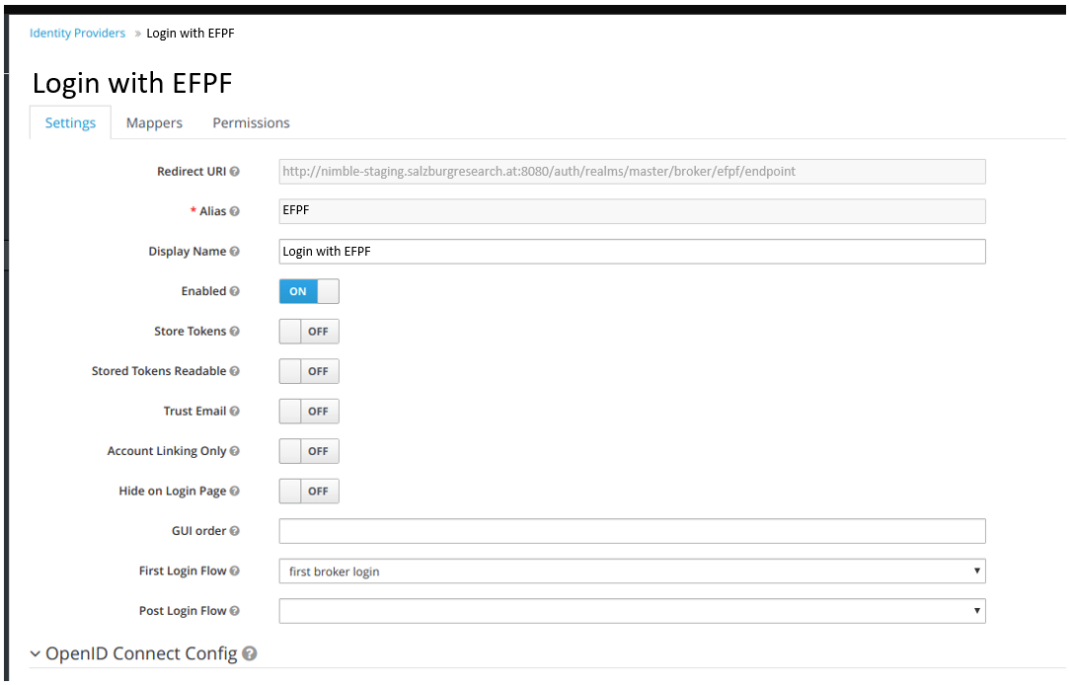
- User is automatically redirected to EFPF portal
- User logs-in using EFPF credentials
- Upon successful login message, user is redirected to NIMBLE IDP
- NIMBLE IDP creates a linked user and assigns relevant roles and permissions to the user

In this workflow, the EFPF partners involved in the Challenge 1 implementation phase needed to setup an EFPF Keycloak server as the trusted IDP within their base platforms. Further steps are described below.

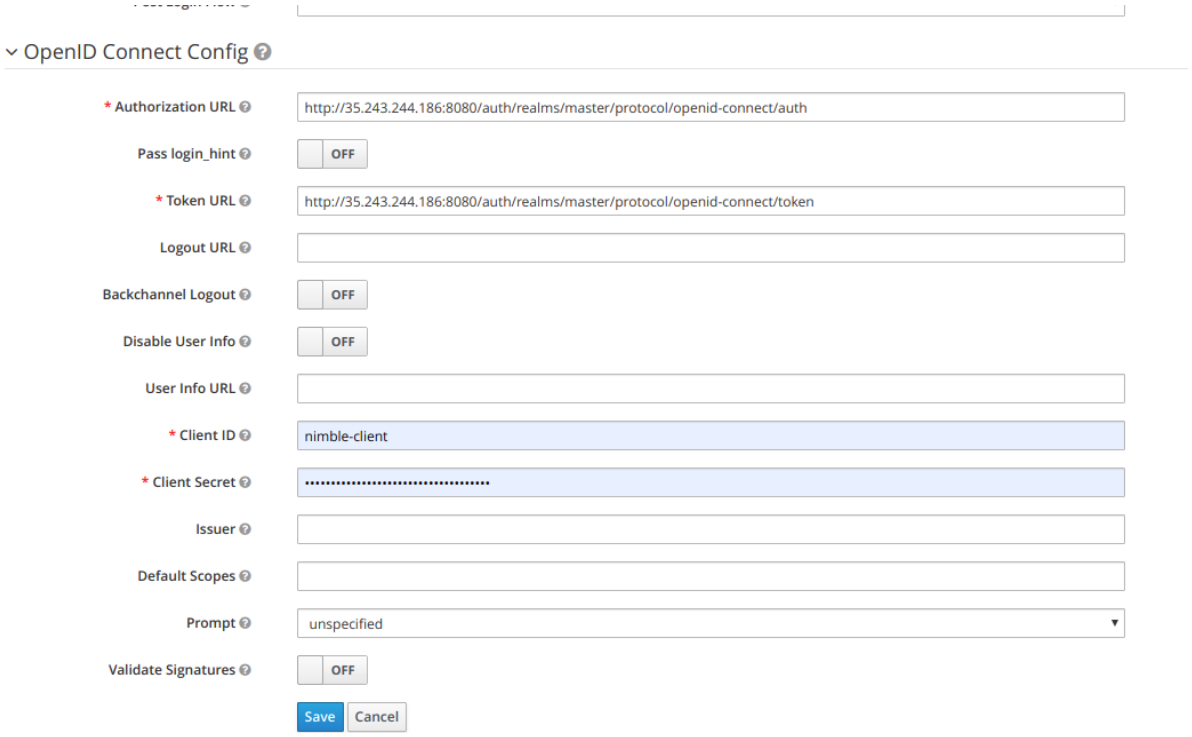
Step 1: Login to the base platform's (KeyCloak) IDP and create a Trusted IDP



Step 2: Provide Display name alias

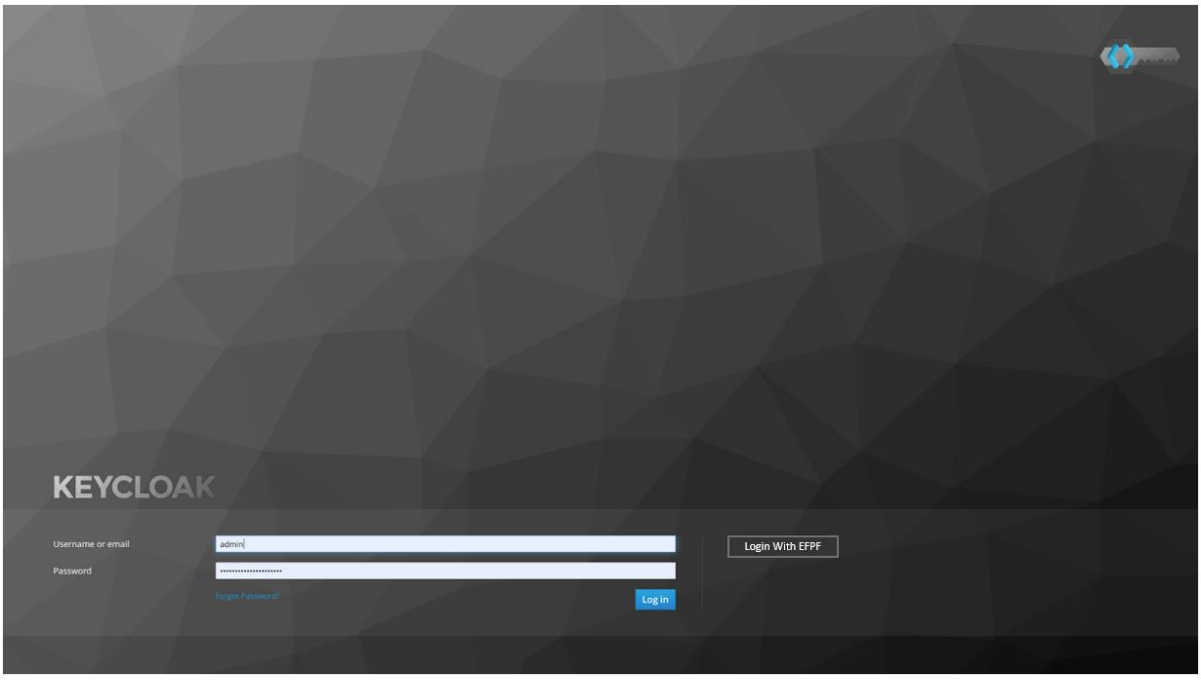


Step 3: Enable the client



Step 4: Link with EFPF IDP

In this step the technical teams from the base platforms needed to setup EFPF IDP related information in their local KeyCloak. This included the copy of client ID and secret to the relevant fields (EFPF Trusted Client). After setting up the trusted IDP, the Keycloak login page showed the login with EFPF option in the base platform.

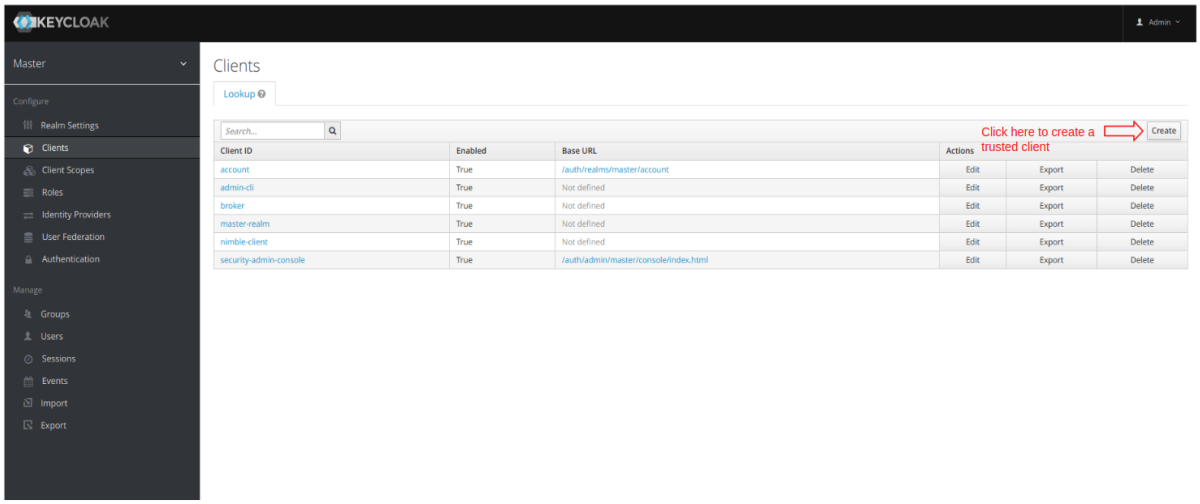


6.1.2 Workflow 2 Implementation: SSO + User Federation

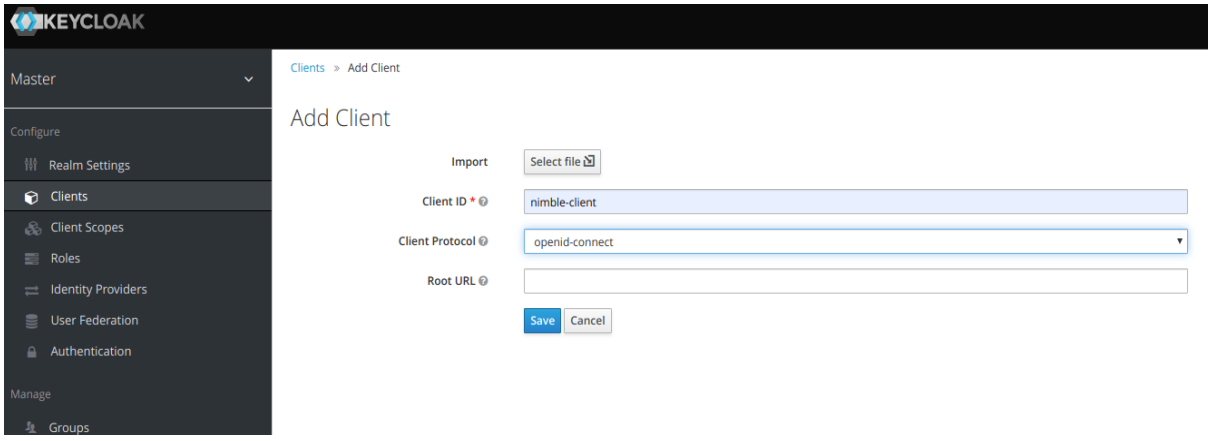
- User goes to the EFPF portal and logs-in with EFPF credentials
- User then goes to a base platform (e.g. NIMBLE) and clicks “Login to EFPF”
- User is redirected to EFPF IDP and is immediately sent back to the base platform’s IDP (because the user is already logged-in to EFPF, and a local session is kept in the EFPF IDP)
- Account linking and relevant details are done by the base platform’s IDP

The steps to setup a trusted client in the EFPF portal, include the following:

Step 1: Create a Trusted Client in the EFPF portal

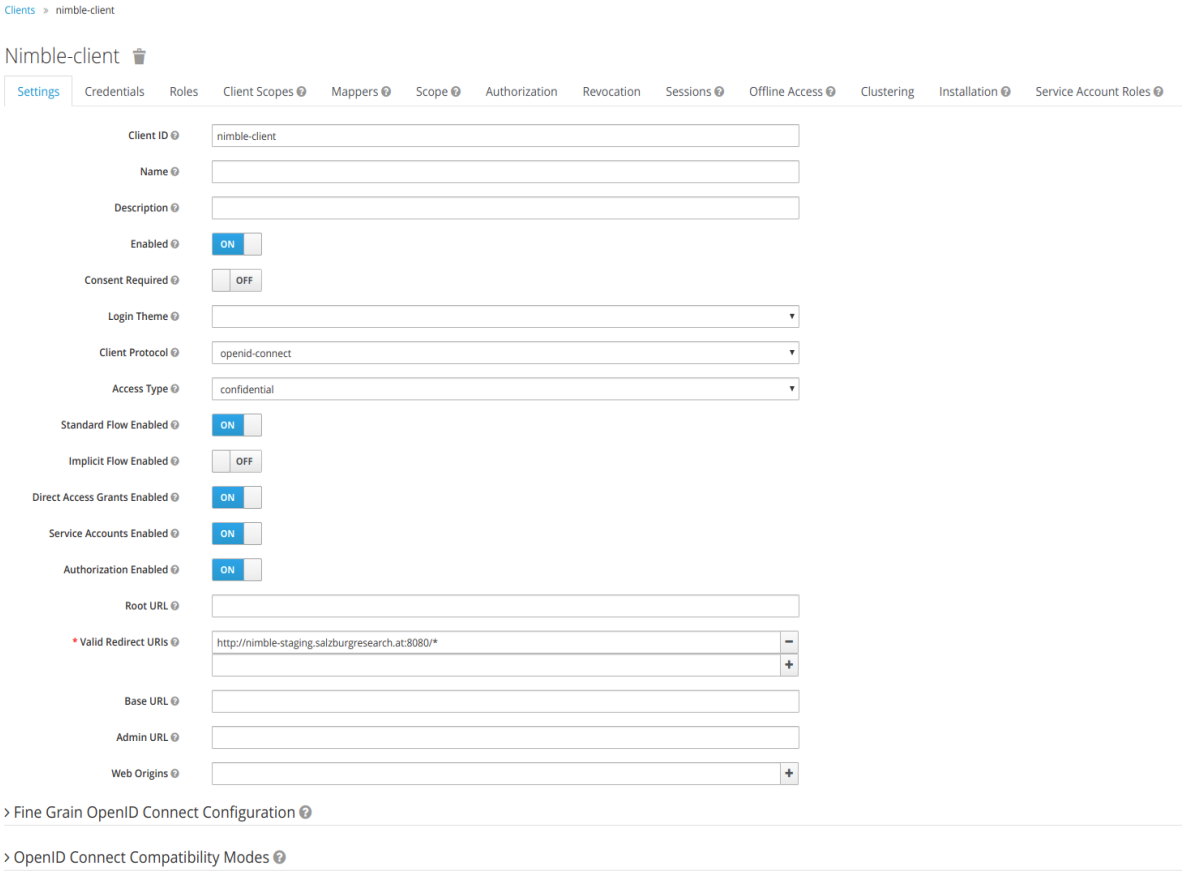


Step 2: Create a client for the relevant base platform and select the openIDConnect as the client protocol.



Step 3: Provide a valid redirect URL to redirect the user, after a successful login.

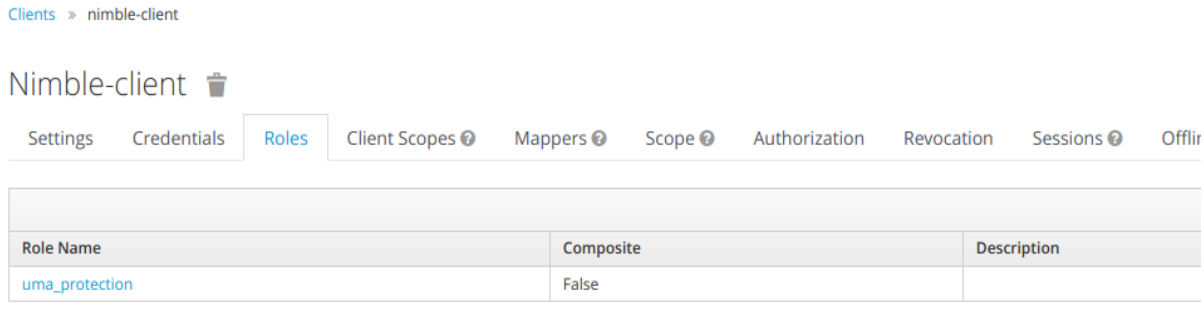
In NIMBLE, the URL of the Keycloak IDP is provided here: http://nimble-staging.salzburgresearch.at:8080/*



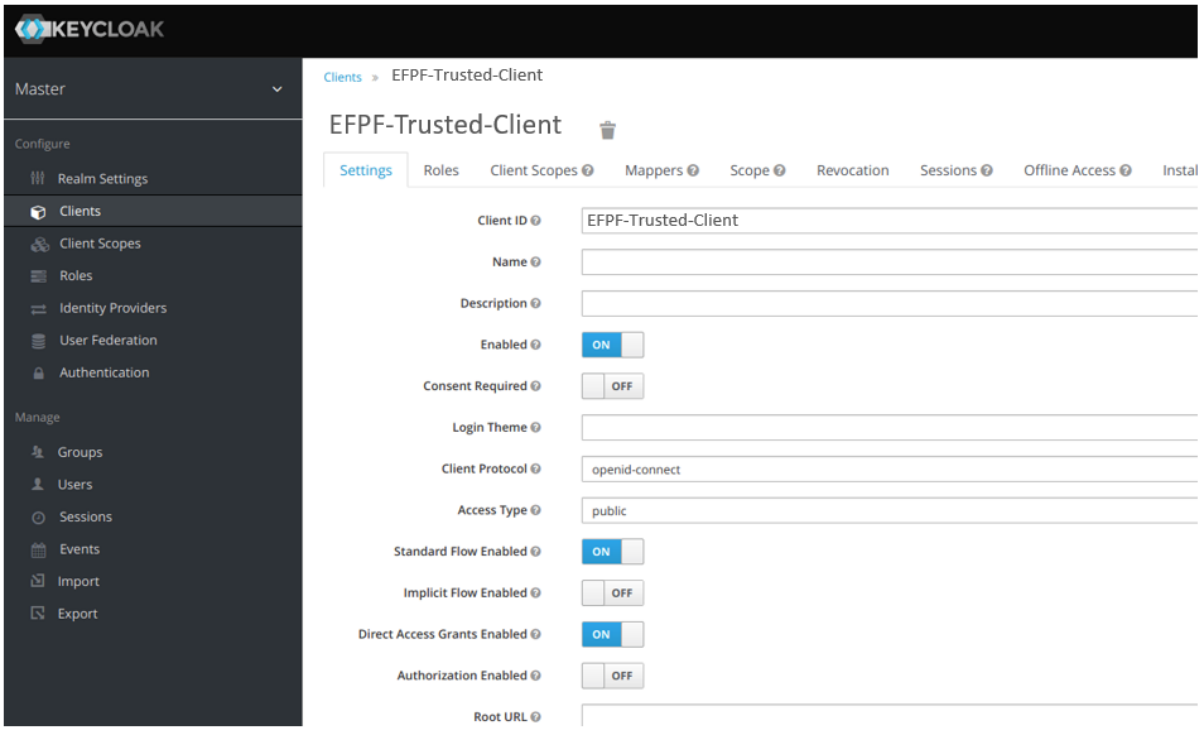
Step 4: Note down the client ID and the secret to be provided to the base platform as a trusted IDP.

Step 5 (Optional)

Based on the security requirements the admin can assign roles and scopes to the client.



An example of the EFPF trusted client is provided below.



6.2 Challenge 5 Implementation: Searching for Products and Services

Partners involved in challenge 5: CNET, SRFG, C2K, ASC

The implementation of Challenge 5 was performed between COMPOSITION (partner CNET) and other two platforms: NIMBLE (partner SRFG) and DIGICOR/SMECluster (partner C2K).

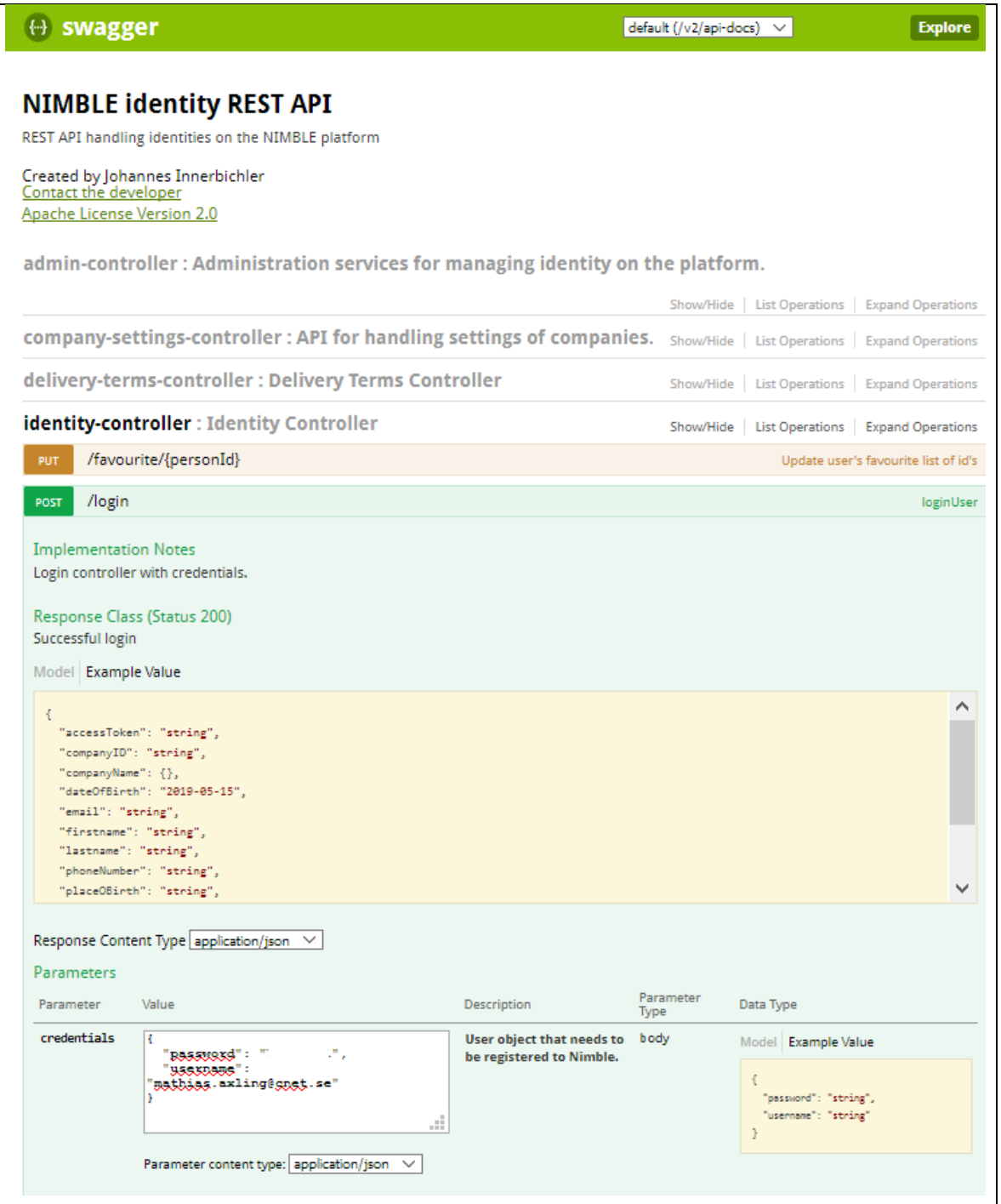
6.2.1 From COMPOSITION to NIMBLE

Note, the technical details on how the platforms involved in the implementation of the Challenge 5 are set up for this experimentation, are provided in Annex D.
The detailed descriptions on all steps required to perform the Challenge 5 are also in Annex D.

Problems Faced:

P1: In many catalogues, there is no “list all” or query with wildcard, which made it necessary to have a priori knowledge of e.g. catalogue IDs.
P2: The index controller search function was in development and it was not obvious how to query using the categories that were found (e.g. http://www.aidimme.es/FurnitureSectorOntology.owl#Door)
P3: In the select method in indexing service, the parameters are a bit cryptic. It was supposed that it requires quite a bit of knowledge about the ontology to use (However, the technical team expects search and select to be quite powerful once understood).

Results Achieved:



- Calling the login method at (https://efpf-nimble.salzburgresearch.at/api/identity/swagger-ui.html#!/party-controller/getAllPartiesUsingGET) from Swagger API, with the result:


```
{
  "username": "*****@cnet.se",
  "firstname": "Mathias",
  "lastname": "*****",
  "email": "*****@cnet.se",
  "userID": 1207,
  "companyID": "1251",
  "companyName": {
    "en": "CNet Svenska AB"
  },
  "accessToken": "eyJhbGciOiJ…HClfXxZVPwuDQyZecg",
  "showWelcomeInfo": true
```

}

- The supplied global Authentication token was used
- Items were retrieved using both the http://nimble-staging.salzburgresearch.at and http://efpf-nimble.salzburgresearch.at endpoints.
- A direct search for a specific class (e.g. “TrapDoor”) in the FurnitureOntology catalogue was not possible
- The documentation provided was very helpful. The select and search methods of the new indexing service are promising but require even more documentation and examples

Lessons Learned:

- The interfaces of any service are often designed for internal use – Interfaces are not designed for exploration and are not self-explanatory (this will likely be true for all base platforms). They require some previous knowledge of the system data model, ids and query terms. Complete REST resource collections with filters, and the possibility to retrieve all results (paged) are useful when exploring a service.
- If the underlying system is an Ontology, a SPARQL interface would make integrating platforms easier by providing exploration and more flexible queries. (Of course, this has some downsides compared to a controlled interface.)
- The Swagger UI was used for testing approaches. This is preferred over building a Postman collection as the user is not expected to be sure of the correct sequence of calls.
- Swiftly provided help and documentation by NIMBLE programmers was very useful.

*The NIMBLE Marketplace search functionality was also tried out by partner ASC through the Postman application, which still provided valuable insight into the required API calls, data exchange and interaction capabilities, but due to similar results (as reported above) it has been agreed to not to integrate into this deliverable

6.2.2 From COMPOSITION to DIGICOR/ SMECluster

Note, the technical details on how the platforms involved in the implementation of the Challenge 5 are set up for this experimentation, are provided in Annex D.
The detailed descriptions on all steps required to perform the Challenge 5 are also in Annex D.

Problems Faced:

P1: The technical team was not sure which fields the “searchTerm” parameter referenced. It seemed that it filters based on “Name”.

Results Achieved:

The technical team managed to call all services from the DIGICOR/SMECluster marketplace without any issues and retrieved all products as well as products by keyword on Name

Lessons Learned:

- The instructions provided in the document worked well.
- To perform cross-platform searches, the categories of vf-OS, product class ontologies of COMPOSITION and NIMBLE and the string-based search of SMECluster will have to be bridged somehow. This is a task for the WP3 & T3.5 to investigate
- The format of the search results (in any cross-platform call) will also have to be transformed to be usable from a different platform. This is for WP3 & T3.5 to investigate

6.3 Challenge 9 Implementation: Cross-Platform Device Access Management

Partners involved in Challenge 9: NXW, C2K

This challenge describes scenarios where the user of the base platform sends the request to access specific devices from the other platform (and interact with them, e.g. monitor devices). The request can be sent for a group of devices, or a type of devices, e.g. manufacturing or infrastructure devices, or hardware pieces networked together as part of a communications infrastructure. After acceptance of the request, the user receives access to the selected devices for a specific time and for a specific, arranged purpose of monitoring, e.g. to monitor only the progress of manufacturing processes.

The implementation of Challenge 9 was performed between COMPOSITION (partner NXW) and DIGICOR (partner C2K) platforms. It included the following steps (Figure 19):

- NXW created an account for C2K on COMPOSITION MQTT broker
- NXW sent to C2K all the info for connecting to the broker (for details, see "COMPOSITION MQTT Broker" in Annex C)
- NXW sent to C2K details on topics to subscribe to and a format of the data to be exchanged (for details, see "COMPOSITION MQTT Broker" in Annex C)
- C2K received the data through DIGICOR/ Industreweb monitoring components.

To keep the challenge implementation practical, a step that requires SSO login (to the broker) was omitted. Note that login procedure is a target of Challenge 1.

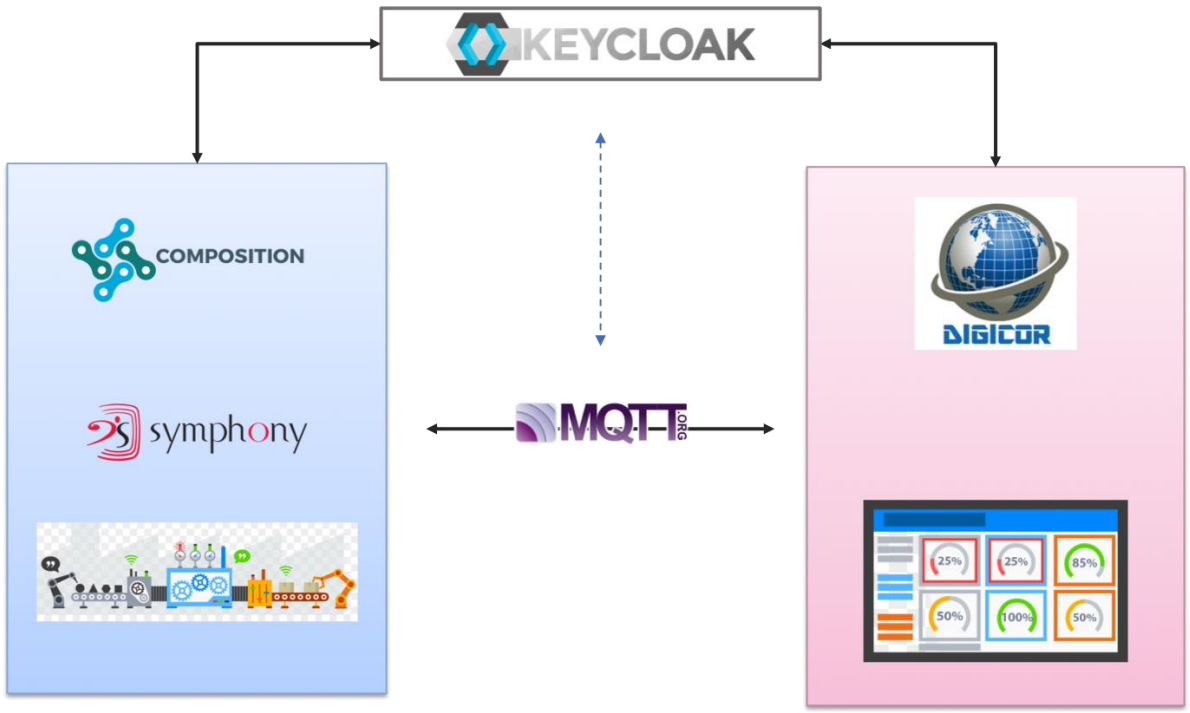


Figure 19: Challenge 9 Implementation: Cross-platform device access management between COMPOSITION and DIGICOR platforms

6.3.1 From COMPOSITION to DIGICOR

Note, the technical details on how the platforms involved in the implementation of the Challenge 9 are set up for this experimentation, are provided in Annex E.
The detailed descriptions on all steps required to perform the Challenge 9 are also in Annex E.

Observations and/ or Problems Faced:

COMPOSITION ran smoothly in publishing data to the MQTT broker

Results Achieved:

- COMPOSITION is able to publish data towards DIGICOR
- DIGICOR is able to read and render the data with one of its monitoring components

Lessons Learned:

For a complete platform-to-platform communication, it would be needed:

- A Registry / Catalogue component for providing information about resources, identities, APIs, protocols (e.g. MQTT, AMQP, HTTP), coordinates (e.g. URLs, topics), etc.
- A Search component to look for resource of interest
- A common data model or a set of supported data models with mapping
- Secure credentials to access to resources
- Shared credentials between platforms to access to resources
- Support for different communication protocols (http, mqtt, amqp, etc.)

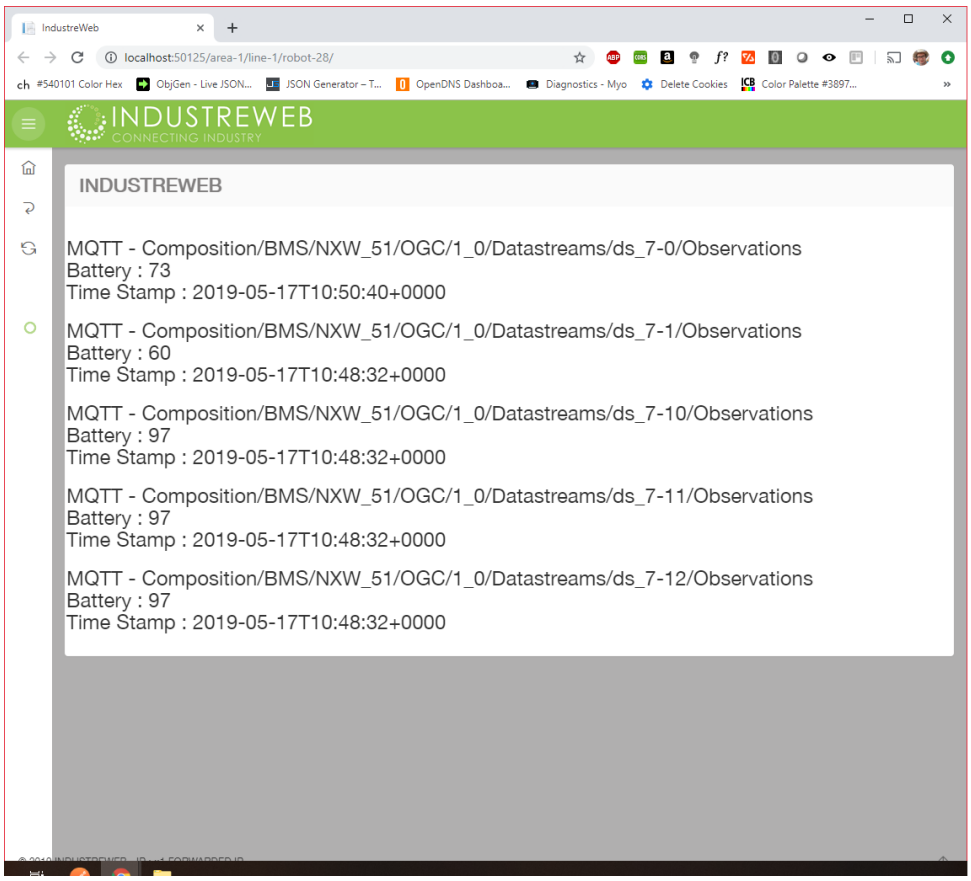


Figure 20: Challenge 9 Implementation: DIGICOR/ Industreweb receiving data streams from the COMPOSITION shop floor

6.3.2 From DIGICOR to COMPOSITION

Note, the technical details on how the platforms involved in the implementation of the Challenge 9 are set up for this experimentation, are provided in Annex E.
The detailed descriptions on all steps required to perform the Challenge 9 are also in Annex E.

Observation and/ or Problems Faced:

- The access to the COMPOSITION Raw Data Storage component is allowed only by a specific IP address. This because the access is allowed only from the RAAS COMPOSITION component, that secures all COMPOSITION REST APIs.
- COMPOSITION Raw Data Storage doesn't accept any data that is not a valid JSON
- COMPOSITION needs details about the data received, in particular
  - What is the actual data valid in the JSON message (i.e. the axis position)?
  - The ID of the data flow / sensor / axis, whereas in this case all values are sent together

Results Achieved:

- COMPOSITION is able to get data towards DIGICOR
- COMPOSITION is able to push data into its Raw Data Storage
- Data is available from COMPOSITION Raw Data Storage URL, at https://iot-hub.nextworks.it/entities/777

Lessons Learned:

For a complete platform-to-platform communication, it would be needed:

- A Registry / Catalogue component for providing information about resources, identities, APIs, protocols (e.g. MQTT, AMQP, HTTP), coordinates (e.g. URLs, topics), etc.
- A Search component to look for resource of interest
- A common data model or a set of supported data models with mapping
- Secure credentials to access to resources
- Shared credentials between platforms to access to resources
- Support for different communication protocols (http, mqtt, amqp, etc.)

7 Concluding Remarks and Next Steps

This task T2.2 practically kick-started the technical requirements gathering and implementation activities in the EFPF project. By design the task has a short life-span, and was actively running in the first 5 months of the project duration. The task required:

- **User community** in the pilots to promptly respond with an overview of their business requirements and expectations from the project, which helped immensely to definite the initial platform interoperation challenges
- **Technical teams** of all four base platforms, to setup the EFPF instances of their platforms and keep searching for solutions and patches to make their tools and services available and functional in EFPF project
- **Everyone** involved in the project, to discuss the gaps, alternatives, accept new views and possibly competing technologies, at the prospect of realising the first digital manufacturing ecosystem in Europe

Through the practical implementation of 3 selected challenges (out of 11 initially defined challenges of interest for T2.2), the technical team found some simplified solutions for crossing the borders of one platform and emerging into another one, through a set of controlled instructions provided by platform's owners. Apart from working solutions, the technical teams faced some real problems that require substantial efforts in future. Such issues opened the definition of new platform requirements (for WP2 and WP3), including security task T6.2, and helped the technical teams to decide on technologies to be used for interlinking the four base platforms, and create a space for brining even more external platforms to the EFPF federation in future.

Finally, EFPF sees this task as a way of mitigating project risks through the early explorative prototypes. Early risks identification and mitigation is of immense benefit of such a large and complex project.

As the next steps emerging out of T2.2, the task partners emphasise the following actions to be done through ongoing technical tasks:

- **The Challenge 1** allows the EFPF users to seamlessly login to other systems and platforms in the EFPF federation, by complying to open standards. Further work should be done to understand the base platform level policies needed for the user in a specific base platform, to perform the same actions in the context of the EFPF federation. The Challenge 1 addresses only the platform level interoperation (which was indeed, the objective of T2.2). To attain a full interoperation, so-called "integration level interoperation" needs to be achieved through the EFPF Data Spine. The platforms in the EFPF federation should support service accounts allowing the EFPF Data Spine to control intermediate calls between the base platform's services. Further work should be also done to implement a policy engine in the EFPF Data Spine middleware to federate service calls and to manage the access to the base platforms services. The next steps related to Challenge 1 will be continued through activities of task T6.2 "Holistic Security, Privacy and User Management Framework" (M04 – M36)
- **The Challenge 5** highlighted a topic of data formats of the search results that will need to be transformed in a standard way, in order to be usable from a different platform. For example, in order to perform cross-platform search services, the categories of vf-OS, product class ontologies of COMPOSITION

and NIMBLE, and the string-based search of DIGICOR/SMECluster will need to be bridged, in future. The next steps related to Challenge 5 will be investigated through tasks T3.5 “Data Model Interoperability Layer” and T4.2 “Data Interoperability and Analytics”

- **The Challenge 9** is performed in T2.2 without SSO login (to the broker). To create more interesting interoperability challenge, further experimentation could expose another instance of the Symphony to provide secure interaction between several platforms. Moreover, the definition and publishing of agreed data formats and standards need to be pursued in order to address the interoperability issues faced at the data exchange level. The platforms should also provide the connectors in a discoverable way, paving for the way for dynamic interactions across multiple platforms

Annex A: History

Document History	
Versions	V1.0 • Final version of the document V0.3 • The document sent out for QA V0.2 • Added Executive Summary and other content in various section V0.1 • Setup of the document structure and adding base content
Contributions	SRFG • Violeta Damjanovic-Behrendt • Nirojan Salvenathan • Dileepa Jayakody FOR • Nisrine Bnouhanna • Georg Neugschwandtner C2K • Simon Osborne NXW • Gianluca Insolvibile • Matteo Pardi • Luca Tomaselli CNET • Mathias Axling ASC • Norman Wessel FIT • Jannis Warnat ICE: • Usman Wajid • Cesar Marin Marie-Luise Lux

Annex B: EFPF Instance of the NIMBLE Platform

Figure 20 shows the entry point to the specifically created EFPF instance of the NIMBLE platform. The instance will be used and updated in the course of the project.

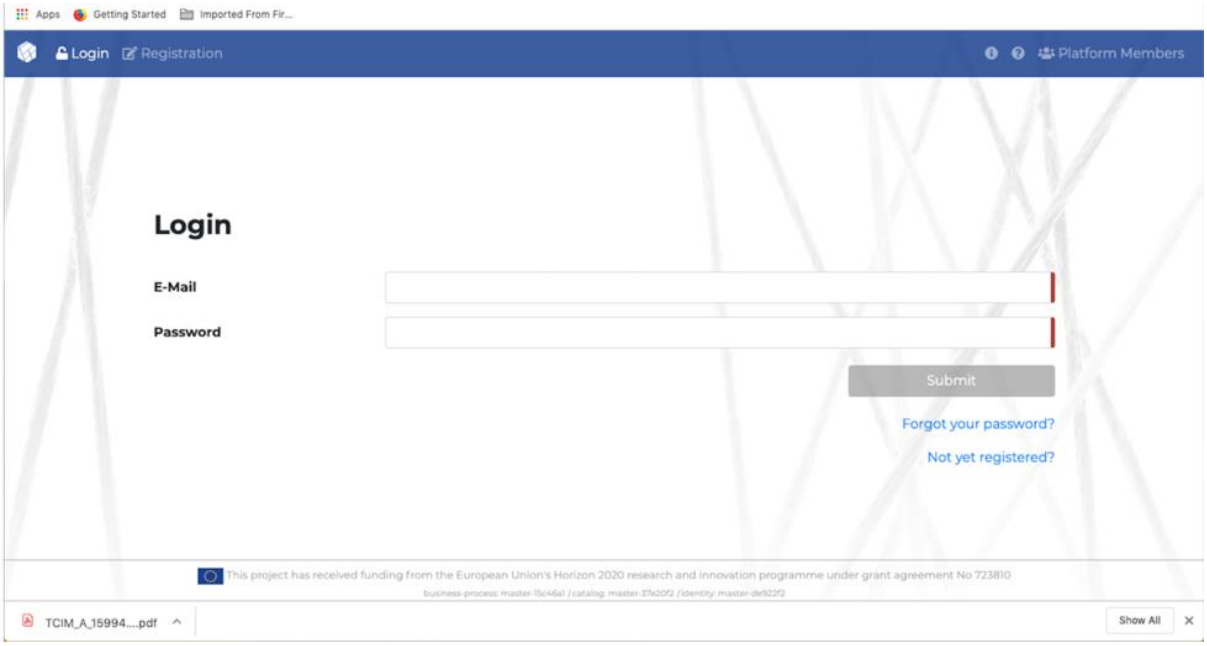


Figure 20: The EFPF instance of the NIMBLE platform

Annex C: COMPOSITION MQTT Broker

The MQTT broker URL is: inter.composition-ecosystem.eu port ****

This port is secured using Nginx with a CA-signed certificate. It should be possible for the MQTT client to connect to this port using the operating system's Root certificates.

The topic is
Composition/BMS/NXW_51/OGC/1_0/Datastreams/ds_{ID}/Observations

where ID is one of these

```
ID = ["7-0", "7-1", "7-2", "7-10", "7-11", "7-12", "7-13", "7-14", "7-15",
   "7-16", "7-17", "7-18", "7-19", "7-20", "7-21", "7-22", "7-23"]
```

This is an example of subscribing a topic using the tool “Mosquitto”:

```
mosquitto_sub -v -h inter.composition-ecosystem.eu -p 8885 -u '{user}' -P
   '{password}' -d -t 'Composition/BMS/NXW_51/OGC/1_0/Datastreams/ds_7-
   10/Observations' --cafile {path}/cacert.pem
```

{user} and {password} are the broker credentials and {path} is where you stored the cert file.
The received data is formatted as an OGC Sensor Things observation, e.g.:

```
{
    "datastream":{
        "@iot.id":"ds_7-10",
        "sensor":{
            "@iot.id":"7-10"
        }
    },
    "parameters":[
        {
            "battery":97.0
        },
        {
            "error":0.0
        }
    ],
    "phenomenonTime":"2019-04-29T12:41:36+0000",
    "result":68.0,
    "resultTime":"2019-04-29T12:41:34+0000",
    "id":"9a4f35a7-aee3-48a4-a35f-c9ea082332a3"
}
```

Where:

- @iot.id is the ID of the topic you subscribed to
- battery,error and result are the values that we received from the sensor
- phenomenonTime is the time when the sensor observed the event
- resultTime is the time when the platform gets the event (could be the same as phenomenonTime or not)

Note: in case of data coming from sensors with ID [7-0,7-1,7-2] the error field is not present.

Annex D: Challenge 5 Implementation Details

Partners involved in challenge 5: CNET, SRFG, C2K, ASC

Challenge 5 Implementation Details (from COMPOSITION to NIMBLE)

This section describes the technical details of the Challenge 5 implementation.

Local Setup:

LS1: Base URL: https://efpf-nimble.salzburgresearch.at/
LS2: Base URL: https://nimble-staging.salzburgresearch.at/
LS3: Using instructions from "Developer Tutorial on NIMBLE APIs"
LS4: Using instructions from "Search & Indexing Service – NIMBLE – SR Wiki
LS5: Using *****@cnet.se user
LS6: Swagger UI for accessing remote NIMBLE APIs and passing parameters
LS7: Using supplied general Authorization Token

Steps Carried Out to Achieve the Challenge:

S1: Step 1 : Login using Identity Service in Swagger UI https://efpf-nimble.salzburgresearch.at/.

Response:


```
{
  "username": "*****@cnet.se",
  "firstname": "Mathias",
  "lastname": "*****",
  "email": "*****@cnet.se",
  "userID": 1207,
  "companyID": "1251",
  "companyName": {
    "en": "CNet Svenska AB"
  },
  "accessToken": ".....",
  "showWelcomeInfo": true
}
```


S2: Performing "Step 2.1" Get Catalogue IDs using token and companyID from S1 in Swagger UI https://efpf-nimble.salzburgresearch.at/.

No response from server – no catalogue for this user.

S3: Calling getAllParties from party-controller using supplied general Auth token in Swagger UI https://efpf-nimble.salzburgresearch.at/.

S4: Calling getDefaultCatalogue with supplied general token to find a catalogue UUID in Swagger UI https://efpf-nimble.salzburgresearch.at/. Iterating over partyId. Found 532, catalogue uuid: "6174c25e-de91-4de0-9fce-f9eab4a7d5ce".

S5: In the response from S4, goodsItem "id": "947807e1-b56a-4d41-84f9-82648b78d8d7" was found.

S6: Calling getAvailableTaxonomyId in Swagger UI https://efpf-nimble.salzburgresearch.at/ with general Auth token to find a taxonomy for searching. Found "FurnitureOntology"

S7: Calling getRootCategories in Swagger UI https://efpf-nimble.salzburgresearch.at/ with general Auth token to find classes in taxonomy.

http://www.aidimme.es/FurnitureSectorOntology.owl#Door matches the itemClassificationCode of the product in the goodsItem above.

S8: After referring to “NIMBLE_API_Developer_Guide_V1.pdf” called:

curl -X GET "http://nimble-staging.salzburgresearch.at/index/item/select?facet.limit=15&facet.mincount=1&q=*%3A*&rows=10&start=0" -H "accept: */*"

This returned the items, but the technical team could not quite figure out how to query using the classes.

Challenge 5 Implementation Details (from COMPOSITION to DIGICOR/ SMECluster)

Local Setup:

LS1: Base URL: https://www.smecluster.com/api/CatalogueUtilsWebService/
LS2: Using instructions from “SMECluster Search API”
LS3: Calling the service

Steps Carried Out to Achieve the Challenge:

S1: Get all products and services:
curl -X GET 'https://www.smecluster.com/api/CatalogueUtilsWebService/GetAllProducts'

Response:

[
NOTE some parts of the provided code are intentionally omitted from the document
{
"Name":"Industreweb Collect",
"Description":"industreweb collect links to all your sources of data providing real-time monitoring and decision making via its high speed logic engine. monitor data, create alerts and trigger actions in other systems.",
"Price":6999.00,
"URL":"https://www.smecluster.com/tools/industreweb-collect/",
"ImageURL":"https://www.smecluster.com/media/10291/iw.jpg"
},
{
"Name":"Industreweb Vactory",
"Description":"industreweb makes the digital factory a reality using state of the art technology to produce a real-time digital twin of your production facility in virtual environment.",
"Price":3999.00,
"URL":"https://www.smecluster.com/tools/industreweb-vactory/",
"ImageURL":"https://www.smecluster.com/media/10298/iw_vr.jpg"
},
{
"Name":"DIGICOR Production Monitoring",
"Description":"",
"Price":999.00,
"URL":"https://www.smecluster.com/tools/digicor-production-monitoring/",
"ImageURL":"https://www.smecluster.com/media/10308/digicor-prod-mon.jpg"
}
]

S2: :Search for products and services
curl -X GET 'https://www.smecluster.com/api/CatalogueUtilsWebService/Search?searchTerm=global'

Response:

{
"Name":"Industreweb Global",
"Description":"a web-based private interface gets the right information to the right people, allowing lead engineers, plant managers and even over-seas company directors to get a simple overview of their responsibilities and their equipment performance. charts and graphs provide summaries and a technique to 'drill-down' into detailed information including average production, down-time, up-time, energy consumption and waste production.",
"Price":3999.00,
"URL":"https://www.smecluster.com/tools/industreweb-global/",

```
      "ImageURL":"https://www.smecluster.com/media/10291/iw.jpg"
    },
    {
      "Name":"Industreweb Global Standard",
      "Description":null,
      "Price":3999.00,
      "URL":"https://www.smecluster.com/tools/industreweb-global/industreweb-global-standard/",
      "ImageURL":""
    },
    {
      "Name":"Industreweb Global with Support",
      "Description":null,
      "Price":5999.00,
      "URL":"https://www.smecluster.com/tools/industreweb-global/industreweb-global-with-support/",
      "ImageURL":""
    }
  ]
```

Annex E: Challenge 9 Implementation Details

Partners involved in challenge 9: NXW, C2K

Challenge 9 Implementation Details (from COMPOSITION to DIGICOR/ Industreweb)

Local Setup:

LS1: Fill level sensors (17 sensors) installed in COMPOSITION pilot located at KLEEMANN
LS2: Symphony BMS collecting data from fill level sensors
LS3: MQTT Broker details:
url: inter.composition-ecosystem.eu
port *****
topics: Composition/BMS/NXW_51/OGC/1_0/Datastreams/#
LS4: Using broker account set up for enabling DIGICOR access
LS5: OGC Sensor Things data format for publishing data (as Observations)

Steps Carried Out to Achieve the Challenge:

S1: Setting up MQTT connection in COMPOSITION to publish fill-level sensor data to
Host: inter.composition-ecosystem.eu
Port: *****
Topic: Composition/BMS/NXW_51/OGC/1_0/Datastreams/ds_{ID}/Observations
{ID} = "7-0", "7-1", "7-2", "7-10", "7-11", "7-12", "7-13", "7-14", "7-15", "7-16", "7-17", "7-18", "7-19", "7-20", "7-21", "7-22", "7-23"
S2: Sharing MQTT broker coordinates at S1 with DIGICOR
S3: Sharing specific format of published data with DIGICOR (OGC Sensor Things Observation format):

```
{
   "datastream": {
      "@iot.id":"{ID}",
      "sensor": {
         "@iot.id":"{ID}"
      }
   },
   "parameters": [
      {
         "battery": {value}
      },
      {
         "error": {value}
      }
   ],
   "phenomenonTime": "yyyy-MM-ddThh:mm:ssZ",
   "result": {value},
   "resultTime": "yyyy-MM-ddThh:mm:ssZ",
   "id":"{deviceID}"
}
```

S3: Testing COMPOSITION stream via local MQTT client tool (Mosquitto):

```
mosquitto_sub -v -h inter.composition-ecosystem.eu -p *****-u '{user}' -P '{password}' -d -t
'Composition/BMS/NXW_51/OGC/1_0/Datastreams/ds_7-10/Observations' --cafile {ca_cert_file}
```

Example of data received:

```
{
   "datastream": {
      "@iot.id":"ds_7-10",
      "sensor": {
         "@iot.id":"7-10"
      }
   },
   "parameters": [
```

```
            {
               "battery":97.0
            },
            {
               "error":0.0
            }
         ],
         "phenomenonTime":"2019-04-29T12:41:36+0000",
         "result":68.0,
         "resultTime":"2019-04-29T12:41:34+0000",
         "id":"9a4f35a7-aee3-48a4-a35f-c9ea082332a3"
      }
```


Challenge 9 Implementation Details (from DIGICOR/ Industreweb to COMPOSITION)

Local Setup:

LS1: 6 Axis ABB Robot
LS2: DIGICOR collecting data from ABB robot
LS3: MQTT Broker details:
url: broker.smecluster.com
port: *****
topic: abbc2k
LS4: No authentication required from the broker
LS5: Custom data format

Steps Carried Out to Achieve the Challenge:

S1: Received MQTT broker coordinates from DIGICOR
S2: Setting up MQTT connection in COMPOSITION to get ABB robot data from
url: broker.smecluster.com
port: *****
topic: abbc2k
S3: Got example of published data from DIGICOR.


```
{
  "MessageId": "GUID",
  "MessageType": "",
  "PublisherId": "",
  "DataSetClassId": "GUID",
  "Messages": [
   {
    "DataSetWriterId": "",
    "SequenceNumber": 123,
    "MetaDataVersion": {
     "MajorVersion": 1,
     "MinorVersion": 1
    },
    "Payload": {
     "AssetId": {
      "Value": {
       "Type": 12,
       "Body": "GRU_Robot_eDO_1"
      },
      "SourceTimestamp": "yyyy-MM-ddThh:mm:ss.fffffffZ"
     },
     "AllAxes.ActualPosition.EngineeringUnits": {
      "Value": {
       "Type": 22,
       "Body": {
        "TypeId": {
         "Id": 887
        },
```

```
          "Body": {
            "namespaceUri": "http://www.opcfoundation.org/UA/units/un/cefact",
            "unitId": 17476,
            "displayName": "°",
            "description": "degree [unit of angle]"
          }
        }
      },
      "SourceTimestamp": "yyyy-MM-ddThh:mm:ss.fffffffZ"
    },
    "Axis1.ActualPosition.EURange": {
      "Value": {
        "Type": 22,
        "Body": {
          "TypeId": {
            "Id": 884
          },
          "Body": {
            "low": 0,
            "high": 359.99
          }
        }
      },
      "SourceTimestamp": " yyyy-MM-ddThh:mm:ss.fffffffZ"
    },
    "Axis1.ActualPosition": {
      "Value": {
        "Type": 11,
        "Body": 0
      },
      "SourceTimestamp": " yyyy-MM-ddThh:mm:ss.fffffffZ "
    },
```


NOTE the part of code referring to Axis2 – Axis 6 is intentionally omitted, from the document

…

DIGICOR also specified details of such data:

Each axis has a section


```
"Axis1.ActualPosition": {
  "Value": {
    "Type": 11,
    "Body": 12.3
  },
  "SourceTimestamp": "2019-03-26T18:28:54.2940194Z"
},
```


where the body tag is the actual position in degrees of that axis. There technically is no sensor id – the data comes directly from the ABB Robot controller industrial PC

S4: Tested COMPOSITION access to DIGICOR data with local MQTT client tool (Mosquitto):

mosquitto_sub -v -h broker.smecluster.com -p ***** -d -t ' abbc2k'

Example of data received:


```
{
  "MessageId": "b7956541-d471-4719-8708-ce3a79254ed5",
  "MessageType": "ua-data",
  "PublisherId": "Waterton",
  "DataSetClassId": "12780977-7397-4cb3-bde8-02d6db2ab257",
  "Messages": [
    {
      "DataSetWriterId": "1",
      "SequenceNumber": 123,
      "MetaDataVersion": {
        "MajorVersion": 1,
        "MinorVersion": 1
```

```
    },
    "Payload": {
      "AssetId": {
        "Value": {
          "Type": 12,
          "Body": "C2KRobotABB"
        },
        "SourceTimestamp": "2019-05-17T11:21:08.08Z"
      },
      "AllAxes.ActualPosition.EngineeringUnits": {
        "Value": {
          "Type": 22,
          "Body": {
            "TypeId": {
              "Id": 887
            },
            "Body": {
              "namespaceUri": "http://www.opcfoundation.org/UA/units/un/cefact",
              "unitId": 17476,
              "displayName": "°",
              "description": "degree [unit of angle]"
            }
          }
        },
        "SourceTimestamp": "2019-05-17T11:21:08.08Z"
      },
      "Axis1.ActualPosition.EURange": {
        "Value": {
          "Type": 22,
          "Body": {
            "TypeId": {
              "Id": 884
            },
            "Body": {
              "low": 0,
              "high": 359.99
            }
          }
        },
        "SourceTimestamp": "2019-05-17T11:21:08.08Z"
      },
      "Axis1.ActualPosition": {
        "Value": {
          "Type": 11,
          "Body": 99.41748
        },
        "SourceTimestamp": "2019-05-17T11:21:08.08Z"
      },
```


NOTE the part of code referring to Axis2 – Axis 6 is intentionally omitted, from the document

…

S5: Configured COMPOSITION to push received data from DIGICOR to its Raw Data Storage component. The component relies to an ElasticSearch backend.
S6: Queried COMPOSITION Raw Data Storage component to get the stored DIGICOR data.

European Factory Platform

www.efpf.org